การพัฒนาโปรแกรมสื่อสารข้อมูลระบบไร้สายเคลื่อนที่ดิจิตอล บนโครงข่ายเวิลด์ดาต้า

DEVELOPMENT OF SOFTWARE PACKAGE FOR DIGITAL WIRELESS MOBILE DATA ON WORLDDATA NETWORK

รังสรรค์ น้อยจินดา

RANGSARN NOYCHINDA

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2542

ISBN 974-622-357-7

DEVELOPMENT OF SOFTWARE PACKAGE FOR DIGITAL WIRELESS MOBILE DATA ON WORLDDATA NETWORK

RANGSARN NOYCHINDA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

1999

ISBN 974-622-357-7

หัวข้อวิทยานิพนธ์	การพัฒนาโปรแกรมสื่อสารข้อมูลระบบไร้สายเคลื่อนที่ดิจิตอลบนโครงข่ายเวิลด์ดาต้า
นักศึกษา	นายรังสรรค์ น้อยจินดา
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ. เกษตร์ ศิริสันติสัมฤทธิ์
หลักสูตร	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมไฟฟ้า
พ.ศ.	2542

บทคัดย่อ

การสื่อสารข้อมูลระบบไร้สายเคลื่อนที่ดิจิตอลบนโครงข่ายเวิลด์ดาต้าเป็นการอาศัยการรับส่งข้อมูลแบบแพคเก็ตสวิตชิ่งด้วยความเร็ว 19.2 kbps และใช้ช่องความถี่วิทยุในย่าน 800 MHz โดยใช้โปรโตคอล NCL ที่เป็นภาษาในการควบคุมการแลกเปลี่ยนข้อมูลแบบอะซิงโครนัสระหว่างอุปกรณ์คอมพิวเตอร์ปลายทางและโมเด็มไร้สาย ข้อมูลแต่ละเฟรมที่ส่งจะถูกบีบอัดก่อน โดยได้นำเสนอการปรับปรุงอัลกอริทึมการบีบอัดข้อมูลของ Lempel-Ziv อาศัยหลักการของการคาดการณ์ทำการค้นหาข้อมูลที่ซ้ำซ้อนกันให้ได้ยาวที่สุด ถ้าข้อมูลแมทช์กันได้จะเขียนค่าตามความยาวของการแมทช์ไว้ แต่ถ้าไม่แมทช์กันจะเขียนข้อมูลเดิมไว้ หลังจากนั้นใช้ขบวนการของ Static Huffman coding หาความถี่ของข้อมูลที่เกิดขึ้นแต่ละตัวอักขระและแทนข้อมูลนั้นด้วยบิตแพทเทิร์นที่สั้นลง

หลังจากผ่านขบวนการบีบอัดข้อมูลแล้ว ข้อมูลจะถูกเข้ารหัสลับแบบ RC5 ซึ่งเป็นวิธีการเข้ารหัสลับแบบบล็อคชนิดซิมเมทริกซ์ โดยมีการแบ่งข้อมูลออกเป็นบล็อคที่เท่ากัน และสามารถกำหนดขนาดของบล็อคได้ตามสถาปัตยกรรมของเครื่องคอมพิวเตอร์ เช่น 16 บิต 32 บิต และ 64 บิต ทั้งนี้ระดับความปลอดภัยยังกำหนดได้จากความยาวของกุญแจรหัสลับ และวงรอบของการคำนวณวิธีการแบบ RC5 นี้ในการเข้าและถอดรหัสลับจะใช้กุญแจรหัสลับเดียวกัน เราเรียกกุญแจรหัสลับของวิธีนี้ว่ากุญแจปกปิด ในการแลกเปลี่ยนกุญแจปกปิดเราใช้วิธีการเข้ารหัสลับแบบ RSA โดยที่สมาชิกแต่ละคนจะต้องมีกุญแจรหัสลับ 2 ชนิด คือ กุญแจส่วนตัวและกุญแจสาธารณะ ในการสร้างกุญแจรหัสลับนั้นเราได้ใช้ตัวเลขสุ่มจากแหล่งกำเนิดของช่วงเวลาคลิกส์เม้าส์ ช่วงเวลาการกดแป้นพิมพ์ เวลาของระบบและสัญญาณนาฬิกามาประกอบกัน จากนั้นนำมาเข้าฟังก์ชั่นไดเจสเอกสารเพื่อให้กลุ่มบิตข้อมูลเป็นอิสระต่อกัน เป็นการยากที่ผู้บุกรุกจะคาดเดาและทำลายกุญแจรหัสลับได้

Thesis Title	Development of Software Package for Digital Wireless Mobile Data on WorldData Network
Student	Mr. Rangsarn Noychinda
Thesis Advisor	Assoc.Prof. Kaset Sirisantisamrid
Degree	Master of Engineering in Electrical Engineering
Year	1999

Abstract

The digital wireless mobile data communication is on a World Data network. The data communicates be utilizing the packet switching technology at a speed of 19.2 kbps and radio frequency of 800 MHz. The protocol utilized to transfer information between the mobile terminal and the radio packet modem is known as NCL (Native Command Language) which is an asynchronous transaction.

In each input data frame would be compressed before sending. We propose the compression method improved will be the Lempel-Ziv algorithm by prediction. It works by searching for redundant the longest data context in the input data. If an appropriate match is found, then the match length is written. If no appropriate match is found, then the literal is written. At this point, the Static Huffman coding algorithm will look at the frequency of occurrence of the literal and match length symbol are computed first, and then constructs a full binary tree with shorter bit patterns.

After data compression processed, the data encrypted by RC5 (Rivest Cipher) algorithm that is a symmetric block cipher. The plaintext and ciphertext are fixed-length bit blocks which can be utilized by processors of difference word-lengths. This is because a 16-bit, 32-bit, or a 64-bit processors are capable of effectively increasing computation speed. Furthermore, RC5 has a variable-length secret key and a variable number of rounds which provide flexibility in its security level. We called the RC5 key is secret key. We propose to use the RSA encryption method for RC5 key exchange. In the RSA (Rivest, Shamir, Adleman) method, each person has a pair of keys, One key is a public key, and the other one is a private key. The public key will be available for all to see, but the private key will be kept by each person. We can only decrypt by a private key. The RC5 and RSA key generator used random number sources from mouse click timing, keystroke timing, time and clock. The random output run its through MD5 (Message Digest) algorithm to produce the bits are all independent. The attacker difficult to guess and break the key.

กิตติกรรมประกาศ

ขอกราบขอบพระคุณอาจารย์ รศ. เกษตร์ ศิริสันติสัมฤทธิ์ เป็นอย่างสูง ผู้ซึ่งประสาทวิชาความรู้ ตลอดจนให้คำปรึกษาแนะนำแนวทาง และวิธีการแก้ไข จนกระทั่งงานวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี

และขอบพระคุณ คุณวิวรรธน์ อิ่มอารมณ์ จากบริษัท มัลติมีเดีย แอนด์ เซอร์วิสเซส จำกัด ที่กรุณาให้ความช่วยเหลือทางด้านข้อมูลและอุปกรณ์โมเด็มไร้สายคลื่นวิทยุ ที่ใช้บนโครงข่ายเวิลด์คาต้า สำหรับการทดสอบและงานวิจัยในการทำวิทยานิพนธ์ฉบับนี้

รังสรรค์ น้อยจินดา

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ หน้า

สารบัญภาพ

ภาพที่ หน้า

สารบัญภาพ (ต่อ)

ภาพที่ หน้า

สารบัญภาพ (ต่อ)

ภาพที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัญหาที่พบในระบบการสื่อสารข้อมูลไร้สายเคลื่อนที่ดิจิตอลบนโครงข่ายเวิลด์ดาต้า ที่เกิดจากคุณลักษณะทางโปรแกรมสื่อสารข้อมูล สามารถสรุปเป็นหัวข้อที่สำคัญคือ

1.1.1 การรับส่งข้อมูลไม่มีความปลอดภัยในความลับของข้อมูล เป็นไปได้ที่อาจถูกดักจับข้อมูลสำคัญเอาไปใช้ประโยชน์ อันจะเป็นผลเสียต่อเจ้าของข้อมูลได้

1.1.2 การรับส่งข้อมูลในโครงข่ายมีการใช้แบนด์วิต (Bandwidth) ร่วมกัน ที่อัตราความเร็ว 19.2 กิโลบิตต่อวินาที ทำให้เกิดความล่าช้าในการส่งข้อมูล เมื่อข้อมูลมีขนาดใหญ่มาก เช่น ข้อมูลภาพ ข้อมูลแฟกซ์ และการรับส่งไฟล์เป็นต้น

1.2 วัตถุประสงค์ของวิทยานิพนธ์

การสื่อสารข้อมูลบนโครงข่ายเวิลด์ดาต้า ที่ผ่านมามีการประยุกต์ใช้งานเกี่ยวกับ การรายงานราคาหุ้น การตรวจสอบวงเงินในบัตรเครดิตตามปั๊มน้ำมัน การรับส่งแฟกซ์ อินเตอร์เน็ตอีเมล์ การส่งข้อมูลไปเพจเจอร์ และการติดตามหาตำแหน่งยานพาหนะ เป็นต้น ทำให้ต้องคำนึงถึงประเด็นเหล่านี้คือ ความเร็วของช่องการสื่อสาร (Speed communication channel) และความปลอดภัยในความลับของข้อมูล (Secret data) เนื่องจากเทคโนโลยีในปัจจุบันของระบบไร้สาย ไม่สามารถใช้ความเร็วของช่องสื่อสารสูงๆ ได้ และต้องลงทุนระบบโครงข่ายสูงมากเมื่อต้องการเพิ่มความเร็วในการรับส่งข้อมูลให้สูงขึ้น นอกจากนี้การรับส่งข้อมูลในระบบนี้อาจถูกดักฟังข้อมูลได้ถ้าไม่มีวิธีการป้องกันดีพอ ข้อมูลที่สำคัญของเราอาจถูกผู้อื่นนำไปใช้ประโยชน์ ทำให้เกิดผลเสียต่อเจ้าของข้อมูลได้

จากเหตุผลดังกล่าวข้างต้นในวิทยานิพนธ์ฉบับนี้ ได้นำเสนอวิธีการพัฒนาโปรแกรมสื่อสารข้อมูลระบบไร้สายใน 2 ส่วนหลัก คือ เทคนิคการบีบอัดข้อมูล (Data compression) และการเข้ารหัสลับข้อมูล (Data encryption) เพื่อให้การรับส่งข้อมูลบนโครงข่ายมีความปลอดภัยในข้อมูลมากขึ้น และลดเวลาการรับส่งข้อมูลให้น้อยลง ทั้งนี้ยังเป็นการเพิ่มประสิทธิภาพการรองรับแบนด์วิต (Bandwidth) บนโครงข่ายเวิลด์ดาต้าให้มากขึ้นด้วย กล่าวขั้นตอนโดยย่อก็คือข้อมูลจะถูกบีบอัด (Compress) ก่อนกระบวนการเข้ารหัสลับข้อมูล หลังจากนั้นข้อมูลถูกนำส่งผ่านคอมพิวเตอร์ปลายทางด้วยโปรคอลสื่อสารแบบ NCL (Native Control Language) ติดต่อกับโมเด็มไร้สายแปลง

ให้เป็นคลื่นวิทยุส่งออกอากาศไปยังเครื่องใช้งานผู้รับ ฝ่ายเครื่องผู้รับทำการย้อนกลับ (Backward) วิธีการจากเครื่องผู้ส่ง เพื่อให้ได้ข้อมูลต้นฉบับ (Source data) ออกมา

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1.3.1 ลดเวลาการรับส่งข้อมูลให้น้อยลง

1.3.2 ทำให้โครงข่ายเวิลด์คาต้ามีช่องสัญญาณว่างมากขึ้น จากการที่ใช้เวลารับส่งน้อยลง

1.3.3 ทำให้มั่นใจในความปลอดภัยข้อมูลมากขึ้น

1.4 ขอบเขตของงานวิจัย

การพัฒนาวิจัยได้เสนอเทคนิคการบีบอัดข้อมูล และการเข้ารหัสลับข้อมูลเพื่อเพิ่มความเร็วและความปลอดภัยข้อมูลให้มากขึ้นเมื่อมีการรับส่งข้อมูลบนระบบไร้สายเคลื่อนที่ดิจิตอลบนโครงข่ายเวิลด์คาต้า รายละเอียดของวิทยานิพนธ์ฉบับนี้จะแบ่งออกเป็น 7 บท ในแต่ละบทมีหัวข้อและเนื้อหา ดังนี้

บทที่ 1 เป็นบทนำ จะกล่าวถึงความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ของวิทยานิพนธ์ ประโยชน์ที่คาดว่าจะได้รับ และขอบเขตของงานวิจัย

บทที่ 2 จะกล่าวถึงโครงสร้างพื้นฐานของระบบโครงข่ายเวิลด์คาต้า ซึ่งเป็นโครงข่ายที่เกี่ยวข้องในงานวิจัยในครั้งนี้ การรับส่งข้อมูลในโครงข่ายเป็นแบบแพ็คเก็ตสวิตชิ่ง (Packet switching) มีอัตราความเร็ว 19.2 Kbps และใช้ช่องสัญญาณความถี่วิทยุในย่าน 800 MHz ระบบดิจิตอล การรับส่งข้อมูลของผู้ใช้จะผ่านโมเด็มไร้สาย เพื่อติดต่อไปยังบริการคอมพิวเตอร์หลัก (Host computer) หรือจะใช้รับส่งข้อมูลระหว่างโมเด็มไร้สายด้วยกัน โดยใช้โปรแกรมที่ได้รับการพัฒนาจากวิทยานิพนธ์ฉบับนี้รับส่งข้อมูลระหว่างกัน

บทที่ 3 แสดงให้เห็นถึงคุณสมบัติของโปรโตคอลสื่อสารที่ใช้ในระบบ คือ โปรโตคอลสื่อสาร NCL (Native Control Language) เป็นภาษาควบคุมการแลกเปลี่ยนข้อมูลแบบอะซิงโครนัสระหว่างอุปกรณ์คอมพิวเตอร์ปลายทางและโมเด็มไร้สาย เพื่อควบคุมการส่งข้อมูลออกไปเป็นคลื่นวิทยุไปยังโครงข่าย เมื่อผู้ใช้ (User) ต้องการติดต่อกับคอมพิวเตอร์หลัก (Host computer) จะใช้โปรโตคอล SCR (Standard Context Routing) ทำหน้าที่ควบคุมเซสชั่นการเชื่อมโยงของอุปกรณ์สื่อสารเคลื่อนที่กับคอมพิวเตอร์หลัก โดยอยู่บนเลเยอร์ของโปรโตคอลมาตรฐาน X.25 หรือ TCP/IP

บทที่ 4 กล่าวถึงเทคนิคการบีบอัดข้อมูล (Data compression) เป็นการปรับปรุงอัลกอริทึมของ Lempel-Ziv โดยใช้หลักการของการคาดการณ์ (Prediction) เพื่อค้นหาข้อมูลที่ซ้ำซ้อนกันให้ได้ยาวที่สุด เราจึงเรียกวิธีการใหม่นี้ว่า Lempel-Ziv-Prediction (LZP) การทำงานจะแบ่งออกเป็น 2 ขั้นตอน คือ ขั้นตอนแรกลดความซ้ำซ้อนของข้อมูลที่จะทำการบีบอัดลงก่อน โดยใช้วิธีการของ Context Modeling ทำการเปรียบเทียบข้อมูลก่อนหน้าและข้อมูลปัจจุบัน ถ้าข้อมูลแมทช์กันจะเขียนค่าความยาวของการแมทช์ไว้ แต่ถ้าไม่แมทช์กันจะเขียนข้อมูลเดิมเก็บไว้ จากนั้นขั้นตอนที่สองใช้หลักการของ Static Huffman coding ลดจำนวนบิตข้อมูลให้น้อยลง โดยพิจารณาจากความเป็นไปได้ที่เกิดขึ้นของอักขระแต่ละตัว ถ้าอักขระมีความเป็นไปได้สูงหรือมีจำนวนมากแล้วจะถูกแทนด้วยจำนวนบิตที่น้อย ในทางตรงกันข้ามถ้าอักขระมีจำนวนน้อยจะถูกแทนด้วยจำนวนบิตที่มาก

บทที่ 5 กล่าวถึงเทคนิคการเข้ารหัสลับข้อมูล โดยอาศัยการเข้ารหัสลับแบบผสมระหว่าง RC5 และ RSA วิธีการของ RC5 เป็นแบบบล็อคชนิดซิมเมทริกซ์ มีการแบ่งข้อมูลออกเป็นบล็อคๆ ที่เท่ากัน สามารถกำหนดขนาดของบล็อคได้ตามสถาปัตยกรรมของเครื่องคอมพิวเตอร์ เช่น 16 บิต 32 บิต หรือ 64 บิต เป็นต้น และใช้วิธีหมุนบิตสลับบิตข้อมูล วิธีการนี้จะให้ความเร็วในการเข้ารหัสลับสูง และความปลอดภัยสามารถกำหนดได้จากความยาวของกุญแจรหัสลับและวงรอบของการคำนวณ ส่วนวิธีการของ RSA ถูกนำมาใช้ในขั้นตอนการแลกเปลี่ยนกุญแจรหัสลับ ในที่นี้คือใช้แลกเปลี่ยนกุญแจปกปิดของ RC5 วิธีการนี้จะใช้หลักการของโมดูลัส (Modulus) โดยเลือกค่าจำนวนเฉพาะ (Prime number) ที่มีค่ามาก เพื่อให้การแยกตัวประกอบ (Factoring) ทำได้ยาก ซึ่งเป็นจุดแข็งด้านความปลอดภัยของอัลกอริทึมแบบนี้ ทำให้มีความมั่นใจในการแลกเปลี่ยนกุญแจปกปิดมากขึ้น สำหรับวิธีการสร้างกุญแจรหัสลับทั้งแบบ RC5 และ RSA ในวิทยานิพนธ์นี้ได้ใช้วิธีการสุ่มจากตัวเลขสุ่มจริง และตัวเลขสุ่มเทียมมาประกอบกัน เพื่อให้ยากต่อการทำลายกุญแจรหัสลับ

บทที่ 6 เป็นส่วนของโปรแกรมสื่อสารข้อมูลระบบไร้สาย อยู่บนโครงข่ายเวิลด์ดาต้า ประกอบด้วยส่วนสำคัญ 5 ส่วนคือ การติดต่อระหว่างคอมพิวเตอร์ปลายทางและโมเด็มไร้สาย การบีบอัดข้อมูล การสร้างกุญแจรหัสลับ การแลกเปลี่ยนกุญแจรหัสลับ และการเข้ารหัสลับข้อมูล

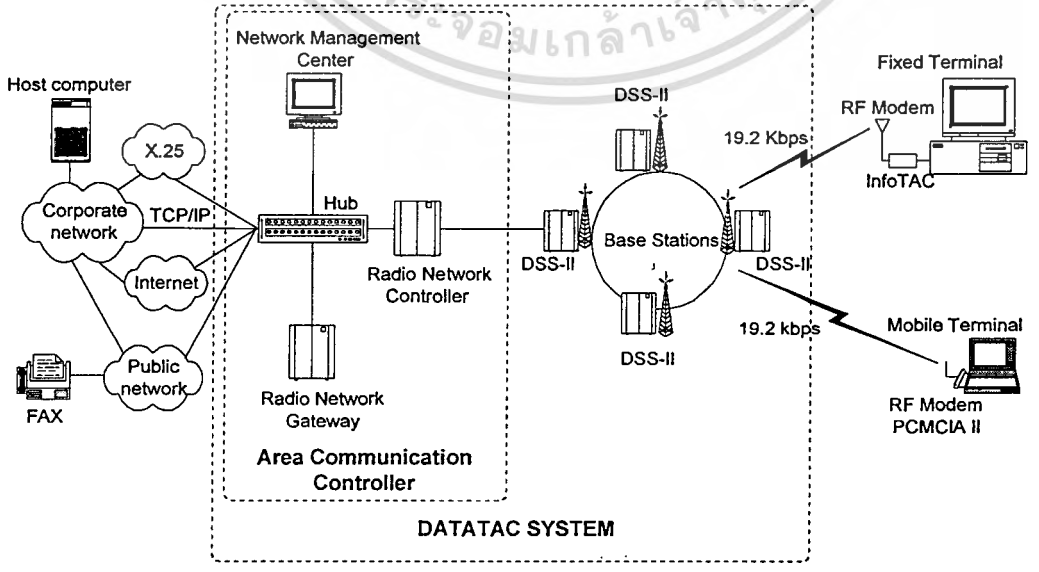
บทที่ 7 เป็นบทสรุปและผลการทดลอง

ในส่วนสุดท้ายเป็นบรรณานุกรม และภาคผนวกแสดงผลงานวิจัยที่ได้รับการตีพิมพ์ แสดงการใช้งานโปรแกรมและแสดงฟิลด์ข้อมูลในเฟรมหน่วยบริการข้อมูล (SDU) ของโปรโตคอล NCL

บทที่ 2
โครงสร้างของระบบโครงข่ายเวิลด์ดาต้า

โครงข่ายเวิลด์ดาต้าเป็นระบบสื่อสารข้อมูลไร้สายเคลื่อนที่ของบริษัทในกลุ่มยูคอม ได้รับสัมปทานให้บริหารโครงข่ายเป็นระยะเวลา 20 ปี จากการสื่อสารแห่งประเทศไทย โดยนำเทคโนโลยีมาจากโมโตโรล่า มีชื่อเรียกว่า DataTAC ที่สามารถขยายขนาดโครงข่ายได้จำนวนมาก และมีความยืดหยุ่นในการขยายการเชื่อมต่อกับคอมพิวเตอร์หลัก (Host computer) ได้หลากหลาย เช่น อินเตอร์เน็ต แฟกซ์ การรับส่งไฟล์ เป็นต้น คุณสมบัติของระบบมีการรับส่งข้อมูลในรูปแบบแพ็คเก็ตสวิตชิ่ง (Packet switching) ที่ความเร็ว 19.2 กิโลบิตต่อวินาที และใช้ช่องสัญญาณความถี่วิทยุระบบดิจิตอลย่าน 800 เมกะเฮริท์ อุปกรณ์คอมพิวเตอร์ปลายทาง (Terminal) เป็นได้ทั้งแบบเคลื่อนที่ (Mobile) และแบบตั้งอยู่กับที่ (Fixed) ต่อพ่วงอยู่กับอุปกรณ์โมเด็มไร้สาย (Wireless modem) เราเรียกชื่ออุปกรณ์คอมพิวเตอร์ปลายทางและโมเด็มไร้สายรวมกันว่าอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ (Mobile terminal) การรับส่งข้อมูลระหว่างกันบนโครงข่ายจะต้องเขียนโปรแกรมเพิ่มเติม ซึ่งคุณลักษณะของโปรแกรมใช้งานจะขอกล่าวรายละเอียดในบทต่อ ๆ ไป

รูปที่ 2.1 โครงสร้างของโครงข่าย DataTAC



2.1 ส่วนประกอบของระบบ

โครงข่าย DataTAC นี้ ประกอบด้วยอุปกรณ์หลัก 3 ส่วน ดังรูปที่ 2.1 คือ

2.1.1 อุปกรณ์ควบคุมพื้นที่ติดต่อสื่อสาร (Area Communications Controller)

ทำหน้าที่ควบคุมขอบเขตพื้นที่ (Coverage area) การติดต่อสื่อสาร ประกอบด้วยอุปกรณ์ย่อย 4 ส่วน ดังนี้

2.1.1.1 อุปกรณ์ควบคุมโครงข่ายคลื่นวิทยุ (Radio Network Controller)

ทำหน้าที่ควบคุมการรับส่งข้อมูล ติดตามการใช้งานของอุปกรณ์สื่อสารข้อมูลเคลื่อนที่บนโครงข่ายความถี่วิทยุ (RF Network) ดูแลการเชื่อมต่อกับอุปกรณ์โครงข่ายวิทยุ (Radio Network Gateway) ดูแลการทำโรมมิ่ง และการกำหนดเส้นทางข้อมูล เราสามารถแบ่งเบาภาระทำงาน (Load balancing) โดยเพิ่มอุปกรณ์ในส่วนนี้ให้ช่วยกันทำงานได้

2.1.1.2 อุปกรณ์เกตเวย์โครงข่ายวิทยุ (Radio Network Gateway)

ส่วนนี้ใช้เป็นจุดเชื่อมต่อเพื่อแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์ควบคุมโครงข่ายคลื่นวิทยุและคอมพิวเตอร์หลัก โดยมีหน้าที่จัดเก็บฐานข้อมูลผู้ใช้ เพื่อกำหนดสิทธิการใช้งาน

2.1.1.3 ชุมสายสื่อสาร (Communications hub)

ทำหน้าที่เป็นเกตเวย์ระหว่างคอมพิวเตอร์บนโครงข่าย DataTAC และคอมพิวเตอร์หลัก ซึ่งอาจจะเป็นเร้าท์เตอร์ (Router) หรือ อุปกรณ์สื่อสารแม่ข่าย (Communication server) ที่ใช้งานได้กับโปรโตคอล TCP/IP และ X.25 ได้

2.1.1.4 ศูนย์สั่งการและควบคุมโครงข่าย (Network Management Center)

การใช้งานเป็นลักษณะกราฟิก หรือ GUI (Graphical User Interface) มีหน้าที่หลัก ๆ คือ การเฝ้ามองสิ่งผิดพลาดในระบบ เก็บสถิติของจำนวนข้อมูลที่รับส่งกันบนโครงข่าย และการตั้งค่าพารามิเตอร์ในระบบ DataTAC ให้เป็นไปตามความต้องการ

2.1.2 อุปกรณ์สถานีรับส่งสัญญาณ (Base stations)

เรารวมอุปกรณ์ควบคุมสถานีรับส่งสัญญาณและอุปกรณ์รับส่งความถี่วิทยุเข้าด้วยกัน เราเรียกว่า Data System Station II หรือ DSS-II โดยทำหน้าที่รับส่งสัญญาณความถี่วิทยุที่มีลักษณะโปรโตคอลสื่อสารแบบ RD-LAP (Radio Data-Link Access Protocol)

2.1.3 อุปกรณ์สื่อสารข้อมูลเคลื่อนที่ (Mobile terminal) หรือ เครื่องผู้ใช้

โดยทั่วไปเครื่องผู้ใช้จะประกอบด้วย 2 ส่วนหลักคือ ตัวคอมพิวเตอร์ปลายทางพร้อมโปรแกรมใช้งาน และโมเด็มไร้สายชนิดคลื่นวิทยุแบบแพ็คเก็ต (Radio Packet Modem) หน้าที่ของโมเด็มไร้สาย คือจัดการกับเฟรมข้อมูล และแปลงข้อมูลที่ใช้รับส่งกันระหว่างอุปกรณ์คอมพิวเตอร์ปลายทางและสถานีรับส่งสัญญาณวิทยุ

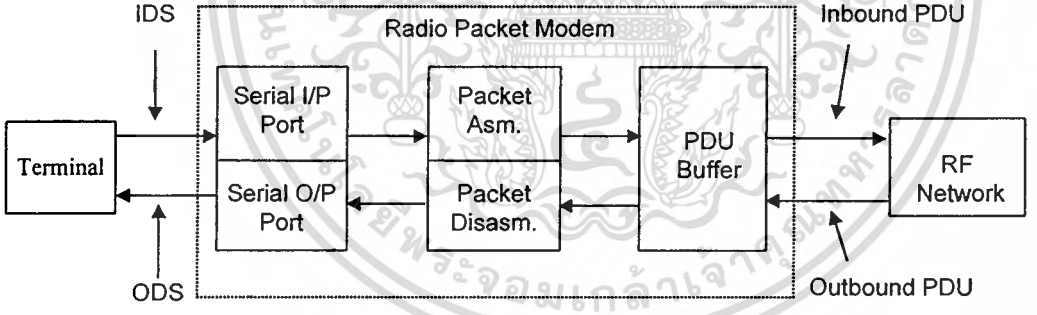
2.2 ส่วนของโมเด็มไร้สาย

วิธีการเชื่อมต่อระหว่างโมเด็มไร้สาย และอุปกรณ์คอมพิวเตอร์ปลายทาง สามารถกระทำได้ 2 โหมด คือ โหมดทรานสพาเรนท์ (Transparent mode) และโหมดเนทีฟ (Native mode)

2.2.1 โหมดทรานสพาเรนท์ (Transparent mode)

เป็นการจำลองคำสั่งมาจาก AT Command ตามมาตรฐานของ Hayes โดยมีการทำงานเปรียบเสมือนกับโมเด็มที่มีสาย (Wire-line modem) ทั่ว ๆ ไป แต่มีการใช้คำสั่งพิเศษเพิ่มเติมให้กับโมเด็มไร้สาย เช่น การเลือกช่องสัญญาณความถี่วิทยุ การควบคุมการหาเส้นทางโครงข่าย และการตั้งค่าพารามิเตอร์ต่าง ๆ ที่ใช้ในการควบคุมการทำงาน ในโหมดนี้มีการทำงาน 2 แบบ คือ โหมดออนไลน์ (On-line mode) ใช้สำหรับการรับส่งข้อมูล และโหมดคำสั่ง (Command mode) ใช้สำหรับตั้งค่าพารามิเตอร์ ควบคุมการทำงาน และจัดการติดต่อคำสั่งจากอุปกรณ์คอมพิวเตอร์ปลายทาง รูปที่ 2.2 รูปแบบการสื่อสารของโหมดทรานสพาเรนท์

รูปที่ 2.2 รูปแบบการสื่อสารของโหมดทรานสพาเรนท์



จากรูปที่ 2.2 เมื่อมีข้อมูลส่งจากอุปกรณ์คอมพิวเตอร์ปลายทาง (terminal) ข้อมูลต่อเนื่องส่งเข้า (Inbound Data Stream หรือ IDS) ถูกส่งเข้าไปยังพอร์ตอนุกรมขาเข้า (Serial I/P Port) ของโมเด็มไร้สายชนิดคลื่นวิทยุแบบแพ็คเก็ต (Radio Packet Modem) และนำข้อมูลไปประกอบกันเป็นแพ็คเก็ต (Packet Assembler) ในรูปแบบของหน่วยข้อมูลมาตรฐาน (Protocol Data Unit หรือ PDU) ขนาดแพ็คเก็ตขึ้นอยู่กับวิธีการนำแพ็คเก็ตมาประกอบกัน (Packetization method) จากนั้นนำไปเก็บไว้ในบัฟเฟอร์ก่อนที่จะส่งออกเข้าสู่โครงข่ายความถี่วิทยุ (RF Network)

ในทางกลับกันเมื่อมีแพ็คเก็ตของหน่วยข้อมูลมาตรฐาน ส่งมาจากโครงข่ายความถี่วิทยุ อุปกรณ์โมเด็มไร้สายจะทำหน้าที่รับเข้ามาเก็บไว้ในบัฟเฟอร์ และทำการแปลงแพ็คเก็ต (Packet

Disassembler) ให้เป็นข้อมูลต่อเนื่องส่งออก (Outbound Data Stream หรือ ODS) แล้วส่งข้อมูลออกทางพอร์ตอนุกรมขาออก (Serial O/P Port) ของโมเด็มไร้สาย ส่งไปยังอุปกรณ์คอมพิวเตอร์ปลายทาง

2.2.2 โหมดเนทีฟ (Native mode)

การทำงานในโหมดนี้จัดเป็นการทำงานเชิงการติดต่อแลกเปลี่ยนทันทีทันใด (Transaction-oriented) และมีการสื่อสารที่ปราศจากการเชื่อมต่อตลอดเวลา (Connectionless communication) ใช้หน่วยบริการข้อมูลที่เรียกว่า Service Data Units หรือ SDU ทำหน้าที่แลกเปลี่ยนข้อมูลระหว่างอุปกรณ์คอมพิวเตอร์ปลายทางและโมเด็มไร้สาย ซึ่งในแต่ละ SDU สามารถทำงานในขบวนการเดียวกันที่เวลาเดียวกัน (Multi-threaded) ได้ SDU มีบริการอยู่ 3 ชนิด คือ คำสั่ง (Command) ตอบสนอง (Response) และรายงานเหตุการณ์ (Event report) บริการต่าง ๆ เหล่านี้เราเรียกว่า ภาษาควบคุมการทำงานในโหมดเนทีฟ (Native Control Language หรือ NCL) ซึ่งจะได้กล่าวรายละเอียดต่อไปในบทที่ 3

2.3 ชนิดของการเชื่อมต่อระหว่างอุปกรณ์เกตเวย์โครงข่ายวิทยุ และคอมพิวเตอร์หลัก

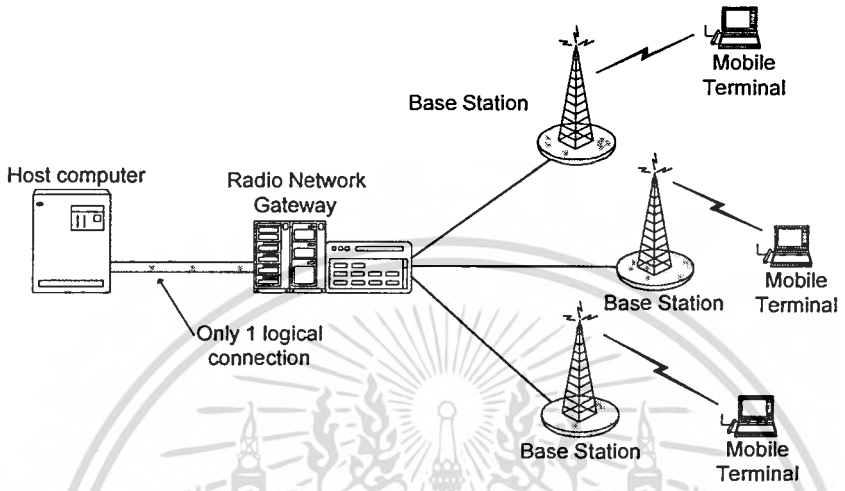
ระบบของการเชื่อมต่อบนโครงข่าย DataTAC นี้ มีลักษณะการเชื่อมต่อเป็น 2 ประเภทใหญ่ ๆ คือ แบบเป็นกลุ่ม (Fleet Connectivity) หรือกล่าวอีกอย่างหนึ่งว่าเป็นการสื่อสารแบบหลายผู้ใช้ต่อหนึ่งคอมพิวเตอร์หลัก และแบบส่วนบุคคล (Personal Connectivity) หรือเรียกว่าเป็นการสื่อสารแบบหนึ่งผู้ใช้ต่อหนึ่งคอมพิวเตอร์หลัก

2.3.1 การต่อเชื่อมแบบเป็นกลุ่ม (Fleet Connectivity)

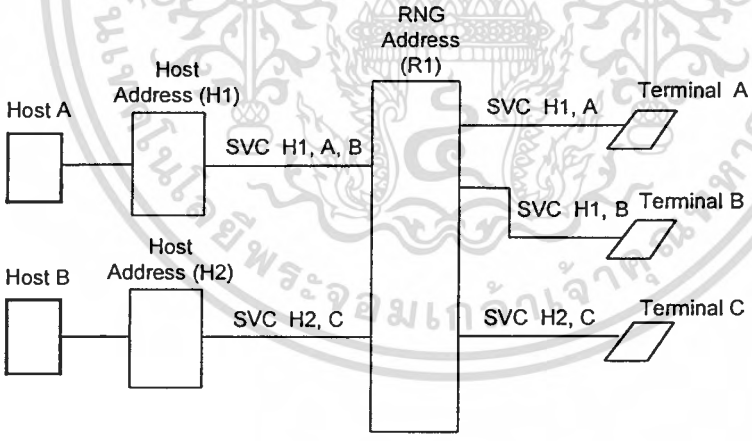
การเชื่อมต่อแบบนี้ต้องการเพียงหนึ่งช่องการเชื่อมโยงทางลอจิก (Logical link) เท่านั้นระหว่างคอมพิวเตอร์หลักและโครงข่าย DataTAC สำหรับการให้บริการเครื่องผู้ใช้ลักษณะกลุ่มใหญ่ ๆ ซึ่งจะเหมาะกับงานประเภทติดต่อธุรกิจนอกสำนักงานและออกภาคสนาม เช่น กิจการรถแท็กซี่ กิจการรับส่งข่าวสาร กิจการขนส่งสินค้า เดินรถและเดินเรือ เป็นต้น ดูรูปที่ 2.3 แสดงลักษณะการเชื่อมต่อแบบเป็นกลุ่ม (Fleet Connectivity)

การสื่อสารแบบเป็นกลุ่มระหว่างคอมพิวเตอร์หลักและโครงข่าย DataTAC ใช้โปรโตคอลสื่อสารที่เรียกว่า Standard Context Routing (SCR) ซึ่งเป็นโปรโตคอลที่พัฒนาโดยบริษัทโมโตโลร่าเป็นโปรโตคอลระดับสูง (High-Level protocol) ที่ใช้งานอยู่บนโปรโตคอลมาตรฐาน TCP/IP และ X.25 ทั้งแบบ SVC และ PVC ซึ่งจะกล่าวรายละเอียดอีกครั้งหนึ่งในบทที่ 3 รูปที่ 2.4 แสดงลักษณะการเชื่อมต่อแบบเป็นกลุ่มบนโปรโตคอล X.25 แบบ SVC

รูปที่ 2.3 แสดงลักษณะการเชื่อมต่อแบบเป็นกลุ่ม (Fleet Connectivity)



รูปที่ 2.4 แสดงลักษณะการเชื่อมต่อแบบเป็นกลุ่มบนโปรโตคอล X.25 แบบ SVC



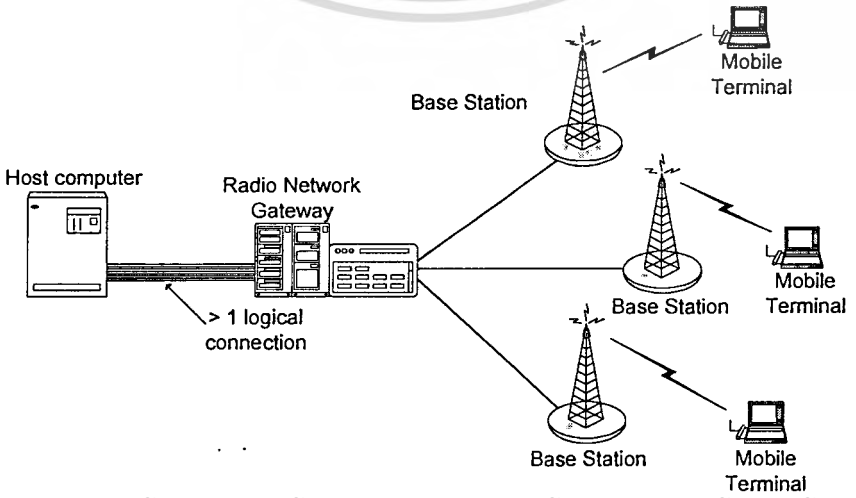
จากรูปที่ 2.4 จะเห็นว่าคอมพิวเตอร์ปลายทาง (Terminal) A และ B เชื่อมต่อกับคอมพิวเตอร์หลัก (Host) A ในทำนองเดียวกันคอมพิวเตอร์ปลายทาง C เชื่อมต่ออยู่กับคอมพิวเตอร์หลัก B ซึ่งในแต่ละคอมพิวเตอร์หลักสามารถรองรับการเชื่อมต่อจากคอมพิวเตอร์ปลายทางได้ครั้งละหลายตัว โดยอาศัยการเชื่อมต่อระหว่างเกตเวย์โครงข่ายวิทยุ (RNG) และคอมพิวเตอร์หลักแต่ละตัวเพียงหนึ่งช่องสัญญาณทางลอจิกเท่านั้น และการเชื่อมต่อระหว่างเกตเวย์โครงข่ายวิทยุและคอมพิวเตอร์หลักจะต้องเกิดขึ้นก่อนที่เครื่องผู้ใช้จะทำการติดต่อสื่อสารด้วย ซึ่งฝ่ายหนึ่งฝ่ายใดสามารถเป็นผู้ร้องขอ (Request) การเชื่อมต่อได้ตามหมายเลขตำบลที่อยู่ (Address) ที่กำหนด ใช้งาน

อยู่บนโปรโตคอลมาตรฐาน X-25 แบบวงจรสลับช่องสัญญาณชั่วคราว (SVC) สำหรับการใช้งานรับส่งข้อมูลของคอมพิวเตอร์ปลายทาง (Terminal) นั้น จะใช้หมายเลขเซสชั่น (Session ID) เป็นตัวกำหนดการใช้งานของแต่ละโปรแกรมระบบงาน (Application) โดยมีเกตเวย์โครงข่ายวิทยุทำหน้าที่คัดเลือกการผ่านข้อมูลไปยังระบบงานต่างๆ ตามที่ระบุในหมายเลขเซสชั่นของแต่ละเฟรมข้อมูล ซึ่งอุปกรณ์คอมพิวเตอร์ปลายทางจะต้องมีการลงทะเบียนหมายเลขตำแหน่งที่อยู่ประจำเครื่อง (Logical Line Identifier หรือ LLI) ที่เกตเวย์โครงข่ายวิทยุไว้ก่อน และกำหนดว่ามีการใช้งานกับระบบงานไหนบ้าง การส่งข้อมูลจากคอมพิวเตอร์หลักสามารถส่งข้อมูลไปที่อุปกรณ์ปลายทางได้หลายเครื่องในเวลาเดียวกัน หรือเรียกว่าเป็นการส่งข้อมูลแบบเป็นกลุ่มนั่นเอง

2.3.2 การเชื่อมต่อแบบส่วนบุคคล (Personal Connectivity)

ลักษณะการเชื่อมต่อแบบนี้จะแตกต่างกับการเชื่อมต่อแบบกลุ่มคือ การเชื่อมต่อจะใช้ช่องสัญญาณระหว่างคอมพิวเตอร์หลักกับโครงข่าย DataTAC แบบหนึ่งต่อหนึ่งของเครื่องผู้ใช้แต่ละคน ทำให้มีการจองช่องสัญญาณมากกว่าหนึ่งช่องสัญญาณ และไม่มีโปรโตคอลระดับสูง (High-level Protocol) ช่วยจัดการแลกเปลี่ยนข้อมูล จะมีเฉพาะโปรโตคอลมาตรฐาน TCP/IP และ X.25 เท่านั้น ทำให้การเชื่อมต่อแบบนี้เหมาะกับงานประเภทอิเล็กทรอนิกส์เมล์ (E-Mail) การเข้าถึงข้อมูลแบบออนไลน์ (Online Database access) และการสั่งซื้อสินค้าทางไกล (Remote order) เป็นต้น แต่วิธีการเชื่อมต่อแบบส่วนบุคคลนี้มีข้อจำกัด เมื่อมีเครื่องผู้ใช้มาก จะทำให้ช่องสัญญาณไม่พอ เนื่องจากเครื่องผู้ใช้แต่ละคนต้องจองช่องสัญญาณคนละหนึ่งช่องสัญญาณ ในรูปที่ 2.5 แสดงลักษณะเชื่อมต่อแบบส่วนบุคคล (Personal Connectivity)

รูปที่ 2.5 แสดงลักษณะเชื่อมต่อแบบส่วนบุคคล (Personal Connectivity)

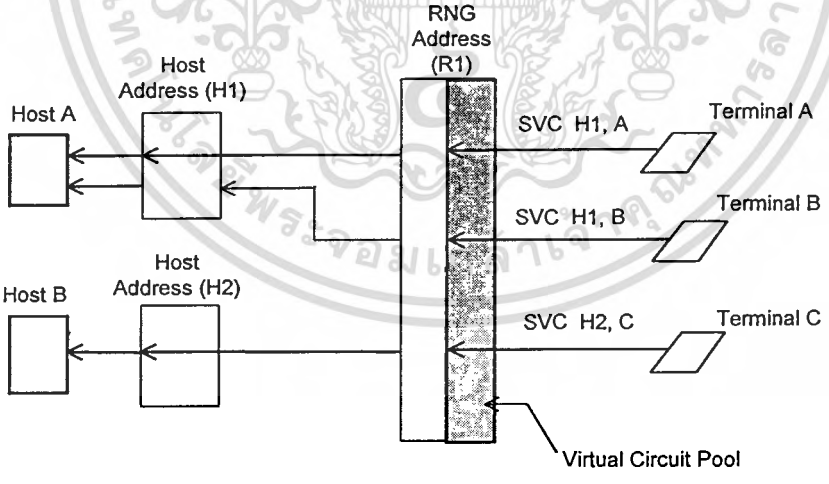


การเชื่อมต่อแบบส่วนบุคคลนี้ไม่มีบริการแบบกระจายข่าว (Broadcast) หรือการส่งข่าวสารแบบเป็นกลุ่ม การบริการแบบนี้เปรียบได้คล้าย ๆ กับบริการแบบจุดต่อจุดของการติดต่อบนโครงข่ายโทรศัพท์โดยใช้โมเด็ม สามารถแบ่งการเชื่อมต่อแบบส่วนบุคคลนี้ออกเป็น 2 ประเภท คือ แบบใช้ช่องสัญญาณร่วมกัน (Shared personal connectivity) และแบบจองช่องสัญญาณ (Dedicated personal connectivity)

2.3.2.1 การเชื่อมต่อแบบใช้ช่องสัญญาณร่วมกัน (Shared personal connectivity)

การทำงานจะเกิดขึ้นโดยอุปกรณ์ของเครื่องผู้ใช้เป็นผู้เริ่มร้องขอการเชื่อมต่อกันระหว่างคอมพิวเตอร์หลักและโครงข่าย DataTAC อาจจะเป็นโครงข่ายโปรโตคอลมาตรฐาน X.25 หรือ TCP/IP เมื่อเครื่องผู้ใช้ร้องขอไปยังโครงข่าย DataTAC ให้ทำงานเชื่อมต่อคอมพิวเตอร์หลัก เครื่องผู้ใช้จะได้รับหมายเลขประจำโครงข่ายผู้ใช้งาน (Network User Address) จากโครงข่าย DataTAC เพื่อจะได้ส่งข้อมูลกลับจากคอมพิวเตอร์หลักไปยังเครื่องผู้ใช้งาน หมายเลขประจำโครงข่ายนี้จะถูกจัดสรรให้ใช้ร่วมกันระหว่างอุปกรณ์เครื่องผู้ใช้ทั้งหมด ในรูปที่ 2.6 ลักษณะการเชื่อมต่อแบบใช้ช่องสัญญาณร่วมกัน

รูปที่ 2.6 ลักษณะการเชื่อมต่อแบบใช้ช่องสัญญาณร่วมกัน

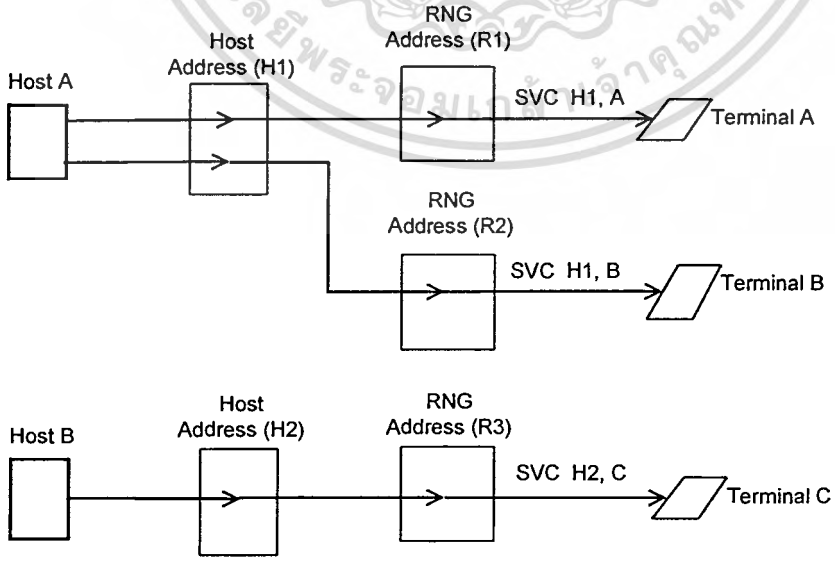


จากรูปที่ 2.6 จะเห็นว่าคอมพิวเตอร์ปลายทาง (Terminal) A, B และ C จะใช้วงจรสลับช่องสัญญาณ (SVC) เดียวกัน โดยมีเกตเวย์โครงข่ายวิทยุ (RNG) ทำหน้าที่จัดสรรหมายเลขประจำโครงข่ายให้กับคอมพิวเตอร์ปลายทางทั้งหมด และการเชื่อมโยงทางลอจิกระหว่างคอมพิวเตอร์หลักและคอมพิวเตอร์ปลายทางจะมีมากกว่าหนึ่งช่องการเชื่อมโยง คือการเชื่อมโยงจากคอมพิวเตอร์ปลายทาง A ไปยังคอมพิวเตอร์หลัก A หนึ่งช่องสัญญาณ และการเชื่อมโยงจากคอมพิวเตอร์ปลายทาง B ไปยังคอมพิวเตอร์หลัก A หนึ่งช่องสัญญาณ

2.3.2.2 การเชื่อมต่อแบบการจองช่องสัญญาณ (Dedicated personal connectivity)

การเชื่อมต่อแบบนี้อุปกรณ์เครื่องผู้ใช้จะมีการลงทะเบียนหมายเลขตำแหน่งที่อยู่ประจำเครื่อง (Logical Link Identifier หรือ LLI) ที่ฝังอยู่บนอุปกรณ์โมเด็มไร้สาย ลงบนโครงข่าย DataTAC เพื่อทำการจับคู่กับหมายเลขโครงข่าย (Network address) ถ้าเป็นการเชื่อมต่อโดยใช้โปรโตคอลมาตรฐาน TCP/IP จะใช้ IP address และหมายเลขพอร์ต แต่ถ้าเป็นโปรโตคอลมาตรฐาน X.25 จะใช้หมายเลขประจำโครงข่าย ซึ่งในการเริ่มต้นทำงานฝ่ายคอมพิวเตอร์หลักจะเป็นผู้จัดหาหมายเลขโครงข่ายที่เราได้จับคู่ไว้กับหมายเลขตำแหน่งที่อยู่ประจำเครื่องส่งไปยังเครื่องผู้ใช้ เพื่อขอติดต่อสื่อสารระหว่างกัน และจะจองช่องสัญญาณนั้นไว้ตลอดเวลาจนกว่าจะสั่งยกเลิกการติดต่อ ในรูปที่ 2.7 แสดงลักษณะการเชื่อมต่อแบบจองช่องสัญญาณ

รูปที่ 2.7 ลักษณะการเชื่อมต่อแบบจองช่องสัญญาณ



จากรูปที่ 2.7 จะเห็นว่าคอมพิวเตอร์ปลายทาง A, B และ C จะจองหมายเลขโครงข่ายที่ RNG และช่องสัญญาณเป็นอิสระจากกันในการเชื่อมโยงระหว่างเกตเวย์โครงข่ายวิทยุ (RNG) และคอมพิวเตอร์หลัก โดยที่เกตเวย์โครงข่ายวิทยุจะทำหน้าที่จัดสรรหมายเลขประจำตำบลที่อยู่ (Address) ของค่าที่มีการสื่อสารระหว่างกัน

สำหรับในวิทยานิพนธ์ฉบับนี้ได้ใช้ลักษณะการเชื่อมต่อแบบเป็นกลุ่ม สำหรับการใช้งานโปรแกรมรับส่งข้อมูลบนโครงข่ายเวิลด์คาต้า และในปัจจุบันบนระบบโครงข่ายเวิลด์คาต้ายังไม่เปิดให้บริการการเชื่อมต่อแบบส่วนบุคคล

2.4 ความสามารถในการรับส่งข้อมูลบนระบบโครงข่าย

ความสามารถของระบบ (System capacity) ถูกกำหนดขึ้นมาจากปริมาณข้อมูล (Throughput) ที่ระบบประมวลผลได้ หรือปริมาณการส่งผ่านข้อมูล โดยเทียบกับเวลาที่ใช้ ทั้งนี้ความสามารถของระบบจะขึ้นอยู่กับความสามารถของอุปกรณ์ที่รองรับแต่ละตัว ดังนี้

2.4.1 ความสามารถของช่องสัญญาณวิทยุ (Radio channel capacity)

2.4.2 ความสามารถของอุปกรณ์ควบคุมโครงข่ายคลื่นวิทยุ (RNC capacity)

2.4.3 ความสามารถของอุปกรณ์เกตเวย์โครงข่ายวิทยุ (RNG capacity)

2.4.4 ความสามารถของการเชื่อมโยงคอมพิวเตอร์ (Host link capacity)

การประเมินค่าความสามารถของระบบ เรากำหนดรูปแบบของข้อมูลมาตรฐาน (Standard message profile) 2 ชนิดคือ ข้อมูลต่อหนึ่งแพ็คเก็ตที่ส่งออกจากโมเด็มไร้สายมีจำนวน 50 ตัวอักษร และข้อมูลต่อหนึ่งแพ็คเก็ตที่รับเข้าโมเด็มไร้สายมีจำนวน 100 ตัวอักษร มีข้อมูลผิดพลาดขณะทำการรับส่งบนช่องสัญญาณวิทยุประมาณ 5 เปอร์เซ็นต์ และให้ Workload ของโครงข่ายมีสถานะคงที่ (Steady state) โดยที่เวลาของการตอบสนองข้อมูล (Message response time) หรือเวลาขนส่งข้อมูล (Message transit time) จะเป็นการรวมกันของขบวนการรับส่งข้อมูลที่ผ่านอุปกรณ์แต่ละตัวจากต้นทางไปยังปลายทาง และมีปัจจัยของการล่าช้าของข้อมูลเข้ามาเกี่ยวข้องด้วย ซึ่งอาจเกิดมาจากวิธีการเข้าถึงช่องสัญญาณวิทยุ การส่งข้อมูลซ้ำเมื่อมีการสูญหาย ขบวนการประมวลผล และการเข้าแถวคอย (Queueing) เป็นต้น นอกจากนี้ถ้าเฟรมข้อมูลที่รับส่งที่มีความยาวตั้งแต่ 513 ถึง 2048 ไบต์ จะถูกแบ่งแยกเป็นแพ็คเก็ตย่อยๆ หลายแพ็คเก็ต ซึ่งในหนึ่งแพ็คเก็ตย่อยมีความยาวสูงสุด 512 ไบต์ การทำเช่นนี้ก็เพื่อให้การรับส่งข้อมูลมีประสิทธิภาพบนช่องสัญญาณวิทยุมากขึ้น แต่ขบวนการนี้มีผลให้เกิดการล่าช้าของข้อมูลด้วย ซึ่งเวลาการส่งข้อมูลโดยรวมต่อเฟรมข้อมูลระหว่างคอมพิวเตอร์หลัก (Host) และอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ (Mobile terminal) จะไม่เกิน 4 วินาที แต่ถ้าระหว่างอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ด้วยกันเวลาโดยรวมอาจมากกว่านี้ เนื่องจากต้องผ่านอุปกรณ์ที่อยู่บน

โครงข่ายมากกว่านั่นเอง จากข้อกำหนดและข้อพิจารณาดังกล่าวเราสามารถแสดงถึงความสามารถของระบบ DataTAC ได้ในตารางที่ 2.1

ตารางที่ 2.1 แสดงความสามารถของระบบ DataTAC

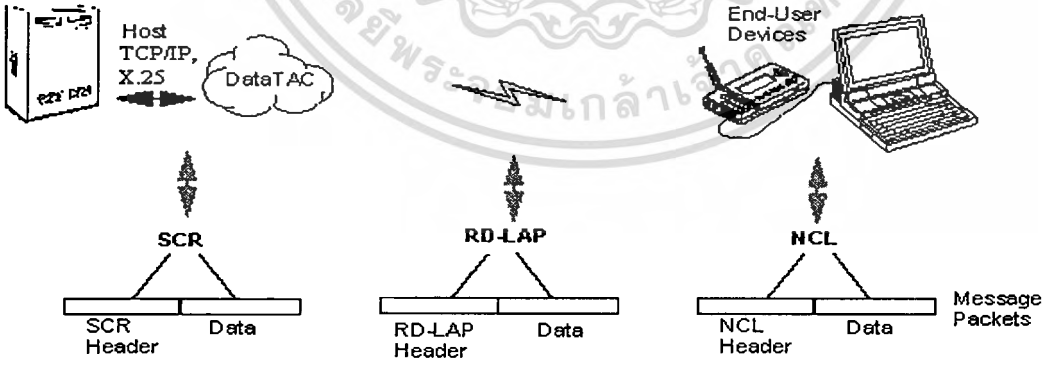
อุปกรณ์หรือสายเชื่อมโยง	จำนวนของสายเชื่อมโยงทางกายภาพที่มีได้	ค่าความสามารถต่อหนึ่งสายเชื่อมโยง	หมายเหตุ
ช่องสัญญาณวิทยุ 19.2 kbps	สูงสุด 63 สถานีฐาน กับ 12 –slot RNC	24,400 ข้อความต่อชั่วโมง	อยู่บนพื้นฐานของข้อมูลมาตรฐาน
RNC	-	100,000 ข้อความต่อชั่วโมง	อยู่บนพื้นฐานของข้อมูลมาตรฐาน
RNG	-	200,000 ข้อความต่อชั่วโมง	อยู่บนพื้นฐานของข้อมูลมาตรฐาน
การเชื่อมโยง Host แบบ TCP/IP	1	10 Mbps Ethernet 200 fleet connections	Non-blocking
การเชื่อมโยง Host แบบ X.25	9	64 kbps 720 วงจรเสมือน(VC)	-

จากตารางที่ 2.1 อ้างอิงข้อมูลมาจาก [6] สามารถอธิบายได้ว่าปริมาณข้อมูลสูงสุดที่ได้รับจากช่องสัญญาณวิทยุเท่ากับ 24,400 ข้อความต่อชั่วโมง เมื่อเราคิดจากความยาวข้อมูลมาตรฐานเท่ากับ 100 ตัวอักษรต่อข้อความ เราจะได้ปริมาณข้อมูลที่ช่องสัญญาณวิทยุในหน่วยบิตต่อวินาทีคือ (24,400x100)/3600 เท่ากับ 5.4 kbps สำหรับความสามารถในการประมวลผลของอุปกรณ์ควบคุมโครงข่ายคลื่นวิทยุ (RNC) เท่ากับ 22 kbps และอุปกรณ์เกตเวย์โครงข่ายวิทยุ (RNG) เท่ากับ 44 kbps ส่วนปริมาณข้อมูลของการเชื่อมโยง Host นั้น จะขึ้นอยู่กับชนิดของการเชื่อมต่อ เช่น การเชื่อมต่อแบบอีเธอร์เน็ตจะให้ความเร็ว 10 Mbps และการเชื่อมต่อแบบ X.25 จะให้ความเร็วที่ 64 kbps เป็นต้น

บทที่ 3
โปรโตคอลสื่อสารบนระบบโครงข่ายเวิลด์ดาต้า

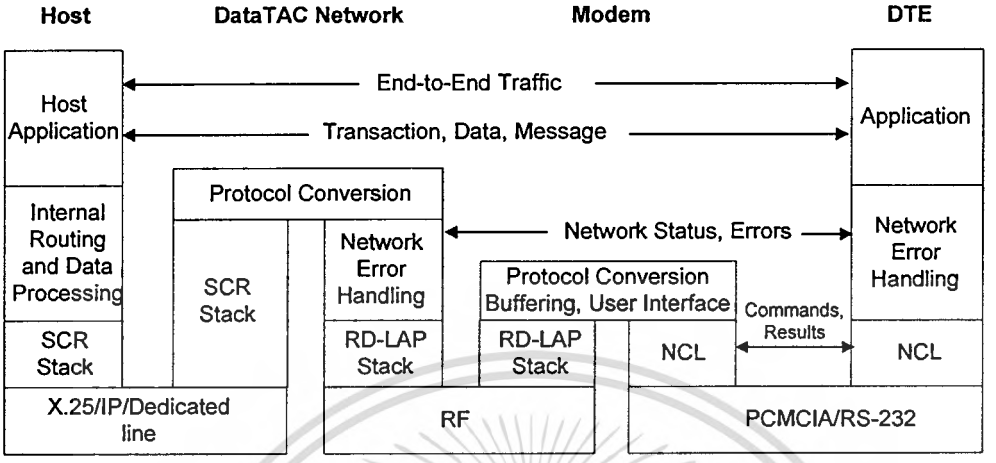
การแลกเปลี่ยนข้อมูลข่าวสารระหว่างกันทั้งผู้ผลิตเดียวกัน หรือต่างผู้ผลิต จำเป็นต้องกำหนดมาตรฐาน เพื่อควบคุมให้การสื่อสารเชื่อมโยงกันได้โดยสมบูรณ์ มีการแบ่งเป็นระดับชั้นต่าง ๆ กัน แต่ละระดับชั้นมีหน้าที่ของตนเองโดยเฉพาะ และทำหน้าที่ตามมาตรฐานที่กำหนดไว้ ในบางผู้ผลิตระบบสื่อสารข้อมูลได้กำหนดมาตรฐานมาเฉพาะ เพื่อใช้งานกับผลิตภัณฑ์ของตนเอง สำหรับโครงข่ายเวิลด์ดาต้านี้ ได้กำหนดรูปแบบของโปรโตคอลขึ้นมาในลักษณะเฉพาะคือ โปรโตคอล SCR สำหรับการสื่อสารระหว่างคอมพิวเตอร์หลักและโครงข่าย DataTAC โดยมีการใช้งานร่วมกับโปรโตคอลมาตรฐาน X.25 และ TCP/IP โปรโตคอล RD-LAP สำหรับการสื่อสารระหว่างโมเด็มไร้สายและสถานีฐานของโครงข่าย DataTAC และโปรโตคอล NCL สำหรับการสื่อสารระหว่างอุปกรณ์คอมพิวเตอร์ปลายทางและโมเด็มไร้สาย ดังแสดงในรูปที่ 3.1

รูปที่ 3.1 แสดงโปรโตคอลที่ใช้บนระบบโครงข่ายเวิลด์ดาต้า



เราสามารถแสดงโปรโตคอลที่ใช้บนโครงข่ายเวิลด์ดาต้าจากรูปที่ 3.1 ให้อยู่ในรูปแบบของโปรโตคอลสแตคดังในรูปที่ 3.2 เพื่อแสดงถึงรายละเอียดแต่ละชั้นการเชื่อมต่อระหว่างโปรโตคอลต่าง ๆ ตั้งแต่ระดับชั้นทางกายภาพจนถึงระดับชั้นโปรแกรมประยุกต์

รูปที่ 3.2 แสดงลักษณะของโปรโตคอลสแตค (Protocol stack) บนระบบโครงข่ายเวิลด์ดาต้า



จากรูปที่ 3.2 ในการเชื่อมโยงทางแนวนอนจะต้องมีการติดต่อสื่อสารด้วยโปรโตคอลเดียวกัน เมื่อมีการข้ามระบบจะมีการแปลง (Conversion) ให้เป็นโปรโตคอลเดียวกันก่อนทำการแลกเปลี่ยนข้อมูล ซึ่งในส่วนของโปรโตคอล NCL และ SCR นั้น ทางบริษัทผู้ผลิตจะมี API(Application Programming Interface) มาให้ เพื่อให้นักพัฒนาโปรแกรมสามารถนำไปใช้ได้ง่ายและรวดเร็ว

3.1 คุณสมบัติโปรโตคอลมาตรฐาน

3.1.1 แบบจำลอง OSI (OSI model)

เป็นรูปแบบของโปรโตคอลที่ถูกพัฒนาขึ้นมาโดย International Standards Organization (ISO) ที่เป็นมาตรฐานสากลเป็นกฎเกณฑ์กำหนดโปรโตคอลที่มีอยู่มากมายหลายชนิด โดยตั้งชื่อว่า Open Systems International (OSI) มีด้วยกัน 7 ระดับชั้น (Layer) ดังแสดงในรูปที่ 3.3

รูปที่ 3.3 โครงสร้างแบบจำลอง OSI

7	Application
6	Presentation
5	Session
4	Transport
3	Network
2	Data Link
1	Physical

3.1.1.1 ระดับชั้นทางกายภาพ (Physical layer)

ในระดับชั้นนี้จะเกี่ยวข้องกับการพิจารณาการส่งข้อมูลดิบ คือ พิจารณาในแง่ของบิตที่ส่งผ่านช่องทางการสื่อสาร แรงดันไฟฟ้าเป็นกี่โวลต์ เมื่อแทนด้วยบิต "1" และกี่โวลต์ เมื่อแทนด้วยบิต "0" มีการบอกสภาวะจุดเริ่มต้นและสิ้นสุดของการส่งผ่านข้อมูล และต้องมีการกำหนดมาตรฐานขึ้นมาว่าปลั๊ก (Socket) ที่ใช้เชื่อมโยงโครงข่ายจะต้องมีกี่ขา ดังนั้นในการออกแบบจึงพิจารณาให้ครอบคลุมไปถึงกลไกด้านกำลังไฟฟ้า และส่วนต่อเชื่อมกันเป็นโครงข่าย

3.1.1.2 ระดับชั้นเชื่อมโยงข้อมูล (Data link layer)

ทำหน้าที่อำนวยการส่งข้อมูลเข้าสู่โครงข่ายสื่อสาร โดยปราศจากข้อผิดพลาด คือ มีการบังคับการไหลของข้อมูล (Flow control) จัดลำดับ (Sequencing) การส่งกรอบข้อมูล (Data frame) ตรวจสอบข้อผิดพลาดของข้อมูล และนำกลับข้อมูลให้ถูกต้อง

3.1.1.3 ระดับชั้นโครงข่ายสื่อสาร (Network layer)

ทำหน้าที่ควบคุมการสื่อสารผ่านข้อมูลในโครงข่าย โดยการกำหนดเส้นทาง (Routing) ของกรอบข้อมูลโดยการตรวจสอบตำแหน่งที่อยู่ (Addressing) ตลอดจนจัดกลุ่มข้อมูลข่าวสาร และรวมกลุ่มข้อมูลเข้าด้วยกัน แล้วกำหนดว่าข้อมูลนี้จะส่งผ่านไประดับชั้นขนส่งข้อมูล Ttransport layer) ภายในโหนดตัวเอง หรือจะส่งผ่านข้อมูลผ่านไปยังระดับชั้นเชื่อมโยงข้อมูล(Data link layer) เพื่อส่งข้อมูลไปยังโหนดต่างๆ

3.1.1.4 ระดับชั้นการขนส่งข้อมูล (Transport layer)

ระดับชั้นที่ทำหน้าที่กำหนดตำแหน่งที่อยู่ของการขนส่งข้อมูลที่ไม่ซ้ำกันให้แต่ละผู้ใช้งาน (User) จำนวนของการเชื่อมต่อสำหรับการขนส่งข้อมูล ตรวจสอบลำดับของกรอบข้อมูล (Frame) และบังคับการไหลของข้อมูลไม่ให้มีการสูญหายระหว่างรับส่ง

3.1.1.5 ระดับชั้นการโต้ตอบระหว่างกัน (Session layer)

เป็นระดับชั้นที่ผู้ใช้ทำการติดต่อกับโครงข่ายสื่อสาร โดยมีการควบคุมการเริ่มต้นติดต่อ (Establishment) ของโปรแกรมใช้งาน (Application) รวมถึงการดูแลการคงอยู่ (Maintain) และการปิด (Close) ยกเลิกการเชื่อมต่อของโปรแกรมใช้งาน ตลอดจนกำหนดรูปแบบของการรับส่งข้อมูลในลักษณะสื่อสารแบบทิศทางเดียว (One-way) หรือมีการสื่อสารแบบสองทิศทาง (Two-way)

3.1.1.6 ระดับชั้นการแสดงผล (Presentation layer)

ระดับชั้นนี้มีหน้าที่เกี่ยวกับการรับข้อมูลในลักษณะต่าง ๆ กัน และตีความให้สามารถเข้าใจกันได้ในระดับโปรแกรมใช้งานที่ติดต่อสื่อสารกัน ทั้งนี้ยังสามารถเปลี่ยนแปลงข้อมูลกลับไปกลับมาให้ได้ตามสิ่งที่ต้องการ อาจจะในรูปของรหัสแอสกี (ASCII) หรือ EBCDIC ก็ได้

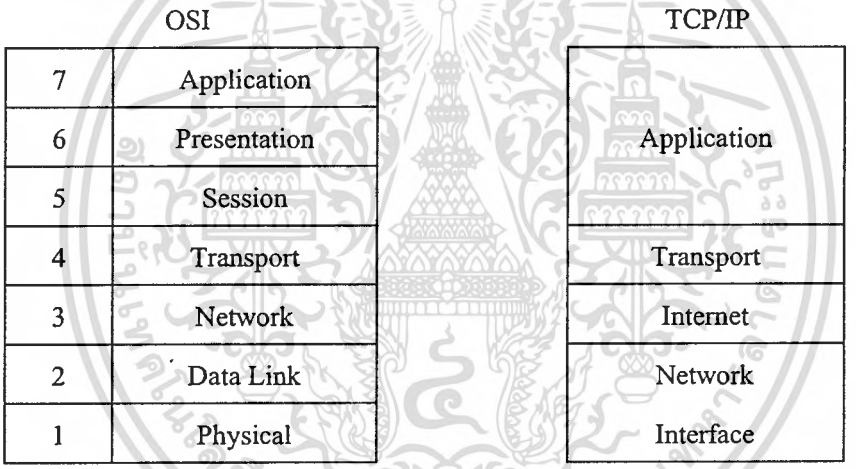
3.1.1.7 ระดับชั้นการประยุกต์ใช้งาน (Application layer)

ในระดับชั้นนี้จะครอบคลุมถึงระดับผู้ใช้ที่จะประยุกต์ใช้งานของแต่ละคน เช่น เมื่อผู้ใช้งานสองคนจัดทำโปรแกรมต่างเครื่องกัน ซึ่งทั้งสองฝ่ายสามารถกำหนดรูปแบบของข้อความเพื่อทำงานระหว่างกันและกัน โดยไม่ต้องคำนึงถึงว่าผู้ใช้อีกฝ่ายหนึ่งที่ติดต่อด้วยใช้เครื่องชนิดอะไร

3.1.2 โปรโตคอลมาตรฐาน TCP/IP

โปรโตคอล TCP/IP คือข้อตกลงหรือวิธีการในการส่งข้อมูลประเภทหนึ่ง เป็นมาตรฐานของ Defacto ไม่ใช่ของหน่วยงานมาตรฐานสากล (ISO) ที่กำหนดมาตรฐาน OSI ขึ้นมา ซึ่งเราสามารถเปรียบเทียบแบบจำลองของ OSI 7 ระดับชั้น และ TCP/IP ดังแสดงในรูปที่ 3.4

รูปที่ 3.4 เปรียบเทียบแบบจำลองของ OSI 7 ระดับชั้น และ TCP/IP



3.1.2.1 ระดับชั้นโครงข่ายสื่อสาร (Network Interface)

เป็นการรวมระดับชั้นทางกายภาพ และระดับชั้นการเชื่อมโยงเข้าด้วยกัน โดยทำหน้าที่กำหนดคุณสมบัติทางสัญญาณไฟฟ้า มาตรฐานรูปแบบการเชื่อมต่อสัญญาณ เช่น การเชื่อมต่อแบบอีเธอร์เน็ต (Ethernet) X.25, ATM และ Serial line เป็นต้น

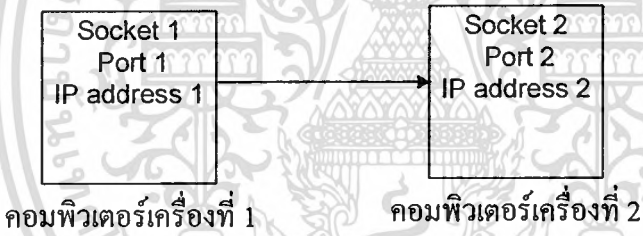
3.1.2.2 ระดับชั้นอินเตอร์เน็ต (Internet)

ทำหน้าที่ควบคุมการส่งข้อมูลไปยังผู้รับให้ถูกต้องครบถ้วน โดยกำหนดตำบลที่อยู่ของผู้ส่งและผู้รับที่เราเรียกว่า IP address ซึ่งจะต้องมีค่าไม่ซ้ำกันในแต่ละตำบลที่อยู่ของเครื่องคอมพิวเตอร์ กรณีที่ข้อมูลมีความยาวมาก ๆ การรับส่งข้อมูลระหว่างกันจะมีการจัดการแบ่งแยก (Fragmentation) ข้อมูล เมื่อไปถึงผู้รับจะทำการรวบรวม (Reassembly) ข้อมูลนั้นๆ เข้าด้วยกันอีกทีหนึ่ง เพื่อให้การรับส่งข้อมูลมีประสิทธิภาพมากขึ้นในโครงข่ายสื่อสารที่มีแบนด์วิตไม่มากนัก

3.1.2.3 ระดับชั้นการขนส่งข้อมูล (Transport)

เป็นการบริการความน่าเชื่อถือระดับปลายของคอมพิวเตอร์ทั้งสองข้าง (End-to-End) ผ่านระบบโครงข่ายสื่อสาร มีอยู่ 2 รูปแบบ คือ Transmission Control Protocol (TCP) และ User Datagram Protocol (UDP) คุณสมบัติของ TCP จะรักษาการเชื่อมโยงไว้ตลอดที่มีการใช้งานถึงไม่มีข้อมูลรับส่งก็ตาม เราเรียกว่าเป็นลักษณะของ Connection-oriented ข้อมูลรับส่งเป็นแบบต่อเนื่อง (Stream data) ขนาดความยาวของข้อมูลไม่คงที่ ขึ้นอยู่กับข้อมูลและจังหวะเวลาในการรับส่ง มีการควบคุมการไหลผ่าน (Flow control) ข้อมูล ตรวจสอบความผิดพลาด (Error detection) และลำดับ (Sequencing) ข้อมูล ไม่ให้มีการสูญหายในระหว่างรับส่งข้อมูล การเชื่อมต่อ (Connection) ระหว่างกัน จะสร้างซ็อคเก็ต (Socket) ซึ่งเป็นช่องทางสำหรับรับส่งข้อมูลขึ้นมาในแต่ละครั้งที่มีการใช้งานทั้งสองข้าง โดยมีการระบุหมายเลขพอร์ต (Port number) และ IP address ดังแสดงในรูปที่ 3.5

รูปที่ 3.5 การเชื่อมต่อในรูปแบบ TCP



การระบุหมายเลขพอร์ตที่รู้จักกันดีในโปรแกรมใช้งาน เช่น FTP (File Transfer Protocol) มีหมายเลขพอร์ต คือ 21 และ Telnet มีหมายเลขพอร์ต คือ 23 สำหรับโปรแกรมประยุกต์ที่เราสร้างขึ้นมาใช้งานเองสามารถใช้หมายเลขพอร์ตอะไรก็ได้ แต่ต้องไม่ซ้ำกับหมายเลขพอร์ตมาตรฐานที่ถูกกำหนดใช้งานแล้ว

สำหรับคุณสมบัติของ UDP จะไม่รักษาการเชื่อมโยงไว้ตลอดการใช้งาน ที่เรียกกันว่า Connectionless แต่จะสร้างการเชื่อมต่อเฉพาะที่มีการร้องขอ (Request) และตอบกลับตามการร้องขอ (Reply request) เท่านั้น ทำให้โปรแกรมที่ใช้ UDP มี Overhead ต่ำ ใช้หน่วยความจำน้อย สามารถที่จะนำโปรแกรมไปใส่ไว้ใน FLASH memory ได้ แต่โปรแกรมประเภทนี้มีความน่าเชื่อถือต่ำ ไม่มีการตรวจสอบข้อผิดพลาดลำดับข้อมูลและการ Handshaking ทำให้ไม่เหมาะกับโปรแกรมที่ต้องการความน่าเชื่อถือ และต้องการความถูกต้องสูง สำหรับโปรแกรมที่ใช้คุณสมบัติของ UDP เช่น SNMP (Simple Network Management Protocol), BOOTP (BOOT Protocol), DNS (Domain Name Service), TFTP (Trivial File Transfer Protocol) และ NFS (Network File System) เป็นต้น

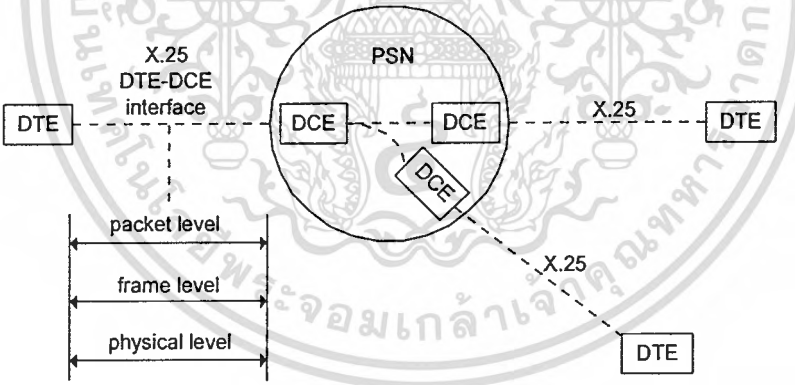
3.1.2.4 ระดับชั้นโปรแกรมประยุกต์ (Application)

เป็นการรวม 3 ระดับชั้นของ OSI 7 ระดับชั้นเข้าด้วยกัน คือ ระดับชั้นการโต้ตอบระหว่างกัน ระดับชั้นการแสดงผล และระดับชั้นการประยุกต์ใช้งาน มีหน้าที่กำหนดรูปแบบและโครงสร้างในการแลกเปลี่ยนข้อมูลระหว่างกัน ซึ่งในแต่ละโปรแกรมประยุกต์จะมีคุณลักษณะการใช้งานแตกต่างกันออกไป โปรแกรมประยุกต์ที่เรารู้จักกันดี เช่น FTP (File Transfer Protocol), Telnet และ SNMP (Simple Network Management Protocol)

3.1.3 โปรโตคอลมาตรฐาน X.25

เป็นโปรโตคอลที่ถูกกำหนดมาตรฐานโดย CCITT เพื่อให้มีการเชื่อมโยงกันระหว่างอุปกรณ์รับส่งข้อมูลปลายทาง (Data Terminal Equipment หรือ DTE) และโครงข่ายสลับช่องแพ็คเก็ต (Packet Switching Network หรือ PSN) หรือเราเรียกว่าเป็นอุปกรณ์สื่อสารข้อมูล (Data Communication Equipment หรือ DCE) ให้สามารถส่งข้อมูลผ่านกันได้ ดังแสดงในรูปที่ 3.6

รูปที่ 3.6 รายละเอียดการเชื่อมโยงอุปกรณ์ DTE และ DCE



โปรโตคอล X.25 เป็นโปรโตคอลที่ประกอบด้วย 3 ระดับชั้น คือ

3.1.3.1 ระดับชั้นทางกายภาพ (Physical layer)

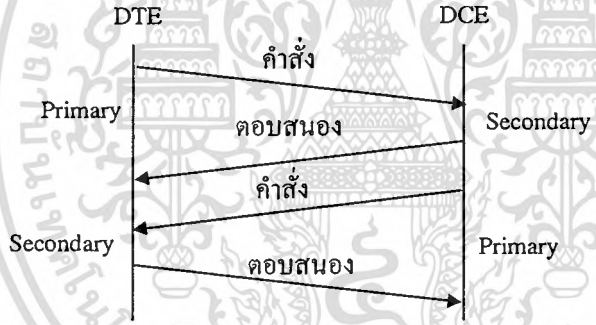
เป็นระดับต่ำสุดทางด้านฮาร์ดแวร์ เพื่อก่อให้เกิดการเชื่อมโยงทางกายภาพระหว่างอุปกรณ์รับส่งข้อมูลปลายทาง และโครงข่ายสลับช่องแพ็คเก็ต สำหรับคุณลักษณะเฉพาะของการเชื่อมโยงชนิดนี้จะเป็นไปตามมาตรฐาน X.21 หรือ X.21 bis ของ CCITT

3.1.3.2 ระดับชั้นการเชื่อมโยง (Link layer) หรือระดับเฟรม (Frame level)

เป็นส่วนกำหนดกฎเกณฑ์วิธีการเข้ามาเชื่อมโยง (Link Access Procedure หรือ LAP) เพื่อให้มีการแลกเปลี่ยนข้อมูลกันระหว่าง DTE และ DCE มาตรฐานแบบหนึ่งที่ใช้ในระดับชั้นนี้คือ

HDLC (High level Data Link Control) กำหนดมาตรฐานโดยองค์กรกำหนดมาตรฐานระหว่างประเทศ หรือ ISO จากนั้นได้มีการปรับปรุง HDLC ใหม่โดยคณะกรรมการสื่อสารสากลโทรศัพท์และโทรเลข หรือ CCITT ให้เป็นมาตรฐาน LAPB (Link Access Procedure Balance) เพื่อให้สามารถรับส่งข้อมูลที่เป็นคำสั่ง (Command) และตอบสนอง (Response) ได้ทั้งสองฝั่งของ DTE และ DCE คือ ถ้า DTE เป็นฝ่ายส่งคำสั่ง จะถูกเรียกว่าเป็นสถานีหลัก (Primary) ฝั่ง DCE จะเป็นฝ่ายตอบสนองกลับ โดยทำหน้าที่เป็นสถานีรอง (Secondary) ในทางกลับกัน ฝั่ง DCE สามารถเป็นฝ่ายส่งคำสั่งหรือเป็นสถานีหลักได้ ฝั่งตรงข้ามจะเป็นฝ่ายตอบสนองกลับหรือเป็นสถานีรอง ดังแสดงในรูปที่ 3.7 ความสัมพันธ์ภายใต้ LAPB ระหว่างอุปกรณ์ DTE และ DCE

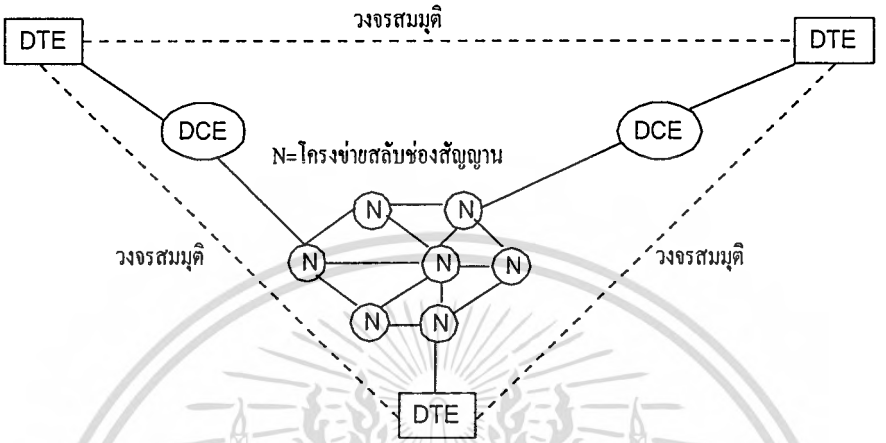
รูปที่ 3.7 ความสัมพันธ์ภายใต้ LAPB ระหว่างอุปกรณ์ DTE และ DCE



3.1.3.3 ระดับชั้นโครงข่ายสื่อสาร (Network layer) หรือระดับแพ็คเก็ต (Packet level)

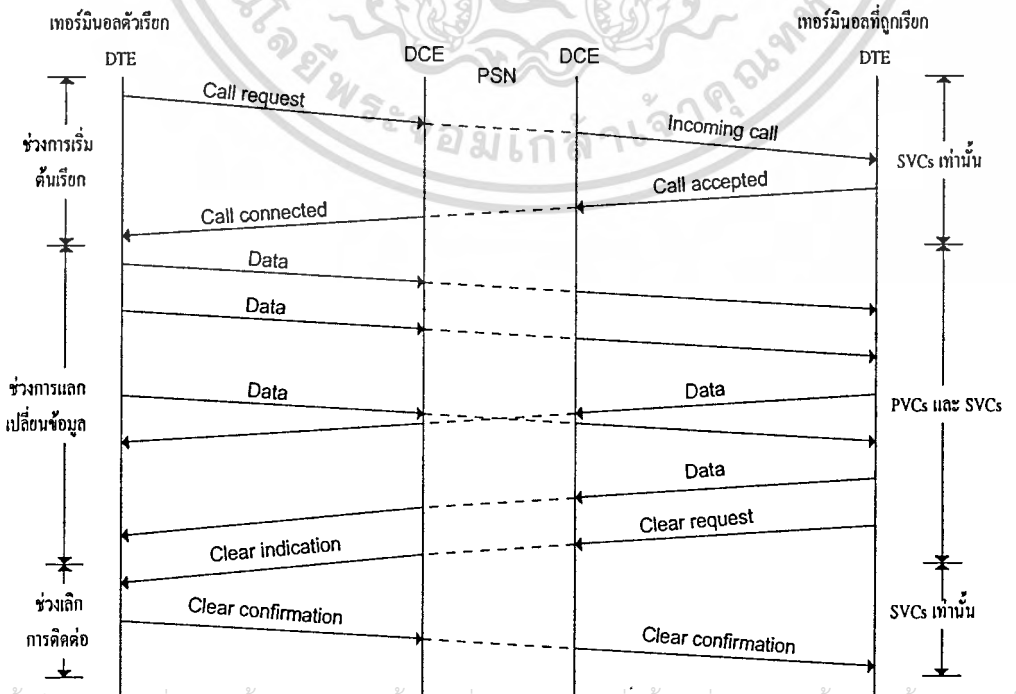
มีหน้าที่กำหนดรูปแบบและการควบคุมแพ็คเก็ตในขั้นตอนการแลกเปลี่ยนข้อมูลระหว่าง DTE และ DCE ก่อนการแลกเปลี่ยนข้อมูลจะมีการสร้างวงจรสมมุติ (Virtual circuit) ขึ้นมาก่อน ในระบบวงจรสมมุติมีอยู่ 2 ลักษณะ คือ วงจรสมมุติแบบถาวร (Permanent Virtual Circuit หรือ PVC) และวงจรสมมุติแบบสลับช่อง (Switched Virtual Circuit หรือ SVC) โดยที่วงจรสมมุติแบบถาวรจะมีการเชื่อมต่อแบบคงที่ระหว่าง DTE และ DCE ซึ่งไม่ต้องการสร้างแพ็คเก็ตการเรียกใช้ (Call establishment) และแพ็คเก็ตยกเลิกการติดต่อ (Call-clearing) ในขณะที่วงจรสมมติแบบสลับช่องมีการทำงานดังกล่าว และมีการสร้างวงจรเชื่อมโยงการทำงานให้อยู่ในชั่วขณะหนึ่งของเวลาเท่านั้น ในรูปที่ 3.8 วงจรสมมุติ (Virtual circuit) บนโครงข่ายสลับช่องสัญญาณ (PSN)

รูปที่ 3.8 วงจรสมมุติ (Virtual circuit) บนโครงข่ายสลับช่องสัญญาณ (PSN)



ขั้นตอนการแลกเปลี่ยนข้อมูลในระดับแพ็คเก็ตนี้ประกอบด้วย 3 ช่วง คือ ช่วงการเริ่มต้นเรียก (Call set-up) ช่วงการแลกเปลี่ยนข้อมูล (Data transfer phase) และช่วงเลิกการติดต่อ (Call clearing phase) เป็นไปดังในรูปที่ 3.9 ขั้นตอนการแลกเปลี่ยนข้อมูลในระดับแพ็คเก็ต

รูปที่ 3.9 ขั้นตอนการแลกเปลี่ยนข้อมูลในระดับแพ็คเก็ต

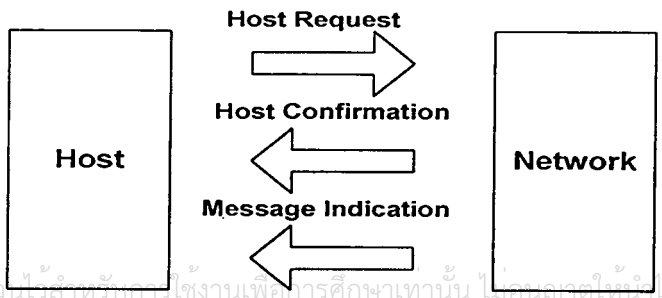


จากรูปที่ 3.9 ช่วงการเริ่มต้นเรียกนี้ จะมีอยู่บนวงจรสมมุติแบบสลับช่องเท่านั้น เริ่มต้นที่เทอร์มินอลตัวเรียก หรือ DTE ส่งสัญญาณเรียก (Call request) ในรูปของแพ็คเก็ต ส่วนหัว (Header) ของแพ็คเก็ตนี้ประกอบด้วยรหัสตำแหน่งที่อยู่ Address ของเทอร์มินอลที่ถูกเรียก พร้อมทั้งช่องทางการสื่อสารทางลอจิก (Logical Channel Identifier หรือ LCI) ฝ่ายเทอร์มินอลที่ถูกเรียกจะรับสัญญาณเรียกในรูปแบบของสัญญาณที่ได้รับการเรียก (Incoming call) เทอร์มินอลที่ถูกเรียกอาจยอมรับหรือปฏิเสธก็ได้ ถ้ายอมรับการเรียกจะตอบกลับไปด้วยสัญญาณว่าพร้อมจะรับ (Call accepted) ทำให้เทอร์มินอลตัวเรียกได้รับสัญญาณพร้อมที่จะติดต่อกันได้ (Call connected) และสามารถทำการแลกเปลี่ยนข้อมูล (Data) ซึ่งกันและกันได้ แต่ถ้าปฏิเสธจะส่งสัญญาณต้องการยกเลิกการติดต่อ (Clear request) ออกไป พร้อมทั้งยกเลิกการติดต่อไปเลย และเมื่อเวลาใด ๆ ก็ตามที่อุปกรณ์คอมพิวเตอร์ปลายทางทั้งสองต้องการยุติการติดต่อกัน มันสามารถทำได้โดยการส่งแพ็คเก็ตของสัญญาณยกเลิกการติดต่อ (Clear request) แพ็คเก็ตนี้จะไปปรากฏที่ปลายทางของคู่การติดต่อด้วยแพ็คเก็ตของการระบุการยกเลิก (Clear indication) ดังนั้นเทอร์มินอลคู่การติดต่อก็จะส่งแพ็คเก็ตการยืนยันการยกเลิก (Clear confirmation) ไปให้ ทำให้การติดต่อสิ้นสุดลง

3.2 คุณสมบัติของโปรโตคอล SCR

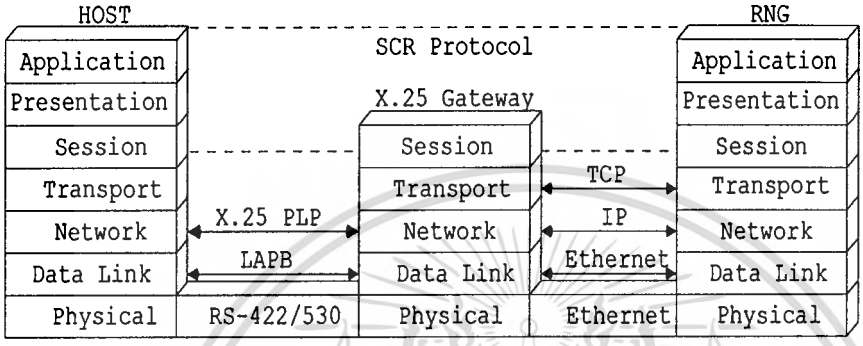
โปรโตคอล SCR (Standard Context Routing) ทำหน้าที่ควบคุมเซสชั่น (Session) ของการเชื่อมโยงกันระหว่างอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ (Mobile terminal) และคอมพิวเตอร์หลัก (Host computer) เฉพาะการต่อเชื่อมแบบเป็นกลุ่ม (Fleet connectivity) ซึ่งอยู่บนเลเยอร์ของโปรโตคอลมาตรฐาน X.25 หรือ TCP/IP โดยที่โปรโตคอล SCR นี้อยู่ระหว่างคอมพิวเตอร์หลัก (Host) และเกตเวย์โครงข่ายวิทยุ (RNG) มีรูปแบบดังในรูปที่ 3.10 และจะต้องเชื่อมโยงไว้ก่อนที่จะมีการรับส่งข้อมูลกับอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ ในรูปที่ 3.11 แสดงการเชื่อมต่อโปรโตคอล SCR ร่วมกับ X.25 และ TCP/IP ระหว่างคอมพิวเตอร์หลักและเกตเวย์โครงข่ายวิทยุ

รูปที่ 3.10 แสดงรูปแบบโปรโตคอล SCR

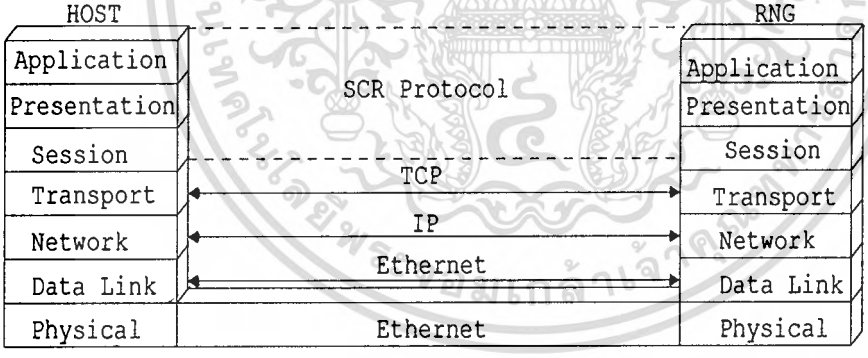


รูปที่ 3.11 แสดงการเชื่อมต่อโปรโตคอล SCR ร่วมกับ X.25 และ TCP/IP ระหว่างคอมพิวเตอร์หลักและเกตเวย์โครงข่ายวิทยุ

(ก) การเชื่อมต่อโปรโตคอล SCR ร่วมกับ X.25



(ข) การเชื่อมต่อโปรโตคอล SCR ร่วมกับ TCP/IP

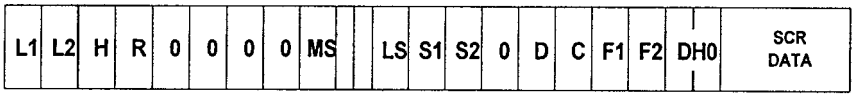


รูปแบบของโปรโตคอล SCR ดังแสดงในรูปที่ 3.10 มี 3 รูปแบบด้วยกัน คือ Host Request, Host Confirmation และ Message Indication

3.2.1 รูปแบบ Host Request (HR)

คือ คอมพิวเตอร์หลักร้องขอไปยังโครงข่ายเพื่อที่จะทำการส่งข้อมูลออกไปยังอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ต่าง ๆ เช่น การส่งอิเล็กทรอนิกส์เมล์ เป็นต้น โดยมีรูปแบบเฟรมที่ส่งออกไปดังแสดงในรูปที่ 3.12

รูปที่ 3.12 แสดงรูปแบบเฟรมของ Host Request



ในแต่ละฟิลด์มีรายละเอียดดังนี้คือ

- L1 และ L2 คือความยาวข้อมูล (Byte Length) ของเฟรม มีขนาด 2 ไบต์ เริ่มนับจากฟิลด์ H เป็นต้นไป

- H และ R คือชนิดของเฟรมข้อมูล (SCR message type) มีขนาด 2 ไบต์ อยู่ในรูปแบบของรหัส ASCII โดยย่อมา จาก Host Request

- MS ถึง LS คือค่า Link Logical Identification (LLI) มีขนาด 4 ไบต์ เป็นการระบุตำแหน่งที่อยู่ของอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ มีค่าตั้งแต่ 80000000 ถึง FFFFFFFF สามารถที่จะระบุตำแหน่งเฉพาะบุคคลและเป็นกลุ่มๆ ได้

- S1 และ S2 คือค่า Save byte มีขนาด 2 ไบต์ แสดงหมายเลขป้ายประกาศ (Tag) หรือ Sequence number ของเฟรมนั่นเอง มีค่าตั้งแต่ 0x0000 ถึง 0xFFFF เพื่อบอกให้อุปกรณ์โครงข่ายตอบสนองกลับมาด้วย Sequence number เดียวกัน

- D คือ Delivery option มีขนาด 1 ไบต์ มีค่า “0” และ “1” เมื่อเลือก “0” ในการส่งข้อมูลจะส่งครั้งเดียวออกไปเลย หรือเลือก “0” จะส่ง แล้วรอจนกระทั่งถึงผู้รับหรือมีค่า Timeout

- C คือ โหมดการตอบกลับ (Confirm mode) มีขนาด 1 ไบต์ มีค่าเท่ากับ “0”, “1” และ “2” เมื่อเลือก “0” จะไม่มีการตอบกลับ และเมื่อเลือก “1” และ “2” จะมีการตอบกลับ และดูว่ามาจากอุปกรณ์เทอร์มินอลหรือโครงข่ายตามลำดับ

- F1 และ F2 คือ Format byte มีขนาด 2 ไบต์ ถูกสงวนไว้ใช้กับ RD-LAP Format

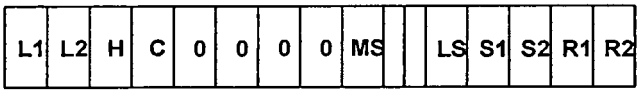
- DHO คือ Data Header Offset มีขนาด 2 ไบต์ เป็นตัวบอกความยาวของส่วนหัว (Header) ที่ประยุกต์ใช้ในส่วนของ SCR DATA มีความยาวเริ่มตั้งแต่ 0 ถึง 63

- SCR DATA คือข้อมูลที่ใช้รับส่ง มีขนาดไม่เกิน 2048 ไบต์ ในที่นี้เรากำหนด DHO 2 ไบต์แรกเป็นตัวกำหนด Session ในการรับส่งข้อมูลของแต่ละโปรแกรมระบบงานต่างๆ

3.2.2 รูปแบบ Host Confirmation (HC)

คือข้อมูลที่ตอบสนองจากโครงข่าย เมื่อคอมพิวเตอร์หลักทำการร้องขอมา เช่นแสดงถึงเครื่องผู้ใช้ไม่อยู่ในพื้นที่บริการ หรือไม่มีสัญญาณตอบรับจากเครื่องผู้ใช้ เป็นต้น แสดงเป็นรูปแบบเฟรมได้ดังรูปที่ 3.13

รูปที่ 3.13 แสดงรูปแบบเฟรมของ Host Confirmation



ในแต่ละฟิลด์มีรายละเอียดดังนี้คือ

- L1 และ L2 คือความยาวข้อมูล (Byte Length)มีขนาด 2 ไบต์

- H และ C คือชนิดของเฟรมข้อมูล (SCR message type) มีขนาด 2 ไบต์ ย่อมาจาก Host Confirmation

- MS ถึง LS คือค่า Link Logical Idenfication (LLI) มีขนาด 4 ไบต์ มีค่าตั้งแต่ 80000000 ถึง FFFFFFFF เป็นค่าที่ได้มาจากเฟรม Host Request เพื่อตอบสนองกลับไปยังคอมพิวเตอร์หลัก

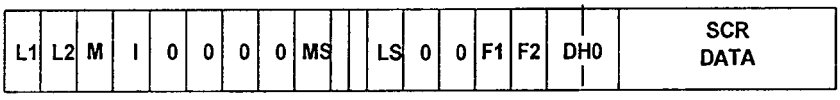
- S1 ถึง S2 คือค่า Save byte มีขนาด 2 ไบต์ มีค่าตั้งแต่ 0x0000 ถึง 0xFFFF ที่ได้มาจากเฟรม Host Request เพื่อตอบสนองกลับไปยังคอมพิวเตอร์หลักด้วยค่า Sequence number ที่เหมือนกัน

- R1 และ R2 คือค่าแสดงรหัสตอบสนอง (Response Code) มีขนาด 2 ไบต์ คือค่า 08-0F แสดง ACK และค่า 40-47 แสดง NAK เป็นตัวบอกถึงสถานะของอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ว่าปกติ (ACK) หรือไม่ปกติ (NAK) อย่างไร

3.2.3 รูปแบบ Message Indication (MI)

คือรูปแบบเฟรมที่แสดงถึงข้อมูลที่รับมาจากอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ และข้อมูลที่ได้จะถูกคัดเลือกเอาเฉพาะส่วนที่ต้องการส่งต่อไปยังคอมพิวเตอร์หลัก แสดงเป็นรูปแบบเฟรมดังในรูปที่ 3.14

รูปที่ 3.14 แสดงรูปแบบเฟรมของ Message Indication



ในแต่ละฟิลด์มีรายละเอียดดังนี้คือ

- L1 และ L2 คือความยาวของข้อมูล (Byte Length) มีขนาด 2 ไบต์

- M และ I คือชนิดของเฟรมข้อมูล (SCR message type identifier) ย่อมาจาก Message Indication

- MS ถึง LS คือค่า Link Logical Idenfication (LLI) มีขนาด 4 ไบต์ มีค่าตั้งแต่ 80000000 ถึง FFFFFFFF เป็นค่าที่ได้มาจากเฟรม Host Request เพื่อตอบสนองกลับไปยังคอมพิวเตอร์หลัก

- F1 และ F2 คือรูปแบบข้อมูล (Format byte) มีขนาด 2 ไบต์

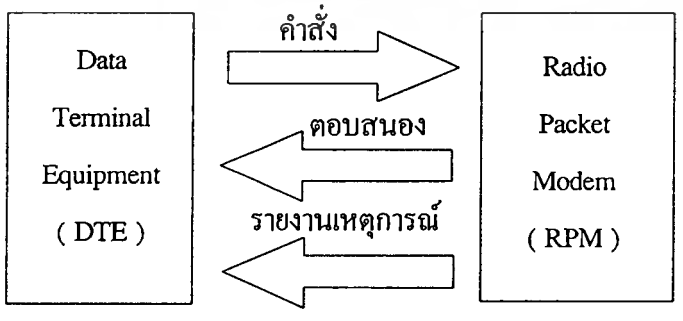
- DHO คือ Data Header Offset มีขนาด 2 ไบต์ เป็นตัวบอกความยาวของส่วนหัว (Header) ที่ประยุกต์ใช้ในส่วนของ SCR DATA มีความยาวเริ่มตั้งแต่ 0 ถึง 63

- SCR DATA คือข้อมูลที่ใช้รับส่ง มีขนาดไม่เกิน 2048 ไบต์ ในที่นี้เรากำหนด DHO 2 ไบต์แรกเป็นตัวกำหนด Session ในการรับส่งข้อมูล ของแต่ละโปรแกรมระบบงานต่าง ๆ

3.3 คุณสมบัติของโปรโตคอล NCL

โปรโตคอล NCL (Native Control Language) เป็นภาษาควบคุมการแลกเปลี่ยนข้อมูลแบบอะซิงโครนัส ของการติดต่อสื่อสารระหว่างอุปกรณ์คอมพิวเตอร์ปลายทาง(DTE) และโมเด็มไร้สายชนิดคลื่นวิทยุแบบแพ็คเก็ต (Radio Packet Modem หรือ RPM) ก่อนที่จะรับหรือส่งข้อมูลด้วยคลื่นวิทยุ การรับหรือส่งข้อมูลจะเป็นระบบสลับกัน (Transaction-oriented) และมีการสื่อสารแบบไม่ต้องเชื่อมโยงกันตลอดเวลา (Connectionless) ในรูปที่ 3.15 แสดงรูปแบบของโปรโตคอล NCL และในรูปที่ 3.16 แสดงตัวอย่างไดอะแกรมการรับส่งข้อมูลด้วยโปรโตคอล NCL

รูปที่ 3.15 แสดงรูปแบบของโปรโตคอล NCL



รูปที่ 3.16 แสดงตัวอย่างไดอะแกรมการรับส่งข้อมูลด้วยโปรโตคอล NCL

* สถานะการตอบสนองไม่จำเป็นต้องเรียงลำดับ (Sequence)

การรับส่งข้อมูลด้วยโปรโตคอล NCL ประกอบด้วย 3 ระดับชั้น (Layer) คือ

1). ระดับชั้นทางกายภาพ (Physical Layer)

2). ระดับชั้นเชื่อมโยงข้อมูล (Data Link Layer)

3). ระดับชั้นแสดงผล (Presentation Layer)

3.3.1 การเชื่อมต่อทางกายภาพ (Physical Level Interface)

เป็นการกำหนดมาตรฐานของส่วนโปรโตคอล NCL ในระดับชั้นทางกายภาพ บริเวณจุดเชื่อมต่ออนุกรมระหว่างอุปกรณ์คอมพิวเตอร์ปลายทางและโมเด็มไร้สาย ซึ่งโมเด็มไร้สายจะไม่มีพอร์ตอนุกรม แต่จะใช้วิธีการจำลอง (Emulate) พอร์ตอนุกรมขึ้นมาตามมาตรฐานที่กำหนดในระดับชั้นทางกายภาพ

การเชื่อมโยงแบบอนุกรมมีคุณสมบัติดังนี้

1). รูปแบบข้อมูลเป็นแบบอะซิงโครนัส 8 bits, 1 Start bit, 1 Stop bit, no Parity

2). อัตราความเร็วการส่งข้อมูลเริ่มต้นที่ 9600 bps ถึง 19.2 kbps

3). ในขณะที่สัญญาณ Data Set Ready (DSR) ถูกใช้งาน เราจะให้สัญญาณ Data Carrier Detect (DCD) เป็นไปตามสัญญาณ DSR

ลักษณะของสัญญาณต่าง ๆ ที่พอร์ตอนุกรมของโมเด็มไร้สายสามารถอธิบายได้ดังนี้

3.3.1.1 Data Carrier Detect (DCD)

สัญญาณ DCD แสดงสถานะของโครงข่ายคลื่นวิทยุที่อยู่ในพื้นที่บริการ และนอกพื้นที่บริการ เป็นสัญญาณเอ้าท์พุตของ RPM

3.3.1.2 Transmit Data (TXD)

สัญญาณ TXD แสดงการส่งข้อมูลจากอุปกรณ์ DTE ไปยังอุปกรณ์ RPM เป็นสัญญาณอินพุตที่เข้าอุปกรณ์ RPM

3.3.1.3 Receive Data (RXD)

สัญญาณ RXD แสดงสถานะการรับข้อมูลจากอุปกรณ์ RPM เข้ามายังอุปกรณ์ DTE

3.3.14 Data Set Ready (DSR)

สัญญาณ DSR เป็นสถานะของอุปกรณ์ RPM ใช้สำหรับตรวจสอบกำลังไฟแบตเตอรี่ด้วย ในกรณีที่พอร์ตอนุกรมของ DTE เป็นแบบ 9 ขา (DB9) เราจะเชื่อมสัญญาณ DCD เข้ากับ DSR

3.3.1.5 Ground (GND)

สัญญาณ GND เป็นจุดต่อลงกราวด์ (Ground)

3.3.1.6 Data Terminal Ready (DTR)

สัญญาณ DTR เป็นสถานะของอุปกรณ์ DTE ส่งไปยังอุปกรณ์ RPM เมื่ออุปกรณ์ DTE พร้อมที่จะรับส่งข้อมูลกับอุปกรณ์ RPM สัญญาณ DTR ยังคงสถานะตลอดไป จนกระทั่งอุปกรณ์ RPM ส่งสัญญาณปิด (OFF) มาที่อุปกรณ์ DTE สถานะของ DTR จะถูกปิดไปด้วย

3.3.1.7 Request To Send (RTS)

เป็นสัญญาณอินพุทเข้าอุปกรณ์ RPM สัญญาณ RTS ใช้งานร่วมกับสัญญาณ CTS ในการควบคุมการรับส่งข้อมูล (Flow control) ถ้า RTS ถูกเลือกใช้ อุปกรณ์ DTE จะพร้อมรับข้อมูลจาก RPM RTS จะถูกยกเลิกเมื่ออุปกรณ์ DTE ไม่พร้อมรับข้อมูล และถ้า RTS/CTS ไม่ถูกใช้งาน (Disable) สถานะของ RTS นี้จะไม่ทำงาน ทำให้ข้อมูลสามารถส่งผ่านได้ตลอดเวลา

3.3.1.8 Clear To Send (CTS)

เป็นสัญญาณเอ้าท์พุตจากอุปกรณ์ RPM สัญญาณ CTS ใช้ร่วมกับสัญญาณ RTS สำหรับควบคุมการรับส่งข้อมูล (Flow Control) เมื่อ CTS ถูกเลือกใช้ อุปกรณ์ RPM จะพร้อมรับข้อมูลจากอุปกรณ์ DTE ถ้า RTS/CTS ถูกยกเลิกการใช้งาน สัญญาณ CTS นี้จะพร้อมรับข้อมูลตลอดเวลา

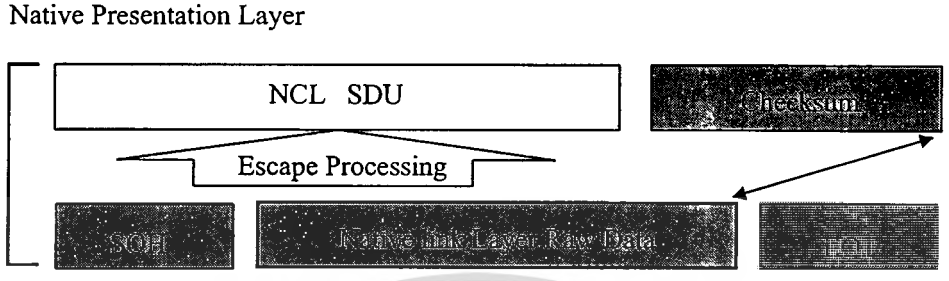
3.3.1.9 Ring Indicator (RI)

เป็นสัญญาณเอ้าท์พุตจากอุปกรณ์ RPM

3.3.2 การเชื่อมต่อในระดับชั้นเชื่อมโยงข้อมูล (Data Link Layer Protocol)

ในระดับชั้นนี้โปรโตคอล NCL จะมีความน่าเชื่อถือมากขึ้นในการแลกเปลี่ยนเฟรมของโปรโตคอล NCL ในรูปที่ 3.17 แสดงรูปแบบเฟรมโปรโตคอล NCL ในระดับชั้น Data-Link

รูปที่ 3.17 แสดงรูปแบบเฟรมโปรโตคอล NCL ในระดับชั้น Data-Link



3.3.2.1 ตัวอักขระ SOH และ EOT

ตัวอักขระ SOH (Start Of Header) เป็นตัวกำหนดจุดเริ่มต้นและตัวอักขระ EOT (End Of Text) เป็นตัวกำหนดจุดสิ้นสุดเฟรมข้อมูลของโปรโตคอล NCL ทั้ง SOH และ EOT เป็นรูปแบบมาตรฐานของรหัสแอสกี มีค่าตามเลขฐานสิบหกคือ 0x01H และ 0x04H ตามลำดับ

3.3.2.2 ฟิลด์ NCL SDU

เฟรมของคำสั่ง สถานะการตอบสนอง และรายงานเหตุการณ์ จะถูกบรรจุอยู่ภายในฟิลด์ของ NCL SDU จะกล่าวรายละเอียดอีกครั้งหนึ่งในการเชื่อมต่อระดับชั้นการแสดงผล

3.3.2.3 ฟิลด์ Checksum

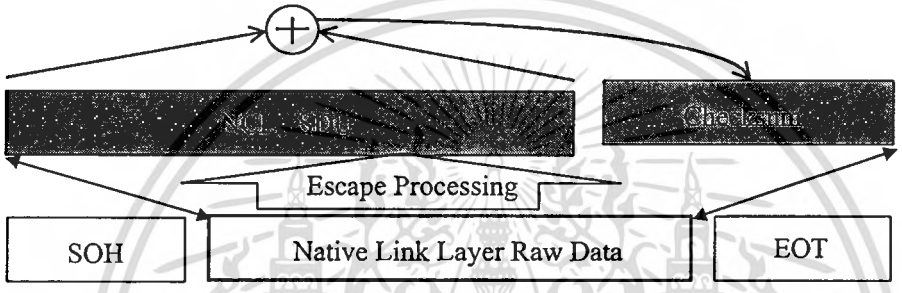
ฟิลด์ Checksum มีขนาด 16 บิต ทำหน้าที่คำนวณข้อมูลที่รับเข้ามา แล้วเปรียบเทียบกับค่าที่อยู่ในฟิลด์ Checksum โดยที่ผู้ส่ง (Sender) ทำการคำนวณค่า Checksum โดยการบวกค่าทุกๆ ไบต์ข้อมูลตั้งแต่จุดเริ่มต้นเฟรม (SOH) จนถึงจุดสิ้นสุดเฟรม (EOT) ยกเว้นฟิลด์ Checksum ตัวมันเอง เมื่อผู้รับ (Receiver) คำนวณข้อมูลที่รับเข้ามาแล้วเท่ากับค่าในฟิลด์ Checksum ก็แสดงว่าเฟรมข้อมูลนั้นถูกต้อง

สำหรับเฟรมคำสั่ง SDU ที่ส่งจากอุปกรณ์คอมพิวเตอร์ปลายทางไปยังโมเด็มไร้สาย สามารถกำหนดรูปแบบการส่งข้อมูลเป็นแบบ Confirmed หรือ Unconfirmed ก็ได้ อุปกรณ์โมเด็มไร้สายจะมีการตรวจสอบที่ฟิลด์ Checksum นี้ คือ ถ้าเราให้รูปแบบการส่งข้อมูลเป็นแบบ Confirmed ฟิลด์ Checksum ต้องมีค่าที่ถูกต้อง ถ้าโมเด็มไร้สายคำนวณค่า Checksum แล้วไม่เท่ากับค่าในฟิลด์ Checksum ภายในเฟรมคำสั่ง SDU อุปกรณ์โมเด็มไร้สายจะปฏิเสธเฟรมคำสั่งนั้นทันที และส่งสถานะตอบสนองแจ้งกลับไปยังอุปกรณ์คอมพิวเตอร์ปลายทางด้วยรูปแบบ XFAIL: PACKET_ERROR และพยายามส่งเฟรมคำสั่งนั้นใหม่อีกครั้ง จนกระทั่งทำคำสั่งนั้นสำเร็จ หรือ

Time-out แต่ถ้าอยู่ในรูปแบบ Unconfirmed อุปกรณ์โมเด็มไร้สาย จะยกเลิกการตรวจสอบฟิลด์ Checksum ทุก ๆ เฟรมที่รับเข้ามา และถือว่าเป็นเฟรมถูกต้องทั้งหมด

ในกรณีที่ส่งเฟรมจากโมเด็มไร้สายไปยังอุปกรณ์คอมพิวเตอร์ปลายทาง อุปกรณ์โมเด็มไร้สายจะรวมค่า Checksum ที่คำนวณขึ้นมาเสมอ เพียงแต่กำหนดว่าจะตรวจสอบฟิลด์ Checksum ในเฟรมข้อมูลหรือไม่ตรวจสอบเท่านั้น สามารถแสดงไดอะแกรมการ Checksum ได้ในรูปที่ 3.18

รูปที่ 3.18 แสดงไดอะแกรมการ Checksum



3.3.2.4 ตัวอักขระพิเศษ Escaping (Escaping Special Character)

อักขระที่แสดงในตารางที่ 3.1 ถูกสงวนไว้ใช้สำหรับการใช้งานพิเศษ อักขระพิเศษเหล่านี้ต้องไม่ปรากฏอยู่ภายใน Stream data ยกเว้นในฟิลด์ Checksum เท่านั้น ปกติอักขระพิเศษเหล่านี้จะพบในการรับส่งข้อมูลแบบไบนารี (Binary data) เมื่อพบจะต้องบวกค่าด้วย 0x20H และปะอักขระพิเศษ DLE(Data-Link Escape) คือ 0x10H ไว้ส่วนหน้าของอักขระพิเศษที่พบ เพื่อบอกให้ชั้นการเชื่อมโยง (Data link) แยกแยะข้อมูลเวลาแปรคำสั่งด้วยโปรโตคอล NCL ในตารางที่ 3.1 แสดงอักขระต่าง ๆ ในการใช้งานพิเศษ

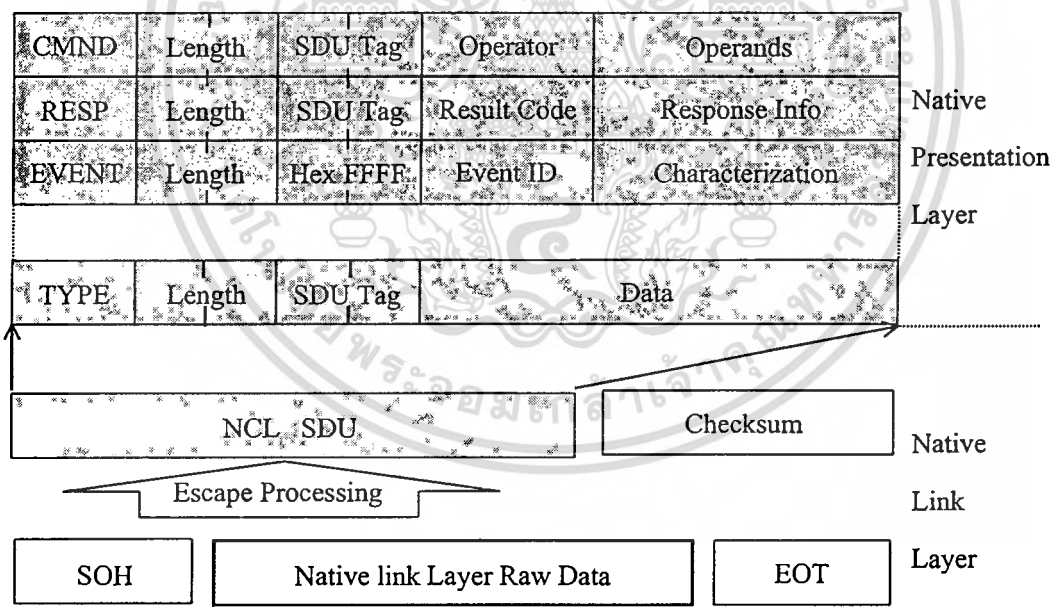
ตารางที่ 3.1 แสดงอักขระต่าง ๆ ในการใช้งานพิเศษ

รหัสแอสกี	เลขฐานสิบหก	หน้าที่การใช้งาน	อักขระ Escaped
SOH	0x01	เริ่มต้นเฟรม (Start of Frame)	แอสกี '!'
EOT	0x04	สิ้นสุดเฟรม (End of Frame)	แอสกี '$'
DLE	0x10	Data-Link Escape	แอสกี '0'
XON	0x11	ซอฟต์แวร์ควบคุมการส่งข้อมูล (Restart flow)	แอสกี '1'
XOFF	0x13	ซอฟต์แวร์ควบคุมการส่งข้อมูล (Stop flow)	แอสกี '3'

3.3.3 การเชื่อมต่อในระดับชั้นการแสดงผล (Presentation Layer Protocol)

โปรโตคอล NCL ในระดับชั้นการแสดงผล ประกอบไปด้วยกลุ่มหน่วยข้อมูลที่ติดต่อสื่อสารกันในย่านความถี่วิทยุระหว่างอุปกรณ์คอมพิวเตอร์ปลายทางและโมเด็มไร้สาย หน่วยข้อมูลที่ส่งระหว่างอุปกรณ์คอมพิวเตอร์ปลายทาง และโมเด็มไร้สาย นั้นเรียกว่า Service Data Unit (SDU) ทำหน้าที่เกี่ยวกับการรายงานเหตุการณ์ต่างๆ จากโมเด็มไร้สาย รวมถึงการตั้งค่าพารามิเตอร์ และควบคุมการทำงานไปยังระบบโครงข่าย ในแต่ละเฟรม SDU จะถูกตัดทอนออกเป็นหน่วยข้อมูลเล็กๆ เรียกว่า Protocol Data Unit (PDU) หลายๆ แพ็คเก็ตส่งออกไป ทางผู้รับจะทำการจัดเรียงแพ็คเก็ตให้เป็นเฟรม SDU อีกครั้งหนึ่ง ในการส่งแพ็คเก็ต PDU แต่ละครั้งจะมีการใส่แอดเดรสต้นทางของโมเด็มไร้สายลงไปอย่างอัตโนมัติ โดยที่แอดเดรสนี้จะมีเพียงแอดเดรสเดียวไม่มีการซ้ำกัน ในรูปที่ 3.19 แสดงรูปแบบเฟรมโปรโตคอล NCL ในระดับชั้นการแสดงผล

รูปที่ 3.19 แสดงรูปแบบเฟรมโปรโตคอล NCL ในระดับชั้นการแสดงผล



รายละเอียดของฟิลด์ NCL SDU มีดังนี้ คือ

1). ฟิลด์ชนิดเฟรม (Frame Type Field) มีขนาด 8 บิต จะแสดงว่าเป็นชนิดคำสั่ง(CMND) ตอบสนอง (RESP) หรือรายงานเหตุการณ์ (EVENT)

2). ฟิลด์ความยาว (Length Field) มีขนาด 16 บิต แสดงจำนวนของข้อมูลภายในฟิลด์ข้อมูล (Data Field) ถ้ามีตัวอักขระ Escaping จะไม่ถูกนับรวมให้อยู่ในฟิลด์แสดงความยาวข้อมูล โดยที่

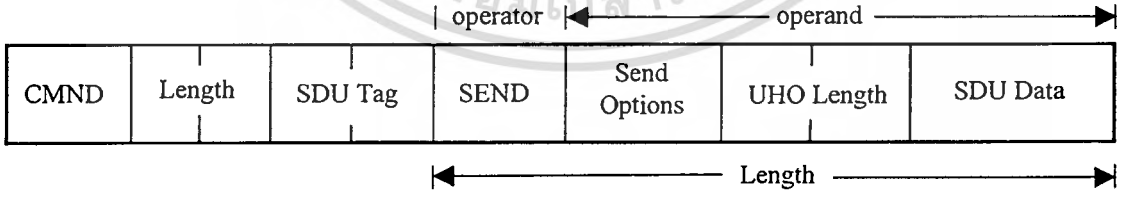
ตัวอักขระ Escaping จะถูกตัดออกไปเวลาแปลคำสั่ง NCL และถ้ามี Flow Control ภายในเฟรมด้วย จะไม่มีการนับจำนวนความยาวในฟิลด์เช่นเดียวกัน

3). ฟิลด์ป้ายประกาศข้อมูล (SDU Tag Field) มีขนาด 16 บิต มีค่าตั้งแต่ 0x000 ถึง 0xFFFF แสดงหมายเลขลำดับ (Sequence number) ในเฟรม SDU ของการรับส่งข้อมูลระหว่างอุปกรณ์ DTE และ RPM เช่น เมื่อมีการส่งเฟรมคำสั่งจาก DTE ไปที่ RPM จะมีการปะหมายเลขลำดับไปในเฟรมด้วย อุปกรณ์ RPM ก็จะมีการตอบสนองด้วยหมายเลขลำดับเดียวกันกับเฟรมคำสั่งที่ส่งมา ซึ่งเป็นไปได้อาจจะมีเฟรมรายงานเหตุการณ์ส่งไปก่อนเฟรมตอบสนองได้ สำหรับเฟรมรายงานเหตุการณ์จะมีหมายเลขลำดับเป็น Hex FFFF เพื่อรายงานเหตุการณ์ต่างๆ เช่น สภาพของแบตเตอรี่ และความแรงของสัญญาณ เป็นต้น ดังนั้นเมื่อมีข้อมูลมาที่อุปกรณ์ DTE ก็สามารถที่จะแยกแยะข้อมูลได้ว่าเป็นของเฟรมตอบสนองหรือว่าเฟรมรายงานเหตุการณ์ โดยมี SDU Tag เป็นตัวบอก

4). ฟิลด์ข้อมูล (Data Field) มีขนาดตามที่ผู้ใช้กำหนด แต่ต้องไม่มากกว่า 2048 ไบต์ ซึ่งรูปแบบของฟิลด์ข้อมูลนี้จะถูกกำหนดโดยฟิลด์ชนิดเฟรม (Frame Type Field) และในแต่ละชนิดเฟรมจะแบ่งออกเป็นฟิลด์ข้อมูลย่อย ๆ แตกต่างกัน

ในทุกๆ เฟรมข้อมูลของโปรโตคอล NCL SDU จะมีฟิลด์ข้อมูลร่วมกัน คือ ฟิลด์ชนิดเฟรม ความยาว และป้ายประกาศข้อมูล จากรูปที่ 3.19 เราสามารถเขียนเฟรมคำสั่ง (CMND) ในรูปแบบของเฟรมคำสั่งการส่งข้อมูล (Send Message) ดังในรูปที่ 3.20 และรูปแบบเฟรมการอ่านข้อมูล (Read Message) ดังแสดงในรูปที่ 3.23

รูปที่ 3.20 แสดงรูปแบบเฟรมคำสั่งการส่งข้อมูล (Send Message)

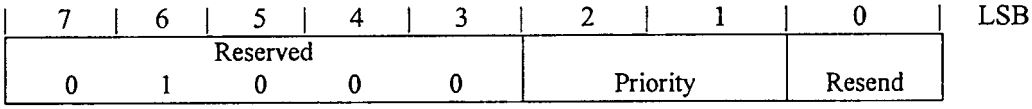


รายละเอียดของเฟรมคำสั่งการส่งข้อมูลมีดังนี้

1) ฟิลด์ SEND มีขนาด 1 ไบต์ มีค่าเป็นแอสกี '1'

2) ตัวเลือกการส่ง (Send options) มีขนาด 1 ไบต์ และมีการทำงานดังในรูปที่ 3.21

รูปที่ 3.21 แสดงการใช้บิตฟิลด์ของตัวเลือกการส่ง



ตารางที่ 3.2 แสดงการเลือกค่าระดับความสำคัญ (Priority)

ตัวเลือกบิต	ระดับความสำคัญ
00	ธรรมดา
01	ต่ำมาก
10	ต่ำ
11	สูง

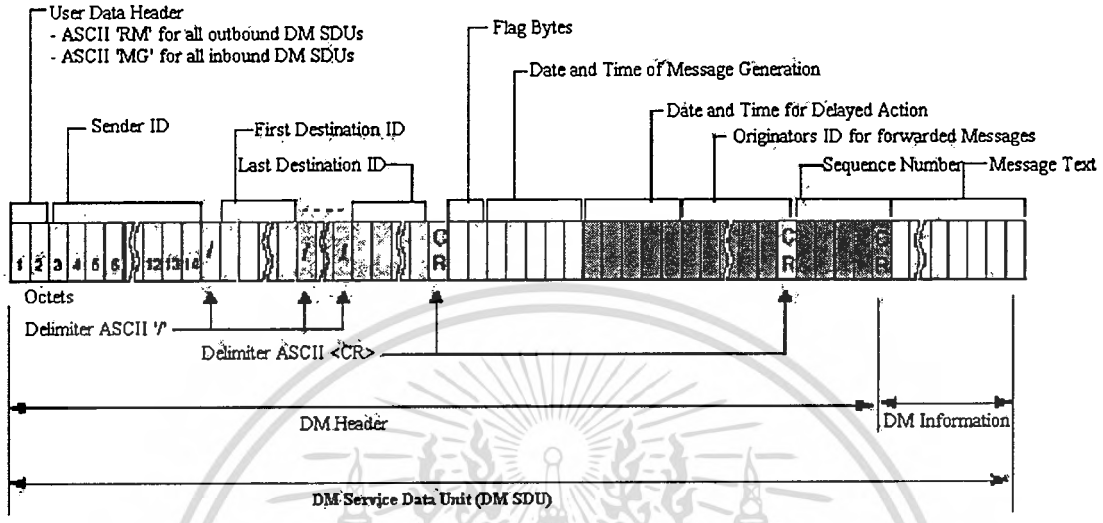
ตารางที่ 3.3 แสดงการเลือกค่าบิตของ Resend

ตัวเลือกบิต	หน้าที่ใช้งาน
0	ไม่มีการส่งข้อมูลซ้ำ
1	ให้ทำการส่งข้อมูลซ้ำ

3) ค่าออฟเซ็ตส่วนหัวของผู้ใช้ (User Header Offset หรือ UHO) มีขนาด 2 ไบต์ เป็นตัวกำหนดค่าความยาวของส่วนหัว (Header) เฟรมข้อมูลการส่ง มีค่าความยาวตั้งแต่ 30 ถึง 199

4) ข้อมูล SDU (SDU data) มีขนาดสูงสุด 2048 ไบต์ เป็นส่วนเนื้อข้อมูล ซึ่งส่วนหัวของข้อมูลจะระบุข่าวสารที่สำคัญในการส่งข้อมูลโดยมีความยาวตามที่ระบุไว้ในฟิลด์ UHO และในส่วนหัวข้อมูลนี้ ถ้าเป็นการรับส่งข้อมูลระหว่างอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ (Wireless Terminal) ด้วยกัน จะใช้รูปแบบข้อมูลที่เรียกว่า DataTAC Messaging (DM) แสดงให้เห็นรูปแบบข้อมูลของ DataTAC Messaging ในรูปที่ 3.22 แต่ถ้าอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ติดต่อกับคอมพิวเตอร์หลักที่เป็นโปรแกรมระบบงานอื่นๆ เช่น อินเตอร์เน็ต เพจเจอร์ และแฟกซ์ เป็นต้น ผู้ใช้สามารถกำหนดรูปแบบข้อมูลส่วนหัวได้เองตามต้องการ

รูปที่ 3.22 แสดงรูปแบบข้อมูลในส่วนของ DataTAC Messaging (DM)



รายละเอียดของ DataTAC Messaging (DM) มีดังนี้

1) ฟิลด์ส่วนหัวข้อมูลผู้ใช้ (User Data Header) มีขนาด 2 ไบต์ เป็นตัวกำหนดเส้นทางการส่งผ่านข้อมูล มีอยู่ 2 ชนิด คือ ใช้แอสกี "MG" เมื่อต้องการส่งข้อมูลจากอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ไปยังเกตเวย์โครงข่ายวิทยุ และใช้แอสกี "RM" สำหรับส่งข้อมูลจากเกตเวย์โครงข่ายวิทยุมายังอุปกรณ์สื่อสารข้อมูลเคลื่อนที่

2) ฟิลด์แสดงเลขหมายประจำเครื่องผู้ส่ง (Sender ID) มีขนาด 14 ไบต์ ซึ่ง 6 ไบต์แรกแสดงถึงหมายเลขพื้นที่บริการ (Home Area) ของเครื่องผู้ส่ง และ 8 ไบต์ที่เหลือเป็นเลขหมายของโมเด็มไร้สายหรือเราเรียกว่า Link Logical Identification (LLI) มีค่าตั้งแต่ 80000000 ถึง FFFFFFFF

3) ฟิลด์แสดงเลขหมายประจำเครื่องปลายทาง (Destination ID) มีลักษณะเช่นเดียวกันกับเลขหมายประจำเครื่องผู้ส่ง ฟิลด์นี้จะมีแอสกี '/' คั่นระหว่างหมายเลขประจำเครื่องผู้ส่ง และยอมให้ใส่หมายเลขประจำเครื่องปลายทางได้สูงสุด 10 เลขหมาย โดยแต่ละเลขหมายคั่นด้วยแอสกี '/'

4) ฟิลด์แฟล็ก (Flag Field) มีขนาด 2 ไบต์ ในแต่ละบิตมีรายละเอียดการใช้งานดังตารางที่ 3.4 และ 3.5 ซึ่งถ้าเลือกใช้จะตั้งค่าบิตเป็น "1"

ตารางที่ 3.4 แสดงรายละเอียดแฟล็กไบต์ที่ 1

บิตที่	รายละเอียด
0	สงวนไว้
1	บ่งบอกว่าข้อมูลนี้บรรจุหมายเลขลำดับ (Sequence Number) ด้วย
2	สงวนไว้
3 - 5	สงวนไว้
6	ร้องขอการส่งซ้ำข้อมูล
7	สงวนไว้

ตารางที่ 3.5 แสดงรายละเอียดแฟล็กไบต์ที่ 2

บิตที่	รายละเอียด
0	บ่งบอกให้มีการกำหนดเวลาล่าช้า (Delay) ของข้อมูล
1	ลำดับความสำคัญ (Priority)
2	บ่งบอกว่าต้องการลงทะเบียน (Registered) ข้อมูลสำหรับข้อมูลนี้
3	บ่งบอกว่าเป็นข้อมูลลงทะเบียน (Registered Receipt)
4	สงวนไว้
5	ข้อมูลนี้เป็นแบบส่งต่อ (Forward Message)
6	การร้องขอการตอบกลับของข้อมูลนี้
7	บ่งบอกว่าข้อมูลนี้เป็นข้อมูลตอบกลับ (Reply)

5) ฟิลด์วันที่ (Date Field) มีขนาด 5 ไบต์ แสดงถึงวันที่และเวลาปัจจุบัน โดยเรียงลำดับจากเดือน วัน ปี ชั่วโมง นาที ยกตัวอย่างเช่น March 31, 1998 1:05 pm เขียนเป็น 0x0331981305

6) ฟิลด์วันที่กำหนดเวลาล่าช้าของข้อมูล (Data Field for Delayed Action) มีลักษณะเช่นเดียวกันกับฟิลด์วันที่ ฟิลด์นี้จะใช้งานได้ต้องตั้งค่าบิตที่ 0 เป็น "1" ที่ฟิลด์แฟล็กไบต์ที่ 2 ก่อน

7) ฟิลด์เลขหมายประจำเครื่องผู้ใช้อื่นที่ต้องการส่งต่อข้อมูล (Forwarded wireless terminal ID) มีขนาด 14 ไบต์ มีลักษณะเดียวกันกับฟิลด์แสดงเลขหมายประจำเครื่องผู้ส่ง ฟิลด์นี้จะใช้งานได้ต้องตั้งค่าบิตที่ 5 เป็น "1" ที่ฟิลด์แฟล็กไบต์ที่ 2 ก่อน

8) ฟิลด์แสดงหมายเลขลำดับข้อมูล (Sequence Number) มีขนาด 4 ไบต์ มีค่าตั้งแต่ 0x0000 ถึง 0xFFFF ฟิลด์นี้จะใช้งานได้เมื่อตั้งค่าบิตที่ 1 เป็น "1" ที่ฟิลด์แฟล็กไบต์ที่ 1 ก่อน

9) ฟิลด์ส่วนข้อมูล (Message Text) มีความยาวเมื่อรวมกับส่วนหัว (Header) ด้วย เท่ากับ 2048 ไบต์ สามารถเป็นได้ทั้งข้อมูลแบบไบนารี และแบบแอสกี

รูปที่ 3.23 แสดงรูปแบบเฟรมคำสั่งของการอ่านข้อมูล

			Operator	Operands
CMND	Length	SDU Tag	READ_MSG	Request Options

รายละเอียดของเฟรมคำสั่งการอ่านข้อมูล มีดังนี้

1) ฟิลด์ READ_MSG มีขนาด 1 ไบต์ มีค่าเป็นแอสกี '2'

2) ตัวเลือกการร้องขอ (Request Options) มีขนาด 1 ไบต์ เป็นตัวกำหนดว่าจะให้มีการร้องขอการอ่านข้อมูลก่อนหน้านั้นซ้ำใหม่หรือไม่ เมื่อเฟรมข้อมูลนั้นมีข้อมูลผิดพลาด (Error data) มีรูปแบบบิตดังในรูปที่ 3.24

รูปที่ 3.24 แสดงรูปแบบบิตของเฟรมคำสั่งของการอ่านข้อมูล

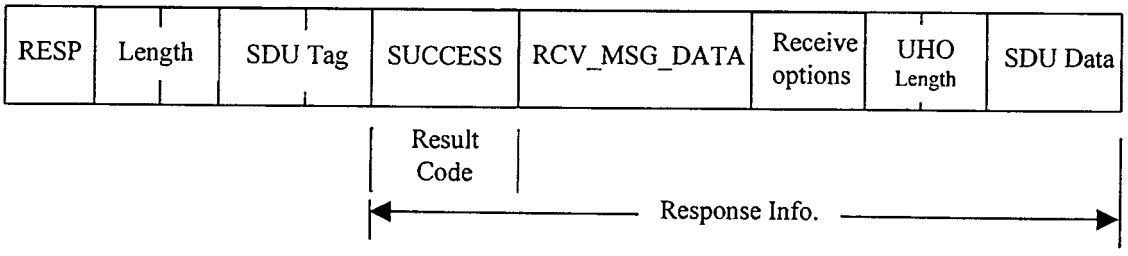
7	6	5	4	3	2	1	0	LSB
Reserved							Reread	
0	1	0	0	0	0	0		

ตารางที่ 3.6 แสดงตัวเลือกบิต Reread

ตัวเลือกบิต	หน้าที่ใช้งาน
0	อ่านข้อมูล SDU ใน queue ถัดไป
1	ร้องขอการอ่านข้อมูล SDU ก่อนหน้าซ้ำ

เมื่อใช้เฟรมคำสั่งของการอ่านข้อมูลแล้ว ข้อมูลที่สมบูรณ์ถูกอ่านอยู่ในรูปแบบของเฟรมตอบสนอง (RESP) ดังแสดงในรูปที่ 3.25

รูปที่ 3.25 แสดงรูปแบบเฟรมตอบสนองข้อมูลที่สมบูรณ์ (Success)



รายละเอียดของเฟรมตอบสนองการอ่านข้อมูลมีดังนี้

1) ฟิลด์ SUCCESS มีขนาด 1 ไบต์ มีค่าเป็นแอสกี '1' แสดงถึงเฟรมข้อมูลที่สมบูรณ์

2) ฟิลด์ RCV_MSG_DATA มีขนาด 1 ไบต์ มีค่าเป็นแอสกี 'A'

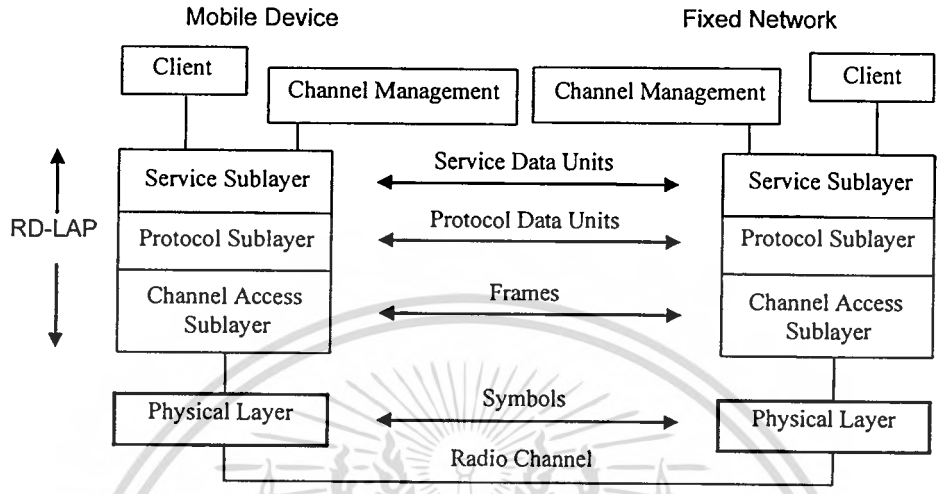
3) ฟิลด์ Receive options, UHO Length และ SDU Data มีลักษณะเช่นเดียวกันกับฟิลด์ในรูปแบบเฟรมคำสั่งของการส่งข้อมูล

สำหรับรูปแบบเฟรมข้อมูล NCL อื่นๆ ที่ไม่ได้กล่าวถึงในที่นี้ สามารถหารายละเอียดเพิ่มเติมได้ใน [4] และรายละเอียดของฟิลด์ข้อมูลย่อยต่าง ๆ ภายในเฟรม SDU ของโปรโตคอล NCL แสดงให้เห็นในภาคผนวก ค.

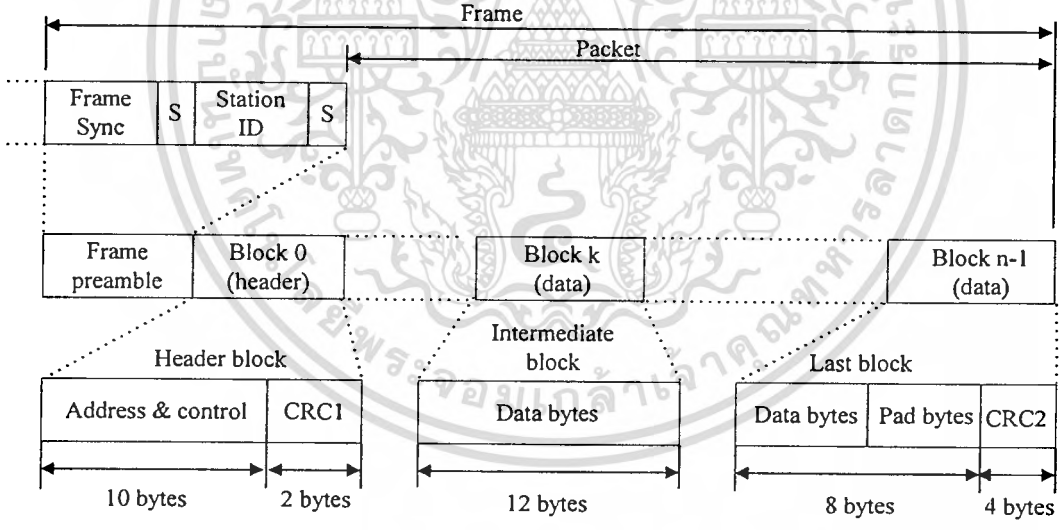
3.4 คุณสมบัติของโปรโตคอล RD-LAP (Radio Data - Link Access Protocol)

เป็นโปรโตคอลที่ใช้รับส่งข้อมูลกันระหว่างโมเด็มไร้สายและสถานีฐานของโครงข่ายเวิลด์ดาต้า ซึ่งได้ถูกพัฒนาเพื่อให้ได้ความเร็วในการสื่อสารสัญญาณวิทยุโดยใช้เทคโนโลยีของ Packet switching และใช้เทคนิคการผสมสัญญาณ (Modulation) แบบ 4-Level FSK ซึ่งเป็นวิธีการผสมสัญญาณทางความถี่ โดยที่ความถี่จะเปลี่ยนแปลงไปตามสัญญาณข้อมูลดิจิตอลที่นำมาผสมสัญญาณ ให้อัตราความเร็วข้อมูลสูงสุดที่ 19.2 kbps และมีการจัดการสัญญาณขาเข้าสถานีวิทยุของช่องสัญญาณหนึ่ง ๆ นั้น โดยใช้เทคนิคของ Slotted Digital Sense Mutiple Access (Slotted-DSMA) เพื่อหลีกเลี่ยงการชนกัน (Collision) ของช่องสัญญาณ (Channel slotting) โปรโตคอล RD-LAP ประกอบด้วย 3 ระดับชั้น คือ Physical Layer, Data link layer และ Channel management layer ในส่วนของ Data link layer แบ่งออกเป็นระดับชั้นย่อยๆ ได้เป็น 3 Sublayers คือ Channel access sublayer, Protocol sublayer และ Service sublayer ในรูปที่ 3.26 แสดงลักษณะโปรโตคอลสเตคของ RD-LAP

รูปที่ 3.26 แสดงลักษณะโปรโตคอลสแตคของ RD-LAP



รูปที่ 3.27 แสดงลักษณะโครงสร้างของเฟรม RD-LAP



จากรูปที่ 3.26 Service sublayer จะทำหน้าที่รับส่งข้อมูล ถ้าข้อมูลมีความยาวมากๆ จะถูกแบ่งออกเป็นหน่วยบริการข้อมูล (PDU) แพ็คเก็ตย่อย ๆ ในที่นี้ความยาวข้อมูลต่อหนึ่งแพ็คเก็ตเท่ากับ 512 ไบต์ จากนั้นจะมี Protocol sublayer ทำหน้าที่ตรวจสอบลำดับ (Sequencing) ของแต่ละ PDU, การตอบกลับ (Acknowlegdement), การร้องขอแพ็คเก็ตใหม่ (Retransmission) และการตรวจสอบการซ้ำกันของแพ็คเก็ต PDU โดยมี Channel access sublayer ทำการเติมฟิลด์ Frame preamble ให้กับทุกๆ แพ็คเก็ต PDU Frame preamble จะประกอบด้วย Frame synchronous และ Station Identifier ซึ่งจะเป็นตัวบอกสถานีและช่องสัญญาณที่กำลังใช้งานอยู่ สำหรับการทำงานของ

Channel management จะคอยควบคุมการสแกนหาช่องสัญญาณ ประเมินหาคุณภาพของสัญญาณ และการประกาศหมายเลขของโมเด็มไร้สายเข้าสู่โครงข่ายสื่อสาร

รายละเอียดของฟิลด์ในเฟรม RD-LAP ดังรูปที่ 3.27 มีดังนี้

1). ฟิลด์ Frame Preamble เป็นส่วนควบคุมเฟรมข้อมูล ประกอบด้วย Frame sync บอกถึงเฟรมข้อมูลเริ่มต้น, Station ID เป็นตัวบอกสถานีและช่องสัญญาณที่ใช้งาน และ S หรือ Channel status symbol ควบคุมการใช้ช่องสัญญาณ โดยจะส่งเฟรมข้อมูลเมื่อช่องสัญญาณว่าง

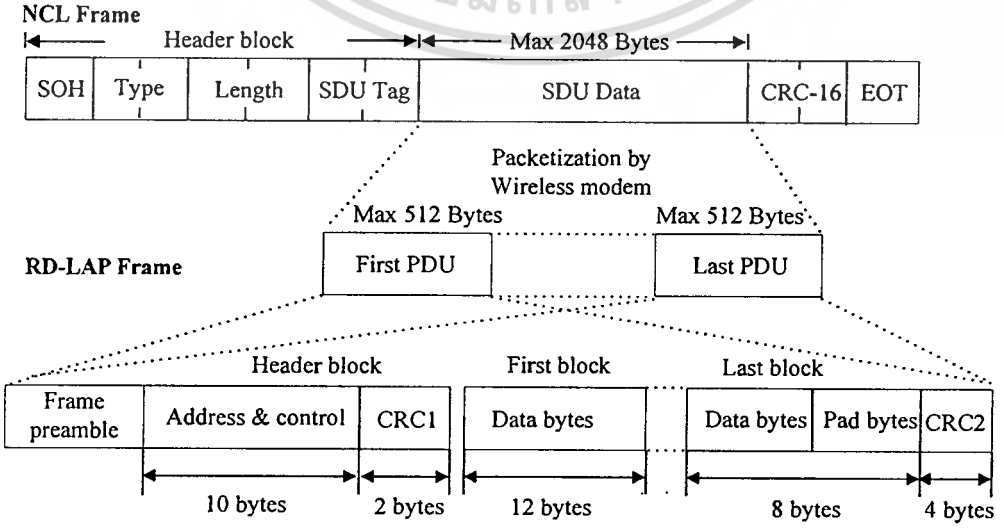
2). ฟิลด์ Address & control และ CRC1 (Cyclic Redundancy Check) หรือ CRC-16 เป็นส่วนหัวของแพ็คเก็ต (Packet) ประกอบด้วยตำแหน่งที่อยู่ของผู้รับและผู้ส่ง, ส่วนการตรวจสอบลำดับข้อมูล (Sequence number) ของผู้ส่ง ให้ทางฝ่ายผู้รับสามารถตรวจสอบความถูกต้องได้ และมี CRC1 เป็นตัวตรวจสอบความถูกต้องของข้อมูลส่วนหัวแพ็คเก็ต

3). ฟิลด์ Data bytes เป็นส่วนข้อมูล สามารถมีหลาย ๆ บล็อคข้อมูลต่อหนึ่งแพ็คเก็ต ในแต่ละบล็อคข้อมูลมีขนาด 12 ไบต์

4). ฟิลด์ Data bytes บล็อคสุดท้าย (Last block) จะมีส่วน Pad bytes สำหรับเติมข้อมูล (Dummy) เพื่อให้ได้ขนาดเท่ากับ 12 ไบต์ และมี CRC2 หรือ CRC-32 ทำหน้าที่ตรวจสอบความถูกต้องของส่วนข้อมูลทั้งหมด

เราสามารถแสดงรูปแบบการแปลงเฟรมข้อมูลจากโปรโตคอล NCL มาเป็นโปรโตคอล RD-LAP ดังในรูปที่ 3.28

รูปที่ 3.28 แสดงรูปแบบการแปลงเฟรมข้อมูลจากโปรโตคอล NCL เป็น RD-LAP



ในรูปที่ 3.28 แสดงการแปลงเฟรมโปรโตคอลระหว่าง NCL และ RD-LAP การทำงานตรงส่วนนี้จัดการโดยโปรแกรมที่ฝังอยู่บนโมเด็มไร้สาย เวลาส่งข้อมูลโปรแกรมประยุกต์ของผู้ใช้จะปะส่วนหัวเฟรมด้วยอักขระเริ่มต้นเฟรม ชนิดของเฟรม ความยาวของข้อมูล และป้ายประกาศข้อมูล เป็นต้น มีการคำนวณค่า Checksum และปิดท้ายด้วยอักขระสิ้นสุดเฟรม ซึ่งทั้งหมดนี้จะถูกตัดออกโดยโมเด็มไร้สายเหลือแต่เพียงส่วนข้อมูล SDU ซึ่งมีความยาวไม่เกิน 2048 ไบต์ เท่านั้น แล้วส่งออกไปด้วยโปรโตคอล RD-LAP โดยข้อมูลจะถูกแบ่งเป็นแพ็คเก็ตย่อย ๆ (PDU) มีความยาวไม่เกิน 512 ไบต์ ส่งออกไปยังสถานีฐานบนโครงข่ายเวิลด์ดาต้า ตัวสถานีฐานจะทำการจัดเรียงแพ็คเก็ตใหม่ เพื่อให้ได้ข้อมูล SDU อีกครั้งหนึ่ง จากนั้นส่งต่อไปยังอุปกรณ์เกตเวย์โครงข่ายวิทยุ (RNG) เพื่อทำหน้าที่หาเส้นทางว่าข้อมูล SDU นี้จะส่งต่อไปยังผู้ใด ถ้าต้องการส่งไปยังอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ (Mobile terminal) ก็จะส่งไปยังสถานีฐาน แล้วส่งด้วยโปรโตคอล RD-LAP ออกไปยังโครงข่าย ผู้รับจะรับข้อมูลเข้ามาที่โมเด็มไร้สายด้วยโปรโตคอล RD-LAP เช่นเดียวกัน จากนั้นแปลงเป็นเฟรมโปรโตคอล NCL โดยโมเด็มไร้สายจะทำการปะส่วนหัวเฟรม ค่า Checksum และส่วนท้ายเฟรมเข้าไป แล้วส่งเฟรมข้อมูลไปยังเครื่องคอมพิวเตอร์ต่อไป

บทที่ 4
เทคนิคการบีบอัดข้อมูล

การบีบอัดข้อมูลเป็นการลดความซ้ำซ้อนของข้อมูลโดยทำให้ข้อมูลมีขนาดเล็กลง ซึ่งเรามักพบกับการใช้งาน 2 ประเภท คือ ในอุปกรณ์จัดเก็บข้อมูล (Storage device) และการส่งข้อมูลบนโครงข่าย (Data transmission) สิ่งที่เราต้องการของการใช้งานทั้ง 2 ประเภท คือ เวลาที่ใช้ในการประมวลผล (Computation time) หน่วยความจำที่ใช้ในการประมวลผล และอัตราการบีบอัดข้อมูล (Compression ratio) ในระบบการจัดเก็บข้อมูลนั้นเราต้องการอัตราการบีบอัดข้อมูลกับการได้ขนาดของแฟ้มข้อมูลที่เล็กลงมากๆ เป็นสำคัญ ทั้งนี้ก็เพื่อประหยัดค่าใช้จ่ายในการจัดหาอุปกรณ์จัดเก็บเพิ่มเติม แต่ในทางกลับกันการส่งข้อมูลบนโครงข่ายสื่อสารเราสนใจเวลาที่ใช้ในการประมวลผลเป็นอันดับแรก แต่เราก็ต้องสนใจอัตราการบีบอัดของข้อมูลด้วย ทั้งนี้ก็เพื่อไม่ให้ประสิทธิภาพโดยรวมด้อยลงไป ดังนั้นในบทนี้จึงได้นำเสนอการบีบอัดข้อมูลด้วยวิธีการของ LZP (Lempel-Ziv-Prediction) โดยการปรับปรุงอัลกอริทึมเดิมของ Lempel-Ziv[10], [11] ให้มีความเหมาะสมทางด้านความเร็วและอัตราการบีบอัดข้อมูลกับการส่งข้อมูลในลักษณะโต้ตอบ เป็นการอาศัยหลักการของ Context Modeling และ Static Huffman Coding ช่วยในการบีบอัดข้อมูลให้เล็กลง เพื่อแก้ปัญหาแบนด์วิต (Bandwidth) ที่มีอย่างจำกัด และลดเวลาการส่งข้อมูลให้มีประสิทธิภาพมากยิ่งขึ้น

วิธีการบีบอัดข้อมูลสามารถแบ่งออกเป็น 2 ประเภทคือ Lossy data compression และ Lossless data compression ทั้งสองประเภทมีวิธีการแตกต่างกันคือ Lossy data compression จะยอมให้ข้อมูลผิดเพี้ยนไปจากข้อมูลต้นฉบับได้เมื่อทำการขยายขนาดข้อมูลที่ถูกบีบอัดกลับคืนมา เช่น ข้อมูลประเภทรูปภาพและเสียง เป็นต้น ซึ่งเราให้การผิดเพี้ยนของข้อมูลเกิดขึ้นได้มากหรือน้อยตามคุณภาพที่เรายอมรับได้ ส่วนวิธีการของ Lossless data compression จะไม่อนุญาติให้ข้อมูลมีการผิดเพี้ยนไปจากข้อมูลต้นฉบับได้เมื่อทำการขยายขนาดข้อมูลที่ถูกบีบอัดกลับคืนมา เพราะการผิดเพี้ยนของข้อมูลจะสังเกตเห็นได้และยอมรับไม่ได้ เรามักพบวิธีการนี้ในการบีบอัดข้อมูลประเภทตัวอักษร (Text compression) ได้มีการพัฒนาขึ้นมาในหลาย ๆ วิธี เช่น Channon-Fano coding, Huffman coding, Arithmetic coding และที่นิยมใช้กันมากก็คือวิธีการของ Lempel-Ziv ซึ่งในบทนี้ได้นำแนวทางของ Lempel-Ziv มาใช้ โดยจะให้มีการทำงานใน 2 ขั้นตอนคือ พยายามที่จะหารูปแบบของข้อมูลที่เหมาะสมก่อนที่จะทำการบีบอัด ที่เรียกว่าการทำ Modeling และพยายามที่จะแทนข้อมูลที่จัดรูปแบบแล้วด้วยรหัสตัวเลขที่มีความยาวน้อยกว่า หรือเราเรียกว่าการทำ Coding

4.1 เปรียบเทียบกับหลักการเดิม

อัลกอริทึมของ LZ ถูกค้นพบในปี 1977 หรือเรียกว่า LZ77 [10] เป็นวิธีการทำสตริงแมทชิ่ง (String Matching) ระหว่างข้อมูลก่อนหน้า (Previous message) และข้อมูลปัจจุบัน (Current message) เมื่อเปรียบเทียบแล้วมีข้อมูลแมทช์กันได้จะเขียนแมทช์แฟล็ก (Match flag) ตำแหน่งที่มีการแมทช์ (Match position) และความยาวของข้อมูลที่แมทช์กัน (Match Length) แต่ถ้าไม่มีข้อมูลแมทช์กันจะเขียนแมทช์แฟล็กและข้อมูลเดิมเก็บไว้ เพื่อใช้เป็นตัวชี้ตำแหน่งข้อมูลเวลามีการขยายขนาดข้อมูลเกิดขึ้น ซึ่งการเปรียบเทียบข้อมูลของวิธีนี้จะกระทำภายในตารางหน้าต่าง (Window table) ที่กำหนดไว้เท่านั้น และยังมีการกำหนดขนาดตารางหน้าต่างไว้คงที่ (Fix window) เมื่อข้อมูลมีขนาดใหญ่จะทำให้ตารางหน้าต่างถูกแบ่งย่อยออกเป็นหลายหน้าต่าง ซึ่งข้อมูลที่อยู่แต่ละตารางหน้าต่างมีโอกาสเหมือนกันได้ แต่เราไม่สามารถนำมาแมทช์กันได้ ทำให้ข้อมูลมีความซ้ำซ้อนกันมากเกินไป อัตราการบีบอัดข้อมูลที่ได้รับจึงไม่ดีเท่าที่ควร ครั้นเมื่อเราต้องการเพิ่มประสิทธิภาพตรงส่วนนี้ ก็จะต้องเพิ่มขนาดตารางหน้าต่างให้ใหญ่ขึ้นและต้องใช้หน่วยความจำมากขึ้น

ต่อมาในปี 1978 Lempel และ Ziv ได้ปรับปรุงอัลกอริทึมของ LZ77 และเรียกชื่อใหม่ว่า LZ78 [14] เวอร์ชั่นนี้ได้ยกเลิกการใช้ตารางหน้าต่างไป เปลี่ยนมาใช้วิธีการเก็บข้อมูลทั้งหมดลงในตารางข้อมูลแบบพจนานุกรม (Dictionary) แทน โดยใช้ดัชนี (Index) เป็นตัวค้นหาตำแหน่งข้อมูลที่จัดเก็บภายในตารางข้อมูล ถ้ามีการแมทช์กันเกิดขึ้นจะเขียนตำแหน่งข้อมูลที่แมทช์กันและเขียนตัวอักขระถัดไป (Next character) ที่ไม่แมทช์กันไว้ ในขณะเดียวกันให้รวมตัวอักขระถัดไปที่ไม่แมทช์กันเข้ากับข้อมูลที่แมทช์กันก่อนหน้าเพิ่มเข้าไปในตารางข้อมูล แต่ถ้าไม่มีการแมทช์กันจะเขียนตำแหน่งข้อมูลเป็นศูนย์และให้เขียนข้อมูลเดิมเพิ่มเข้าไปในตารางข้อมูล ซึ่งจะเห็นว่าข้อมูลเอ้าท์พุทที่ได้รับจะคล้ายคลึงกับ LZ77 แต่ LZ78 นี้จะได้รับประสิทธิภาพด้านความเร็วและอัตราการบีบอัดข้อมูลที่ดีกว่า เนื่องจากไม่มีการจำกัดความยาวของข้อมูลที่ใช้ในการเปรียบเทียบนั่นเอง หลังจากนั้นในปี 1984 Terry Welch ได้ใช้ LZ78 เป็นพื้นฐานในการพัฒนาอัลกอริทึมการบีบอัดข้อมูลบนอุปกรณ์ควบคุมดิสก์ (Disk controller) โดยใช้ชื่อว่า LZW (Lempel-Ziv-Welch) [12] วิธีการนี้นำหลักการแบบพจนานุกรม (Dictionary) มาใช้ในการเปรียบเทียบข้อมูลเช่นเดียวกับ LZ78 แต่แตกต่างกันที่ข้อมูลเอ้าท์พุท คือวิธีการของ LZW จะเก็บเพียงรหัสตัวเลข (Codeword) แสดงตำแหน่งข้อมูลในพจนานุกรมเท่านั้น เช่นเลข 0-255 แทนตัวอักขระ (Character) ตามค่าแอสกี และตั้งแต่เลข 256 เป็นต้นไป จะแทนวลี (Phrase) ที่มากกว่า 2 ตัวอักขระขึ้นไป ทำให้วิธีการนี้ได้รับอัตราการบีบอัดและประสิทธิภาพด้านความเร็วดีกว่าทั้งของ LZ77 และ LZ78 ซึ่งต่อมา LZW นี้จึงได้มีการนำมาใช้กันอย่างแพร่หลาย และกลายมาเป็นมาตรฐานบนระบบปฏิบัติการ UNIX นอกจากนี้ยังได้กำหนดเป็นมาตรฐาน CCITT V.42 bis ใช้ในอุปกรณ์ประเภทโมเด็มอีกด้วย

วิธีการต่าง ๆ ตามที่กล่าวมาข้างต้น ข้อมูลบีบอัดที่ถูกแทนด้วยรหัสตัวเลขยังใช้บิตมากเกินไป มีขบวนการทำงานล่าช้า และมีอัตราการบีบอัดข้อมูลไม่ดีนัก ดังนั้นแนวทางในการปรับปรุงอัลกอริทึมของ LZ นี้ ทำโดยการลดขนาดรหัสตัวเลขของข้อมูลบีบอัดให้น้อยลง โดยพิจารณาประเด็นหลักคือการเพิ่มค่าความเป็นไปได้ (Probability) ให้มากขึ้น ซึ่งค่าความเป็นไปได้เท่ากับ 1/N เมื่อ N คือ จำนวนข้อมูลที่เกิดขึ้นทางเอ้าท์พุททั้งหมด และเราต้องทำให้ข้อมูลเกิดการแมทช์กันให้ได้ยาวที่สุด หรือไม่มีการจำกัดความยาวของข้อมูลที่ทำการแมทช์ ทั้งนี้ก็เพื่อลดค่า N ให้น้อยลงนั่นเอง โดยอาศัยหลักการของ Context modeling และ Static Huffman coding เราเรียกอัลกอริทึมที่ปรับปรุงใหม่นี้ว่า LZP (Lempel-Ziv-Prediction) เราใช้วิธีการของ Context modeling มาลดความซ้ำซ้อนของข้อมูลอินพุทลงก่อน จากนั้นจึงใช้วิธีการของ Static Huffman coding ทำการแทนข้อมูลด้วยรหัสตัวเลขที่มีความยาวน้อยกว่า โดยพื้นฐานแล้ววิธีการของ Context modeling จะซ้ำและต้องการหน่วยความจำมากในการปฏิบัติคำสั่ง มีการกำหนดรูปแบบเป็น Order-i context เมื่อกำหนดค่า i เป็นค่าคงที่แล้ว ยิ่งให้ค่าลำดับ (Order) มีค่าสูงจะยิ่งใช้หน่วยความจำมากในการเก็บข้อมูลก่อนหน้าเพื่อนำมาเปรียบเทียบกับข้อมูลปัจจุบัน สำหรับกรณีที่เป็น Order-3 ที่มีการใช้โครงสร้างข้อมูลแบบลิงค์ลิสต์หรืออาร์เรย์ในการปฏิบัติอัลกอริทึมจะใช้หน่วยความจำมากถึง 500 กิโลไบต์ แต่ถ้าเราใช้โครงสร้างข้อมูลแบบแฮชชิ่ง (Hashing) จัดเก็บข้อมูลแล้ว จะช่วยลดขนาดหน่วยความจำได้มาก เราสามารถเลือกจำนวนบิตของค่าแฮชได้ตามขนาดข้อมูลที่ใช้บีบอัด เช่น เมื่อเลือกจำนวนบิตของค่าแฮชเป็น 12 บิต จะได้กลุ่มข้อมูลเก็บในตารางดัชนีค่าแฮช (Hash table index) เท่ากับ 2^{12} หรือ 4096 กลุ่ม แต่ละกลุ่มมีขนาดเวิร์ดเท่ากับ 2 ไบต์ ดังนั้นจะใช้หน่วยความจำในตารางแฮชเท่ากับ 8 กิโลไบต์ หลังจากขบวนการ Context modeling สิ้นสุดลง เราจะได้ข้อมูลเอ้าท์พุท 2 ส่วนคือ ข้อมูลที่แมทช์กันได้ (Literal) และความยาวข้อมูลที่แมทช์กันได้ (Match-length) จากนั้นนำข้อมูลทั้งสองส่วนนี้ไปผ่านขบวนการ Static Huffman coding เพื่อลดจำนวนบิตข้อมูลให้น้อยลง โดยพิจารณาจากความเป็นไปได้ (Probability) ที่เกิดขึ้นของตัวอักขระแต่ละตัว ถ้าตัวอักขระนั้นมีความเป็นไปได้สูงหรือมีจำนวนมากแล้ว ตัวอักขระจะถูกแทนด้วยรหัสตัวเลขที่มีจำนวนบิตน้อย ในทางกลับกันถ้าตัวอักขระมีความเป็นไปได้ต่ำก็จะถูกแทนด้วยรหัสตัวเลขที่มีจำนวนบิตมาก จึงกล่าวได้ว่ายิ่งข้อมูลซ้ำซ้อนกันมากๆ แล้วอัลกอริทึมนี้จะให้อัตราการบีบอัดข้อมูลมากขึ้น

4.2 หลักการทำงานของ LZW

การทำงานของ LZW จะเริ่มต้นพิจารณาตัวอักขระแต่ละตัวตามลำดับ เพื่อหากลุ่มสตริงย่อย (Sub string) ที่ยาวที่สุดที่ตรงกับสตริงในพจนานุกรม จนกระทั่งไม่สามารถหาได้ จึงทำการเพิ่มสตริงที่มาใหม่เข้าไว้ในพจนานุกรม สตริงนี้จะถูกแทนด้วยรหัสตัวเลขออกเป็นเอ้าท์พุท เช่น รหัส

ตัวเลขตั้งแต่ 0-255 เป็นค่าแอสกีของตัวอักขระที่มีขนาด 8 บิต และรหัสตัวเลขที่เพิ่มเข้ามาใหม่จะมีค่าตั้งแต่ 256 เป็นต้นไป ที่มีขนาด 9 บิต โดยค่าจำนวนบิตของรหัสตัวเลขนี้จะเพิ่มค่าครั้งละหนึ่งเมื่อนับถึง $2^{n_{บิต}}$ เช่น รหัสตัวเลขตั้งแต่ 512-1024 จะมีขนาดเท่ากับ 10 บิต เป็นต้น

กลุ่มสตริงย่อยที่กล่าวถึงตั้งแต่ต้นประกอบด้วย ส่วนสตริงนำหน้า (Prefix string หรือ w) และส่วนตัวอักขระที่เพิ่ม (Extension character หรือ K) สามารถแสดงอัลกอริทึมการบีบอัดข้อมูลของ LZW ได้ดังนี้

```
set w เป็นค่าเริ่มต้น
loop
    อ่านตัวอักขระ K ตัวถัดไป
    if wK มีอยู่ในพจนานุกรม
        w = wK
    else
        ให้รหัสตัวเลขของ w เป็นเอ้าท์พุท
        เพิ่ม wK เข้าไปในพจนานุกรม
        w = K
endloop
```

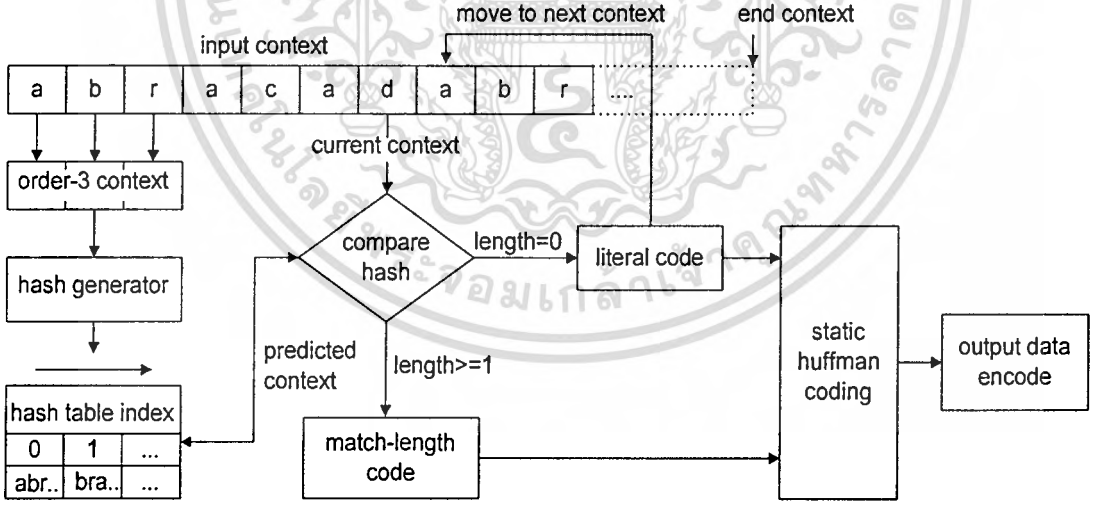
จากอัลกอริทึมของ LZW ในทุก ๆ รอบการทำงาน จะมีการรวมสตริงนำหน้า (w) เข้ากับตัวอักขระใหม่ (K) แล้วเกิดเป็นสตริง wK และค้นหาว่ามีอยู่ในพจนานุกรมหรือไม่ ถ้ามีอยู่ในพจนานุกรม จะให้สตริงนำหน้ายาวขึ้นอีก 1 ตัวอักขระ แต่ถ้า wK ไม่อยู่ในพจนานุกรม ก็จะเพิ่มสตริงใหม่เข้าไปในพจนานุกรม แล้วให้เอ้าท์พุทเป็นรหัสตัวเลขของสตริงนำหน้า สตริงนำหน้าก็จะเริ่มต้นใหม่มีค่าเป็น K ยาวเพียง 1 ตัวอักขระ ขบวนการนี้จะทำซ้ำจนกระทั่งหมดข้อมูลอินพุท

เมื่อต้องการขยายขนาดข้อมูลก็จะสร้างตารางข้อมูลแบบพจนานุกรมขึ้นมาใหม่ โดยตารางข้อมูลนั้นจะเป็นลักษณะเดียวกันกับตอนทำการบีบอัดข้อมูล เมื่ออ่านค่ารหัสตัวเลขขึ้นมาจะทำการหาสตริงใหม่และเพิ่มเข้าไปในพจนานุกรม เพื่อนำกลับข้อมูลเดิมออกมา สามารถหารายละเอียดเพิ่มเติมได้ใน [12]

4.3 หลักการทำงานของ LZP

หลักการทำงานเบื้องต้นของ LZP จะเริ่มต้นด้วยขบวนการ Context Modeling โดยที่ข้อมูลอินพุทแต่ละไบต์จะถูกจัดอยู่ในรูปแบบของ Order-3 context จากนั้นนำไปคำนวณค่าในฟังก์ชั่นแฮช (Hash function) ผลลัพธ์จะได้ค่าแฮช (Hash value) ที่เป็นตัวเลขจำนวนเต็ม ใช้เป็นตัวชี้ตำแหน่งข้อมูลก่อนหน้า (Previous message) ที่เก็บในดัชนีตารางแฮช (Hash table index) เพื่อนำไปเปรียบเทียบกับข้อมูลปัจจุบัน (Current message) ถ้าพบว่าข้อมูลสามารถแมทช์กันได้ จะทำการเก็บค่าความยาวข้อมูลที่แมทช์กันได้ (Match-length) ไว้ พร้อมกับนำข้อมูลปัจจุบันที่แมทช์กันไปเก็บไว้ในดัชนีตารางแฮชแทนที่ตำแหน่งของข้อมูลก่อนหน้านั้น แต่ถ้าไม่แมทช์กันจะเก็บข้อมูลเดิมไว้ ผลลัพธ์ของข้อมูลที่ไม่สามารถแมทช์กันได้เราเรียกว่า Literal ขบวนการนี้จะกระทำจนกระทั่งจบบล็อคข้อมูล หลังจากนั้นจึงนำข้อมูลที่ไม่สามารถแมทช์กันได้และค่าความยาวข้อมูลที่แมทช์กันได้มาผ่านขบวนการของ Static Huffman coding ในรูปที่ 4.1 แสดงไดอะแกรมการทำงานการบีบอัดข้อมูล (Compression) ของ LZP

รูปที่ 4.1 ไดอะแกรมการทำงานการบีบอัดข้อมูล (Compression) ของ LZP



จากรูปที่ 4.1 เราสามารถแสดงขั้นตอนการทำงานการบีบอัดข้อมูลของ LZP ได้ดังนี้

ขั้นตอนที่ 1 กลุ่มข้อมูลอินพุท (Input context) ถูกจัดให้อยู่ในรูปแบบของ Order-3 context คือมีความยาวข้อมูลขนาด 3 ไบต์ เริ่มต้นการทำงานกลุ่มข้อมูล 3 ไบต์แรกจะถูกนำเก็บไว้เป็นข้อมูลก่อนหน้าไว้ก่อน

<u>ขั้นตอนที่ 2</u> นำกลุ่มข้อมูลอินพุทขนาด 3 ไบต์มาคำนวณในฟังก์ชั่นแฮช (Hash function) เพื่อใช้เป็นตัวชี้ตำแหน่งข้อมูลในขั้นตอนของการเปรียบเทียบข้อมูลก่อนหน้าและข้อมูลปัจจุบัน การทำฟังก์ชั่นแฮชเป็นการประหยัดหน่วยความจำในการจัดเก็บข้อมูลที่ใช้ในการเปรียบเทียบ รูปแบบของฟังก์ชั่นแฮชของ Order-3 context คือ

$$H = (C_1 \wedge (C_2 << p_1) \wedge (C_3 << p_2)) \& ((1 << h) - 1) \quad \text{.....(4.1)}$$

ในเมื่อ ^ คือการทำลอจิก Exclusive-OR

& คือการทำลอจิก AND

<< คือการเลื่อนตำแหน่งบิตไปทางซ้าย

C_1, C_2, C_3 คือค่าของกลุ่มข้อมูลอินพุท (Context) มีขนาด 3 ไบต์ หรือ 24 บิต

H คือค่าของดัชนีตารางแฮช (Hash table index)

ℓ คือขนาดความยาวของ Context มีหน่วยเป็นบิต

h คือขนาดของตารางแฮช มีหน่วยเป็นบิต ; $0 < h < \ell$

p_1 คือพารามิเตอร์ตัวที่ 1 มีค่ามาจาก ((h >> 1) -2) | 1 ; $0 < p_1 < h$

p_2 คือพารามิเตอร์ตัวที่ 2 มีค่ามาจาก (h - (h >> 2) - 2) | 1 ; $0 < p_2 < h$

| คือการทำลอจิก OR

ยกตัวอย่างจากสมการที่ 4.1 โดยการเลือก h เท่ากับ 12 บิต ค่า ℓ เท่ากับ 24 บิต เราได้ค่า p_1 จาก ((12 >> 1) - 2) | 1 เท่ากับ 5 และ p_2 จาก (12 - (12 >> 2) - 2) | 1 เท่ากับ 7 ดังนั้นเราได้ฟังก์ชั่นแฮชตามตัวอย่างนี้คือ

$$H = (C_1 \wedge (C_2 << 5) \wedge (C_3 << 7)) \& ((1 << 12) - 1) \quad \text{.....(4.2)}$$

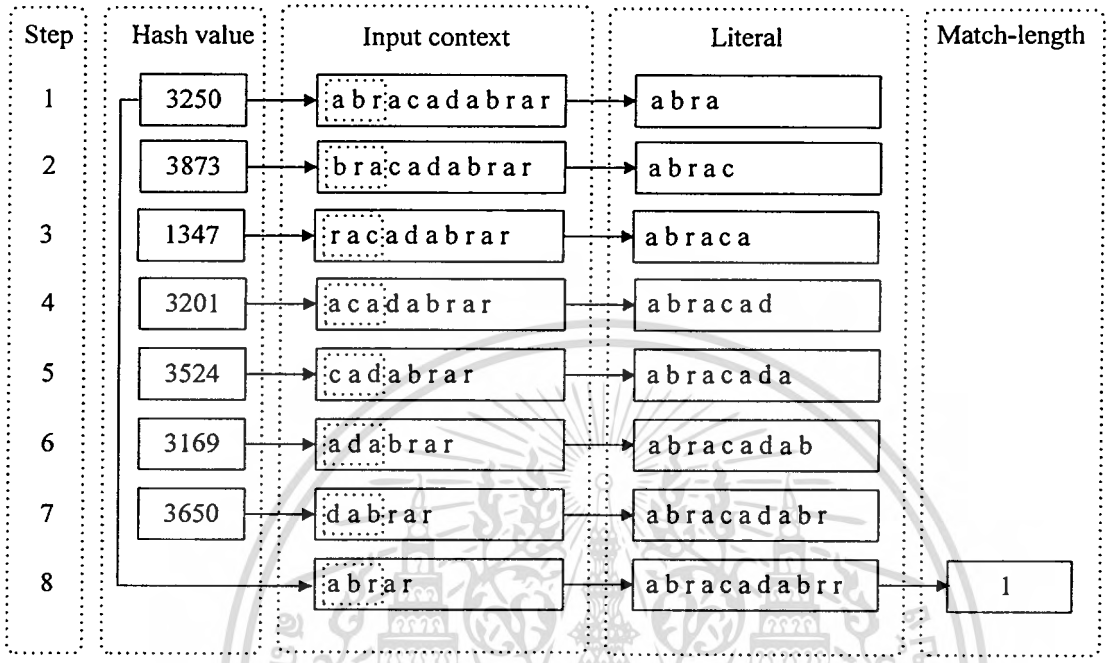
พารามิเตอร์ที่สำคัญของฟังก์ชั่นแฮช คือค่า h และ (p_1, p_2) โดยค่า h เป็นตัวกำหนดการใช้ขนาดหน่วยความจำของข้อมูลที่ใช้ในการเปรียบเทียบ และค่า (p_1, p_2) เป็นตัวกำหนดการกระจายตัว (Distribution) ของค่า H ทั้งสองค่านี้มีความสัมพันธ์ซึ่งกันและกัน การเลือกค่าขนาดของตารางแฮช (h) จะขึ้นอยู่กับขนาดของข้อมูลอินพุทที่ใช้ในการบีบอัด คือถ้าข้อมูลเป็นลักษณะแฟ้มข้อมูล การเลือกค่า h จะต้องสูงขึ้น เช่น 15 บิต จะได้ขนาดหน่วยความจำในตารางแฮชเท่ากับ 64 กิโลไบต์ เป็นต้น แต่ถ้าข้อมูลเป็นลักษณะเฟรมข้อมูลที่ใช้รับส่งในโครงข่ายสื่อสาร การเลือกค่า h จะน้อย

ลง เช่น ใช้ 11 บิต จะได้ขนาดหน่วยความจำในตารางแฮชเท่ากับ 4 กิโลไบต์ และการกระจายตัวของ (p1 และ p2) ก็น้อยลงตามด้วย ทั้งนี้ก็เพื่อประหยัดการใช้หน่วยความจำนั่นเอง ในทางอุดมคติแล้วค่าของ p_1 หรือ p_2 จะเท่ากับ ℓ - h ควรเลือกค่าของ p_1 และ p_2 เป็นตัวเลขจำนวนเฉพาะ (Prime) ที่หารด้วย 1 และตัวมันเองลงตัวเท่านั้น เพื่อป้องกันไม่ให้ข้อมูลสองค่าเกิดการชนกัน (Collision) หรือให้เกิดการชนกันน้อยที่สุด ซึ่งการชนกันเกิดขึ้นจากกรณีที่ข้อมูลสองค่าอยู่ในตำแหน่งเดียวกันในตารางแฮช จึงเกิดการรวมกันขึ้นในตารางแฮชเดียวกัน การเกิดปรากฏการณ์เช่นนี้ไม่มีผลต่อความถูกต้องในการบีบอัดข้อมูลแต่อย่างใด แต่จะมีผลต่อประสิทธิภาพ (Performance) ของอัลกอริทึม การป้องกันให้เกิดการชนกันน้อยที่สุดทำได้โดยการเลือกลำดับ (Order) ให้ต่ำ แต่ถ้าเลือกใช้ลำดับสูง ๆ จะเกิดการชนกันมากและต้องมีการแก้ปัญหาการชนกันไว้ด้วย นอกจากนี้การเลือกค่า h น้อยเกินไปจะมีโอกาสเกิดการชนกันมากขึ้นเช่นเดียวกัน สำหรับ Order-3 ที่นำเสนอนี้มีความเป็นไปได้ที่จะเกิดการชนกันน้อย จึงไม่มีผลต่อประสิทธิภาพด้านความเร็วมากนัก

ขั้นตอนที่ 3 เราจะได้ค่า H เป็นตัวชี้ (Index) ตำแหน่งข้อมูลก่อนหน้าที่เก็บในดัชนีตารางแฮช และถูกนำมาเปรียบเทียบกับค่า H ที่คำนวณจากข้อมูลปัจจุบัน ถ้าค่า H ของข้อมูลปัจจุบันเหมือนกับค่า H ของข้อมูลก่อนหน้า ให้ทำการนับจำนวนความยาวข้อมูลทีละไบต์ต่อจากข้อมูล 3 ไบต์ที่ใช้คำนวณค่าแฮช จนกระทั่งไม่มีข้อมูลแมทช์กันได้อีก โดยไม่มีการจำกัดจำนวนความยาว จากนั้นทำการเก็บค่าความยาวของข้อมูลที่แมทช์กันได้ (Match-length) ไว้ และทำการแทนที่ข้อมูลปัจจุบันที่ตำแหน่งค่า H ของข้อมูลก่อนหน้าชี้อยู่ แต่ถ้าข้อมูลไม่สามารถแมทช์กันได้ จะเก็บค่าข้อมูลอินพุทในปัจจุบันไว้คงเดิม เราเรียกข้อมูลอินพุทที่ไม่สามารถแมทช์กันได้นี้ว่า Literal ในรูปที่ 4.2 แสดงขบวนการทำงานของ Order-3 Context modeling

ทั้งนี้ค่าความยาวข้อมูลที่แมทช์กันได้กำหนดให้มีขนาด 1 ไบต์ เนื่องจากต้องการให้อยู่ในขอบเขตเดียวกันกับข้อมูล Literal ซึ่งเป็นรหัสแอสกี โดยมีค่าตั้งแต่ 0 ถึง 254 เราสงวนค่า 255 หรือ 0xFF ไว้ เพื่อบ่งบอกถึงค่าความยาวที่เกิน 255 ซึ่งเมื่อความยาวเกิน 255 จะต้องนำค่าความยาวที่เกินไปหักลบด้วยเลขจำนวนเต็ม 255 และให้ใส่ค่า 0xFF ตรงเนื้อที่ส่วนหน้าค่าความยาวที่ได้จากการหักลบแล้ว เพื่อบอกให้ส่วนขยายข้อมูล (Data Decompress) แปลความหมายได้

รูปที่ 4.2 แสดงขบวนการทำงานของ Order-3 Context modeling



ตัวอย่างการคำนวณค่าแฮช (H) ในขั้นตอนที่ 1 ได้มาจากสมการที่ 4.2 คือ

$H = (\text{'a'} \wedge (\text{'b'} << 5) \wedge (\text{'r'} << 7)) \,\&\, ((1 << 12) - 1)$

$= (97 \wedge (98 << 5) \wedge (114 << 7)) \,\&\, ((1 << 12) - 1)$

$= 3250$

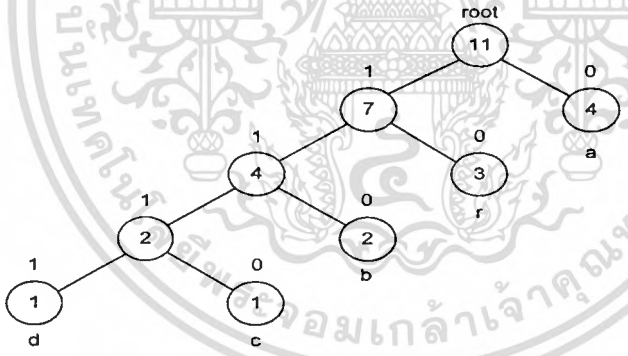
จากรูปที่ 4.2 ข้อมูลอินพุททั้งหมดคือ "abracadabrar"ในขั้นตอนที่ 1 กลุ่มข้อมูลอินพุท "abr" คำนวณด้วยฟังก์ชั่นแฮชแล้วได้ค่าแฮชเท่ากับ 3250 และเก็บเป็นข้อมูล Literal ไว้ก่อน และตัวอักษร "a" ยังไม่สามารถแมทช์กับใครได้จึงเป็นข้อมูล Literal ด้วย มาถึงในขั้นตอนที่ 8 จะได้ค่าแฮชเท่ากันกับในขั้นตอนที่ 1 และมีตัวอักษร "a" สามารถแมทช์กันได้กับข้อมูลที่เก็บในตารางแฮชของขั้นตอนที่ 1 ได้ความยาวของข้อมูลที่แมทช์กันเท่ากับ 1 สุดท้ายทำให้เราได้ข้อมูล Literal เป็น "abracadabrr"

<u>ขั้นตอนที่ 4</u> ขบวนการของ Order-3 context modeling จะดำเนินการไปจนกระทั่งสิ้นสุดบล็อคข้อมูล เราจะได้ข้อมูลเอ้าท์พุทออกมา 2 รูปแบบ คือ ข้อมูลที่ไม่สามารถแมทช์กันได้ (Literal) และค่าความยาวข้อมูลที่แมทช์กันได้ (Match-length)

<u>ขั้นตอนที่ 5</u> ข้อมูลที่ไม่สามารถแมทช์กันได้ และค่าความยาวข้อมูลที่แมทช์กันได้ ถูกนำไปผ่านขบวนการของ Static Huffman coding ตามลำดับ ทั้ง 2 ค่านี้จะแยกกันทำคนละครั้งกัน เริ่ม

แรกใช้วิธีการนับจำนวนตัวอักขระ (Character) ทั้งหมดที่เกิดขึ้นใน Literal และจำนวนตัวเลขของค่าความยาวข้อมูลที่แมทช์กันได้ จำนวนตัวอักขระที่พบถูกกำหนดเป็นค่าน้ำหนัก (Weight) ที่แสดงถึงค่าความเป็นไปได้ (Probability) ของข้อมูล โดยทำการจัดเรียงค่าน้ำหนักจากน้อยไปหามากด้วยวิธีการของ Radix sort ต่อจากนั้นจัดข้อมูลใหม่ให้อยู่ในโครงสร้างข้อมูลแบบไบนารีทรี (Binary tree) โดยมีโหนด (Node) เป็นกิ่งก้านสาขา ใช้เก็บข้อมูลข่าวสาร กฎเกณฑ์การเพิ่มโหนดแต่ละโหนดจะเริ่มจากค่าน้ำหนัก 2 ค่าที่ต่ำสุดก่อน ค่าน้ำหนักเมื่อรวมกันจะสร้างเป็นโหนดยอด (Parent node) แล้วทำการต่อยอดขึ้นไปจนกระทั่งเหลือเพียงโหนดเดียวเป็นจุดยอด หรือที่เราเรียกว่า รูตโหนด (Root node) ที่มีค่าน้ำหนักสูงที่สุด หลังจากนั้นจะทำการใส่บิตลงไปโดยเริ่มต้นจากจุดยอดของทรีลงมา ให้ใส่บิต "0" เมื่อท่องไปยังบริเวณกิ่งก้าน (Branch) ทางขวามือ และใส่บิต "1" บริเวณกิ่งก้านทางซ้ายมือ ทำจนกระทั่งเหลือเพียงหนึ่งใบ (Leaf) ถือว่าสิ้นสุด เราจะได้รหัสไบนารีที่ไม่ซ้ำกันมีจำนวนน้อยลงทดแทนตัวอักขระ แสดงตัวอย่างการสร้างไบนารีทรีของ Huffman ในรูปที่ 4.3

รูปที่ 4.3 ตัวอย่างการสร้างไบนารีทรีของ Huffman



จากรูปที่ 4.3 ตัวอย่างข้อมูลของ Literal คือ "abracadabrr" มีค่าน้ำหนักของ a เท่ากับ 4, b เท่ากับ 2, c เท่ากับ 1, d เท่ากับ 1 และ r เท่ากับ 3 การรวมค่าน้ำหนักจะเริ่มจากค่าต่ำที่สุดก่อนจนกระทั่งได้ค่าน้ำหนักของโหนดสุดท้ายเป็นจุดยอด (Root) จากนั้นใส่บิต "0" ที่โหนดขวามือ และใส่บิต "1" ทางซ้ายมือ โดยทำการสแกนอ่านค่าบิตจากจุดยอดลงมาหาโหนดของตัวอักษรแต่ละตัว ดังนั้นข้อมูล "abracadabrr" เราจะได้ค่าบิตในแต่ละตัวอักษรคือ a เท่ากับ "0" b เท่ากับ "110" c เท่ากับ "1110" d เท่ากับ "1111" และ r เท่ากับ "10" แสดงให้เห็นในตารางที่ 4.1 จะเห็นว่าตัวอักษรที่มีค่าความเป็นไปได้สูงสุดคือตัวอักษร a จะถูกแทนด้วยรหัสตัวเลขที่มีจำนวนบิตน้อยที่สุด และตัวอักษร c และ d ถูกแทนด้วยรหัสตัวเลขที่มีจำนวนบิตมากที่สุด เนื่องจากมีความเป็นไปได้ต่ำสุดนั่นเอง สุดท้ายเรารวมบิตทั้งหมดสามารถแทนค่าข้อมูลได้ 24 บิต คือ "011010011100111101101010"

ตารางที่ 4.1 แสดงค่าความเป็นไปได้และการแทนค่ารหัสตัวเลข (Codeword) ของ Huffman

ตัวอักษร	ค่าความเป็นไปได้	รหัสตัวเลขของ Huffman
a	4/11	0
b	2/11	110
c	1/11	1110
d	1/11	1111
r	3/11	10

ขั้นตอนที่ 6 สร้างตารางเข้ารหัส (Encode table) ของข้อมูลที่ไม่แมทช์กัน และค่าความยาวของข้อมูลที่แมทช์กัน จุดประสงค์การสร้างตารางเข้ารหัส ก็คือ เพื่อกำหนดรูปแบบการเก็บจำนวนตัวอักขระที่พบในข้อมูลอินพุททั้งหมด รหัสตัวเลขของ Huffman ของตัวอักขระแต่ละตัว และความยาวของรหัสตัวเลขแต่ละตัวอักขระลงในแฟ้มบีบอัดข้อมูล โดยพยายามหารูปแบบที่ประหยัดขนาดในการจัดเก็บให้มากที่สุด ยกตัวอย่างในตารางที่ 4.2 เป็นการจัดเก็บข้อมูลที่ไม่แมทช์กัน (Literal) ลงในตารางเข้ารหัส (Encode table) หลังจากผ่านขั้นตอนที่ 5 มาแล้ว สำหรับค่าความยาวของข้อมูลที่แมทช์กัน ก็มีวิธีการเก็บในลักษณะเดียวกัน

ขั้นตอนที่ 7 เขียนข้อมูลส่วนหัว (Header) และรหัสข้อมูลที่ผ่านการบีบอัดข้อมูลแล้วลงบนแฟ้มข้อมูลใหม่ ที่เรียกว่าแฟ้มข้อมูลบีบอัด ซึ่งข้อมูลส่วนหัวมีขนาด 16 ไบต์ แต่ละฟิลด์มีขนาด 4 ไบต์ ประกอบด้วย ขนาดของแฟ้มข้อมูลเดิม จำนวนของข้อมูลที่แมทช์กันไม่ได้ จำนวนของค่าความยาวข้อมูลที่แมทช์กันได้ และขนาดของข้อมูลที่แมทช์กันไม่ได้หลังผ่านขบวนการ Static Huffman coding แล้ว ตามลำดับ เพื่อเป็นข้อมูลให้ส่วนขยายขนาดข้อมูล (Decompression) ใช้สำหรับการขยายขนาดให้เป็นแฟ้มข้อมูลเดิมออกมา ส่วนสุดท้ายจะเป็นรหัสข้อมูลที่ได้ในขั้นตอนที่ 6 ประกอบด้วย รหัสข้อมูลของข้อมูลที่แมทช์กันไม่ได้ และรหัสข้อมูลของค่าความยาวข้อมูลที่แมทช์กันได้ ตามลำดับ

ตารางที่ 4.2 แสดงตัวอย่างการเก็บรหัสตัวเลขของ Huffman ในตารางเข้ารหัส (Encode table)

ความยาว(บิต)	ตัวเลขที่เก็บ*	รายละเอียด
8	114	ค่าสูงสุดของตัวอักษรในข้อมูลอินพุท จากตัวอย่างคือตัวอักษร 'r'
4	4	ค่าความยาวสูงสุดของรหัสตัวเลข Huffman ในข้อมูลอินพุท
1	0	แสดงแฟล็กเริ่มต้นค้นหาตัวอักษร
5	31	ค้นหาไม่พบตัวอักษรตั้งแต่ค่าแอสกี 0-32
1	0	แสดงแฟล็กค้นหาตัวอักษรต่อไป
5	31	ค้นหาไม่พบตัวอักษรอีกขณะนี้ค้นหาถึงค่าแอสกี 64
1	0	แสดงแฟล็กค้นหาตัวอักษรต่อไป
5	31	ค้นหาไม่พบตัวอักษรอีกขณะนี้ค้นหาถึงค่าแอสกี 96
1	1	แสดงแฟล็กพบตัวอักษรถัดไปคือค่าแอสกี 97
5	4	แสดงจำนวนความยาวตัวอักษรที่พบติดกัน ตั้งแต่ค่าแอสกี 97-100
3	3	ค่าความยาวรหัสตัวเลขของแอสกี 97 นำไปหักลบกับค่าความยาวสูงสุด
3	1	ค่าความยาวรหัสตัวเลขของแอสกี 98 นำไปหักลบกับค่าความยาวสูงสุด
3	0	ค่าความยาวรหัสตัวเลขของแอสกี 99 นำไปหักลบกับค่าความยาวสูงสุด
3	0	ค่าความยาวรหัสตัวเลขของแอสกี 100 นำไปหักลบกับค่าความยาวสูงสุด
1	0	แสดงแฟล็กค้นหาตัวอักษรต่อไป
5	12	แสดงตำแหน่งที่ค้นพบตัวอักษรถัดไป คือค่าแอสกี 114
1	1	แสดงแฟล็กพบตัวอักษร
5	0	แสดงการสิ้นสุดการค้นหาตัวอักษร เมื่อตัวเลขที่เก็บเป็น 0
3	2	ค่าความยาวรหัสตัวเลขของแอสกี 114 นำไปหักลบกับค่าความยาวสูงสุด
1	0	เลขจำนวนเต็มของรหัสตัวเลข "0" หรือตัวอักษร 'a'
3	6	เลขจำนวนเต็มของรหัสตัวเลข "110" หรือตัวอักษร 'b'
2	2	เลขจำนวนเต็มของรหัสตัวเลข "10" หรือตัวอักษร 'r'
1	0	เลขจำนวนเต็มของรหัสตัวเลข "0" หรือตัวอักษร 'a'
4	14	เลขจำนวนเต็มของรหัสตัวเลข "1110" หรือตัวอักษร 'c'
1	0	เลขจำนวนเต็มของรหัสตัวเลข "0" หรือตัวอักษร 'a'
4	15	เลขจำนวนเต็มของรหัสตัวเลข "1111" หรือตัวอักษร 'd'
1	0	เลขจำนวนเต็มของรหัสตัวเลข "0" หรือตัวอักษร 'a'
3	6	เลขจำนวนเต็มของรหัสตัวเลข "110" หรือตัวอักษร 'b'
2	2	เลขจำนวนเต็มของรหัสตัวเลข "10" หรือตัวอักษร 'r'
2	2	เลขจำนวนเต็มของรหัสตัวเลข "10" หรือตัวอักษร 'r'

*<u>หมายเหตุ</u> ตัวเลขที่เก็บ ประกอบด้วย เลขฐานสิบของรหัสแอสกี ค่าแฟล็ก และค่าความยาว

วิธีการขยายขนาดข้อมูล (Decompression) จะทำการย้อนกลับ (backward) วิธีการของการบีบอัดข้อมูล (Compression) โดยเริ่มต้นจากขบวนการของ Static Huffman coding และ Order-3 context modeling ตามลำดับ มีหลักการคือ เริ่มต้นด้วยการอ่านค่าข้อมูลส่วนหัว (Header) ของแฟ้มข้อมูลบีบอัด เพื่อใช้กำหนดขอบเขตสิ้นสุดการทำงานของโปรแกรมขยายขนาดข้อมูล จากนั้นอ่านค่าบิตขึ้นมาโดยพิจารณาตามความยาวที่เก็บตามโครงสร้างของตารางเข้ารหัสซึ่งมีรูปแบบดังแสดงในตารางที่ 4.2 เพื่อกำหนดหาความสัมพันธ์ระหว่างรหัสตัวเลขและค่าความยาวของรหัสตัวเลขของตัวอักขระแต่ละตัว แล้วเก็บความสัมพันธ์เป็นพจนานุกรมไว้ ให้อ่านรหัสตัวเลขจากแฟ้มข้อมูลบีบอัดอีกครั้งหนึ่ง แล้วนำมาเปรียบเทียบกับรหัสตัวเลขในพจนานุกรม เพื่อนำกลับตัวอักขระเดิมออกมา เช่น รหัสตัวเลข “0” นำกลับมาเป็นตัวอักขระ “a” เป็นต้น หลังจากผ่านขบวนการขยายขนาดข้อมูลของ Static Huffman coding แล้ว เราจะได้เอ้าท์พุทคือ ข้อมูลที่ไม่แมทช์กัน และค่าความยาวของข้อมูลที่แมทช์กัน จากนั้นนำข้อมูลที่ไม่แมทช์กันส่งผ่านไปยังขบวนการของ Context Modeling โดยอ่านข้อมูลครั้งละ 3 ไบต์ ตามรูปแบบของ Order-3 Context ใช้ตารางแฮชทำการเปรียบเทียบข้อมูล ถ้าข้อมูลสามารถแมทช์กันได้ให้นำค่าความยาวที่แมทช์กันได้ (Match-length) มาขยายขนาดข้อมูลออกไปตามความยาวที่ได้รับ ถ้าข้อมูลไม่แมทช์กัน ถือว่าข้อมูลนั้นเป็นข้อมูลต้นฉบับ แล้วกระทำซ้ำจนกระทั่งนำกลับข้อมูลต้นฉบับออกมาทั้งหมด

4.4 การทดสอบอัลกอริทึมการบีบอัดข้อมูล

ในการทดสอบได้ทำการเปรียบเทียบระหว่างอัลกอริทึมการบีบอัดข้อมูลของ LZP และ LZW ซึ่งอัลกอริทึมทั้งสองเป็นอัลกอริทึมที่มีพื้นฐานมาจาก Lempel-Ziv และถูกพัฒนาให้ใช้งานบนโครงข่ายสื่อสารเช่นเดียวกัน การเลือกค่าตารางเปรียบเทียบข้อมูลมีขนาด 15 บิต หรือใช้หน่วยความจำขนาด 64 กิโลไบต์ สำหรับ LZP ใช้ Order-3 context ที่มีโครงสร้างข้อมูลแบบแฮชชิ่ง แต่ LZW ใช้โครงสร้างข้อมูลแบบอาร์เรย์ ข้อมูลที่ใช้ทดสอบเป็นไฟล์ต้นฉบับจาก Calgory Corpus [13] ซึ่งเป็นไฟล์มาตรฐานประเภทตัวอักษร (Text compression) สำหรับทดสอบการบีบอัดข้อมูลแบบต่าง ๆ เพื่อใช้ในการเปรียบเทียบประสิทธิภาพกับอัลกอริทึมทั้งสองแบบ ได้แสดงผลการทดสอบในตารางที่ 4.3 และ 4.4 นอกจากนั้นได้มีการทดสอบกับไฟล์ข้อมูลประเภทรูปภาพ (Image) ที่มีรูปแบบ (Format) ต่าง ๆ กันด้วย โดยข้อมูลที่ใช้ทดสอบเป็นดังรูปที่ 4.4 เป็นภาพสี และเก็บอยู่ในรูปแบบของ Bitmap โดยใช้โปรแกรม Lview Pro ทำการแปลงรูปแบบ (Format) ของไฟล์ให้เป็นชนิดต่าง ๆ กัน แสดงผลการทดสอบในตารางที่ 4.5 ซึ่งการทดสอบนี้แสดงให้เห็นถึงการเปรียบเทียบขนาดของไฟล์หลังผ่านการบีบอัดข้อมูล ความเร็วที่ได้รับจากการบีบอัดข้อมูลในรูปแบบของไบต์ต่อวินาที และแสดงถึงอัตราการบีบอัดข้อมูล (Compression ratio) ในรูปแบบของจำนวนบิตต่อ

ไบต์ เราสามารถหาค่าอัตราการบีบอัดข้อมูลได้จากสมการที่ 4.3 นอกจากนี้ในตารางที่ 4.6 ยังแสดงให้เห็นถึงการเลือกค่าลำดับของ Context modeling ค่าต่าง ๆ กัน ของ LZP เพื่อให้เห็นผลลัพธ์ของหน่วยความจำที่ใช้ ความเร็วในการประมวลผลและอัตราการบีบอัดข้อมูลที่ได้รับ ซึ่งในการทดสอบใช้ไฟล์ข้อมูลของ Book2 ที่มีขนาด 610,856 ไบต์ โปรแกรมนั้นทดสอบบนเครื่องคอมพิวเตอร์ Intel Pentium ความเร็ว 150 MHz รันบนแพลทฟอร์ม Windows 95 ใช้คอมไพล์เลอร์ Borland C++Builder 3.0

$$\text{อัตราการบีบอัดข้อมูล (บิตต่อไบต์)} = \frac{\text{ขนาดข้อมูลหลังจากถูกบีบอัด} * 8}{\text{ขนาดข้อมูลก่อนการบีบอัด}} \quad \ldots\ldots (4.3)$$

ตารางที่ 4.3 ผลการทดสอบของ LZP กับไฟล์ต้นฉบับจาก Calgory Corpus

ไฟล์ต้นฉบับ	ขนาดของไฟล์ (ไบต์)	ขนาดของไฟล์หลังผ่านการบีบอัดข้อมูล (ไบต์)	ความเร็วที่ได้รับจากการบีบอัดข้อมูล (กิโลไบต์ต่อวินาที)	อัตราการบีบอัดข้อมูล (บิตต่อไบต์)
Bib	111,261	39,863	398	2.87
Book1	768,771	341,756	342	3.56
Book2	610,856	231,748	333	3.03
Geo	102,400	71,133	125	5.56
News	377,109	148,207	296	3.14
Obj1	21,504	11,932	146	4.44
Obj2	246,814	105,563	247	3.42
Paper1	53,161	21,694	226	3.26
Paper2	82,199	33,756	229	3.29
Pic	513,216	57,998	704	0.90
Progc	39,611	16,037	248	3.24
Progl	71,646	18,881	326	2.11
Progp	49,379	13,192	299	2.14
Trans	93,695	21,925	317	1.87
		ค่าเฉลี่ย	303	3.06

ตารางที่ 4.4 ผลการทดสอบของ LZW กับไฟล์ต้นฉบับจาก Calgory Corpus

ไฟล์ต้นฉบับ	ขนาดของไฟล์ (ไบต์)	ขนาดของไฟล์หลังผ่านการบีบอัดข้อมูล (ไบต์)	ความเร็วที่ได้รับจากการบีบอัดข้อมูล (กิโลไบต์ต่อวินาที)	อัตราการบีบอัดข้อมูล (บิตต่อไบต์)
Bib	111,261	46,538	224	3.35
Book1	768,771	350,719	307	3.65
Book2	610,856	266,679	307	3.49
Geo	102,400	78,512	196	6.13
News	377,109	193,991	296	4.12
Obj1	21,504	14,056	138	5.23
Obj2	246,814	131,156	302	4.25
Paper1	53,161	25,084	307	3.77
Paper2	82,199	36,171	308	3.52
Pic	513,216	62,671	560	0.98
Progc	39,611	19,150	317	3.87
Progl	71,646	27,158	330	3.03
Progp	49,379	19,217	355	2.57
Trans	93,695	38,250	341	3.27
		ค่าเฉลี่ย	306	3.67

จากผลการทดสอบในตารางที่ 4.3 และ 4.4 เมื่อทดสอบกับไฟล์มาตรฐานของ Calgary Corpus จะเห็นว่าอัลกอริทึม LZP จะได้อัตราการบีบอัดข้อมูลโดยเฉลี่ยที่ 3.06 บิตต่อไบต์ ซึ่งมีอัตราการบีบอัดข้อมูลดีกว่า LZW โดยอยู่ที่ 3.67 บิตต่อไบต์ ส่วนทางด้านความเร็วที่ได้รับจากการบีบอัดข้อมูลของ LZP จะเทียบเท่ากับ LZW ถ้าเรานำอัลกอริทึม LZP ไปใช้บนโครงข่ายสื่อสารแล้ว จะทำให้ได้ทั้งความเร็วในการประมวลผล และได้ปริมาณการส่งข้อมูล (Throughput) มากขึ้น โดยจะขอกล่าวประสิทธิภาพของอัลกอริทึมการบีบอัดข้อมูลเมื่อนำไปใช้บนโครงข่ายเวิลด์ดาต้าในบทที่ 6

รูปที่ 4.4 แสดงลักษณะรูปภาพ (Image) ที่ใช้ในการทดสอบ

ตารางที่ 4.5 ผลการทดสอบของ LZP กับไฟล์ประเภทรูปภาพ (Image) ชนิดต่าง ๆ

ประเภทไฟล์รูปภาพ	ขนาดของไฟล์ (ไบต์)	ขนาดของไฟล์หลังผ่านการบีบอัดข้อมูล (ไบต์)	ความเร็วที่ได้รับจากการบีบอัดข้อมูล (กิโลไบต์ต่อวินาที)	อัตราการบีบอัดข้อมูล (บิตต่อไบต์)
Bmp	644,354	127,869	935	1.59
Jpg	22,391	20,806	153	7.43
Tif	646,714	129,906	919	1.61
Pcx	330,453	121,341	441	2.94
Gif	95,240	96,594	165	8.11
Tga	643,560	127,265	975	1.58
		ค่าเฉลี่ย	598	3.88

จากตารางที่ 4.5 เมื่อให้ไฟล์รูปภาพต้นฉบับเป็นแบบ BMP การบีบอัดข้อมูลด้วย LZP จะได้ขนาดของไฟล์ใหญ่กว่า JPG และเมื่อใช้ LZP บีบอัดไฟล์รูปภาพประเภท JPG และ GIF อีก เราจะได้อัตราการบีบอัดข้อมูลไม่ดี คือ JPG ขนาดเล็กลงเล็กน้อย และ GIF มีขนาดเพิ่มขึ้นอีกเล็กน้อย ทั้งนี้เนื่องจากไฟล์รูปภาพทั้งสองชนิดนี้มีโครงสร้างการเก็บรายละเอียดของภาพที่มีการลดขนาดอยู่แล้ว แต่สำหรับไฟล์รูปภาพประเภท BMP, TIF, PCX และ TGA จะได้รับอัตราการบีบอัดที่ดี และมีความเร็วในการบีบอัดข้อมูลสูง ถึงอย่างไรการเก็บภาพแบบ JPG และ GIF จะมีขนาดของไฟล์เล็กกว่ามาก ทำให้ไฟล์รูปภาพประเภท JPG และ GIF นิยมนำมาใช้งานในทางธุรกิจเช่น ระบบอินเตอร์เน็ต เป็นต้น

ตารางที่ 4.6 ผลการเปรียบเทียบประสิทธิภาพเมื่อเลือกค่าลำดับต่าง ๆ กัน ของ Context modeling

Order-i Context modeling	ขนาดหน่วยความจำที่ใช้ (กิโลไบต์)	ความเร็วที่ได้รับจากการบีบอัดข้อมูล (กิโลไบต์ต่อวินาที)	อัตราการบีบอัดข้อมูล (บิตต่อไบต์)
Order-1	1	257	3.96
Order-2	8	333	3.38
Order-3	32	368	3.06
Order-4	2,000	366	3.00
Order-5	8,000	352	3.07

จากตารางที่ 4.6 ทำการทดสอบบนไฟล์ข้อมูล Book2 จะเห็นว่าการเปลี่ยนค่าลำดับ (Order) ของ Context modeling จะมีผลต่อความเร็วในการประมวลผลและอัตราการบีบอัดข้อมูล การเลือกค่าลำดับต่ำจะมีข้อดีคือใช้หน่วยความจำน้อยลง ผลเสียจะทำให้ได้รับอัตราการบีบอัดข้อมูลน้อยลง แต่ถ้าเราให้ค่าลำดับยิ่งสูงขึ้นก็ยิ่งใช้หน่วยความจำมากขึ้น ผลที่ได้รับจะมีอัตราการบีบอัดข้อมูลมากขึ้น ซึ่งเมื่อพิจารณาที่ Order-3, Order- 4 หรือ Order-5 แล้ว อัตราการบีบอัดข้อมูลที่ได้รับจะใกล้เคียงกันมาก

สำหรับ Order-3 ถ้าดูอัตราการบีบอัดข้อมูลที่ได้รับจากตารางที่ 4.3 แล้วจะเท่ากับ 3.03 บิตต่อไบต์ จากการที่เลือกขนาดหน่วยความจำเท่ากับ 64 กิโลไบต์ แต่จากตารางที่ 4.6 นี้เลือกขนาดหน่วยความจำเท่ากับ 32 กิโลไบต์ ทำให้ได้รับอัตราการบีบอัดข้อมูลเท่ากับ 3.06 บิตต่อไบต์ มีผลต่างกัน 0.03 บิตต่อไบต์ ที่เป็นเช่นนี้เนื่องจากว่าการเพิ่มขนาดตารางเปรียบเทียบข้อมูลจะช่วยลดการเกิดการชนกันของข้อมูลได้นั่นเอง แต่ช่วยเพิ่มอัตราการบีบอัดได้ไม่มากนัก และถ้าเพิ่มหน่วยความจำเป็น 128 กิโลไบต์ขึ้นไป จะไม่มีผลทำให้ได้รับการบีบอัดข้อมูลดีขึ้น ดังนั้นจากการเลือก Order-3 แบบคงที่สำหรับอัลกอริทึมของ LZP จะเหมาะสมที่สุดกับ 3 องค์ประกอบหลักคือ ด้านหน่วยความจำที่ใช้ ความเร็วในการประมวลผลและอัตราการบีบอัดข้อมูล

การเลือกค่าของลำดับ (Order) ที่ใช้กับ LZP นี้เป็นแบบคงที่ แนวทางผู้ที่จะปรับปรุงต่อไปเป็นไปได้ที่จะให้มีการเลือกค่าลำดับหลาย ๆ ค่าอย่างอัตโนมัติ เพื่อให้แต่ละค่าลำดับเหมาะกับลักษณะข้อมูลที่นำมาทำการบีบอัด ในการที่จะค้นหาข้อมูลที่แมทช์กันดีที่สุด การทำเช่นนี้จะทำให้ได้รับอัตราการบีบอัดข้อมูลดีขึ้น

บทที่ 5

การเข้ารหัสลับและถอดรหัสลับข้อมูลแบบผสม RC5 และ RSA

ระบบการเข้ารหัสลับ (Cryptosystem) คือ การนำเอกสารปกติ (Plaintext) มาเข้ารหัสลับ (Encrypt) จะได้เอกสารที่ถูกแปลงไปจากเดิมเรียกว่า เอกสารรหัสลับ (Ciphertext) เมื่อเราต้องการอ่านเอกสารเราต้องมีกุญแจรหัสลับ (Key) สำหรับถอดรหัสลับ (Decrypt) ให้เป็นเอกสารปกติเพื่อให้สามารถอ่านเอกสารนั้นได้ โดยที่ระบบการเข้ารหัสลับจะต้องครอบคลุมคุณสมบัติที่สำคัญ 4 ข้อ คือ

1). ความเป็นส่วนตัว (Privacy) ข้อมูลส่วนตัวต้องไม่ถูกบุคคลอื่นเข้าถึงได้

2). ความน่าเชื่อถือ (Integrity) ข้อมูลที่ไปถึงผู้รับต้องไม่ถูกเปลี่ยนแปลงระหว่างทางโดยผู้ไม่มีหน้าที่ และข้อมูลต้องไม่ถูกทำลายทำให้เกิดการสูญเสียจากผู้บุกรุกได้

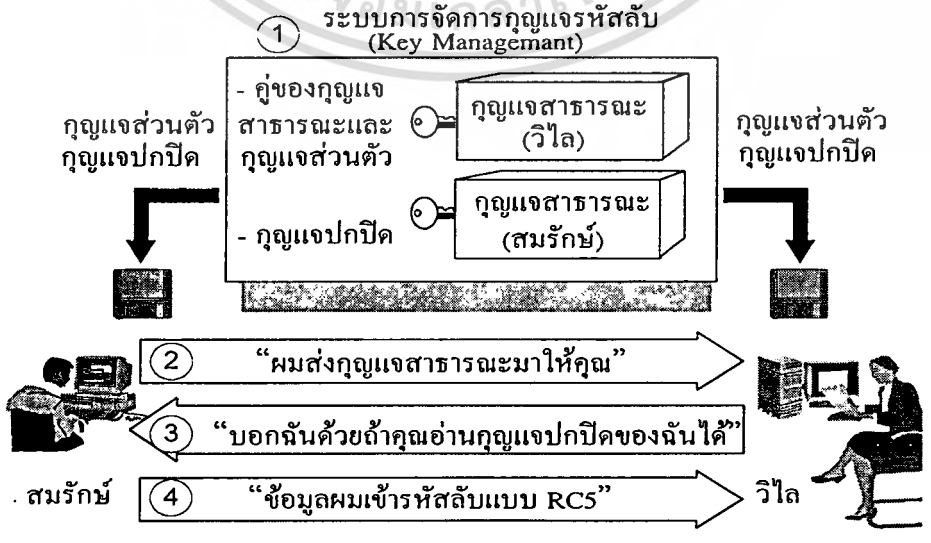
3). การพิสูจน์ความเป็นเจ้าของ (Authentication) ข้อมูลที่ผู้รับได้รับนั้น ต้องทราบว่ามาจากใครและพิสูจน์ได้ว่าผู้นั้นเป็นคนส่งจริง

4). ปฏิเสธไม่ได้ (Non-repudiation) ผู้ส่งไม่สามารถปฏิเสธได้ว่าไม่ได้ส่งข้อมูลมา

กระบวนการเข้ารหัสลับมีอยู่ 2 วิธีการด้วยกันคือ การเข้ารหัสลับชนิดซิมเมทริกซ์ (Symmetric cryptosystem) และการเข้ารหัสลับชนิดอะซิมเมทริกซ์ (Asymmetric cryptosystem) ทั้ง 2 วิธีมีความแตกต่างกัน คือ การเข้ารหัสลับชนิดซิมเมทริกซ์จะใช้กุญแจรหัสลับชนิดเดียวกันในการเข้ารหัสลับและถอดรหัสลับ กุญแจรหัสลับชนิดเดียวกันนี้เราเรียกว่ากุญแจปกปิด (Secret key) ในการรับส่งข้อมูลต้องมีการแลกเปลี่ยนกุญแจปกปิดระหว่างกันก่อน ซึ่งอาจถูกดักฟังกุญแจปกปิดจากบุคคลอื่นที่อยู่บนโครงข่ายสื่อสารเดียวกันได้ แต่วิธีการนี้สามารถให้ความเร็วในการเข้ารหัสลับและถอดรหัสลับสูงกว่าแบบอะซิมเมทริกซ์ ตัวอย่างอัลกอริทึมที่ใช้ในการเข้ารหัสลับแบบนี้ เช่น DES (Data Encryption Standard), Skipjack, RC2, RC4 และ RC5 (Rivest Cipher) เป็นต้น สำหรับการเข้ารหัสลับชนิดอะซิมเมทริกซ์ แต่ละคนจะมีกุญแจรหัสลับ 2 อัน คือกุญแจสาธารณะ(Public key) และกุญแจส่วนตัว (Private key) โดยที่กุญแจสาธารณะสามารถเปิดเผยต่อบุคคลใดก็ได้ ในขณะที่กุญแจส่วนตัวจะถูกปกปิดเป็นความลับ ดังนั้นการแลกเปลี่ยนข้อมูลข่าวสารระหว่างกัน ในวิธีนี้จะมีเฉพาะกุญแจสาธารณะเท่านั้นอยู่ในโครงข่ายสื่อสาร ตัวอย่างอัลกอริทึมของวิธีนี้ เช่น Diffie-Hellman, DSS (Digital Signature Standard), และ RSA (Rivest-Shamir-Adleman) เป็นต้น

ในบทนี้ได้นำข้อดีของการเข้ารหัสลับแบบ RC5 และ RSA มาผสมผสานกัน คือข้อดีของ RC5 จะให้ความเร็วในการประมวลผล เนื่องจากใช้วิธีการหมุนบิตและสลับบิต นอกจากนี้การประมวลผลสามารถกำหนดได้ตามสถาปัตยกรรมของเครื่องคอมพิวเตอร์ เช่น 16 บิต 32 บิต และ 64 บิต เป็นต้น ซึ่งมีผลต่อประสิทธิภาพในการประมวลผล ทั้งนี้ระดับความปลอดภัยยังสามารถกำหนดได้จากความยาวของกุญแจปกปิดและวงรอบของการคำนวณ และได้นำข้อดีของ RSA ซึ่งจะให้ความปลอดภัยสูง มาใช้ในขั้นตอนการแลกเปลี่ยนกุญแจปกปิดของ RC5 เพื่อป้องกันการดักฟังกุญแจปกปิดจากผู้อื่น เนื่องจากการทำลายกุญแจปกปิดจากอัลกอริทึมของ RSA จะขึ้นอยู่กับการแยกตัวประกอบ (Factoring) เป็นสำคัญ ถ้าให้ความยาวกุญแจสาธารณะมีค่ามากแล้ว ทำให้วิธีการแยกตัวประกอบ เพื่อหาค่ากุญแจส่วนตัวทำได้ไม่ง่ายนัก โดยที่กุญแจส่วนตัวเป็นกุญแจสำคัญในการเปิดอ่านกุญแจปกปิดของ RC5 ที่ถูกปกปิดเอาไว้ เมื่อสมาชิกอีกฝ่ายหนึ่งเปิดอ่านกุญแจปกปิดได้แล้ว การรับส่งข้อมูลระหว่างกันจะใช้ระบบการเข้ารหัสลับแบบ RC5 นอกจากนี้วิธีการสร้างกุญแจรหัสลับนั้นก็นับว่าสำคัญอย่างยิ่ง เนื่องจากเป็นสิ่งกำหนดระดับความปลอดภัยของระบบการเข้ารหัสลับ และป้องกันไม่ให้ถูกโจมตีกุญแจรหัสลับได้ กุญแจรหัสลับในที่นี้คือ กุญแจสาธารณะ กุญแจส่วนตัว และกุญแจปกปิด จะต้องถูกสร้างจากแหล่งกำเนิดตัวเลขสุ่มที่คาดเดาได้ยาก ซึ่งจะขอกล่าวถึงรายละเอียดในหัวข้อต่อไป ทั้งนี้ การสร้าง การเก็บ และการแลกเปลี่ยนกุญแจรหัสลับ เรามักรวมกันเรียกว่า ระบบการจัดการกุญแจรหัสลับ สามารถเขียนเป็นไดอะแกรมการใช้งานแบบผสม RC5 และ RSA ดังรูปที่ 5.1

รูปที่ 5.1 ไดอะแกรมการใช้งานแบบผสม RC5 และ RSA



จากรูปที่ 5.1 แสดงถึงหลักการใช้งานระบบการเข้ารหัสลับแบบผสม RC5 และ RSA และสามารถแสดงขั้นตอนการใช้งานได้ดังนี้

ขั้นตอนที่ 1 เริ่มแรกทั้งสมรักษ์และวิไลจะต้องสร้างกุญแจรหัสลับคือกุญแจปกปิด กุญแจส่วนตัว และกุญแจสาธารณะขึ้นมาก่อน จากนั้นทำการจัดเก็บกุญแจปกปิด และกุญแจส่วนตัวไว้เป็นความลับที่สุด ส่วนกุญแจสาธารณะสามารถเปิดเผยได้

ขั้นตอนที่ 2 เมื่อสมรักษ์ต้องการติดต่อกับวิไล สมรักษ์จะส่งกุญแจสาธารณะของตนเองไปให้วิไล เพื่อขอให้วิไลส่งกุญแจปกปิดกลับมาหาตน

ขั้นตอนที่ 3 ทางฝ่ายวิไลจะใช้กุญแจสาธารณะของสมรักษ์ทำการเข้ารหัสลับกุญแจปกปิดของตนเองด้วยอัลกอริทึมของ RSA แล้วส่งกุญแจปกปิดของวิไลที่ถูกเข้ารหัสลับเรียบร้อยแล้วกลับไปให้สมรักษ์

ขั้นตอนที่ 4 ทางฝ่ายสมรักษ์จะใช้กุญแจส่วนตัวของตนเองเปิดอ่านกุญแจปกปิดของวิไลออกมา แล้วใช้กุญแจปกปิดของวิไลนี้ เข้ารหัสลับข้อมูลแบบ RC5 ส่งข้อมูลที่เข้ารหัสลับแล้วไปให้วิไล เมื่อวิไลรับข้อมูลที่ถูกเข้ารหัสลับ (Ciphertext) จากสมรักษ์ วิไลจะใช้กุญแจปกปิดของตนเองเปิดอ่านข้อมูลออกมา

5.1 อัลกอริทึมการเข้ารหัสลับแบบ RC5 (RC5 Encryption Algorithm)

RC5(Rivest Cipher) เป็นการเข้ารหัสลับอย่างเร็วแบบบล็อคชนิดซิมเมทริกซ์ (Fast Symmetric block cipher) ที่เหมาะกับการใช้งานทั้งซอฟต์แวร์และฮาร์ดแวร์ มีความยืดหยุ่นในการกำหนดระดับความปลอดภัยและความเร็วของแต่ละโปรแกรมใช้งาน เราสามารถกำหนดรูปแบบมาตรฐานขึ้นมาคือ RC5-w/r/b โดยที่พารามิเตอร์แต่ละตัวสามารถอธิบายได้ดังนี้

w คือ ขนาดข้อมูลที่เก็บได้ (word size) มีหน่วยเป็นบิต สามารถมีค่าได้ตามสถาปัตยกรรมของเครื่องคอมพิวเตอร์ เช่น 16 บิต 32 บิต และ 64 บิต เป็นต้น ในการเข้ารหัสจะใช้บล็อคขนาด 2 w ยกตัวอย่างเช่น ถ้าเราใช้ w เท่ากับ 32 บิต จะได้ขนาดบล็อค 2x32 = 64 บิต หรือ 8 ตัวอักษร เป็นต้น

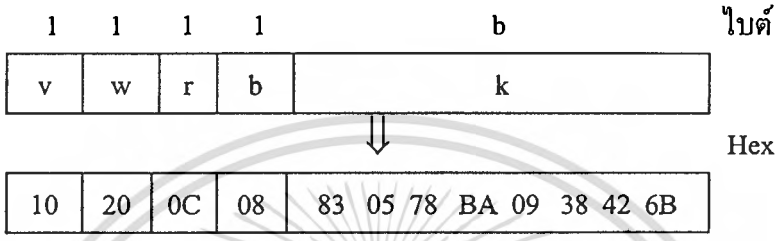
r คือ จำนวนวงรอบ (round) ของการเลือกระดับความปลอดภัย มีค่าตั้งแต่ 0 ถึง 255

b คือ ขนาดของกุญแจปกปิด (secret key) มีหน่วยเป็นไบต์ มีค่าได้ตั้งแต่ 0 ถึง 255 ไบต์

นอกจากนี้ยังมีพารามิเตอร์ที่เกี่ยวข้องอีกคือ S เป็นตารางขยายกุญแจปกปิด (Expanded key table) ที่ได้รับมาจากกุญแจปกปิด (k) โดยขนาดของตาราง (t) ขึ้นอยู่กับจำนวนวงรอบ (r) ที่ใช้ในการคำนวณ คือ t = 2(r+1) มีหน่วยเป็น words

เวลาใช้งานการกำหนดพารามิเตอร์เหล่านี้จะถูกรวมไว้ด้วยกันเป็นบล็อคควบคุม (Control block) ประกอบด้วยฟิลด์ต่างๆ ดังในรูปที่ 5.2

รูปที่ 5.2 แสดงฟิลด์ต่างๆ ในบล็อคควบคุมของ RC5



จากรูปที่ 5.2 จะได้ว่า RC5 เวอร์ชั่น (v) 1.0 มีขนาด (w) เท่ากับ 32 วงรอบ (r) เท่ากับ 12 รอบ มีขนาดของกุญแจปกปิด (b) เท่ากับ 8 ไบต์ หรือ 64 บิต และมีกุญแจปกปิด (k) เริ่มจาก 83 05 ... 6B ซึ่งการจัดการกุญแจรหัสลับ (Key management) จะกระทำในบล็อคควบคุม (Control block) นี้ และถูกจัดเก็บไว้เป็นไฟล์ข้อมูลอย่างปกปิด สำหรับส่งไปยังบุคคลที่ต้องการติดต่อระหว่างกันในโครงข่ายสื่อสาร

RC5 ประกอบด้วยส่วนสำคัญ 3 ส่วนด้วยกัน คือ ส่วนขยายกุญแจปกปิด (Key expansion) ส่วนการเข้ารหัสลับ (Encryption) และส่วนการถอดรหัสลับ (Decryption)

5.1.1 ส่วนขยายกุญแจปกปิด (Key expansion)

เป็นอัลกอริทึมที่ขยายกุญแจปกปิด k ที่อยู่ในบล็อคควบคุมให้ยาวขึ้นในระหว่างการทำอัลกอริทึมการเข้าและถอดรหัสลับข้อมูล แล้วเก็บลงในอาร์เรย์ S ตามขนาดของ $t = 2(r + 1)$ การทำเช่นนี้ก็เพื่อเพิ่มระดับความปลอดภัยให้สูงขึ้น คือยิ่งให้ค่าวงรอบ (r) สูง จะทำให้ระดับความปลอดภัยสูงขึ้นไปด้วย โดยเราให้กุญแจปกปิดที่ถูกขยายแล้วเป็นตัวเลขไบนารีแบบสุ่ม (Random binary word) และกำหนดค่าเริ่มต้นของ Pw และ Qw เป็นแบบ Magic Contant ดังนี้

$Pw = Odd((e-2)2^w)$

$Qw = Odd((\phi-1)2^w)$

เมื่อ $e = 2.718281828459$ (Natural logarithms)

$\phi = 1.618033988749$ (Golden ratio)

ในขณะที่ Odd(x) เป็น เลขจำนวนเต็มคี่ (Odd integer) และมีค่า w เป็นค่าคงที่ คือ 16, 32 หรือ 64 เป็นต้น สามารถเขียน Pw และ Qw เป็นเลขฐานสิบหกได้คือ

P16 = 0 x b7e1

Q16 = 0 x 9e37

P32 = 0 x b7e15163

Q32 = 0 x 9e3779b9

P64 = 0 x b7e151628aed296b

Q64 = 0 x 9e3779b97f4a7c15

การทำงานของส่วนขยายกุญแจปกปิดแบ่งเป็น 3 ขั้นตอน คือ

ขั้นตอนที่ 1 ทำการแปลงกุญแจปกปิดจากไบต์ไปเป็นเวิร์ด (Byte to word) เพื่อการคำนวณทางคณิตศาสตร์ กระทำโดยคัดลอกกุญแจปกปิดจากอาร์เรย์ k[0,...,b-1] ลงไปในอาร์เรย์ L[0,...,c-1] โดยขนาดของอาร์เรย์ L เป็นไปตามค่า c = [b/u]; เมื่อ u = w/8 สามารถเขียนอัลกอริทึมได้ดังนี้

```
for i = b-1 downto 0 do
  L[i/u] = (L[i/u] <<< 8) + k[i];
```

ขั้นตอนที่ 2 กำหนดค่าเริ่มต้นในอาร์เรย์ S โดยกำหนดค่าแบบสุ่ม จากค่า Pw และ Qw สามารถเขียนอัลกอริทึมได้ดังนี้

```
S[0] = Pw;
for i = 1 to t-1 do
   S[i] = S[i-1] + Qw;
```

ขั้นตอนที่ 3 การรวมกันระหว่างค่าในอาร์เรย์ของกุญแจปกปิด (k) ที่อยู่ในรูปแบบเวิร์ด (L) และกุญแจปกปิดที่ถูกขยาย (S) โดยใช้วิธีการคำนวณซ้ำๆ (Iteration) ตั้งแต่ 3 ครั้งขึ้นไป เนื่องจากความแตกต่างด้านขนาดของอาร์เรย์ S และ L พร้อมทั้งอาร์เรย์มีขนาดใหญ่ ทำให้มีการวนรอบเริ่มต้นจาก 3 ครั้งขึ้นไป สามารถเขียนอัลกอริทึมได้ดังนี้

```
i = j = 0;
A = B = 0;
  do    3 * max(t,c) times:
        A = S[i] = (S[i] + A + B) <<< 3;
        B = L[j] = (L[j] + A + B) <<< (A+B);
        i = (i + 1) mod (t);
        j = (j + 1) mod (c);
```

เมื่อขั้นตอนที่ 3 สิ้นสุดลง เราจะได้กุญแจปกปิดที่ขยายแล้วอยู่ในอาร์เรย์ของ S เพื่อนำไปใช้สำหรับการเข้าและถอดรหัสลับแบบ RC5 ซึ่งฟังก์ชั่นการขยายกุญแจปกปิดนี้เป็นแบบทิศทางเดียว (one-way) ทำให้ไม่ง่ายที่จะนำกลับค่ากุญแจปกปิด (k) จากกุญแจปกปิดที่ขยายแล้ว

5.1.2 การเข้ารหัสลับ (Encryption)

เราให้ข้อมูลที่เข้ารหัสลับมีขนาด 2w บิต แทนด้วยตัวแปร A และ B ก่อนที่จะถึงขั้นตอนนี้จะผ่านวิธีการขยายส่วนของกุญแจปกปิดในหัวข้อ 5.1.1 ก่อน เพื่อให้ได้อาร์เรย์ S[0,...,t-1] ของกุญแจปกปิดที่ถูกขยายมาแล้ว เมื่อผ่านขบวนการเข้ารหัสลับแล้วผลลัพธ์ที่ได้จะถูกเก็บลงในตัวแปร A และ B ดังเดิม เราสามารถแสดงอัลกอริทึมได้ดังนี้

```
A = A + S[0];
B = B + S[1];
for i = 1 to r do
        A = ((A ⊕ B) <<< B) + S [2*i];
        B = ((B ⊕ A) <<< A) + S [2*i+1];
```

เมื่อ ⊕ คือ Bitwise XOR

<<< คือ การหมุนบิตไปทางซ้าย (Left-rotation)

5.1.3 การถอดรหัสลับ (Decryption)

อัลกอริทึมของการถอดรหัสลับจะเป็นส่วนกลับของอัลกอริทึมการเข้ารหัสลับ สามารถแสดงอัลกอริทึมได้ดังนี้

for i = r downto 1 do

B = ((B - S[2*i+1]) >>> A) $\oplus$ A;

A = ((A - S[2*i]) >>> B) $\oplus$ B;

B = B - S[1];

A = A - S[0];

เมื่อ $\oplus$ คือ Bitwise XOR

>>> คือการหมุนบิตไปทางขวา (Right-rotation)

ตัวอย่างการแสดงผลลัพธ์จากอัลกอริทึม RC5 มีการกำหนดรูปแบบเป็น RC5-32/12/16 (RC5-w/r/b) คือ ขนาดข้อมูลที่เก็บได้ (w) เท่ากับ 32 บิต, จำนวนวงรอบ (r) เท่ากับ 12 รอบ และขนาดกุญแจปกปิด (b) เท่ากับ 16 ไบต์ หรือ 128 บิต

1). กุญแจปกปิด = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
เอกสารปกติ = 00 00 00 00 00 00 00 00
เอกสารรหัสลับ = EE DB A5 21 6D 8F 4B 15

2). กุญแจปกปิด = 91 5F 46 19 BE 41 B2 51 63 55 A5 01 10 A9 CE 91
เอกสารปกติ = EE DB A5 21 6D 8F 4B 15
เอกสารรหัสลับ = AC 13 C0 F7 52 89 2B 5B

5.2 อัลกอริทึมการเข้ารหัสลับแบบ RSA (RSA Algorithm)

การเข้ารหัสลับแบบ RSA ใช้หลักการของโมดูลัส (Modulus) โดยเลือกค่าจำนวนเฉพาะ (Prime number) ที่มีค่ามาก คือ p และ q โดยที่ q > p เมื่อให้ n = pq และเลือกค่า e ให้น้อยกว่าค่า n โดยค่า e มีความสัมพันธ์กับค่า p และ q ซึ่งเมื่อหาค่า ห.ร.ม. (Greatest common divisors หรือ gcd) ระหว่างค่า e และ (p-1)(q-1) แล้วจะเหลือเศษ 1 ซึ่งเราสามารถใช้อัลกอริทึมของ Euclidean [17] พิสูจน์ค่า gcd ได้ดังนี้

gcd (e, (p-1) (q-1)) = 1 ..... (5.1)

<u>ตัวอย่างที่ 1</u> ถ้าเราให้ค่า e เท่ากับ 13, p เท่ากับ 43 และ q เท่ากับ 59 จะได้ n = 43.59 เท่ากับ 2537 เมื่อพิสูจน์ gcd (13, (43-1) . (59-1)) = 1 แล้วจะได้

gcd (13, 2436)	=	1
2436	=	187 . (13) + 5
13	=	2 . (5) + 3
5	=	1 . (3) + 2
3	=	1 . (2) + 1
2	=	1 . (1) + 1
1	=	1 . (1)

5.2.1 การเข้ารหัสลับ (Encryption)

ข้อมูลที่จะเข้ารหัสลับจะแปลงให้อยู่ในรูปของตัวเลข เพื่อสะดวกในการคำนวณ อาจจะเป็นข้อมูลไบต์เดียว หรือจะรวมกันเป็นบล็อคก็ได้ เราจะให้ M เป็นตัวเลขจำนวนเต็ม (Integer) ของเอกสารปกติ (Plaintext) และให้ C เป็นตัวเลขจำนวนเต็มของเอกสารรหัสลับ (Ciphertext) สามารถเขียนเป็นฟังก์ชันการเข้ารหัสลับ RSA ดังนี้

$$C = M^c \bmod n \qquad \text{..... (5.2)}$$

เมื่อ (n, e) คือ กุญแจสาธารณะ (Public key)

ในกรณีที่ค่า M และ e มีค่ามาก ไม่สามารถคำนวณด้วยฟังก์ชั่นคณิตศาสตร์เลขยกกำลังได้ เราต้องแปลงสมการที่ 5.2 ให้อยู่ในรูปของอินเนอร์เวิร์คกิ้ง (Inner workings) ดังเช่นตัวอย่างในสมการที่ 5.3 คือ

$$C = M^{5c} \bmod n = ((M^{2c} \bmod n)*(M^{3c} \bmod n)) \bmod n \qquad \text{.....(5.3)}$$

<u>ตัวอย่างที่ 2</u> จากตัวอย่างที่ 1 ถ้าเราให้ค่า M เท่ากับ 65 หรือเท่ากับตัวอักษร ‘A’ เมื่อแทนค่าให้อยู่ในรูปของอินเนอร์เวิร์คกิ้ง จะได้

$C = M^e \bmod n = 65^{13} \bmod 2537$

$= ((65^3 \bmod 2537)*(65^5 \bmod 2537)*(65^5 \bmod 2537)) \bmod 2537$

$= (629*1286*1286) \bmod 2537 = 1040237684 \bmod 2537 = 1722$

เราสามารถหาค่า C จากอัลกอริทึมของ Chinese remainder theorem [17]

5.2.2 การถอดรหัสลับ (Decryption)

เอกสารปกติจะถูกถอดรหัสลับได้โดยใช้กุญแจส่วนตัว d ค่า d เป็นส่วนกลับของ e mod (p-1)(q-1) สามารถหาค่าได้จากส่วนกลับของอัลกอริทึม Euclidean[17] และเราสามารถเขียนความสัมพันธ์ระหว่างค่า d และ e ได้ดังนี้

$de \bmod (p-1)(q-1) = 1$ ..... (5.4)

<u>ตัวอย่างที่ 3</u> จากตัวอย่างที่ 1 เราสามารถหาค่า d จากส่วนกลับของอัลกอริทึม Euclidean ได้ดังนี้

เราให้ x = x2 – y * x1 ; เมื่อ y = a/b , r = a – y * b

1 = 1 – 0 . (0) ; ให้ค่าเริ่มต้น a = 13, b = 2436, x = 1, x1 = 0, x2 = 1

-187 = 0 – 187 . (1) ; ให้ a = b, b = r, x2 = x1, x1 = x ในแต่ละรอบการคำนวณ

375 = 1 – 2 . (-187)

-562 = -187 – 1 . (375)

937 = 375 – 1 . (-562)

-2436 = -562 – 2 . (937) ; หยุดการคำนวณเมื่อ b = 0 และให้ d = x1

ดังนั้น d มีค่าเท่ากับ 937 เมื่อแทนค่าในสมการที่ 5.4 แล้วจะได้

(937) . (13) mod 2436 = 1

เราสามารถถอดรหัสลับเอกสารได้จากสมการที่ 5.5 และใช้อัลกอริทึมของ Chinese remainder theorem [17] หาค่าของ P เช่นเดียวกับวิธีการเข้ารหัสลับ

$P = C^d \bmod n$ ..... (5.5)

<u>ตัวอย่างที่ 4</u> จากตัวอย่างที่ 1, 2 และ 3 เราสามารถหาค่า P ได้คือ

$$P = 1722^{937} \bmod 2537$$
$$= 65$$

เมื่อเราได้ค่า (n, e) ซึ่งเป็นกุญแจสาธารณะ และ d เป็นกุญแจส่วนตัวแล้ว ค่า Prime number p และ q จะต้องเก็บเป็นความลับ เพราะถ้าคนอื่นรู้ค่า p และ q จะทำให้หาค่า d ได้

5.3 ระบบการจัดการกุญแจรหัสลับ (Key management)

ระบบการจัดการกุญแจรหัสลับประกอบด้วย การสร้าง การแลกเปลี่ยน การเก็บ การรักษา และการสำรอง เป็นต้น ทั้งหมดนี้การทำให้มีความปลอดภัยนั้นเป็นสิ่งที่ยากยิ่ง และมักจะตกเป็นเป้าหมายการถูกจู่โจมเพื่อทำลายอยู่เสมอ กุญแจรหัสลับอาจถูกสร้างขึ้นจากหน่วยงานที่รับผิดชอบการออกกุญแจรหัสลับ หรือสร้างขึ้นมาใช้เองก็ได้ ต่อจากนั้นทำการแจกจ่ายหรือแลกเปลี่ยนให้กับสมาชิกโดยใส่ลงแผ่นฟล็อปปี้ดิสก์ หรือส่งไปในช่องการสื่อสารที่คิดว่าปลอดภัย แต่ในความเป็นจริงแล้วการส่งกุญแจรหัสลับบนโครงข่ายข้อมูลมักจะถูกดักฟังได้ง่าย สำหรับกุญแจส่วนตัว (Private key) ของการเข้ารหัสลับแบบ RSA และกุญแจปกปิด (Secret key) ของการเข้ารหัสแบบ RC5 ไม่สมควรอย่างยิ่งที่จะส่งไปในโครงข่ายสื่อสารให้กับสมาชิก โดยไม่มีการปกปิดใดๆ เมื่อสมาชิกแต่ละคนได้รับกุญแจรหัสลับไปแล้ว ในทางปฏิบัติสำหรับกุญแจปกปิดมักจะแลกเปลี่ยนกับสมาชิกกับคนอื่นๆ ในโครงข่ายสื่อสารอีก

5.3.1 การสร้างกุญแจรหัสลับ

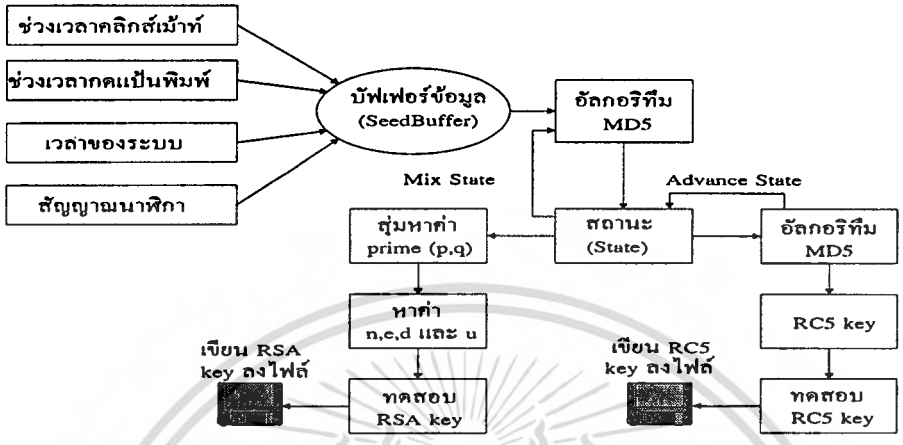
เราสามารถสร้างกุญแจรหัสลับของ RC5 และ RSA ได้โดยเลือกแหล่งกำเนิดตัวเลขสุ่ม (Random number source) ที่เหมาะสมตามระบบปฏิบัติการ (Operating system) ที่ใช้ การเลือกระบบใดระบบหนึ่งหรือหลายระบบประกอบกันต้องอยู่บนพื้นฐานคือ การเกิดของสัญญาณต้องไม่แน่นอน และบิตข้อมูลที่เกิดขึ้นต้องไม่มีความสัมพันธ์ซึ่งกันและกัน อันจะมีผลทำให้ยากต่อการสุ่มหาค่ากุญแจรหัสลับ เราสามารถแบ่งแหล่งกำเนิดตัวเลขสุ่มออกเป็น 2 แบบ คือตัวเลขสุ่มจริง (True random number) และตัวเลขสุ่มเทียม (Pseudo random number) ในตารางที่ 5.1 แสดงให้เห็นแหล่งกำเนิดตัวเลขสุ่มจริงลักษณะต่าง ๆ กัน แหล่งกำเนิดตัวเลขสุ่มจริงที่ใช้ในบทนี้คือ ช่วงเวลาการคลิกส์เม้าท์ ช่วงเวลาการกดแป้นพิมพ์ เวลาของระบบและสัญญาณนาฬิกา นำมาเรียงลำดับกันลงบนบัฟเฟอร์ข้อมูล (Seed buffer) เพื่อใช้เป็นจุดเริ่มต้นของขบวนการกำเนิดตัวเลขสุ่มเทียม

โดยนำมาผ่านกลไกฟังก์ชั่นไดเจสของเอกสาร (Message Digest หรือ MD5) ทำซ้ำ ๆ ไปมาหลาย ๆ รอบตามสัญญาณที่เกิดขึ้นจากแหล่งกำเนิดตัวเลขสุ่มจริง เพื่อทำให้ความสัมพันธ์ของกลุ่มบิตที่เกิดขึ้นยุ่งยากและซับซ้อนในเชิงสถิติ และนำผลลัพธ์จากตัวเลขสุ่มเทียมสร้างเป็นกุญแจรหัสลับ ประกอบด้วย กุญแจปกปิด กุญแจส่วนตัว และกุญแจสาธารณะ ในรูปที่ 5.3 แสดงไดอะแกรมการสร้างกุญแจรหัสลับ RC5 และ RSA

ตารางที่ 5.1 แสดงแหล่งกำเนิดตัวเลขสุ่มจริง (True random number) ลักษณะต่างๆ กัน

คุณลักษณะเฉพาะของระบบ (System unique)	แหล่งกำเนิดตัวเลขสุ่มทั่วไป	ตัวเลขสุ่มภายนอก (External random)
1. คอนฟิกกูเรชั่นไฟล์ 2. ไดร์ฟคอนฟิกกูเรชั่น 3. เนื้อความแสดงสภาวะแวดล้อม	1. รายละเอียดข้อมูลจอภาพ 2. วันที่และเวลา 3. สัญญาณนาฬิกาความละเอียดสูง 4. แป้นพิมพ์ที่กดคีย์สุดท้าย 5. ขนาดของ log file 6. ข้อมูลการใช้ระบบโครงข่าย 7. ตัวเลขแสดงกระบวนการทำงานของโปรแกรม 8. โปรแกรมนับจำนวน 9. กระบวนการแสดงการทำงานของโปรแกรม	1. ตำแหน่งเคอร์เซอร์กับเวลา 2. ช่วงเวลาการกดแป้นพิมพ์ 3. สัญญาณอินพุตไมโครโฟนกับสัญญาณไมโครโฟนที่ต่ออยู่ 4. ช่วงเวลาการคลิกส์เม้าท์ 5. ตำแหน่งการเคลื่อนเม้าท์ 6. ตัวเลขแสดงการใช้หน่วยความจำ 7. สัญญาณอินพุตวิดีโอ

รูปที่ 5.3 แสดงไดอะแกรมการสร้างกุญแจรหัสลับ RC5 และ RSA



จากรูปที่ 5.3 สามารถแสดงขั้นตอนการทำงานต่างๆ ดังนี้

<u>ขั้นตอนที่ 1</u> เลือกความยาวรหัสลับตามขนาดที่ต้องการ เช่น กุญแจปกปิดของ RC5 ขนาด 256 บิต และกุญแจส่วนตัวกับกุญแจสาธารณะของ RSA ขนาด 512 บิต เป็นต้น

<u>ขั้นตอนที่ 2</u> เลือกบัฟเฟอร์ข้อมูล (Seed Buffer) ขนาด 512 บิต หรือ 64 ไบต์ สำหรับเก็บกลุ่มบิตข้อมูลที่รับมาจากแหล่งกำเนิดตัวเลขสุ่มจริงประกอบด้วย ช่วงเวลาการคลิกส์เม้าท์ ช่วงเวลาการกดแป้นพิมพ์ เวลาของระบบและสัญญาณนาฬิกา เราสามารถประกอบกันเข้าอยู่ในบัฟเฟอร์ข้อมูลดังนี้

```
SeedBuffer[0] = MouseClick_Timing & 0xFF;
SeedBuffer[1] = MouseClick_Timing >> 8;
SeedBuffer[2] = KeyStoke_Timing & 0xFF;
SeedBuffer[3] = KeyStoke_Timing >> 8;
SeedBuffer[4] = time() & 0xFF;
SeedBuffer[5] = time() >> 8;
SeedBuffer[6] = clock() & 0xFF;
SeedBuffer[7] = clock() >> 8;
```

<u>ขั้นตอนที่ 3</u> นำบัฟเฟอร์ข้อมูล (Seed Buffer) ที่ได้รับจากแหล่งกำเนิดตัวเลขสุ่มจริงในขั้นตอนที่ 2 มากระทำอัลกอริทึมของ MD5 ซึ่งเป็นขบวนการสร้างตัวเลขสุ่มเทียม โดยสามารถเข้า

รหัสลับบล็อคข้อมูลที่มีขนาดความยาวแตกต่างกันได้ แต่จะให้ข้อมูลเอ้าท์พุตที่ไม่เหมือนกันในการสุ่มแต่ละครั้ง และมีขนาดคงที่เท่ากับ 128 บิต โดยใช้วิธีการสร้างสถานะ (State) ซ้ำ ๆ กันหลายรอบจากฟังก์ชั่นที่ไม่เป็นเชิงเส้น (Nonlinear function) ต่อจากนั้นนำสถานะที่ได้มากระทำซ้ำกับบัฟเฟอร์ข้อมูล (Seed buffer) ของการสุ่มรอบต่อไป จนกระทั่งจำนวนรอบเท่ากับจำนวนนับตัวเลขสุ่มที่กำหนดไว้ สุดท้ายเราจะได้สถานะที่มีลักษณะบิตเรียงตัวเป็นอิสระต่อกันและยากต่อการคาดเดา เราสามารถแสดงอัลกอริทึมของ MD5 ได้ดังนี้

กำหนดค่าเริ่มต้นของตัวแปร A, B, C และ D ให้เป็นค่า Magic constant คือ

A = 0x67452301

B = 0xefcdab89

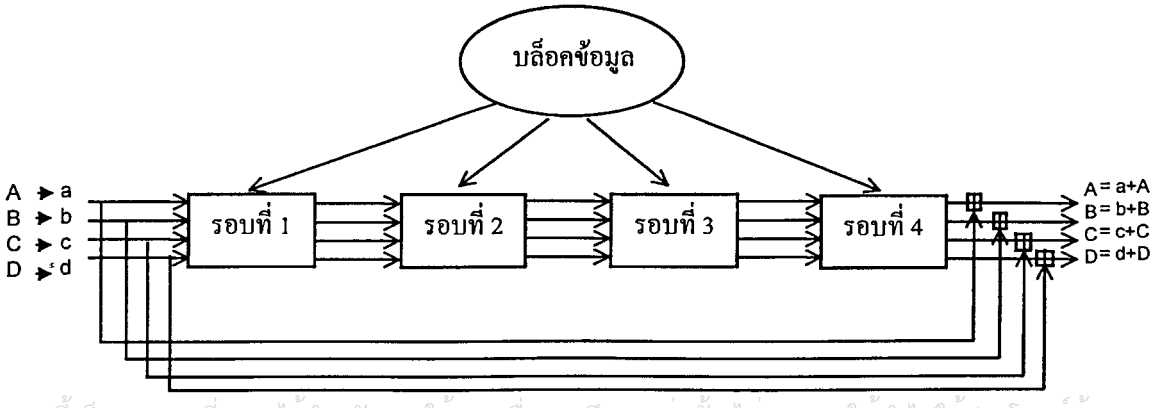
C = 0x98badcfe

D = 0x10325476

ลูปหลักของอัลกอริทึม MD5 แสดงให้เห็นในรูปที่ 5.4 ลูปนี้กระทำต่อเนื่องกับหลายๆ บล็อคข้อมูลขนาด 512 บิต จากบัฟเฟอร์ข้อมูล (Seed Buffer) การทำงานของลูปหลักมีการวนรอบ 4 รอบด้วยกัน ในแต่ละรอบการทำงานมีการกระทำตามฟังก์ชั่นที่ไม่เป็นเชิงเส้น (Nonlinear function) 16 ครั้ง โดยให้แต่ละฟังก์ชั่นมีลักษณะแตกต่างกันออกไป

เริ่มต้นการทำงานค่าเริ่มต้นของตัวแปร a, b, c และ d จะคัดลอกมาจากตัวแปร A, B, C และ D เพื่อใช้เป็นสถานะเริ่มต้นการทำงาน เมื่อผ่านฟังก์ชั่นที่ไม่เป็นเชิงเส้นครบ 4 รอบแล้ว สุดท้ายเราจะนำผลลัพธ์ของตัวแปร a, b, c และ d บวกกันทางเรขาคณิตเข้ากับตัวแปร A, B, C และ D ตามลำดับ และเก็บผลลัพธ์ที่บวกกันได้แทนค่าในตัวแปร A, B, C และ D เช่นเดิม เมื่อมีบล็อคข้อมูลใหม่เข้ามา ข้อมูลในตัวแปร A, B, C และ D จะถูกกำหนดเป็นสถานะเริ่มต้นการทำงานครั้งต่อไป

รูปที่ 5.4 แสดงการทำงานลูปหลักของ MD5



ฟังก์ชั่นที่ไม่เป็นเชิงเส้นในแต่ละรอบการทำงาน มีลักษณะแตกต่างกันดังนี้

F(X,Y,Z) = XY or not (X) Z

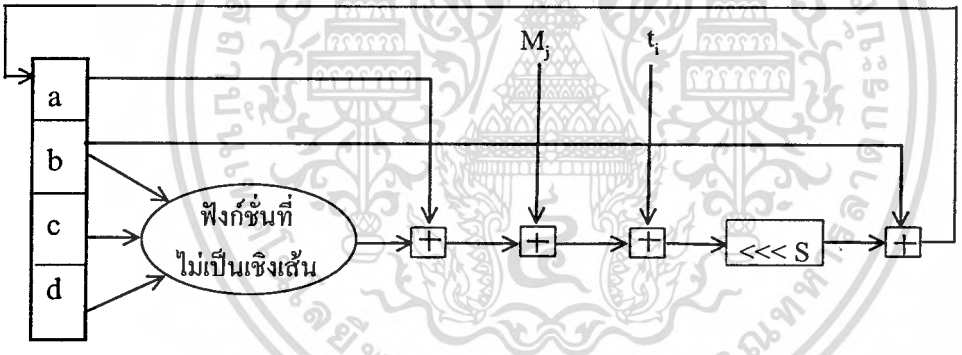
G(X,Y,Z) = XZ or Y not (Z)

H(X,Y,Z) = X xor Y xor Z

I(X,Y,Z) = Y xor (X or not (Z))

ฟังก์ชั่นเหล่านี้เรากำหนดขึ้นมา เพื่อให้กลุ่มบิตของ X, Y และ Z มีลักษณะที่เป็นอิสระต่อกัน และแต่ละบิตไม่มีความสัมพันธ์ซึ่งกันและกัน ในรูปที่ 5.5 แสดงไดอะแกรมการทำงานในหนึ่งฟังก์ชั่นของ MD5

รูปที่ 5.5 แสดงไดอะแกรมการทำงานในหนึ่งฟังก์ชั่นของ MD5



จากรูปที่ 5.5 เรากำหนดให้

ค่า M_j แสดงถึงบล็อคข้อมูลย่อยจากตัวเลขสุ่มจริงเริ่มจากไบต์ 0 ถึง 15 ในแต่ละรอบการทำงานจะใช้การสุ่มวางตำแหน่งของไบต์สลับกัน

ค่า t_i เป็นตัวเลขจำนวนเต็มของ 2^{32} * abs (Sin(i)) เมื่อ i คือ ค่ารัศมี (Radians) แล้วแปลงเป็นเลขฐานสิบหก

ค่า <<<S แสดงถึงการเลื่อนตำแหน่งไปทางซ้ายจำนวน S บิต ตัวเลขที่ได้ใช้วิธีการสุ่มวาง

เราสามารถแสดงฟังก์ชั่นที่ไม่เป็นเชิงเส้นใน 4 รอบการทำงานดังนี้

$a = b+((a + F(b,c,d) + M_j + t_i) <<<S)$ หรือเขียนเป็น $FF\ (a,b,c,d,M_j,S,t_i)$

$a = b+((a + G(b,c,d) + M_j + t_i) <<<S)$ หรือเขียนเป็น $GG\ (a,b,c,d,M_j,S,t_i)$

$a = b+((a + H(b,c,d) + M_j + t_i) <<<S)$ หรือเขียนเป็น $HH (a,b,c,d,M_j,S,t_i)$

$a = b+((a + I(b,c,d) + M_j + t_i) <<<S)$ หรือเขียนเป็น $II (a,b,c,d,M_j,S,t_i)$

แต่ละบล็อคข้อมูลย่อยถูกกระทำตามฟังก์ชั่นที่ไม่เป็นเชิงเส้นจำนวน 4 รอบ รอบละ 16 ขั้นตอน รวมเป็น 64 ขั้นตอนด้วยกัน คือ

<u>รอบที่ 1</u>:

FF (a, b, c, d, M_0, 7, 0xd76aa478)

FF (d, a, b, c, M_1, 12, 0xe8c7b756)

FF (c, d, a, b, M_2, 17, 0x242070db)

FF (b, c, d, a, M_3, 22, 0xc1bdceee)

FF (a, b, c, d, M_4, 7, 0xf57c0faf)

FF (d, a, b, c, M_5, 12, 0x4787c62a)

FF (c, d, a, b, M_6, 17, 0xa8304613)

FF (b, c, d, a, M_7, 22, 0xfd469501)

FF (a, b, c, d, M_8, 7, 0x698098d8)

FF (d, a, b, c, M_9, 12, 0x8b44f7af)

FF (c, d, a, b, M_{10}, 17, 0xffff5bb1)

FF (b, c, d, a, M_{11}, 22, 0x895cd7be)

FF (a, b, c, d, M_{12}, 7, 0x6b901122)

FF (d, a, b, c, M_{13}, 12, 0xfd987193)

FF (c, d, a, b, M_{14}, 17, 0xa679438e)

FF (b, c, d, a, M_{15}, 22, 0x49b40821)

<u>รอบที่ 2</u>:

GG (a, b, c, d, M_1, 5, 0xf61e2562)

GG (d, a, b, c, M_6, 9, 0xc040b340)

GG (c, d, a, b, M_{11}, 14, 0x256e5a51)

GG (b, c, d, a, M_0, 20, 0xe9b6c7aa)

GG (a, b, c, d, M_5, 5, 0xd62f105d)

GG (d, a, b, c, M_{10}, 9, 0x02441453)

GG (c, d, a, b, M_{15}, 14, 0xd8a1e681)

GG (b, c, d, a, M_4, 20, 0xe7d3fbc8)

GG (a, b, c, d, M_9, 5, 0x21e1cde6)

GG (d, a, b, c, M_{14}, 9, 0xc33707d6)

GG (c, d, a, b, M_3, 14, 0xf4d50d87)

GG (b, c, d, a, M_8, 20, 0x455a14ed)

GG (a, b, c, d, M_{13}, 5, 0xa9e3e905)

GG (d, a, b, c, M_2, 9, 0xfcefa3f8)

GG (c, d, a, b, M_7, 14, 0x676f02d9)

GG (b, c, d, a, M_{12}, 20, 0x8d2a4c8a)

<u>รอบที่ 3</u>:

HH (a, b, c, d, M_5, 4, 0xfffa3942)

HH (d, a, b, c, M_8, 11, 0x8771f681)

HH (c, d, a, b, M_{11}, 16, 0x6d9d6122)

HH (b, c, d, a, M_{14}, 23, 0xfde5380c)

HH (a, b, c, d, M_1, 4, 0xa4beea44)

HH (d, a, b, c, M_4, 11, 0x4bdecfa9)

HH (c, d, a, b, M_7, 16, 0xf6bb4b60)

HH (b, c, d, a, M_{10}, 23, 0xbebfbc70)

HH (a, b, c, d, M_{13}, 4, 0x289b7ec6)

HH (d, a, b, c, M_0, 11, 0xeaa127fa)

HH (c, d, a, b, M_3, 16, 0xd4ef3085)

HH (b, c, d, a, M_6, 23, 0x04881d05)

HH (a, b, c, d, M_9, 4, 0xd9d4d039)

HH (d, a, b, c, M_{12}, 11, 0xe6db99e5)

HH (c, d, a, b, M_{15}, 16, 0x1fa27cf8)

HH (b, c, d, a, M_2, 23, 0xc4ac5665c)

<u>รอบที่ 4</u>:

II (a, b, c, d, M_0, 6, 0xf4292244)

II (d, a, b, c, M_7, 10, 0x432aff97)

II (c, d, a, b, M_{14}, 15, 0xab9423a7)

II (b, c, d, a, M_5, 21, 0xfc93a039)

II (a, b, c, d, M_{12}, 6, 0x655b59c3)

II (d, a, b, c, M_3, 10, 0x8f0ccc92)

II (c, d, a, b, M_{10}, 15, 0xffeff47d)

II (b, c, d, a, M_1, 21, 0x85845dd1)

II (a, b, c, d, M_8, 6, 0x6fa87e4f)

II (d, a, b, c, M_{15}, 10, 0xfe2ce6e0)

II (c, d, a, b, M_6, 15, 0xa3014314)

II (b, c, d, a, M_{13}, 21, 0x4e0811a1)

II (a, b, c, d, M_4, 6, 0xf7567e82)

II (d, a, b, c, M_{11}, 10, 0xbd3af235)

II (c, d, a, b, M_2, 15, 0x2ad7d2bb)

II (b, c, d, a, M_9, 21, 0xeb86d391)

หลังจากบล็อคข้อมูลผ่านขั้นตอนทั้งหมดแล้ว ตัวแปร a, b, c และ d จะบวกเข้ากับตัวแปร A, B, C และ D ตามลำดับ และนำผลลัพธ์เก็บไว้ในตัวแปร A, B, C และ D ตามเดิม สุดท้ายเราจะได้ข้อมูลจากตัวแปร A, B, C และ D นำมาเรียงประกอบกันได้ขนาดเท่ากับ 128 บิต จากนั้นนำข้อมูลมาทำเอ็กซ์คลูซีฟออร์ (Exclusive-OR) กับบัฟเฟอร์รวมตัวเลขสุ่ม (Random pool) ที่มีขนาด 256 ไบต์ และเก็บผลลัพธ์ไว้ที่บัฟเฟอร์รวมตัวเลขสุ่มเช่นเดิม เพื่อใช้เป็นข้อมูลอินพุตในขั้นตอนที่ 4 และขั้นตอนที่ 9 จากนั้นนำบล็อคข้อมูลถัดไปมากระทำซ้ำอัลกอริทึม MD5 จนสิ้นสุดบล็อคข้อมูลที่ต้องการ

<u>ขั้นตอนที่ 4</u> การสร้างกุญแจสาธารณะและกุญแจส่วนตัวของ RSA เริ่มต้นโดยการสุ่มหาค่า Prime p จากบิตที่มีนัยสำคัญเท่ากับความยาวกุญแจรหัสลับ เนื่องจากค่า Prime p นี้มีการคำนวณแบบโมดูลัสที่ขึ้นอยู่กับความยาว ดังนั้นเริ่มต้นเราตั้งค่า 2 บิตบน (MSB) ของ Prime p มีค่าเป็น "1" เพื่อให้แน่ใจว่าค่า Prime p มีความยาวของกุญแจรหัสลับตามที่กำหนดไว้ สามารถแสดงอัลกอริทึมได้ดังนี้

1). นำบัฟเฟอร์รวมตัวเลขสุ่ม (Random pool) ที่ได้รับจากขั้นตอนที่ 3 มาสุ่มค่าให้แตกต่างไปจากเดิม(Random stir) ด้วยอัลกอริทึม MD5 อีกครั้งหนึ่ง แล้วเก็บเอ้าท์พุทที่ได้เป็น Prime p

2). ตรวจสอบค่า p เปรียบเทียบกับค่าในตาราง Primetable ซึ่งค่า p ต้องมีค่ามากกว่าค่าใน ตาราง Primetable จุดประสงค์ในการสร้างตาราง Primetable ไว้ก่อนจะช่วยให้การค้นหาค่า Prime ทำได้เร็วขึ้น โดยกำหนดให้ตัวเลขใน Primetable เป็นเลขคี่ และมีขนาดเท่ากับ 16 บิต แสดงได้ดังนี้

static word16 primetable[] = { /* กำหนดค่า prime เป็นค่าคงที่ และใช้ค่า "0" เมื่อสิ้นสุดตาราง */

3, 5, 7, 11, 13, 17, 19,23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89,97, 101, 103, 107, 109, 113, 127, 131,137, 139, 149, 151, 157, 163, 167, 173,179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, ..., 8167, 8171, 8179, 8191, 0 };

3). ตั้งค่า 2 บิตล่าง (LSB) ของค่า p เป็น "1" เพื่อทำให้เป็นเลขคี่ (Odd) เนื่องจากค่า p ต้องเป็นเลขคี่เท่านั้น

4). สร้างตารางเลือกเฟ้น (Buildsieve) สำหรับค่า p ที่คาดว่าจะเป็นค่า Prime ให้เป็นแบบลำดับ (Sequential) เพื่อช่วยให้การค้นหาค่า Prime ทำได้เร็วขึ้น โดยจัดเก็บเศษเหลือไว้ในบัฟเฟอร์ชั่วคราว (Remainders) ก่อน แสดงวิธีการได้ดังนี้

```
while (primetable[i]  !=  0)
        remainders[i] = p mod primetable[i];
        i++;
endloop
```

5). เลือกเฟ้นค่า p ที่คาดว่าจะเป็นค่า Prime โดยใช้ขบวนการหาค่าที่เร็ว (Fastseive) ถ้าพบค่า p ที่คาดว่าจะเป็นค่า Prime ให้ทดสอบค่า p ตามข้อ 6. แสดงวิธีการได้ดังนี้

```
poffset = 0;  /* ให้ค่าเริ่มต้น offset ของ p เป็น 0 */
while (primetable[i] != 0)
        if (( (poffset + remainders[i]) mod primetable[i] ) == 0 )
                then p + poffset is not prime
```

```
                continue /* ทำลูป while ต่อ */
        else   /* คาดว่าค่า p จะเป็น prime */
                test  p follow item 6.
                if p is not prime
                        then continue
                else /* พบค่า p เป็น prime */
                        goto endloop
        poffset = poffset + 2  /* ลองเพิ่มค่า offset ของ p ขึ้นไป 2 แล้วพยายามหาดูใหม่ */
endloop
```

6). ทดสอบค่า p ที่คาดว่าจะเป็น Prime โดยใช้ทฤษฎีของ Fermat คือ

```
if ((x**(p-1)) mod p ) ! = 1
    then p is not prime
```

โดยที่ ค่า x เป็นค่าที่นำมาจากตาราง Primetable[i] ; 0<i<4 เพื่อใช้ในการทดสอบ

** คือยกกำลัง

<u>ขั้นตอนที่ 5</u> สุ่มหาค่า Prime q โดยใช้วิธีการเดียวกันกับขั้นตอนที่ 4 หลังจากนั้นตรวจสอบว่า q > p หรือไม่ ให้สุ่มหาจนกระทั่งพบค่า q เป็นค่า Prime

<u>ขั้นตอนที่ 6</u> คำนวนหาค่า n, e , d และ u หลังจากได้ค่า Prime p และ q มาแล้ว มีวิธีการคือ

1). ตรวจสอบเพื่อให้แน่ใจว่า q > p

2). คำนวณหาค่า phi(n) โดยใช้ฟังก์ชั่น Euler totient (phi(n)) คือ

phi(n) = (p-1)*(q-1)

3). คำนวณหาค่า G(n) คือ

G(n) = gcd(p-1, q-1)

4). คำนวณหาค่า F(n) คือ

F(n) = phi (n) / G(n)

5). เลือกค่า e ที่มีค่าน้อยกว่า n และมีความสัมพันธ์กับค่า Prime (p-1)(q-1) โดยทดสอบด้วยการหาค่า gcd ระหว่างค่า e และ (p-1)(q-1) แล้วจะเหลือเศษ 1 ในที่นี้เลือกค่า e เริ่มต้นเท่ากับ 11 ถ้าไม่สามารถคำนวณได้จะเพิ่มค่าเป็น 13, 15 และ 17 ตามลำดับ

6). คำนวณหาค่า d จากส่วนกลับของ e, mod F(n) คือ

(e*d) mod F(n) = 1

7). คำนวณหาค่า u จากส่วนกลับของ p, mod q คือ

(p*u) mod q = 1

8). คำนวณหาค่า n คือ

n = p*q

ขั้นตอนที่ 7 ทดสอบกุญแจรหัสลับของ RSA ที่สร้างขึ้นมากับฟังก์ชั่นการเข้าและถอดรหัสลับของ RSA ด้วยข้อความสั้น ๆ ทั้งนี้ก็เพื่อให้แน่ใจว่าสามารถใช้งานได้ถูกต้อง

ขั้นตอนที่ 8 เขียนแฟ้มข้อมูลกุญแจส่วนตัวและกุญแจสาธารณะของ RSA ลงไฟล์ โดยเก็บแยกกันคนละไฟล์ มีรูปแบบดังนี้

Type	Bits	Date	n	e	d	p	q	u
Pub	1024	yyyy-mm-dd	xxxxxxx	xxxxxxx	NULL	NULL	NULL	NULL
Pri	1024	yyyy-mm-dd	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx

ขั้นตอนที่ 9 การสร้างกุญแจปกปิดของ RC5 ทำได้โดยการนำข้อมูลที่เก็บไว้ในบัฟเฟอร์รวมตัวเลขสุ่ม (Random pool) จากขั้นตอนที่ 3 มาผ่านอัลกอริทึม MD5 อีกครั้งหนึ่ง เพื่อที่จะเพิ่มความซับซ้อนของสถานะ (Advance state) ขึ้นไปอีก ผลลัพธ์ที่ได้จะเก็บไว้ในบัฟเฟอร์ข้อมูลใหม่ที่เรียกว่า Advance random pool มีขนาดเท่ากับ 256 ไบต์ โดยข้อมูลในบัฟเฟอร์ข้อมูลใหม่นี้จะมีการผสมกันใหม่ทุกครั้งเมื่อมีบล็อคข้อมูลตัวเลขสุ่มเข้ามา เมื่อสิ้นสุดบล็อคข้อมูลตัวเลขสุ่ม เราจะได้บัฟเฟอร์ข้อมูลใหม่นี้เป็นกุญแจปกปิด RC5 ตามขนาดที่เลือกไว้ในขั้นตอนที่ 1

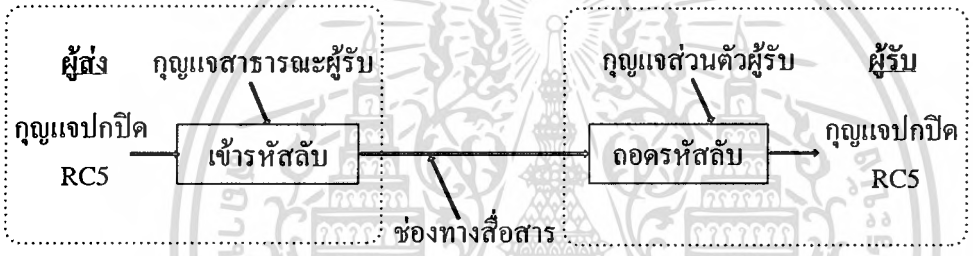
ขั้นตอนที่ 10 เขียนแฟ้มข้อมูลกุญแจปกปิดของ RC5 ลงไฟล์ มีรูปแบบดังนี้

version	word	round	bytes	key
10	20	0C	08	83 05 78 BA 09 38 42 6B

5.3.2 การแลกเปลี่ยนกุญแจรหัสลับ

หลักการเบื้องต้นของการแลกเปลี่ยนกุญแจปกปิด RC5 ในโครงข่ายสื่อสาร คือเมื่อผู้ส่งร้องขอการติดต่อไปยังผู้รับ ผู้รับจะจัดการส่งกุญแจสาธารณะกลับมาให้ ผู้ส่งจะนำข้อมูลที่เป็นกุญแจปกปิดมาเข้ารหัสลับด้วยอัลกอริทึม RSA ด้วยกุญแจสาธารณะที่ผู้รับส่งกลับมา ต่อจากนั้นกุญแจปกปิดที่เข้ารหัสลับไว้จะถูกส่งไปยังผู้รับ ผู้รับจะใช้กุญแจส่วนตัวของตนเองเปิดอ่านกุญแจปกปิดออกมา หลังจากนั้นผู้รับก็จะใช้กุญแจปกปิดเข้ารหัสลับข้อมูลที่รับส่งระหว่างกันต่อไป ดังรูปที่ 5.6 แสดงการส่งกุญแจปกปิดในโครงข่ายสื่อสาร

รูปที่ 5.6 แสดงการส่งกุญแจปกปิดในโครงข่ายสื่อสาร



สามารถแสดงขั้นตอนการแลกเปลี่ยนกุญแจรหัสลับระหว่างผู้รับและผู้ส่ง ดังนี้

ขั้นตอนที่ 1 เริ่มต้นโดยผู้ส่งร้องขอการติดต่อไปยังผู้รับด้วยเฟรมข้อมูลที่ประกอบไปด้วยหมายเลขเซสชั่น (Session ID) เวอร์ชั่นของอัลกอริทึมการแลกเปลี่ยนกุญแจรหัสลับ และระบบกุญแจปกปิดที่ใช้ ตามลำดับ

ขั้นตอนที่ 2 ผู้รับจะทำการตรวจสอบเฟรมข้อมูลผู้ส่งก่อนว่าคุณสมบัติตรงกันหรือไม่ ถ้าตรงกันจะตอบกลับด้วยเฟรมข้อมูลเดียวกันกับผู้ส่ง พร้อมกับส่งกุญแจสาธารณะของผู้รับกลับไปด้วย และเก็บหมายเลขเซสชั่นของผู้ส่งไว้ใช้อ้างอิง สำหรับการส่งกลับเฟรมข้อมูลในครั้งต่อไป แต่ถ้าตรวจสอบเฟรมข้อมูลแล้วไม่ตรงกันจะยกเลิกการติดต่อทันที เพื่อป้องกันผู้แอบอ้างและดักฟังเอากุญแจปกปิดไป

ขั้นตอนที่ 3 เมื่อผู้ส่งได้รับเฟรมข้อมูลยืนยันการติดต่อและเฟรมข้อมูลกุญแจสาธารณะตอบกลับจากผู้รับ จะนำกุญแจปกปิดที่ถูกสร้างมาแล้วจากขั้นตอนการสร้างกุญแจรหัสลับขึ้นมาเข้ารหัสลับด้วยอัลกอริทึม RSA จากสมการที่ 5.2 แล้วส่งกุญแจปกปิดที่เข้ารหัสลับแล้วไปยังผู้รับ

ขั้นตอนที่ 4 ผู้รับได้รับข้อมูลจากผู้ส่งในขั้นตอนที่ 3 จะใช้กุญแจส่วนตัวถอดรหัสลับที่เป็นข้อมูลกุญแจปกปิดออกมาโดยใช้สมการที่ 5.5 ต่อจากนั้นผู้รับจะใช้กุญแจปกปิดเข้ารหัสลับข้อมูลส่งกลับไปหาผู้ส่ง เพื่อใช้ในการแลกเปลี่ยนข้อมูลระหว่างกัน

5.3.3 การเก็บรักษากุญแจรหัสลับ

กุญแจปกปิด (Secret key) และกุญแจส่วนตัว (Private key) จะต้องถูกเก็บเป็นความลับที่สุด เพื่อไม่ให้เกิดการบุกรุกได้ง่ายๆ เช่น มีการใช้ Password เข้าระบบและเก็บกุญแจรหัสลับไว้บนอุปกรณ์จัดเก็บข้อมูลที่ไม่สามารถเข้าได้จากโครงข่ายสื่อสารประเภทฟลอปปี้ดิสค์ หรือฮาร์ดดิสค์ เป็นต้น นอกจากนั้นเราอาจจัดเก็บในอุปกรณ์ฮาร์ดแวร์ประเภทชิปรอมหรือสมาร์ทการ์ด (Smart card) จะมีความปลอดภัยมากกว่าการใช้ Password เข้าระบบ แต่ถ้าเราเพิ่มความปลอดภัยโดยการเข้ารหัสลับกุญแจรหัสลับนั้นก่อนจะเก็บรักษาก็ยิ่งเพิ่มความปลอดภัยให้มากยิ่งขึ้นไปอีก

5.3.4 การสำรองกุญแจรหัสลับ

การสำรองกุญแจรหัสลับเพื่อป้องกันการสูญหาย การลืม Password เข้าระบบ หรืออุปกรณ์จัดเก็บเกิดการชำรุดไม่สามารถนำกลับคืนมาได้ เหตุการณ์ต่างๆ เหล่านี้สามารถเกิดขึ้นได้ และทำให้นำข้อมูลออกมาใช้งานไม่ได้ วิธีป้องกันปัญหาที่จะเกิดขึ้นทำได้หลายวิธี เช่น สำรองกุญแจรหัสลับกับหน่วยงานที่รับผิดชอบแจกจ่ายและดูแลกุญแจรหัสลับ สำรองกุญแจรหัสลับไว้ในสมาร์ทการ์ด และสำรองกุญแจรหัสลับไว้บนฮาร์ดดิสค์หรือฟล็อปปี้ดิสค์อีกชุดหนึ่ง เป็นต้น การสำรองกุญแจรหัสลับไว้กับคนอื่นจะต้องมั่นใจว่ามีระบบความปลอดภัยดีพอที่ไม่สามารถเข้าไปอ่านกุญแจรหัสลับของเราได้

5.3.5 ระยะเวลาสิ้นสุดกุญแจรหัสลับ

การกำหนดระยะเวลาสิ้นสุดกุญแจรหัสลับ เป็นการป้องกันการถูกทำลายกุญแจรหัสลับ เช่น การทำลายด้วยวิธีการแยกตัวประกอบของ RSA ที่มีระยะเวลาที่คาดว่าจะทำลายกุญแจรหัสลับได้ และการใช้กุญแจรหัสลับตัวเดิมนานๆ ถือว่าเป็นสิ่งที่ท้าทายผู้บุกรุกอย่างมาก ดังนั้นระยะเวลาสิ้นสุดของกุญแจรหัสลับควรกำหนดให้น้อยกว่าระยะเวลาที่คาดว่าจะทำลายกุญแจรหัสลับได้ การกำหนดระยะเวลาสิ้นสุดควรมีความสอดคล้องกับความยาวกุญแจรหัสลับ และยังขึ้นอยู่กับโครงข่ายสื่อสารที่ใช้แลกเปลี่ยนกุญแจรหัสลับ คือถ้ามีการแลกเปลี่ยนกันบนโครงข่ายสื่อสารสาธารณะแล้วควรกำหนดระยะเวลาสิ้นสุดกุญแจรหัสลับให้สั้นลง หลังจากสิ้นสุดระยะเวลาการใช้กุญแจรหัสลับแล้ว ผู้ใช้ต้องเลือกกุญแจรหัสลับใหม่ และควรให้มีความยาวกุญแจรหัสลับมากกว่าเดิม เนื่องจากคอมพิวเตอร์จะมีความสามารถในการคำนวณเพิ่มมากขึ้นเรื่อยๆ ทำให้สามารถใช้เวลาคำนวณ เพื่อทำลายกุญแจรหัสลับได้เร็วขึ้น

5.3.6 การทำลายกุญแจรหัสลับ

เมื่อเปลี่ยนกุญแจรหัสลับใหม่ กุญแจรหัสลับตัวเก่าต้องถูกทำลายไป ถึงจะไม่ใช้มันอีกก็ตาม เนื่องจากข้อมูลที่ถูกเข้ารหัสลับด้วยกุญแจรหัสลับตัวเก่าไว้ อาจถูกอ่านขึ้นได้ในภายหลัง การทำลายกุญแจรหัสลับสามารถทำได้หลายวิธี เช่น เมื่อเราเขียนกุญแจรหัสลับไว้บนแผ่นกระดาษ อาจ

ทำลายด้วยการเผาไฟหรือฉีกเป็นชิ้นเล็กๆ ไม่ให้นำมาปะติดปะต่อได้ เมื่อกุญแจรหัสลับถูกเก็บไว้ในอุปกรณ์ฮาร์ดแวร์ประเภท EEPROM ให้เขียนข้อมูลทับลงไปหลายๆ ครั้ง และถ้าเป็น EPROM หรือ ROM ควรทุบให้แตกละเอียด แต่ถ้าเก็บไว้ในอุปกรณ์จัดเก็บข้อมูลประเภทฮาร์ดดิสค์หรือดิสค์เก็ต ควรลบทิ้งและเขียนข้อมูลทับหลายๆ ครั้ง ถ้าสามารถทำการฟอร์แมตได้ก็จะดี เนื่องจากว่าการลบแฟ้มข้อมูลรหัสลับออกไป สามารถใช้โปรแกรมกู้กลับมาได้

5.4 การทดสอบอัลกอริทึมแบบผสม RC5 และ RSA

การทดสอบอัลกอริทึมกระทำบนเครื่องคอมพิวเตอร์ Intel Pentium 150 MHz ใช้คอมไพล์เลอร์ Borland C++Builder เวอร์ชั่น 3.0 รันบนแพล็ตฟอร์ม Windows 95 ข้อมูลที่ใช้ทดสอบเป็นแบบไบนารี่ไฟล์ขนาด 1.3 Mbytes เราแยกอัลกอริทึม RC5 ออกมาทดสอบความเร็วแต่เพียงอย่างเดียว โดยการทดสอบความเร็วของ RC5 อยู่ภายใต้รูปแบบ RC5-w/r/b โดยกำหนดค่า w คงที่ 32 บิต ให้ r และ b มีค่าเปลี่ยนแปลงไป แล้วเปรียบเทียบความเร็วในการคำนวณ สามารถแสดงการเปรียบเทียบได้จากตารางที่ 5.2

ตารางที่ 5.2 แสดงการเปรียบเทียบความเร็วจากการเลือกค่า r และ b ต่างๆ กัน

r (รอบ)	b (บิต)	ความเร็ว (กิโลบิต/ วินาที)
0	512	488
0	128	567
6	128	486
8	128	460
12	40	436
12	128	427
12	256	434
16	128	400
16	256	400

จากผลการทดสอบ RC5 ที่ค่า r และ b ต่าง ๆ กัน จะเห็นว่าจำนวนรอบ (r) จะมีผลต่อความเร็วในการคำนวณมาก แต่เมื่อให้ค่า b เปลี่ยนแปลงไปจะไม่มีผลด้านความเร็วมากนัก ซึ่งการเพิ่มค่า b จะมีผลโดยตรงกับระดับความปลอดภัยที่มากขึ้น ในการเลือกค่า r และ b เท่าใดนั้นเป็นสิ่งสำคัญในด้านความเร็วและความปลอดภัย คือ ถ้าเราต้องการจำนวนรอบน้อยเราต้องใช้ขนาดของกุญแจปกปิดมาก เช่น เมื่อเราเลือก r เท่ากับ 0 รอบ เราควรเลือก b เท่ากับ 512 บิต ไม่ควรเลือก b เท่ากับ 40 บิต หรือ เราเลือก r เท่ากับ 12 รอบ เราสามารถเลือก b เป็น 40 บิต หรือ 256 บิตก็ได้ เพราะว่าถ้าค่า r และค่า b น้อย ความปลอดภัยจะต่ำไปด้วย ในทางตรงกันข้ามเมื่อค่า b หรือ r มีค่ามากจะช่วยเพิ่มความปลอดภัยให้สูงขึ้นได้

การทดสอบอัลกอริทึมของ RC5 นี้ ต้องการแสดงให้เห็นถึงประสิทธิภาพทางด้านความเร็วในการเข้ารหัสลับแบบ RC5 ซึ่งจะเร็วกว่าอัลกอริทึมแบบ RSA ยกตัวอย่างเปรียบเทียบกันที่กุญแจรหัสลับขนาด 512 บิต จากตารางที่ 1 ของ RC5 จะให้ความเร็วในการคำนวณเท่ากับ 488 กิโลบิตต่อวินาที ในขณะที่ RSA มีความเร็วในการคำนวนเพียง 25 กิโลบิตต่อวินาที โดยทดสอบบนเครื่องเดียวกันและใช้อินพุทไฟล์เดียวกันกับ RC5 ทั้งนี้เนื่องจาก RSA มีข้อจำกัดเกี่ยวกับความเร็วในการคำนวณ ไม่เหมาะกับการเข้ารหัสลับข้อมูลขนาดใหญ่ จึงนำมาใช้เพื่อการแลกเปลี่ยนกุญแจรหัสลับแทน

การทดสอบความปลอดภัยของ RSA ที่ใช้ในการแลกเปลี่ยนกุญแจรหัสลับ และกุญแจรหัสลับที่สร้างขึ้น ไม่ได้แสดงให้เห็นในที่นี้ เนื่องจากกำหนดวิธีการทดสอบค่อนข้างลำบาก ส่วนใหญ่จะให้ผู้ที่เป็นแฮ็กเกอร์ (Hacker) ทำการทดสอบจุดอ่อนของอัลกอริทึม แต่เมื่อพิจารณาประเด็นความปลอดภัยของ RSA ที่ใช้ในการแลกเปลี่ยนกุญแจรหัสลับแล้ว จะพบว่าขึ้นอยู่กับขนาดของกุญแจรหัสลับเป็นสำคัญ ในปัจจุบันถือว่ากุญแจรหัสลับของ RSA ที่มีขนาดใหญ่กว่า 512 บิตขึ้นไปถือว่าปลอดภัยแล้ว ถึงกระนั้นยังได้มีผู้บุกรุกพยายามคิดค้นหาวิธีการใหม่ ๆ เพื่อทำลายอัลกอริทึมของ RSA ยกตัวอย่างวิธีการที่ผู้บุกรุกทำลายกุญแจรหัสลับ RSA จากสมการ $P = C^d \bmod n$ และรู้กุญแจสาธารณะ (n,e) โดยที่ $n = p*q$ ผู้บุกรุกพยายามที่จะหาค่า d โดยใช้วิธีการแยกตัวประกอบ n ออกมา เพื่อหาค่า p และ q นำไปสู่ค่า d จากการนับหรือสุ่ม สามารถแสดงตัวอย่างอัลกอริทึมการทำลายรหัสลับ RSA ดังนี้

```
p[0] = 1;
c[0] = x;
for i = 0 to (bits in d-1)
   if (bit i of d) = 1 then
        p[i]+1 = (p[i] * c[i]) mod n
   else
        p[i]+1 = p[i];
    d[i]+1 = d[i]^2 mod n;
```

ถ้าเราพิจารณาที่กุญแจรหัสลับขนาด 512 บิต หรือคิดเป็นตัวเลขจำนวนเต็ม (Interger) ที่เก็บได้ในตัวแปรขนาด 512 บิต หรือ 64 ไบต์ คือ Integer มีขนาด 2 ไบต์ เก็บตัวเลขได้ 5 หลัก (Digits) ดังนั้น 64 ไบต์ จะเก็บได้ 160 หลัก เราจะต้องสุ่มหาค่า d ถึง 160 หลัก เลยทีเดียว ซึ่งในปี 1995 Rivest ผู้ร่วมคิดค้นอัลกอริทึม RSA ได้กล่าวไว้ว่าที่ขนาดกุญแจรหัสลับ 384 บิต หรือ 120 หลัก ใช้เวลาการทำลายจากอัลกอริทึม Number Field Sieve (NFS) ซึ่งเป็นอัลกอริทึมที่เร็วที่สุดในปัจจุบันถึง 400 ปี ของเวลาการคำนวณที่ 1 ล้านคำสั่งต่อวินาทีในหนึ่ง CPU time สามารถแสดงฟังก์ชั่นการแยกตัวประกอบเพื่อหาค่า n ของ NFS ดังในสมการที่ 5.7 คือ

$$H(n) = \exp\left(1.526 (\ln n)^{1/3} (\ln \ln n)^{2/3}\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (5.7)$$

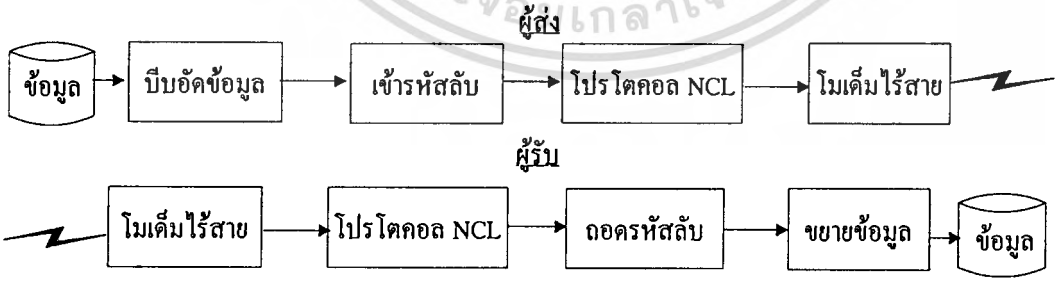
นอกจากนั้นผู้บุกรุกสามารถมุ่งทำลายกุญแจรหัสลับด้วยวิธีการใช้พละกำลัง (Brute force attack) เพื่อสุ่มหาข้อมูลอินพุท ที่เป็น Seed buffer ไปเรื่อย ๆ โดยใช้แหล่งกำเนิดตัวเลขสุ่มที่ล่วงรู้มา และคาดเดาข้อมูลเอาท์พุทที่ควรจะเป็นไปได้ ซึ่งการใช้วิธีการดังกล่าวนี้ถ้าใช้เครื่องคอมพิวเตอร์ที่มีความเร็วในการคำนวณ 1,000,000,000 คำสั่งต่อวินาที ต้องใช้เวลาถึง 1.07E22 ปี เพื่อหาค่าข้อมูลเอ้าท์พุทต้นฉบับออกมา

บทที่ 6
ส่วนโปรแกรมสื่อสารข้อมูลระบบไร้สาย

6.1 โครงสร้างการทำงานทั่วไป

ลักษณะของโปรแกรมเป็นแบบรับส่งจดหมายอิเล็กทรอนิกส์ (E-mail) สามารถทำการแนบ (Attach) แฟ้มข้อมูล (File) ที่ต้องการส่งไปพร้อมกันได้ การรับส่งข้อมูลจะกระทำกันระหว่างอุปกรณ์สื่อสารข้อมูลเคลื่อนที่ไร้สายทั้งสอง โดยที่อุปกรณ์ทั้งสองจะผลัดกันรับส่งข้อมูล แล้วแต่ฝ่ายใดเป็นฝ่ายร้องขอการส่งข้อมูล ข้อมูลจากผู้ส่ง (Sender) ประกอบด้วยแฟ้มข้อมูล และข้อความแบบอิเล็กทรอนิกส์เมล์ ข้อมูลที่ส่งจะถูกแบ่งออกเป็นเฟรมข้อมูล(Data frame) ในแต่ละเฟรมข้อมูลจะผ่านขบวนการบีบอัด (Compression) ก่อนการเข้ารหัสลับ (Encryption) แล้วส่งไปยังผู้รับโดยใช้โปรโตคอล NCL ผ่านโมเด็มไร้สายแปลงเป็นสัญญาณคลื่นวิทยุ ส่งต่อไปยังโครงข่ายเวิลด์ดาต้า ทางผู้รับ (Receiver) จะรับข้อมูลโดยโปรโตคอล NCL แล้วทำการถอดรหัสลับ (Decryption) ข้อมูลและขยายขนาด (Decompress) ข้อมูล ฝ่ายผู้รับสามารถอ่านข้อความจากผู้ส่งได้ และบันทึกแฟ้มข้อมูลลงเก็บไว้ ดังแสดงในรูปที่ 6.1 โครงสร้างการทำงานโปรแกรมสื่อสารข้อมูลระบบไร้สาย

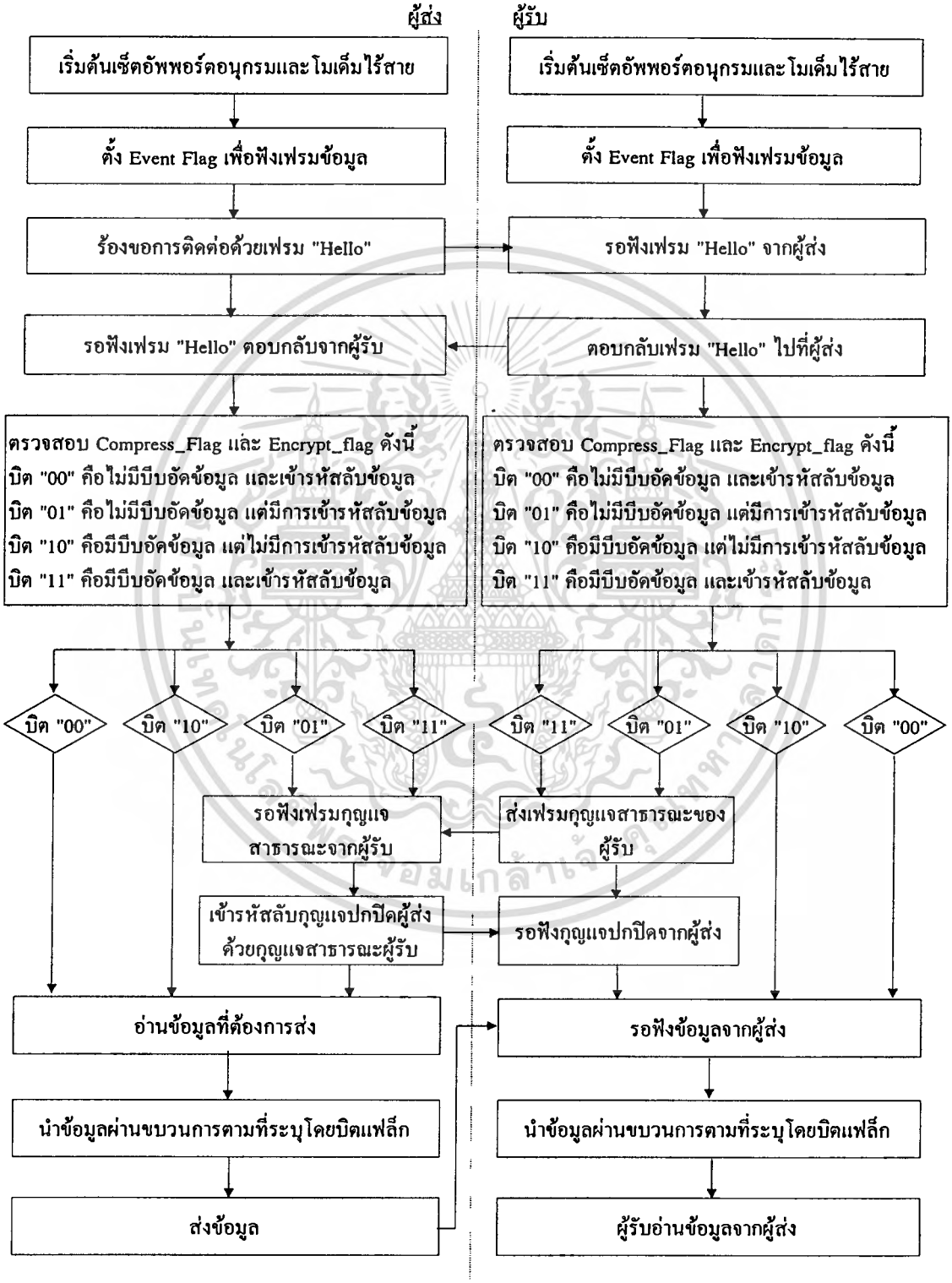
รูปที่ 6.1 โครงสร้างการทำงานโปรแกรมสื่อสารข้อมูลระบบไร้สาย



6.2 หลักการทำงานของโปรแกรม

ผู้รับและผู้ส่งจะต้องสร้างกุญแจปกปิด RC5 กุญแจสาธารณะและกุญแจส่วนตัว RSA ไว้ก่อนที่จะมีการแลกเปลี่ยนข้อมูลระหว่างกัน โปรแกรมมีความยืดหยุ่นในการใช้งานสามารถกำหนดการเลือกใช้หรือยกเลิกการบีบอัดข้อมูล และการเข้ารหัสลับข้อมูลได้ ในรูปที่ 6.2 แสดงโฟล์วชาร์ทการทำงานของโปรแกรมรับส่งข้อมูลระบบไร้สาย

รูปที่ 6.2 แสดงโฟล์วชาร์ทการทำงานของโปรแกรมรับส่งข้อมูลระบบไร้สาย



ขั้นตอนการทำงานของโปรแกรมรับส่งข้อมูลดังในรูปที่ 6.2 คือ

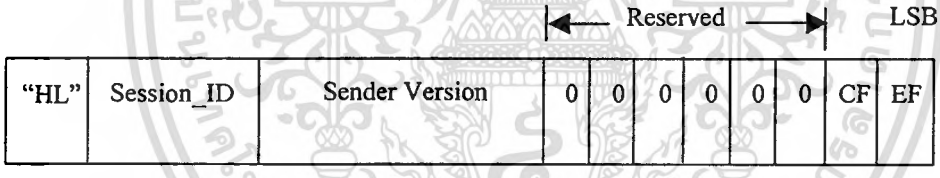
ขั้นตอนที่ 1 ผู้รับและผู้ส่งสร้างกุญแจรหัสลับ โดยเลือกความยาวกุญแจรหัสลับตามขนาดที่ต้องการ

ขั้นตอนที่ 2 การทำงานจะเริ่มต้นที่การเลือกหมายเลขพอร์ต (Port number), การเซ็ตอัพความเร็ว (Speed), ขนาดบิตข้อมูล, ชนิดการควบคุมการไหลข้อมูล (Flow control) และการเริ่มสถานะใหม่ (Initialize) ของอุปกรณ์โมเด็มไร้สาย

ขั้นตอนที่ 3 ตั้ง Event Flag เพื่อรอฟังเฟรมข้อมูล เช่น ข้อมูลจากผู้ส่ง การตรวจสอบสถานะของโมเด็มไร้สายเกี่ยวกับสภาพของแบตเตอรี่ และตรวจสอบความแรงของสัญญาณ เป็นต้น การทำงานของ Event Flag จะถูก Set เมื่อมีข้อมูลเข้ามา ถ้าไม่มีข้อมูลเข้ามา จะอยู่ในโหมดหลับ (Sleep mode)

ขั้นตอนที่ 4 การติดต่อจะเริ่มต้นจากผู้ส่ง ทำการร้องขอการเชื่อมต่อ โดยใช้เฟรม "Hello" ส่งไปยังผู้รับ ในรูปที่ 6.3 แสดงรูปแบบเฟรม "Hello" ของผู้ส่ง

รูปที่ 6.3 แสดงรูปแบบเฟรม "Hello" ของผู้ส่ง



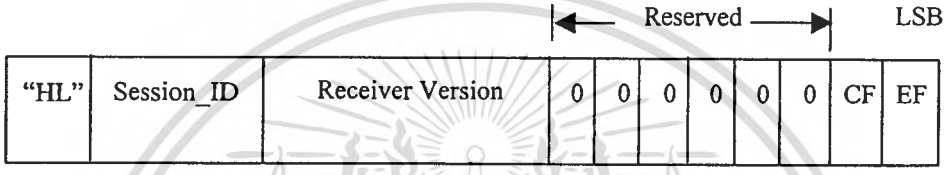
รายละเอียดของรูปแบบเฟรม "Hello" มีดังนี้

1. ฟิลด์ "HL" มาจากคำว่า Hello มีขนาด 2 ไบต์ เป็นตัวบอกชนิดของเฟรม "Hello"
2. ฟิลด์ Session_ID มีขนาด 1 ไบต์ แสดงหมายเลขการเชื่อมต่อกับผู้รับ โดยการเชื่อมต่อในครั้งแรกจะมีค่าเป็น "0" เมื่อมีการเรียกคืน (Resume) การเชื่อมต่อจะต้องเพิ่มค่า Session_ID ขึ้นไป
3. ฟิลด์ Sender Version มีขนาด 2 ไบต์ แสดงเวอร์ชั่นการแลกเปลี่ยนกุญแจรหัสลับของผู้ส่ง ในที่นี้กำหนดเป็นเวอร์ชั่น 1.0
4. ฟิลด์ Compress_Flag หรือ CF มีขนาด 1 บิต มีการใช้ตัวแปรขนาด 1 ไบต์ ร่วมกับ Encrypt_flag โดยใช้ตำแหน่งบิตที่ 1 แสดงถึงบิตแฟล็กการใช้ /ยกเลิกการบีบอัดข้อมูล โดยแทนค่า "0" เมื่อยกเลิก และค่า "1" เมื่อเลือกใช้
5. ฟิลด์ Encrypt_Flag หรือ EF มีขนาด 1 บิต ใช้ตำแหน่งบิตที่ 0 แสดงถึงบิตแฟล็กการใช้ /ยกเลิกการเข้ารหัสลับข้อมูล โดยแทนค่า "0" เมื่อยกเลิก และค่า "1" เมื่อเลือกใช้

ขั้นตอนที่ 5 ผู้รับรอฟังเฟรม “Hello” จากผู้ส่ง

ขั้นตอนที่ 6 ผู้รับตรวจสอบเฟรม “Hello” จากผู้ส่ง และส่งกลับไปยังผู้ส่งด้วยเฟรมเดียวกัน พร้อมกับเก็บค่า Session_ID ไว้ใช้อ้างอิงในการเชื่อมต่อในครั้งต่อไป และเก็บค่า Compress_Flag และ Encrypt_Flag ไว้เป็นตัวเลือกการทำการบีบอัดข้อมูลและการเข้ารหัสลับข้อมูล ซึ่งรูปแบบเฟรม “Hello” ของผู้รับแสดงให้เห็นในรูปที่ 6.4

รูปที่ 6.4 แสดงรูปแบบเฟรม “Hello” ของผู้รับ



รายละเอียดของรูปแบบเฟรม “Hello” ของผู้รับมีดังนี้

1. ฟิลด์ “HL” มาจากคำว่า Hello มีขนาด 2 ไบต์ เป็นตัวบอกชนิดของเฟรม “Hello”
2. ฟิลด์ Session_ID มีขนาด 1 ไบต์ แสดงหมายเลขการเชื่อมต่อกับผู้รับ การเชื่อมต่อในครั้งแรกจะมีค่าเป็น “0” และจะทำการเปรียบเทียบกับ Session_ID ของผู้ส่ง ถ้าไม่ตรงกันจะยกเลิกการเชื่อมต่อทันที
3. ฟิลด์ Receiver Version มีขนาด 2 ไบต์ แสดงเวอร์ชั่นการแลกเปลี่ยนกุญแจรหัสลับของผู้รับ ในที่นี้กำหนดเป็นเวอร์ชั่น 1.0
4. ฟิลด์ Compress_Flag หรือ CF มีขนาด 1 บิต มีการใช้ตัวแปรขนาด 1 ไบต์ ร่วมกับ Encrypt_flag โดยใช้ตำแหน่งบิตที่ 1 แสดงถึงบิตแฟล็กการใช้ /ยกเลิกการบีบอัดข้อมูล โดยแทนค่า “0” เมื่อยกเลิก และค่า “1” เมื่อเลือกใช้
5. ฟิลด์ Encrypt_Flag หรือ EF มีขนาด 1 บิต ใช้ตำแหน่งบิตที่ 0 แสดงถึงบิตแฟล็กการใช้ /ยกเลิกการเข้ารหัสลับข้อมูล โดยแทนค่า “0” เมื่อยกเลิก และค่า “1” เมื่อเลือกใช้

ขั้นตอนที่ 7 ผู้ส่งรอฟังเฟรม “Hello” ตอบกลับจากผู้รับ

ขั้นตอนที่ 8 ทั้งผู้ส่งและผู้รับตรวจสอบว่ามีการกำหนด Compress_Flag และ Encrypt_Flag ไว้อย่างไร เราใช้ตัวแปรขนาด 1 ไบต์ โดยให้บิตที่ “0” แทน Encrypt_Flag และบิตที่ “1” แทน Compress_flag การตรวจสอบบิต จะกระทำทั้งสองบิตพร้อมกัน แสดงได้ดังนี้

บิต “00” แสดงถึงไม่มีการบีบอัดข้อมูล และเข้ารหัสลับข้อมูล

บิต “01” แสดงถึงไม่มีการบีบอัดข้อมูล แต่มีการเข้ารหัสลับข้อมูล

บิต “10” แสดงถึงมีการบีบอัดข้อมูล แต่ไม่มีการเข้ารหัสลับข้อมูล

บิต “11” แสดงถึงมีการบีบอัดข้อมูล และเข้ารหัสลับข้อมูล

ขั้นตอนที่ 9 ผู้รับ (Receiver) ทำการตรวจสอบทั้งสองบิตพร้อมกัน คือ

กรณีที่ 1 ถ้าพบบิต “00” ให้รอฟังเฟรมข้อมูลจากผู้ส่ง และรับข้อมูลที่เข้ามาทางโมเด็มไร้สายด้วยโปรโตคอล NCL มีรูปแบบเฟรมข้อมูลดังที่กล่าวมาแล้วในบทที่ 3

กรณีที่ 2 ถ้าพบบิต “01” จะทำการส่งเฟรมกุญแจสาธารณะดังแสดงในรูปที่ 6.5 ของผู้รับไปยังผู้ส่ง จากนั้นให้รอฟังเฟรมกุญแจปกปิดดังแสดงในรูปที่ 6.6 จากผู้ส่ง โดยรับข้อมูลที่เข้ามาทางโมเด็มไร้สายด้วยโปรโตคอล NCL และใช้กุญแจปกปิดของผู้ส่งเปิดอ่านข้อมูลที่รับได้

รูปที่ 6.5 แสดงรูปแบบเฟรมการส่งกุญแจสาธารณะ RSA

“KP”	Session_ID	Public Key
2 bytes	1 byte	Max 512 bytes

“KP” คือ Key Public

รูปที่ 6.6 แสดงรูปแบบเฟรมการส่งกุญแจปกปิด RC5

“KS”	Session_ID	Secret Key Encrypted
2 bytes	1 byte	Max 512 bytes

“KS” คือ Key Secret

กรณีที่ 3 ถ้าพบบิต “10” ให้รอฟังเฟรมข้อมูลจากผู้ส่ง โดยรับข้อมูลที่เข้ามาทางโมเด็มไร้สายด้วยโปรโตคอล NCL และทำการขยายขนาดข้อมูล (Decompression) ที่รับได้ ทำให้เป็นข้อมูลต้นฉบับ

กรณีที่ 4 ถ้าพบบิต “11” จะทำการส่งเฟรมกุญแจสาธารณะของผู้รับไปยังผู้ส่ง จากนั้นให้รอฟังเฟรมกุญแจปกปิดจากผู้ส่ง มีรูปแบบเฟรมเช่นเดียวกันกับที่แสดงในรูปที่ 6.5 และ 6.6 ตาม

ลำดับ โดยรับข้อมูลที่เข้ามาทางโมเด็มไร้สายด้วยโปรโตคอล NCL และใช้กุญแจปกปิดของผู้ส่งเปิดอ่านข้อมูล จากนั้นทำการขยายข้อมูลที่รับได้ ทำให้เป็นข้อมูลต้นฉบับ

ขั้นตอนที่ 10 ผู้ส่ง (Sender) ทำการตรวจสอบทั้งสองบิตพร้อมกัน คือ

กรณีที่ 1 ถ้าพบบิต "00" ให้อ่านข้อมูลการพิมพ์ข้อความจากจดหมายอิเล็กทรอนิกส์ และอ่านแฟ้มข้อมูลกรณีเราแนบส่งไปพร้อมกัน นำมาประกอบกันเป็นเฟรมข้อมูลดังแสดงในรูปที่ 6.7 (ก), (ข) และ (ค) แล้วทำการส่งด้วยโปรโตคอล NCL ผ่านโมเด็มไร้สายไปยังผู้รับ

รูปที่ 6.7 แสดงรูปแบบเฟรมที่เป็นส่วนข้อมูล (Data frame)

(ก) รูปแบบเฟรมข้อความจากจดหมายอิเล็กทรอนิกส์

"DT"	Sender/ Recipient	U S	Subject	U S	Body Text	U S	File name	U S	File length
2 bytes	< 40 bytes		< 100 bytes		< 1500 bytes		< 40 bytes		5 bytes

DT คือ Data Type

US คือ Unit Seperator มีค่า 0x1F

(ข) รูปแบบเฟรมส่งแฟ้มข้อมูล

"FC"	Ascii or Binary File Content
2 bytes	Max 1024 bytes

FC คือ File Content

(ค) รูปแบบเฟรมที่บอกถึงการสิ้นสุดแฟ้มข้อมูล

"FT"	ETX
2 bytes	1 byte

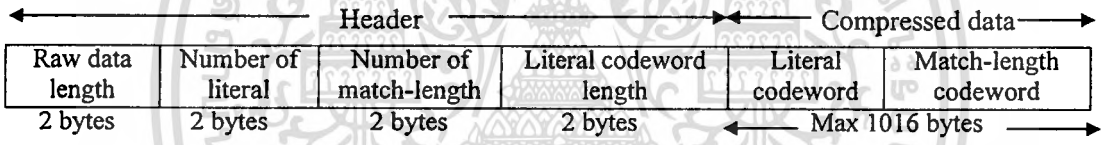
FT คือ File Terminate

ETX คือ End Transmission มีค่า 0x03

กรณีที่ 2 ถ้าพบบิต "01" จะรอฟังเฟรมกุญแจสาธารณะจากผู้รับ เมื่อรับเฟรมแล้วให้ทำการเข้ารหัสลับกุญแจปกปิดของผู้ส่งด้วยอัลกอริทึมของ RSA ส่งกลับไปยังผู้รับ มีรูปแบบเฟรมเช่นเดียวกันกับที่แสดงในรูปที่ 6.5 และ 6.6 หลังจากนั้นให้อ่านข้อมูลการพิมพ์ข้อความจากจดหมายอิเล็กทรอนิกส์ และอ่านแฟ้มข้อมูลกรณีเราแนบส่งไปพร้อมกัน นำส่วนเฟรมข้อมูลที่ประกอบกันดังในรูปที่ 6.7 (ก) และ 6.7 (ข) มาเข้ารหัสลับด้วยอัลกอริทึม RC5 แล้วทำการส่งด้วยโปรโตคอล NCL ผ่านโมเด็มไร้สายไปยังผู้รับ

กรณีที่ 3 ถ้าพบบิต "10" ให้อ่านข้อมูลการพิมพ์ข้อความจากจดหมายอิเล็กทรอนิกส์ และอ่านแฟ้มข้อมูลกรณีเราแนบส่งไปพร้อมกัน จากนั้นให้นำส่วนเฟรมข้อมูลดังในรูปที่ 6.7 (ก) และ (ข) มาผ่านขบวนการบีบอัดข้อมูล (Compression) ด้วยอัลกอริทึมของ LZP โดยมีรูปแบบเฟรมข้อมูลหลังจากการบีบอัดแล้วประกอบด้วยส่วนหัว (Header) และส่วนข้อมูลบีบอัดดังแสดงในรูปที่ 6.8 แล้วทำการส่งด้วยโปรโตคอล NCL ผ่านโมเด็มไร้สายไปยังผู้รับ

รูปที่ 6.8 แสดงรูปแบบเฟรมข้อมูลหลังจากผ่านการบีบอัดข้อมูล



Raw data length คือขนาดของข้อมูลเดิมก่อนการบีบอัด

Number of literal คือจำนวนของข้อมูลที่ไม่แมทช์กัน

Number of match-length คือจำนวนของความยาวข้อมูลที่แมทช์กันได้

Literal codeword length คือขนาดของรหัสข้อมูลที่ไม่แมทช์กัน

Literal codeword คือรหัสข้อมูลของข้อมูลที่ไม่แมทช์กัน

Match-length codeword คือรหัสข้อมูลของความยาวข้อมูลที่แมทช์กัน

กรณีที่ 4 ถ้าพบบิต "11" จะรอฟังเฟรมกุญแจสาธารณะจากผู้รับ เมื่อรับเฟรมแล้วให้ทำการเข้ารหัสลับกุญแจปกปิดของผู้ส่งด้วยอัลกอริทึมของ RSA ส่งกลับไปยังผู้รับ มีรูปแบบเฟรมเช่นเดียวกันกับที่แสดงในรูปที่ 6.5 และ 6.6 หลังจากนั้นให้อ่านข้อมูลการพิมพ์ข้อความจากจดหมายอิเล็กทรอนิกส์ และอ่านแฟ้มข้อมูลกรณีเราแนบส่งไปพร้อมกัน มีรูปแบบเฟรมข้อมูลเช่นเดียวกันกับที่แสดงในรูปที่ 6.7 จากนั้นนำมาบีบอัดข้อมูล (Compression) ด้วยอัลกอริทึม LZP ก่อนผ่านขบวนการเข้ารหัสลับด้วยอัลกอริทึม RC5 แล้วทำการส่งด้วยโปรโตคอล NCL ผ่านโมเด็มไร้สายไปยังผู้รับต่อไป

นอกจากนี้ข้อมูลที่ส่งออกไปจะมีการตรวจสอบการสูญหายของข้อมูลด้วย โดยทางผู้ส่งจะสร้างบัฟเฟอร์วงแหวน (Ring buffer) ไว้ 10 บัฟเฟอร์ เพื่อใช้เก็บหมายเลขลำดับ (Sequence number) และข้อมูลก่อนส่งออกไป เมื่อส่งด้วยเฟรมคำสั่ง (Command) ของโปรโตคอล NCL ไปแล้ว จะมีการรอจนกว่าจะได้รับการตอบสนอง (Response) จากโมเด็มไร้สายก่อนที่จะทำการส่งเฟรมข้อมูลถัดไป ทั้งนี้ก็เพื่อให้แน่ใจว่าไม่มีการสูญหายของเฟรมการส่งข้อมูล สำหรับฝ่ายรับจะสร้างบัฟเฟอร์วงแหวนไว้ 10 บัฟเฟอร์เช่นเดียวกัน สำหรับเก็บหมายเลขลำดับและข้อมูลที่รับได้โดยเก็บเรียงตามลำดับ (Sequence) ของข้อมูล เมื่อหมายเลขลำดับครบ 10 แล้ว จะทำการตรวจสอบดูว่าข้อมูลสูญหายที่หมายเลขลำดับใดบ้าง ถ้ามีการสูญหายของข้อมูลจะร้องขอไปยังผู้ส่งด้วยเฟรมควบคุมการร้องขอดังในรูปที่ 6.9 (ข) และระบุหมายเลขลำดับที่ต้องการลงในฟิลด์ Sequence number บนรูปแบบข้อมูล DataTAC Messaging ในรูปที่ 3.22 ที่แสดงไว้ในบทที่ 3 การร้องขอนี้จะมีการส่งซ้ำสูงสุด 3 ครั้งของแต่ละหมายเลขลำดับ เมื่อผู้รับรับข้อมูลครบถ้วนแล้วจะเขียนลงแฟ้มข้อมูลพร้อมกับส่งเฟรมตอบกลับดังในรูปที่ 6.9 (ก) ไปยังผู้ส่ง เพื่อบอกให้ส่งเฟรมถัดไปได้ และถ้าผู้รับได้รับเฟรมสิ้นสุดการส่งข้อมูลดังในรูปที่ 6.7 (ค) แล้ว จะส่งเฟรมควบคุมการตอบกลับไปยังผู้ส่งด้วยเช่นกัน ซึ่งการตอบกลับนี้จะมีขึ้นทุก ๆ 10 บัฟเฟอร์ข้อมูล หากกรณีได้รับเฟรมสิ้นสุดการส่งข้อมูลก่อนที่จะครบ 10 บัฟเฟอร์ จะทำการส่งเฟรมตอบกลับไปทันที

รูปที่ 6.9 รูปแบบเฟรมควบคุมการรับส่งข้อมูล

(ก) รูปแบบเฟรมตอบกลับ

"AC"	ACK
2 bytes	1 byte

AC คือ ACK บ่งบอกว่าเป็นเฟรมควบคุมการส่งข้อมูล

ACK คือ Acknowledge มีค่า 0x06

(ข) รูปแบบเฟรมการร้องขอเมื่อข้อมูลมีการสูญหาย

"AC"	NAK
2 bytes	1 byte

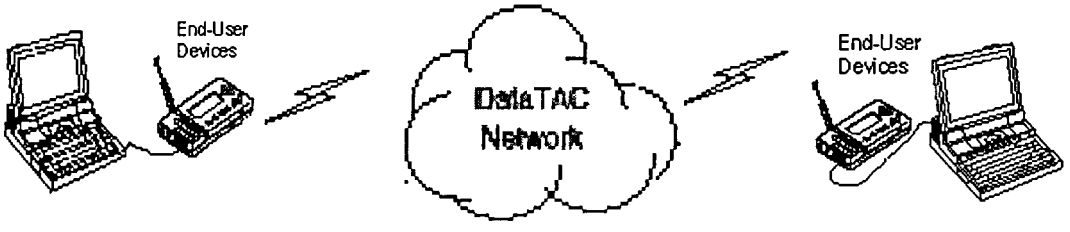
AC คือ ACK บ่งบอกว่าเป็นเฟรมควบคุมการส่งข้อมูล

NAK คือ No Acknowledge มีค่า 0x15

6.3 การทดสอบโปรแกรมการใช้งานจริง

การทดสอบใช้เครื่องคอมพิวเตอร์ Pentium 150 MHz จำนวน 2 เครื่อง และโมเด็มไร้สายของโมโตโรล่า รุ่น InfoTAC จำนวน 2 ตัว มีหมายเลข ID คือ EE022E45 และ EE022E4F ต่อเข้าเครื่องคอมพิวเตอร์ทางพอร์ตอนุกรม RS-232 มีความเร็วที่พอร์ตอนุกรม (Baud rate) เท่ากับ 9600 บิตต่อวินาที ใช้โปรโตคอล NCL ส่งข้อมูลเป็นคลื่นวิทยุแบบ RD-LAP ที่ความเร็ว 19.2 กิโลบิตต่อวินาที การส่งข้อมูลจะต้องอ้างหมายเลข ID ของโมเด็มไร้สายปลายทางที่ต้องการติดต่อสื่อสารด้วย แสดงรูปแบบที่ใช้ในการทดสอบในรูปที่ 6.10 สำหรับอัลกอริทึมการบีบอัดข้อมูลของ LZP ใช้ตารางแฮช เปรียบเทียบข้อมูลมีขนาดเท่ากับ 11 บิต หรือเท่ากับ 4 กิโลไบต์ และ LZW ใช้หน่วยความจำแบบอาร์เรย์ในการเปรียบเทียบข้อมูลมีขนาดเท่ากับ 4 กิโลไบต์ แฟ้มข้อมูลที่ใช้ทดสอบเป็นแฟ้มข้อมูลมาตรฐานจาก Calgory Corpus [13] โดยตัดเนื้อข้อมูลมาทดสอบเพียงบางส่วนเนื่องจากไม่ต้องการไปกระทบกับการใช้งานของผู้อื่น ที่ใช้แบนด์วิตร่วมกันบนระบบโครงข่ายเวิลด์คาต้า และแนบข้อมูลไปพร้อมกับอิเล็กทรอนิกส์เมล์ มีขนาดแฟ้มข้อมูลดังแสดงให้เห็นในตารางที่ 6.1 กรณีที่ไม่มีการบีบอัดข้อมูลจะให้ความยาวของข้อมูลที่อ่านจากแฟ้มข้อมูลครั้งละ 512 ไบต์ แต่ถ้ามีการบีบอัดข้อมูลจะให้ความยาวที่อ่านจากแฟ้มข้อมูลเท่ากับ 1024 ไบต์ จากนั้นทำการเปรียบเทียบปริมาณการส่งข้อมูล (Throughput) ที่ได้รับเมื่อไม่ผ่านการบีบอัดข้อมูลและผ่านการบีบอัดข้อมูลทั้งแบบ LZP และ LZW โดยปริมาณข้อมูล (Throughput) ที่ได้รับจะคำนวณที่ฝ่ายส่ง ซึ่งคำนวณเทียบกับเวลารวมทั้งหมดของการส่งข้อมูลบนโครงข่ายคือ เวลาของการบีบอัดข้อมูล และเวลาของการเข้ารหัสลับ โดยนับเวลาเริ่มต้นก่อนการบีบอัดและเข้ารหัสลับข้อมูล และเวลาสิ้นสุดได้รับต่อเมื่อส่งเฟรมสิ้นสุดการส่งข้อมูลไปแล้ว จนกระทั่งได้รับเฟรมตอบกลับจากผู้รับ หลังจากนั้นจึงคำนวณเปรียบเทียบข้อมูลที่ส่งและเวลาที่ใช้ เพื่อหาปริมาณข้อมูลที่ได้รับต่อไป

รูปที่ 6.10 แสดงรูปแบบที่ใช้ในการทดสอบ



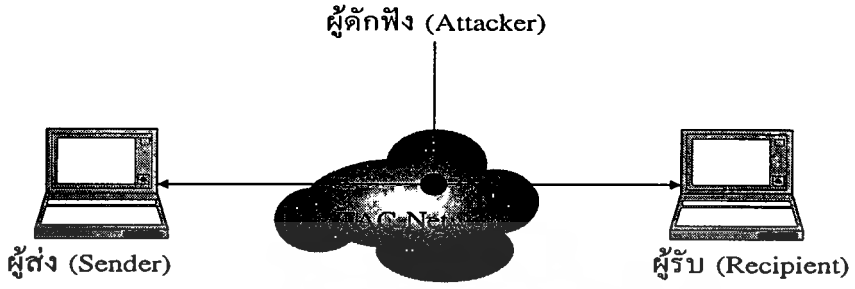
ตารางที่ 6.1 ผลการเปรียบเทียบปริมาณการส่งข้อมูล (Throughput) ที่ได้รับเมื่อไม่ผ่านการบีบอัดข้อมูลและผ่านการบีบอัดข้อมูลทั้งแบบ LZP และ LZW

ข้อมูลต้นฉบับ	ขนาดข้อมูล (ไบต์)	ปริมาณข้อมูลที่ได้รับเมื่อไม่มีการบีบอัดและเข้ารหัสลับข้อมูล (ไบต์/วินาที)	ปริมาณข้อมูลที่ได้รับเมื่อมีการบีบอัดแบบ LZP และเข้ารหัสลับข้อมูล (ไบต์/วินาที)	ปริมาณข้อมูลที่ได้รับเมื่อมีการบีบอัดแบบ LZW และเข้ารหัสลับข้อมูล (ไบต์/วินาที)
Obj1	3,638	80	139	103
Paper4	2,892	75	137	100
Paper5	2,743	81	141	105
Paper6	4,243	86	150	106
Progc	4,378	86	151	106
Progp	2,979	81	142	102
ค่าเฉลี่ย		81.5	143.3	103.7

ผลการทดสอบจากตารางที่ 6.1 จะเห็นว่าถ้าข้อมูลไม่ได้ผ่านการบีบอัดและเข้ารหัสข้อมูลแล้วการส่งข้อมูลบนโครงข่ายเวิลด์ดาต้าจะได้ปริมาณการส่งข้อมูล (Throughput) โดยเฉลี่ยเท่ากับ 81.5 ไบต์ต่อวินาที แต่เมื่อได้มีการบีบอัดข้อมูลด้วยอัลกอริทึม LZP แล้ว Throughput ที่ได้รับโดยเฉลี่ยจะเท่ากับ 143.3 ไบต์ต่อวินาที มีประสิทธิภาพเพิ่มขึ้น 76 เปอร์เซ็นต์ และอัลกอริทึมของ LZW มี Throughput โดยเฉลี่ยเท่ากับ 103.7 ไบต์ต่อวินาที มีประสิทธิภาพเพิ่มขึ้น 27 เปอร์เซ็นต์ ทั้งนี้เนื่องจากอัลกอริทึมของ LZP และ LZW มีอัตราการบีบอัดข้อมูล (Compression ratio) อยู่ที่ 3.06 และ 3.67 บิตต่อไบต์ ตามลำดับ จึงทำให้อัลกอริทึมของ LZP ใช้เวลาส่งผ่านข้อมูลน้อยกว่า LZW สำหรับเวลาที่ใช้ประมวลผลทั้งของการบีบอัดและการเข้ารหัสลับข้อมูลมีผลน้อยมากต่อความเร็วที่ใช้รับส่งข้อมูลบนโครงข่ายเวิลด์ดาต้า

สำหรับการทดสอบของการเข้ารหัสลับข้อมูลกำหนดวิธีค่อนข้างลำบาก แต่ในที่นี้จะขอยกตัวอย่างวิธีการต่างๆ ที่ผู้ดักฟังสามารถที่จะกระทำได้เมื่อมีการรับส่งข้อมูลผ่านโครงข่ายเวิลด์ดาต้า ในรูปที่ 6.11 แสดงการดักฟังข้อมูลที่รับส่งระหว่างผู้ส่งและผู้รับ

รูปที่ 6.11 แสดงการดักฟังข้อมูลที่รับส่งระหว่างผู้ส่งและผู้รับ



จากรูปที่ 6.11 เมื่อผู้ส่งต้องการจะส่งข้อมูลไปยังผู้รับ ผู้รับจะต้องมีกุญแจปกปิดของผู้ส่งก่อนถึงจะอ่านข้อมูลได้ ดังนั้นก่อนมีการเข้ารหัสลับข้อมูล ผู้ส่งต้องทำการส่งกุญแจปกปิดไปให้ผู้รับก่อน ซึ่งในขั้นตอนต่างๆ เหล่านี้ผู้ดักฟังอาจดักจับข้อมูลได้ ซึ่งผู้ดักฟังสามารถเป็นได้ทั้งบุคคลทั่วไปที่ใช้งานอยู่บนโครงข่าย หน่วยงานข่าวกรอง หรือผู้บริหารและผู้ดูแลระบบโครงข่ายเอง การดักฟังจะกระทำได้บุคคลนั้นต้องรู้อัลกอริทึมการเข้ารหัสลับที่ใช้ รู้จุดอ่อนของอัลกอริทึม และใช้พละกำลังในการสุ่มหาจนกว่าจะทำลายรหัสลับได้ ซึ่งในที่นี้จะขอกล่าวถึงวิธีการที่ผู้ดักฟังสามารถกระทำได้ เพื่อเป็นแนวทางในการพิจารณาถึงความปลอดภัยของอัลกอริทึมที่นำเสนอในวิทยานิพนธ์ฉบับนี้ ดังนี้คือ

1). ผู้ดักฟังพยายามจะดักฟังระหว่างการแลกเปลี่ยนกุญแจปกปิด ในที่นี้เราได้เข้ารหัสกุญแจปกปิดด้วยอัลกอริทึม RSA เมื่อผู้ดักฟังได้ข้อมูลของกุญแจปกปิดที่เข้ารหัสลับไว้แล้วไป ก็ไม่สามารถที่จะเปิดอ่านได้ การเปิดอ่านนั้นต้องใช้กุญแจส่วนตัว ซึ่งกุญแจส่วนตัวจะไม่มีการส่งไปในโครงข่ายสื่อสาร ทำให้ผู้ดักฟังต้องใช้เวลานานในการสุ่มหาค่ากุญแจส่วนตัว

2). ผู้ดักฟังสุ่มหากุญแจปกปิดและกุญแจส่วนตัว โดยหาจุดอ่อนจากวิธีการสร้างกุญแจรหัสลับและลอกเลียนวิธีการดังกล่าว สุ่มหาไปเรื่อย ๆ จนกว่าจะพบค่ากุญแจรหัสลับ การสุ่มหาจะทำได้ยากเมื่อเราสร้างจากแหล่งกำเนิดตัวเลขสุ่มที่มีช่วงเวลาการเกิดของสัญญาณไม่แน่นอน เช่น สัญญาณการคลิกส์เม้าท์ และสัญญาณการกดแป้นพิมพ์ เป็นต้น

3). ผู้ดักฟังสังเกตการเปลี่ยนแปลงของบิตข้อมูลจากเอกสารรหัสลับ วิธีการนี้ต้องใช้พละกำลัง (Brute force attack) ในการสุ่มเพื่อแปลงกลับมาเป็นเอกสารปกติ ซึ่งอัลกอริทึมที่ใช้ในการเข้ารหัสลับเอกสารที่นำเสนอนี้ใช้อัลกอริทึมของ RC5 จากการที่สามารถเพิ่มวงรอบของการคำนวณให้สูงขึ้น จะมีผลทำให้ความปลอดภัยสูงขึ้นตามวงรอบที่เพิ่ม

4). ความเร็วในการคำนวณทางคอมพิวเตอร์เพิ่มขึ้นทุกปี ทำให้ผู้ดักฟังใช้เวลาในการบุกรุกน้อยลง เราจำเป็นต้องเพิ่มกุญแจรหัสลับให้ใหญ่ขึ้นและเปลี่ยนแปลงให้บ่อยขึ้นด้วย

บทที่ 7
บทสรุปและผลการทดลอง

7.1 สรุปผลการวิจัย

การรับส่งข้อมูลบนโครงข่ายเวิลด์ดาต้าเมื่อเรานำเทคนิคการบีบอัดข้อมูลและการเข้ารหัสลับข้อมูลมาใช้ จะทำให้เราได้ปริมาณการส่งข้อมูลและมีความปลอดภัยในข้อมูลที่รับส่งมากขึ้น กล่าวคือเทคนิคการบีบอัดข้อมูลด้วยอัลกอริทึมของ LZP มีการลดขนาดข้อมูลใน 2 ขั้นตอนคือ ลดความซ้ำซ้อนและลดจำนวนบิตของข้อมูล ทำให้อัลกอริทึมของ LZP นี้ได้รับอัตราการบีบอัดข้อมูล (Compression ratio) ที่ดีโดยเฉลี่ยอยู่ที่ 3.06 บิตต่อไบต์ โดยยังได้ความเร็วในการประมวลผลโดยเฉลี่ยเท่ากับ 303 กิโลไบต์ต่อวินาที นอกจากนี้การใช้ตารางเปรียบเทียบข้อมูลแบบแฮชซึ่งจะช่วยประหยัดหน่วยความจำที่ใช้เก็บลงอย่างมาก ทั้งนี้ขนาดของตารางเปรียบเทียบข้อมูลสามารถกำหนดได้ตามต้องการ ซึ่งขึ้นอยู่กับขนาดของข้อมูลที่จะทำการบีบอัดหรือขยายขนาดข้อมูล ดังเช่นเมื่อนำมาใช้งานบนโครงข่ายเวิลด์ดาต้าขนาดของเฟรมข้อมูลที่ใช้รับส่งระหว่างกันจะไม่ใหญ่มาก ทำให้การจองเนื้อที่หน่วยความจำน้อยลงโดยเลือกใช้ตารางข้อมูลขนาด 11 บิต หรือหน่วยความจำขนาด 4 กิโลไบต์ ดังนั้นเมื่อใช้อัลกอริทึมการบีบอัดข้อมูลของ LZP บนโครงข่ายเวิลด์ดาต้าแล้วจะทำให้ประสิทธิภาพการรับส่งข้อมูลเพิ่มขึ้น 76 เปอร์เซ็นต์ ในขณะที่อัลกอริทึมของ LZW เพิ่มขึ้น 27 เปอร์เซ็นต์ ส่วนทางด้านความเร็วในการประมวลผลของอัลกอริทึมการบีบอัดข้อมูลนั้น จะไม่มีผลมากนักที่จะทำให้ประสิทธิภาพของความเร็วที่ได้รับในการรับส่งข้อมูลด้อยลงไป

และเมื่อพิจารณาถึงระบบของการเข้ารหัสลับแบบผสม RC5 และ RSA แล้ว วิธีการสร้างกุญแจรหัสลับ และการแลกเปลี่ยนกุญแจรหัสลับ จะให้ความปลอดภัยในการใช้งาน และการรับส่งเอกสารรหัสลับที่มีขนาดใหญ่ จะให้ความเร็วในการประมวลผล ซึ่งวิธีการของ RC5 จะใช้วิธีการหมุนบิตและสลับบิต นอกจากนี้การประมวลผลสามารถกำหนดได้ตามสถาปัตยกรรมของเครื่องทำให้มีประสิทธิภาพในการคำนวณสูงขึ้น เช่น ใช้งานบนเครื่อง Alpha chip ขนาด 64 บิต เป็นต้น จะให้ความเร็วเพิ่มขึ้นเป็นเท่าตัวเมื่อเทียบกับเครื่องขนาด 32 บิต และวิธีการของ RSA จะให้ความปลอดภัยสูงในการแลกเปลี่ยนกุญแจรหัสลับ เนื่องจากการทำลายกุญแจรหัสลับจากอัลกอริทึมของ RSA จะขึ้นอยู่กับการแยกตัวประกอบ (Factoring) n เป็นสำคัญ ถ้าให้ค่าของ n มีค่ามากแล้ว ทำให้

วิธีการแยกตัวประกอบ n เพื่อหาค่า d ทำได้ไม่ง่ายนัก ส่วนใหญ่ผู้บุกรุกมุ่งทำลายจุดอ่อนตรงส่วนอื่นแทน เช่น การสุ่มหากุญแจรหัสลับ การเก็บแฟ้มข้อมูลกุญแจรหัสลับ การดักฟังกุญแจรหัสลับในโครงข่ายสื่อสาร และการสุ่มหาบิตของเอกสารปกติเปรียบเทียบกับเอกสารรหัสลับ เป็นต้น

สำหรับการสร้างกุญแจรหัสลับนั้นยิ่งใช้แหล่งกำเนิดตัวเลขสุ่มประกอบกันมากเท่าใดยิ่งปลอดภัยมากขึ้นเท่านั้น และทำให้มีโอกาสคาดเดาน้อยมาก ซึ่งการเลือกใช้แหล่งกำเนิดตัวเลขสุ่มจากอุปกรณ์ภายนอก เช่น สัญญาณจากเม้าท์ หรือแป้นพิมพ์ จะให้ความปลอดภัยสูงกว่าการใช้คุณสมบัติเฉพาะที่มีมากับระบบ เช่น คอนฟิกกูเรชั่นไฟล์ พารามิเตอร์แสดงสภาวะแวดล้อมของระบบ หรือตัวเลขสุ่มที่เป็นฟังก์ชั่นคณิตศาสตร์ในภาษา C เป็นต้น

7.2 ปัญหาที่พบและข้อเสนอแนะ

การรับส่งข้อมูลระหว่างกันบนโครงข่ายเวิลด์ดาต้านั้น ถึงแม้โปรโตคอลที่ใช้จะมีการป้องกันการสูญหายของข้อมูลไว้แล้ว แต่จากการทดสอบเมื่อมีการส่งข้อมูลอย่างต่อเนื่องติดต่อกัน พบว่ามีข้อมูลสูญหายบางเฟรม ทั้งนี้เนื่องจากโมเด็มไร้สายมีบัฟเฟอร์จำกัด หรือสูญหายระหว่างทางโดยที่ผู้รับไม่ได้รับข้อมูล ดังนั้นวิธีการแก้ปัญหานี้ทำได้โดยผู้ส่งจะมีการตรวจสอบเฟรมตอบสนอง (Response) จากโมเด็มไร้สายทุกครั้งเมื่อมีการส่งเฟรมคำสั่ง (Command) และผู้รับจะมีการร้องขอข้อมูลใหม่ (Retransmission) เมื่อมีลำดับข้อมูลเกิดการสูญหาย การทำเช่นนี้จะทำให้ปริมาณการส่งข้อมูล (Throughput) ลดลงไป

การบีบอัดข้อมูลบนโครงข่ายสื่อสาร (Communications compression) ต้องเป็นแบบทันทีทันใด (Real-time) และต้องลดกระบวนการปฏิบัติอัลกอริทึมให้เกิดการสูญเสียเวลา (Delay) น้อยที่สุดเท่าที่จะทำได้ เนื่องจากถ้ามี Overhead มากเกินไปแทนที่จะลดเวลาการรับส่งข้อมูล กลับเพิ่มปัญหาทำให้ล่าช้ากว่าการที่ไม่ทำการบีบอัดข้อมูลเสียอีก ส่วนใหญ่แล้วถ้าโครงข่ายความเร็วสูงตั้งแต่ 64 กิโลบิตขึ้นไป และมีข้อมูลไม่หนาแน่นมากนักบนโครงข่าย มักจะไม่มีขบวนการบีบอัดข้อมูล อย่างไรก็ตามเรามักจะพบขบวนการบีบอัดข้อมูลบนโครงข่ายสื่อสารความเร็วต่ำ ที่มีข้อจำกัดเรื่องเทคโนโลยี และการลงทุนค่าอุปกรณ์โครงข่ายที่สูงเมื่อต้องเพิ่มความเร็วให้สูงขึ้น สำหรับโครงข่ายเวิลด์ดาต้านั้นจะมีการใช้แบนด์วิตร่วมกันที่ความเร็ว 19.2 กิโลบิตต่อวินาที ดังนั้นการรับส่งข้อมูลในแต่ละครั้งไม่สามารถรับประกันความเร็วได้ ขึ้นอยู่กับว่าช่วงเวลานั้น ๆ มีแบนด์วิตว่างใช้งานมากน้อยเพียงไร ซึ่งปริมาณการส่งข้อมูลโดยเฉลี่ยมีเพียง 81.5 ไบต์ต่อวินาทีเท่านั้น นอกจากนี้การล่าช้าของข้อมูลยังขึ้นอยู่กับการส่งข้อมูลซ้ำเมื่อเกิดการสูญหาย ระบบการประมวลผลของอุปกรณ์บนโครงข่ายแต่ละตัว การเข้าแถวคอย (Queueing) ของการรับส่งข้อมูล เป็นต้น ดังนั้นถ้าอัลกอริทึมการบีบอัดข้อมูลมีอัตราการบีบอัดข้อมูลมาก ๆ แล้ว จะช่วยทำให้แบนด์วิตว่างมากขึ้น

ซึ่งถ้าจะมีการพัฒนาต่อไป จึงควรพิจารณาประเด็นนี้เป็นสำคัญ แต่ต้องไม่มีผลทำให้ความเร็วในการประมวลผลลดลงไป

ความเร็วของการคำนวณทางคอมพิวเตอร์ที่ได้รับการพัฒนาขีดความสามารถให้เพิ่มขึ้นในแต่ละปี จะมีผลทำให้ประสิทธิภาพของอัลกอริทึมที่ใช้ในการเข้ารหัสลับด้อยลงไป การเพิ่มขนาดกุญแจรหัสลับให้ใหญ่ขึ้น หรือกำหนดเวลาหมดอายุ (Expire date) ของกุญแจรหัสลับให้สั้นลง ดูจะเป็นหนทางการแก้ปัญหาอีกทางหนึ่ง และการแลกเปลี่ยนกุญแจรหัสลับที่ใช้ในวิทยานิพนธ์นี้ยังไม่ได้ป้องกันการปลอมแปลงไว้ คือ ถ้ามีบุคคลใดสามารถบุกรุก แล้วลักลอบเปลี่ยนแปลงกุญแจสาธารณะของผู้อื่นให้เป็นของตนเอง ก็สามารถแอบอ้างเพื่อรับกุญแจปกปิดได้ การแก้ปัญหานี้สามารถทำได้โดยมีองค์กรกลางออกใบรับรอง (Certificate Authority) ให้กับผู้เป็นเจ้าของกุญแจสาธารณะที่แท้จริง เพื่อสามารถตรวจสอบความเป็นเจ้าของได้ นอกจากนี้การมีระบบจัดเก็บกุญแจรหัสลับไม่ดีพอ ก็อาจจะตกเป็นเป้าหมายในการจู่โจมเพื่อทำลายความปลอดภัยได้ ดังนั้นควรจะมีวิธีการป้องกันที่ดีพอ เช่นมีการใช้ Password เข้าระบบ และเก็บกุญแจรหัสลับไว้บนอุปกรณ์จัดเก็บข้อมูลที่ไม่สามารถเข้าได้จากโครงข่ายสื่อสาร

อัลกอริทึมการบีบอัดข้อมูลและการเข้ารหัสลับข้อมูลที่ทำการวิจัยนี้ สามารถนำไปประยุกต์ใช้กับการรับส่งข้อมูลในระบบไร้สาย (Wireless data communication) อื่น ๆ ได้ เช่น Wireless LAN เป็นต้น และสามารถนำไปประยุกต์ใช้งานบนโครงข่ายที่มีสาย (Landline) ได้ เช่นระบบอินเตอร์เน็ต เป็นต้น

บรรณานุกรม

[1] ปรีชา วีระอาชากุล, "ระบบสื่อสารข้อมูลเคลื่อนที่ในเครือข่ายเซลลูลาร์ AMPS", หนังสือ IT Management , VOL.45, June 1995.

[2] Dan Stanescu, "Protocol and Programming for WorldData Network", Microsoft Developers Conference, June 1995.

[3] Gordon White, MCGI, Ceng, MIEE, "Mobile Radio Technology", Gordon White, 1994.

[4] Motorola, "Open Protocol Specifications: Native Control Language Release 1.2", Canada, April 1994.

[5] Motorola, "Open Protocol Specifications: DataTAC Messaging Release 1.0", Canada, November 1995.

[6] Motorola, "DataTAC 5000 System Release 4.0: System Description", Canada, May 1994.

[7] Motorola, "DataTAC 5000 System Release 4.0: Host Application Programmer's Manual", Canada, April 1994.

[8] Motorola, "RPM Transparent Mode Interface R1.0: Reference Manual", Canada, June 1993.

[9] Motorola, "RPM Native Mode Interface R1.1: Reference Manual", Canada, June 1993.

[10] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," IEEE Trans. Inform. Theory, vol. IT-23, no.3, pp. 337-343, May 1977.

[11] Charles Bloom, "LZP: A new Data Compression Algorithm", Proceedings of the IEEE Data Compression Conference (DCC) 96, 1996.

[12] T.A. Welch, "A technique for high performance data compression," IEEE Comput., vol. 17, pp.8-19, June 1984.

[13] The Calgary Compression Corpus, available by FTP from: ftp.cpsc.ucalgary.ca/pub/projects/text.compression.corpus/

[14] Mark Nelson, "The Data Compression Book", Prentice Hall, 1991.

[15] Charles Bloom, "New Techniques in Context Modeling and Arithmetic Encoding", Proceedings of the IEEE Data Compression Conference (DCC) 96, 1996.

[16] Ian Goldberg and David Wagner, "Randomness and the Netscape Browser", Dr. Dobbs Journal, January 1996.

[17] Bruce Schneier, "Applied Cryptography Protocols, Algorithms and Source Code in C", Second Edition, John Wiley & Sons Inc., 1996.

[18] Ronald L. Rivest, "The RC5 Encryption Algorithm", MIT Laboratory for Computer Science, Cambridge U.S.A, 1994.

[19] Baldwin, R.W., and Rivest R.L., "Draft: The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms", RSA Data Security Inc., March 1996.

[20] R. Rivest, "RFC1321: The MD5 Message-Digest Algorithm", Network Working Group, April 1992.

[21] Janne Frosen, "Practical Cryptosystems and their Strength", Department of Computer Science Helsinki University of Technology, 1996.

[22] Kenneth H. Rosen, "Discrete Mathematics and its applications", Second Edition, McGraw-Hill, Inc., 1991.

[23] ดร.พิสิษฐ์ ชาญเกียรติก้อง, "รหัสลับและการรักษาความปลอดภัยของข้อมูลกุญแจสู่ยุคแห่งสังคมข่าวสารที่แท้จริง", วารสารทางวิชาการสื่อสารโทรคมนาคม, ปีที่ 2 ฉบับที่ 6, หน้า 9-29, ตุลาคม 2537.

[24] M. Shand, and J. Vuillemin, "Fast Implementations of RSA Cryptography", Digital Equipment Corp., Paris Research Laboratary (PRL), France, 1993.

[25] RSA Data Security, Inc., "RSA's Frequently Asked Questions About Today's Cryptograph", Web Page, http://www.rsa.com/.

ภาคผนวก ก.
ผลงานวิจัยที่ได้รับการตีพิมพ์

[1] รังสรรค์ น้อยจินดา, เกษตร์ ศิริสันติสัมฤทธิ์, "การติดต่อสื่อสารวิทยุติดตามตัวโดยใช้โมเด็มไร้สายบนโครงข่ายเวิลด์ดาต้า", วารสารการประชุมทางวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 18, หน้า 234-239, พฤศจิกายน 2538.

[2] รังสรรค์ น้อยจินดา, เกษตร์ ศิริสันติสัมฤทธิ์, "ระบบการเข้ารหัสลับแบบผสม RC5 และ RSA สำหรับการสื่อสารข้อมูล", วิศวสารลาดกระบัง, ปีที่ 14 ฉบับที่ 1, หน้า 44-51, เมษายน 2541.

การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 18

การติดต่อสื่อสารวิทยุติดตามตัวโดยใช้โมเด็มไร้สายบนโครงข่ายเวิลด์ดาต้า

(Pager Communication Using Wireless Modem on WorldData Network)

เกษตร์ ศิริสันติสัมฤทธิ์ * , รังสรรค์ น้อยจินดา **

บทคัดย่อ

การติดต่อสื่อสารวิทยุติดตามตัวโดยใช้โมเด็มไร้สาย ที่เสนอในบทความนี้เป็นตัวอย่างหนึ่งของระบบการสื่อสารเคลื่อนที่ภายใต้โครงข่ายเวิลด์ดาต้า ที่มีการรับส่งข้อมูลในแบบแพคเก็ตสวิทชิ่ง ด้วยความเร็ว 19.2 กิโลบิตต่อวินาที และใช้ช่องสัญญาณความถี่วิทยุในย่าน 800 เมกะเฮริท์ โดยใช้โปรโตคอล NCL ที่เป็นภาษาควบคุมการแลกเปลี่ยนข้อมูลแบบอะซิงโครนัส ในการติดต่อระหว่างอุปกรณ์เทอร์มินอลและโมเด็มคลื่นวิทยุ เพื่อส่งข้อมูลไปที่โครงข่ายเวิลด์ดาต้า ต่อจากนั้นอุปกรณ์เกตเวย์จะทำหน้าที่คัดเลือกแพ็คเก็ตข้อมูลส่งไปที่คอมพิวเตอร์หลักด้วยโปรโตคอล SCR ที่เป็นภาษาควบคุมเส้นทางข้อมูลแบบเส้นทางเดียวระหว่างคอมพิวเตอร์หลักและระบบ DataTAC ให้อุปกรณ์เทอร์มินอลใช้เส้นทางเดียวกัน บนโครงข่ายการเชื่อมโยงแบบ X.25 และคอมพิวเตอร์หลักนี้จะทำการส่งข้อความที่ต้องการไปที่อุปกรณ์ส่งข้อมูลวิทยุติดตามตัว เพื่อกระจายข้อมูลไปตามหมายเลขวิทยุติดตามตัวที่ระบุไว้

Abstract

This is a proposal for a pager communication system using wireless modem. The pager communication system is an example of the mobile communication on WorldData network. The data communicates by utilizing the packet switching technology at a speed of 19.2 kbps and radio frequency of 800 MHz. The protocol utilized to transfer information between the mobile terminal and the radio packet modem is known as NCL which is an asynchronus transaction. Once the information is sent to the WorldData network, a gateway device will receive the data packet and send it to the host computer through the SCR protocol - one logical connection between the host and the DataTAC system for a number of a terminal device, on X.25 data layer link. The host computer will then transfer the information to the pager communication system which will distribute the information to the specific pager identification number.

1. บทนำ

การติดต่อสื่อสารด้วยวิทยุติดตามตัวในปัจจุบัน จะผ่านวงจรอัตโนมัติหรือโอเปอร์เรเตอร์ของศูนย์วิทยุติดตามตัวทำหน้าที่ส่งข้อความไปที่เครื่องลูกข่ายต่าง ๆ ซึ่งการใช้โอเปอร์เตอร์จำนวนมาก จะทำให้สิ้นเปลืองค่าใช้จ่ายค่อนข้างสูง แต่ถ้านำระบบที่นำเสนอในบทความนี้มาช่วยเสริม จะแบ่งเบาภาระการลงทุนของศูนย์ไปได้ เมื่อเรานำระบบนี้ไปอยู่ในองค์กรที่มีส่วนเกี่ยวข้องกับการรายงานข้อมูล ทำให้องค์กรนั้น ๆ สามารถรายงานข้อมูลเองได้ด้วยความรวดเร็วตามต้องการ เช่น ใช้รายงานการซื้อขายหลักทรัพย์ของลูกค้า และแจ้งข่าวสารกับพนักงานปฏิบัติการภายในองค์กร เป็นต้น สำหรับการสื่อสารข้อมูลโดยใช้โมเด็มไร้สายผ่านโครงข่ายเวิลด์ดาต้านี้ สามารถนำไปประยุกต์ใช้กับงาน อื่น ๆ ได้มากมาย เช่น การเชื่อมเข้ากับโครงข่ายอินเตอร์เน็ต การรับส่งอิเล็กทรอนิกส์เมล์ การส่งแฟ็กซ์ การรายงานตำแหน่งยานพาหนะ การรายงานสภาพอากาศและการจราจร เป็นต้น

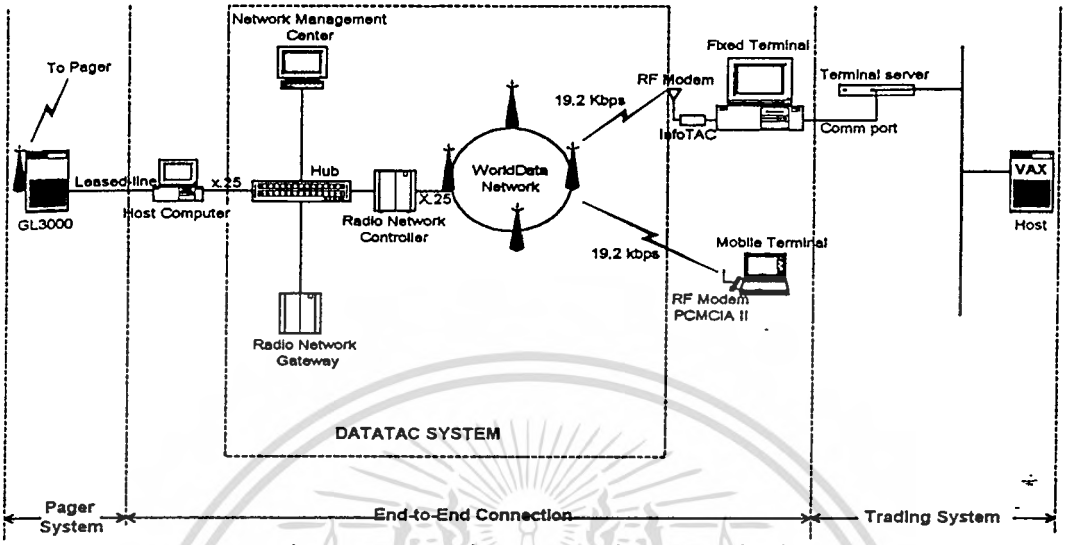
การสื่อสารข้อมูลแบบเคลื่อนที่ ในอนาคตมีการคาดการณ์กันว่าในอีก 5 ปีข้างหน้าจะแซงหน้าการใช้งานโทรศัพท์เคลื่อนที่ มีการกล่าวไว้ใน [3] และ [4] แต่ทั้งนี้ทั้งนั้นก็ต้องคำนึงถึงความพร้อมทางด้านโครงข่ายบริการจะต้องให้ครอบคลุมทั่วประเทศมากที่สุด และทางด้านซอฟแวร์จะต้องมีซอฟแวร์พัฒนาขึ้นมาให้เหมาะกับการใช้งานด้วย ซึ่งการวิจัยนี้จะมุ่งเน้นในส่วนการพัฒนาซอฟแวร์การติดต่อสื่อสารระหว่างอุปกรณ์รับส่งข้อมูลเคลื่อนที่ (Mobile terminal) และคอมพิวเตอร์หลัก (Host computer) เพื่อจะได้เป็นแนวทางในการศึกษาและวิจัยต่อไป

2.รูปแบบของการเชื่อมต่อระบบการสื่อสารข้อมูลเคลื่อนที่

การรับส่งข้อมูลผ่านโมเด็มไร้สายในบทความนี้ได้ประยุกต์ใช้กับการส่งข้อความจากเทอร์มินอลที่มาจากโอเปอร์เรเตอร์คีย์ข้อมูลเข้าไป และข้อความที่มาจากระบบซื้อขายหลักทรัพย์ (Trading System) เป็นข้อมูลการซื้อขายหุ้นของลูกค้า เพื่อที่จะส่งข้อความไปยังวิทยุติดตามตัวตามหมายเลขที่ระบุไว้ สามารถที่จะแสดงการเชื่อมต่อระบบการสื่อสารข้อมูลเคลื่อนที่ได้ดังในรูปที่ 1

* ผู้ช่วยศาสตราจารย์ ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม คณะวิศวกรรมศาสตร์ สจล.

** นักศึกษาปริญญาโท คณะวิศวกรรมศาสตร์ สจล.



รูปที่1 แสดงรูปแบบการเชื่อมต่อระบบการสื่อสารข้อมูลเคลื่อนที่

รายละเอียดของส่วนต่าง ๆในรูปที่ 1 สามารถแบ่งได้ดังต่อไปนี้คือ

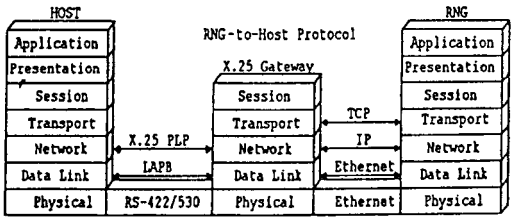
2.1 ส่วนของระบบโครงข่ายเวิลด์คาต้า (World Data Network)

ระบบที่ใช้นี้เป็นระบบ DataTAC 5000 ของโมโตโรล่า ที่สามารถขยายขนาดเน็ตเวิร์คได้จำนวนมาก และมีความยืดหยุ่นในการขยายการเชื่อมต่อกับคอมพิวเตอร์หลัก (Host computer) และสามารถกำหนดขอบเขตพื้นที่ (coverage area) ของการรับส่งข้อมูลที่ต้องการได้ โดยระบบดังกล่าวจะมีศูนย์สั่งการและควบคุมโครงข่าย (Network Management Center) เป็นเครื่องมือใช้ในการเก็บข้อมูลระบบโครงข่ายทั้งหมดและควบคุมฟังก์ชันในการทำงานของระบบโครงข่ายให้เป็นไปตามความต้องการ

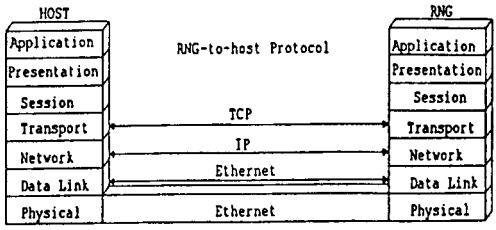
สำหรับอุปกรณ์ที่ทำหน้าที่เชื่อมโยงระบบสื่อสารข้อมูลจะประกอบไปด้วย อุปกรณ์เกตเวย์โครงข่ายคลื่นวิทยุ (Radio Network Gateway), อุปกรณ์ควบคุมโครงข่ายคลื่นวิทยุ (Radio Network Controller) และอุปกรณ์เชื่อมโยงกับระบบสื่อสารข้อมูลอื่นๆ (Communication Hub) ซึ่งอุปกรณ์เหล่านี้มีหน้าที่รับผิดชอบการสวิทชิ่งข้อมูลและการทำฟังก์ชันเราท์ดิ้ง (routing function), ดูแลการเชื่อมต่อของคอมพิวเตอร์หลักเข้ากับโครงข่ายเวิลด์คาต้าโดยใช้โปรโตคอลมาตรฐาน TCP/IP หรือ X.25, เก็บข้อมูลลงทะเบียนของลูกค้าเพื่อกำหนดสิทธิการใช้งานในระบบ โดยมีการจัดการเกี่ยวกับการควบคุมการทำโรมมิ่ง (Roaming), ควบคุมสถานีฐาน (Base Station), ดูแลการใช้งานและบันทึกจำนวนข้อมูลข่าวสารที่ใช้งานอยู่บนระบบ และการทำบิลลิ่ง (Billing) เพื่อจะเก็บเงินตามปริมาณการใช้งานของลูกค้าบนโครงข่าย สิ่งเหล่านี้เป็นหน้าที่ของระบบโครงข่ายเวิลด์คาต้าจะไม่ขอ กล่าวในรายละเอียดมากนัก

2.2ส่วนของการติดต่อคอมพิวเตอร์หลักกับคอมพิวเตอร์ในระบบ DataTAC (Host Computer to Radio Network Gateway : RNG)

การติดต่อของระบบในส่วนนี้เป็นการติดต่อระหว่างศูนย์คอมพิวเตอร์เกตเวย์ของคาต้าแทค (DataTAC) กับคอมพิวเตอร์หลัก (Host Computer) ซึ่งอาจจะต่อกันในโครงข่ายภายในหรือต่อระยะไกลไปยังคอมพิวเตอร์อื่นๆได้โดยใช้วงจรเช่า (Leased-line) , PSDN และ X.25 เป็นต้น โดยใช้โปรโตคอลมาตรฐานที่เป็นยอมรับกันทั่วโลก คือ TCP/IP หรือ X.25 ในรูปที่ 2 แสดงการเชื่อมต่อโปรโตคอลแบบ X.25 และ TCP/IP ของคอมพิวเตอร์หลักและเกตเวย์



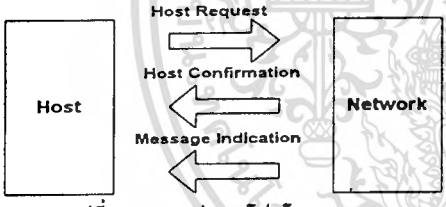
(a) การเชื่อมต่อโปรโตคอลแบบ X.25



(b) การเชื่อมต่อโปรโตคอลแบบ TCP/IP

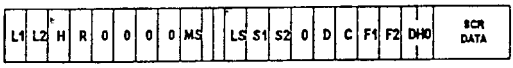
รูปที่ 2 แสดงการเชื่อมต่อโปรโตคอลแบบ X.25 และ TCP/IP ของคอมพิวเตอร์หลักและเกตเวย์

นอกจากนี้ยังมีการใช้โปรโตคอล SCR (Standard Context Routing) เพื่อใช้ในการควบคุมเซสชั่น (session) การเชื่อมโยงของอุปกรณ์การรับส่งข้อมูลเคลื่อนที่ (mobile terminal) กับคอมพิวเตอร์หลัก (Host Computer) ซึ่งอยู่บนเลเยอร์ของ X.25 หรือ TCP/IP รูปแบบของโปรโตคอล SCR มี 3 รูปแบบด้วยกัน คือ Host Request, Host Confirmation และ Message Indication ดังแสดงในรูปที่ 3



รูปที่ 3 แสดงรูปแบบโปรโตคอล SCR

1) Host Request คือ คอมพิวเตอร์หลักร้องขอไปยังโครงข่ายเพื่อที่จะทำการส่งข้อมูลออกไปยังอุปกรณ์สื่อสารเคลื่อนที่ต่าง ๆ เช่น การส่งอิเล็กทรอนิกส์เมล์ เป็นต้น โดยมีรูปแบบเฟรมที่ส่งออกไปดังแสดงในรูปที่ 4



รูปที่ 4 แสดงรูปแบบเฟรมของ Host Request

ในแต่ละฟิลด์มีรายละเอียดดังนี้คือ

- L1 และ L2 คือความยาวข้อมูล (Byte Length) มีขนาด 2 ไบท์
- H และ R คือชนิดของเฟรมข้อมูล (SCR message type) มีขนาด 2 ไบท์ โดยย่อมา จาก Host Request
- MS ถึง LS คือค่า Link Logical Identification (LLI) มีขนาด 4 ไบท์ เป็นการระบุตำแหน่งที่อยู่ของอุปกรณ์ปลายทาง สามารถระบุตำแหน่งเฉพาะบุคคลและเป็นกลุ่มๆ ได้
- S1 และ S2 คือค่า Save byte มีขนาด 2 ไบท์ แสดงหมายเลขป้ายประกาศ (tag) หรือ Sequence number ของเฟรมนั่นเอง เพื่อบอกให้อุปกรณ์โครงข่ายตอบสนองกลับมาด้วย Sequence number เดียวกัน
- D คือ delivery option มีขนาด 1 ไบท์ เป็นตัวเลือกในการส่งข้อมูลว่าส่งครั้งเดียวออกไปเลย หรือว่าส่งแล้วรอจนกระทั่งถึงผู้รับหรือมีค่า Timeout
- C คือ โหมดการตอบกลับ (Confirm mode) มีขนาด 1 ไบท์ เป็นการเลือกให้มีการตอบกลับหรือไม่มีการตอบกลับ ถ้ามีการตอบกลับจะดูว่ามาจากอุปกรณ์เทอร์มินอลหรือโครงข่าย
- F1 และ F2 คือ Format byte มีขนาด 2 ไบท์ ถูกสงวนไว้ใช้กับ RD-LAP Format
- DHO คือ Data Header Offset มีขนาด 2 ไบท์ เป็นตัวบอกข่าวสารเส้นทางของโครงข่าย (network routing information) ที่กำหนดโดย user ว่าจะส่งไปยังอุปกรณ์โครงข่ายใด
- SCR DATA คือข้อมูลที่ใช้รับส่ง มีขนาดไม่เกิน 2048 ไบท์ โดยมี 2 ไบท์แรกเป็นตัวกำหนด Session ในการรับส่งข้อมูล

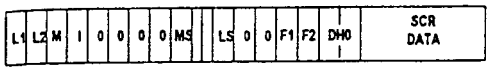
2) Host Confirmation คือข้อมูลที่ตอบสนองจากโครงข่ายเมื่อคอมพิวเตอร์หลักทำการร้องขอมา เช่นแสดงถึงเครื่องผู้ใช้ไม่อยู่ในพื้นที่บริการ หรือไม่มีสัญญาณตอบรับจากเครื่องผู้ใช้ เป็นต้น แสดงเป็นรูปแบบเฟรมได้ดังรูปที่ 5

รูปที่ 5 แสดงรูปแบบเฟรมของ Host Confirmation

ในแต่ละฟิลด์มีรายละเอียดดังนี้คือ

- L1และL2 คือความยาวข้อมูล(Byte Length) มีขนาด 2 ไบท์
- H และ C คือชนิดของเฟรมข้อมูล (SCR message type) มีขนาด 2 ไบท์ ย่อมาจาก Host Confirmation
- MS ถึง LS คือค่า Link Logical Idenfication (LLI) มีขนาด 4 ไบท์ เป็นค่าที่ได้มาจากเฟรม Host Request เพื่อตอบสนองกลับไปยังคอมพิวเตอร์หลัก
- S1 ถึง S2 คือค่า Save byte มีขนาด 2 ไบท์ ที่ได้มาจากเฟรม Host Request เพื่อตอบสนองกลับไปยังคอมพิวเตอร์หลักด้วยค่า Sequence number ที่เหมือนกัน
- R1 และ R2 คือตัวอักขระของ ACK หรือ NACK มีขนาด 2 ไบท์ เป็นตัวบอกถึงสถานะของอุปกรณ์สื่อสารเคลื่อนที่ว่าปกติ หรือไม่ปกติอย่างไร

3) Message Indication คือรูปแบบเฟรมที่แสดงถึงข้อมูลที่รับมาจากเทอร์มินอลเคลื่อนที่ (Mobile Terminal) และข้อมูลที่ได้จะถูกคัดเลือกเอาเฉพาะส่วนที่ต้องการส่งต่อไปยังเครื่องส่งวิทยุติดตามตัว แสดงเป็นรูปแบบเฟรมดังในรูปที่ 6

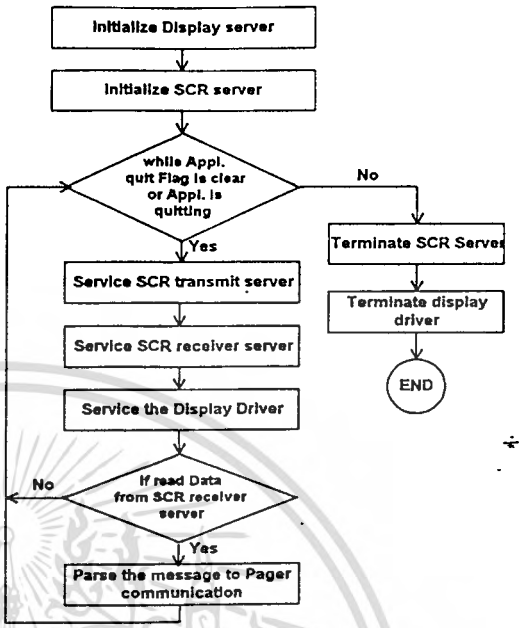


รูปที่ 6 แสดงรูปแบบเฟรมของ Message Indication

ในแต่ละฟิลด์มีรายละเอียดดังนี้คือ

- L1 และ L2 คือความยาวของข้อมูล (Byte Length) มีขนาด 2 ไบท์
- M และ I คือชนิดของเฟรมข้อมูล (SCR message type) ย่อมาจาก Message Indication
- MS ถึง LS คือค่า Link Logical Idenfication (LLI) มีขนาด 4 ไบท์ เป็นค่าที่ได้มาจากเฟรม Host Request เพื่อตอบสนองกลับไปยังคอมพิวเตอร์หลัก
- F1 และ F2 คือรูปแบบข้อมูล (Format byte) มีขนาด 2 ไบท์
- DH0 คือ Data Header Offset มีขนาด 2 ไบท์ เป็นตัวบอกข่าวสารเส้นทางของโครงข่าย(network routing information) ที่กำหนดโดย user ว่าจะส่งไปยังอุปกรณ์โครงข่ายใด
- SCR DATA คือข้อมูลที่ใช้รับส่ง มีขนาดไม่เกิน 2048 ไบท์ โดยมี 2 ไบท์แรกเป็นตัวกำหนด Session ในการรับส่งข้อมูล

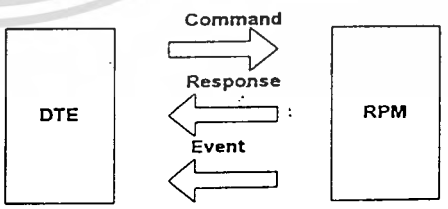
อัลกอริทึมของโปรแกรมบนคอมพิวเตอร์หลักเพื่อใช้ติดต่อกับอุปกรณ์เกตเวย์ สามารถแสดงเป็นไดอะแกรมได้ดังรูปที่ 7 ในส่วนนี้คอมพิวเตอร์หลัก (Host computer) ที่ใช้เป็น PC มี X.25 Card ติดตั้งอยู่ด้วยและต้องโหลด X.25 Driver ก่อนที่จะรันโปรแกรมใช้งานจากรูป ที่ 7 เริ่มต้นการทำงานจะมีการทำฟังก์ชั่นเกี่ยวกับการแสดงผลทางจอภาพ แล้วมาทำการติดต่อโมเด็มที่ต่ออยู่กับเครื่องส่งวิทยุติดตามตัว อีกส่วนหนึ่งจะทำหน้าที่ตั้งค่าพารามิเตอร์ต่าง ๆ ในการใช้ฟังก์ชั่นของ X.25 และทำการเปิดเซสชั่น (session) เพื่อให้อุปกรณ์เกตเวย์ขอติดต่อเข้ามา ถ้าไม่มีการติดต่อเข้ามาภายในเวลาที่กำหนด (timeout) จะเริ่มต้นติดต่อกันใหม่ รอจนกว่าอุปกรณ์เกตเวย์ติดต่อเข้ามาได้ และเมื่อมีข้อมูลมาจากอุปกรณ์เกตเวย์เป็นชนิดข้อมูล (message type) "MI" จะมีการคัดเลือกข้อมูลจากฟิลด์ SCR DATA ส่งไปยังเครื่องส่งวิทยุติดตามตัว ต่อจากนั้นโปรแกรมจะทำงานในลักษณะวนรอบต่อไป



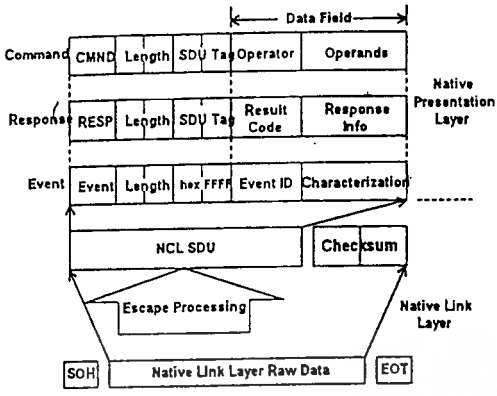
รูปที่ 7 แสดงไดอะแกรมส่วนโปรแกรมบนคอมพิวเตอร์หลัก

2.3 ส่วนการติดต่ออุปกรณ์เทอร์มินอลและโมเด็มคลื่นวิทยุ (Terminal : DTE to Radio Packet Modem :RPM)

ส่วนนี้เป็นส่วนที่เครื่องผู้ใช้ทำการส่งข้อความไปที่คอมพิวเตอร์หลักโดยผ่านโมเด็มไร้สายบนช่องสัญญาณความถี่คลื่นวิทยุ 800 เมกะเฮริท์ โดยใช้โปรโตคอล NCL (Native Control Language) ซึ่งมีคุณสมบัติในด้าน Transaction-oriented ที่เหมาะกับการสื่อสารแบบ Connectionless ในรูปที่ 8 แสดงรูปแบบของโปรโตคอล NCL และในรูปที่ 9 แสดงรูปแบบเฟรม NCL



รูปที่ 8 แสดงรูปแบบของโปรโตคอล NCL



รูปที่ 9 แสดงรูปแบบเฟรม NCL

จากรูปที่ 9 ในชั้นการเชื่อมโยง (Native Link Layer) จะใช้อักขระ SOH และ EOT ทำหน้าที่ซิงค์โครไนซ์เฟรมเมื่อไปถึงคอมพิวเตอร์ปลายทาง แสดงให้รู้ถึงจุดเริ่มต้นและสิ้นสุดเฟรม เนื่องจากข้อมูลที่ส่งออกไปจะเป็นลักษณะ Stream Data ถ้ามีการส่งข้อมูลแบบไบนารี (Binary Data) จะต้องใช้อักขระพิเศษ escaping (DLE:0x20) เพื่อทำหน้าที่เป็นตัวบอกชั้นการเชื่อมโยง (Link Layer) ให้แยกแยะข้อมูลเวลาแปลคำสั่ง NCL สำหรับฟิลด์ Checksum มีขนาด 16 บิท จะคำนวณข้อมูลที่รับมาและนำมาเปรียบเทียบกับค่าในฟิลด์ Checksum ว่าถูกต้องหรือไม่ การคำนวณค่า Checksum นี้จะไม่รวมอักขระ escaping และฟิลด์ NCL SDU (NCL Service Data Unit) จะประกอบไปด้วยชนิดของเฟรม 3 รูปแบบด้วยกันคือ คำสั่ง (Command) สถานะการตอบสนอง (Response) และรายงานเหตุการณ์ (Event) ซึ่งฟิลด์นี้จะอยู่ใน Presentation Layer

หน่วยข้อมูลที่ส่งระหว่าง DTE และ RPM เรียกว่า Service Data Unit ทำหน้าที่เกี่ยวกับการรายงานเหตุการณ์ต่าง ๆ จาก RPM รวมถึงการตั้งค่าพารามิเตอร์และควบคุมการทำงานของ RPM ในเมื่อมีการส่งผ่านข่าวสารจาก RPM ไปยังระบบโครงข่าย ในแต่ละเฟรม SDU จะมีถูกตัดทอนออกเป็นหน่วยข้อมูลที่เรียกว่า Protocol Data Unit (PDU) หลาย ๆ แพ็กเก็ตส่งออกไป ทางผู้รับจะทำการจัดเรียงแพ็คเก็ตให้เป็น เฟรม SDU อีกครั้งหนึ่ง ในการส่งแพ็คเก็ต PDU แต่ละครั้งจะมีการใส่แอดเดรสต้นทางของ RPM ลงไปอย่างอัตโนมัติ โดยที่แอดเดรสของ RPM นี้จะมีเพียงแอดเดรสเดียวไม่ซ้ำกับใคร รายละเอียดของฟิลด์ SDU มีดังนี้ คือ

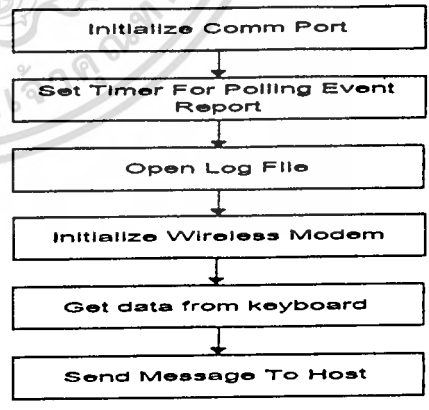
1) ฟิลด์ชนิดเฟรม (Type Field) มีขนาด 8 บิท จะแสดงว่าเป็นชนิดคำสั่ง, การตอบสนอง หรือว่า รายงานเหตุการณ์

2) ฟิลด์ความยาว (Length Field) มีขนาด 16 บิท แสดงจำนวนของข้อมูลภายในฟิลด์ข้อมูล (Data field) ถ้ามีตัวอักขระ escaping จะไม่ถูกนับให้อยู่ในฟิลด์แสดงความยาวข้อมูล โดยที่ตัวอักขระ escaping จะถูกตัดออกไปเวลาแปลคำสั่ง NCL และถ้ามี flow control ภายในเฟรมด้วยก็จะไม่มีการนับจำนวนความยาวในฟิลด์ด้วยเช่นกัน

3) ฟิลด์ป้ายประกาศข้อมูล (SDU Tag Feild) มีขนาด 16 บิท แสดงหมายเลขลำดับ (Sequence number) ในเฟรม SDU ของการรับส่งข้อมูลระหว่างอุปกรณ์ DTE และ RPM เมื่อมีการส่งเฟรมคำสั่งจาก DTE ไปที่ RPM จะมีการประหมายเลขลำดับไปในเฟรมด้วย อุปกรณ์ RPM ก็จะมีการตอบสนองด้วยหมายเลขลำดับเดียวกันกับเฟรมคำสั่งที่ส่งมา ซึ่งเป็นไปได้อาจจะมีเฟรมรายงานเหตุการณ์ส่งไปก่อนเฟรมตอบสนองได้ สำหรับเฟรมรายงานเหตุการณ์จะมีหมายเลขลำดับเป็น Hex FFFF เพื่อรายงานเหตุการณ์ต่างๆ เช่น สภาพของแบตเตอรี่ และความแรงของสัญญาณ เป็นต้น ดังนั้นเมื่อมีข้อมูลมาที่อุปกรณ์ DTE ก็สามารถที่จะแยกแยะข้อมูลได้ว่าเป็นของเฟรมตอบสนองหรือว่าเฟรมรายงานเหตุการณ์ โดยมี SDU Tag เป็นตัวบอก

4) ฟิลด์ข้อมูล (Data Field) มีขนาดตามที่ผู้ใช้กำหนด แต่ต้องไม่มากกว่า 2048 ไบท์ ซึ่งรูปแบบของฟิลด์ข้อมูลนี้จะถูกกำหนดโดยฟิลด์ชนิดเฟรม (Type Field)

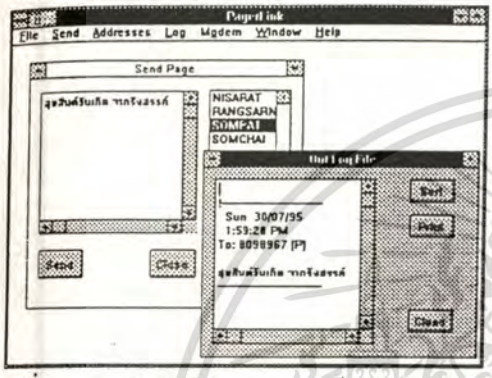
อัลกอริทึมการเชื่อมต่ออุปกรณ์เทอร์มินอล และโมเด็มคลื่นวิทยุ แสดงเป็นไดอะแกรมได้ดังรูปที่ 10



รูปที่ 10 แสดงไดอะแกรมส่วนโปรแกรมการเชื่อมต่อเทอร์มินอลและโมเด็มคลื่นวิทยุ

ลักษณะการทำงานส่วนโปรแกรมในรูปที่ 10 มีลักษณะไม่ซับซ้อนมากนัก โดยการทำงานจะเริ่มต้นที่การเซ็ทอัพความเร็ว (Speed) และหมายเลขพอร์ต (Port number) หลังจากนั้นตั้งค่าเวลาสำหรับการ ตรวจสอบสถานะของโมเด็มไร้สายเกี่ยวกับ

สภาพของแบตเตอรี่และตรวจสอบความแรงของสัญญาน เป็นต้น นอกจากนั้นยังมีการเปิดไฟล์บันทึกข้อมูล เพื่อจะได้ตรวจสอบข้อมูลย้อนหลังได้ ข้อมูลจะรับโดยคีย์บอร์ดหรือจากระบบซื้อขายหลักทรัพย์ หลังจากนั้นข้อมูลจะถูกประกอบเข้าเป็นเฟรมแบบ SDU เพื่อส่งให้โมเด็มไร้สายส่งต่อไปยังโครงข่ายเวิลด์คาต้าต่อไป ในรูปที่ 11 แสดงตัวอย่างโปรแกรมการส่งข้อความไปยังวิทยุติดตามตัวบนโครงข่ายเวิลด์คาต้า



รูปที่ 11 แสดงตัวอย่างโปรแกรมการส่งข้อความไปยังวิทยุติดตามตัวบนโครงข่ายเวิลด์คาต้า

2.4 ส่วนการส่งข้อมูลซื้อขายหุ้นไปยังอุปกรณ์เทอร์มินอลสื่อสารเคลื่อนที่

ในส่วนนี้ข้อมูลจะมาจากระบบซื้อขายหลักทรัพย์ (Trading System) ของนายหน้าซื้อขายหลักทรัพย์ (Broker) เป็นการรายงานการซื้อขายหุ้นของลูกค้าเช่น หุ้นที่มีการสั่งซื้อหรือขายได้รับการแมทชิ่งกับทางตลาดหลักทรัพย์ (SET) และเมื่อมีการยกเลิกข้อมูลซื้อขายก็จะมีการรายงานมาให้ทราบทางวิทยุติดตามตัว โดยลักษณะการเชื่อมต่อของอุปกรณ์เทอร์มินอลเคลื่อนที่จะผ่านทางพอร์ตอนุกรม (Comm port) ไปเข้าเทอร์มินอลเซิฟเวอร์ (Terminal Server) ที่ทำหน้าที่เป็นพอร์ตสื่อสารแบบอะซิงโครนัส เพื่อรับข้อมูลจากระบบซื้อขายหลักทรัพย์ ลักษณะข้อมูลที่รับโดยอุปกรณ์เทอร์มินอลเคลื่อนที่จะประกอบไปด้วยข้อมูลหุ้นและหมายเลขลูกค้า ก่อนการส่งข้อมูลออกไปจะมีการตรวจสอบหมายเลขลูกค้ากับหมายเลขวิทยุติดตามตัวของลูกค้า ว่ามีในตาราง (table) หรือไม่ ถ้ามีในตารางก็จะเก็บข้อมูลลง logfile ก่อน ต่อจากนั้นก็จะทำการแปลงเป็นข้อความที่ต้องการ ส่งผ่านโมเด็มไร้สายไปที่โครงข่ายเวิลด์คาต้าต่อไป

2.5 ส่วนของการติดต่อคอมพิวเตอร์หลักไปยังเครื่องส่งข้อความวิทยุติดตามตัว

การเชื่อมต่อตรงส่วนนี้ใช้โปรโตคอลพื้นฐานแบบอะซิงโครนัสผ่าน RS232C ของเครื่องคอมพิวเตอร์หลักซึ่งใช้ PC ผ่านวงจรเช่า (Leased-line) ไปที่เครื่องส่งวิทยุติดตามตัว (GL3000) ก่อนการส่งข้อความต้องทำการแปลงให้อยู่ในรูปแบบของอุปกรณ์ GL3000 เสียก่อน ต่อจากนั้นอุปกรณ์ GL3000 จะทำหน้าที่ส่งข้อความไปที่วิทยุติดตามตัวที่ระบุไว้ตามต้องการ โดยจะไม่ขอกล่าวรายละเอียดในส่วนนี้มากนัก

3. บทสรุป

ระบบการสื่อสารข้อมูลระหว่างอุปกรณ์สื่อสารข้อมูลเคลื่อนที่และคอมพิวเตอร์หลักนั้น ในส่วนของการใช้โปรโตคอล NCL ที่เสนอในบทความนี้มีข้อจำกัด คือ ไม่สามารถจะเปิดโปรแกรมหลาย ๆ ตัวในเวลาเดียวกัน โดยใช้โมเด็มตัวเดียวกันได้ พูดได้ว่าหนึ่งโปรแกรมต่อหนึ่งโมเด็ม แต่ถ้าเรามีการพัฒนา โปรโตคอล NCL ให้เป็นลักษณะ transport protocol ทำหน้าที่จัดการให้โปรแกรมเปิดได้หลาย ๆ session ในเวลาเดียวกันโดยที่เราจะต้องเขียน Driver ตรงส่วน transport นี้ขึ้นมา หรือว่าถ้าเราจะใช้โปรโตคอล TCP/IP ที่เรียกว่า Mobile Internet Protocol ที่มีคุณสมบัติด้าน transport อยู่แล้วโดยมีการติดต่อผ่าน TCP/IP Socket interface บน MS Windows ที่เรียกว่า winsock ก็สามารถแก้ปัญหานี้ได้ นอกจากนั้นในการรับส่งข้อมูลนั้นยังจะต้องนำข้อมูลมาเข้ารหัสเสียก่อน เพื่อป้องกันการลักลอบข้อมูลที่สำคัญ แล้วจึงทำการถอดรหัสข้อมูลที่คอมพิวเตอร์ปลายทาง ตลอดจนมีการบีบอัดข้อมูลก่อนที่จะส่ง เพื่อเพิ่มประสิทธิภาพความรวดเร็วในการรับส่งข้อมูลและยังช่วยลดปริมาณข้อมูลภายในโครงข่ายเวิลด์คาต้าด้วย ซึ่งสิ่งต่าง ๆ เหล่านี้จะได้มีการพัฒนาต่อไป

4. เอกสารอ้างอิง

[1] Motorola, "*Open Protocol Specifications : Native Control Language Release 1.2*", Canada, April 1994.

[2] Dan Stanescu, "*Protocol and Programming for WorldData Network*", Microsoft Developers Conference, June 1995.

[3] ปรีชา วีระอาชากุล, "*ระบบสื่อสารข้อมูลเคลื่อนที่ในเครือข่ายเซลลูลาร์ AMPS*", หนังสือ IT Management , VOL.45, June 1995.

[4] คนึงจิตร วิจิตรปิยะกุล, "*ธุรกิจสื่อสารไร้สายในทศวรรษ 90*", วารสารทางวิชาการสื่อสารโทรคมนาคม, ปีที่ 2 ฉบับที่ 6, เดือนตุลาคม 2537

ระบบการเข้ารหัสลับแบบผสม RC5 และ RSA สำหรับการสื่อสารข้อมูล

(Combining of RC5 and RSA Cryptosystems for Data Communications)

รังสรรค์ น้อยจินดา เกษตร์ ศิริสันติสัมฤทธิ์

ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

การเข้ารหัสลับแบบ RC5 นำมาใช้ในการเข้ารหัสลับกับข้อมูลที่มีขนาดใหญ่ โดยใช้วิธีการเข้ารหัสลับแบบบล็อกชนิดซิมเมทริกซ์ มีการแบ่งข้อมูลออกเป็นบล็อก ๆ ละเท่า ๆ กัน สามารถกำหนดขนาดของบล็อกได้ตามสถาปัตยกรรมของเครื่องคอมพิวเตอร์ เช่น 16 บิต, 32 บิต และ 64 บิต เป็นต้น ซึ่งมีผลต่อประสิทธิภาพในการประมวลผล ทั้งนี้ระดับความปลอดภัยยังสามารถกำหนดได้จากความยาวของกุญแจรหัสลับและวงรอบของการคำนวณ วิธีการแบบนี้กุญแจรหัสลับของการเข้ารหัสและถอดรหัสจะใช้กุญแจรหัสลับเดียวกัน ดังนั้นขั้นตอนการส่งกุญแจรหัสลับไปให้ฝ่ายตรงข้ามที่สื่อสารกันอาจจะถูกดักฟังได้ บทความนี้ได้นำวิธีการของการเข้ารหัสลับแบบ RSA มาใช้ในขั้นตอนของการแลกเปลี่ยนกุญแจปกปิดของ RC5 โดยที่สมาชิกแต่ละคนจะต้องมีกุญแจ 2 ชนิด คือ กุญแจส่วนตัวและกุญแจสาธารณะ ซึ่งเราสามารถเปิดเผยกุญแจสาธารณะให้ใครก็ได้ที่จะส่งเอกสารมาหาตน และเราถอดรหัสลับเอกสารนั้นด้วยกุญแจส่วนตัว ทำให้การติดต่อสื่อสารระหว่างกันมีความปลอดภัยในข้อมูลมากขึ้น นอกจากนั้นยังได้นำเสนอวิธีการสร้างกุญแจรหัสลับของ RC5 และ RSA โดยใช้ตัวเลขสุ่มจากแหล่งกำเนิดของช่วงเวลาการคลิกส์เม้าท์ ช่วงเวลาการกดแป้นพิมพ์ เวลาของระบบ และสัญญาณนาฬิกา มาประกอบกัน และนำผลลัพธ์ที่ได้มาเข้าฟังก์ชั่นไดเจสของเอกสาร (MD5) เพื่อให้กลุ่มของบิตข้อมูลเป็นอิสระซึ่งกันและกัน ทำให้ผู้บุกรุกคาดเดาและทำลายกุญแจรหัสลับได้ยากขึ้น

Abstract

RC5 (Rivest Cipher) encryption with a symmetric block cipher is suitable for large data sizes. The plaintext and ciphertext are fixed-length bit blocks which can be utilized by processors of different word-lengths. This is because a 16-bit, 32-bit, or a 64-bit processors are capable of effectively increasing computation speed. Furthermore, RC5 has a variable-length secret key and a variable number of rounds which provide flexibility in its security level.

The RC5 method utilizes one cryptographic key for encryption and decryption. The disadvantage is that anyone who knows the key can decrypt all the encrypted messages using that one key. Therefore, we propose to use the RSA encryption method for key exchange. In the RSA (Rivest-Shamir-Adleman) method, each person has a pair of keys. One key is a public key, and the other one is a private key. The public key will be available for all to see, but the private key will be kept by each person. We can only decrypt by a private key. This method provides a highly secure communication channel.

Additionally, we propose the RC5 and RSA key generator are used random number sources from mouse click timing, keystroke timing, time and clock. The random output runs it through Message Digest (MD5) algorithm to produce the bits are all independent. It is very difficult to guess and break the key by attackers.

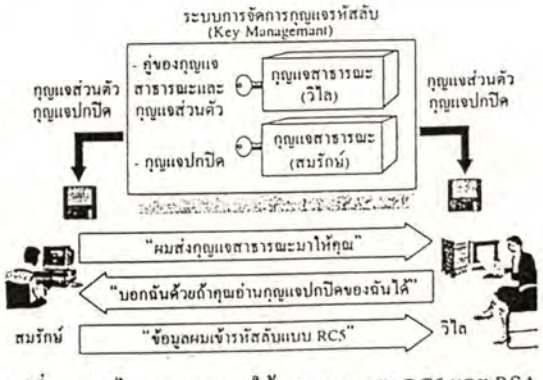
Ladkrabang Engineering Journal, *Vol. 14, No. 1, April 1998*

1. บทนำ

ระบบการเข้ารหัสลับ (Cryptosystem) คือ การนำเอกสารปกติ (Plaintext) มาเข้ารหัสลับ (Encrypt) จะได้เอกสารที่ถูกแปลงไปจากเดิมเรียกว่า เอกสารรหัสลับ (Ciphertext) เมื่อเราต้องการอ่านเอกสารเราต้องมีกุญแจรหัสลับ (Key) สำหรับถอดรหัสลับ (Decrypt) ให้เป็นเอกสารปกติเพื่อให้สามารถอ่านได้ วิธีการเข้ารหัสลับมีอยู่ 2 วิธีการด้วยกันคือ การเข้ารหัสลับชนิดซิมเมทริกซ์ (Symmetric cryptosystem) และการเข้ารหัสลับชนิดอะซิมเมทริกซ์ (Asymmetric cryptosystem) ทั้ง 2 วิธีมีความแตกต่างกัน คือ การเข้ารหัสลับชนิดซิมเมทริกซ์จะใช้กุญแจรหัสลับเดียวกันในการเข้ารหัสลับและถอดรหัสลับ ที่เราเรียกว่าเป็นระบบกุญแจปกปิด (Secret key) ในการรับส่งข้อมูลต้องมีการแลกเปลี่ยนกุญแจรหัสลับระหว่างกัน เป็นไปได้ว่าอาจจะถูกดักฟังกุญแจรหัสลับจากบุคคลอื่นที่อยู่บนโครงข่ายสื่อสารเดียวกัน ฉะนั้นควรจะมีการเปลี่ยนแปลงกุญแจรหัสลับบ่อยๆ สำหรับความปลอดภัย วิธีการนี้สามารถให้ความเร็วในการเข้ารหัสลับและถอดรหัสลับสูงกว่าแบบอะซิมเมทริกซ์ ตัวอย่างอัลกอริทึมที่ใช้ในการเข้ารหัสลับแบบนี้ เช่น DES (Data Encryption Standard) Skipjack และ RC5 (Rivest Cipher) เป็นต้น สำหรับการเข้ารหัสลับชนิดอะซิมเมทริกซ์แต่ละคนจะมีคู่กุญแจสาธารณะ(Public key) และกุญแจส่วนตัว (Private key) โดยที่กุญแจสาธารณะจะเปิดเผยต่อบุคคลใดก็ได้ ในขณะที่กุญแจส่วนตัวจะถูกปกปิดเป็นความลับ ดังนั้นการแลกเปลี่ยนข้อมูลของวิธีนี้จะมีเฉพาะกุญแจสาธารณะเท่านั้นอยู่ในโครงข่ายสื่อสาร ตัวอย่างอัลกอริทึมของวิธีนี้ เช่น Diffie-Hellman DSS(Digital Signature Standard) และ RSA(Rivest-Shamir-Adleman) เป็นต้น

วิธีการสร้างกุญแจรหัสลับ จะอาศัยแหล่งกำเนิดตัวเลขสุ่ม (Random source) ที่มาจากแหล่งกำเนิดทางซอฟต์แวร์และฮาร์ดแวร์ เราสามารถแบ่งออกเป็น 2 แบบ คือ ตัวเลขสุ่มจริง (True random number) และตัวเลขสุ่มเทียม (Pseudo random number) ปัญหาที่พบกับการสร้างกุญแจรหัสลับทั่ว ๆ ไป เกิดมาจากการใช้ตัวเลขสุ่มที่คาดเดาได้ง่าย เช่น การใช้แต่เพียงฟังก์ชั่นคณิตศาสตร์ srand() หรือ rand() ในภาษา C หรือการใช้ตัวเลขสุ่มแต่เพียงเวลาของระบบและหมายเลขแสดงกระบวนการทำงานของโปรแกรม (Process ID) ใน Netscape browser เวอร์ชั่น 1.1 บนระบบยูนิกซ์ ได้มีผู้ทำลายกุญแจรหัสลับมาแล้ว มีกล่าวไว้ใน [1] ดังนั้นในบทความนี้ได้พัฒนาวิธีการสร้างกุญแจรหัสลับโดยใช้ตัวเลขสุ่มทั้งสองแบบประกอบกัน คือเลือกแหล่งกำเนิดจากตัวเลขสุ่มจริงที่มีลักษณะต่าง ๆ กัน เช่น ช่วงเวลาการคลิกส์เม้าท์ ช่วงเวลาการกดแป้นพิมพ์ เวลาของระบบ และสัญญาณนาฬิกา เป็นต้น นำมาเรียงลำดับกันเป็นบัฟเฟอร์ข้อมูล (Seed buffer) เพื่อใช้เป็นจุดเริ่มต้นของขบวนการกำเนิดตัวเลขสุ่มเทียม โดยนำมาผ่านกลไกฟังก์ชั่นไดเจสของเอกสาร(Message Digest หรือ MD5) ทำซ้ำ ๆ ไปมาหลาย ๆ รอบตามสัญญาณที่เกิดขึ้นจากแหล่งกำเนิดตัวเลขสุ่มจริง เพื่อทำให้ความสัมพันธ์ของกลุ่มบิตที่เกิดขึ้นยุ่งยากและซับซ้อนในเชิงสถิติ และนำผลลัพธ์จากตัวเลขสุ่มเทียมสร้างเป็นกุญแจรหัสลับ

กุญแจรหัสลับทั้งหมดที่สร้างขึ้นประกอบด้วยกุญแจสาธารณะ กุญแจส่วนตัว และกุญแจปกปิด โดยเราสามารถเปิดเผยกุญแจสาธารณะได้ สำหรับกุญแจส่วนตัวจะต้องเก็บเป็นความลับ ส่วนกุญแจปกปิด ไม่สมควรอย่างยิ่งที่จะส่งไปในโครงข่ายสื่อสารให้กับสมาชิก โดยไม่มีการปกปิดใดๆ บทความนี้ได้นำเสนอวิธีการแลกเปลี่ยนกุญแจปกปิด ด้วยการเข้ารหัสลับแบบ RSA เพื่อป้องกันการดักฟังจากผู้อื่นเมื่อสมาชิกอีกฝ่ายหนึ่งเปิดอ่านกุญแจปกปิดได้แล้ว การส่งข้อมูลระหว่างกันจะใช้ระบบการเข้ารหัสลับแบบ RC5 ทั้งนี้ การสร้าง การเก็บ และการแลกเปลี่ยนกุญแจรหัสลับ รวมกันเรียกว่า ระบบการจัดการกุญแจรหัสลับ สามารถเขียนเป็นไดอะแกรมการใช้งานแบบผสม RC5 และ RSA ดังรูปที่ 1



รูปที่1 แสดงไดอะแกรมการใช้งานแบบผสม RC5 และ RSA

2. อัลกอริทึมการเข้ารหัสลับแบบ RC5

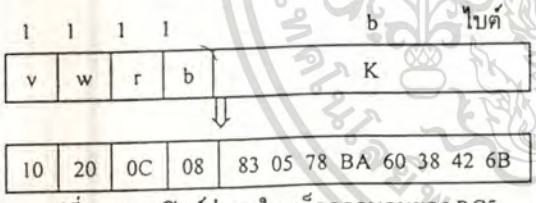
RC5 มีความยืดหยุ่นในการกำหนดระดับความปลอดภัย จึงได้กำหนดรูปแบบมาตรฐานขึ้นมาคือ RC5-w/r/b โดยที่พารามิเตอร์แต่ละตัวสามารถอธิบายได้ดังนี้

w คือ ขนาดข้อมูลที่เก็บได้ (word size) มีหน่วยเป็นบิต สามารถมีค่าได้ตามสถาปัตยกรรมของเครื่องคอมพิวเตอร์ เช่น 16 บิต, 32 บิต และ 64 บิต เป็นต้น ในการเข้ารหัสจะใช้ขนาดบล็อกๆ ละ 2w ยกตัวอย่างเช่น ถ้าเราใช้ w เท่ากับ 32 บิต จะได้ขนาดบล็อก 2x32 = 64 บิต หรือ 8 ตัวอักษร เป็นต้น

r คือ จำนวนวงรอบ (round) ของการเลือกระดับความปลอดภัย มีค่าตั้งแต่ 0 ถึง 255

b คือ ขนาดของกุญแจปกปิด (Secret key) มีหน่วยเป็นไบต์ มีค่าได้ตั้งแต่ 0 ถึง 255 ไบต์

เวลาใช้งานการกำหนดพารามิเตอร์เหล่านี้จะถูกรวมไว้ด้วยกันเป็นบล็อก (Block) ประกอบด้วยฟิลด์ต่างๆ ดังในรูปที่ 2



รูปที่ 2 แสดงฟิลด์ต่างๆ ในบล็อกควบคุมของ RC5

จากรูปที่ 2 จะได้ว่า RC5 เวอร์ชั่น (v) 1.0 มีขนาด (w) เท่ากับ 32, วงรอบ (r) เท่ากับ 12 รอบ และมีขนาดของกุญแจปกปิด (b) เท่ากับ 8 ไบต์ (64 บิต) คือมีกุญแจปกปิด (k) เริ่มจาก 83 05 .... 6B ซึ่งการจัดการกุญแจรหัสลับนี้ จะกระทำในบล็อกควบคุม และถูกจัดเก็บไว้เป็นไฟล์ข้อมูลอย่างปกปิด สำหรับส่งไปยังบุคคลที่ต้องการติดต่อระหว่างกันในโครงข่ายสื่อสาร

RC5 ประกอบด้วยส่วนสำคัญ 3 ส่วนด้วยกัน คือ ส่วนการเข้ารหัสลับ (Encryption), ส่วนการถอดรหัสลับ (Decryption) และส่วนขยายกุญแจรหัสลับ (Key expansion) ในที่นี้จะไม่ขอกล่าวถึงรายละเอียดของอัลกอริทึม RC5 สามารถดูได้ใน [2],[3]

3. อัลกอริทึมการเข้ารหัสลับแบบ RSA

การเข้ารหัสลับแบบ RSA ใช้หลักการของโมดูลัส (Modulus) โดยเลือกค่าจำนวนเฉพาะ (Prime number) ที่มีค่ามาก คือ p และ q เมื่อให้ n = pq และเลือกค่า e ให้น้อยกว่าค่า n โดยค่า e มีความสัมพันธ์กับค่า p และ q. ซึ่งเมื่อหาค่า ห.ร.ม. (Greatest common divisors) ระหว่าง e กับ (p-1)(q-1) แล้วจะเหลือเศษ 1 เราสามารถเขียนสมการของ ห.ร.ม. โดยใช้อัลกอริทึมของ Euclidean [4] ดังนี้

$$\gcd(e, (p-1)(q-1)) = 1 \qquad (1)$$

3.1 การเข้ารหัสลับ

ข้อมูลที่จะเข้ารหัสลับจะแปลงให้อยู่ในรูปของตัวเลข เพื่อสะดวกในการคำนวณ อาจจะเป็นข้อมูลไบต์เดียวหรือจะรวมกันเป็นบล็อกก็ได้ ให้ M เป็นตัวเลขจำนวนเต็ม (Integer) ของเอกสารปกติ (Plaintext) และให้ C เป็นตัวเลขจำนวนเต็มของเอกสารรหัสลับ (Ciphertext) สามารถเขียนเป็นฟังก์ชันการเข้ารหัสลับ RSA ดังนี้

$$C = M^e \bmod n \qquad (2)$$

เมื่อ (n, e) คือ กุญแจสาธารณะ

ในกรณีที่ค่า M และ e มีค่ามาก ไม่สามารถคำนวณด้วยฟังก์ชั่นคณิตศาสตร์เลขยกกำลังได้ เราต้องแปลงสมการที่ (2) ให้อยู่ในรูปของอินเนอร์เวิร์คกิ้ง (Inner workings) ดังนี้

$$C = M^{3e} \bmod n = ((M^{2e} \bmod n)*(M^{3e} \bmod n)) \bmod n \qquad (3)$$

เราสามารถหาค่า C จากอัลกอริทึมของ Chinese Remainder Theorem [4]

3.2 การถอดรหัสลับ

เอกสารปกติจะถูกถอดรหัสลับได้โดยใช้กุญแจส่วนตัว d ค่า d เป็นส่วนกลับของ e mod (p-1)(q-1) สามารถเขียนความสัมพันธ์ระหว่างค่า d และ e ได้คือ

$$de \equiv 1 \pmod{(p-1)(q-1)} \quad (4)$$

หรือ

$$de \bmod (p-1)(q-1) = 1 \quad (5)$$

จากสมการที่ (4) เรียกว่า (de) Congruent (≡) กับ 1 mod (p-1)(q-1) หรือกล่าวได้ว่า (d*e) - 1 หารด้วย (q-1)(p-1) ลงตัว

ค่า d สามารถหาค่าได้จากส่วนกลับของอัลกอริทึม Euclidean [4]

เราสามารถถอดรหัสลับเอกสารได้จากสมการ

$$P = C^d \bmod n \quad (6)$$

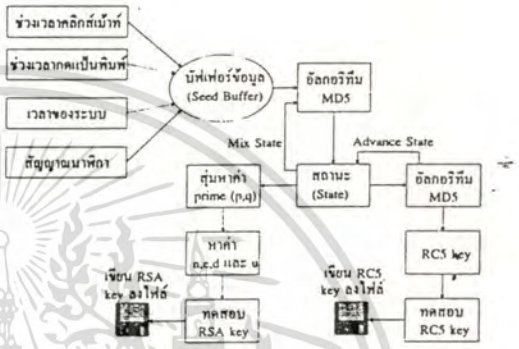
จากสมการที่ (6) ใช้อัลกอริทึมของ Chinese Remainder Theorem[4] หาค่า P เช่นเดียวกับวิธีการเข้ารหัสลับ

เมื่อเราได้ค่า (n, e) ซึ่งเป็นกุญแจสาธารณะ และ d เป็นกุญแจส่วนตัวแล้ว ค่าจำนวนเฉพาะ p และ q จะต้องเก็บเป็นความลับ หรือทำลายในทันที เพราะถ้าคนอื่นรู้ค่า p และ q จะทำให้หาค่า d ได้

4. ระบบการจัดการกุญแจรหัสลับ

4.1 การสร้างกุญแจรหัสลับ

เราสามารถสร้างกุญแจรหัสลับของ RC5 และ RSA ได้โดยเลือกแหล่งกำเนิดตัวเลขสุ่มที่เหมาะสมตามระบบปฏิบัติการ (Operating system) ที่ใช้ การเลือกระบบใดระบบหนึ่งหรือหลายระบบประกอบกันต้องอยู่บนพื้นฐานคือ การเกิดของสัญญาณต้องไม่แน่นอน และบิตข้อมูลที่เกิดขึ้นต้องไม่มีความสัมพันธ์ซึ่งกันและกัน อันจะมีผลทำให้ยากต่อการสุ่มหาค่ากุญแจรหัสลับ แหล่งกำเนิดตัวเลขสุ่มที่ใช้ในบทความนี้คือ ช่วงเวลาการคลิกส์เม้าท์ ช่วงเวลาการกดแป้นพิมพ์ เวลาของระบบและสัญญาณนาฬิกา เป็นต้น แหล่งกำเนิดตัวเลขสุ่มที่กล่าวมานี้ สามารถหาได้จากเครื่องคอมพิวเตอร์ทั่ว ๆ ไป และยังให้สัญญาณที่เกิดขึ้นมีความไม่แน่นอนอีกด้วย ในรูปที่ 3 แสดงไดอะแกรมการสร้างกุญแจรหัสลับของ RC5 และ RSA



รูปที่ 3 แสดงไดอะแกรมการสร้างกุญแจรหัสลับ RC5 และ RSA

ในรูปที่ 3 ขั้นตอนแรกเลือกความยาวรหัสลับตามขนาดที่ต้องการและเลือกบัฟเฟอร์ (Seed Buffer) ขนาด 512 บิต สำหรับเก็บกลุ่มบิตข้อมูลที่ได้มาจากแหล่งกำเนิดตัวเลขสุ่มจริงต่าง ๆ ผลลัพธ์ที่ได้นำมากระทำอัลกอริทึมของ MD5 [5] ซึ่งเป็นขบวนการเข้ารหัสลับข้อมูลอินพุทที่ขนาดความยาวเท่าใดก็ได้ เพื่อให้ได้ข้อมูลเอ้าท์พุทที่มีขนาดคงที่คือ 128 บิต โดยใช้วิธีการสร้างสถานะ (State) ต่อจากนั้นนำสถานะที่ได้มากระทำซ้ำกับ Seed Buffer ตามสัญญาณที่เปลี่ยนแปลงไปของแหล่งกำเนิดจนกระทั่งเท่ากับจำนวนนับตัวเลขสุ่ม (Random counter) ที่กำหนดไว้ และได้ตรงตามขนาดของกุญแจรหัสลับที่เลือกไว้ ผลลัพธ์ที่ได้ส่วนหนึ่งนำไปสร้างเป็นกุญแจสาธารณะและกุญแจส่วนตัวของ RSA โดยนำไปสุ่มหาค่าจำนวนเฉพาะ (p, q) และคำนวนหาค่า n, e, d, u ตามลำดับ อีกส่วนหนึ่งนำไปสร้างเป็นกุญแจปกปิดของ RC5 มีการเพิ่มสถานะการทำงานและการทำอัลกอริทึม MD5 อีกครั้งหนึ่ง เพื่อให้กลุ่มบิตของกุญแจปกปิดมีความซับซ้อนมากขึ้น และให้ยากต่อการทำลายกุญแจปกปิด กุญแจรหัสลับทั้งหมดจะมีการทดสอบด้วยข้อความสั้นๆ เพื่อให้แน่ใจว่ากุญแจรหัสลับสามารถใช้งานได้ถูกต้องก่อนที่จะเก็บลงแฟ้ม

ข้อมูล เขียนเป็นอัลกอริทึมการสร้างกุญแจรหัสลับของ RC5 และ RSA ดังนี้

```
/* ฟังก์ชั่นที่ทำให้กลุ่มบิตข้อมูลเป็นอิสระต่อกัน */
F(X,Y,Z) = XY or not(X) Z
G(X,Y,Z) = XZ or Y not(Z)
H(X,Y,Z) = X xor Y xor Z
I(X,Y,Z)  = Y xor (X or not(Z))
/* วนลูปจนกระทั่งเท่ากับจำนวนตัวเลขสุ่ม */
WHILE until to RandomCounter
  /* เก็บสัญญาณแหล่งกำเนิดตัวเลขสุ่มลงบัฟเฟอร์ */
  SeedBuffer[0] = MouseClick_Timing & 0xFF;
  SeedBuffer[1] = MouseClick_Timing >> 8;
  SeedBuffer[2] = KeyStoke_Timing & 0xFF;
  SeedBuffer[3] = KeyStoke_Timing >> 8;
  SeedBuffer[4] = time() & 0xFF;
  SeedBuffer[5] = time() >> 8;
  SeedBuffer[6] = clock() & 0xFF;
  SeedBuffer[7] = clock() >> 8;
  /* กระทำบล็อกข้อมูลทีละ 16 w.) */
  for i = 0 to N/16-1 do
    for j = 0 to 15 do  /* เก็บบล็อกข้อมูล i ลง X.*/
      X[j] = SeedBuffer[i*16+j];
    AA = A; BB = B; CC = C; DD = D;
    /* ทำไดเจสข้อมูล 4 รอบ รอบละ 16 operations      */
    /* อยู่ภายในฟังก์ชั่น MD5Transform()             */
    /* รอบ 1: กระทำ 16 operations เมื่อ T[i] = T[1,.,16] */
    a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s);
    /* รอบ 2: กระทำ 16 operations เมื่อ T[i]=T[17,.,32] */
    a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s);
    /* รอบ 3: กระทำ 16 operations เมื่อ T[i]=T[33,.,48] */
    a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s);
    /* รอบ 4: กระทำ 16 operations เมื่อ T[i]=T[49,.,64] */
    a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s);
    A = A + AA; B = B + BB; /* เพิ่มค่าสถานะ (state) */
    C = C + CC; D = D + DD; /* เพิ่มค่าสถานะ (state) */
    randpool = ABCD; /* เก็บค่าสถานะลง randpool      */
END WHILE;
/* ส่วนที่ 1: สุ่มหาค่า prime (p,q) เพื่อหา RSA key      */
randomprime( p, pbits , randpool ); /* หา p จาก randpool */
randomprime( q, qbits, randpool ); /* q < p              */
derive_rsakey( n, e, d, p , q, u, ebits ); /* หาค่า n,e,d,u */
status = test_rsakey(); /* ทดสอบ RSA key */
if ( status > 0 )
  /* เขียนแฟ้มข้อมูลกุญแจส่วนตัว */
  writekeyfile(priname, timestamp, userid, n, e, d, p, q, u);
  /* เขียนแฟ้มข้อมูลกุญแจสาธารณะ */
  writekeyfile(pubname, timestamp, userid, n, e, 0, 0, 0, 0);
else
  return( keyfailure );
/* ส่วนที่ 2: หาค่า RC5 key */
MD5Transform( advrandpool, randpool );
/* หาค่ากุญแจปกปิด RC5 */
derive_rc5key( rc5_key, sbits, advrandpool );
status = test_rc5key(); /* ทดสอบ RC5 key */
if ( status > 0 )
  /* เขียนแฟ้มข้อมูลกุญแจปกปิด RC5 */
  writekeyfile(secname,timestamp,userid,v,w,r,b,rc5_key);
else
  return( keyfailure );
```

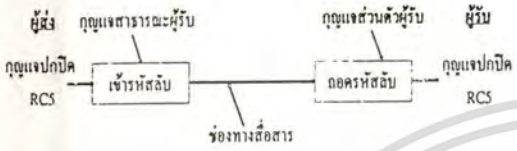
เมื่อ T[i] คือส่วนตารางที่กำหนดค่าคงที่ตัวเลขจำนวนเต็มจากฟังก์ชั่นซายน์ (Sine function) คือ 2^{32} * abs(sin(i)) โดยมี i เป็นรัศมี (Radians) มีค่าเท่ากับ 1,.....,64

<<<s แสดงถึงการเลื่อนตำแหน่งบิตไปทางซ้ายจำนวน s บิต

4.2 การแลกเปลี่ยนกุญแจรหัสลับ

หลักการเบื้องต้นของการแลกเปลี่ยนกุญแจปกปิด RC5 ในโครงข่ายสื่อสาร คือเมื่อผู้ส่งร้องขอการติดต่อไปยังผู้รับ ผู้รับจะจัดการส่งกุญแจสาธารณะกลับมาให้ ผู้ส่งจะนำข้อมูลที่เป็นกุญแจปกปิดมาเข้ารหัสลับด้วยอัลกอริทึม RSA ด้วย

กุญแจสาธารณะที่ผู้รับส่งกลับมา ต่อจากนั้นกุญแจปกปิดที่เข้ารหัสลับไว้ จะถูกส่งไปยังผู้รับ ผู้รับจะใช้กุญแจส่วนตัวเปิดอ่านกุญแจปกปิดออกมา หลังจากนั้นผู้รับก็จะใช้กุญแจปกปิดเข้ารหัสลับข้อมูลที่รับส่งระหว่างกัน ดังรูปที่ 4 แสดงการส่งกุญแจปกปิดในโครงข่ายสื่อสาร



รูปที่ 4 การส่งกุญแจปกปิดในโครงข่ายสื่อสาร

สามารถแสดงขั้นตอนการแลกเปลี่ยนกุญแจรหัสลับระหว่างผู้รับและผู้ส่ง ดังนี้

ขั้นตอนที่ 1 เริ่มต้นโดยผู้ส่งร้องขอการติดต่อไปยังผู้รับด้วยเฟรมข้อมูลที่ประกอบไปด้วย หมายเลขเซสชั่น (Session ID) ตัวเลขสุ่มโทเคน (Token random) เวอร์ชั่นของอัลกอริทึมการแลกเปลี่ยนกุญแจรหัสลับ และระบบกุญแจปกปิดที่ใช้ ตามลำดับ

ขั้นตอนที่ 2 ผู้รับจะทำการตรวจสอบเฟรมข้อมูลผู้ส่งก่อนว่าคุณสมบัติตรงกันหรือไม่ ถ้าตรงกันจะตอบกลับด้วยเฟรมข้อมูลเดียวกันกับผู้ส่ง พร้อมกับส่งกุญแจสาธารณะของผู้รับกลับไปด้วย และเก็บหมายเลขเซสชั่นและตัวเลขสุ่มโทเคนของผู้ส่งไว้ใช้อ้างอิง สำหรับการส่งกลับเฟรมข้อมูลในครั้งต่อไป แต่ถ้าตรวจสอบเฟรมข้อมูลแล้วไม่ตรงกันจะยกเลิกการติดต่อทันที

ขั้นตอนที่ 3 เมื่อผู้ส่งได้รับเฟรมข้อมูลยืนยันการติดต่อและเฟรมข้อมูลกุญแจสาธารณะ ตอบกลับจากผู้รับ จะนำกุญแจปกปิดที่ถูกสร้างมาแล้วจากขั้นตอนการสร้างกุญแจรหัสลับขึ้นมาผสมกับตัวเลขสุ่มโทเคนของผู้ส่งและผู้รับด้วยอัลกอริทึมของ MD5 ได้เป็นกุญแจปกปิดใหม่ ก่อนที่จะเข้ารหัสลับด้วยอัลกอริทึม RSA จากสมการที่ (2) แล้วส่งกุญแจปกปิดใหม่ที่เข้ารหัสลับแล้วไปยังผู้รับ

ขั้นตอนที่ 4 ผู้รับได้รับข้อมูลจากผู้ส่งในขั้นตอนที่ 3 จะใช้กุญแจส่วนตัวถอดรหัสลับที่เป็นข้อมูลกุญแจปกปิดออกมาโดยใช้สมการที่ (6) ต่อจากนั้นผู้รับจะใช้กุญแจปกปิดเข้ารหัสลับข้อมูลส่งกลับไปหาผู้ส่ง ในการแลกเปลี่ยนข้อมูลระหว่างกัน

การกำหนดหมายเลขเซสชั่นของผู้ส่งในครั้งแรกควรจะมีค่าเป็น "0" แต่ถ้ามีค่าไม่เป็น "0" ผู้รับจะตรวจสอบเทียบกับค่าที่เก็บไว้หลังสุด ถ้าตรงกันจะทำการเรียกคืน (Resume) การเชื่อมต่อกันใหม่อีกครั้งหนึ่ง แต่ถ้าไม่ตรงกันจะทำการสร้างหมายเลขเซสชั่นใหม่ขึ้นมา สำหรับตัวเลขสุ่มโทเคนมีไว้เพื่อเปลี่ยนแปลงค่ากุญแจปกปิดใหม่ทุกครั้งเมื่อมีการแลกเปลี่ยนกุญแจรหัสลับ โดยทั้งผู้ส่งและผู้รับจะกำเนิดตัวเลขสุ่มของตัวเอง การกระทำเช่นนี้เป็นการป้องกันผู้ดักฟังเอากุญแจปกปิดไป ซึ่งกุญแจปกปิดที่ผู้ดักฟังได้ไปจะใช้ไม่ได้กับการรับส่งข้อมูลครั้งต่อไป

5. การทดสอบอัลกอริทึมแบบผสม RC5 และ RSA

การทดสอบอัลกอริทึมกระทำบนเครื่องคอมพิวเตอร์ 486 DX4-100 ใช้คอมไพล์เลอร์ Borland C++ 3.1 (16 บิต) ข้อมูลที่ใช้ทดสอบเป็นไบนารี่ไฟล์ขนาด 1.2 Mbytes การทดสอบความเร็วของ RC5 อยู่ภายใต้รูปแบบ RC5-w/r/b โดยกำหนดค่า w คงที่ 32 บิต ให้ r และ b มีค่าเปลี่ยนแปลงไป แล้วเปรียบเทียบความเร็วในการคำนวณ สามารถแสดงการเปรียบเทียบได้จากตารางที่ 1

ตารางที่ 1 แสดงการเปรียบเทียบความเร็วจากการเลือกค่า r และ b ต่างๆ กัน ของ RC5

r (รอบ)	b (บิต)	ความเร็ว (กิโลบิต/ วินาที)
0	512	200
0	128	200
6	128	140
8	128	130
12	40	100
12	128	100
12	256	100
16	128	95
16	256	95

จากผลการทดสอบ RC5 ที่ค่า r และ b ต่าง ๆ กัน จะเห็นว่าจำนวนรอบ (r) จะมีผลต่อความเร็วในการคำนวณมาก แต่เมื่อให้ค่า b เปลี่ยนแปลงไปจะไม่มีผลด้านความเร็วมากนัก ซึ่งการเพิ่มค่า b จะมีผลโดยตรงกับระดับความปลอดภัยที่มากขึ้น ในการเลือกค่า r และ b เท่าใดนั้นเป็นสิ่งสำคัญในด้านความเร็วและความปลอดภัย คือ ถ้าเราต้องการจำนวนรอบน้อยเราต้องใช้ขนาดของกุญแจปกปิดมาก เช่น เราเลือก r เท่ากับ 0 รอบ เราควรเลือก b เท่ากับ 512 บิต ไม่ควรเลือก b เท่ากับ 40 บิต หรือ เราเลือก r เท่ากับ 12 รอบ เราสามารถเลือก b เป็น 40 บิต หรือ 256 บิตก็ได้ เพราะว่าถ้าค่า r และค่า b น้อย ความปลอดภัยจะต่ำไปด้วย ในทางตรงกันข้ามเมื่อค่า b หรือ r มีค่ามากจะช่วยเพิ่มความปลอดภัยให้สูงขึ้นได้

การทดสอบอัลกอริทึมของ RC5 นี้ ต้องการแสดงให้เห็นถึงประสิทธิภาพทางด้านความเร็วในการเข้ารหัสลับแบบ RC5 ซึ่งจะเร็วกว่าอัลกอริทึมแบบ RSA ยกตัวอย่างเปรียบเทียบกันที่กุญแจรหัสลับขนาด 512 บิต จากตารางที่ 1 ของ RC5 จะให้ความเร็วในการคำนวณเท่ากับ 200 กิโลบิตต่อวินาที ในขณะที่ RSA มีความเร็วในการคำนวนเพียง 11 กิโลบิตต่อวินาที โดยทดสอบบนเครื่องเดียวกันและใช้อินพุทไฟล์เดียวกันกับ RC5 เนื่องจาก RSA มีข้อจำกัดเกี่ยวกับความเร็วในการคำนวณ ไม่เหมาะกับการเข้ารหัสลับข้อมูลขนาดใหญ่ จึงนำมาใช้เพื่อการแลกเปลี่ยนกุญแจรหัสลับแทน

การทดสอบความปลอดภัยของ RSA ที่ใช้ในการแลกเปลี่ยนกุญแจรหัสลับ และกุญแจรหัสลับที่สร้างขึ้น ไม่ได้แสดงให้เห็นในที่นี้ เนื่องจากกำหนดวิธีการทดสอบค่อนข้างลำบาก ส่วนใหญ่จะให้ผู้ที่เป็นแฮ็กเกอร์ (Hacker) ทำการทดสอบจุดอ่อนของอัลกอริทึม แต่เมื่อพิจารณาประเด็นความปลอดภัยของ RSA ที่ใช้ในการแลกเปลี่ยนกุญแจรหัสลับแล้วจะพบว่าขึ้นอยู่กับขนาดของกุญแจรหัสลับเป็นสำคัญ ในปัจจุบันถือว่ากุญแจรหัสลับของ RSA ที่มีขนาดใหญ่กว่า 512 บิตขึ้นไป ถือว่าปลอดภัยแล้ว ถึงกระนั้นยังได้มีผู้บุกรุกพยายามคิดค้นหาวิธีการใหม่ ๆ เพื่อทำลายอัลกอริทึมของ RSA ยกตัวอย่างวิธีการที่ผู้บุกรุกทำลายกุญแจรหัสลับ RSA จากสมการ $P = C^d \bmod n$ และรู้กุญแจสาธารณะ (n,e) โดยที่ $n = p*q$ ผู้บุกรุกพยายามที่จะหาค่า d โดยใช้วิธีการแยกตัวประกอบ n ออกมา เพื่อหาค่า p และ q นำไปสู่ค่า d จากการนับหรือสุ่ม สามารถแสดงตัวอย่างอัลกอริทึมการทำลายรหัสลับ RSA ดังนี้

```
p[0] = 1;
c[0] = x;
for i = 0 to (bits in d-1)
   if (bit i of d) = 1 then
         p[i]+1 = (p[i] * c[i]) mod n
   else
         p[i]+1 = p[i];
   d[i]+1 = d[i]^2 mod n;
```

ถ้าเราพิจารณาที่กุญแจรหัสลับขนาด 512 บิต หรือคิดเป็นตัวเลขจำนวนเต็ม(Interger) ที่เก็บได้ในตัวแปรขนาด 512 บิต เท่ากับ 154 หลัก (Digits) เราจะต้องสุ่มหาค่า d ถึง 154 หลัก เลยทีเดียว ซึ่งในปี 1995 Rivest ผู้ร่วมคิดค้นอัลกอริทึม RSA ได้กล่าวไว้ว่าที่ขนาดกุญแจรหัสลับ 384 บิต หรือ 116 หลัก (Digits) ใช้เวลาการทำลายอัลกอริทึมจากอัลกอริทึม Number Field Sieve(NFS) ซึ่งเป็นอัลกอริทึมที่เร็วที่สุดในปัจจุบัน ถึง 400 ปี ของเวลาการคำนวณที่ 1 ล้านคำสั่งต่อวินาทีในหนึ่ง CPU time สามารถแสดงฟังก์ชั่นการแยกตัวประกอบเพื่อหาค่า n ของ NFS ดังนี้

$$H(n) = \exp\left(1.526\,(\ln n)^{1/3}\,(\ln \ln n)^{2/3}\right) \qquad (7)$$

เมื่อพิจารณาถึงวิธีการสร้างกุญแจรหัสลับแล้ว ผู้บุกรุกสามารถที่จะทำลายด้วยวิธีการใช้พละกำลัง (Brute force attack) เพื่อสุ่มหาข้อมูลอินพุท ที่เป็น Seed buffer ไปเรื่อย ๆ โดยใช้แหล่งกำเนิดตัวเลขสุ่มที่ล่วงรู้มา และคาดเดาข้อมูลเอาท์พุทที่ควรจะเป็นไปได้ การใช้วิธีการดังกล่าวนี้ถ้าใช้เครื่องคอมพิวเตอร์ที่มีความเร็วในการคำนวณ 1,000,000,000 คำสั่งต่อวินาที ต้องใช้เวลาถึง 1.07E22 ปี เพื่อหาค่าข้อมูลเอ้าท์พุทออกมา

Ladkrabang Engineering Journal, *Vol. 14, No. 1, April 1998*

6. บทสรุป

เมื่อพิจารณาระบบโดยรวมของการเข้ารหัสลับแบบผสม RC5 และ RSA แล้ว วิธีการสร้างกุญแจรหัสลับและการแลกเปลี่ยนกุญแจรหัสลับ จะให้ความปลอดภัยในการใช้งาน และการรับส่งเอกสารรหัสลับที่มีขนาดใหญ่ จะให้ความเร็วในการประมวลผล กล่าวคือวิธีการของ RC5 จะใช้วิธีการหมุนบิดและสลับบิด นอกจากนี้การประมวลผลสามารถกำหนดได้ตามสถาปัตยกรรมของเครื่อง ทำให้มีประสิทธิภาพในการคำนวณสูง เช่น บนเครื่อง Alpha chip ขนาด 64 บิต เป็นต้น สามารถให้ความเร็วเพิ่มเป็น 2-3 เท่าเลยทีเดียวเมื่อเทียบกับเครื่องขนาด 32 บิต และวิธีการของ RSA จะให้ความปลอดภัยสูงในการแลกเปลี่ยนกุญแจรหัสลับ เนื่องจากการทำลายกุญแจรหัสลับจากอัลกอริทึมของ RSA จะขึ้นอยู่กับการแยกตัวประกอบ (Factoring) n เป็นสำคัญ ถ้าให้ค่าของ n มีค่ามากแล้ว ทำให้วิธีการแยกตัวประกอบ n เพื่อหาค่า d ทำได้ไม่ง่ายเลย ส่วนใหญ่ผู้บุกรุกมุ่งทำลายจุดอ่อนตรงส่วนอื่นแทน เช่น การสุ่มหากุญแจรหัสลับ การเก็บแฟ้มข้อมูลกุญแจรหัสลับ การดักฟังกุญแจรหัสลับในโครงข่ายสื่อสาร และการสุ่มหาบิตของเอกสารปกติเปรียบเทียบกับเอกสารรหัสลับ เป็นต้น จึงควรคำนึงในสิ่งเหล่านี้เพื่อความปลอดภัยที่มากขึ้น นอกจากนี้ความเร็วของการคำนวณทางคอมพิวเตอร์ที่เพิ่มขึ้นในแต่ละปี จะมีผลทำให้ประสิทธิภาพของอัลกอริทึมที่ใช้ในการเข้ารหัสลับด้อยลงไป การเพิ่มขนาดกุญแจรหัสลับให้ใหญ่ขึ้น หรือกำหนดเวลาหมดอายุ (Expire date) ของกุญแจรหัสลับให้สั้นลง ดูจะเป็นหนทางการแก้ปัญหาอีกทางหนึ่ง และการแลกเปลี่ยนกุญแจรหัสลับที่นำเสนอนี้ยังไม่ได้ป้องกันการปลอมแปลงไว้ คือ ถ้ามีบุคคลใดสามารถบุกรุก แล้วลักลอบเปลี่ยนแปลงกุญแจสาธารณะของผู้อื่นให้เป็นของตนเอง ก็สามารถแอบอ้างเพื่อรับกุญแจปกปิดได้ การแก้ปัญหานี้สามารถทำได้โดยมีองค์กรกลางออกใบรับรอง (Certificate Authority) ให้กับผู้เป็นเจ้าของกุญแจสาธารณะที่แท้จริง เพื่อให้สามารถตรวจสอบความเป็นเจ้าของได้

สำหรับการสร้างกุญแจรหัสลับนั้นยิ่งใช้แหล่งกำเนิดตัวเลขสุ่มประกอบกันมากเท่าใดยิ่งปลอดภัยมากเท่านั้น ทำให้มีโอกาสคาดเดาน้อยมาก และการเลือกใช้แหล่งกำเนิดตัวเลขสุ่มจากอุปกรณ์ภายนอก เช่น สัญญาณจากเม้าท์ หรือแป้นพิมพ์ จะให้ความปลอดภัยสูงกว่าการใช้คุณสมบัติเฉพาะที่มีมากับระบบ เช่น คอนฟิกกูเรชั่นไฟล์ หรือพารามิเตอร์แสดงสภาวะแวดล้อมของระบบ เป็นต้น นอกจากนี้การมีระบบจัดเก็บกุญแจรหัสลับไม่ดีพอ ก็อาจจะตกเป็นเป้าหมายในการจู่โจมเพื่อทำลายความปลอดภัยได้ ดังนั้นควรจะมีวิธีการป้องกันที่ดีพอ เช่นมีการใช้ Password เข้าระบบ และเก็บกุญแจรหัสลับไว้บนอุปกรณ์จัดเก็บข้อมูลที่ไม่สามารถเข้าได้จากโครงข่ายสื่อสาร

7. เอกสารอ้างอิง

[1] Ian Goldberg and David Wagner, "Randomness and the Netscape Browser", Dr. Dobbs Journal, January 1996.

[2] Ronald L. Rivest, "The RC5 Encryption Algorithm", MIT Laboratory for Computer Science, Cambridge U.S.A, 1994.

[3] Baldwin, R.W., and Rivest R.L., "Draft: The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms", RSA Data Security Inc., March 1996.

[4] Bruce Schneier, "Applied Cryptography Protocols, Algorithms and Source Code in C", Second Edition, John Wiley & Sons Inc., 1996.

[5] R. Rivest, "RFC1321: The MD5 Message-Digest Algorithm", Network Working Group, April 1992.

[6] Janne Frosen, "Practical Cryptosystems and their Strength", Department of Computer Science Helsinki University of Technology, 1996.

[7] Kenneth H. Rosen, "Discrete Mathematics and its applications", Second Edition, McGraw-Hill, Inc., 1991.

[8] ดร.พิสิษฐ์ ชาญเกียรติก้อง, "รหัสลับและการรักษาความปลอดภัยของข้อมูลกุญแจสู่ยุคแห่งสังคมข่าวสารที่แท้จริง", วารสารทางวิชาการสื่อสารโทรคมนาคม, ปีที่ 2 ฉบับที่ 6, หน้า 9-29, ตุลาคม 2537.

[9] M. Shand, and J. Vuillemin, "Fast Implementations of RSA Cryptography", Digital Equipment Corp., Paris Research Laboratary (PRL), France, 1993.

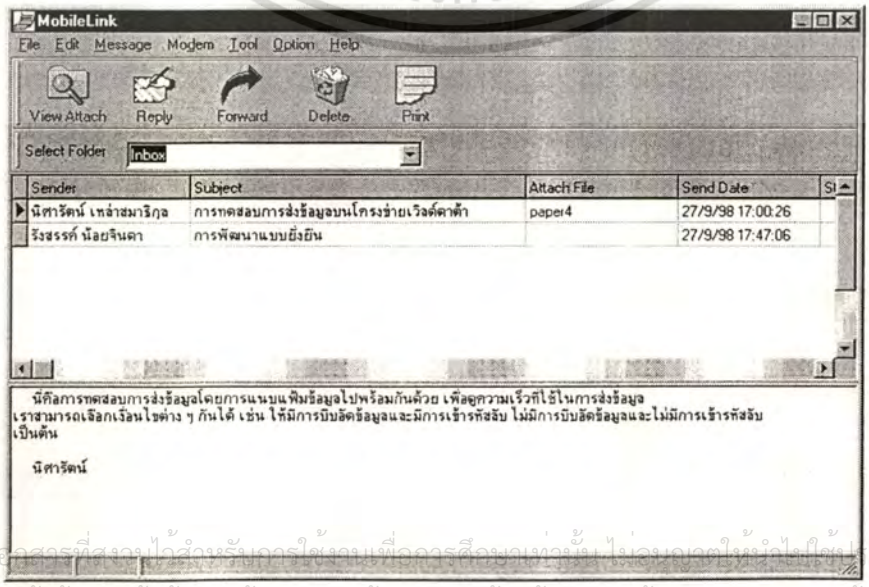
ภาคผนวก ข.

ตัวอย่างการใช้งานโปรแกรม

โปรแกรมสื่อสารข้อมูลระบบไร้สายเคลื่อนที่ดิจิตอล หรือเราเรียกสั้น ๆ ว่า MobileLink ถูกเขียนให้ใช้งานบน Windows 95 ลักษณะของโปรแกรมเป็นแบบรับส่งจดหมายอิเล็กทรอนิกส์ (E-mail) สามารถทำการแนบ (Attach) แฟ้มข้อมูล (File) ที่ต้องการส่งไปพร้อมกันได้ การเรียกโปรแกรมทำโดยใช้เม้าท์ดับเบิ้ลคลิกส์ที่ไอคอนชื่อ MobileLink ดังในรูปที่ ข-1

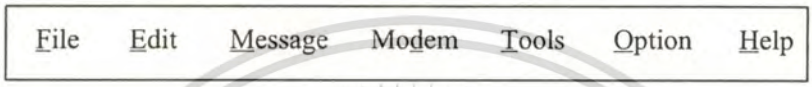
รูปที่ ข-1 แสดงการเรียกโปรแกรม MobileLink จาก Windows 95

รูปที่ ข-2 แสดงหน้าจอแรกเมื่อผู้ใช้งานเริ่มต้นเรียกโปรแกรม MobileLink ใช้งาน



จากรูปที่ ข-2 จะมีเมนูหลักแบบ Pull-down Menu เมื่อผู้ใช้เลื่อนเม้าท์ไปชี้ยังข้อความในเมนูดังกล่าว แล้วคลิกส์ปุ่มด้านซ้ายของเม้าท์หนึ่งครั้ง ผู้ใช้จะได้รายการที่มีให้เลือก สั่งให้คอมพิวเตอร์ทำงานตามที่ทำงานได้ แสดงเมนูหลักในรูปที่ ข-3 และแสดงปุ่มไอคอนคำสั่งทางลัดในการสั่งงานคอมพิวเตอร์ในรูปที่ ข-4

รูปที่ ข-3 แสดงเมนูหลักของโปรแกรม



รูปที่ ข-4 แสดงปุ่มไอคอนคำสั่งทางลัด

1. เมนู File ประกอบด้วย

Save	= บันทึกข้อความจดหมายลงไฟล์
Save Attach File As	= บันทึกไฟล์ที่แนบมากับจดหมาย ไปยัง Directory ที่ต้องการ
Page Setup	= กำหนดขนาดของหน้าจอ
Print	= พิมพ์หน้าปัจจุบัน
Print Preview	= แสดงหน้าจอที่ต้องการพิมพ์ก่อนพิมพ์จริง
Exit	= ออกจากโปรแกรม MobileLink

2. เมนู Edit ประกอบด้วย

Undo	= ยกเลิกคำสั่งก่อนหน้านี้
Cut	= ตัดข้อความที่ไม่ต้องการออก
Copy	= คัดลอกข้อความ
Paste	= ย้ายข้อความ
Delete	= ลบจดหมาย
View Attach File	= ขอดูข้อความในไฟล์ เฉพาะเท็กซ์ไฟล์เท่านั้น

3. เมนู Message ประกอบด้วย

New Message	= เขียนข้อความเพื่อส่งไปยัง Pager, Mobile data และ Cellular
Reply	= ตอบจดหมายกลับไปยังผู้ส่ง
Forward	= ส่งต่อจดหมายไปให้บุคคลอื่น
Address Book	= เก็บตำบลที่อยู่ของสมาชิกที่จะสื่อสารระหว่างกัน

4. เมนู Modem ประกอบด้วย

Setup	= ตั้งค่าพารามิเตอร์ของโมเด็มไร้สาย
Initialize	= ให้เริ่มต้นสถานะของโมเด็มไร้สายใหม่
Connect/Disconnect	= สั่งเชื่อมต่อคอมพิวเตอร์และโมเด็มไร้สาย
Status	= ตรวจสอบสถานะของโมเด็มไร้สาย
Debug	= แสดงข้อมูลที่รับส่งผ่านพอร์ตอนุกรม

5. เมนู Tools ประกอบด้วย

Generate Key	= สร้างกุญแจรหัสลับบันทึกลงไฟล์ไว้
Exchange Key	= แลกเปลี่ยนกุญแจรหัสลับลับกับฝ่ายตรงข้าม
Enable Encryption	= เลือกใช้ หรือยกเลิกการเข้ารหัสลับข้อมูล
Enable Compression	= เลือกใช้ หรือยกเลิกการบีบอัดข้อมูล

6. เมนู Option ประกอบด้วย

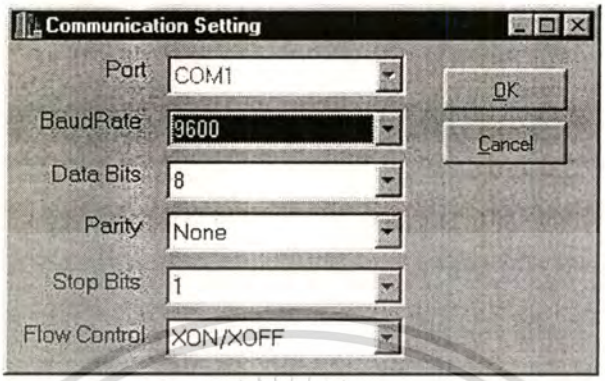
Security	= ตั้งค่าพารามิเตอร์ของการเข้าและถอดรหัสลับ
Compression	= ตั้งค่าพารามิเตอร์ของการบีบอัดข้อมูล
Modem Type	= เลือกชนิดโมเด็มว่าเป็น Radio modem หรือ NULL modem
Save Option	= บันทึกเงื่อนไขที่ตั้งค่าเก็บไว้

7. เมนู Help เป็นส่วนช่วยเหลือการใช้งานโปรแกรม

<u>การตั้งค่าพารามิเตอร์ของพอร์ตสื่อสาร</u>

เลื่อนเม้าท์ไปที่ Modem และคลิกส์เม้าท์ที่ Setup จะปรากฏหน้าจอดังรูปที่ ข-5

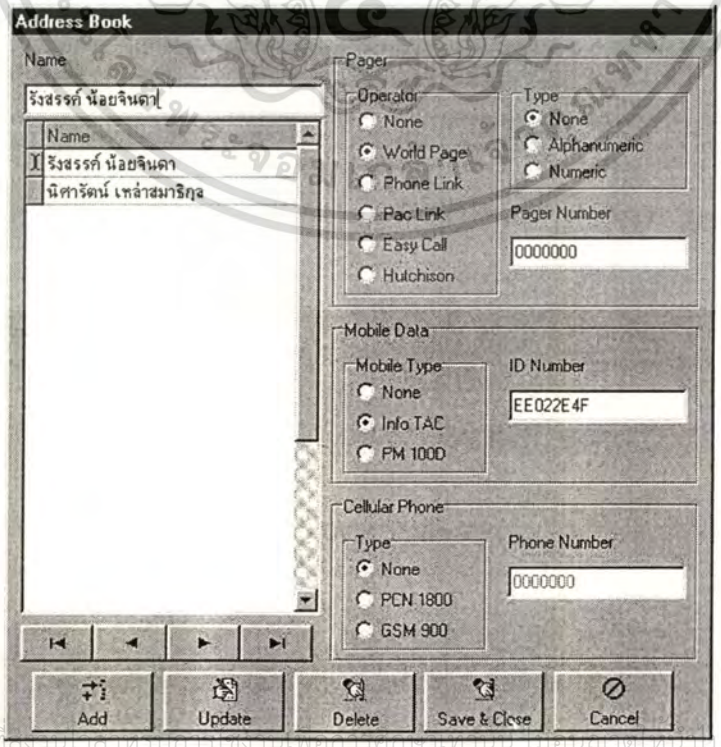
รูปที่ ข-5 แสดงหน้าจอการตั้งค่าพารามิเตอร์ของพอร์ตสื่อสาร



การสร้าง Address Book

การสร้าง Address Book ในโปรแกรม MobileLink เพื่อความสะดวกในการเขียนส่งข้อความให้กับผู้รับ โดยผู้ส่งสามารถเพิ่ม (Add), แก้ไข (Update), ลบ (Delete) ข้อมูล ซึ่งได้แก่ ชื่อเล่น, ชื่อแรก, ชื่อหลัง, และหมายเลขโมเด็มไร้สาย ในอนาคตสามารถระบุหมายเลขมือถือและเพจเจอร์เพื่อใช้งานได้ ในรูปที่ ข-6 แสดงหน้าจอการสร้าง Address book

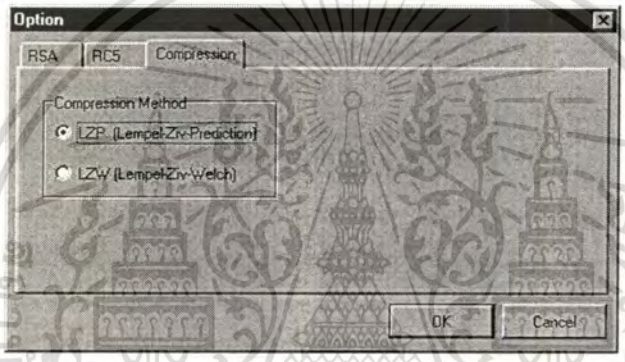
รูปที่ ข-6 แสดงหน้าจอการสร้าง Address book



การเลือกเมนู Option

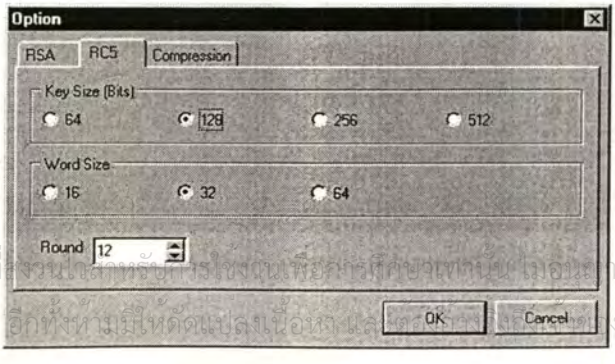
จากเมนูหลักของโปรแกรม MobileLink ให้เลือกที่เมนู Option เพื่อกำหนดค่าพารามิเตอร์ที่ใช้ในการบีบอัดข้อมูล การสร้างกุญแจรหัสลับและการเข้ารหัสลับแบบผสม RSA และ RC5 ในรูปที่ ข-7 แสดงหน้าจอเมื่อเลือกใช้เมนู Option ของการบีบอัดข้อมูล ในรูปที่ ข-8 แสดงหน้าจอเมื่อเลือกใช้เมนู Option ของการเข้ารหัสลับแบบ RSA และในรูปที่ ข-9 แสดงหน้าจอเมื่อเลือกใช้เมนู Option ของการเข้ารหัสลับแบบ RC5

รูปที่ ข-7 แสดงหน้าจอเมื่อเลือกใช้เมนู Option ของการบีบอัดข้อมูล



รูปที่ ข-8 แสดงหน้าจอเมื่อเลือกใช้เมนู Option ของการเข้ารหัสลับ RSA

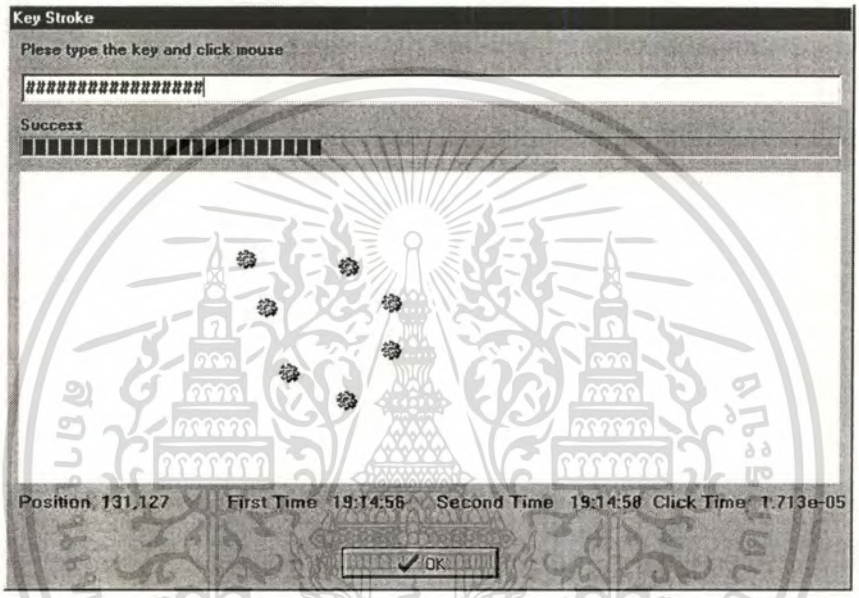
รูปที่ ข-9 แสดงหน้าจอเมื่อเลือกใช้เมนู Option ของการเข้ารหัสลับแบบ RC5



การสร้างกุญแจรหัสลับ

หลังจากตั้งค่าพารามิเตอร์ของการเข้ารหัสลับแล้ว เลือกเมนูหลัก Tools คลิกส์ที่เมนูย่อย Generate Key เพื่อสร้างกุญแจรหัสลับโดยการกดแป้นพิมพ์และคลิกส์เม้าท์ในหน้าจอที่แสดงในรูปที่ ข-10 เราจะได้ผลลัพธ์ดังในรูปที่ ข-11

รูปที่ ข-10 แสดงหน้าจอการสร้างรหัสลับ



รูปที่ ข-11 แสดงผลลัพธ์ที่ได้ของการสร้างกุญแจรหัสลับ

การแลกเปลี่ยนกุญแจรหัสลับ

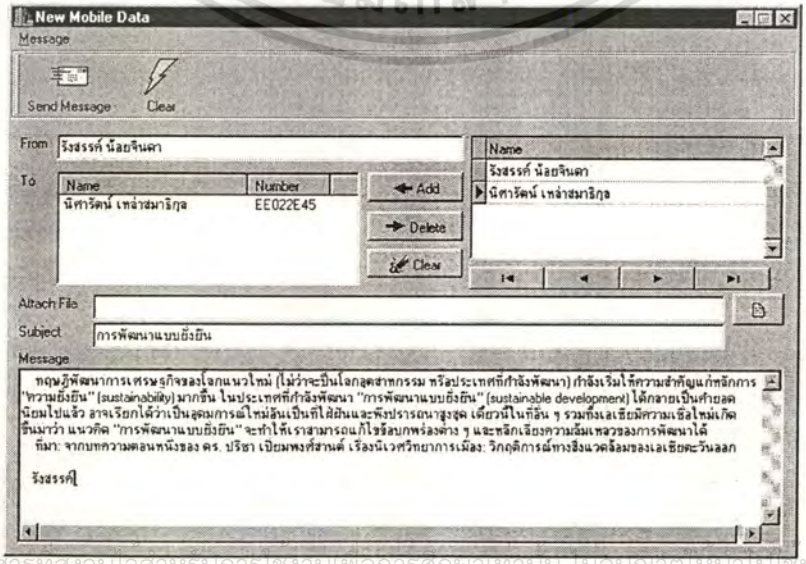
หลังจากสร้างกุญแจรหัสลับแล้ว เมื่อต้องการจะแลกเปลี่ยนกุญแจรหัสลับกับฝ่ายตรงข้าม ให้เลือกเมนูหลัก Tool แล้วคลิกส์ที่เมนูย่อย Exchange Key หน้าจอจะแสดงรายชื่อและตำบลที่อยู่ ให้คลิกรายชื่อที่ต้องการจะส่งกุญแจรหัสลับไปให้ ในรูปที่ ข-12 แสดงหน้าจอการแลกเปลี่ยนกุญแจรหัสลับ

รูปที่ ข-12 แสดงหน้าจอการแลกเปลี่ยนกุญแจรหัสลับ

การเขียนและส่งข้อความจดหมาย

เมื่อผู้ใช้ต้องการเขียนจดหมายถึงผู้อื่นที่มี Address ระบุไว้ในระบบ เราจะเลือก Pull-down ของเมนูหลักที่ Message เลื่อนเม้าท์ไปชี้เมนูย่อยที่ New Message แล้วเลือกที่ Mobile Data หน้าจอจะเข้าสู่การเขียนจดหมายดังรูปที่ ข-13

รูปที่ ข-13 แสดงหน้าจอการเขียนและส่งจดหมาย



ให้เขียนข้อมูลในแต่ละฟิลด์ของส่วนหัวจดหมายดังนี้

From = ระบุชื่อผู้ส่ง

To = ระบุชื่อของผู้รับจดหมาย โดยเลือกจากฟิลด์ Name จะได้ Address ของผู้รับมาด้วย

Attach File = ระบุชื่อไฟล์ที่ต้องการแนบส่งไปพร้อมกับจดหมาย

Subject = ระบุหัวเรื่องของจดหมาย

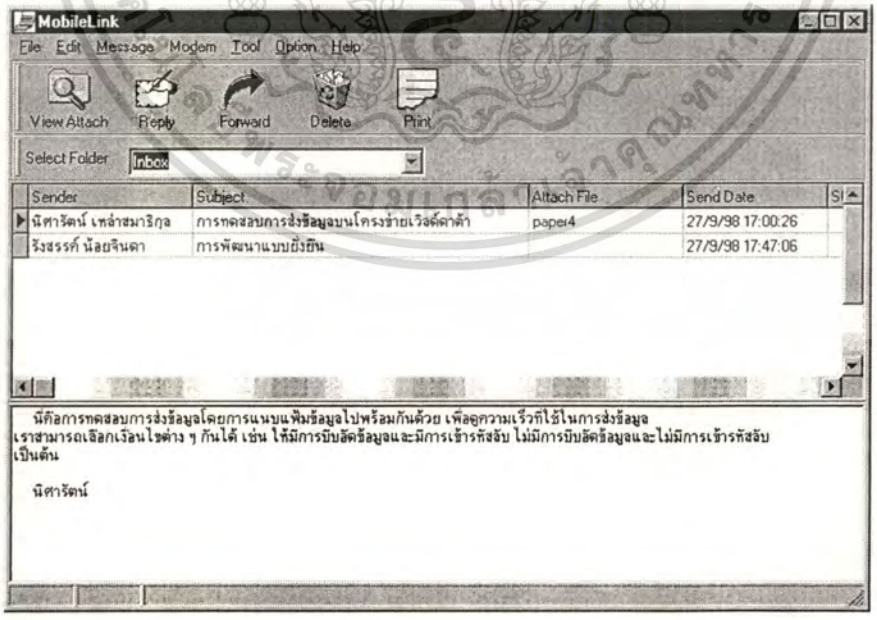
Message = พื้นที่ที่ใช้สำหรับเขียนข้อความที่ต้องการส่งไปยังผู้รับ

เมื่อใส่ข้อความลงฟิลด์ต่าง ๆ ครบถ้วนแล้ว ยกเว้นส่วน Attach File จะใส่หรือไม่ใส่ก็ได้ หลังจากนั้นใช้เม้าท์คลิกส์ปุ่ม Send Message ส่งจดหมายไปยังผู้รับ

การอ่านจดหมาย

จากเมนูหลักเราเลือก Inbox เพื่อแสดงถึงจดหมายที่รับเข้ามา ดังแสดงในรูปที่ ข-14

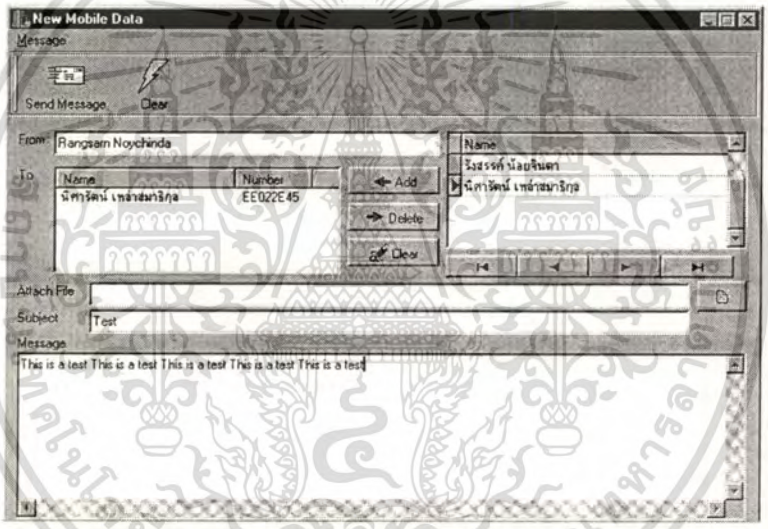
รูปที่ ข-14 แสดงหน้าจอเมนูหลักในการอ่านจดหมาย



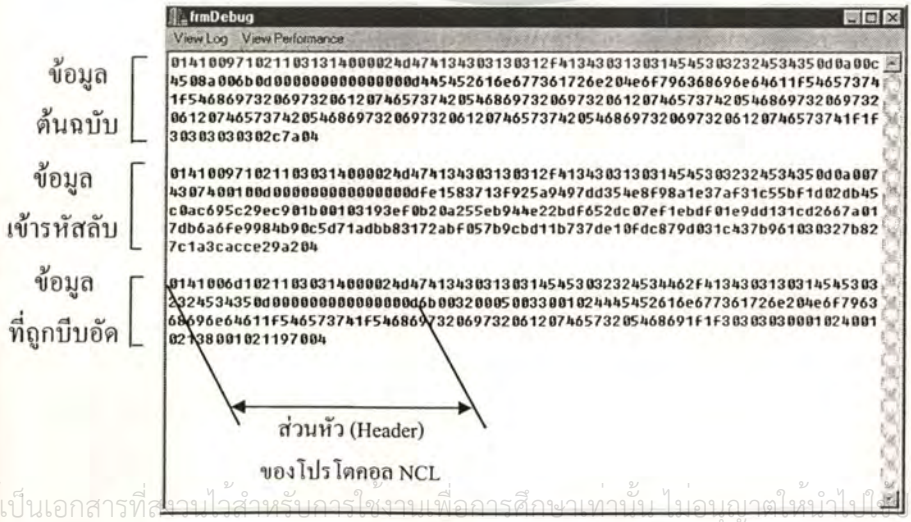
ตัวอย่างข้อมูลที่เข้ารหัสลับและถูกบีบอัด

ในรูปที่ ข-15 แสดงข้อมูลที่ต้องการส่ง และจะถูกประกอบเป็นเฟรมข้อมูลคือ "DTRangsarn Noychinda\x1FTest\x1FThis is a test This is a test This is a test This is a test This is a test\x1F\x1F00000" โดยมีกุญแจปกปิด RC5 ขนาด 128 บิต หรือ 16 ไบต์ อยู่ในรูปแบบของบล็อคควบคุมคือ "0A200C10CA3A63AE1D33D234BE6EAA215494B72E" และในรูปที่ ข-16 แสดงการเปรียบเทียบข้อมูลที่ส่งผ่านพอร์ตอนุกรม ทั้งที่เป็นข้อมูลต้นฉบับ ข้อมูลที่ถูกเข้ารหัสลับปกปิดข้อมูลไว้ และข้อมูลที่ถูกบีบอัด

รูปที่ ข-15 แสดงข้อมูลที่ต้องการส่ง



รูปที่ ข-16 แสดงการเปรียบเทียบข้อมูลที่ส่งผ่านพอร์ตอนุกรม



ภาคผนวก ค.

แสดงฟิลด์ข้อมูลในเฟรมหน่วยบริการข้อมูล (SDU) ของโปรโตคอล NCL

ตาราง ค-1 แสดงฟิลด์ข้อมูลทั่วไป

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
DISABLED	แสดงการ Disable	แอสกี '0'
ENABLED	แสดงการ Enable	แอสกี '1'
UNCONFIRMED	แสดงโหมด Unconfirmed	แอสกี '0'
CONFIRMED	แสดงโหมด Confirmed	แอสกี '1'

ตาราง ค-2 รูปแบบของหน่วยบริการข้อมูล (SDU class)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
CMND (ดูในตาราง ค-3)	ทำหน้าที่เป็นคำสั่ง	แอสกี 'A'
EVENT (ดูในตาราง ค-4)	ทำหน้าที่รายงานเหตุการณ์	แอสกี 'B'
RESP (ดูในตาราง ค-11)	ทำหน้าที่แสดงสถานะการตอบสนอง	แอสกี 'C'
N/A	สงวนไว้ใช้ในอนาคต	แอสกี 'D'...'Z' และ '1'...'9'

ตาราง ค-3 ชุดคำสั่ง (Command) ในหน่วยบริการข้อมูล

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
SEND	ทำหน้าที่ส่งข้อมูล	แอสกี '1'
READ_MSG	อ่านข้อมูลจากโมเด็มไร้สาย เฉพาะโหมด Confirmed	แอสกี '2'
CTL_EVENT	ควบคุมการรายงานเหตุการณ์	แอสกี '3'
GET_STATUS (ดูในตาราง ค-15)	อ่านค่าสถานะและพารามิเตอร์จากโมเด็มไร้สาย	แอสกี '4'
SET_CNF (ดูในตาราง ค-16)	ตั้งค่าพารามิเตอร์ที่โมเด็มไร้สาย	แอสกี '5'
RESET_RPM	ทำการรีเซ็ตโมเด็มไร้สาย	แอสกี '6'
N/A	สงวนไว้ใช้ในอนาคต	แอสกี 'A'...'Y' และ '7'...'9'
VENDOR	ผู้ผลิตต่าง ๆ สามารถระบุหน้าที่การใช้งานเองได้	แอสกี 'Z'

ตาราง ค-4 แสดงบิตรายงานเหตุการณ์ (Event)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
CONTROL_BIT	บิตข้อมูลสำหรับการ enable ฟิลด์ CONTROL	สตริง "20"
RCV_MSG_DATA_BIT	บิตข้อมูลสำหรับการ enable ฟิลด์ RCV_MSG_DATA	สตริง "10"
RCV_MSG_NOTIFY_ BIT	บิตข้อมูลสำหรับการ enable ฟิลด์ RCV_MSG_NOTIFICATION	สตริง "08"
TX_EVENT_BIT	บิตข้อมูลสำหรับการ enable ฟิลด์ TX_EVENT	สตริง "04"
RX_EVENT_BIT	บิตข้อมูลสำหรับการ enable ฟิลด์ RX_EVENT	สตริง "02"
HW_EVENT_BIT	บิตข้อมูลสำหรับการ enable ฟิลด์ HW_EVENT	สตริง "01"
N/A	สงวนบิตไว้ใช้ในอนาคต	สตริง "C0"

ตาราง ค-5 ชุดรายงานเหตุการณ์ (Event report) ในหน่วยบริการข้อมูล

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
RCV_MSG_DATA	รับข้อมูลเข้ามาจากโมเด็มไร้สาย	แอสกี 'A'
RCV_MSG_ NOTIFICATION	แจ้งการรับข้อมูล เฉพาะโหมด Confirmed เท่านั้น	แอสกี 'B'
TX_EVENT (ดูในตาราง ค-7)	รายงานเหตุการณ์ผู้ส่งในระดับชั้นทางกายภาพ	แอสกี 'C'
RX_EVENT (ดูในตาราง ค-8)	รายงานเหตุการณ์ผู้รับในระดับชั้นทางกายภาพ	แอสกี 'D'
HW_EVENT (ดูในตาราง ค-9)	รายงานเหตุการณ์ของอุปกรณ์ฮาร์ดแวร์	แอสกี 'E'
RCV_ERR (ดูในตาราง ค-6)	รายงานเหตุการณ์ที่ไม่สามารถรับข้อมูลได้	แอสกี 'F'
CONTROL (ดูในตาราง ค-10)	ควบคุมการรายงานเหตุการณ์	แอสกี 'G'
N/A	สงวนไว้ใช้ในอนาคต	แอสกี 'H'...'Y'
VENDOR	ผู้ผลิตต่าง ๆ สามารถระบุหน้าที่การใช้งานเองได้	แอสกี 'Z'

ตาราง ค-6 รายงานเหตุการณ์ที่ไม่สามารถรับข้อมูลได้ (RCV_ERR)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
RCV_TX_DISABLED	เมื่อรับแพ็คเก็ต PDU เข้ามา ผู้รับจะต้องมีการตอบกลับ (ACK) แต่ผู้รับไม่สามารถตอบกลับได้เนื่องจากผู้ส่งมีการ Disable อยู่	แอสกี '1'

ตาราง ค-7 รายงานเหตุการณ์ผู้ส่งในระดับชั้นทางกายภาพ (TX_EVENT)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
TX_KEYED	ผู้ส่ง Keyed	แอสกี '1'
TX_DEKEYED	ผู้ส่ง Dekeyed	แอสกี '2'

ตาราง ค-8 รายงานเหตุการณ์ผู้รับในระดับชั้นทางกายภาพ (RX_EVENT)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
RX_IN_RANGE	แสดงถึงผู้รับอยู่ในพื้นที่บริการ	แอสกี '1'
RX_OUT_OF_RANGE	แสดงถึงผู้รับอยู่นอกพื้นที่บริการ	แอสกี '2'
RX_PWR_SAVE_ ENABLED	แสดงการอยู่ในโหมดประหยัดพลังงาน	แอสกี '3'
RX_PWR_SAVE_ DISABLED	แสดงการยกเลิกใช้โหมดประหยัดพลังงาน	แอสกี '4'

ตาราง ค-9 รายงานเหตุการณ์ของอุปกรณ์ฮาร์ดแวร์ (HW_EVENT)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
HW_SELF_TEST	ทดสอบหาข้อผิดพลาดที่ตัวโมเด็มไร้สาย	แอสกี '1'
HW_LOW_BATT	แสดงกำลังไฟต่ำ	แอสกี '2'
HW_MEM_FULL	แสดงหน่วยความจำเต็ม	แอสกี '3'
HW_BATT_OK	แสดงระดับกำลังไฟยังใช้งานได้อยู่	แอสกี '4'
HW_MEM_OK	แสดงหน่วยความจำยังใช้ได้อีก	แอสกี '5'
HW_OFF	แสดงภาวะการปิดเครื่อง	แอสกี '6'

ตาราง ค-10 ควบคุมการรายงานเหตุการณ์ (CONTROL)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
CONNECT	แสดงสภาวะการเชื่อมต่อระหว่างโมเด็มไร้สาย และคอมพิวเตอร์ปลายทางว่าต่อกันได้แล้ว โดยการใช้โปรโตคอล NCL	แอสกี '1'

ตาราง ค-11 สถานะการตอบสนอง (Response)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
SUCCESS (ดูในตาราง ค-12)	กระทำตามคำสั่งสำเร็จ	แอสกี '1'
XFAIL (ดูในตาราง ค-13)	การปฏิบัติคำสั่งผิดพลาด	แอสกี '2'
SYNTAX (ดูในตาราง ค-14)	การใช้คำสั่งผิดรูปแบบ	แอสกี '3'
VENDOR	การตอบสนองสามารถกำหนดโดยผู้ผลิต	แอสกี 'Z'

ตาราง ค-12 รหัสผิดพลาดของ SUCCESS

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
IBQ_FLUSHED	มีการหยุดรอ (Pending) หน่วยบริการข้อมูล ในการเข้าแถว ด้าน Inbound เนื่องจากผู้ส่งมีการ disable	แอสกี 'a'

ตาราง ค-13 รหัสผิดพลาดของ XFAIL

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
NO_RESPONSE	ไม่มีการตอบสนองจากโครงข่าย	แอสกี 'A'
NO_ACK	การตอบกลับเป็นไปในทางตรงกันข้าม (NAK)	แอสกี 'B'
HOST_DOWN	คอมพิวเตอร์ปลายทางไม่ทำงาน	แอสกี 'C'
NOT_REGISTERED	ไม่ได้ลงทะเบียนโมเด็มไร้สายไว้	แอสกี 'D'
LOW_BATTERY	กำลังไฟอ่อน ไม่สามารถส่งข้อมูลได้	แอสกี 'E'
IBQ_FULL	การเข้าแถว (Queue) ข้อมูลเต็ม	แอสกี 'F'
TX_DISABLED	ยกเลิกการส่งสัญญาณวิทยุ	แอสกี 'G'
BUSY	สภาวะไม่พร้อมให้ใช้งาน	แอสกี 'H'
NOT_AVAILABLE	ไม่สามารถให้บริการได้	แอสกี 'I'
HW_ERROR	ฮาร์ดแวร์ผิดพลาด	แอสกี 'J'
INVALID_MODE	โหมดการทำงานไม่ถูกต้อง	แอสกี 'K'
NO_MESSAGES	ไม่มีข้อมูลด้าน Outbound	แอสกี 'L'
MSGS_PENDING	ไม่สามารถปฏิบัติคำสั่งได้ เนื่องจากมีการ Pending ข้อมูลด้าน Inbound	แอสกี 'M'
SW_ERROR	โปรแกรมผิดพลาด	แอสกี 'N'
OUT_OF_RANGE	อยู่นอกพื้นที่บริการ	แอสกี 'O'
PACKET_ERROR	แพ็คเก็ตของหน่วยบริการข้อมูล SDU เสียหาย	แอสกี 'Z'
N/A	สงวนไว้ใช้ในอนาคต	แอสกี 'P'...'Y' และ '1'...'9'

ตาราง ค-14 รหัสผิดพลาดของ SYNTAX

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
INVALID	ใช้คำสั่งไม่ถูกต้อง ให้ยกเลิกการใช้งาน	แอสกี 'b'
TOO_LONG	ข้อมูลยาวเกินกว่าค่าที่กำหนด	แอสกี 'c'

ตาราง ค-15 สถานะการร้องขอ (GET_STATUS)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
R_CONFIG_BLOCK	อ่านบล็อคพารามิเตอร์ของโมเด็มไร้สาย	แอสกี 'A'
R_STATUS_BLOCK	อ่านบล็อคสถานะของโมเด็มไร้สาย	แอสกี 'B'
R_PROD_ID	อ่านผลิตภัณฑ์ ID ของโมเด็มไร้สาย	แอสกี 'C'
R_SW_VERSION	อ่านเวอร์ชั่นของโปรแกรมที่ใช้	แอสกี 'D'
R_RPM_ID	อ่านตำบลที่อยู่ (Address) ของโมเด็มไร้สาย	แอสกี 'E'
N/A	สงวนไว้ใช้ในอนาคต	แอสกี 'F'
R_MAX_DATA_SIZE	อ่านขนาดข้อมูลสูงสุดที่มีได้	แอสกี 'G'
R_RPM_GID	อ่านค่ากลุ่ม ID ของโมเด็มไร้สาย	แอสกี 'H'
R_WAN_TYPE	อ่านรหัสชนิดของ WAN	แอสกี 'I'
R_RF_VERSION	อ่านเวอร์ชั่นของสัญญาณคลื่นวิทยุ	แอสกี 'J'
R_VENDOR_ID	อ่านข้อมูลข่าวสารของผู้ผลิต	แอสกี 'K'
N/A	สงวนไว้ใช้ในอนาคต	แอสกี 'L'...'Z'
R_RCV_MODE	โหมดการแจ้งข่าวสารไปยังคอมพิวเตอร์ในการรับข้อมูลจากหน่วยบริการข้อมูล SDU	แอสกี 'a'
R_RX_STATUS	อ่านสถานะการ Enable ของผู้รับ	แอสกี 'b'
R_TX_STATUS	อ่านสถานะการ Enable ของผู้ส่ง	แอสกี 'c'
R_ANTENNA	อ่านสถานะการเลือกสายอากาศ	แอสกี 'd'
R_RADIO_IN_RANGE	อ่านระดับความเข้มของสัญญาณ	แอสกี 'e'
R_OB_MSG_COUNT	นับจำนวนการเข้าแถวข้อมูลด้าน Outbound	แอสกี 'f'
R_IB_MSG_COUNT	นับจำนวนการเข้าแถวข้อมูลด้าน Inbound	แอสกี 'g'
R_FLOW_CONTROL	อ่านสถานะการควบคุมการไหลข้อมูล	แอสกี 'h'
R_EVENT_STATES	อ่านเหตุการณ์สถานะการ Enable และ Disable	แอสกี 'i'
R_RADIO_CHANNEL	อ่านช่องสัญญาณวิทยุ ตำแหน่งปัจจุบัน	แอสกี 'j'
R_CHAN_BLOCK	อ่านสถานะช่องสัญญาณความถี่วิทยุ	แอสกี 'k'
R_RF_STATISTICS	อ่านค่าสถิติของความถี่วิทยุ	แอสกี 'l'

ตาราง ค-15 (ต่อ) สถานะการร้องขอ (GET_STATUS)

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
R_BAT_LEVEL	อ่านสภาพกำลังไฟ	แอสกี 'm'
N/A	สงวนไว้ในอนาคต	แอสกี 'n'...'x'
R_DCHAN_TABLE	อ่านตารางช่องสัญญาณไม่คงที่	แอสกี 'y'
R_CHAN_TABLE	อ่านตารางช่องสัญญาณคงที่	แอสกี 'z'

ตาราง ค-16 การตั้งค่าพารามิเตอร์

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
S_RCV_MODE	เลือกโหมดยืนยันหรือไม่มีการยืนยันการรับข้อมูล	แอสกี 'A'
S_TX_CONTROL	เลือกใช้หรือยกเลิก การส่งข้อมูล	แอสกี 'B'
S_RX_CONTROL	เลือกใช้หรือยกเลิก สัญญาณวิทยุ	แอสกี 'C'
S_FLOW_CONTROL	เลือกวิธีการควบคุมการไหลข้อมูล	แอสกี 'D'
S_RADIO_CHANNEL	เลือกช่องสัญญาณวิทยุ	แอสกี 'E'
S_CUR_CNF	เก็บค่าพารามิเตอร์ ที่เลือกในปัจจุบันไว้	แอสกี 'F'
R_DEF_CNF	โหลดค่า Default ใช้งาน	แอสกี 'G'
R_STO_CNF	อ่านค่าพารามิเตอร์ของโมเด็มไร้สายเก็บไว้	แอสกี 'H'
S_POWER_SAVE	เลือกโหมดประหยัดพลังงาน	แอสกี 'I'
S_ROAM_MODE	เลือกโหมดการทำโรมมิ่ง	แอสกี 'J'
S_BAUD	เลือกอัตราความเร็วในการสื่อสารด้วยคำสั่ง NCL	แอสกี 'K'
N/A	สงวนไว้ใช้ในอนาคต	แอสกี 'L'...'Z'

ตาราง ค-17 การเลือกพารามิเตอร์ S_FLOW_CONTROL

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
FLOW_NONE	ไม่มีการควบคุมการไหลของข้อมูล	แอสกี '0'
FLOW_XONXOFF	ควบคุมการไหลข้อมูลแบบ XON/XOFF	แอสกี '1'
FLOW_RTSCTS	ควบคุมการไหลข้อมูลแบบ RTS/CTS	แอสกี '2'

ตาราง ค-18 ตั้งค่าพารามิเตอร์ของ S_CUR_CNF,R_DEF_CNF,R_STO_CNF

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
CNF_EVENT_FLAGS	การตั้งค่าแฟล็กควบคุมการรายงานเหตุการณ์	แอสกี '0'
CNF_DELIVERY_MODE	โหมดนำส่งหน่วยบริการข้อมูล Outbound	แอสกี '1'
CNF_RADIO_CONTROL	ควบคุมคลื่นวิทยุด้านส่งและรับ คือ S_RX_CONTROL และ S_TX_CONTROL	แอสกี '2'

ตาราง ค-19 พารามิเตอร์ของ RESET_RPM

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
FLUSH_INBOUND	นำข้อมูลออกจากแถวรอ (Queue) ด้าน Inbound	แอสกี '1'
FLUSH_OUTBOUND	นำข้อมูลออกจากแถวรอ (queue) ด้าน Outbound	แอสกี '2'
FLUSH_BOTH	นำข้อมูลออกทั้งสองด้าน Inbound และOutbound	แอสกี '3'
RESET_WARM	เริ่ม Warm up โมเด็มไร้สาย	แอสกี '4'
RESET_TRANS	ทำการรีเซ็ตในโหมดของ Transparent	แอสกี '5'
RESET_FULL	ทำการรีเซ็ตโมเด็มไร้สายใหม่ทั้งหมด	แอสกี '6'
RESET_NCL	รีเซ็ตส่วนแปลภาษา NCL เท่านั้น	แอสกี '7'
RESET_OFF	สั่งปิดโมเด็มไร้สาย	แอสกี '8'

ตาราง ค-20 พารามิเตอร์ S_ROAM_MODE

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
ROAM_MANUAL	ตั้งค่าการทำโรมมิ่งแบบธรรมดา	แอสกี '0'
ROAM_AUTO	ตั้งค่าการทำโรมมิ่งแบบอัตโนมัติ	แอสกี '1'

ตาราง ค-21 พารามิเตอร์ S_BAUD

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
BAUD_1200	ความเร็ว 1200 บิตต่อวินาที	แอสกี '0'
BAUD_2400	ความเร็ว 2400 บิตต่อวินาที	แอสกี '1'
BAUD_4800	ความเร็ว 4800 บิตต่อวินาที	แอสกี '2'
BAUD_9600	ความเร็ว 9600 บิตต่อวินาที	แอสกี '3'
BAUD_19K2	ความเร็ว 19.2 กิโลบิตต่อวินาที	แอสกี '4'
BAUD_38K4	ความเร็ว 38.4 กิโลบิตต่อวินาที	แอสกี '5'

ตาราง ค-22 พารามิเตอร์ของ R_PROD_ID

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
RF_RDLAP_9.6	โปรโตคอลคลื่นวิทยุชนิด RD-LAP 9600	แอสกี '0'
RF_RDLAP_19.2	โปรโตคอลคลื่นวิทยุชนิด RD-LAP 19200	แอสกี '1'
RF_MDC4800	โปรโตคอลคลื่นวิทยุชนิด MDC 4800	แอสกี '2'
RF_DUAL	โปรโตคอลคลื่นวิทยุสองระบบชนิด RD-LAP 19200 และ MDC 4800	แอสกี '3'

ตาราง ค-23 พารามิเตอร์ของ R_VENDOR_ID

ฟิลด์ข้อมูล	รายละเอียด	แทนค่าด้วย
VENDOR_MOTOROLA	ผู้ผลิตคือโมโตโรล่า	แอสกี '0'

ภาคผนวก ง.

ตารางรหัสแอสกี (Ascii table)

D	H	Ch	Ctrl	Nm	D	H	Ch	Ctrl	Nm
0	00		^@	NUL	16	10	►	^P	DLE
1	01	☺	^A	SOH	17	11	◄	^Q	DC1
2	02	☻	^B	STX	18	12	↕	^R	DC2
3	03	♥	^C	ETX	19	13	‼	^S	DC3
4	04	♦	^D	EOT	20	14	¶	^T	DC4
5	05	♣	^E	ENQ	21	15	§	^U	NAK
6	06	♠	^F	ACK	22	16	▬	^V	SYN
7	07	•	^G	BEL	23	17	↨	^W	ETB
8	08	◘	^H	BS	24	18	↑	^X	CAN
9	09	○	^I	HT	25	19	↓	^Y	EM
10	0A	◙	^J	LF	26	1A	→	^Z	SUB
11	0B	♂	^K	VT	27	1B	←	^[	ESC
12	0C	♀	^L	FF	28	1C	∟	^\	FS
13	0D	♪	^M	CR	29	1D	↔	^]	GS
14	0E	♫	^N	SO	30	1E	▲	^^	RS
15	0F	☼	^O	SI	31	1F	▼	^_	US

D	H	Ch	D	H	Ch	D	H	Ch	D	H	Ch
32	20		48	30	0	64	40	@	80	50	P
33	21	!	49	31	1	65	41	A	81	51	Q
34	22	"	50	32	2	66	42	B	82	52	R
35	23	#	51	33	3	67	43	C	83	53	S
36	24	$	52	34	4	68	44	D	84	54	T
37	25	%	53	35	5	69	45	E	85	55	U
38	26	&	54	36	6	70	46	F	86	56	V
39	27	'	55	37	7	71	47	G	87	57	W
40	28	(	56	38	8	72	48	H	88	58	X
41	29	)	57	39	9	73	49	I	89	59	Y
42	2A	*	58	3A	:	74	4A	J	90	5A	Z
43	2B	+	59	3B	;	75	4B	K	91	5B	[
44	2C	,	60	3C	<	76	4C	L	92	5C	\
45	2D	-	61	3D	=	77	4D	M	93	5D	]
46	2E	.	62	3E	>	78	4E	N	94	5E	^
47	2F	/	63	3F	?	79	4F	O	95	5F	_

D	H	Ch	D	H	Ch	D	H	Ch	D	H	Ch
96	60	`	112	70	p	128	80	Ç	144	90	É
97	61	a	113	71	q	129	81	ü	145	91	æ
98	62	b	114	72	r	130	82	é	146	92	Æ
99	63	c	115	73	s	131	83	â	147	93	ô
100	64	d	116	74	t	132	84	ä	148	94	ö
101	65	e	117	75	u	133	85	à	149	95	ò
102	66	f	118	76	v	134	86	å	150	96	û
103	67	g	119	77	w	135	87	ç	151	97	ù
104	68	h	120	78	x	136	88	ê	152	98	ÿ
105	69	i	121	79	y	137	89	ë	153	99	Ö
106	6A	j	122	7A	z	138	8A	è	154	9A	Ü
107	6B	k	123	7B	{	139	8B	ï	155	9B	ø
108	6C	l	124	7C	|	140	8C	î	156	9C	£
109	6D	m	125	7D	}	141	8D	ì	157	9D	Ø
110	6E	n	126	7E	~	142	8E	Ä	158	9E	×
111	6F	o	127	7F	⌂	143	8F	Å	159	9F	ƒ

ตารางรหัสแอสกี (Ascii table) (ต่อ)

D	H	Ch	D	H	Ch	D	H	Ch	D	H	Ch
160	A0	á	176	B0	▒	192	C0	└	208	D0	ð
161	A1	í	177	B1	▒	193	C1	┴	209	D1	Ð
162	A2	ó	178	B2	▓	194	C2	┬	210	D2	Ê
163	A3	ú	179	B3	│	195	C3	├	211	D3	Ë
164	A4	ñ	180	B4	┤	196	C4	─	212	D4	È
165	A5	Ñ	181	B5	Á	197	C5	┼	213	D5	ı
166	A6	ª	182	B6	Â	198	C6	ã	214	D6	Í
167	A7	º	183	B7	À	199	C7	Ã	215	D7	Î
168	A8	¿	184	B8	©	200	C8	╚	216	D8	Ï
169	A9	®	185	B9	╣	201	C9	╔	217	D9	┘
170	AA	¬	186	BA	║	202	CA	╩	218	DA	┌
171	AB	½	187	BB	╗	203	CB	╦	219	DB	█
172	AC	¼	188	BC	╝	204	CC	╠	220	DC	▄
173	AD	¡	189	BD	¢	205	CD	═	221	DD	¦
174	AE	«	190	BE	¥	206	CE	╬	222	DE	Ì
175	AF	»	191	BF	┐	207	CF	¤	223	DF	▀

D	H	Ch	D	H	Ch	D	H	Display Attr
224	E0	ó	240	F0	≡	0	00	Black
225	E1	ß	241	F1	±	1	01	Blue
226	E2	ô	242	F2	‗	2	02	Green
227	E3	Ò	243	F3	¾	3	03	Cyan
228	E4	õ	244	F4	¶	4	04	Red
229	E5	õ	245	F5	§	5	05	Magenta
230	E6	µ	246	F6	÷	6	06	Brown
231	E7	þ	247	F7	¸	7	07	Light Gray
232	E8	Þ	248	F8	°	8	08	Dark Gray
233	E9	Ú	249	F9	¨	9	09	Light Blue
234	EA	Û	250	FA	·	10	0A	Light Green
235	EB	Ù	251	FB	¹	11	0B	Light Cyan
236	EC	ý	252	FC	³	12	0C	Light Red
237	ED	Ý	253	FD	²	13	0D	Light Magenta
238	EE	¯	254	FE	■	14	0E	Yellow
239	EF	´	255	FF		15	0F	White

ประวัติผู้เขียน

นายรังสรรค์ น้อยจินดา เกิดเมื่อวันที่ 20 กรกฎาคม 2511 ที่จังหวัดฉะเชิงเทรา สำเร็จการศึกษาอุตสาหกรรมศาสตรบัณฑิต สาขาเทคโนโลยีคอมพิวเตอร์อุตสาหกรรม จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2534 และประกาศนียบัตรวิชาชีพชั้นสูง สาขาอิเล็กทรอนิกส์ จากวิทยาลัยเทคนิคฉะเชิงเทรา ปีการศึกษา 2530 ปัจจุบันทำงานที่บริษัท ยูไนเต็ด คอมมูนิเกชั่น อินดัสตรี (เนทเวอร์ค) จำกัด (มหาชน) หรือยูคอม (เนทเวอร์ค)