การเพิ่มอัตราการสร้างกุญแจระดับกายภาพจากค่าความแรงสัญญาณโดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัย

สุนทรี ใจคง

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยบูรพา
ธันวาคม 2561

คณะกรรมการควบคุมวิทยานิพนธ์และคณะกรรมการสอบวิทยานิพนธ์ ได้พิจารณาวิทยานิพนธ์ของ สุนทรี ใจคง ฉบับนี้แล้ว เห็นสมควรรับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า ของมหาวิทยาลัยบูรพาได้

คณะกรรมการควบคุมวิทยานิพนธ์

..... อาจารย์ที่ปรึกษาหลัก
(ดร. ภาณุวัฒน์ ด่านกลาง)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธาน
(ผู้ช่วยศาสตราจารย์ ดร. มนตรี โพธิโสโนทัย)

..... กรรมการ
(ดร. ภาณุวัฒน์ ด่านกลาง)

..... กรรมการ
(รองศาสตราจารย์ วิรุฬห์ ศรีบริรักษ์)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. ฤภูวัลย์ จันทรสา)

คณะวิศวกรรมศาสตร์อนุมัติให้รับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า ของมหาวิทยาลัยบูรพา

..... คณบดีคณะวิศวกรรมศาสตร์
(ผู้ช่วยศาสตราจารย์ ดร. ณยศ กุรุกิจโกศล)
วันที่ 28 เดือน ธันวาคม พ.ศ. 2561

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงด้วยการได้รับความกรุณาให้คำปรึกษาเสนอแนะแนวทางที่ถูกต้อง และตรวจแก้ไขข้อบกพร่องต่าง ๆ อย่างดียิ่งจาก ดร. ภาณุวัฒน์ ด่านกลาง อาจารย์ที่ปรึกษาหลัก ที่ให้คำแนะนำ ชี้แนะแนวทาง และ ดร. อภิรัฐ ลิ่มมณี ที่สนับสนุนช่วยเหลือและเป็นกำลังใจ ประธานคณะกรรมการสอบวิทยานิพนธ์ คณะกรรมการสอบวิทยานิพนธ์ ที่ให้คำแนะนำ ผู้วิจัยรู้สึกซาบซึ้งและขอกราบขอบพระคุณทุกท่านเป็นอย่างสูงไว้ ณ ที่นี้

ขอขอบพระคุณผู้ทรงคุณวุฒิทุกท่านที่ได้กรุณาตรวจสอบความสมบูรณ์และให้คำแนะนำแก้ไขเครื่องมือในการวิจัย รวมทั้งผู้เชี่ยวชาญทุกท่าน

ท้ายที่สุดผู้วิจัยขอกราบขอบพระคุณ บิดา มารดา พี่ น้อง และเพื่อน ๆ ที่ได้ให้ความช่วยเหลือและกำลังใจ ตลอดจนผู้ที่เกี่ยวข้องทุกท่านที่มิได้กล่าวถึงในที่นี้

คุณค่า และประโยชน์ของวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบเป็นกตัญญูกตเวทิตาแด่บิดา มารดา ครู อาจารย์ และผู้มีพระคุณทุกท่าน ที่ได้อบรมสั่งสอน และให้กำลังใจแก่ผู้วิจัยเสมอมา

สุนทรี ใจคง

56910512: สาขาวิชา: วิศวกรรมไฟฟ้า; วศ.ม. (วิศวกรรมไฟฟ้า)
คำสำคัญ: RSSI/ KEY GENERATION/ LEVEL-CROSSING ALGORITHM/ HAMMING CODE/ SECURE NETWORK CODING

สุนทรี ใจคง: การเพิ่มอัตราการสร้างกุญแจระดับกายภาพจากค่าความแรงสัญญาณโดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัย (INCREASING PHYSICAL-LAYER KEY GENERATION RATE FROM RSS WITH SECURE NETWORK CODING) คณะกรรมการควบคุมวิทยานิพนธ์: ภาณุวัฒน์ ด่านกลาง, ปร.ด. 61 หน้า. ปี พ.ศ. 2561.

การเข้ารหัสเป็นการทำให้แน่ใจว่าข้อความที่เข้ารหัสจะไม่สามารถอ่านออกได้โดยศัตรู ถ้าไม่มีกุญแจถอดรหัส แต่ก่อนจะเข้ารหัสทั้งสองฝั่งต้องมีการตกลงวิธีการแลกเปลี่ยนกุญแจกันก่อน สำหรับการส่งข้อมูลแบบไร้สายการแลกเปลี่ยนกุญแจกันอาจทำให้ถูกดักฟังระหว่างทางได้ ในงานวิจัยนี้นำค่าความแรงของสัญญาณที่ได้รับ (Received signal strength indicator, RSSI) มาสร้างเป็นกุญแจเข้ารหัสที่ไม่จำเป็นต้องแลกเปลี่ยนกุญแจกันโดยตรงโดยใช้วิธีการ Level-crossing algorithm ในการสร้างกุญแจ ซึ่งจากการทดลองเบื้องต้น เนื่องจากการสร้างกุญแจจากค่าความแรงสัญญาณนั้นมีลักษณะสุ่มโดยธรรมชาติจึงทำให้บางครั้งการสร้างบิตกุญแจของทั้งสองฝั่งไม่ตรงกันทั้งหมดทำให้ไม่สามารถนำไปใช้งานได้ จึงทดลองเพิ่มการตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming-code generated redundancy: HGR ซึ่งช่วยทำให้ลดบิตกุญแจที่ไม่ตรงกันได้แต่อัตราการสร้างกุญแจที่ได้ยังต่ำ จึงได้เสนอโปรโตคอลใหม่สำหรับเพิ่มอัตราการสร้างกุญแจที่สามารถเพิ่มได้อย่างมากที่สุดเป็นสามเท่า โดยโปรโตคอลนี้อาศัยโหนดเพื่อนบ้าน (Neighboring node) หนึ่งโหนดที่เชื่อถือได้มาช่วยในการสร้างกุญแจ และอาศัยการเข้ารหัสเครือข่ายแบบปลอดภัย (Secure network coding) มาช่วยในการส่งข้อมูลด้านความปลอดภัยบางส่วน

56910512: MAJOR: ELECTRICAL ENGINEERING; M.Eng. (ELECTRICAL ENGINEERING)
KEYWORD: RSSI/ KEY GENERATION/ LEVEL-CROSSING ALGORITHM/ HAMMING CODE/ SECURE NETWORK CODING
SOONTREE JAIKHONG: INCREASING PHYSICAL-LAYER KEY GENERATION RATE FROM RSS WITH SECURE NETWORK CODING. ADVISORY COMMITTEE: PANUWAT DANKLANG, Ph.D. 61 P. 2018.

Encryption ensures that the resulting ciphertext cannot be understood by the enemy if they do not have the key. Before encryption can be done, both the transmitting and receiving sides have to reach an agreement regarding how the key is exchanged. Direct exchange using wireless channel can easily be wiretapped. In this work, we employ the mutual received signal strength indicator (RSSI) to construct the key independently at both sides without the need for key exchange. The level-crossing algorithm is used for this purpose. Due to the randomness and noise in the wireless channel, the key constructed by the algorithm are not exactly the same. Therefore, the key reconciliation using Hamming-code generated redundancy (HGR) is proposed to reduce the number of different key bits. In addition, a new protocol is proposed to increase the key generation rate by maximally three times. The protocol employs a trusted neighboring node for this purpose and uses secure network coding for transmitting some secure side information.

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญตาราง (ต่อ)

สารบัญภาพ

สารบัญภาพ (ต่อ)

บทที่ 1
บทนำ

ความเป็นมาและความสำคัญ

เทคโนโลยีการสื่อสารข้อมูลเป็นสิ่งที่สร้างขึ้นมาเพื่อเพิ่มความสะดวกสบายให้กับมนุษย์ เริ่มต้นจากการสร้างเทคโนโลยีที่มีการสื่อสารข้อมูลผ่านสาย ใช้สื่อสารกันในระยะใกล้หรือที่สามารถลากสายถึงกันได้ ต่อมามีการวิจัยและพัฒนาเทคโนโลยีการสื่อสารให้เป็นแบบไร้สาย ที่สามารถเคลื่อนย้าย ติดตั้งและขยายเครือข่ายได้สะดวกมากขึ้น ซึ่งเครือข่ายเทคโนโลยีแบบไร้สายได้รับความนิยมนำไปใช้งานเป็นจำนวนมากทั้งในองค์กร สถานศึกษา หรือการใช้งานส่วนบุคคล แต่อาจเกิดปัญหาในด้านความปลอดภัยของข้อมูลเพราะระบบเครือข่ายแบบไร้สายเป็นการส่งข้อมูลไปในอากาศ อาจเกิดการโจมตีข้อมูลระหว่างทางจากผู้ไม่หวังดีที่ต้องการขโมยข้อมูลหรือทำให้ข้อมูลสูญหายได้ เพื่อให้ระบบเครือข่ายแบบไร้สายทำงานได้อย่างมีประสิทธิภาพการเลือกวิธีรักษาความปลอดภัยของข้อมูลที่ดีจึงเป็นสิ่งจำเป็นอย่างยิ่งในการส่งข้อมูลบนเครือข่ายไร้สาย

การเข้ารหัสเป็นวิธีการหนึ่งที่พัฒนาขึ้นมาเพื่อรักษาความปลอดภัยให้กับข้อมูล ซึ่งสิ่งที่สำคัญสำหรับการเข้ารหัส คือ การรักษากุญแจให้เป็นความลับเพราะการเข้ารหัสเป็นการนำข้อความและกุญแจลับมาเข้ากระบวนการทำให้ข้อความที่ต้องการส่งไม่สามารถอ่านออกได้ ถ้าไม่มีกุญแจลับที่ใช้ในการถอดรหัส ดังนั้นผู้สื่อสารที่ต้องการแลกเปลี่ยนข้อมูลที่เข้ารหัสจะต้องหาวิธีการแลกเปลี่ยนกุญแจลับที่มีเฉพาะผู้ที่ได้รับอนุญาตเท่านั้นที่สามารถรู้กุญแจลับได้ สำหรับเครือข่ายไร้สาย การรักษาความปลอดภัยมีการแลกเปลี่ยนกุญแจเข้ารหัสผ่านช่องสัญญาณไร้สายทำให้เป็นช่องทางให้ผู้ไม่หวังดีมีโอกาสที่จะพยายามดักฟังกุญแจในระหว่างทางได้ ถึงแม้การแลกเปลี่ยนกุญแจที่นิยมใช้ในปัจจุบันที่มีรูปแบบการแลกเปลี่ยนกุญแจด้วยการใช้กุญแจสาธารณะ (Public key) ที่มีบุคคลที่สามที่เชื่อถือได้เป็นผู้ให้บริการในการแลกเปลี่ยนกุญแจให้กับผู้สื่อสารแต่อัลกอริทึมก็มีความซับซ้อนมากและกุญแจเข้ารหัสก็ต้องส่งผ่านช่องสัญญาณอยู่ดีจึงยังเป็นปัญหาที่ว่าอาจโดนดักฟังกุญแจระหว่างทางที่ส่งได้

จากการส่งข้อมูลแบบเครือข่ายไร้สายนั้นผู้วิจัยเห็นว่าการรับส่งข้อมูลแบบไร้สายผู้รับและผู้ส่งต้องได้รับสัญญาณของเครือข่ายจึงจะสามารถรับส่งข้อมูลได้ ดังนั้นผู้วิจัยจึงสนใจในการศึกษาคุณสมบัติที่ผู้รับและผู้ส่งจะสามารถนำค่าความแรงของสัญญาณ (Received signal strength indicator, RSSI) ที่ได้รับนั้นมาสร้างเป็นกุญแจเข้ารหัสที่ไม่จำเป็นต้องแลกเปลี่ยนกุญแจกัน จากการศึกษาและทบทวนวรรณกรรมนั้นได้พบว่า วิธีการ Level-crossing algorithm เป็นวิธีที่

สามารถสร้างกุญแจเข้ารหัสจากค่าความแรงสัญญาณที่ได้รับ (RSSI) ได้ แต่เนื่องจากการสร้างกุญแจจากค่าความแรงสัญญาณเป็นการสร้างกุญแจเข้ารหัสที่เป็นแบบสมมาตร (Symmetric key) ซึ่งผู้ส่งจะต้องได้กุญแจที่เหมือนกันกับผู้รับจึงจะสามารถถอดรหัสได้ ถึงแม้ว่าวิธีการ Level-crossing algorithm จะสามารถสร้างกุญแจได้เหมือนกันทั้งผู้รับและผู้ส่งแต่จากการทดลองเบื้องต้นวิธีการนี้ให้อัตราการสร้างกุญแจที่ต่ำและเนื่องจากการสร้างกุญแจจากค่าความแรงสัญญาณที่มีลักษณะสุ่มโดยธรรมชาติทำให้บิตกุญแจที่ทั้งสองฝั่งสร้างได้ไม่ตรงกันทั้งหมดทำให้ไม่สามารถนำไปใช้งานได้ เราจึงทดลองเพิ่มการตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming-code generated redundancy: HGR เข้ามาช่วยเพื่อกำจัดบิตกุญแจที่ไม่ตรงกันออกแล้วเราได้เสนอโปรโตคอลที่สามารถเพิ่มอัตราการสร้างกุญแจมากขึ้นจากแบบเดิมเป็นสามเท่า โปรโตคอลนี้อาศัยโหนดเพื่อนบ้าน (Neighboring node) หนึ่งโหนดที่เชื่อถือได้มาช่วยในการสร้างกุญแจ และอาศัยการเข้ารหัสเครือข่ายแบบปลอดภัย (Secure network coding) มาช่วยในการส่งกุญแจของเพื่อนบ้านโดยไม่ให้ผู้ดักฟังทราบ โปรโตคอลนี้เหมาะกับเครือข่ายที่มีหลายโหนดที่ต้องการทำงานเพื่อบรรลุวัตถุประสงค์อย่างเดียวกัน เช่น เครือข่ายเซ็นเซอร์ไร้สาย

วัตถุประสงค์ของการวิจัย

1. เพื่อศึกษาการเข้ารหัสข้อมูล
2. เพื่อศึกษาการสร้างกุญแจเข้ารหัสข้อมูล
3. เพื่อวิเคราะห์และค้นหาเทคนิคการสร้างกุญแจเข้ารหัสข้อมูลจากการวัดค่าความแรงของสัญญาณที่ได้รับ
4. เพื่อเสนอวิธีการเพิ่มอัตราการสร้างกุญแจจากค่าความแรงสัญญาณโดยใช้เทคนิคการเข้ารหัสเครือข่ายแบบปลอดภัย
5. เพื่อวัดประสิทธิภาพของวิธีการใหม่ที่นำเสนอเมื่อเทียบกับวิธีการที่มีอยู่แล้ว

ขอบเขตของการวิจัย

1. ศึกษาการทำงานและทบทวนวรรณกรรมที่เกี่ยวกับการสร้างกุญแจเข้ารหัสข้อมูลจากค่าความแรงสัญญาณที่ได้รับ
2. วิเคราะห์ความปลอดภัยของกุญแจเข้ารหัสที่สร้างขึ้นจากวิธีการใหม่ที่นำเสนอ
3. วัดประสิทธิภาพของการสร้างกุญแจจากวิธีการที่มีอยู่ในวรรณกรรม
4. วัดประสิทธิภาพของการสร้างกุญแจจากวิธีการใหม่ที่นำเสนอขึ้นมา

5. เปรียบเทียบความปลอดภัยและประสิทธิภาพในการสร้างกุญแจระหว่างวิธีการที่มีอยู่ในวรรณกรรมกับวิธีการใหม่ที่นำเสนอ

ประโยชน์ที่คาดว่าจะได้รับจากการวิจัย

1. เข้าใจหลักการเข้ารหัสข้อมูล
2. ได้ความรู้เรื่องเทคนิคการสร้างกุญแจเข้ารหัสข้อมูลจากค่าความแรงของสัญญาณที่ได้รับ
3. สามารถสร้างระบบการสร้างกุญแจลับที่มีอัตราเร็วในการสร้างกุญแจเพิ่มขึ้น
4. สามารถเปรียบเทียบความปลอดภัยและประสิทธิภาพในการสร้างกุญแจระหว่างวิธีการที่มีอยู่ในวรรณกรรมกับวิธีการใหม่ที่นำเสนอได้

ขั้นตอนและวิธีการดำเนินการวิจัย

1. ศึกษารูปแบบการสร้างกุญแจจากค่าความแรงสัญญาณ
2. ออกแบบระบบการสร้างกุญแจที่มีอัตราเร็วในการสร้างกุญแจเพิ่มขึ้น
3. ออกแบบการทดลอง และเปรียบเทียบความปลอดภัยและประสิทธิภาพในการสร้างกุญแจระหว่างวิธีการที่มีอยู่ในวรรณกรรมกับวิธีการใหม่ที่นำเสนอ
4. สรุปผลความปลอดภัยและประสิทธิภาพในการสร้างกุญแจระหว่างวิธีการที่มีอยู่ในวรรณกรรมกับวิธีการใหม่ที่นำเสนอ

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

การเพิ่มอัตราการสร้างกุญแจระดับกายภาพจากค่าความแรงสัญญาณโดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัย ผู้วิจัยได้ทำการศึกษา ค้นคว้า เอกสารและงานวิจัยที่เกี่ยวข้องต่าง ๆ สรุปสาระสำคัญ ดังนี้

1. การเข้ารหัส
2. เทคโนโลยีเครือข่ายไร้สาย
3. ภัยคุกคามเครือข่ายไร้สาย
4. การรักษาความปลอดภัยทางกายภาพ
5. การสร้างกุญแจเข้ารหัส
6. ความสุ่ม
7. การเข้ารหัสเครือข่าย

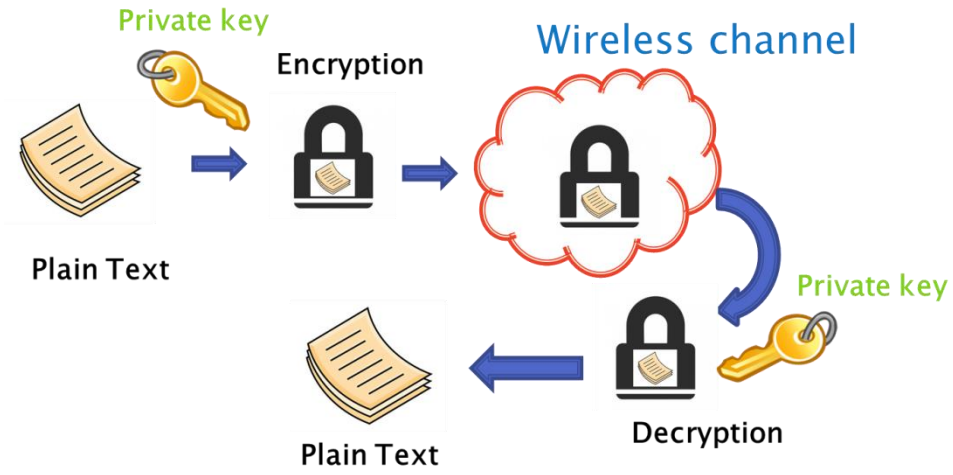
การเข้ารหัส

การเข้ารหัสเป็นวิธีการแปลงข้อความที่ต้องการส่งให้เป็นความลับ ซึ่งจะมีเฉพาะบุคคลที่ได้รับอนุญาตเท่านั้นที่จะสามารถรู้ข้อความที่เข้ารหัสได้ ในเริ่มแรกการเข้ารหัสถูกใช้ในทางทหาร แต่ต่อมาถูกใช้ทั่วไปกับการส่งข้อมูลที่ต้องการเก็บเป็นความลับ สิ่งสำคัญที่จะทำให้การเข้ารหัสปลอดภัยขึ้นอยู่กับการเลือกใช้อัลกอริทึมและประเภทของกุญแจ ปัจจุบันการเข้ารหัสสามารถแบ่งกุญแจได้เป็น 2 ประเภท ดังนี้

แบบกุญแจสมมาตร (Symmetric key) เป็นการใช้กุญแจส่วนบุคคล (Private keys) ในการเข้ารหัสและถอดรหัส ซึ่งบุคคลที่จะสามารถรู้ข้อความได้ต้องมีกุญแจที่ได้ตกลงที่ใช้ในการเข้าและถอดรหัสก่อนเท่านั้นจึงจะสามารถรู้ข้อความได้

ข้อดี เป็นอัลกอริทึมที่มีประสิทธิภาพในเรื่องความเร็วในการทำงานเพราะกระบวนการไม่ซับซ้อนและยังให้ความปลอดภัยอยู่ในระดับที่ได้รับมาตรฐาน

ข้อเสีย เนื่องจากจะต้องใช้กุญแจส่วนบุคคล (Private keys) ในการเข้าและถอดรหัสทำให้เมื่อต้องการส่งข้อความเข้ารหัสให้กับผู้รับที่มากกว่าหนึ่งจำเป็นต้องทำการสร้างกุญแจส่วนบุคคล (Private keys) ให้กับแต่ละผู้รับทำให้การจัดการกุญแจมีความยุ่งยาก



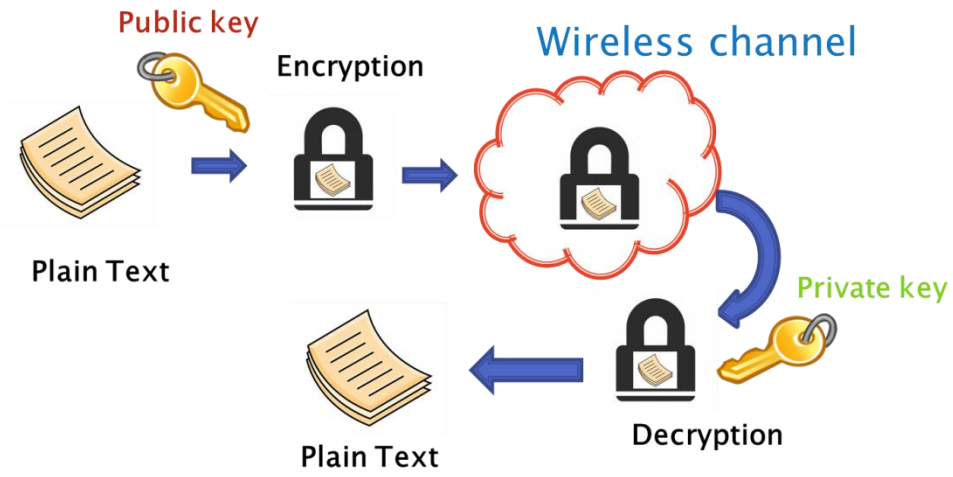
ภาพที่ 2-1 การเข้ารหัสข้อมูลแบบกุญแจสมมาตร

ปัจจุบันอัลกอริทึมที่นิยมใช้การเข้ารหัสแบบกุญแจสมมาตรนั้นได้แก่ Advanced encryption standard (AES) เนื่องจากเป็นอัลกอริทึมที่มีกระบวนการทำงานที่ใช้การสลับตำแหน่งข้อความ 4 แบบ ประกอบด้วย การเข้ารหัสจะเริ่มสลับตำแหน่งแบบ Sub-bytes Shift-rows Mix-columns และ Add-round-key ตามลำดับ ส่วนการถอดรหัสการสลับตำแหน่งจะสลับโดยทำย้อนกลับจากการเข้ารหัส สำหรับกุญแจการเข้ารหัสของ AES มีขนาดตั้งแต่ 128 192 และ 256 บิต ให้เลือกใช้ ซึ่งถ้าเลือกใช้กุญแจที่มีความยาวมากก็จะทำให้ความปลอดภัยมากขึ้นตามไปด้วย ผู้ใช้สามารถเลือกใช้ได้ตามความเหมาะสมตามระดับความปลอดภัยที่ต้องการ

แบบกุญแจอสมมาตร (Asymmetric key) เป็นวิธีที่เข้ามาช่วยแก้ปัญหาในด้านการจัดการกุญแจให้กับการเข้ารหัสแบบกุญแจสมมาตร ซึ่งการเข้ารหัสแบบกุญแจอสมมาตรนั้นจะใช้กุญแจแบบคู่ ซึ่งประกอบด้วย กุญแจส่วนบุคคล (Private keys) และกุญแจสาธารณะ (Public keys) โดยจะใช้กุญแจสาธารณะซึ่งเป็นกุญแจที่ทุกคนสามารถรู้ได้นำมาเข้ารหัสและผู้ที่ต้องการถอดรหัสจะต้องใช้กุญแจส่วนบุคคลที่เป็นคู่กุญแจกับกุญแจเข้ารหัสเท่านั้นจึงจะสามารถถอดรหัสได้ ซึ่งการเข้ารหัสแบบกุญแจอสมมาตรรหัสนั้น ถ้าต้องการส่งข้อความให้กับผู้รับหลายคนก็ไม่ต้องกังวลว่าจะจัดการส่งกุญแจเข้ารหัสอย่างไร

ข้อดี ช่วยทำให้การจัดการกุญแจง่ายขึ้นกว่าวิธีการใช้กุญแจแบบสมมาตรที่จะต้องหาวิธีที่ปลอดภัยในการแลกเปลี่ยนกุญแจก่อนใช้จริง

ข้อเสีย อัลกอริทึมที่ใช้ในการคำนวณกุญแจมีความซับซ้อนทำให้ถ้านำมาใช้เข้ารหัสข้อความที่มีขนาดใหญ่จะใช้เวลามาก



ภาพที่ 2-2 การเข้ารหัสข้อมูลแบบอสมมาตร

ในปัจจุบันมีอัลกอริทึม RSA เป็นวิธีการใช้สมการคณิตศาสตร์การหาจำนวนเฉพาะมาใช้ในการคำนวณแลกเปลี่ยนกุญแจ และมีการพัฒนา ECC ที่ใช้สมการเส้นโค้งวงรีมาคำนวณทำให้ได้กุญแจที่สั้นกว่าอัลกอริทึม RSA แต่ให้ความปลอดภัยเท่ากัน

เทคโนโลยีเครือข่ายไร้สาย

เทคโนโลยีเครือข่ายไร้สายถูกพัฒนามาให้เหมาะกับการใช้งานในพื้นที่ที่อุปกรณ์แบบมีสายไม่สามารถเข้าถึงได้ทำให้การติดตั้งและขยายเครือข่ายสะดวกมากขึ้น ซึ่งการสื่อสารข้อมูลของเทคโนโลยีเครือข่ายไร้สายจะอาศัยคลื่นแม่เหล็กไฟฟ้าในช่วงความถี่เดียวกันเป็นสื่อกลางในการส่งสัญญาณหากัน ปัจจุบันมีการพัฒนาเทคโนโลยีไร้สายมากมายและนิยมใช้กันอย่างแพร่หลาย เช่น เทคโนโลยี โทรศัพท์มือถือ Wi-Fi Bluetooth UWB และ ZigBee เป็นต้น

Raychaudhuri and Mandayam (2012) ที่ได้แบ่งเทคโนโลยีไร้สายที่ใช้คลื่นวิทยุในการสื่อสารเป็น 2 กลุ่ม กลุ่มแรกเป็นการสื่อสารสำหรับส่งข้อมูลไปยังผู้ใช้ปลายทางและกลุ่มที่สองเป็นการสื่อสารระหว่างอุปกรณ์ไปยังอุปกรณ์ (Machine-to-machine) ที่ใช้คลื่นวิทยุระยะสั้นในการสื่อสาร

เทคโนโลยีโทรศัพท์มือถือ (Cellular) ได้เริ่มพัฒนาระบบจากเทคโนโลยี Second-generation (2G) ที่ย่านความถี่ 850 MHz และ 1.9 GHz สามารถรับส่งข้อมูลด้วยความเร็ว 22.8 Kbps ใช้ในการสื่อสารข้อมูลแบบเสียง ต่อมาได้พัฒนามาเป็น 2.4G 3G และ 4G ตามลำดับ ซึ่งเทคโนโลยี 4G นั้นสามารถรับส่งข้อมูลที่ความเร็ว 100 Mbps ได้

สำหรับในส่วนของเครือข่ายท้องถิ่นไร้สาย (Wireless local area network, WLAN) หรือเทคโนโลยี Wi-Fi ที่ได้รับการรับรองมาตรฐาน IEEE 802.11 โดยสถาบันวิชาชีพวิศวกรไฟฟ้าและอิเล็กทรอนิกส์ (Institute of electrical and electronics engineers, IEEE) นั้นใช้ย่านความถี่ 2.4 5 และ 60 GHz ในการส่งข้อมูลซึ่งให้ความเร็วในการส่งข้อมูลต่ำสุดอยู่ที่ 11 Mbps และมากสุดที่ 1 Gbps จากเทคโนโลยีทั้งสองที่กล่าวจะอยู่ในกลุ่มแรกดังภาพที่ 2-3

Technology	PHY	MAC Protocol	Frequency band	Channel spacing	Service Bit-Rate (downlink)
Cellular					
2G(GSM)	GMSK	TDMA/FDMA	850MHz &1.9GHz	200kHz	22.8 Kbps
2G(IS95)	QPSK	CDMA	825-849 MHz	1.25MHz	115 Kbps
2.5G (Edge)	GMSK/ 8PSK	FDMA/ TDMA	850MHz &1.9GHz	200kHz	236.8 Kbps
3G (UMTS)	DQPSK	WCDMA	1.8 – 2.5 GHz	5MHz	384- 2048 Kbps
3G+ (CDMA 2000)	BPSK/ QPSK	CDMA	1.8 – 2.5 GHz	5 MHz	2.5 – 15.7 Mbps
3G+ (HSPA)	BPSK/QPSK	CDMA	1.8 – 2.5 Ghz	5 Mhz	22-56 Mbps
4G(LTE)	64QAM/ MIMO	OFDMA/ SC-FDMA	2 - 8 GHz	20MHz	100 Mbps
WLAN					
802.11a/g/n	64QAM	OFDM	5/2.4/5 GHz	20/20/40 MHz	54/54/200 Mbps
802.11b	DQPSK	FDMA	2.4 GHz	20MHz	11 Mbps
802.11ac	256QAM/ MU-MIMO	OFDM/ SDMA	5 GHz	80MHz	500 Mbps
802.11ad	64QAM	OFDM	60GHz	2160MHz	1 Gbps

ภาพที่ 2-3 Radio technologies for local and wide area networks

Technology	PHY	MAC Protocol	Frequency band	Channel spacing	Service Bit-Rate (downlink)
Bluetooth	GFSK	FHSS	2.4 GHz	1 MHz	1 Mbps
UWB	BPSK/QPSK	DS-UWB/ MB-OFDM	3.1-10.6 GHz	500MHz-7.5GHz	54-1024 Mbps
ZigBee	BPSK/ O-QPSK	DSSS	868/915MHz/ 2.4 GHz	0.3/0.6 MHz 2 MHz	250 Kbps

ภาพที่ 2-4 Radio technologies for short-range communication

จากภาพที่ 2-4 แสดงให้เห็นถึงเทคโนโลยีไร้สายในกลุ่มที่ใช้คลื่นวิทยุระยะสั้นในการสื่อสารระหว่างอุปกรณ์ไปยังอุปกรณ์ เช่น เทคโนโลยี Bluetooth ที่ได้รับมาตรฐาน IEEE 802.15 ที่ใช้ย่านความถี่ 2.4 GHz สามารถส่งข้อมูลที่ความเร็ว 1 เมกะบิตต่อวินาที ซึ่งใช้งานในอุปกรณ์ที่ส่งข้อมูลระยะสั้นโดยไม่ต้องใช้สาย เช่น การส่งข้อมูลเสียงจากหูฟังไปยังโทรศัพท์มือถือ เทคโนโลยีต่อมา Ultra-wide band (UWB) ใช้ความถี่ช่วง 3.1 ถึง 10.6 GHzในการสื่อสารสามารถส่งข้อมูลได้เร็ว 50 ถึง 1,024 Mbps ใช้สำหรับส่งข้อมูลที่มีจำนวน เช่น กล้องวีดิโอ แต่จะส่งได้ในระยะทางที่ใกล้ ๆ เท่านั้น และเทคโนโลยี ZigBee ที่ได้มีการออกแบบให้ใช้พลังงานในการส่งข้อมูลต่ำ ใช้ย่านความถี่ 2.4 GHz การใช้งานมักใช้ในการส่งข้อมูลประเภทเซนเซอร์

จากเทคโนโลยีไร้สายที่หลากหลายนั้นผู้วิจัยมีความสนใจในการศึกษาเทคโนโลยีเครือข่ายท้องถิ่นไร้สาย หรือที่มีชื่อเรียกอีกแบบว่า เทคโนโลยี Wi-Fi ซึ่งเทคโนโลยีที่นิยมใช้กันเป็นจำนวนมาก

ภัยคุกคามเครือข่ายไร้สาย

ในด้านความปลอดภัยของการส่งข้อมูลเครือข่ายไร้สายเนื่องจากเป็นการส่งข้อมูลโดยการแปลงข้อมูลให้สามารถส่งไปในช่องสัญญาณแบบไร้สาย ซึ่งอาจทำให้บุคคลที่ไม่ไดรับอนุญาตในการเข้าถึงข้อมูลแต่อยู่ในระหว่างทางที่ส่งข้อมูลสามารถเข้าถึงสัญญาณที่ทำการส่งข้อมูลได้ทำให้เกิดภัยคุกคาม (จตุชัย แพงจันทร์, 2558) ดังนี้

1. การดักฟัง (Eavesdropping) เป็นการดักฟังระหว่างทางที่มีการส่งข้อมูลแบบไร้สาย
2. การปลอมตัว (Spoofing) เป็นการโจมตีที่หลอกให้ผู้รับหรือส่งข้อมูลเชื่อว่าผู้โจมตีเป็นบุคคลที่ต้องการส่งข้อมูลให้จึงทำให้ได้รับข้อมูลที่ผู้รับและส่งต้องการจะส่งให้กัน

3. การดัดแปลง (Modification) บางครั้งการโจมตีไม่เพียงแต่ต้องการขโมยข้อมูลแต่อาจทำให้ข้อมูลเสียโดยการนำข้อมูลที่ถูกต้องจริง ๆ ไปดัดแปลงแก้ไขแล้วส่งไปยังผู้รับทำให้ได้รับข้อมูลที่ไม่ถูกต้องหรือหลอกลวงเพื่อให้ได้ข้อมูลที่อาจก่อให้เกิดการเสียทรัพย์สินมีค่า

4. การรบกวนเครือข่าย (Jamming) โดยการส่งสัญญาณรบกวนให้กับระบบเครือข่ายไร้สายเพื่อให้ระบบไม่สามารถทำงานได้ตามปกติ

5. การโจมตีแบบ Denial-of-service (D-o-S) เป็นการพยายามขัดขวางผู้ใช้งานไม่ให้สามารถร้องขอการใช้งานจากผู้ให้บริการได้

จากการโจมตีแบบต่าง ๆ ที่ได้กล่าวมานั้น การป้องกันโดยการเข้ารหัสที่ได้กล่าวรายละเอียดไว้ในหัวข้อการเข้ารหัสนั้นเป็นการรักษาความปลอดภัยของข้อมูลบนเครือข่ายไร้สายที่ทำงานในชั้นบน (Upper layers) ถึงแม้ข้อมูลที่ต้องการส่งจะถูกเข้ารหัสไว้แต่ในส่วนของชั้นกายภาพ (Physical layer) ที่ทำหน้าที่ในการรับส่งสัญญาณและเชื่อมต่อกันแบบไร้สายนั้นเราไม่สามารถรู้ได้ว่าอุปกรณ์นั้นอยู่ตำแหน่งใดและเป็นอุปกรณ์ที่ได้รับอนุญาตอย่างถูกต้องในการเข้าถึงข้อมูลจริงหรือไม่

การรักษาความปลอดภัยทางกายภาพ

ในการเข้ารหัสแบบสมมาตรนั้น สิ่งสำคัญ คือ ผู้ส่งและรับข้อมูลจะต้องมีกุญแจเดียวกันโดยไม่ให้ผู้ดักฟังได้รับกุญแจนี้ ดังนั้น เมื่อมีการตกลงว่าจะใช้กุญแจอะไร หรือเมื่อมีการตกลงจะใช้กุญแจใหม่ ควรจะมั่นใจได้ว่าการสร้างกุญแจดังกล่าว (Key generation) เป็นกระบวนการสุ่ม นอกจากนี้ การส่งข้อมูลเกี่ยวกับกุญแจหรือการแลกกุญแจ (Key exchange) จะต้องส่งผ่านช่องทางที่ปลอดภัยด้วย

หนึ่งในวิธีที่นิยมใช้ในการสร้างกุญแจจากกระบวนการสุ่ม (Randomness) นั้น คือการเก็บปรากฏการที่มีลักษณะสุ่มอยู่แล้วโดยธรรมชาติ เช่น สัญญาณรบกวนทางความร้อน (Thermal Noise) แล้วนำมาสร้างกุญแจ จากงานวิจัยของ Aono, Higuchi, Ohira, Komiyama and Sasaoka (2005) ได้เสนอแนวคิดการสร้างกุญแจจากคุณสมบัติของช่องสัญญาณของการสื่อสารไร้สาย ซึ่งย่อมมีลักษณะสุ่มอยู่แล้วโดยธรรมชาติ เช่น Rayleigh fading channel หรือ Rician fading channel ในบทความนั้น พวกเขายังได้อธิบายเกี่ยวกับการสร้างสายอากาศที่มีประสิทธิภาพในการสร้างกุญแจดังกล่าวโดยใช้คุณสมบัติหนึ่งของช่องสัญญาณที่เรียกว่า Received signal strength indication (RSSI) ซึ่งหมายถึง ความแรงของสัญญาณที่ได้รับนั่นเอง จะเห็นว่าวิธีของ Aono et al. (2005) นั้น นอกจากจะทำให้ได้กุญแจที่มีลักษณะสุ่มแล้ว ยังสามารถสร้างกุญแจที่เหมือนกันในภาคส่งและภาครับโดยไม่ต้องแลกกุญแจกันโดยตรง เนื่องจากช่องสัญญาณจากภาคส่งไปยังภาครับ

เป็นช่องเดียวกับภาครับไปภาคส่ง ถ้าทั้งสองฝ่ายเก็บช่องสัญญาณได้ถูกต้อง กุญแจที่แต่ละฝ่ายสร้างขึ้นเองย่อมเหมือนกันโดยไม่ต้องมีการตกลงกันเป็นพิเศษ ทำให้ตัดปัญหาในเรื่องความไม่ปลอดภัยในการแลกเปลี่ยนกุญแจ

อย่างไรก็ตาม การเก็บ RSSI นี้อาจมีความผิดพลาดในการเก็บซึ่งเกิดจากตัวอุปกรณ์หรือสัญญาณรบกวน ซึ่งอาจทำให้กุญแจที่สร้างขึ้นไม่ตรงกันได้ ดังนั้น Wallace (2009) จึงได้เสนอวิธี Channel quantization และ Random pre-encryption เพื่อให้สามารถสร้างกุญแจจาก Multipath channel ได้ตรงกัน โดยกุญแจที่สร้างไม่ได้มาจากความแรงของสัญญาณ (RSSI) เท่านั้นแต่ยังมาจากเฟสของสัญญาณได้ด้วย ทั้งหมดนี้เรียกรวมกันว่า ข้อมูลสถานะช่องสัญญาณ (Channel state information, CSI) นอกจากนี้ยังได้เสนอขีดจำกัดทางทฤษฎีสารสนเทศ (Information theoretic limit) เพื่อระบุว่าอะไร คือ ปริมาณกุญแจสูงสุดที่สามารถสร้างได้ในแต่ละช่องสัญญาณ ต่อมา Sharma and Wallace (2010) ได้นำแนวคิดนี้มาสร้างกุญแจจากช่องสัญญาณ MIMO (Multiple-input Multiple-output) โดยใช้สายอากาศแบบอะเรย์สี่สายที่ตัวส่งและสี่สายที่ตัวรับ ทำการทดลองภายในอาคารโดยตัวส่งและตัวรับติดตั้งอยู่บนรถที่สามารถเคลื่อนที่ไปมาได้ จากผลการทดลองพบว่าอัตราการสร้างกุญแจไม่คงที่แต่ส่วนใหญ่จะอยู่ในช่วง 5 ถึง 10 บิตต่อช่องสัญญาณระหว่างรถหยุดนิ่ง อย่างไรก็ตามเมื่อมีการเคลื่อนที่สามารถสร้างกุญแจได้เพิ่มขึ้น โดยเมื่อเคลื่อนที่ไป 50 เมตร สามารถสร้างกุญแจรวมได้ถึง 15 Kbps

Limmanee and Henkel (2010) ได้พิจารณาในกรณีที่ผู้ดักฟังอยู่ใกล้ตัวส่งและตัวรับ ซึ่งอาจจะทำให้สามารถรู้ข้อมูลบางส่วนของกุญแจ และเสนอวิธีเข้ารหัสกุญแจที่สร้างจากช่องสัญญาณทำให้ได้กุญแจอีกชุดซึ่งผู้ดักฟังจะไม่สามารถรู้ข้อมูลใด ๆ ได้เลย

Parvez, Abdul and Sarwat (2016) ได้นำเสนอวิธีการรักษาความปลอดภัยข้อมูลเครือข่ายไร้สายโดยวิธีการหาตำแหน่งจากความแรงสัญญาณ (RSSI) และประมาณค่าความเป็นไปได้สูงสุด (Maximum likelihood estimator, MLE) จากแนวคิด เมืองอัจฉริยะ (Smart cities) ที่มีระบบมิเตอร์อัจฉริยะ (Advanced metering infrastructure, AMI) ทำหน้าที่ให้บริการในส่วนของ การเก็บควบคุมและตรวจสอบการใช้ไฟฟ้าอัตโนมัติ ซึ่งมิเตอร์อัจฉริยะ (Smart meters) ที่ทำหน้าที่บันทึกและรายงานการใช้ไฟฟ้าของแต่ละบ้านไปยัง ศูนย์ควบคุมส่วนกลาง (Control center) โครงสร้างของระบบมิเตอร์อัจฉริยะนี้ใช้เครือข่ายการสื่อสารข้อมูลแบบไร้สาย ถ้าเกิดการโจมตีมิเตอร์อัจฉริยะแล้วเรียนรู้รูปแบบของข้อมูลการใช้งานอุปกรณ์อิเล็กทรอนิกส์ ระบบจะเกิดความเสียหายได้ จึงเสนอวิธีการรักษาความปลอดภัยข้อมูลของมิเตอร์อัจฉริยะระหว่างที่ส่งข้อมูลเข้ารหัสไปยังศูนย์ควบคุมส่วนกลางมีการตรวจสอบมิเตอร์ทุกตัวก่อนรับข้อมูลเพื่อให้แน่ใจว่าเป็นมิเตอร์ที่อยู่ในเขตพื้นที่ที่ได้รับอนุญาตเท่านั้น

การสร้างกุญแจเข้ารหัส

Zhang, Duong, Marshall and Woods (2016) ได้แบ่งขั้นตอนสำหรับการวิเคราะห์และพิจารณาในการสร้างกุญแจจากการเก็บค่าความแรงของสัญญาณที่ได้รับเป็น 4 ขั้นตอน

1. Probing channel เป็นการตรวจสอบช่องสัญญาณของผู้รับและผู้ส่ง เพื่อเตรียมความพร้อมก่อนเข้ากระบวนการสร้างกุญแจ ซึ่งค่าที่ต้องพิจารณามี 3 ส่วนหลักดังต่อไปนี้

1.1 Channel parameter เป็นการกำหนดว่าจะใช้ข้อมูลใดในช่องสัญญาณเป็นแหล่งสร้างกุญแจ ซึ่งสิ่งที่สำคัญ คือ แหล่งข้อมูลนั้นจะต้องมีความสุ่ม สำหรับลักษณะแหล่งข้อมูลช่องสัญญาณมี 2 แบบดังนี้

1.1.1 Channel state information (CSI) จะให้รายละเอียดเกี่ยวกับข่าวสารข้อมูลในช่องสัญญาณ โดยมีแหล่งอ้างอิงข้อมูลจากสองแหล่งที่สำคัญ คือ Channel impulse response (CIR) เป็นผลการตอบสนองพัลส์ ซึ่งจะมีค่าการเปลี่ยนเฟส (Phase shift) และ หน่วงเวลา (Time delay) สามารถนำมาพิจารณาหาลักษณะของช่องสัญญาณได้ และ Channel frequency response (CFR) เป็นผลการตอบสนองความถี่ โดยพิจารณาที่การเปรียบเทียบสเปกตรัมความถี่ระหว่างฝั่งส่งและรับสัญญาณ เพื่อประมาณค่าช่องสัญญาณ

1.1.2 ความแรงสัญญาณ (RSS) เป็นการเก็บค่าความแรงจากสัญญาณที่ได้รับ ปัจจุบันเป็นที่นิยมในการนำไปสร้างกุญแจสามารถประยุกต์ใช้กับระบบ IEEE 802.11 หรือ ระบบ IEEE 802.15.4 แต่เนื่องจากข้อมูล RSS บอกลักษณะของสัญญาณไม่ระเอียดมากพอจึงอาจเป็นจุดอ่อนที่จะเกิดการโจมตีช่องสัญญาณได้ ดังนั้นในการนำไปใช้สร้างกุญแจจึงมีความจำเป็นที่จะต้องเพิ่มวิธีการที่ช่วยให้ปลอดภัยมากขึ้น

1.2 Signal pre-processing ในขั้นตอนนี้เป็นการประมาณค่าสัญญาณล่วงหน้า เนื่องจากการเก็บสัญญาณของทั้งสองฝั่งไม่ได้เกิดในช่วงเวลาเดียวกันส่งผลทำให้ค่าสัญญาณที่เก็บไว้ไม่เท่ากันจึงต้องทำการประมาณการเก็บสัญญาณให้มีจำนวนที่เพียงพอต่อการนำไปใช้งาน

1.3 Probing channel rate เป็นค่าที่บอกถึงอัตราการตรวจสอบช่องสัญญาณ ซึ่งการเก็บสัญญาณที่ไม่มีการเปลี่ยนตำแหน่งจะได้อัตราการตรวจสอบช่องสัญญาณคงที่ ทำให้ถ้าค่าอัตราการตรวจสอบช่องสัญญาณเปลี่ยนแปลงอย่างรวดเร็วก็จะสามารถทราบได้ว่ามีความผิดปกติเกิดขึ้นในกระบวนการตรวจสอบช่องสัญญาณ

2. Quantization เป็นการแปลงค่าสัญญาณที่ได้รับให้อยู่ในรูปแบบ Binary values ที่จะนำไปใช้ในการสร้างกุญแจ ซึ่งสามารถกำหนดระดับ Quantization ด้วยค่า thresholds ซึ่งเป็นสิ่งสำคัญที่มีผลต่อ อัตราการสร้างกุญแจ (Key generation rate) และ อัตรากุญแจที่ต่างกัน (Key disagreement rate) ในวรรณกรรมของ Mathur, Trappe, Mandayam, Ye and Reznik (2008). ได้

เลือกใช้การคำนวณหาค่า Thresholds จากค่าเฉลี่ย (μ) กับค่าเบี่ยงเบนมาตรฐานของสัญญาณ (σ) และฟังก์ชั่นการแจกแจงสะสม (Cumulative distribution function, CDF) ร่วมกันในการอ้างอิงระดับของสัญญาณการ Quantization ซึ่งมีสมการดังนี้

$$\begin{aligned} q+ &= \mu + (\alpha \times \sigma) \\ q- &= \mu - (\alpha \times \sigma) \end{aligned} \tag{2-1}$$

เมื่อ $\alpha \neq 0$

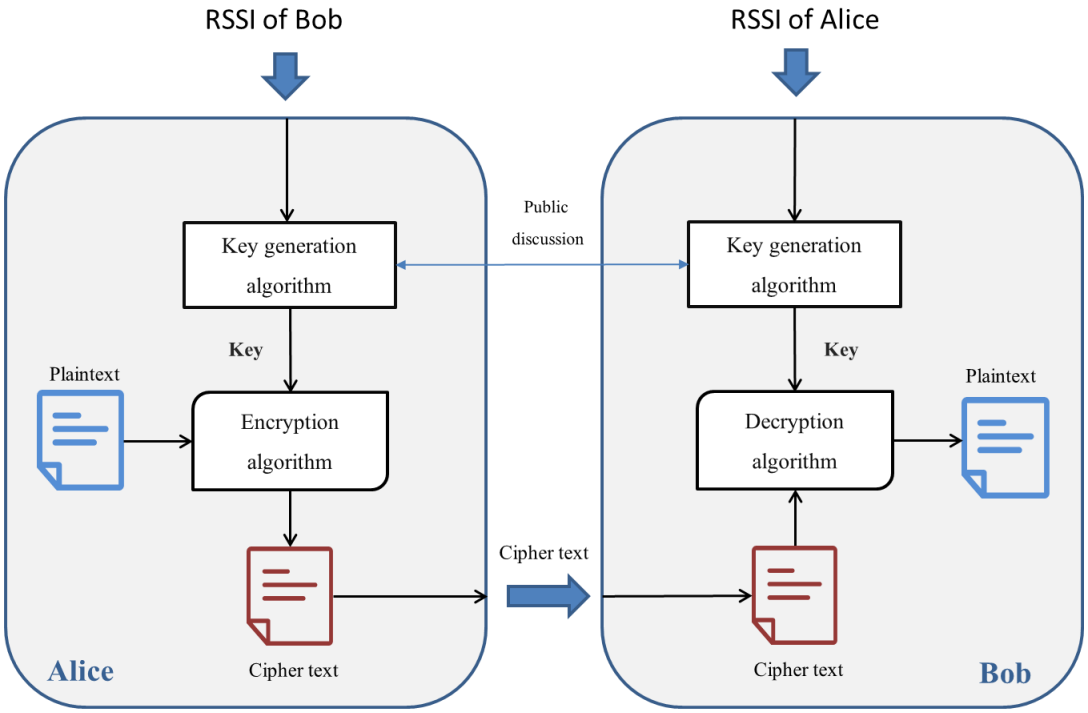
α คือ ค่าที่กำหนดระดับของ thresholds

จากการกำหนดค่าอ้างอิงสำหรับการ Quantization ที่กล่าวนี้เป็นวิธีที่สามารถปรับระดับของการ Quantization และค่า thresholds ได้ตามความเหมาะสมที่จะทำให้ได้ค่าที่เป็นความสุ่มอย่างมีประสิทธิภาพ

3. Information reconciliation หลังจากการประมวลผลสัญญาณและทำการแปลงสัญญาณให้เป็นบิตที่จะใช้ในการสร้างกุญแจที่ได้กล่าวมาแล้วนั้นมีโอกาสที่จะเกิดความผิดพลาดจากค่าบิตที่สร้างไม่ตรงกัน ดังนั้นควรมีการกำหนดค่าพารามิเตอร์ให้ตรงกันหรือเลือกใช้โปรโตคอลที่เข้ามาช่วยลดความผิดพลาด เช่น Error correcting code low-density parity-check และ BCH code เป็นต้น โดยขั้นตอนนี้จะช่วยตรวจสอบความผิดพลาดและแก้ไขข้อความ แต่ถ้าข้อความมีความผิดพลาดมากเกินไประบบจำเป็นจะต้องกลับไปเริ่มต้นกระบวนการ Channel probing ใหม่

4. Privacy amplification จากขั้นตอน Information reconciliation จะมีข้อความบางส่วนที่จะถูกเปิดเผยต่อสาธารณะ ดังนั้นในขั้นตอนนี้เป็นการป้องกันไม่ให้ข้อความบางส่วนที่เปิดเผยนั้นถูกนำไปใช้ในการคาดเดากุญแจได้ สามารถป้องกันโดยการนำข้อความไปผ่านกระบวนการ Hash function ที่ทำให้ข้อความที่แตกต่างกันมี hash value ที่ต่างกันแต่มีขนาดเท่ากัน ซึ่งข้อความที่ผ่าน Hash function จะไม่สามารถทำย้อนกลับได้ ซึ่งสามารถนำไปใช้ในการตรวจสอบการปลอมแปลงข้อความได้

จากการวิเคราะห์และการพิจารณาที่ได้กล่าวมานั้นผู้วิจัยมีความสนใจในการศึกษาระบบการสร้างกุญแจจากค่าความแรงสัญญาณ (RSSI) ซึ่งกุญแจที่สร้างนั้นจะเป็นประเภทกุญแจแบบสมมาตร (Symmetric key) ที่การเข้าและถอดรหัสต้องเป็นกุญแจเดียวกันจึงจะสามารถถอดรหัสข้อความลับที่ได้



ภาพที่ 2-5 ภาพรวมการเข้ารหัสโดยใช้กุญแจที่สร้างจากค่าความแรงสัญญาณ (RSSI)

จากภาพที่ 2-5 แสดงให้เห็นถึงภาพรวมการเข้ารหัสโดยใช้กุญแจที่สร้างจากค่าความแรงสัญญาณ (RSSI) สมมติให้ Alice ต้องการส่งข้อความลับให้ Bob เริ่มต้นการทำงานโดย Alice และ Bob ทำการเก็บค่าความแรงสัญญาณ (RSSI) ของกันและกัน เมื่อได้ค่าความแรงสัญญาณกระบวนการต่อไปทั้งสองฝั่งทำการสร้างกุญแจด้วยอัลกอริทึมการสร้างกุญแจ หลังจากนั้น Alice นำกุญแจไปใช้สำหรับเข้ารหัสแล้วส่งข้อความลับให้กับ Bob เมื่อ Bob ได้รับข้อความลับก็ทำการถอดรหัสด้วยกุญแจที่สร้างขึ้น ซึ่งกุญแจที่ทั้ง Alice และ Bob สร้างนั้นเป็นประเภทกุญแจแบบกุญแจสมมาตร (Symmetric key) ดังนั้น Bob จะต้องสร้างกุญแจให้ได้เหมือนกันกับ Alice จึงจะสามารถถอดรหัสข้อความลับที่ได้รับจาก Alice ได้

Mathur et al. (2008). ได้เสนอวิธีการนำสัญญาณที่ได้รับมาสร้างกุญแจ โดยระบบไม่จำเป็นต้องแลกเปลี่ยนกุญแจแต่สามารถใช้ Level crossing algorithm เป็นพื้นฐานในการกำหนดและคัดเลือกสัญญาณที่จะมาแปลงเป็นกุญแจ ซึ่งพวกเขาได้พิจารณาการสร้างกุญแจดังนี้

1. ความยาวกุญแจที่เหมาะสม พิจารณาจากขนาดความยาวกุญแจที่สามารถใช้งานร่วมกับรูปแบบกุญแจเข้ารหัสแบบสมมาตร คือ ตั้งแต่ขนาด 128 บิต ถึง 512 บิต
2. สถิติการสุ่ม โดยบิตที่ได้จากกระบวนการสร้างกุญแจถ้าวิเคราะห์ทางสถิติแล้วจะต้อง

ให้มั่นใจได้ว่าจะไม่สามารถคาดเดากุญแจได้

เนื่องจากสภาพแวดล้อมในการส่งสัญญาณเมื่อมีสิ่งกีดขว้างจะทำให้เกิดการรบกวนต่อสัญญาณและเกิดการจางหายทำให้ค่าสัญญาณจะมีความแปรปรวน ดังนั้นเขาจึงใช้การประมาณค่าสัญญาณ เพื่อกำหนดระดับของสัญญาณที่ได้รับและเงื่อนไขการแปลงบิต 0 หรือ 1 ดังนี้

$$Q(x) = \begin{cases} 1 & if\ x > q+ \\ 0 & if\ x < q- \end{cases} \tag{2-2}$$

$Q(x)$ คือ ระดับ Quantization

x คือ ค่าสัญญาณที่นำมาพิจารณา

$q+$ คือ ค่ามากสุดของช่วงสัญญาณที่ประมาณได้

$q-$ คือ ค่าน้อยสุดของช่วงสัญญาณที่ประมาณได้

จากสมการถ้าค่าสัญญาณที่ได้รับมีค่ามากกว่า $q+$ จะกำหนดให้สัญญาณนั้นแปลงเป็นบิต 1 แต่ถ้าน้อยกว่า $q-$ จะกำหนดสัญญาณนั้นเป็นบิต 0 สามารถหาได้จากสมการที่ 2-1 ต่อไปจะเป็นการกล่าวถึงขั้นตอนการทำงาน Level-crossing algorithm มีดังนี้

ตารางที่ 2-1 ตัวแปรและความหมายของการสร้างกุญแจวิธีการ Level-crossing algorithm

สัญลักษณ์	ความหมาย
m	ขนาดความต่อเนื่องของการประมาณค่าสัญญาณ
L	รายการตำแหน่งที่ประกอบด้วย $L = \{l_1, l_2, \dots, l_n\}$
$\widetilde{L}$	รายการตำแหน่งที่ตัดบางรายการของ L ออก
i_{start}	ตำแหน่งเริ่มต้นที่บิตถัดไปจะเหมือนกันต่อเนื่องจำนวน m
i_{end}	ตำแหน่งสุดท้ายที่บิตก่อนหน้าจะเหมือนกันต่อเนื่องจำนวน m

กำหนดให้ Alice และ Bob เก็บค่าสัญญาณกันและกันโดยมีจำนวนสมาชิกที่เท่ากัน กำหนดค่า α แล้วคำนวณหาค่า $q+$ และ $q-$ ตามสมการที่ 2-1 และกำหนดค่าพารามิเตอร์ m

ขั้นตอนแรก: Alice พิจารณาสัญญาณที่ได้รับจาก Bob ถ้าค่าสัญญาณมากกว่า $q+$ ให้เป็นบิต 1 หรือถ้าน้อยกว่า $q-$ ให้เป็นบิต 0

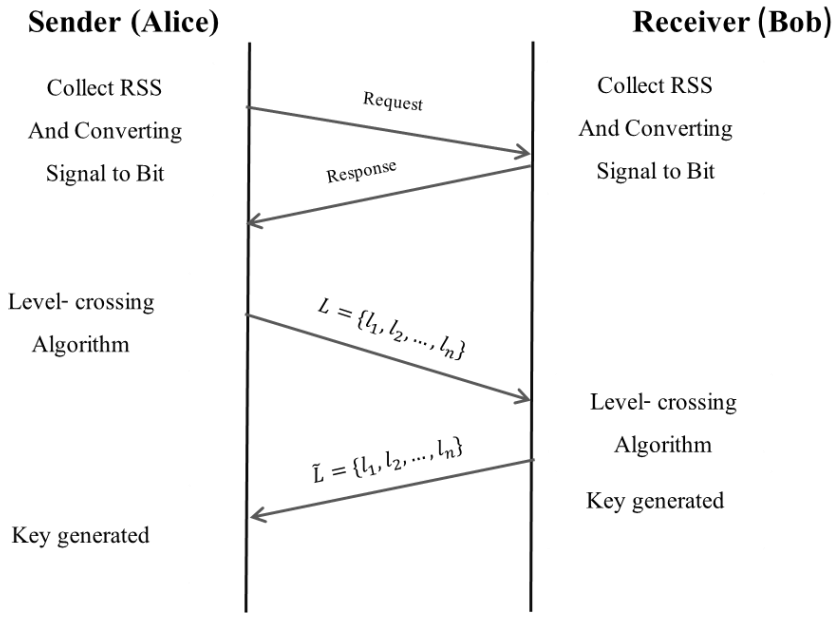
ขั้นตอนที่2: จากการประมาณค่าสัญญาณในขั้นตอนแรก Alice ทำการสร้างรายการตำแหน่ง L เพื่อส่งให้กับ Bob โดยหาจากตำแหน่งเริ่มต้น (i_{start}) ที่สามารถประมาณค่าสัญญาณแล้วมีค่ามากกว่า $q+$ หรือน้อยกว่า $q-$ ซ้ำกันต่อเนื่องจนมีความยาวเท่ากับ m เป็นตำแหน่งสุดท้าย (i_{end}) แล้วคำนวณหา รายการตำแหน่งจาก $l_i = \frac{(i_{start,i}+i_{end,i})}{2}, i = 1, 2, 3, \ldots, n$

ขั้นตอนที่3: Bob ตรวจสอบบิตที่แปลงจากสัญญาณที่ได้รับจาก Alice ตามรายการตำแหน่ง L ที่ได้รับจาก Alice แล้วตัดรายการที่ไม่มีบิตเกิดขึ้นทิ้งได้เป็นรายการตำแหน่ง $\tilde{L}$

ขั้นตอนที่4: หลังจาก Bob ตรวจสอบตามตำแหน่งที่ได้รับจาก Alice แล้ว Bob ส่งรายการตำแหน่ง $\tilde{L}$ ส่งให้กับ Alice

ขั้นตอนที่5: Alice และ Bob ใช้ลำดับบิตตามรายการตำแหน่ง $\tilde{L}$ ในการสร้างกุญแจ

จากการสร้างกุญแจด้วย Level crossing algorithm ในการประมาณค่าสัญญาณนั้นสิ่งที่จะทำให้ได้กุญแจที่ทั้งสองฝั่งสร้างแล้วเกิดบิตผิดพลาดน้อยลง โดยการกำหนดค่า m ยิ่งมีขนาดความต่อเนื่องมากยิ่งทำให้โอกาสการผิดพลาดน้อยลง แต่ก็จะส่งผลต่อการประมาณค่าสัญญาณที่จะต้องเพิ่มการเก็บค่าสัญญาณที่จะใช้ในการสร้างกุญแจมากขึ้น



ภาพที่ 2-6 กระบวนการสร้างกุญแจจากค่าความแรงสัญญาณโดยใช้ Level-crossing algorithm

จากภาพที่ 2-6 แสดงให้เห็นถึงการสื่อสารกันระหว่าง Alice และ Bob เริ่มต้นกระบวน การโดย Alice ส่งคำร้องขอไปยัง Bob เมื่อ Bob ได้รับคำร้องขอ Bob ส่งการตอบสนองกลับไปยัง Alice ทั้งสองทำการเก็บค่าความแรงสัญญาณของกันและกันไว้ ต่อมา Alice และ Bob ทำการสร้างกุญแจด้วยวิธีที่ Level-crossing algorithm

Hamming Code

ในปี 1950 Richard W. Hamming เป็นบุคคลแรก ๆ ที่พบ รหัสแก้ความผิดพลาด (Error-correcting code) เป็นวิธีลดปัญหาข้อมูลผิดพลาดที่เกิดเมื่อช่องสัญญาณถูกรบกวนระหว่างการส่ง โดยนำข้อความมาเข้ารหัสด้วยวิธีการ Hamming code ในงานวิจัยนี้เราสนใจความเป็น Linear block code ที่มีการสร้าง Parity-check ที่เป็นขั้นตอนสำคัญของวิธีการ Hamming code ซึ่งจะอธิบายรายละเอียดดังต่อไปนี้

Shu and Costello (2004) เริ่มต้นกระบวน Hamming code โดยการแบ่งข้อความอินพุตเป็นบล็อก (Message block, $\mathbb{m}$) ที่ประกอบด้วยข้อมูลไบนารี่จำนวน k ที่สามารถสร้างรูปแบบข้อความต่างกันทั้งหมดจำนวน 2^k แบบ สำหรับการเข้ารหัสจะทำการแปลงข้อความอินพุตแต่ละบล็อก $\mathbb{m}$ ไปเป็น $\mathbb{v}$ หรือเรียกว่า Codeword ของ $\mathbb{m}$ มีความยาวเท่ากับ n โดยที่ $n > k$ ดังนั้น จะได้ลักษณะที่เป็นไปได้ของ Codeword มีจำนวน 2^k แบบ ซึ่งเซตของ 2^k Codewords เรียกว่า block code โดยข้อความแต่ละบล็อก $\mathbb{m}$ จะมี Codeword ของตัวเอง สาเหตุที่ Block code น่าสนใจนั้นเพราะ linear ที่สามารถนำไปใช้ในทางปฏิบัติเพื่อลดความซับซ้อนในการเข้ารหัสได้

เมื่อแบ่งข้อความเป็นบล็อกแล้ว เราสามารถเข้ารหัสข้อความแต่ละบล็อกโดยการคูณข้อความบล็อก $\mathbb{m}$ กับ ค่า Matrix generator $\mathbb{G}$ จะได้ Codeword ของข้อความนั้น

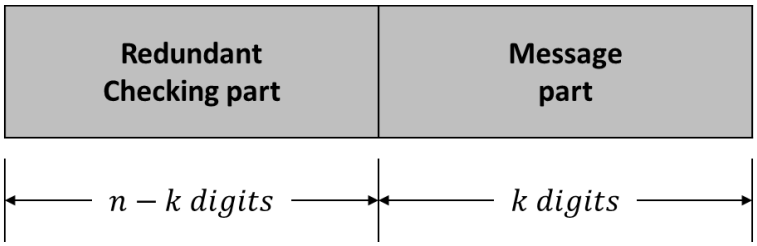
$$\mathbb{v} = \mathbb{m} \cdot \mathbb{G} \tag{2-3}$$

จากตารางที่ 2-2 เป็นรูปแบบ Codeword ที่เป็นไปได้ทั้งหมดเมื่อคูณกับ Matrix generator $\mathbb{G}$ โดย Codeword มีความยาว $n = 7$ และข้อความบล็อกความยาว $k = 4$

$$\mathbb{G} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \mathbb{g}_2 \\ \mathbb{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2-4}$$

ตารางที่ 2-2 Linear block code with k = 4 and n = 7

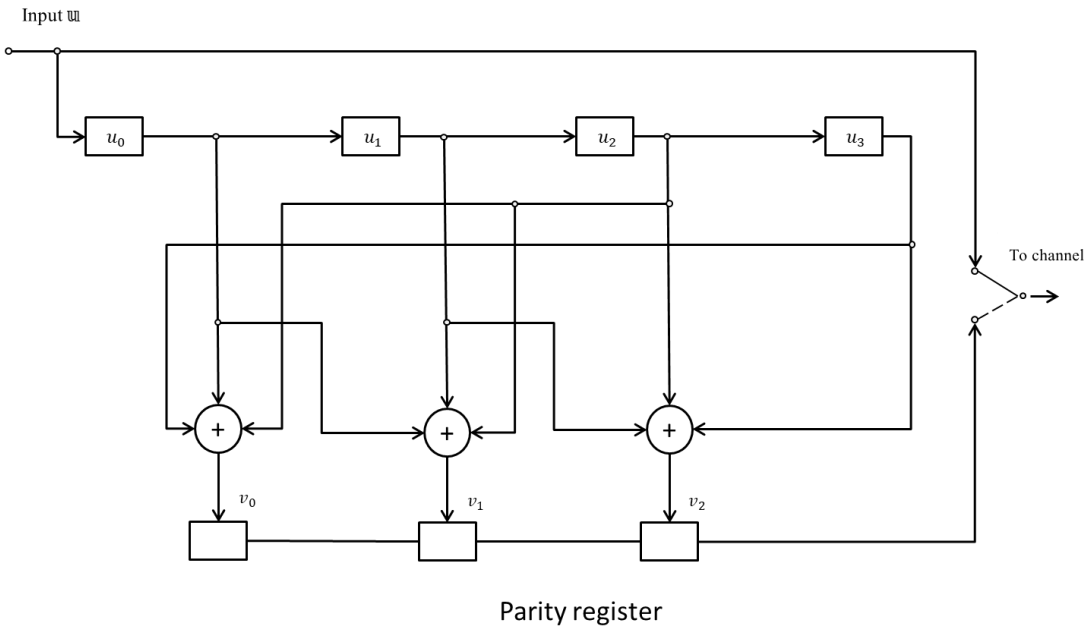
Messages	Codewords
(0 0 0 0)	(0 0 0 0 0 0 0)
(1 0 0 0)	(1 1 0 1 0 0 0)
(0 1 0 0)	(0 1 1 0 1 0 0)
(1 1 0 0)	(1 0 1 1 1 0 0)
(0 0 1 0)	(1 1 1 0 0 1 0)
(1 0 1 0)	(0 0 1 1 0 1 0)
(0 1 1 0)	(1 0 0 0 1 1 0)
(1 1 1 0)	(0 1 0 1 1 1 0)
(0 0 0 1)	(1 0 1 0 0 0 1)
(1 0 0 1)	(0 1 1 1 0 0 1)
(0 1 0 1)	(1 1 0 0 1 0 1)
(1 1 0 1)	(0 0 0 1 1 0 1)
(0 0 1 1)	(0 1 0 0 0 1 1)
(1 0 1 1)	(1 0 0 1 0 1 1)
(0 1 1 1)	(0 0 1 0 1 1 1)
(1 1 1 1)	(1 1 1 1 1 1 1)



ภาพที่ 2-7 รูปแบบระบบของ Codewords (Systematic format of codewords)

จากภาพที่ 2-7 แสดงรูปแบบของ Codewords ที่สร้างนั้นแบ่งเป็นสองส่วน คือ ส่วนของข้อความที่ไม่มีการเปลี่ยนแปลงความยาว k ที่อยู่ด้านขวาสุดของภาพและส่วนของ Redundant

checking part ด้านซ้ายสุดของภาพที่มีความยาวเท่ากับ $n-k$ ซึ่งเป็น Parity-check ที่มาจากผลรวมเชิงเส้นของข้อความด้านขวา



ภาพที่ 2-8 Encoding circuit for the (7, 4) systematic code

จากภาพที่ 2-8 อธิบายการเข้ารหัสเมื่อข้อความ $\mathfrak{m} = (u_0, u_1, \cdots, u_{k-1})$ เป็นอินพุตที่เลื่อนเข้ามาในระบบ ซึ่งจะสามารถหาค่า v_0, v_1, v_2 ได้จากความเชื่อมโยงกันของข้อความอินพุตตามสมการที่ 2-3

เนื่องจากเราต้องการเพียงส่วนของ Redundant checking part ด้านซ้ายสุดของภาพที่ 2-7 มาช่วยในการตรวจจับความผิดพลาดของบิตกุญแจที่ได้จากการสร้างบิตกุญแจโดย Level-crossing algorithm ในงานวิจัยเราเรียกวิธีตรวจจับความผิดพลาดของบิตกุญแจนี้ว่า Hamming-code generated redundancy: HGR ซึ่งขั้นตอนการออกแบบการทำงานจะกล่าวในบทต่อไป

ความสุ่ม

Bassham III et al. (2010) เมื่อได้สร้างกุญแจขึ้นมาแล้วนั้นสิ่งที่จะยืนยันได้ว่ากุญแจที่สร้างขึ้นสามารถนำมาใช้เข้ารหัสและมีความน่าเชื่อถือมากเพียงพอสำหรับจะไม่ทำให้เกิดการคาดเดากุญแจได้จึงจำเป็นต้องมีการทดสอบความสุ่ม ซึ่งใน 2010 สถาบันมาตรฐานและเทคโนโลยีแห่งชาติสหรัฐอเมริกา (NIST) ได้กำหนดมาตรฐานในการทดสอบความสุ่มและแนะนำค่าที่

เหมาะสมในการนำไปใช้งานด้านการเข้ารหัส สำหรับประเภทที่เป็นพื้นฐานในการสร้างค่าสุ่มมี 2 ประเภท

1. Random number generators (RNGs) เป็นการสร้างค่าสุ่มจากการใช้แหล่งยุ่งเหยิง (Entropy source) กับฟังก์ชันบางอย่างเพื่อให้ได้ค่าความเป็นสุ่ม (Randomness) แหล่งความยุ่งเหยิงประกอบด้วยปริมาณทางกายภาพบางส่วน เช่น สัญญาณรบกวนในวงจรไฟฟ้า เวลาที่เกิดจากการขยับเมาส์ หรือจากผลควอนตัมในเซมิคอนดักเตอร์ เป็นต้น ซึ่งถ้าเป้าหมายการใช้งานสำหรับเข้ารหัสนั้นค่าที่ได้จาก RNG จำเป็นต้องได้ค่าที่ไม่แน่นอน แต่อย่างไรก็ตามถ้าแหล่งที่มาของความยุ่งเหยิงเกิดจากแหล่งทางกายภาพบางส่วนที่อาจคาดการณ์ได้ ดังนั้น RNG ยังจำเป็นที่จะต้องมีการทดสอบทางสถิติเพื่อให้ได้ค่าความเป็นสุ่มที่มีคุณภาพ

2. Pseudorandom number generators (PRNGs) เป็นการสร้างตัวเลขสุ่มเทียมขึ้นมาแล้วทำการเก็บสถานะของการเกิดไว้ในค่าที่เรียกว่า seed เพื่อตรวจสอบว่าชุดตัวเลขสุ่มเทียมนี้จะวนซ้ำมาเกิดค่าเดิมอีกเมื่อไร

ในการทดสอบทางสถิติที่มีสมมติฐานหลัก (Null hypothesis) แบ่งออกเป็นสองแบบ คือ การยอมรับว่าเป็นสุ่ม (Random) และการปฏิเสธว่าเป็นสุ่ม (Non-random) จะใช้การคำนวณหาค่า P-value และเปรียบเทียบกับ ค่าระดับนัยสำคัญ (Level of significance) ที่สามารถกำหนดได้สองแบบ คือ แบบความน่าจะเป็นที่คาดว่า 1 ใน 1,000 หรือ 0.001 ค่า มีปฏิเสธว่าเป็นสุ่ม และ แบบความน่าจะเป็นที่คาดว่า 1 ใน 100 หรือ 0.01 ค่า มีปฏิเสธว่าเป็นสุ่ม ในงานวิจัยนี้ใช้ค่าระดับนัยสำคัญเป็น 0.01 ถ้าคำนวณได้ค่า P-value $<$ 0.01 หมายความว่า ลำดับตัวเลขของชุดตัวเลขที่ใช้ทดสอบนั้น ไม่เป็นสุ่ม (Non-random) แต่ถ้าคำนวณได้ P-value $\geq$ 0.01 หมายความว่าลำดับตัวเลขของชุดตัวเลขที่ใช้ทดสอบนั้น เป็นสุ่ม (Random)

จากที่กล่าวมา NIST ได้ให้ชุดทดสอบความสุ่มไว้ 15 แบบเพื่อประเมินคุณสมบัติการสุ่ม มีดังนี้

1. The frequency (Monobit) test
2. Frequency test within a block
3. The runs test
4. Tests for the longest-run-of-ones in a block
5. The binary matrix rank test
6. The discrete fourier transform (Spectral) test
7. The non-overlapping template matching test
8. The overlapping template matching test

9. Maurer's "Universal Statistical" test
10. The linear complexity test
11. The serial test
12. The approximate entropy test
13. The cumulative sums (Cusums) test
14. The random excursions test
15. The random excursions variant test

จากวิธีการทดสอบทั้งหมด 15 แบบนี้ งานวิจัยนี้เลือกใช้ วิธีการทดสอบแบบ The frequency (Monobit) test เพื่อทำการทดสอบที่มุ่งเน้นในเรื่องสัดส่วนของบิต 0 และ 1 จากลำดับของบิตกุญแจทั้งหมด และตามด้วยการทดสอบ The runs test เป็นการทดสอบความถี่ของการเปลี่ยนบิตจาก 0 ไป 1 หรือ 1 ไป 0 จากลำดับบิตกุญแจทั้งหมด สำหรับจำนวนบิตที่ใช้ทดสอบแนะนำอย่างน้อย 100 บิตขึ้นไป โดยทั้งสองวิธีมีขั้นตอนการคำนวณดังนี้

1. The frequency (Monobit) Test

ขั้นตอนแรก: ทำการแปลงบิตที่ต้องการทดสอบ โดยให้บิตที่ 0 เป็น -1 และบิต 1 เป็น +1 แล้วนำมาบวกกันตามลำดับอินพุต (ε) จะได้ผลรวมเป็น $S_n = X_1 + X_2 + \cdots + X_n$ เมื่อ $X_i = 2\varepsilon_i - 1$

ตัวอย่าง ถ้า $\varepsilon = 1011010101$, เมื่อ $n = 10$

$$S_n = 1 + (-1) + 1 + 1 + (-1) + 1 + (-1) + 1 + (-1) + 1 = 2 \quad (2\text{-}5)$$

S_n คือ ผลรวมของบิตกุญแจทั้งหมด

n คือ ความยาวบิตกุญแจที่ใช้ทดสอบ

ขั้นตอนที่สอง: คำนวณหาสถิติที่ใช้ทดสอบ (Test statistic) จากสมการ

$$S_{obs} = \frac{|S_n|}{\sqrt{n}} \quad (2\text{-}6)$$

s_{obs} คือ ค่าสัมบูรณ์ของผลรวมตามลำดับบิตทั้งหมดหารด้วยรากที่สองของจำนวนบิตทั้งหมด

ขั้นตอนที่สาม: คำนวณหาค่า P-value ได้จากสมการที่ 2-7

$$P - value = erfc\left(\frac{S_{obs}}{\sqrt{2}}\right) \quad (2\text{-}7)$$

เมื่อ $erfc$ คือ ฟังก์ชันคอมพลีเมนทารีผิดพลาด (Complementary error function) ตามสมการที่ 2-12

การทดสอบสามารถสังเกตได้จากค่า S_n ที่เป็นผลรวมของลำดับบิตกุญแจ ถ้ามีค่าเป็นจำนวนเต็มบวกมาก ๆ แสดงว่าสัดส่วนการเกิดบิต 1 มีมากกว่าบิต 0 แต่ถ้าเป็นตรงข้ามแสดงว่ามีบิต 0 มากกว่าบิต 1 ซึ่งต้องมาคำนวณหาค่าทางสถิติ s_{obs} แล้วหาค่า P-value ถ้า $P - value \geq 0.01$ แสดงว่าเป็นสุ่ม

2. The runs test

ขั้นตอนแรก: คำนวณสัดส่วนการเกิดบิต 0 และ 1 ก่อนการทดสอบ

$$\pi = \frac{\sum_j \varepsilon_j}{n} \tag{2-8}$$

π คือ pre-test proportion

ขั้นตอนที่สอง: ทำการตรวจสอบ ถ้า $\left|\pi - {}^1/_2\right| \geq \tau$ แสดงว่า ชุดบิตกุญแจนี้ไม่จำเป็นทำการทดสอบ The runs test โดยสามารถหาค่า τ ได้จากสมการที่ 2-9

$$\tau = \frac{2}{\sqrt{n}} \tag{2-9}$$

ขั้นตอนที่สาม: ถ้า $\left|\pi - {}^1/_2\right| < \tau$ ทำการคำนวณหาสถิติที่ใช้ทดสอบ (Test statistic) จากสมการ 2-10

$$V_n(obs) = \sum_{k=1}^{n-1} r(k) + 1 \tag{2-10}$$

เงื่อนไข ถ้า $\varepsilon_k = \varepsilon_{k+1}, r(k) = 0$ แต่ ถ้า $\varepsilon_k \neq \varepsilon_{k+1}, r(k) = 1$

ขั้นตอนที่สี่: คำนวณหาค่า P-value จากสมการดังต่อไปนี้

$$P - value = erfc\left(\frac{V_n(obs) - 2n\pi(1-\pi)}{2\sqrt{2n}\pi(1-\pi)}\right) \tag{2-11}$$

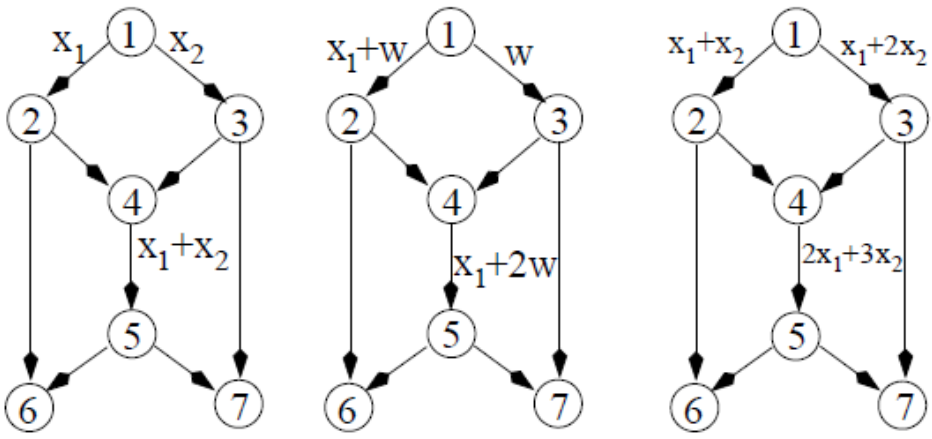
การทดสอบนี้สามารถสังเกตได้จากค่า $V_n(obs)$ ที่แสดงถึงความถี่การเปลี่ยนจากบิต 0 ไป 1 หรือตรงข้ามจากจำนวนบิตทั้งหมด ถ้าค่าน้อยมาก ๆ แสดงว่ามีบิตที่เหมือนกันอยู่ติดกันมาก ซึ่งค่าการเปลี่ยนแปลงที่ดีควรจะเป็นครึ่งหนึ่งของจำนวนบิตทั้งหมด

สำหรับฟังก์ชันคอมพลีเมนทารีผิดพลาด (Complementary error function) มาจากสมการที่ 2-12 ดังนี้

$$\operatorname{erfc}(z) = \frac{2}{\sqrt{\pi}} \int_z^{\infty} e^{-u^2}\, du \qquad (2\text{-}12)$$

การเข้ารหัสเครือข่าย

Ahlswede, Cai, Li and Yeung (2000) ได้นำเสนอการเข้ารหัสเครือข่าย (Network coding) เป็นเทคนิคที่สามารถเพิ่มอัตราการส่งข้อมูลสำหรับเครือข่ายที่มีผู้รับมากกว่าหนึ่ง (Multicast) โดยการเข้ารหัสเครือข่ายแทนการกำหนดเส้นทางพียงอย่างเดียวต่อมา Cai and Yeung (2002) ได้แสดงให้เห็นถึงความไม่ปลอดภัยในการส่งข้อมูลด้วยการเข้ารหัสเครือข่ายนั้นอาจเกิดการถูกดักฟังตามเส้นทางใดเส้นทางหนึ่งของเครือข่ายได้ จึงเสนอการผสมค่าสุ่มให้กับข้อความที่ต้องการส่งในแต่ละเส้นทาง เพื่อทำให้ข้อความที่ส่งมีความปลอดภัยมากขึ้น



(a) (b) (c)

ภาพที่ 2-9 เครือข่ายจำลองการส่งข้อความของ Bhattad and Narayanan (2005)

จากภาพที่ 2-9 แสดงให้เห็นถึงความปลอดภัยของเครือข่าย เมื่อพิจารณาเครือข่ายที่เป็นการส่งแบบ 2 ผู้รับและกำหนดให้สัญลักษณ์ข้อความที่ส่งเป็น x_1 และ x_2 จากภาพ (a) ถ้าสมมตว่ามีการดักฟังข้อความที่ขอบเส้นทางใดเส้นทางหนึ่งได้จากเครือข่ายนี้ เพื่อป้องกันการถูกดักฟัง ภาพ (b) ได้นำข้อความที่ต้องการส่งผสมกับค่าสุ่มหรือจากภาพแทนสัญลักษณ์ด้วย W ซึ่งถึงแม้จะถูกดักฟังที่ขอบเส้นทางใดก็ตามจะไม่สามารถรู้ข้อความได้ แต่ถึงแม้จะปลอดภัยเพียงใด ถ้าข้อความทั้งหมด คือ x_1 และ x_2 ที่ต้องการส่งวิธีนี้จะส่งได้ที่ละสัญลักษณ์เท่านั้น ส่วนภาพ (c) เป็นการนำสัญลักษณ์ทั้งสองผสมเข้าด้วยกันแล้วส่ง ถ้าเส้นทางใดเส้นทางหนึ่งถูกดักฟังได้ก็ไม่สามารถรู้ความหมายของข้อความได้ ซึ่งเรียกเทคนิคนี้ว่า Weakly secure network coding (Bhattad & Narayanan, 2005) ซึ่งเป็นวิธีการที่ผู้วิจัยสนใจในการนำไปประยุกต์ใช้เพื่อช่วยเพิ่มจำนวนกุญแจในการเข้ารหัสที่สร้างจากการเก็บค่าความแรงของสัญญาณโดยการเพิ่มจำนวนสมาชิกที่มากกว่าหนึ่ง ซึ่งจะกล่าวรายละเอียดในบทต่อไป

บทที่ 3
กระบวนการดำเนินงาน

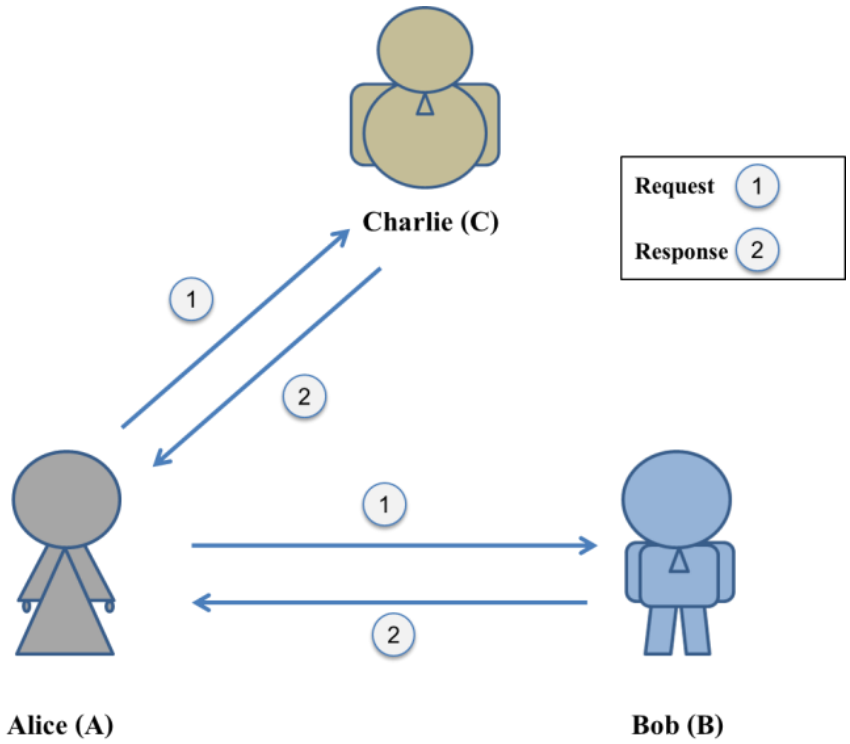
ในส่วนของการออกแบบกระบวนการดำเนินงานของระบบ ผู้วิจัยได้แบ่งออกเป็น 2 ส่วน ส่วนแรกเป็นการสร้างกุญแจจากค่าความแรงสัญญาณตามวิธีการ Level-crossing algorithm เพื่อให้ได้กุญแจที่เหมือนกันระหว่างภาครับและภาคส่งโดยไม่ต้องแลกเปลี่ยนกุญแจกัน หลังจากการสร้างกุญแจผู้วิจัยจะทำการเพิ่มอัตราการสร้างกุญแจระดับกายภาพจากค่าความแรงสัญญาณโดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัยมาช่วยในการส่งกุญแจที่จะใช้สำหรับเพิ่มอัตราการสร้างกุญแจ ซึ่งรายละเอียดขั้นตอนการทำงานทั้งหมดมีดังนี้

กำหนดให้ในเครือข่ายเดียวกันมี Alice Bob และ Charlie โดย Alice ต้องการส่งข้อความเข้ารหัสให้ Bob และใช้กุญแจที่ Charlie มีนำมาเพิ่มอัตราการสร้างกุญแจระดับกายภาพจากค่าความแรงสัญญาณโดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัย ดังนั้น Alice สามารถสร้างกุญแจเข้ารหัสและ Bob สามารถสร้างกุญแจที่ใช้ถอดรหัสได้ตามขั้นตอนต่อไปนี้

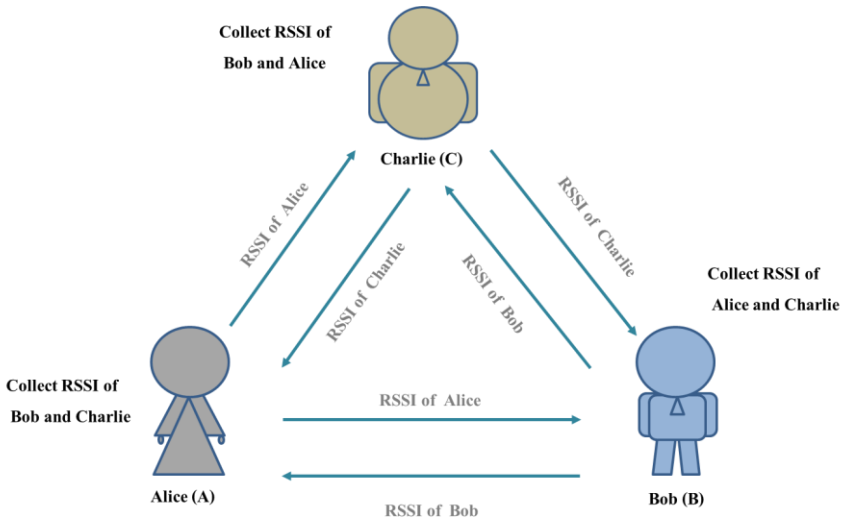
ขั้นตอนการสร้างกุญแจจากค่าความแรงสัญญาณ

กำหนดให้ทุกคนในเครือข่ายทำการสร้างกุญแจจากค่าความแรงของสัญญาณที่ได้รับขั้นตอนดังต่อไปนี้

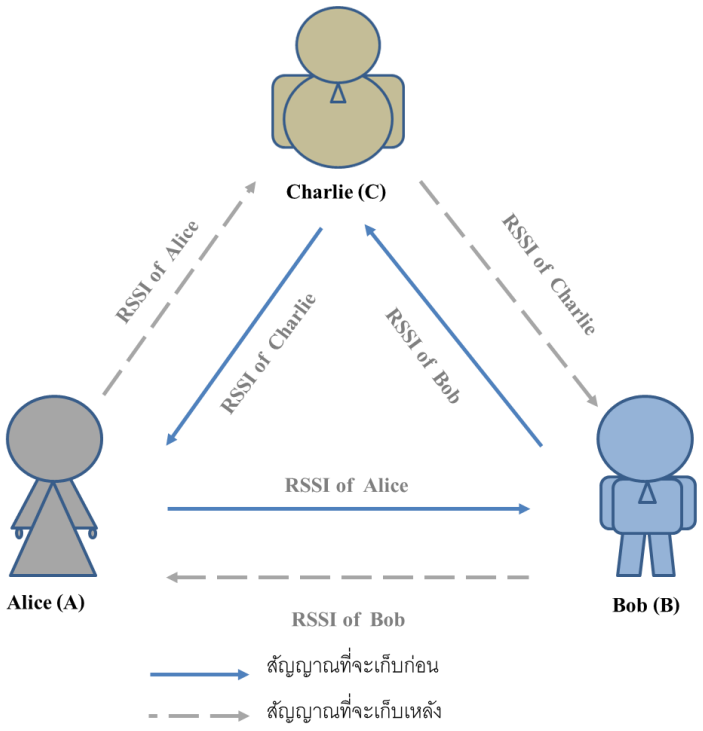
1. Alice ส่งคำร้องไปยัง Bob และ Charlie เพื่อเริ่มต้นกระบวนการสร้างกุญแจ
2. เมื่อ Bob และ Charlie ได้รับคำร้องขอจาก Alice แล้ว Bob และ Charlie ส่งการตอบสนองไปยัง Alice ดังภาพที่ 3-1
3. หลังจากดำเนินการขั้นตอนที่ 1 และ 2 เรียบร้อยแล้ว Alice Bob และ Charlie ทำการเก็บค่าความแรงสัญญาณของกันและกันตามจำนวนที่เพียงพอต่อการสร้างกุญแจ ซึ่งมีรูปแบบการเก็บค่าความแรงสัญญาณดังภาพที่ 3-2 เป็นดังนี้



ภาพที่ 3-1 การร้องขอของ Alice และการตอบสนองของ Bob และ Charlie



(a)



(b)

ภาพที่ 3-2 (a) Alice Bob และ Charlie การเก็บค่าความแรงสัญญาณของกันและกันในเวลาเดียว (b) Alice เก็บค่าความแรงสัญญาณของ Charlie ก่อนแล้วตามด้วยของ Bob ส่วน Bob เก็บค่าความแรงสัญญาณของ Alice ก่อนแล้วตามด้วยของ Charlie และ Charlie เก็บค่าความแรงสัญญาณของ Bob ก่อนตามด้วยของ Alice สลับกันจนครบจำนวนที่กำหนด

จากภาพที่ 3-2 ในรูปแบบ (a) ใช่เมื่อแต่ละคนสามารถเก็บค่าความแรงสัญญาณมากกว่าหนึ่งค่าในเวลาเดียวกัน แต่ถ้าไม่สามารถทำได้ในรูปแบบ (b) จะเป็นการเก็บค่าความแรงสัญญาณทีละค่าสลับการเก็บก่อนและหลังจนครบจำนวนค่าที่กำหนด โดยตำแหน่งที่เก็บค่าความแรงสัญญาณก่อนและหลังจะต้องอยู่ในตำแหน่งใกล้เคียงกันเพื่อให้ได้ค่าความแรงสัญญาณที่ได้รับเท่ากันจึงจะสามารถนำไปใช้ในการสร้างกุญแจในขั้นตอนต่อไปได้

Alice:

```
mode a
define n
for i=1 to n
        CollectRSSI_Bob(i) = RSSI_B(i) and
        CollectRSSI_Charlie(i) = RSSI_C(i)
i++
end
or mode b
for i =1 to n
        if CollectRSSI_Charlie(i)  = RSSI_C(i)
        else if CollectRSSI_Bob(i)= RSSI_B(i)
        i++
        end
end
```

Charlie:

```
mode a
define n
for i=1 to n
        CollectRSSI_Alice(i) = RSSI_A(i) and
        CollectRSSI_Bob(i) = RSSI_B(i)
i++
end
or mode b
for i =1 to n
        if CollectRSSI_Bob(i)  = RSSI_B(i)
        else if CollectRSSI_Alice(i)= RSSI_Alice(i)
        i++
        end
end
```

Bob:

```
mode a
define n
for i=1 to n
        CollectRSSI_Alice(i) = RSSI_A(i) and
        CollectRSSI_Charlie(i) = RSSI_C(i)
i++
end
or mode b
for i =1 to n
        if CollectRSS_Alice(i)  = RSSI_A(i)
        else if CollectRSSI_Charlie(i)= RSSI_C(i)
        i++
        end
end
```

ภาพที่ 3-3 Algorithm: การเก็บค่าความแรงสัญญาณแบบภาพที่ 3-2 (a) และ (b) จำนวน n ครั้ง

4. เมื่อเก็บค่าความแรงสัญญาณครบตามจำนวนที่ต้องการแล้วทุกคนทำการสร้างกุญแจโดยมีขั้นตอนการสร้างกุญแจตามวิธีการ Level-Crossing algorithm มีขั้นตอนดังนี้

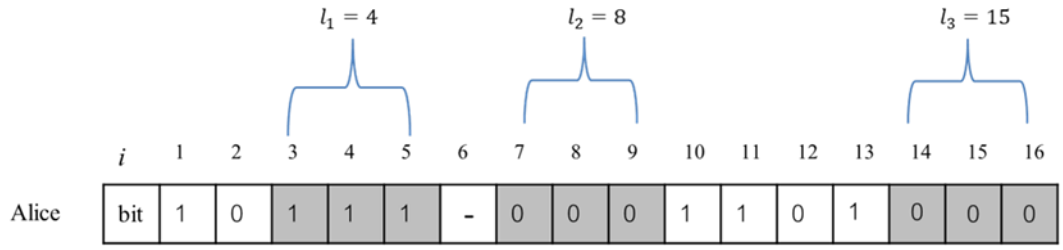
ตัวอย่างการดำเนินการสร้างกุญแจจากสัญญาณที่เก็บไว้ระหว่าง Alice และ Bob

4.1 Alice และ Bob คำนวณหาค่า $q \pm$ ตามสมการที่ 2-1 จากค่าความแรงสัญญาณที่เก็บไว้เพื่อใช้เป็นค่าอ้างอิงในการแปลงสัญญาณให้เป็นบิต

4.2 Alice และ Bob ทำการแปลงสัญญาณจากค่าความแรงสัญญาณที่เก็บไว้เป็นบิต โดยถ้ามากกว่า $q+$ให้เป็นบิต 1 หรือน้อยกว่า $q-$ ให้เป็นบิต 0 ทำซ้ำจนครบความยาวของจำนวน

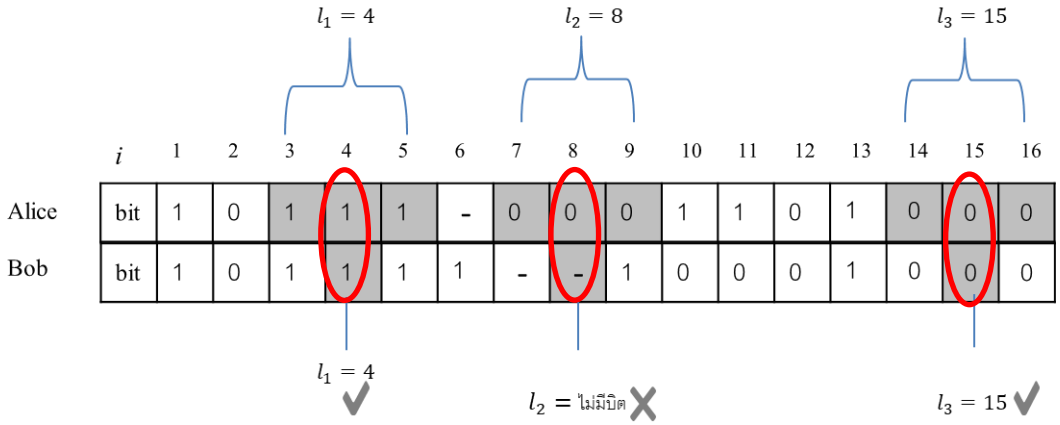
ของค่าความแรงสัญญาณที่เก็บไว้แต่ถ้าไม่มีบิต 0 หรือ 1 เกิดขึ้นเลย แสดงว่า การกำหนดค่า α อาจมากเกินไปให้ปรับลดลง

4.3 หลังจาก Alice แปลงสัญญาณเป็นบิต Alice ตรวจสอบว่ามีการเกิดบิตที่เหมือนกันตำแหน่งเริ่มต้น (i_{start}) ต่อเนื่องที่ความยาว m เป็นตำแหน่งสุดท้าย (i_{end}) แล้วคำนวณหารายการตำแหน่ง $l_i = \frac{(i_{start,i}+i_{end,i})}{2}$, i=1, 2, 3,…, n จะได้รายการตำแหน่ง L แล้วส่งให้กับ Bob แต่ถ้าไม่มีบิตเหมือนกันต่อเนื่องความยาวเท่ากับ m ให้ลดค่า m ลง



ภาพที่ 3-4 ตัวอย่างแสดงค่ารายการตำแหน่ง L = {4, 8, 15} เมื่อ m = 3 ของ Alice โดยหาจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จาก Bob จำนวน 16 ค่าและแปลงเป็นบิตแล้ว

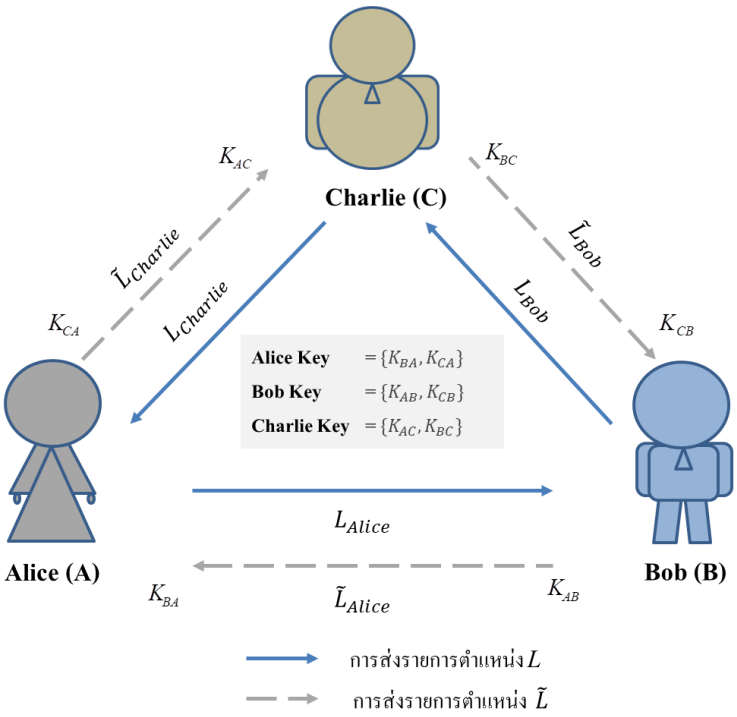
4.4 Bob ทำการตรวจสอบบิตสัญญาณของตนเองตามรายการตำแหน่ง (L) ที่ได้รับจาก Alice ว่ามีบิตที่ได้แปลงจากขั้นตอนที่ 4.2 เกิดขึ้นหรือไม่ ถ้ามีบิตเกิดขึ้นให้เก็บรายการตำแหน่งนั้นไว้แล้วทำซ้ำจนครบตามความยาวของจำนวนรอบที่เก็บสัญญาณไว้ ทำให้เหลือแค่รายการตำแหน่งที่มีบิตเกิดขึ้นของทั้งสองฝั่งเป็นรายการตำแหน่งใหม่ $\tilde{L}$ ส่งไปให้ Alice แต่ถ้าตรวจสอบจนครบแล้วไม่พบรายการตำแหน่งที่ตรงกัน แสดงว่า ชุดข้อมูลค่าความแรงสัญญาณที่เก็บนั้นไม่สามารถนำมาใช้สร้างกุญแจได้ให้ทิ้งไปแล้วแจ้งไปยัง Alice เพื่อเริ่มต้นขั้นตอนแรกใหม่



ภาพที่ 3-5 ตัวอย่าง Bob ตรวจสอบบิตตามรายการตำแหน่ง $\tilde{L}$ ={4, 15} ที่ได้รับจาก Alice

จากภาพที่ 3-5 จะเห็นได้ว่ารายการตำแหน่ง $\tilde{L}$ จะประกอบด้วย l_1 และ l_3 ดังนั้น Bob จะส่ง $\tilde{L}$ ={4, 15} ให้ Alice ซึ่งเป็นรายการตำแหน่งที่ทั้ง Alice และ Bob มีบิตเกิดขึ้นจึงนำบิตตามลำดับรายการตำแหน่ง $\tilde{L}$ ไปใช้ในการสร้างกุญแจ

5. จากขั้นตอนการสร้างกุญแจระหว่าง Alice กับ Bob นั้น ในช่วงเวลาเดียวกัน Bob และ Charlie ก็ดำเนินการหารายการตำแหน่งโดย Bob จะสร้างรายการตำแหน่ง L_{Bob} จากค่าความแรงสัญญาณของ Charlie แล้วส่งให้ Charlie และ Charlie จะสร้าง $L_{Charlie}$ จากค่าความแรงสัญญาณของ Alice แล้วส่งให้ Alice และแต่ละคนจะตรวจสอบรายการตำแหน่งกับค่าความแรงสัญญาณที่ได้รับจากคนที่ส่งรายการตำแหน่งนั้นแล้วส่งรายการตำแหน่งที่ตัดบางรายการออก $\tilde{L}_{Alice}$ $\tilde{L}_{Bob}$ และ $\tilde{L}_{charlie}$ กลับไป ซึ่งการส่งตามลำดับที่กล่าวนี้จะทำให้สามารถสร้างกุญแจได้ในเวลาเดียวกันทั้งสามคนดังภาพที่ 3-6



ภาพที่ 3-6 การสร้างกุญแจจากค่าความแรงสัญญาณตามวิธีการ Level-crossing algorithm

จากภาพที่ 3-6 กุญแจที่สร้างจากค่าความแรงสัญญาณตามวิธีการ Level-crossing algorithm ของ Alice Bob และ Charlie มีดังนี้

K_{BA} และ K_{AB} ที่สร้างบิตกุญแจมีลำดับบิตตามรายการตำแหน่ง $\tilde{L}_{Alice}$ ซึ่งเป็นรายการตำแหน่งที่มีการตัดบางรายการของรายการตำแหน่งที่ Alice สร้างออกทำให้ได้บิตกุญแจชุดเดียวกัน

K_{CB} และ K_{BC} ที่สร้างบิตกุญแจมีลำดับบิตตามรายการตำแหน่ง $\tilde{L}_{Bob}$ ซึ่งเป็นรายการตำแหน่งที่มีการตัดบางรายการของรายการตำแหน่งที่ Bob สร้างออก

K_{CA} และ K_{AC} ที่สร้างบิตกุญแจมีลำดับบิตตามรายการตำแหน่ง $\tilde{L}_{charlie}$ ซึ่งเป็นรายการตำแหน่งที่มีการตัดบางรายการของรายการตำแหน่งที่ Charlie สร้างออก

เนื่องจากการสร้างกุญแจตามวิธี Level-crossing algorithm ที่ใช้ชุดรายการตำแหน่งที่ตรวจสอบแล้วว่ามีบิตเกิดขึ้นในรายการตำแหน่งเดียวกันและอยู่ในช่องสัญญาณเดียวกันนั้นเมื่อนำไปสร้างกุญแจจะได้กุญแจชุดเดียวกัน ดังนั้น K_{XY} ที่เป็นกุญแจที่สร้างจากค่าความแรงสัญญาณของ X ส่งไป Y จะเหมือนกันกับ K_{YX} ที่เป็นกุญแจที่สร้างจากค่าความแรงสัญญาณของ Y ส่งไป X ที่ช่วงเวลาเดียวกัน

Alice:

1.หา L_{Alice}
n =length(CollectRSSI_Bob)
for i = n-m
  if Bit_B(i)= Bit_B(i+1)...= Bit_B(i+(m-1))
    i_{end} = i+m-1
    $L_{Alice}(i_{end}) = (i+i_{end})/2$
  else
    L_{Alice} (i) = 0
    i = i+1
  end
end
2. Alice ส่ง L_{Alice} ให้ Bob
3. ตรวจ Bit_C ตามรายการตำแหน่ง $L_{Charlie}$ ที่ได้รับจาก Charlie
n = length(CollectRSSI_Charlie)
for i=n
  if $L_{Charlie}$ (i) >0
    if Bit_C(i) = 0 or 1
      $L'_{Charlie}$ (i) = $L_{Charlie}$ (i)
    end
  else
  i = i+1
  end
end
4. Alice ส่ง $L'_{Charlie}$ ให้ Charlie และ รับ L'_{Alice} จาก Bob
5 Alice ใช้บิตตามรายการตำแหน่ง L'_{Alice} สร้างกุญแจ K_{BA} และ
บิตตามรายการตำแหน่ง $L'_{Charlie}$ สร้างกุญแจ K_{CA}

Charlie:

1.หา $L_{Charlie}$
n =length(CollectRSSI_Alice)
for i = n-m
  if Bit_A(i)= Bit_A(i+1)...= Bit_A(i+(m-1))
    i_{end} = i+m-1
    $L_{Charlie}(i_{end}) = (i+i_{end})/2$
  else
    $L_{Charlie}$ (i) = 0
    i = i+1
  end
end
2. Charlie ส่ง $L_{Charlie}$ ให้ Alice
3. ตรวจ Bit_B ตามรายการตำแหน่ง L_{Bob} ที่ได้รับจาก Bob
n = length(CollectRSSI_Bob)
for i=n
  if L_{Bob} (i) >0
    if Bit_C(i) = 0 or 1
      L'_{Bob} (i) = L_{Bob} (i)
    end
  else
  i = i+1
  end
end
4. Charlie ส่ง L'_{Bob} ให้ Bob และ รับ $L'_{Charlie}$ จาก Alice
5 Charlieใช้บิตตามรายการตำแหน่ง L'_{Bob} สร้างกุญแจ K_{BC} และ
บิตตามรายการตำแหน่ง $L'_{Charlie}$ สร้างกุญแจ K_{AC}

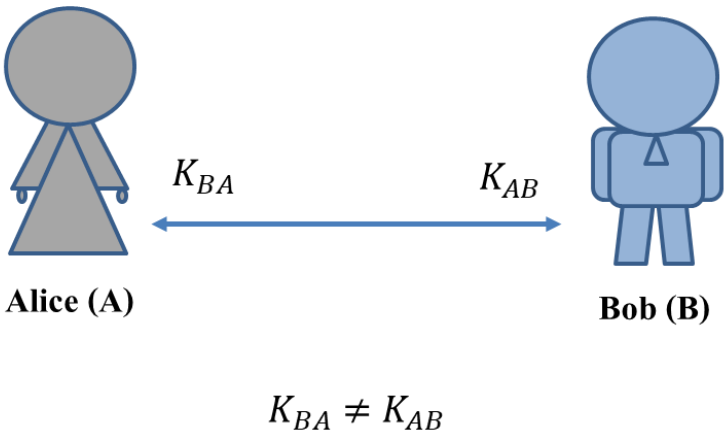
Bob:

1.หา L_{Bob}
n =length(CollectRSSI_Charlie)
for i = n-m
  if Bit_C(i)= Bit_C(i+1)...= Bit_C(i+(m-1))
    i_{end} = i+m-1
    $L_{Bob}(i_{end}) = (i+i_{end})/2$
  else
    L_{Bob} (i) = 0
    i = i+1
  end
end
2. Bob ส่ง L_{Bob} ให้ Charlie
3. ตรวจ Bit_A ตามรายการตำแหน่ง L_{Alice} ที่ได้รับจาก Alice
n = length(CollectRSSI_Alice)
for i=n
  if L_{Alice} (i) >0
    if Bit_C(i) = 0 or 1
      L'_{Alice} (i) = L_{Alice} (i)
    end
  else
  i = i+1
  end
end
4. Bob ส่ง L'_{Alice} ให้ Alice และ รับ L'_{Bob} จาก Charlie
5 Bobใช้บิตตามรายการตำแหน่ง L'_{Alice} สร้างกุญแจ K_{AB} และ
บิตตามรายการตำแหน่ง L'_{Bob} สร้างกุญแจ K_{CB}

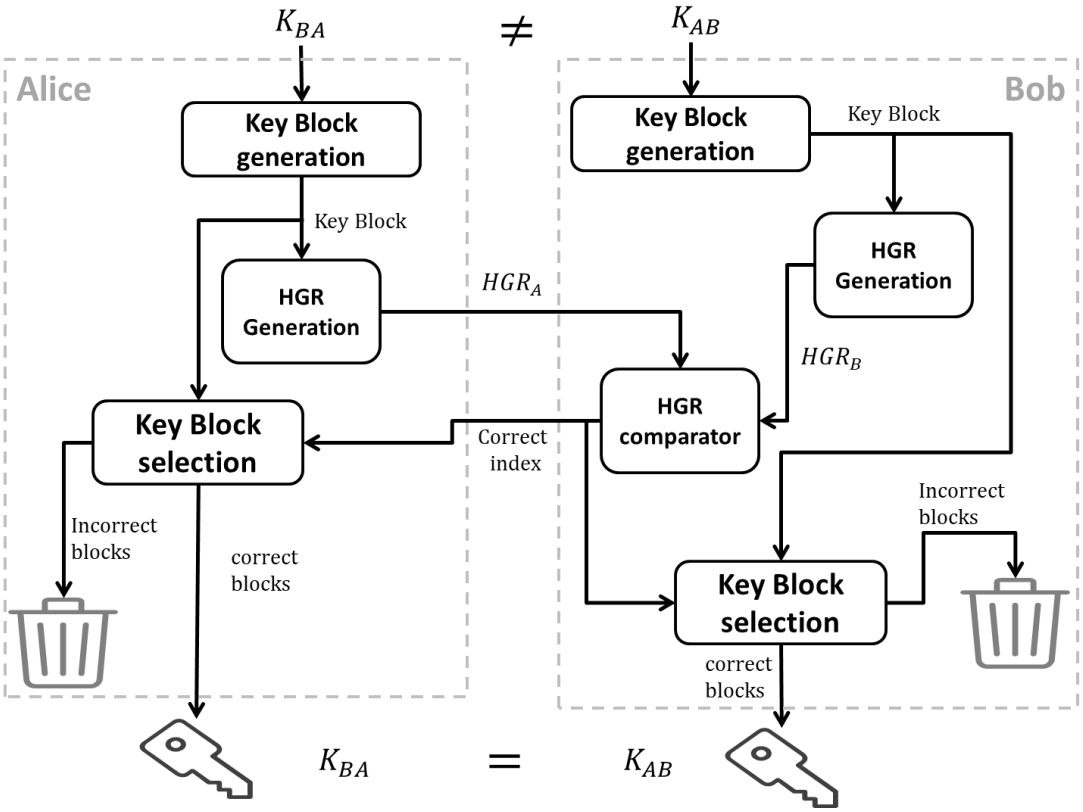
ภาพที่ 3-7 Algorithm: การสร้างกุญแจโดยวิธี Level-crossing ของ Alice Bob และ Charlie

การตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming-code Generated redundancy: HGR

ในงานวิจัยนี้เรามุ่งเน้นที่การสื่อสารของทั้งสองฝ่ายจะต้องได้กุญแจชุดเดียวกัน ซึ่งถ้าไม่ใช้ชุดเดียวกันจะไม่สามารถถอดรหัสได้ทำให้ต้องทิ้งกุญแจชุดนั้นไป จากการสร้างกุญแจจากค่าความแรงสัญญาณตามวิธีการ Level-crossing algorithm ถึงแม้จะสามารถกำหนดขนาดความต่อเนื่องของการประมาณค่าสัญญาณหรือค่า m เพื่อทำให้โอกาสการผิดพลาดบิตกุญแจน้อยลง อย่างไรก็ตาม ในกรณีบิตกุญแจที่ได้กำหนดค่า m ที่เหมาะสมแล้วแต่ยังเกิดข้อผิดพลาดบิตกุญแจไม่มากนักก็ยังต้องทิ้งชุดกุญแจนั้นไปทำให้อาจเสียเวลาในการเริ่มต้นเก็บค่า RSSI สำหรับการสร้างกุญแจชุดใหม่ ดังนั้นเราจึงเสนอ การตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming-code Generated redundancy: HGR



ภาพที่ 3-8 การสร้างกุญแจจากค่าความแรงสัญญาณตามวิธีการ Level-crossing algorithm กรณีกุญแจทั้งสองฝั่งไม่ตรงกัน



ภาพที่ 3-9 ขั้นตอนการตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming- code generated redundancy: HGR

จากภาพที่ 3-9 การตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming- code generated redundancy: HGR กับกุญแจ K_{BA} ของ Alice และ K_{AB} ของ Bob ที่ได้จากการสร้างกุญแจจากค่าความแรงสัญญาณตามวิธีการ Level-crossing algorithm มีรายละเอียดขั้นตอนดังต่อไปนี้

1. Alice และ Bob นำบิตกุญแจของตนแบ่งเป็นบล็อก
2. Alice นำบิตกุญแจของตนแต่ละบล็อกคำนวณสมการที่ 2-3 จะได้ Codewords ที่มีส่วนประกอบตามภาพที่ 2-7 ส่วนซ้ายของภาพ คือ Redundant-checking part ซึ่งเราจะใช้เป็น HGR_A ที่จะส่งให้กับ Bob
3. Bob นำบิตกุญแจของตนแต่ละบล็อกคำนวณหา HGR ตามแบบขั้นตอนที่ 2 จะได้ HGR_B แล้วนำไปเปรียบเทียบกับ HGR_A ที่ได้รับจาก Alice จะได้ค่า Correct index
4. Bob ส่ง Correct index ให้กับ Alice

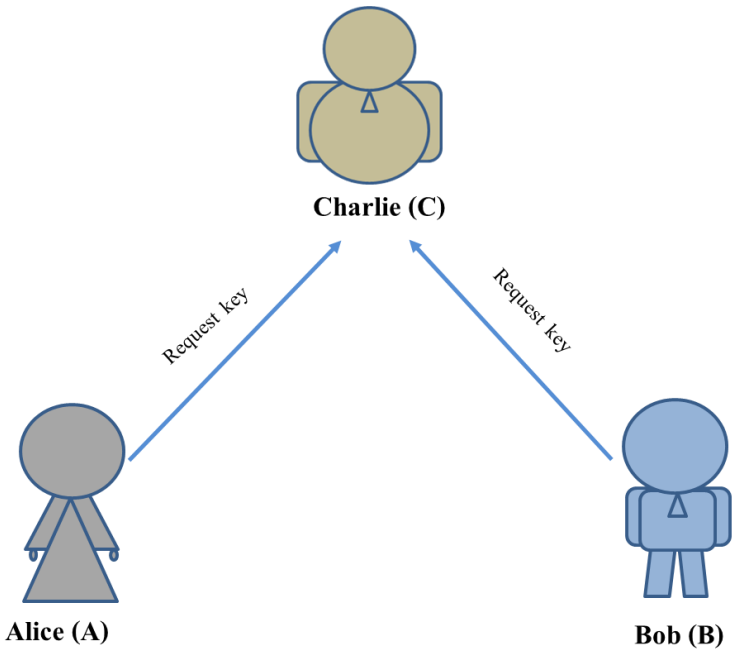
5. Alice และ Bob ใช้ค่า Correct index ในการเลือกบล็อกกุญแจ ถ้าถูกต้องจะเก็บไว้ ถ้าผิดจะทิ้งบล็อกกุญแจนั้นไป

เนื่องจากวิธีการนี้จะต้องแบ่งบล็อกกุญแจก่อนเข้ากระบวนการ ถ้าความยาวกุญแจที่จะนำมาเข้ากระบวนการ หากความยาวบล็อกไม่ลงตัวจะทำให้ต้องทิ้งบล็อกสุดท้ายที่เป็นเศษที่เหลือ ดังนั้นควรกำหนดความยาวบิตกุญแจที่จะสร้างให้สามารถแบ่งบล็อกได้ลงตัวเพื่อไม่ให้สูญเสียบิตโดยไม่จำเป็น

ขั้นตอนเพิ่มอัตราการสร้างกุญแจโดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัย

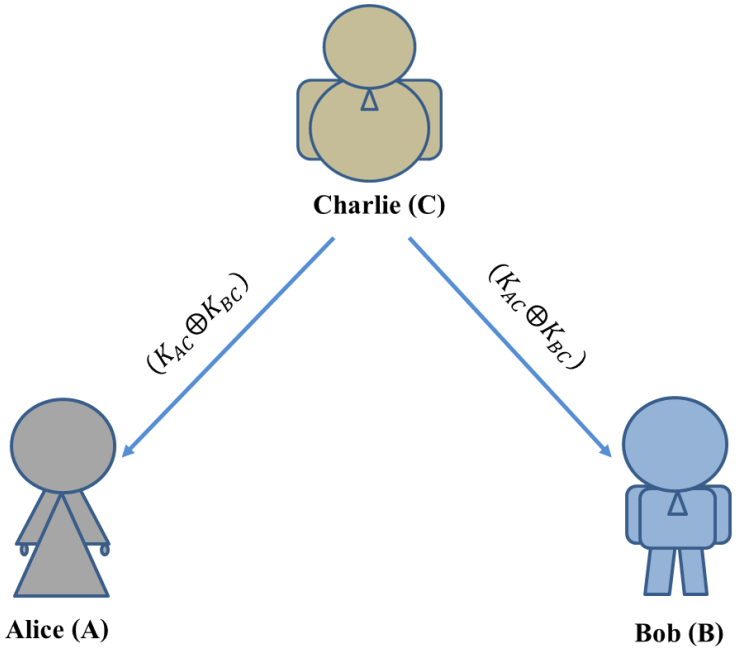
กำหนด Alice และ Bob ต้องการใช้ประโยชน์จากกุญแจที่ Charlie มี เพื่อเพิ่มอัตราการสร้างกุญแจระดับกายภาพจากค่าความแรงสัญญาณโดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัยมีกระบวนการทำงานดังนี้

1. Alice และ Bob ทำการส่งคำร้องขอกุญแจไปยัง Charlie เพื่อเริ่มต้นกระบวนการเพิ่มกุญแจ



ภาพที่ 3-10 เริ่มต้นกระบวนการเพิ่มกุญแจโดย Alice และ Bob ส่งคำร้องขอเพิ่มกุญแจไปยัง Charlie

2. เมื่อ Charlie ได้รับคำร้องขอกุญแจ Charlie จะนำกุญแจที่มีเข้ารหัสด้วยกระบวนการเอ็กซ์คลูซีฟ ออร์ (Exclusive OR, XOR) แล้วจึงทำการส่งกุญแจ $(K_{AC} \oplus K_{BC})$ ของตนให้ทั้ง Alice และ Bob



ภาพที่ 3-11 การส่งกุญแจ $K_{AC} \oplus K_{BC}$ ของ Charlie ไปยัง Alice และ Bob

3. เมื่อ Alice ได้รับกุญแจ $(K_{AC} \oplus K_{BC})$ จาก Charlie แล้ว Alice ทำการถอดรหัสหากุญแจ K_{BC} ซึ่งเป็นกุญแจที่มีเพียง Charlie และ Bob เท่านั้นที่รู้ ได้จากการใช้กุญแจที่ Alice สร้างจากค่าความแรงสัญญาณของ Charlie (K_{CA}) มา XOR กับ $(K_{AC} \oplus K_{BC})$ ที่ได้รับจาก Charlie

$$(K_{AC} \oplus K_{BC}) \oplus K_{CA} = K_{BC} \tag{3-1}$$

4. เมื่อ Alice ได้กุญแจใหม่เพิ่มเข้ามา Alice สามารถสร้างกุญแจเข้ารหัสโดยกุญแจที่มีอยู่ทั้งหมด ดังนี้

$$K_{Encrypt} = \{K_{BA}, K_{CA}, K_{BC}\} \tag{3-2}$$

5. Alice เข้ารหัสข้อความด้วยกุญแจเข้ารหัส ($K_{Encrypt}$) แล้วส่งข้อความที่เข้ารหัสให้กับ Bob

6. ส่วน Bob เมื่อได้รับ $(K_{AC} \oplus K_{BC})$ จาก Charlie จากนั้น Bob ทำการถอดรหัสหากุญแจ K_{AC} ซึ่งเป็นกุญแจที่มีเพียง Charlie และ Alice เท่านั้นที่รู้ได้จากการใช้กุญแจที่ Bob สร้างจากค่าความแรงสัญญาณของ Charlie (K_{CB}) มา XOR กับ $(K_{AC} \oplus K_{BC})$ ที่ได้รับจาก Charlie

$$(K_{AC} \oplus K_{BC}) \oplus K_{CB} = K_{AC} \tag{3-3}$$

7. เมื่อ Bob ได้กุญแจใหม่เพิ่มเข้ามา Bob สามารถสร้างกุญแจถอดรหัสโดยกุญแจที่มีอยู่ทั้งหมด ดังนี้

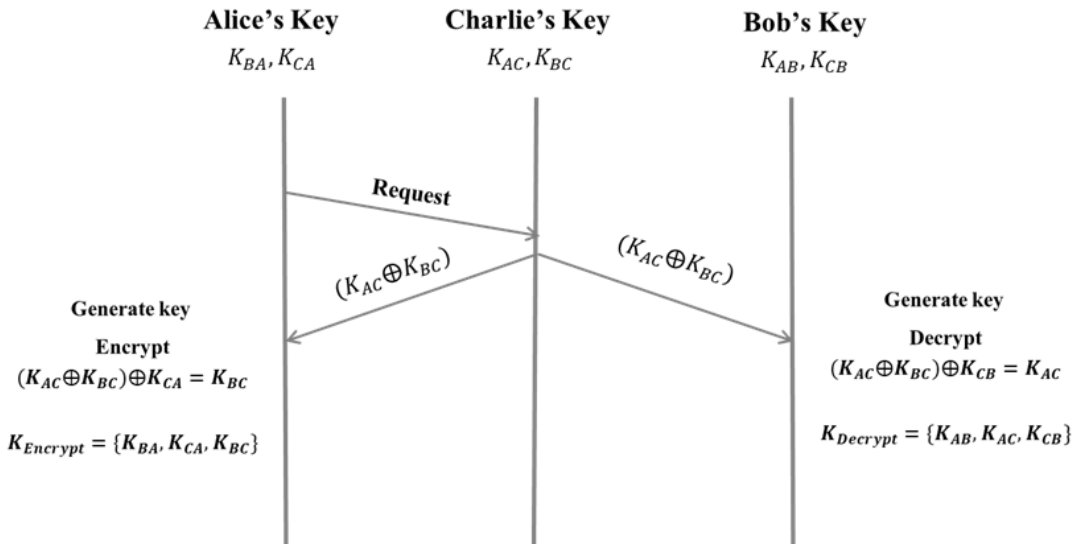
$$K_{Decrypt} = \{K_{AB}, K_{AC}, K_{CB}\} \tag{3-4}$$

8. Bob สามารถถอดรหัสข้อความเข้ารหัสของ Alice ด้วยกุญแจถอดรหัส ($K_{Decrypt}$) ที่สร้างขึ้น ซึ่งในตารางที่ 3-1 แสดงให้เห็นถึงความแตกต่างของจำนวนกุญแจเมื่อมีการเพิ่ม Charlie เข้ามาเป็น ดังนี้

ตารางที่ 3-1 เปรียบเทียบจำนวนกุญแจของ Alice และ Bob ระหว่างใช้โปรโตคอลที่ไม่มี Charlie และโปรโตคอลที่มี Charlie โดยใช้เวลาเท่ากัน

New protocol with Charlie	Alice's key	Bob's key
No	K_{BA}	K_{AB}
Yes	K_{BA}, K_{CA}, K_{BC}	K_{AB}, K_{CB}, K_{AC}

จากตารางที่ 3-1 จะเห็นได้ว่า ถ้า Alice ไม่ใช้โปรโตคอลที่มี Charlie จะมีเพียงชุดกุญแจ K_{BA} เท่านั้นที่สามารถนำไปทำการเข้ารหัสข้อความแล้วส่งข้อความลับให้ Bob โดยที่ Bob สามารถถอดรหัสด้วยกุญแจ K_{AB} ที่เป็นชุดเดียวกันได้ ซึ่งถ้าต้องการกุญแจที่ยาวเป็นสามเท่าจะต้องใช้เวลาเพิ่มขึ้นเป็นสามเท่า



ภาพที่ 3-12 ภาพรวมขั้นตอนเพิ่มอัตราการสร้างกุญแจระหว่าง Alice กับ Bob โดยใช้การเข้ารหัสเครือข่ายแบบปลอดภัย

งานวิจัยนี้ได้เสนอการเก็บค่าความแรงสัญญาณที่ได้รับ (RSSI) ที่มี 2 รูปแบบที่แสดงไว้ในภาพที่ 3-2 และใช้การเข้ารหัสเครือข่ายแบบปลอดภัยที่แสดงขั้นตอนการเพิ่มกุญแจโดยใช้โปรโตคอลที่มี Charlie ช่วยส่งกุญแจดังภาพที่ 3-10 ที่ทำให้สามารถสร้างกุญแจความยาวเพิ่มขึ้นเป็นสามเท่า เมื่อเก็บค่าความแรงสัญญาณที่ได้รับ (RSSI) ตามแบบภาพที่ 3-2 (a) และ ความยาวกุญแจเพิ่มขึ้นเป็นสามเท่าแต่ใช้เวลาเพิ่มขึ้นเพียง 2 เท่า เมื่อเก็บค่าความแรงสัญญาณที่ได้รับ (RSSI) ตามแบบภาพที่ 3-2 (b) และการส่งกุญแจด้วยการเข้ารหัสเครือข่ายความปลอดภัยดังภาพที่ 3-10 ยังช่วยป้องกันความเสี่ยงการถูกดักฟัง โดย Charlie ใช้กระบวนการ XOR เพื่อส่ง $(K_{CA} \oplus K_{BC})$ ให้กับ Alice และ Bob ซึ่ง Alice สามารถถอดรหัสหา K_{BC} ได้ตามสมการที่ 3-1 และ Bob สามารถถอดรหัสหา K_{CA} ได้ตามสมการที่ 3-3 หากเกิดการดักฟัง $(K_{CA} \oplus K_{BC})$ ผู้ดักฟังก็จะไม่ทราบข้อมูลบิตกุญแจใด ๆ โดยตรงแม้แต่บิตเดียว แต่หากผู้ดักฟังบังเอิญเดา K_{CA} ได้จะทำให้สามารถถอดรหัสหา K_{BC} ได้ เราจึงเรียกความปลอดภัยในการเข้ารหัสเครือข่ายของ K_{CA} และ K_{BC} ว่าเป็นอย่างอ่อน (Weakly secure network coding) สำหรับตัว Charlie นั้นจะรู้กุญแจเป็นจำนวน $^2/_3$ ของทั้งหมด แต่ไม่มีผลกระทบเนื่องจากเรากำหนดให้เป็นโหนดที่เชื่อถือได้

สำหรับการเรียงกุญแจเข้ารหัสของ Alice จากสมการที่ 3-2 นั้นเราใช้จากกุญแจที่สร้างจาก Alice กับ Bob ก่อนตามด้วยกุญแจที่ Alice สร้างกับ Charlie และกุญแจที่ได้เพิ่มมาจาก Charlie

ปิดท้าย ส่วน Bob เรียงกุญแจถอดรหัส ในสมการที่ 3-4 เนื่องจากกุญแจที่ใช้สำหรับการเข้ารหัสและถอดรหัสเป็นกุญแจสมมาตรที่กุญแจเข้ารหัสและถอดรหัสจะต้องเหมือนกันดังนั้น เราจึงเรียงกุญแจเพื่อให้ได้กุญแจที่เหมือน Alice โดย เริ่มจากกุญแจที่สร้างจาก Bob กับ Alice ตามด้วยกุญแจที่ได้เพิ่มมาจาก Charlie และกุญแจที่สร้างจาก Bob กับ Charlie ปิดท้าย ก็จะได้กุญแจที่ใช้สำหรับถอดรหัส การจัดเรียงสามารถเปลี่ยนตำแหน่งได้แต่ต้องให้แน่ใจว่าทั้งสองฝ่ายจะได้กุญแจชุดเดียวกัน

บทที่ 4

การทดลองและผลการทดลอง

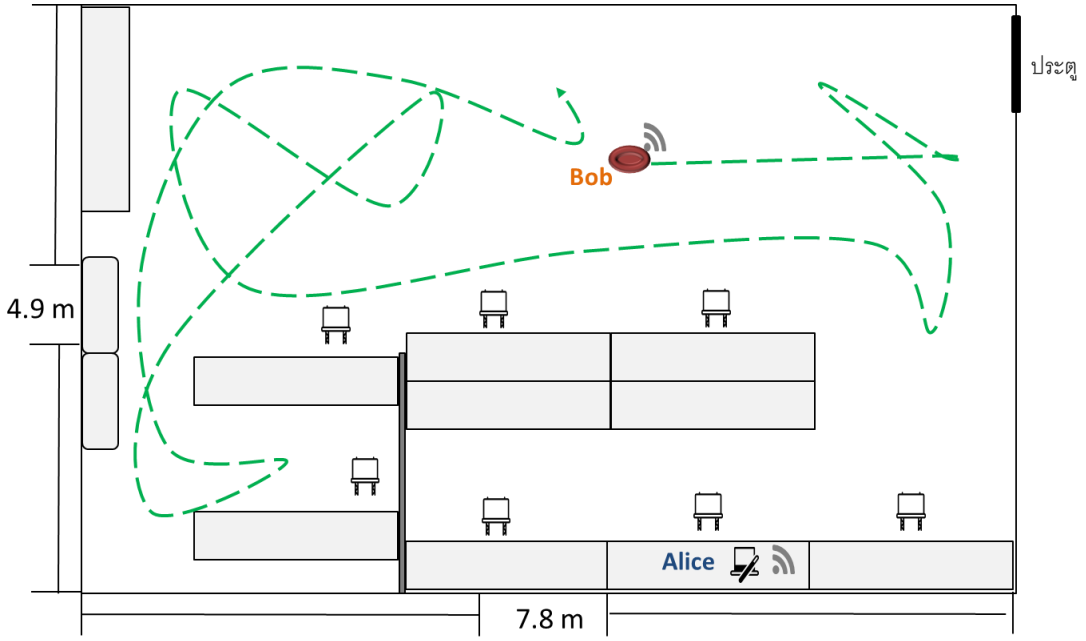
การทดลองนี้ใช้ เทคโนโลยี Wi-Fi ที่ช่วงความถี่ 2.4 GHz จาก ESP8266 รองรับมาตรฐานโปรโตคอล IEEE 802.11 b/g/n สามารถตั้งค่าโหมดการทำงาน Wi-Fi ได้ 3 โหมด คือ WIFI_AP (Access point) สำหรับปล่อยสัญญาณ Wi-Fi, WIFI_STA (Stations) ทำหน้าที่เป็นลูกข่ายในการรับส่งข้อมูลในเครือข่าย (Client) และ WIFI_AP_STA เป็นโหมดที่สามารถทำงานเป็นทั้งโหมด STA และ AP ได้พร้อมกัน ในงานวิจัยนี้เราใช้ ESP8266 จำนวน 2 ตัว ทำงานในโหมด WIFI_AP_STA เพื่อให้ทั้งสองตัวสามารถวัดค่าความแรงสัญญาณของกันและกันได้ในเวลาเดียวกัน

ภาพที่ 4-1 NodeMCU (ESP8266)

การเก็บค่าความแรงสัญญาณ (RSSI)

เราจำลองสถานการณ์ให้ ESP8266 หนึ่งตัวเป็น Alice ทำหน้าที่ปล่อยสัญญาณและสแกนหาสัญญาณของ Bob ซึ่งในช่วงเวลาเดียวกัน Bob เป็น ESP8266 อีกหนึ่งตัว ก็ทำหน้าที่ปล่อยสัญญาณและสแกนหาสัญญาณของ Alice และกำหนดให้ทั้งสองทำการเก็บค่าความแรงสัญญาณ (RSSI) จำนวน 8,000 ค่า ในสภาพแวดล้อมแบบปิดที่อยู่ในห้องเดียวกัน สำหรับ ESP8266 ใช้ความเร็วในการสแกนหาสัญญาณ Wi-Fi และเก็บค่า เฉลี่ยอยู่ที่ 0.625 ค่าต่อวินาที จากการจำลอง

สถานการณ์เพื่อป้องกันการคาดเดาสัญญาณ โดยใช้สถิติ เราจึงทำการกำหนดให้ Bob มีการเปลี่ยนตำแหน่งตลอดเวลาและไม่มีทิศทางแน่นอน โดยติด ESP8266 ไว้กับหุ่นยนต์ดูดฝุ่น ซึ่งหุ่นยนต์ดูดฝุ่นจะเปลี่ยนทิศทางเมื่อพบสิ่งกีดขวาง



ภาพที่ 4-2 สถานที่ใช้ในการทดลองการเก็บค่าความแรงสัญญาณระหว่าง Alice อยู่นิ่ง และ Bob เคลื่อนที่แบบไม่มีทิศทางแน่นอน

ผลทดลองการสร้างกุญแจจากค่าความแรงสัญญาณโดยวิธี Level-crossing algorithm

เพื่อให้มั่นใจว่าการทดลองของเราเป็นไปในทิศทางเดียวกันกับวรรณกรรมที่เราใช้อ้างอิง ในตารางที่ 4-1 เราไปทำการเปรียบเทียบผลการทดลองของเรากับงานวิจัยของ Mathur et al. (2008) เพื่อตรวจสอบอัตราส่วนกุญแจที่สร้างจากจำนวนค่าความแรงสัญญาณที่เก็บผลการทดลองเป็นดังนี้

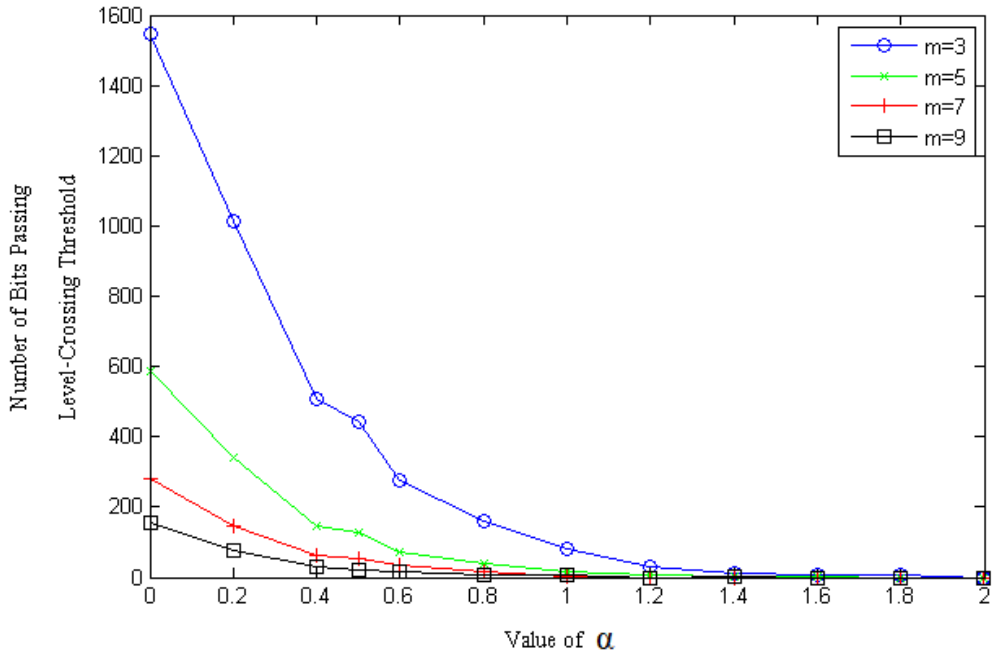
จากตารางที่ 4-1 เปรียบเทียบผลการสร้างกุญแจจากค่าความแรงสัญญาณ โดยวิธี Level-crossing algorithm ระหว่างผลการทดลองในงานวิจัยนี้กับผลการทดลองของ Mathur การทดลองของเรานั้นได้นำวิธี Level-Crossing algorithm มาใช้กับช่องสัญญาณ Wi-Fi โดยใช้ ESP8266 สองตัว เป็น Alice กับ Bob ให้มีการเก็บค่า RSSI จำนวน 8,000 ค่า และกำหนดค่า m เท่ากับ 3 สามารถสร้างบิตกุญแจได้ 443 บิต คิดเป็น 5.54% ซึ่งผลการทดลองของ Mathur ใช้ค่า m เท่ากับ 3 และเก็บค่า RSSI จำนวน 7,840 ค่า สร้างกุญแจบิตได้ 511 บิต คิดเป็น 6.52% จะเห็นว่าอัตราการสร้างบิต

ในการทดลองของเรานั้น เมื่อใช้เพียง Alice กับ Bob เหมือนกับการทดลองของ Mathur จะมีอัตราที่ต่ำกว่าซึ่งอาจเป็นผลมาจากความแม่นยำของอุปกรณ์รับส่งที่ใช้ รวมถึงลักษณะการเคลื่อนที่ของ Alice และ Bob ที่ต่างกัน รวมถึงสภาพแวดล้อมอื่น ๆ

อย่างไรก็ตาม หากมีการเพิ่มอัตราการสร้างกุญแจโดยใช้วิธีที่นำเสนอโดยมีรายละเอียดดังในบทที่ 3 จะทำให้สามารถเพิ่มอัตราการสร้างกุญแจของเราได้อีก กล่าวคือ เมื่อมีการเพิ่ม Charlie เข้ามาโดยอยู่กับที่เช่นเดียวกับ Alice หาก Alice สามารถสร้างกุญแจจากช่องสัญญาณระหว่าง Bob กับ Charlie ได้ในเวลาเดียวกัน และ Bob สามารถสร้างกุญแจจากช่องสัญญาณระหว่าง Alice กับ Charlie ได้ในเวลาเดียวกันแล้ว จะทำให้ในเวลาที่เท่ากัน เราสามารถสร้างบิตกุญแจเพิ่มขึ้นเป็นสามเท่าจากเดิมที่อัตรา 5.54% เป็น 16.62% แต่หากไม่สามารถทำสองสิ่งดังกล่าวในเวลาเดียวกันได้ด้วยข้อจำกัดของอุปกรณ์ที่ใช้ เราจะสามารถสร้างบิตกุญแจเพิ่มขึ้นสามเท่า โดยใช้เวลามากขึ้นเพียงสองเท่า อัตราจึงเพิ่มขึ้นเป็น 8.31% ซึ่งมากกว่าการทดลองของ Mathur

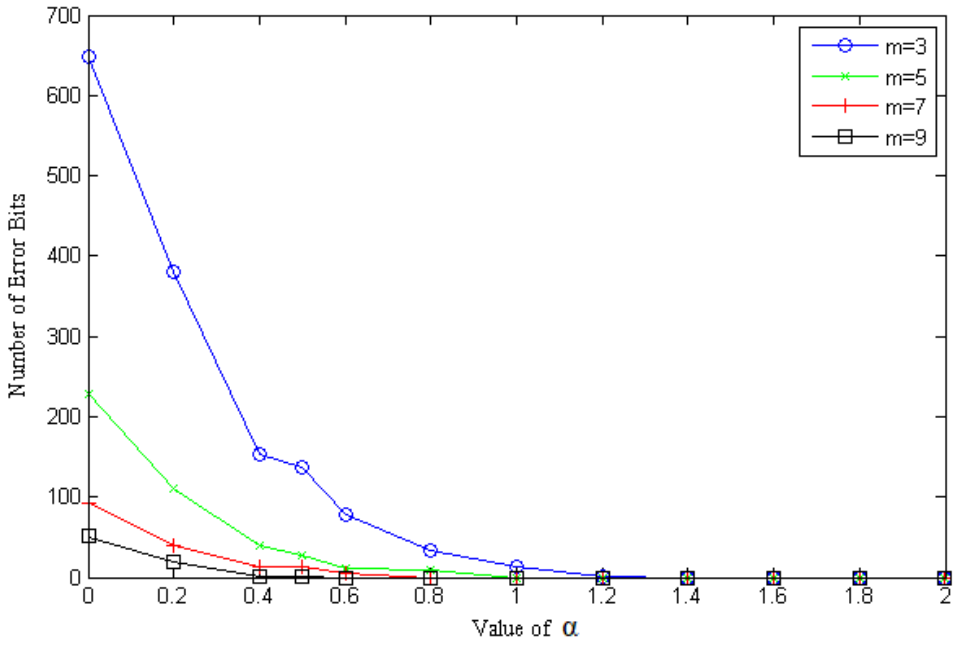
ตารางที่ 4-1 เปรียบเทียบผลการสร้างกุญแจจากค่าความแรงสัญญาณโดยวิธี Level-crossing algorithm ระหว่างจากผลการทดลองในงานวิจัยนี้กับของ Mathur et al. (2008)

References data	Mathur et al. (2008)	Level-crossing algorithm (Our experiment)
Hardware	Laptop	ESP8266
Wi-Fi channel	5 GHz	2.4 GHz
m	4	3
α	0.5	0.5
Number of collected RSSI	7,840	8,000
Number of bits passing Level-crossing threshold	511 bits	443 bits
Percentage of bits passing Level-crossing threshold	6.52%	5.54%



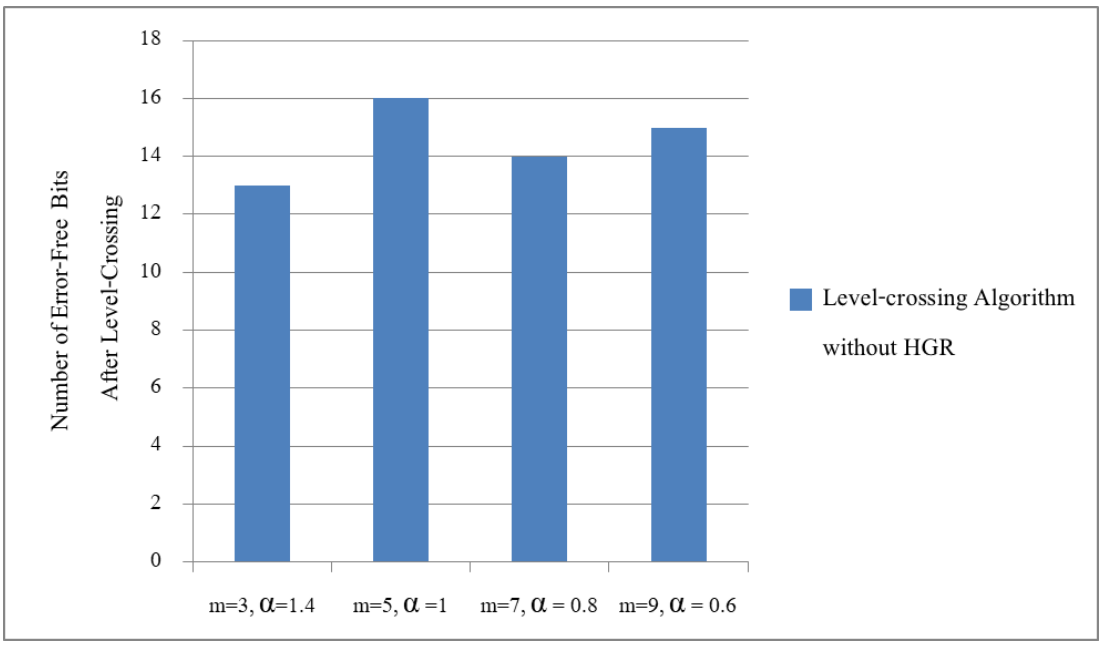
ภาพที่ 4-3 จำนวนบิตกุญแจที่ได้จากการทดลองการสร้างกุญแจจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า โดยวิธี Level-crossing algorithm ที่ค่า m ต่างกัน เมื่อปรับค่า α มากขึ้น

จากภาพที่ 4-3 จำนวนบิตกุญแจที่ได้จากการทดลองการสร้างกุญแจจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า โดยวิธี Level-crossing algorithm ที่ค่า m ต่างกันจะเห็นได้ว่า เมื่อมีการปรับค่า α มากขึ้นจำนวนบิตที่ได้จะน้อยลงจนไม่มีบิตเกิดขึ้น



ภาพที่ 4-4 จำนวนบิตผิดพลาดที่ได้จากการทดลองการสร้างกุญแจจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า โดยวิธี Level-crossing algorithm ที่ค่า m ต่างกัน เมื่อปรับค่า α มากขึ้น

จากภาพที่ 4-4 จำนวนบิตผิดพลาดที่ได้จากการทดลองการสร้างกุญแจจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า โดยวิธี Level-crossing algorithm ที่ค่า m ต่างกัน เมื่อปรับค่า α มากขึ้น เห็นได้ว่าเมื่อค่า m มากขึ้นจำนวนบิตที่ผิดพลาด (บิตที่สองฝั่งไม่ตรงกัน) ลดลง แต่ในกรณีที่วัตถุเคลื่อนที่จะมีการเปลี่ยนตำแหน่งตลอดเวลา ดังนั้นค่า m จึงไม่ควรมากเกินไป เพราะอาจทำให้ได้บิตกุญแจที่สามารถใช้งานได้ลดลงดังภาพที่ 4-5



ภาพที่ 4-5 จำนวนบิตกุญแจที่ได้จากการทดลองการสร้างกุญแจจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า โดยวิธี Level-crossing algorithm ที่ค่า m ต่างกัน เมื่อปรับค่า α น้อยที่สุดที่ทำให้ไม่เกิดความผิดพลาดบิตกุญแจ

จากภาพที่ 4-5 แสดงให้เห็นว่าจากการทดลองนี้การปรับค่า $m = 5$ และค่า $\alpha = 1$ ให้จำนวนบิตกุญแจที่ไม่เกิดความผิดพลาดบิตกุญแจมากที่สุดที่ 16 บิต ซึ่งการปรับค่า m และ α ถึงแม้จะทำให้ได้บิตน้อยลงแต่การปรับที่เหมาะสมจะทำให้ได้จำนวนบิตกุญแจที่ไม่เกิดความผิดพลาดบิตที่มากกว่า

ผลทดลองการตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming-code generated redundancy: HGR

ในภาพที่ 4-5 นั้นแม้ว่าจะสามารถปรับค่า m และค่า α ที่ทำให้ไม่เกิดความผิดพลาดบิตในการสร้างกุญแจได้ แต่อัตราการสร้างกุญแจยังต่ำดังนั้น เราจะปรับค่า α ให้เป็น 0.2 เพื่อเพิ่มอัตราการสร้างกุญแจ โดยนำเอาวิธี HGR มาตรวจจับข้อผิดพลาด ดังตารางที่ 4-2

ตารางที่ 4-2 ผลทดลองการสร้างกุญแจจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า โดยวิธี Level-crossing algorithm มาตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming-code generated redundancy: HGR ที่ค่า α =0.2 ปรับค่า m มากขึ้น

m	3	5	7
Number of bits passing level-crossing threshold	1012	341	144
Number of error bits detected and discarded by HGR process	808	241	112
Number of remaining bits	204	100	32
Number of remaining error bits	28	12	0

จากตารางที่ 4-2 ผลทดลองการสร้างกุญแจจากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า โดยวิธี Level-crossing algorithm มาตรวจจับข้อผิดพลาดบิตกุญแจโดยใช้ Hamming-code generated redundancy: HGR ที่ค่า α = 2 ปรับค่า m เป็นมากขึ้นจะเห็นว่าที่ค่า $m = 7$ ไม่มีบิตผิดพลาดเกิดขึ้นและมีบิตที่สามารถนำไปสร้างกุญแจจำนวน 32 บิต

ต่อไปเป็นการเปรียบเทียบจำนวนบิตที่มากที่สุด เมื่อปรับค่า α และ m ที่ทำให้ไม่เกิดความผิดพลาดบิตระหว่าง บิตที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้วิธี HGR ตรวจจับความผิดพลาดบิตกับแบบที่ใช้วิธี HGR ตรวจจับความผิดพลาดบิต

ตารางที่ 4-3 เปรียบเทียบจำนวนบิตกุญแจที่ได้จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 8,000 ค่า ระหว่างวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR กับวิธี Level-crossing algorithm แบบที่ใช้วิธี HGR โดยทั้งสองวิธีปรับค่า α และ m ที่ทำให้ไม่เกิดความผิดพลาดบิตกุญแจ

Technique	Without HGR Checking		With HGR Checking
m	3	9	7
α	1.4	0.6	0.2
Number of error-free bits after level-crossing	13 bits	15 bits	32 bits
Key bit generation percentage	0.16%	0.19%	0.4%

จากตารางที่ 4-3 พบว่า แบบที่ใช้วิธี HGR ตรวจจับความผิดพลาดบิตนอกจากจะช่วยตรวจจับบิตผิดพลาดแล้วยังสามารถทำให้อัตราส่วนกุญแจเพิ่มขึ้นเป็น 0.4% จากอัตราส่วนกุญแจเป็น 0.16% และ 0.19% ของวิธีที่ไม่มีการตรวจสอบความผิดพลาดบิตที่ค่า m เป็นน้อยสุดที่ 3 กับมากสุดที่ 9 และ α มากสุดเท่ากับ 1.4 กับน้อยสุดเท่ากับ 0.6 ตามลำดับ ถึงแม้อัตราส่วนบิตกุญแจที่ได้จะไม่มากนักแต่เราสามารถยืนยันได้ว่าบิตกุญแจที่ได้นี้ไม่มีบิตผิดพลาดเกิดขึ้น (บิตทั้งสองฝั่งตรงกันทั้งหมด)

เนื่องจากการสร้างกุญแจชุดบิตกุญแจที่สามารถนำไปใช้งานได้จะต้องทดสอบความสุ่ม เราจึงทำการทดสอบความสุ่มด้วยวิธีการ The frequency (Monobit) test ที่มุ่งเน้นที่การทดสอบสัดส่วนของบิตกุญแจที่เกิดระหว่างบิต 0 และ 1 ตามลำดับบิตทั้งหมด และ The runs test มุ่งเน้นการทดสอบการเปลี่ยนบิตของบิตถัดไปตามลำดับบิตทั้งหมด การทดสอบทั้งสองวิธีจะทำการหาค่าทางสถิติ เพื่อวิเคราะห์ว่าชุดบิตกุญแจที่ได้เป็นสุ่มหรือไม่ โดยการทดสอบจะได้ผลเป็นค่า P-value ถ้ามากกว่าเท่ากับ 0.01 แสดงว่าเป็นสุ่มและยิ่งค่า P-value เข้าใกล้ 1 ยิ่งเป็นผลดีที่แสดงถึงความสุ่มของชุดบิตกุญแจนั้น ซึ่งผลการทดสอบเป็นดังต่อไปนี้

ผลการทดสอบสุ่มโดยวิธี The frequency (Monobit) test

สำหรับวิธีนี้สามารถคำนวณหาค่า P-value ได้จากสมการที่ 2-7 โดยการทดสอบนี้ใช้ชุดบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR และแบบที่ใช้วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ซึ่งแบ่งเป็นสองสถานการณ์ คือ สถานการณ์ที่เคลื่อนที่ (Bob) และสถานการณ์ที่อยู่กับที่ (Alice) ตามภาพที่ 4-2 ผลการทดสอบเป็นดังนี้

ตารางที่ 4-4 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่เคลื่อนที่ (Bob)

α	m	Number of bits passing level-crossing threshold	P-value
0	0	16,000	1.68×10^{-4}
0	3	2,964	0.0023
0	5	1,055	0.0604
0	7	478	0.5831
0	9	244	0.8981
0.2	3	1,830	2.10×10^{-5}
0.2	5	572	0.0240
0.2	7	222	0.5021
0.2	9	112	0.8501

จากตารางที่ 4-4 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่เคลื่อนที่ (Bob) จะเห็นการปรับค่า m มากขึ้นชุดบิตที่ใช้ทดสอบได้ค่า P-value มากขึ้น แสดงว่า สัดส่วนบิต 0 และ 1 มีโอกาสเป็นสุ่มมากขึ้นเมื่อเพิ่มค่า m มากขึ้น

ตารางที่ 4-5 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่อยู่กับที่ (Alice)

α	m	Number of bits passing level-crossing threshold	P-value
0	0	16,000	2.40×10^{-46}
0	3	2,964	4.71×10^{-16}
0	5	1,055	9.23×10^{-5}
0	7	478	0.0104
0	9	244	0.2004
0.2	3	1,830	1.30×10^{-15}
0.2	5	572	0.0075
0.2	7	222	0.3474
0.2	9	112	0.4497

จากตารางที่ 4-5 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่อยู่กับที่ (Alice) จะเห็นการปรับค่า m มากขึ้นชุดบิตที่ใช้ทดสอบได้ค่า P-value มากขึ้นเช่นเดียวกับตารางที่ 4-4 แต่จะได้ค่า P-value ที่อยู่ในระดับที่เป็นสุ่มน้อยกว่าตารางที่ 4-4 แสดงว่า ที่ค่า m เท่ากันสถานการณ์เคลื่อนที่ (Bob) มีโอกาสเกิดชุดบิตกุญแจที่มีสัดส่วนบิต 0 และ 1 ที่ได้ค่า P-value เป็นสุ่มมากกว่าสถานการณ์ที่อยู่กับที่ (Alice)

ตารางที่ 4-6 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) Test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่เคลื่อนที่ (Bob)

α	m	Number of bits passing level-crossing threshold	P-value
0	0	2,784	0.0124
0	3	460	0.0932
0	5	240	0.7963
0	7	120	0.8551
0	9	56	-
0.2	3	396	0.0348
0.2	5	168	0.0206
0.2	7	56	-
0.2	9	36	-

จากตารางที่ 4-6 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่เคลื่อนที่ (Bob) เห็นได้ว่าค่า m มากขึ้นค่า P-value มากขึ้นถึงแม้ที่ค่า α = 0.2 และ$m = 9$ จะได้ค่า P-value น้อยกว่า m =7 ซึ่งอาจเป็นผลจากสภาพแวดล้อมที่เปลี่ยนแปลงที่ไม่สามารถควบคุมได้ในขั้นตอนการเก็บค่าความแรงสัญญาณ แต่โดยภาพรวมแล้วค่า m มากขึ้นค่า P-value มีโอกาสจะมากขึ้นตามไปด้วย

ตารางที่ 4-7 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้วิธี HGR จากค่าความแรงสัญญาณที่ได้รับจำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่อยู่กับที่ (Alice)

α	m	Number of Bits Passing Level-crossing Threshold	P-value
0	0	2,784	0.0407
0	3	460	0.1129
0	5	240	0.3017
0	7	120	0.8551
0	9	56	-
0.2	3	396	0.0049
0.2	5	168	0.0206
0.2	7	56	-
0.2	9	36	-

จากตารางที่ 4-7 ผลการทดสอบความสุ่มโดยวิธี The frequency (Monobit) test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่อยู่กับที่ (Alice) จะเห็นการปรับค่า m มากขึ้นชุดบิตที่ใช้ทดสอบได้ค่า P-value มากขึ้นเช่นกัน

ผลการทดลองทดสอบสุ่มโดยวิธี The run test

สำหรับวิธี The run test สามารถคำนวณหาค่า P-value ได้จาก สมการที่ 2-10 โดยการทดสอบนี้ใช้ชุดบิตกุญแจชุดเดียวกันกับการทดสอบความสุ่มวิธี The frequency (Monobit) test ซึ่งจะต้องผ่านการทดสอบสัดส่วนบิตจากวิธีแรกก่อนจึงจะสามารถนำมาทดสอบในวิธีนี้ได้ หรือค่า $|\pi - 1/2|$ ต้องน้อยกว่า τ โดยสามารถหา π ได้จากสามารที่ 2-8 และ τ จากสามการที่ 2-9 ซึ่งผลการทดสอบเป็นดังนี้

ตารางที่ 4-8 ผลการทดสอบความสุ่มโดยวิธี The run test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่เคลื่อนที่ (Bob)

α	m	Number of bits passing level-crossing threshold (n)	$\lvert\pi - 1/2\rvert$	τ	P-value
0	0	16,000	0.4360	0.0158	-
0	3	2,964	0.1670	0.0367	-
0	5	1,055	0.0289	0.0616	4.83×10^{-9}
0	7	478	0.0126	0.0915	1.95×10^{-7}
0	9	244	0.0041	0.1280	4.20×10^{-5}
0.2	3	1,830	0.0497	0.0467	-
0.2	5	572	0.0472	0.0836	3.39×10^{-7}
0.2	7	222	0.0225	0.1342	0.1132
0.2	9	112	0.0089	0.1890	0.0235

จากตารางที่ 4-8 ผลการทดสอบความสุ่มโดยวิธี The run test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่เคลื่อนที่ (Bob) จะเห็นว่าเมื่อค่า m มากขึ้นค่า P-value มากขึ้น แสดงว่าชุดบิตกุญแจที่ใช้ทดสอบนี้บิตถัดมีการเกิดความเปลี่ยนแปลงบ่อยจนเป็นสุ่ม

ตารางที่ 4-9 ผลการทดสอบความสุ่มโดยวิธี The run test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่อยู่กับที่ (Alice)

α	m	Number of bits passing level-crossing threshold (n)	$\lvert\pi - 1/2\rvert$	τ	P-value
0	0	16,000	0.4360	0.0158	-
0	3	2,964	0.1289	0.0367	-
0	5	1,055	0.0602	0.0616	5.93×10^{-48}
0	7	478	0.0586	0.0915	3.92×10^{-21}
0	9	244	0.0410	0.1280	1.36×10^{-16}
0.2	3	1,830	0.0934	0.0468	-
0.2	5	572	0.0559	0.0836	1.44×10^{-25}
0.2	7	222	0.0315	0.1342	4.35×10^{-9}
0.2	9	112	0.0357	0.1890	1.60×10^{-5}

จากตารางที่ 4-9 ผลการทดสอบความสุ่มโดยวิธี The run test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ไม่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่อยู่กับที่ (Alice) จะเห็นว่าเมื่อค่า m มากขึ้นค่า P-value มากขึ้นเช่นกัน

ตารางที่ 4-10 ผลการทดสอบความสุ่มโดยวิธี The run test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่เคลื่อนที่ (Bob)

α	m	Number of bits passing level-crossing threshold (n)	$\lvert\pi - 1/2\rvert$	τ	P-value
0	0	2,784	0.1473	0.0379	-
0	3	460	0.0391	0.0932	1.27×10^{-19}
0	5	240	0.0083	0.1291	2.88×10^{-8}
0	7	120	0.0083	0.1826	2.08×10^{-6}
0	9	56	-	-	-
0.2	3	396	0.0530	0.1005	3.98×10^{-17}
0.2	5	168	0.0893	0.1543	0.0035
0.2	7	56	-	-	-
0.2	9	36	-	-	-

จากตารางที่ 4-10 ผลการทดสอบความสุ่มโดยวิธี The run test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่เคลื่อนที่ (Bob) จะเห็นว่าเมื่อค่า m มากขึ้นค่า P-value มากขึ้น

ตารางที่ 4-11 ผลการทดสอบความสุ่มโดยวิธี The runs test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ที่ค่า m ต่างกันในสถานการณ์ที่อยู่กับที่ (Alice)

α	m	Number of bits passing level-crossing threshold (n)	$\lvert\pi - 1/2\rvert$	τ	P-value
0	0	2,784	0.1390	0.0379	
0	3	460	0.0370	0.0933	5.77×10^{-22}
0	5	240	0.0333	0.1291	3.28×10^{-10}
0	7	120	0.0083	0.1826	1.19×10^{-5}
0	9	56	-	-	-
0.2	3	396	0.0707	0.1005	4.40×10^{-19}
0.2	5	168	0.0893	0.1543	2.73×10^{-5}
0.2	7	56	-	-	-
0.2	9	36	-	-	-

จากตารางที่ 4-11 ผลการทดสอบความสุ่มโดยวิธี The run test กับบิตกุญแจที่ได้จากวิธี Level-crossing algorithm แบบที่ใช้ วิธี HGR จากค่าความแรงสัญญาณที่ได้รับ (RSSI) จำนวน 16,000 ค่า ในสถานการณ์ที่อยู่กับที่ (Alice) จะเห็นว่าเมื่อค่า m มากขึ้นค่า P-value มากขึ้นเช่นกัน

จากการทดสอบความสุ่มโดยใช้ชุดบิตกุญแจที่ได้จากจากวิธี Level-crossing algorithm แบบที่ใช้ วิธี HGR และไม่ใช้วิธี HGR ที่ค่า m มากขึ้น การทดสอบสัดส่วนโดยวิธี The frequency (Monobit) test ได้ค่า P-value ที่มีโอกาสอยู่ในค่าที่เป็นสุ่มมากขึ้น แต่เมื่อทดสอบด้วยวิธี The run test ที่มุ่งเน้นการทดสอบการเปลี่ยนแปลงของบิตจากลำดับทั้งหมด การปรับค่า m มากขึ้นยังไม่ได้ค่า P-value ที่อยู่ในช่วงความสุ่ม ดังนั้นอาจต้องลองทำการทดสอบความสุ่มด้วยวิธีอื่นแทนและเพิ่มจำนวน RSSI ให้มากขึ้น

บทที่ 5
สรุปและอธิบายผลการวิจัย

สรุปและอธิบายผลการเปลี่ยนแปลงค่า พารามิเตอร์ α และ m ของ Level-crossing algorithm

ตารางที่ 5-1 สรุปผลการเปลี่ยนแปลงค่า พารามิเตอร์ α และ m ของ Level-crossing algorithm

พารามิเตอร์		อัตราบิตกุญแจ	อัตราผิดพลาดบิตกุญแจ	ความสุ่ม			
				The frequency (Monobit) test		The run test	
				ตัวอยู่กับที่ (Alice)	ตัวเคลื่อนที่ (Bob)	ตัวอยู่กับที่ (Alice)	ตัวเคลื่อนที่ (Bob)
α	↑	↓	↓	↑	↓	↑	↑
m	↑	↓	↓	↑	↑	↑	↑

จากตารางที่ 5-1 เราสามารถสรุปได้ว่าใน Level-crossing algorithm เมื่อปรับค่า α และ m มากขึ้น อัตราการสร้างกุญแจและความผิดพลาดบิตกุญแจจะลดลง สำหรับการทดสอบความสุ่ม The frequency (Monobit) test ที่บอกถึงสัดส่วนบิต 0 และ 1 ที่มีโอกาสเป็นสุ่มของสถานการณ์ที่อยู่กับที่ (Alice) ให้ค่าเป็นสุ่มมากขึ้นเมื่อ ปรับค่า α และ m มากขึ้น แต่ของสถานการณ์ที่เคลื่อนที่จะให้ค่าเป็นสุ่มลดลงเมื่อปรับค่า α มากขึ้น สำหรับการทดสอบความสุ่มด้วย The run test ปรับค่า α และ m มากขึ้นทำให้เป็นสุ่มมากขึ้น

อย่างไรก็ตาม ถ้าผลที่ได้อยู่ในช่วงที่เป็นความสุ่มก็สามารถยอมรับได้ หรือแนะนำให้ทำการเก็บค่า RSSI มากขึ้นเพื่อสร้างกุญแจจำนวนมากขึ้น เพื่อให้ได้จำนวนบิตกุญแจมากขึ้นและทำการทดสอบกับวิธีการทดสอบความสุ่มในแบบอื่น ๆ ดังรายการที่ให้ไว้ในบทที่ 2 หัวข้อความสุ่ม

สรุปและอธิบายผลด้านความปลอดภัย

การออกแบบโปรโตคอลใหม่ในบทที่ 3 ที่ใช้ Charlie เข้ามาช่วยในการเพิ่มอัตราการสร้างบิตกุญแจ ซึ่งของเรามีสองรูปแบบ ดังภาพที่ 3-2 โดยภาพ (a) เป็นรูปแบบที่สามารถสร้างบิตกุญแจสองชุดในเวลาเดียวกันได้ทำให้ได้บิตกุญแจเพิ่มขึ้นสามเท่าจากแบบเดิมที่ไม่มี Charlie และในภาพ(b) เป็นรูปแบบที่สร้างบิตกุญแจได้ที่ละชุดแต่ก็ยังสามารถเพิ่มอัตราการสร้างกุญแจได้เป็น $\frac{3}{2}$ เท่าจากแบบเดิมที่ไม่มี Charlie ในตารางที่ 5-2 เราจะทำการคำนวณและแสดงการเปรียบเทียบทั้งสองรูปแบบกับวิธีการเดิมที่ไม่มี Charlie เข้ามาช่วยในการเพิ่มอัตราการสร้างบิตกุญแจ โดยนำข้อมูลจากการทดลองในบทที่ 4 ที่ได้ทำการเก็บ RSSI จำนวน 8,000 ค่าใช้เวลา 1.625 วินาทีต่อ RSSI หนึ่งค่า นำไปสร้างบิตกุญแจโดย Level-crossing ซึ่งแบบไม่ใช้วิธี HGR ได้บิตกุญแจที่ตรงกันทั้งหมด 15 บิต และแบบที่ใช้วิธี HGR ได้บิตที่ตรงกันทั้งหมด 32 บิต มาคำนวณหาค่าเวลาที่ใช้ในการสร้างบิตกุญแจจากรูปแบบที่ไม่มี Charlie เข้ามาช่วยในการเพิ่มอัตราการสร้างกุญแจสำหรับการเข้ารหัส AES ที่ความยาวกุญแจ 128 บิต ดังนี้

เวลาที่ใช้สร้างกุญแจ AES-128 แบบไม่ใช้วิธี HGR

$$(\frac{L_{AES}}{15} \times 8000) \times 1.625 \text{ วินาที} \quad (5\text{-}1)$$

เวลาที่ใช้สร้างกุญแจ AES-128 แบบใช้วิธี HGR

$$(\frac{L_{AES}}{32} \times 8000) \times 1.625 \text{ วินาที} \quad (5\text{-}2)$$

L_{AES} คือ ขนาดความยาวกุญแจ AES

จากการคำนวณเราได้สรุปผลการเปรียบเทียบเวลาที่ใช้ในการสร้างบิตกุญแจและวิเคราะห์ความปลอดภัยได้ตามตารางต่อไปนี้

ตารางที่ 5-2 เปรียบเทียบเวลาที่ใช้ในการสร้างบิตกุญแจและความปลอดภัยของแต่ละโปรโตคอล

HGR	New protocol with Charlie	Time needed for AES-128 key generation	Security
No	No	110,933 sec (30.81 hrs)	Strong
	Yes (a)	36,978 sec (10.27 hrs)	Weak
	Yes (b)	73,955 sec (20.54 hrs)	Weak
Yes	No	52,000 sec (14.44 hrs)	Weak
	Yes (a)	17,333 sec (4.81 hrs)	Weak
	Yes (b)	34,667 sec (9.63 hrs)	Weak

จากตารางที่ 5-2 จะเห็นได้ว่าโปรโตคอลใหม่ที่ใช้ Charlie โดยใช้รูปแบบการเก็บค่าความแรงสัญญาณ จากภาพที่ 3-2 แบบ (a) เมื่อไม่ใช้ HGR ถึงแม้ความปลอดภัยจะเป็นอย่างอ่อน (Weak) แต่ใช้เวลาเพียงหนึ่งในสามของแบบที่ไม่ใช้ Charlie และโปรโตคอลใหม่ที่ใช้ Charlie โดยใช้รูปแบบการเก็บค่าความแรงสัญญาณ จากภาพที่ 3-2 แบบ (b) ความปลอดภัยเป็น Weak แต่ก็ยังใช้เวลาเพียงสองในสามของแบบที่ไม่ใช้ Charlie ในส่วนเมื่อใช้ HGR โปรโตคอลใหม่ที่เรานำเสนอก็ยังให้ผลเช่นเดียวกัน แต่แบบที่ใช้ HGR จะใช้เวลาน้อยกว่าแบบที่ไม่ใช้ HGR เพราะสร้างบิตถูกต้องได้มากกว่า

เนื่องจากในตารางที่ 5-2 โปรโตคอลใหม่ของเราให้ความปลอดภัยเป็น Weak ทั้งหมด ซึ่งถ้าต้องการความปลอดภัยให้ได้เท่ากับ แบบที่ไม่ใช้ Charlie ผู้ใช้สามารถทำได้ โดยในขั้นตอนการออกแบบในบทที่ 3 ก่อนที่จะทำการเพิ่มอัตราการสร้างกุญแจให้ Alice และ Bob ทำการตกลงกันว่าจะใช้ บิตกุญแจ K_{CA} หรือ K_{BC} ก่อนทำการร้องขอกุญแจ $K_{CA} \oplus K_{BC}$ จาก Charlie เช่น Alice และBob ตกลงจะใช้ K_{CA} ดังนั้น Alice นำกุญแจที่ได้จาก Charlie เก็บไว้ แต่ Bob จะทำเช่นเดิมโดยการนำกุญแจ K_{BC} ไป XOR กับกุญแจที่ได้จาก Charlie ตามสมการที่ 3-3 จะได้ K_{CA} ใช้ในการเพิ่มอัตราการสร้างกุญแจ ซึ่งวิธีการเลือกกุญแจนี้ผู้ที่ดังฟังกุญแจจะไม่สามารถเดาส่วนใดส่วนหนึ่งของกุญแจได้ทำให้ความปลอดภัยของโปรโตคอลที่ใช้ Charlie แบบไม่ใช้ HGR เป็น Strong เหมือนกับแบบที่ไม่ใช้ Charlie ส่วนแบบที่ใช้ HGR แบ่งกุญแจเป็นบล็อกนั้น Alice และ Bob ต้องทำการตกลงเหลือเพียงหนึ่งบิตต่อบล็อกเพื่อใช้เป็นกุญแจจึงจะได้ความปลอดภัยแข็งแกร่ง (Strong) ในตารางที่ 5-3 แสดงเวลาที่ใช้สร้างกุญแจเมื่อความปลอดภัยเท่ากันทั้งหมดเป็นดังนี้

ตารางที่ 5-3 เปรียบเทียบเวลาที่ใช้ในการสร้างบิตกุญแจโดยที่มีความปลอดภัยของเท่ากันทุกโปรโตคอล

HGR	New protocol with charlie	Time needed for AES-128 key generation	Security
No	No	110,933 sec (30.81 hrs)	Strong
	Yes (a)	55,467 sec (15.41 hrs)	Strong
	Yes (b)	110,933sec (30.81 hrs)	Strong
Yes	No	208,000 sec (57.78 hrs)	Strong
	Yes (a)	104,000 sec (28.89 hrs)	Strong
	Yes (b)	208,000 sec (57.78 hrs)	Strong

จากตารางที่ 5-3 แสดงให้เห็นว่าที่ความปลอดภัยเท่ากันทั้งหมด แบบที่ไม่ใช้ Charlie ใช้เวลาเท่ากับแบบ (b) แต่ใช้เวลามากกว่าแบบ (a) สองเท่า ดังนั้นถ้าต้องการให้ความปลอดภัยที่เท่ากับแบบที่ไม่ใช้ Charlie รูปแบบโปรโตคอลที่แนะนำ คือ แบบ (a) ส่วนการใช้ HGR ไม่เหมาะกับแบบ Strong เพราะจะทำให้อัตราการสร้างกุญแจลดลงสี่เท่าจนใช้เวลาสร้างกุญแจนานกว่าแบบไม่ใช้ HGR

อย่างไรก็ตาม การเลือกใช้ขึ้นอยู่กับความต้องการด้านความปลอดภัย ซึ่งถ้าอุปกรณ์ที่ใช้สามารถเก็บค่าความแรงสัญญาณ (RSSI) สำหรับนำไปสร้างบิตกุญแจได้เร็วเพียงพอการเลือกใช้วิธีที่ให้ความปลอดภัยแบบ Strong ก็จะไม่ส่งผลต่ออัตราการสร้างกุญแจ แต่ถ้ามีข้อจำกัดด้านความเร็วของอุปกรณ์การลดความปลอดภัยลงในระดับที่ยอมรับได้ แต่ทำให้อัตราการสร้างกุญแจเพิ่มขึ้น การเลือกใช้โปรโตคอลใหม่ที่ใช้ Charlie ที่มีรูปแบบการเก็บค่าความแรงสัญญาณเป็นสองแบบดังภาพที่ 3-2 เป็นวิธีที่เราแนะนำ

บรรณานุกรม

จตุชัย แพงจันทร์. (2558). *Master in Security 3rd Edition*. นนทบุรี: ไอดีซี พรีเมียร์.

Ahlswede, R., Cai, N., Li, S. Y., & Yeung, R. W. (2000). Network information flow. *IEEE Transactions on information theory, 46*(4), 1204-1216.

Aono, T., Higuchi, K., Ohira, T., Komiyama, B., & Sasaoka, H. (2005). Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels. *IEEE Transactions on Antennas and Propagation, 53*(11), 3776-3784.

Bassham III, L. E., Rukhin, A. L., Soto, J., Nechvatal, J. R., Smid, M. E., Barker, E. B., & Heckert, N. A. (2010). Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications.

Bhattad, K., & Narayanan, K. R. (2005). Weakly Secure Network Coding. *Workshop on Network Coding*, *Theory and Applications*, *104.*

Cai, N., & Yeung, R. W. (2002). Secure network coding. In *Information Theory, 2002. Proceedings, 2002 IEEE,* 323.

Limmanee, A., & Henkel, W. (2010). Secure physical-layer key generation protocol and key encoding in wireless communications. *In GLOBECOM Workshops (GC Wkshps), 2010 IEEE*. 94-98.

Mathur, S., Trappe, W., Mandayam, N., Ye, C., & Reznik, A. (2008). Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *Proceedings of the 14th ACM international conference on Mobile computing and networking, 2008 ACM*. 128-139.

Parvez, I., Abdul, F., & Sarwat, A. I. (2016). A location based key management system for advanced metering infrastructure of smart grid. *In Green Technologies Conference (GreenTech), 2016 IEEE*, 62-67.

Raychaudhuri, D., & Mandayam, N. B. (2012). Frontiers of wireless and mobile communications. *Proceedings of the IEEE, 100*(4), 824-840.

Shu, L., & Costello, D. J. (2004). Error control coding. the second international edition, *Prentice-Hall,* 66-102.

Sharma, R. K., & Wallace, J. W. (2010). Measured statistics of reciprocal channel key generation of indoor MIMO channels. In *2010 IEEE Antennas and Propagation Society International Symposium*, *2010 IEEE*, 1-4.

Wallace, J. (2009). Secure physical layer key generation schemes: Performance and information theoretic limits. In *2009 IEEE International Conference on Communications, 2009 IEEE*, 1-5.

Zhang, J., Duong, T. Q., Marshall, A., & Woods, R. (2016). Key generation from wireless channels: A review. *IEEE Access, 4*, 614-626.