Setup:

Hardware:

* TonySpace ESP32S3
* TS-COM-S76S(ACSip S76S)
* เสาอากาศ 900 MHz

Software:

* Libraries: TonyS\_S3.h, TonySpace\_LoRaMesh.h, RHRouter.h, RHMesh.h
* IDE: Arduino IDE
* Compiler/Platform: ESP32 environment

Procedure:

LoRa Configuration

* ความถี่ 924.0 MHz
* TX power 12
* Spreading Factor 8
* Bandwidth 125
* Preamble length 10
* Coding Rate 4/5

Node type:

* Sender
* Relay
* Receiver

**Sender Node**

ทำหน้าที่หลักในการส่งข้อมูลไปยัง Node ตัวอื่น

ข้อมูลที่ใช้ทดสอบ(example payload) คือ “Hello from Node 1!”(ส่งจาก Node 1) และ “Hello from Node 2!”(ส่งจาก Node 2) โดยก่อนจะทำการส่งไปจะมีการแปลงเป็น ASCII code แล้วเก็บไว้เป็น uint8\_t array หรือ type cast เป็น uint8\_t pointer

Initialization:

1. เรียกใช้คำสั่ง Tony.begin() เพื่อใช้งาน IO Expander
2. เริ่มใช้งานโมดูล LoRa ด้วย TonyLORA.init(SLOTX) โดย SLOTX คือช่องที่ติดตั้งโมดูล LoRa
3. เรียกใช้งานตัวจัดการจาก RHMesh ตัวจัดการนี้จะทำหน้าที่จัดการรับส่งข้อมูลในระบบ โดยจะใช้ NODE\_ADDRESS เป็นเลขประจำตัวของอุปกรณ์

Main flow:

1. Node จะพยายามส่งข้อมูลไปที่ปลายทาง (DEST\_ADDRESS) ด้วยคำสั่ง sendtowait() ทุก 60 วินาที หากส่งข้อมูลสำเร็จฟังก์ชั่นจะคืนค่า RH\_ROUTER\_ERROR\_NONE หมายความว่าไม่เจอความผิดพลาดในการส่งใดๆ แต่ถ้าทำการส่งไม่สำเร็จจะคืนค่า error แล้วลดเวลาในการรอส่งในรอบถัดไป
2. Node จะทำการจะทำการฟังข้อความที่รับได้ด้วยคำสั่ง recvfromAck() เมื่อมีข้อความเข้ามาจะทำการประมวลผล ถ้าหากเป็นข้อความที่เป็นของเราจะแสดงข้อความนั้นออกมา

**Relay Node**

ทำหน้าที่ระบบการการส่งต่อข้อมูลเป็นหลัก ไม่ได้มีการทำส่งข้อความของตัวเอง

Initialization:

1. เรียกใช้คำสั่ง Tony.begin() เพื่อใช้งาน IO Expander
2. เริ่มใช้งานโมดูล LoRa ด้วย TonyLORA.init(SLOTX) โดย SLOTX คือช่องที่ติดตั้งโมดูล LoRa
3. เรียกใช้งานตัวจัดการจาก RHMesh ตัวจัดการนี้จะทำหน้าที่จัดการรับส่งข้อมูลในระบบ โดยจะใช้ NODE\_ADDRESS เป็นเลขประจำตัวของอุปกรณ์

Main flow:

Node จะทำการจะทำการฟังข้อความที่รับได้ด้วยคำสั่ง recvfromAck() เมื่อมีข้อความเข้ามาจะทำการประมวลผล ถ้าหากเป็นข้อความที่เป็นของเราจะแสดงข้อความนั้นออกมา

**Receiver Node**

เป็นปลายทางของการส่งข้อมูล

Initialization:

1. เรียกใช้คำสั่ง Tony.begin() เพื่อใช้งาน IO Expander
2. เริ่มใช้งานโมดูล LoRa ด้วย TonyLORA.init(SLOTX) โดย SLOTX คือช่องที่ติดตั้งโมดูล LoRa
3. เรียกใช้งานตัวจัดการจาก RHMesh ตัวจัดการนี้จะทำหน้าที่จัดการรับส่งข้อมูลในระบบ โดยจะใช้ NODE\_ADDRESS เป็นเลขประจำตัวของอุปกรณ์

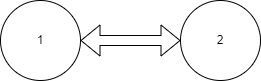
Main flow:

Node จะทำการจะทำการฟังข้อความที่รับได้ด้วยคำสั่ง recvfromAck() เมื่อมีข้อความเข้ามาจะทำการประมวลผล ถ้าหากเป็นข้อความที่เป็นของเราจะแสดงข้อความนั้นออกมาและกระพิบไฟ

**Test case #1 การรับส่งข้อมูลระหว่างอุปกรณ์ 2 ตัว**

Procedure:

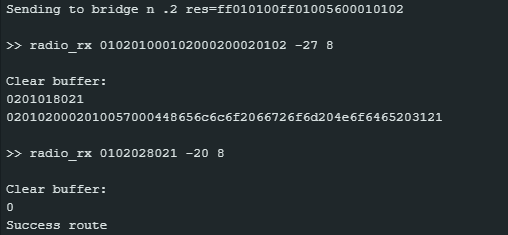
จะใช้ Sender Node 1 ตัวและ Receiver Node 1 ตัว ให้ Sender Node มี NODE\_ADDRESS = 1 และ Receiver Node มี NODE\_ADDRESS = 2 ส่งข้อมูลไปที่ตัวข้างๆ โดยไม่ต้องกระโดด



Result:

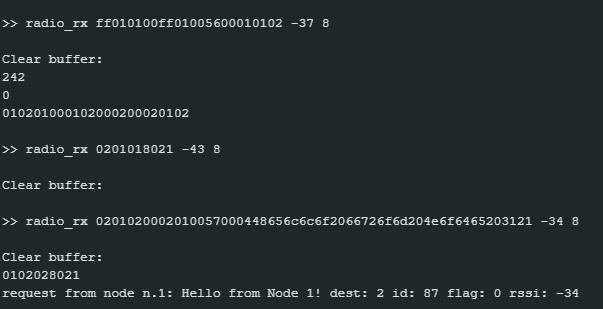
Node 1 ต้องการส่ง payload ไปยัง Node 2 แต่ยังไม่มีเส้นทางจำทำการ broadcast discovery msg : ff010100ff0100056000010102 โดยข้อความจะแบ่งเป็น 3 ส่วนคือ Header, Routing header และ payload โดย header จะมีความยาว 4 byte และ routing header มีความยาว 5 byte ส่วน payload มีขนาดตาม message type

|  |  |  |
| --- | --- | --- |
| Header | Routing header | Payload |
| ff 01 01 00 | ff 01 00 05 60 00 | 01 01 02 |



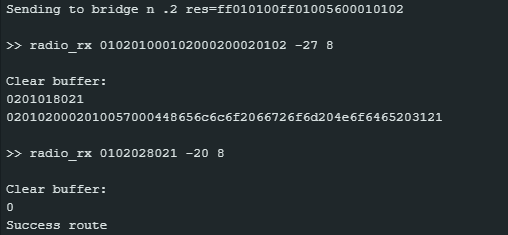
Node 2 ได้รับ discovery msg จึงตอบด้วย discover response -> 010201000102002000020102 การตอบกลับด้วย discovery response จะเปลี่ยน message type จาก 01(discovery request) เป็น 02(discovery response) แต่ข้อมูลอื่นจะคงเดิม

|  |  |  |
| --- | --- | --- |
| Header | Routing header | Payload |
| 01 02 01 00 | 01 02 00 20 00 | 02 01 02 |



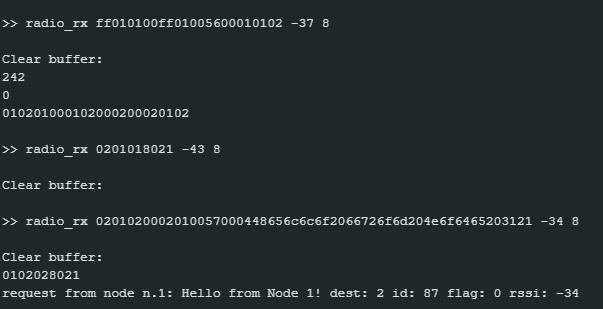
Node 1 ตอบ acknowledge Node 2 แล้วว่าได้รับเส้นทางแล้วจึงส่ง จะส่ง application message ไปยัง Node 2

|  |  |
| --- | --- |
| Header | Acknowledge |
| 02 01 01 80 | 21 |



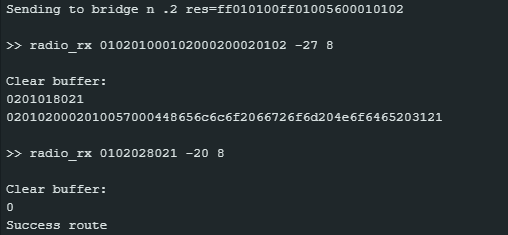
Node 2 ได้รับข้อความและตอบ Acknowledge Node 1 ว่าได้รับ payload แล้วก่อนที่จะแสดงผล payload ผ่าน serial monitor

|  |  |  |
| --- | --- | --- |
| Header | Routing header | Payload |
| 02 01 02 00 | 02 01 00 57 00 | 04 48 65 6c 6c 6f 20 66 72 6f 6d 20 4e 6f 64 65 02 31 21 |



ข้อความที่ได้รับคือ “Hello from Node 1” จุดหมายปลายทางคือ Node address 2 (ตัวรับ)

|  |  |
| --- | --- |
| Header | Acknowledge |
| 02 01 01 80 | 21 |



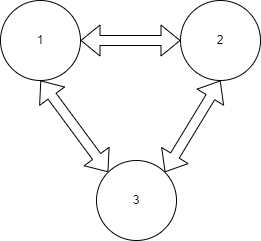
เมื่อ Node 1 ได้รับ Acknowledge จาก Node 2 ถือว่าการส่งเสร็จสมบูรณ์และฟังก์ชั่น sendToWait() จะส่ง 0 คืนกลับมา โดย 0 มีความหมายว่า RH\_ROUTER\_ERROR\_NONE

หากส่งไม่สำเร็จจะได้รับ response แบบอื่นจาก sentToWait()

* 1 = RH\_ROUTER\_ERROR\_INVALID\_LENGTH เมื่อข้อความยาวเกินกว่าที่จะส่งไปได้
* 2 = RH\_ROUTER\_ERROR\_NO\_ROUTE เมื่อหาเส้นทางไปยัง address ปลายทางไม่ได้
* 3 = RH\_ROUTER\_ERROR\_TIMEOUT เมื่อใช้เวลานานเกินไป
* 4 = RH\_ROUTER\_ERROR\_NO\_REPLY เมื่อ address ปลายทางไม่ตอบกลับ
* 5 = RH\_ROUTER\_ERROR\_UNABLE\_TO\_DELIVER เมื่อพยายามส่งไปที่ address ปลายทางที่เคยส่งได้แล้วไม่สำเร็จ

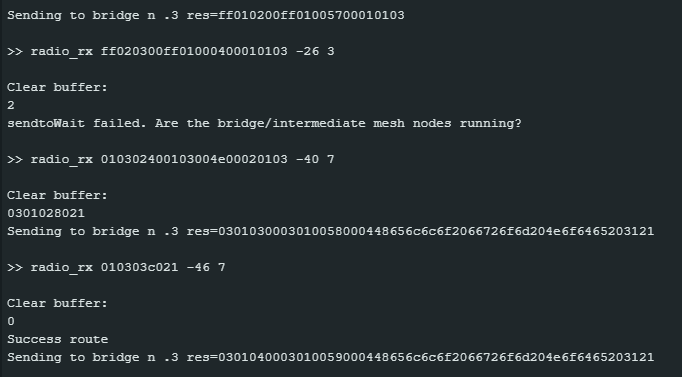
**Test case #2 การรับส่งข้อมูลระหว่างอุปกรณ์ 3 ตัว (Triangle)**

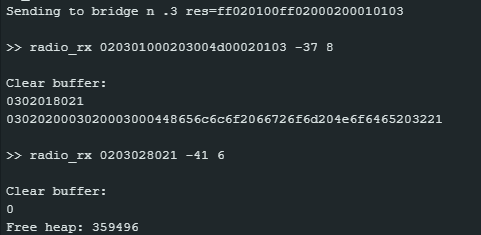
จะใช้ Sender Node 2 ตัวและ Receiver Node 1 ตัว ให้ Sender Node มี NODE\_ADDRESS = 1,2 และ Receiver Node มี NODE\_ADDRESS = 3 โดยจะให้ Sender Node ทั้ง 2 ตัวไปที่ Receiver Node ตัวเดียวกันและในเวลาเดียวกัน



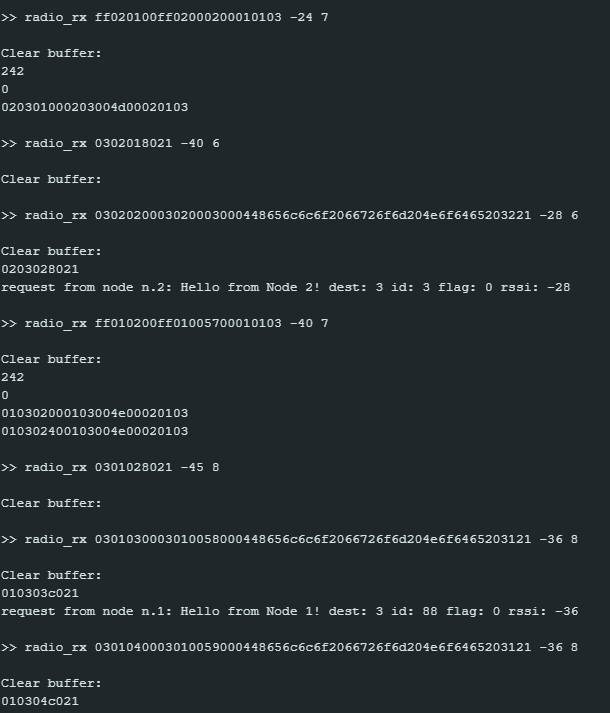
Result:

Sender Node ทั้ง 2 ต้องการส่งข้อมูลไปที่ Node 3 ทั้ง 2 ตัวจึงพยายาม broadcast discovery message ไปที่ Node 3 เพื่อหาเส้นทางส่ง

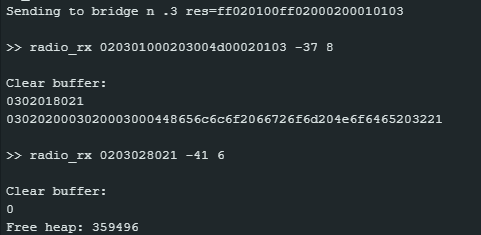




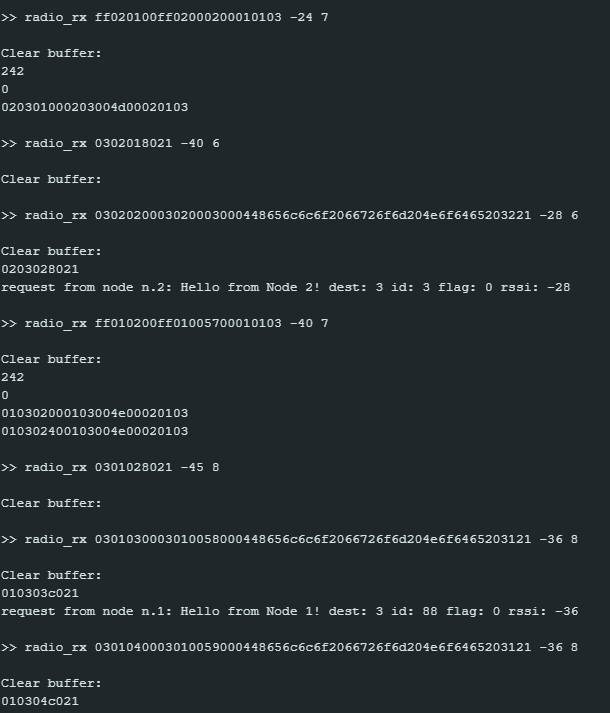
Node 3 ได้ discovery message ของ Node 2 ก่อนจึงทำการติดต่อกับ Node 2 ก่อนด้วยการตอบ discovery response ไป



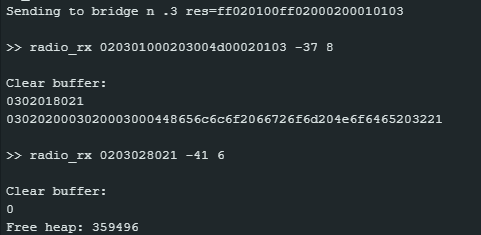
Node 2 เมื่อได้ discovery response จึงส่ง Acknowledge กลับไปที่ Node 3 ก่อนที่จะส่ง payload ไปที่ Node 3



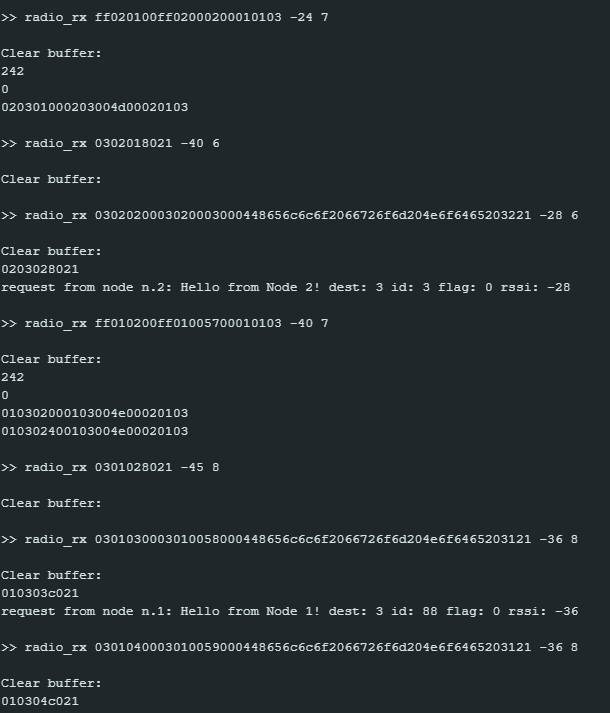
Node 3 ได้รับ Acknowledge จาก discovery response ที่ส่งไปที่ Node 2 จากนั้นก็ได้รับ payload จาก Node 2 และตอบ Node 2 ว่าได้รับ payload แล้วก่อนที่จะแสดงผล payload ผ่าน serial monitor



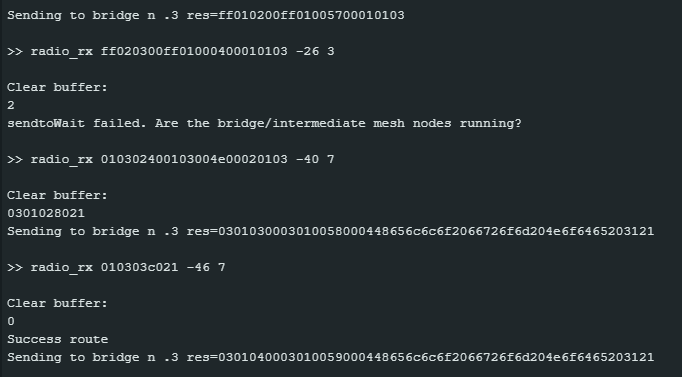
Node 2 ได้รับ Acknoledge จาก Node 3 ถือว่าจบการส่งในส่วนของ Node 2



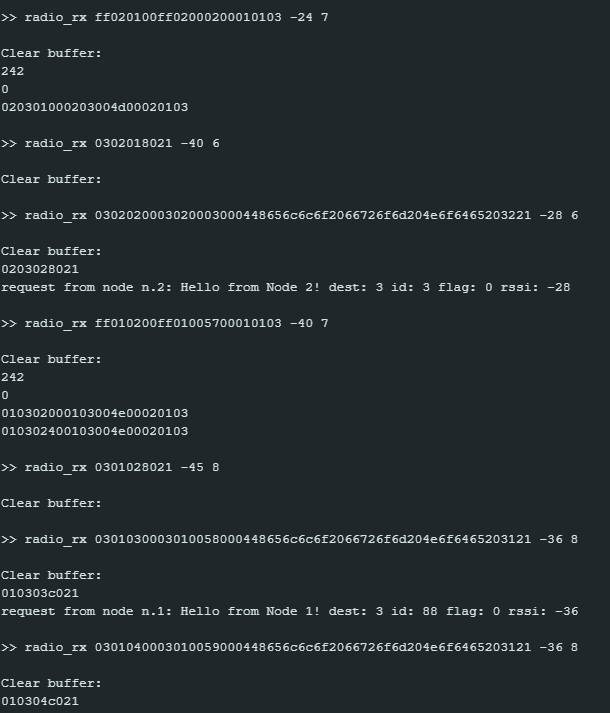
เมื่อได้รับข้อความของ Node 2 แล้วแต่ buffer ของโมดูล LoRa ยังมี discovery message ของ Node 1 อยู่จึงทำงานต่อเนื่องด้วยการตอบ discovery response กลับไปยัง Node 1



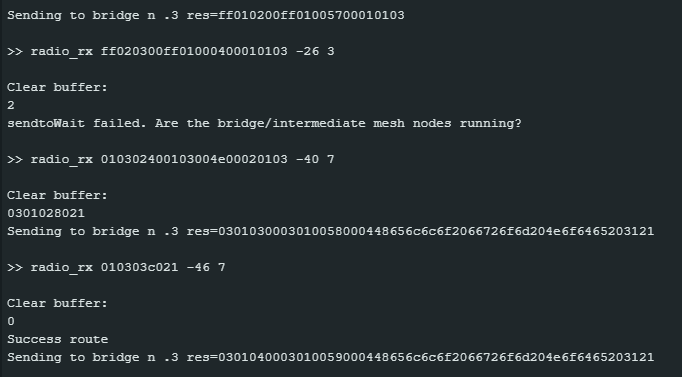
เนื่องจากการตอบ Node 2 ก่อนหน้าทำให้ discovery message ตัวก่อนหน้าไม่สามารถทำงานได้สำเร็จแต่ discovery response ที่ตอบมาช้ายังสามารถเพิ่มเส้นทางได้ ทำให้เมื่อลองส่งข้อความใหม่ไม่ต้องทำ discovery อีกรอบและสามารถส่งไปยังปลายทาง



Node 3 ได้รับข้อความที่ Node 1 ส่งมาและส่ง Acknowledge ไปยัง Node 1 ว่าไดรับข้อความแล้ว

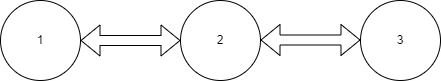


Node 1 ได้รับ Acknowledge จาก Node 3 ถือจบการส่ง



**Test case #3 การรับส่งข้อมูลระหว่างอุปกรณ์ 3 ตัว แบบจำกัดการส่ง (1-1-1)**

จะใช้ Sender Node 1 ตัว, Relay Node 1 ตัว และ Receiver Node 1 ตัว ทำการทดสอบโดยการให้ Sender Node กับ Receiver Node โดยทำให้ไม่สามารถส่งข้อความได้โดยตรง



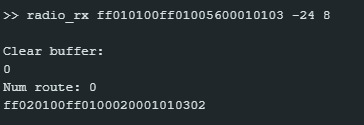
Result:

เมื่อต้องการส่งไปที่ Node 3 แต่ยังไม่มีเส้นทาง Node 1 จึงทำการ Discovery ด้วย Broadcast discovery message ตามแผนผังการติดตั้งทำให้ Node 1 ไม่ส่งไป Node 3 ได้โดยตรง จึงต้องส่งผ่าน Node 2

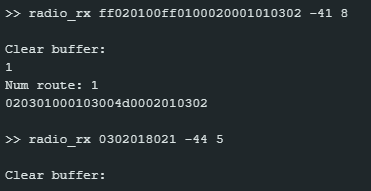


3

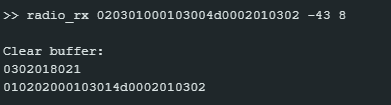
Node 2 จะได้รับ Discovery message จาก Node 1 เมื่อตรวจสอบข้อความแล้วไม่ได้มีปลายทางเป็นตัวมันเองจึงเอา address ของตัวมันไปต่อกับ payload แล้วจึง Re-broadcast discovery msg ออกไป



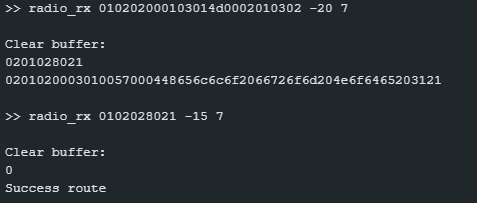
Node 3 ได้รับ Discovery message จาก Node 2 เมื่อตรวจข้อความแล้วมีปลายทางเป็น Node 3 เมื่อทราบแล้วว่า Node 1 ต้องการเส้นทางไปที่ Node 3 ด้วย Discovery message จึงตอบกลับด้วย Discovery response ที่มีเส้นทางกลับไปยัง Node 1 โดยผ่าน Node 2

****

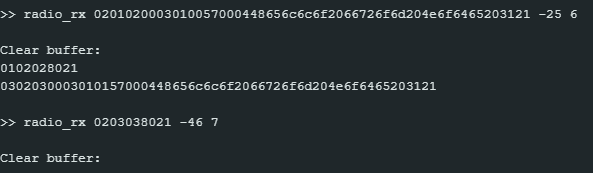
Node 2 เมื่อได้รับข้อความจึงตอบ Acknowledge กลับไปที่ Node 3 ว่าทำการส่งสำเร็จแล้ว ข้อความที่ได้มาจาก Node 3 มีปลายทางคือ Node 1 จึงส่งต่อไปโดยไม่ได้ยุ่งกับข้อความที่จะส่งต่อไป

****

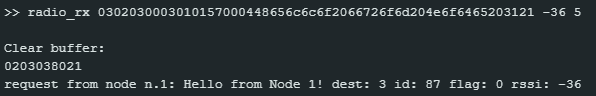
การทำ Discovery ของ Node 1 สำเร็จเมื่อได้รับเส้นทางจาก Discovery response จาก Node 3 เมื่อทราบแล้วว่าเส้นทางแล้วจึงส่งข้อความที่ต้องการไปยัง Node 3



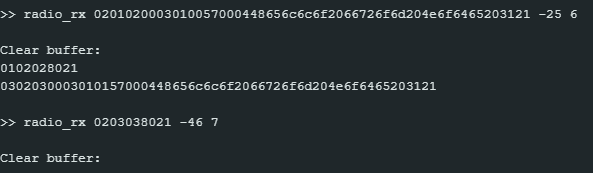
Node 2 ตอบ Acknowledge Node 1 ว่าการส่งสำเร็จและส่งข้อความนี้ต่อไปที่ Node 3



Node 3 ตอบ Acknowledge Node 2 ข้อความที่ได้มาเป็นข้อความของมันเอง

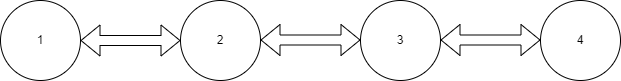


Node 2 ถือว่าเป็นการส่งจบสมบูรณ์



**Test case #4 การรับส่งข้อมูลระหว่างอุปกรณ์ 4 ตัว แบบจำกัดการส่งแบบที่ 1 (1-1-1-1)**

จะใช้ Sender Node 1 ตัว, Relay Node 2 ตัว และ Receiver Node 1 ตัว ทำการทดสอบโดยการให้ Sender Node กับ Receiver Node โดยทำให้ไม่สามารถส่งข้อความได้โดยตรง จะต้องวิ่งผ่าน relay node ทั้ง 2 ตัว

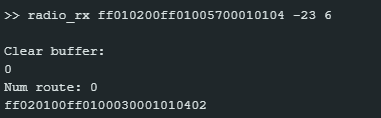


Result:

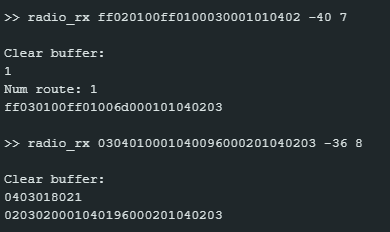
เมื่อต้องการส่งไปที่ Node 4 แต่ยังไม่มีเส้นทาง Node 1 จึงทำการ Discovery ด้วย Broadcast discovery message ตามแผนผังการติดตั้งทำให้ Node 1 ไม่ส่งไป Node 4 ได้โดยตรง จึงต้องส่งผ่าน Node 2 และ 3



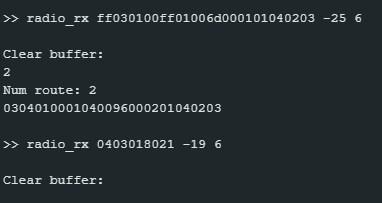
Node 2 ได้รับ broadcast discovery message จาก Node 1 แต่ตัวมันไม่ใช้ปลายทางจึงเพิ่ม address ของตัวเองไปที่ payload ส่วน route แล้วจึงทำการ Re-broadcast discovery message



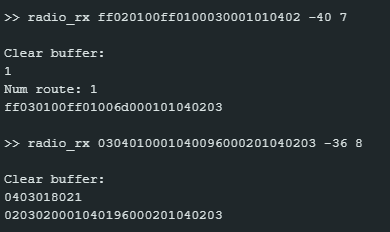
Node 3 ได้รับ re-broadcast discovery message จาก Node 2 แต่ตัวมันไม่ใช้ปลายทางจึงทำการ Re-broadcast แต่ตัวมันไม่ใช้ปลายทางจึงเพิ่ม address ของตัวเองไปที่ payload ส่วน route แล้วจึงทำการ Re-broadcast discovery message อีกที



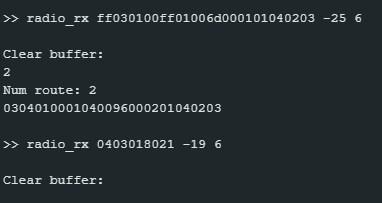
Node 4 ได้รับ re-boardcast ที่ถูกส่งผ่าน relay node 1 และ 2 (Node 2 และ 3) และตัวมันเป็นปลายทางจึงส่ง response ตอบกลับไปโดยจะส่งกลับไปตามทิศทางที่ส่งมา



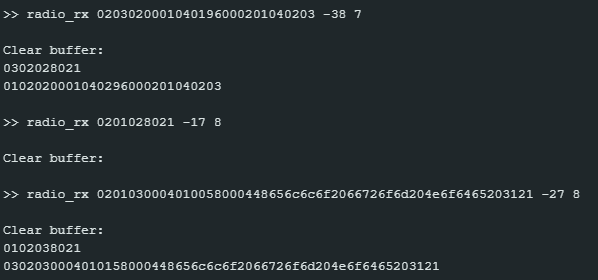
Node 3 ได้รับ discovery response จาก Node 4 และตอบ Acknowledge Node 4 ก่อนที่จะส่งต่อไปที่ Node 2



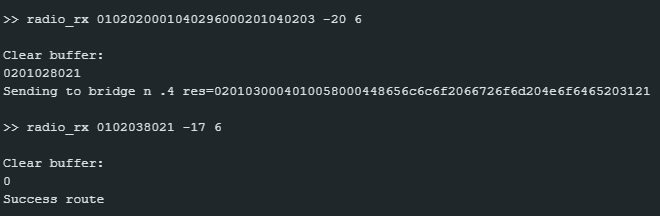
Node 4 ได้รับ Acknowledge จากการส่ง discovery response ไป Node 3



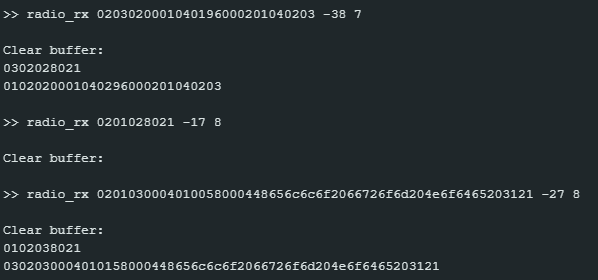
Node 3 ได้รับ discovery response จาก Node 4 และตอบ Acknowledge Node 4 ก่อนที่จะส่งต่อไปที่ Node 2



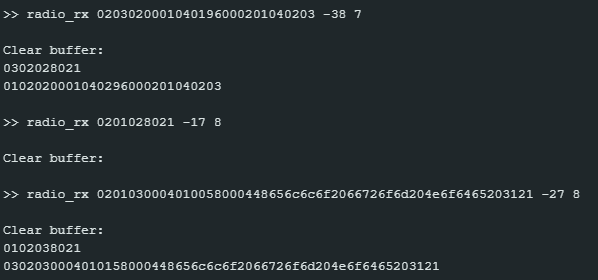
Node 1 ได้รับเส้นทางจาก discovery response และตอบ Acknowledge Node 2 ก่อนทำการทำการส่ง application message ไปทางที่ส่งผ่านไปทาง Node 2



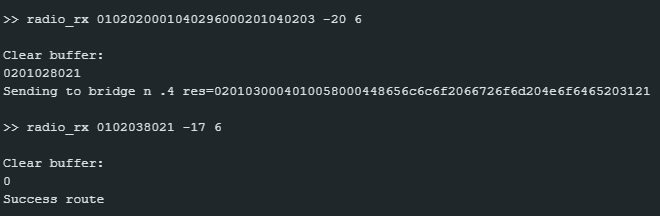
Node 2 ได้รับ Acknowledge จากการ่างต่อ discovery response



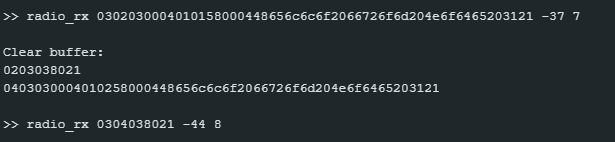
Node 2 ได้รับ application message ที่มีปลายทางเป็น Node 4 และตอบ Acknowledge Node 1 ก่อนที่จะส่งต่อไปที่ Node 3



Node 1 ได้รับ Acknowledge จากการส่ง application message ไปที่ Node 2

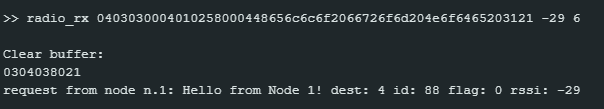


Node 3 ได้รับ application message ที่มีปลายทางเป็น Node 4 และตอบ Acknowledge Node 2 ก่อนที่จะส่งต่อไปที่ Node 4

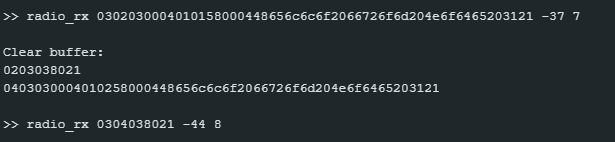


Node 2 ได้รับ Acknowledge จากการส่ง application message ไปที่ Node 3

Node 4 ได้รับ application message จาก Node 3 และตอบ Acknowledge Node 3

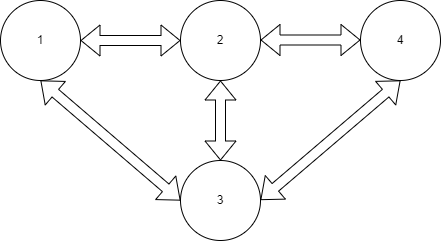


Node 3 ได้รับ Acknowledge จากการส่ง application message ไปที่ Node 4



**Test case #5 การรับส่งข้อมูลระหว่างอุปกรณ์ 4 ตัว แบบจำกัดการส่งแบบที่ 2 (1-2-1)**

(Step 1) จะใช้ Sender Node 1 ตัว, Relay Node 2 ตัว และ Receiver Node 1 ตัว ทำการทดสอบโดยการให้ Sender Node กับ Receiver Node โดยทำให้ไม่สามารถส่งข้อความได้โดยตรง จะต้องวิ่งผ่าน relay node ตัวใดตัวหนึ่ง relay node สามารถส่งต่อกันได้หากใช้เวลาส่งน้อยกว่า

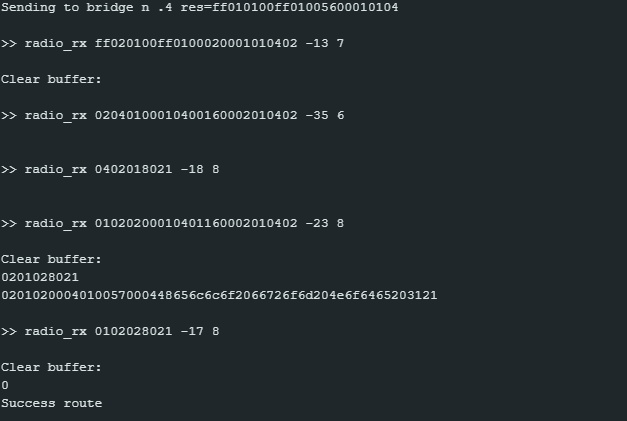


(Step 2) เมื่อทำการส่งสำเร็จแล้วจะทำการปิด relay node ที่ส่งข้อมูลระหว่าง Node 1 และ 4 เพื่อให้ Node 1 ทำ discovery ใหม่ผ่าน relay node อีกตัวที่ไม่สามารถส่งได้ในตอนแรก

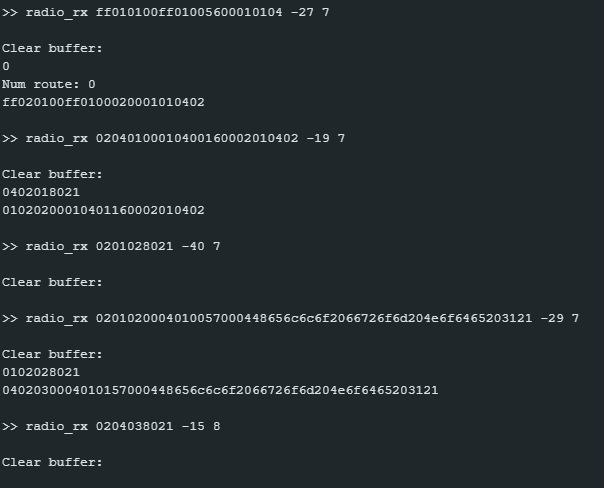
**Result:**

(Step 1)

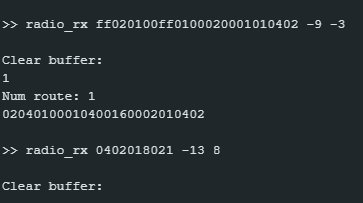
เมื่อต้องการส่งไปที่ Node 4 แต่ยังไม่มีเส้นทาง Node 1 จึงทำการ Discovery ด้วย Broadcast discovery message ตามแผนผังการติดตั้งทำให้ Node 1 ไม่ส่งไป Node 4 ได้โดยตรง ไม่ได้มีการระบุไว้ว่า Node 2 หรือ Node 3 เพราะทั้ง 2 ตัวสามารถใช้งานเป็น relay node ได้



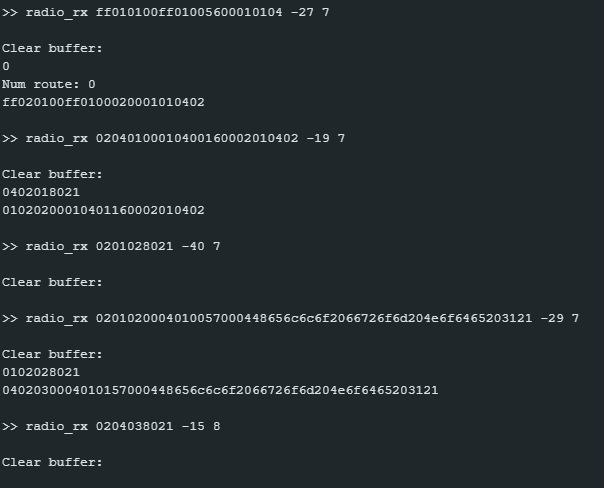
Node 2 ได้รับ broadcast discovery message จาก Node 1 แต่ address ของ Node 2 ไม่ใช่ปลายทางจึงเพิ่มเพิ่ม address ของตัวเองก่อนจะ re-broadcast discovery message



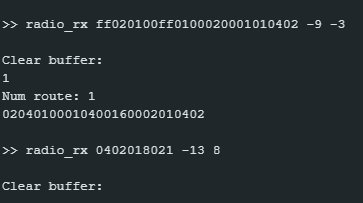
Node 4 ได้รับ broadcast discovery message จาก Node 2 และ address ปลายทางเป็น address ของ Node 4 จึงตอบกลับด้วย discovery response ไปทาง Node 2



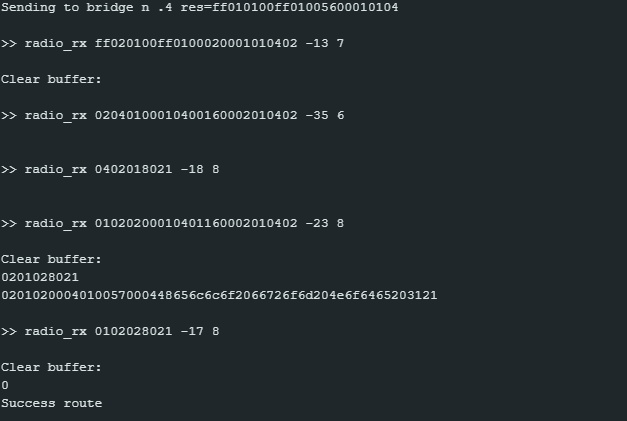
Node 2 ได้รับ discovery response จาก Node 4 และตอบ Acknowledge Node 4 ก่อนที่จะส่งต่อไปที่ Node 1



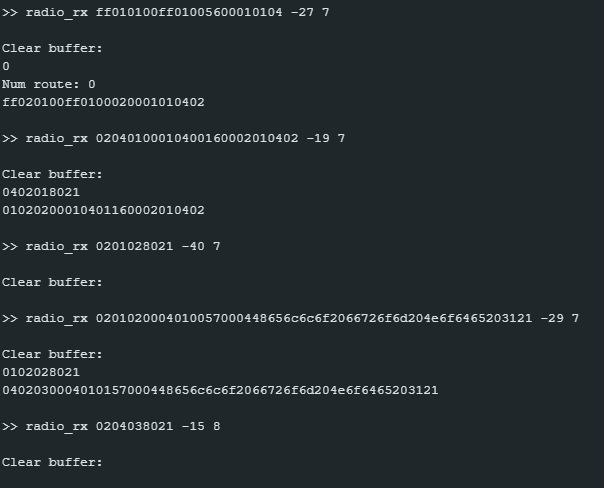
Node 4 ได้รับ Acknowledge จาก Node 2



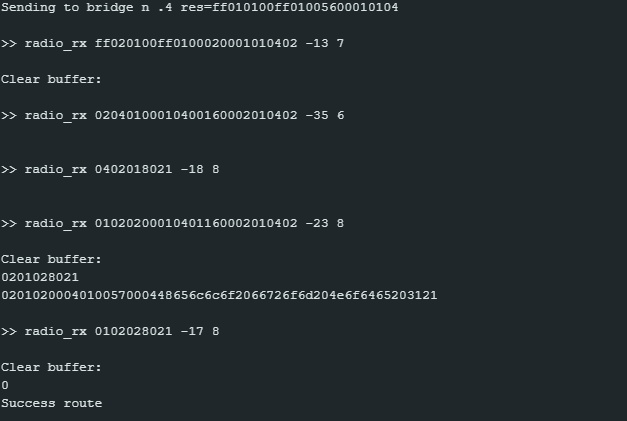
Node 1 ได้รับ discovery response และตอบ Acknowledge Node 2 เมื่อได้รับเส้นทางแล้วจึงส่งข้อความ application message ไปผ่าน Node 2 ไปยัง Node 4



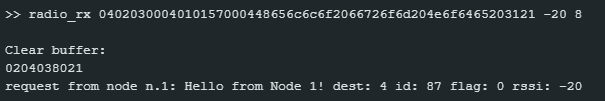
Node 2 ได้รับ application mesasge จาก Node 1 จึงตอบ Acknowledge Node 1 และส่งต่อ application message ไปที่ Node 4



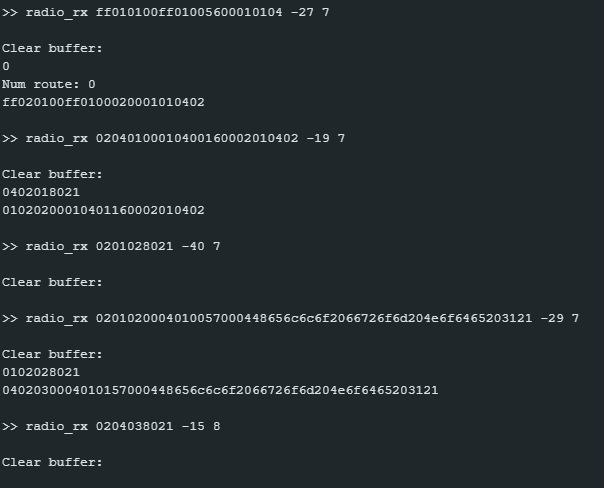
Node 1 ได้รับ Acknowledge จากการส่ง application message ไปที่ Node 2



Node 4 ได้รับ application message ที่ส่งต่อจาก Node 2 และตอบ Acknowledge Node 2 โดยตัวข้อความมี address ปลายทางตรงกับ Node 4



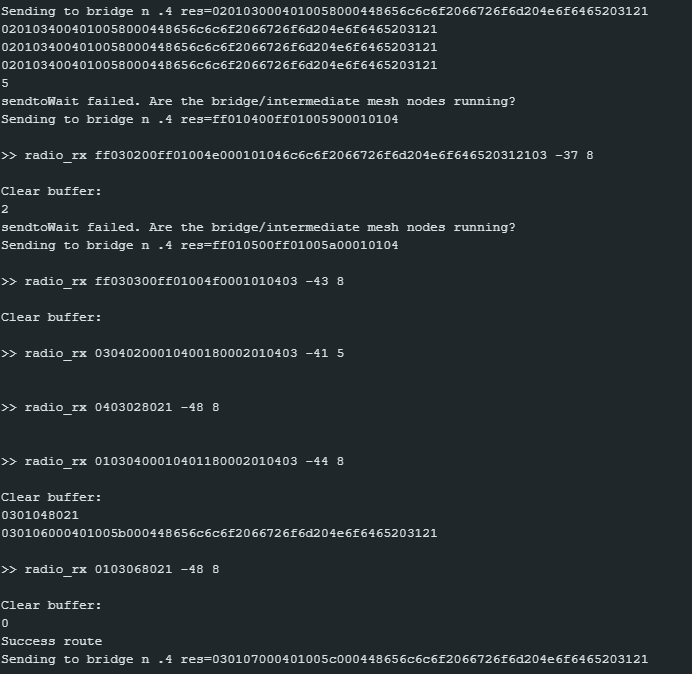
Node 2 ได้รับ Acknowledge จากการส่งต่อข้อความ application message



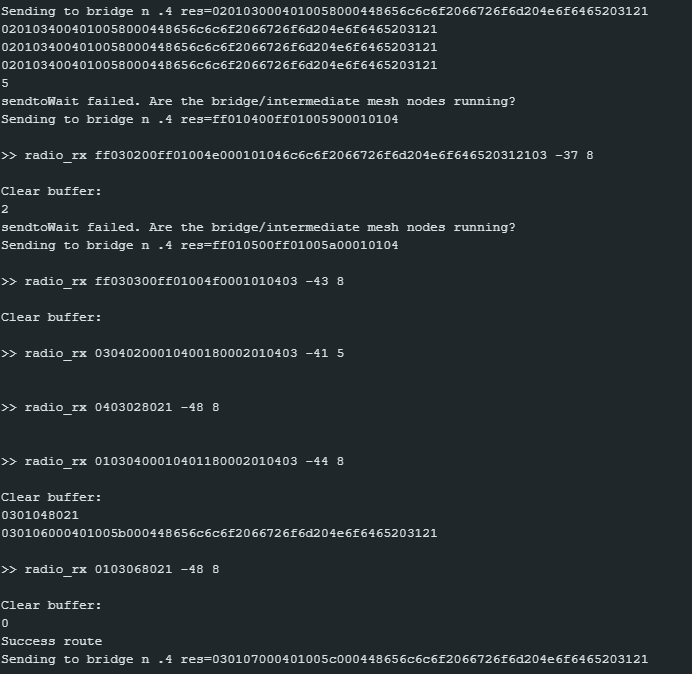
(Step 2)

Action: ทำการปิด Node 2

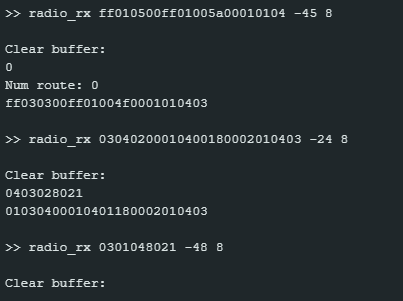
Node 1 ที่พยายามจะส่ง application message ไปที่ Node 4 ผ่าน Node 2 แต่เมื่อ Node 2 ถูกปิดไปทำให้ข้อความไม่สามารถไปถึง Node 4 และได้ฟังก์ชั่น sentToWait คืนค่า 5 ออกมาซึ่งหมายถึงไม่สามารถส่งข้อความได้สำเร็จ



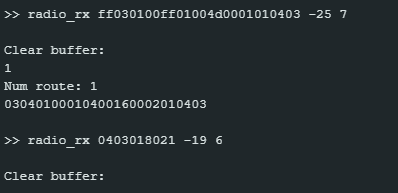
Node 4 จึงต้องทำ discovery ใหม่เพิ่มหาเส้นทางในการส่งใหม่



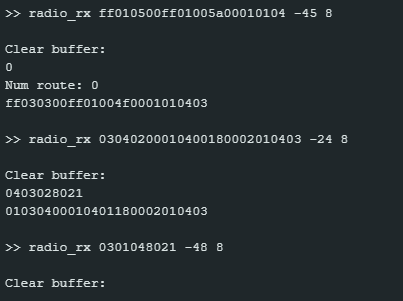
Node 3 ที่ตอนแรกที่ไม่สามารถทำการส่งต่อ broadcast discovery message ได้เร็วกว่า Node 2 จึงเขามาหน้าที่ส่งต่อไปที่ Node 4 แทน



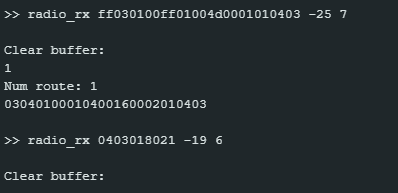
Node 4 ได้รับ re-broadcast discovery message จาก Node 3 และตอบ discovery response กลับไปที่ Node 3



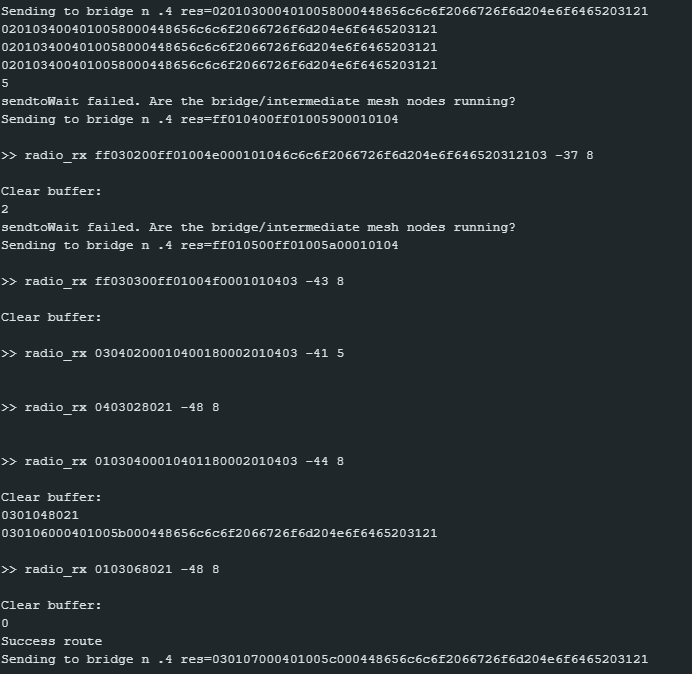
Node 3 ได้รับ discovery response จาก Node 4 และตอบ Acknowledge Node 4 แล้วจึงส่งต่อ discovery response ไปที่ Node 1



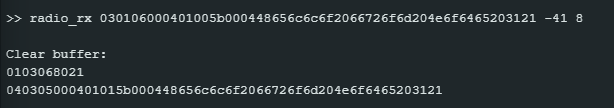
Node 4 ได้รับ Acknowledge จากการส่ง discovery response ไปที่ Node 3



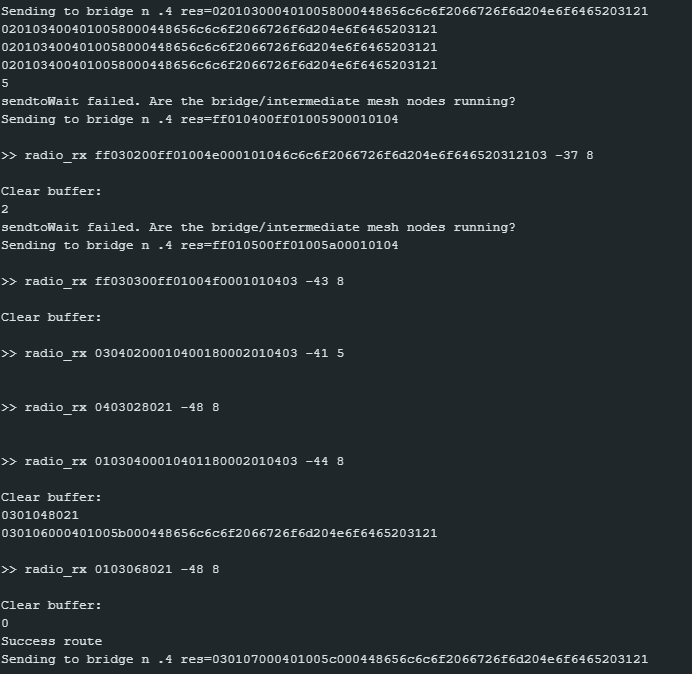
Node 1 ได้รับ discovery response และตอบ Acknowledge Node 3 เมื่อได้รับเส้นทางแล้วจึงส่งข้อความ application message ไปผ่าน Node 3 ไปยัง Node 4

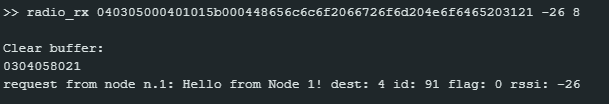


Node 3 ได้รับ application message จาก Node 1 และตอบ Acknowledge Node 1 ก่อนที่จะส่งต่อ application message ไปยัง Node 4

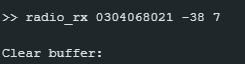


Node 1 ได้รับ Acknowledge จากการส่ง application message ไปยัง Node 3



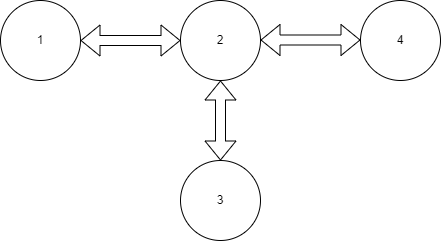
Node 4 ได้รับ application message ที่ส่งต่อจาก Node 3 และตอบ Acknowledge Node 3 โดยตัวข้อความมี address ปลายทางตรงกับ Node 4

Node 3 ได้รับ acknowledge จาก Node 4 จากการส่งต่อ application message



**Test case #6 การรับส่งข้อมูลระหว่างอุปกรณ์ 4 ตัว แบบจำกัดการส่งแบบที่ 3 (Star)**

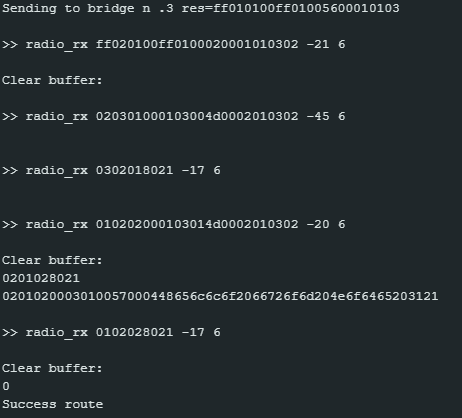
จะใช้ Sender Node 1 ตัว, Relay Node 1 ตัว และ Receiver Node 2 ตัว ทำการทดสอบโดยการให้ Sender Node กับ Receiver Node โดยทำให้ไม่สามารถส่งข้อความได้โดยตรง จะต้องวิ่งผ่าน relay node



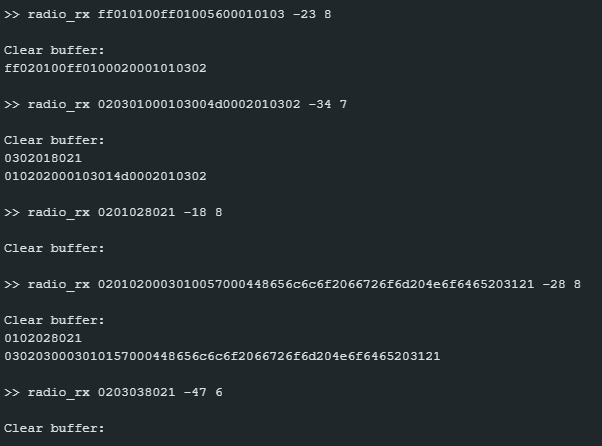
การส่งข้อความจะทำการส่งไปที่ Node 3 ก่อนเมื่อส่งสำเร็จจะทำการส่งไปที่ Node 4 เพื่อแสดงให้เห็นว่า

Result:

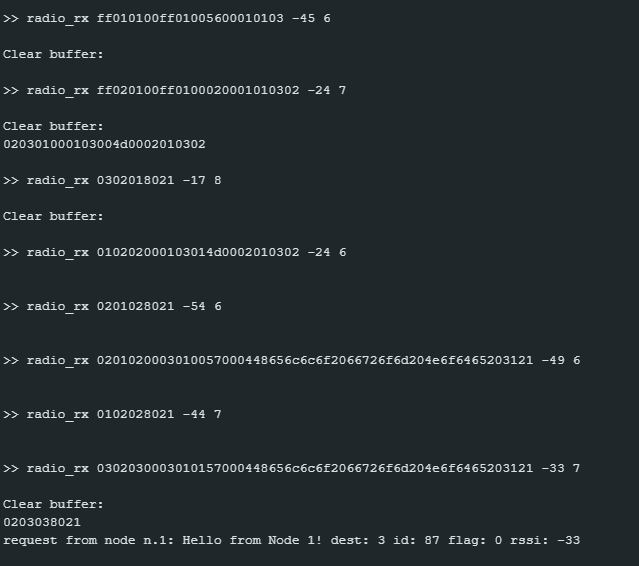
Node 1 ต้องการส่งไปที่ Node 3 แต่ยังไม่มีเส้นจึงทำการส่ง broadcast discovery message



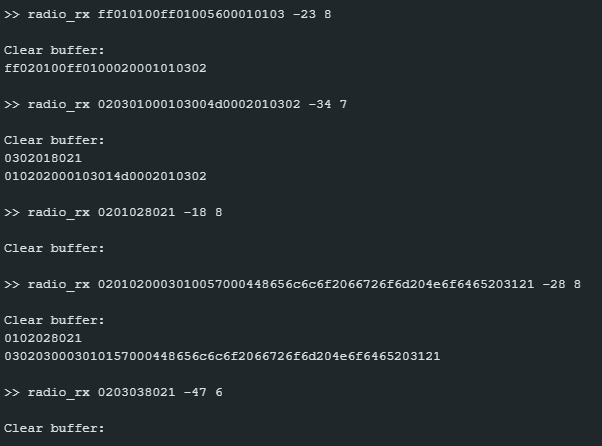
Node 2 ได้รับ broadcast discovery message จาก Node 1 และทำการ re-broadcast discovery message



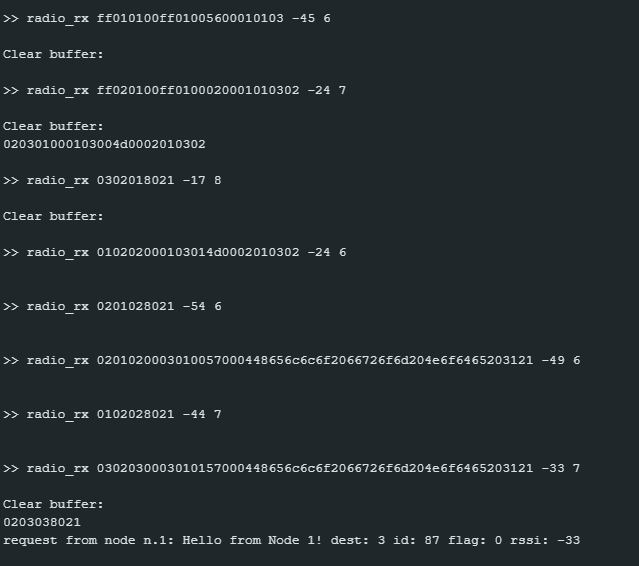
Node 3 ได้รับ discovery message จาก Node 2 และตอบกลับด้วย discovery response ไปยัง Node 2



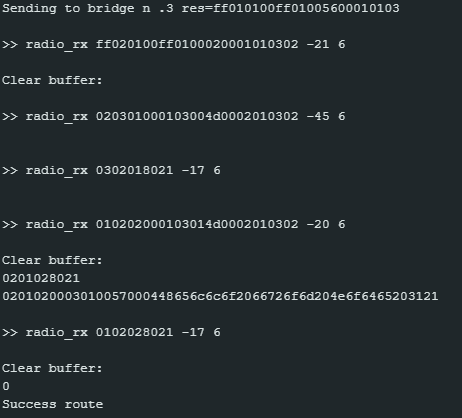
Node 2 ได้รับ discovery response จาก Node 3 และตอบ Acknowledge Node 3 และส่งต่อ discovery response ไปที่ Node 1



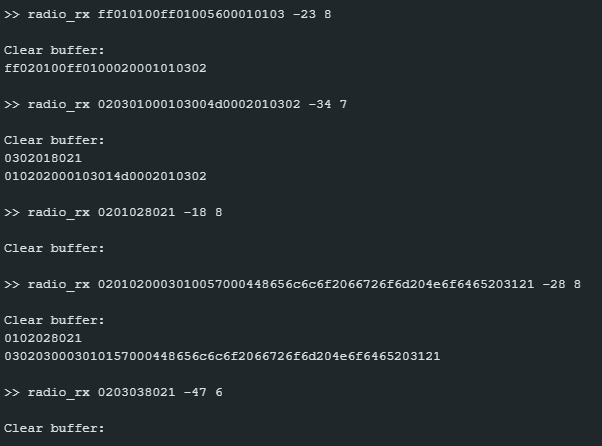
Node 3 ได้รับ Acknowledge จากการส่ง discovery response ไปยัง Node 2



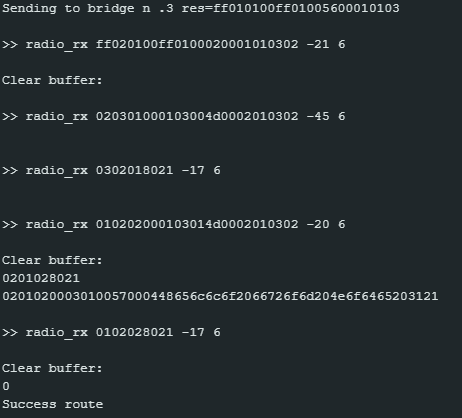
Node 1 ได้รับ discovery response จาก Node 2 และตอบ acknowledge Node 2 ก่อนที่จะส่งต่อไปที่ application message ไปยัง Node 4 ผ่าน Node 2



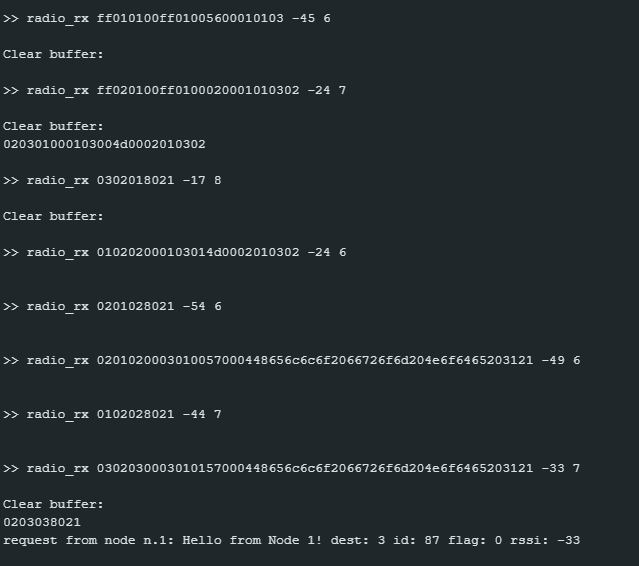
Node 2 ได้รับ Acknowledge จากการส่งต่อ discovery response และได้รับ application message จาก Node 1 จึงตอบ acknowledge Node 1 แล้วจึงส่งต่อไปที่ Node 3



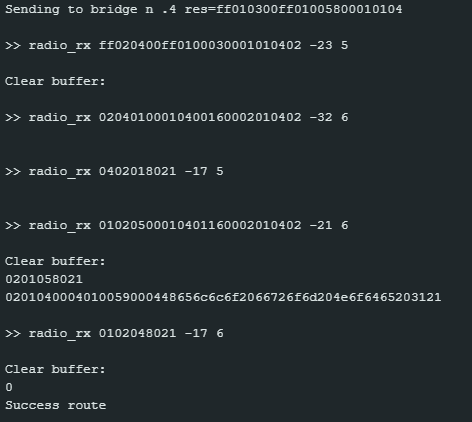
Node 1 ได้รับ acknowledge จากการส่ง application message ไปที่ Node 2



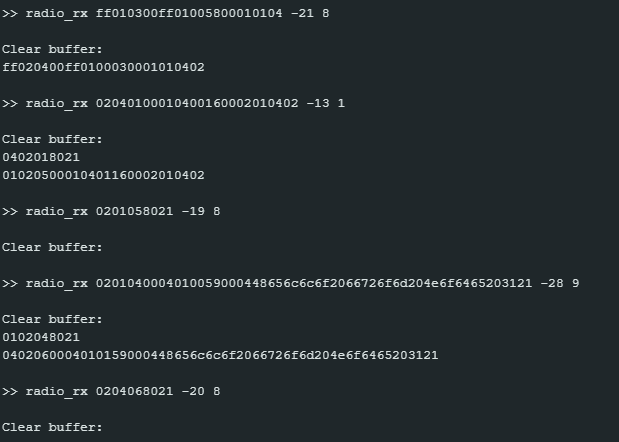
Node 3 ได้รับ application message จาก Node 2 และตอบ acknowledge ไปยัง Node 2 โดยตัวข้อความมี address ปลายทางตรงกับ Node 3



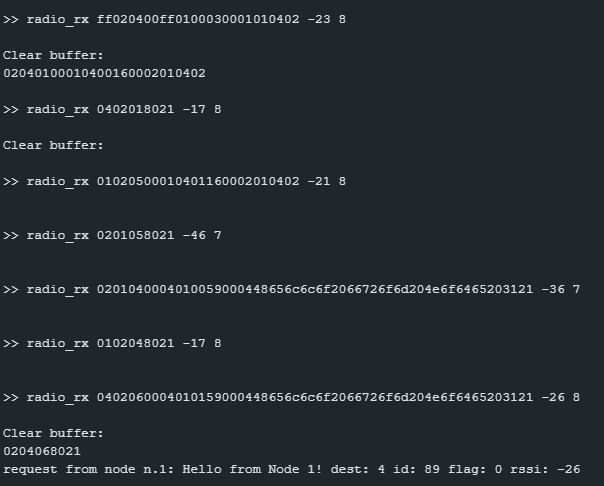
เมื่อส่งไปที่ Node 3 เสร็จแล้วจึงจะส่งไปที่ Node 4 จึงทำการหาเส้นทางโดยการ broadcast discovery message



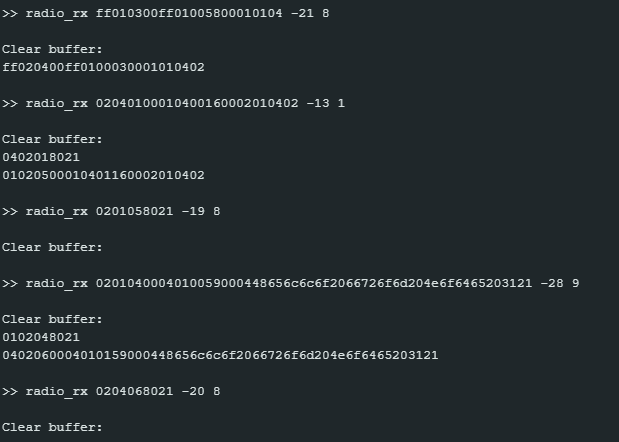
Node 2 ได้รับ broadcast discovery message จาก Node 1 และทำการ re-broadcast discovery message



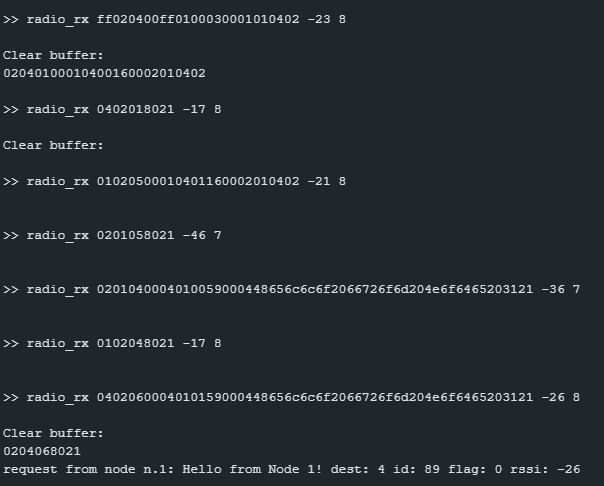
Node 4 ได้รับ discovery message จาก Node 2 และตอบกลับด้วย discovery response ไปยัง Node 2



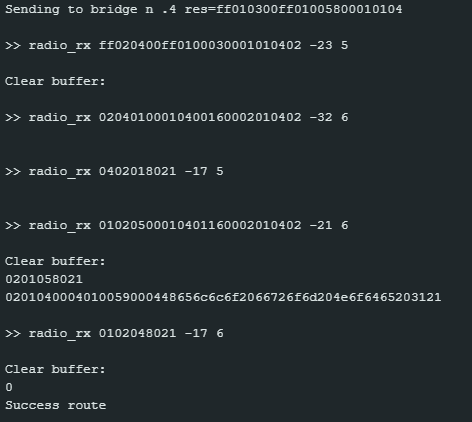
Node 2 ได้รับ discovery response จาก Node 4 และตอบ Acknowledge กลับ Node 2 แล้วส่งต่อ discovery response ไปยัง Node 1



Node 4 ได้รับ acknowledge จากการส่ง discovery response ไปยัง Node 2

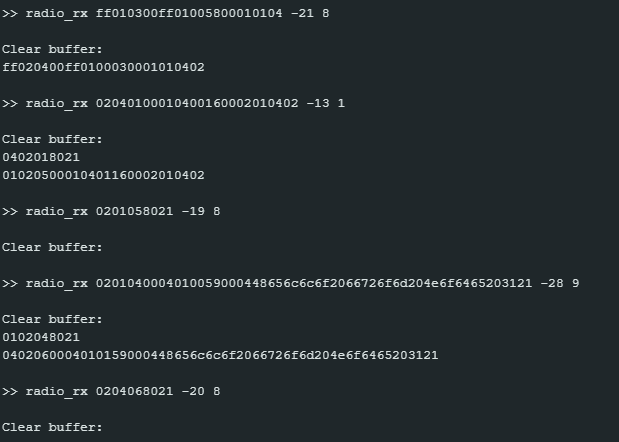


Node 1 ได้รับ discovery response จาก Node 2 และตอบ Acknowledge กลับไปยัง Node 2 เมื่อได้เส้นทางแล้วจึงส่ง application message ไปยัง Node 4 ผ่าน Node 2

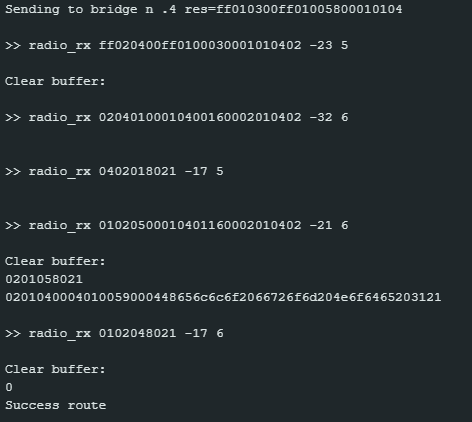


Node 2 ได้รับ acknowledge จากการส่ง discovery response ยัง Node 1

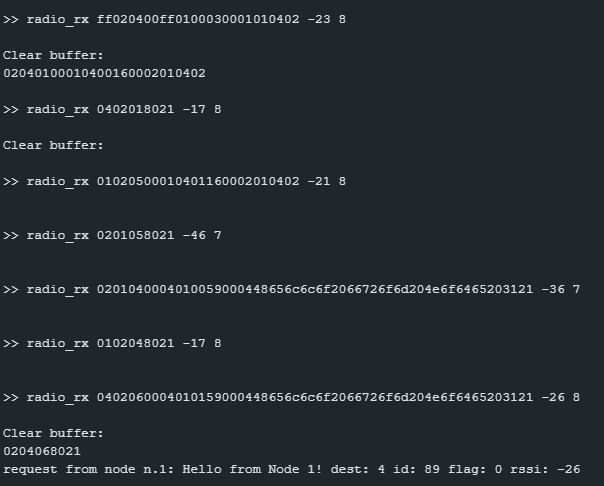
Node 2 ได้รับ appliacatione message จาก Node 1 และตอบ acknowledge Node 1 แล้วจึงส่งต่อไปที่ Node 4



Node 1 ได้รับ acknowledge จากการส่ง application ไป Node 2



Node 4 ได้รับ application message จาก Node 2 และตอบ acknowledge ไปยัง Node 2 โดยตัวข้อความมี address ปลายทางตรงกับ Node 4



Node 2 ได้รับ acknowledge จากการส่งต่อ application message ไปยัง Node 4

