บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงหลักการเข้ารหัสลับข้อมูล (Cryptography) เป็นส่วนสำคัญส่วนหนึ่งในการรักษาความปลอดภัยของข้อมูล จะแบ่งเป็น 2 กลุ่มใหญ่ๆ คือ แบบกุญแจสมมาตร (Symmetric cryptography หรือ Private Key) และแบบกุญแจอสมมาตร (Asymmetric cryptography หรือ Public Key) ในแต่ละแบบจะมีการจัดการกับข้อมูลที่ต่างกัน โดยจะศึกษาในส่วนของอัลกอริทึมในการเข้ารหัสแบบกุญแจสมมาตร

2.1 การเข้ารหัสข้อมูลลับ (Cryptography)

การเข้ารหัส เป็นวิธีการแปลงข้อมูลธรรมดาที่เราสามารถอ่านได้ให้อยู่ในรูปของข้อมูลสุ่มที่ไม่สามารถอ่านได้ ข้อมูลที่สามารถอ่านได้เรียกว่า Plaintext หรือ Cleartext ข้อมูลที่เข้ารหัสแล้วเรียกว่า Cipher text หรือ Code text ข้อมูลที่อยู่ในรูปแบบของการเข้ารหัสเรียกว่า Cryptogram

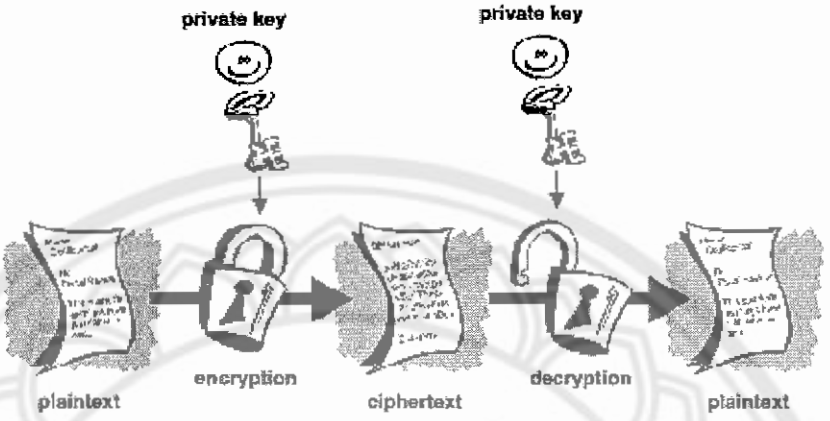
การเข้ารหัสมีหลักการสำคัญคือ เมื่อเข้ารหัสไปแล้วต้องสามารถถอดรหัสกลับมาได้ และสำหรับการถอดรหัส (Decipher) เป็นวิธีที่ใช้เพื่อทำให้อักขระที่ไม่สามารถอ่านได้กลับมาเป็นข้อความเดิม ซึ่งใช้วิธีการวิเคราะห์ข้อมูลเพื่อจะนำไปถอดรหัสที่เรียกว่า Cryptanalysis เทคนิคที่ใช้ในการเข้ารหัสนั้นเรียกว่า อัลกอริทึมในการเข้ารหัส (Encryption Algorithm) ซึ่งเป็นตัวกำหนดวิธีความซับซ้อนของการเข้ารหัส ส่วนการเข้ารหัสและถอดรหัสเราจะใช้สิ่งที่เรียกว่า กุญแจ (Key) ที่จะทราบกันเพียงผู้รับและผู้ส่งข้อมูลเท่านั้น ซึ่งกุญแจนี้จะถูกอัลกอริทึมนำไปใช้ในกระบวนการแปลงข้อมูล การเข้ารหัสแบ่งออกได้เป็น 2 ประเภทใหญ่ๆ คือ Secret Key Encryption หรือ Symmetric Cryptography และ Public Key Encryption หรือ Asymmetric Cryptography

2.2 ประเภทของการเข้ารหัสข้อมูลลับ

2.2.1 Secret Key Encryption หรือ Symmetric Cryptography

การเข้ารหัสและถอดรหัสโดยอาศัยคีย์เดี่ยวหรือการเข้ารหัสข้อมูลลับแบบกุญแจสมมาตร จะใช้ในสภาวะแวดล้อมที่สามารถแลกเปลี่ยนคีย์ระหว่างผู้ใช้กันได้ง่าย เช่น ตามหน่วยงาน ทั่วไป หรืออาจใช้เพื่อเข้ารหัสข้อมูลที่เก็บอยู่ในดิสก์ การเข้ารหัสแบบ Secret Key เป็นการเข้ารหัสที่ ใช้วิธีการแทนที่ (Substitution) และวิธีสลับตำแหน่ง (Transposition) ซึ่งผู้ใช้ทุกคนจะต้องใช้อัลกอริทึมเดียวกันหมด แต่จะมีปัญหาที่เกิดขึ้น คือ ชื่อผู้ส่งอาจถูกปลอมแปลงได้ และ วิธีนี้เป็นวิธีที่ทั้งสองฝ่าย ต้องมีรหัสลับที่ใช้ร่วมกัน ดังนั้นถ้ามีการติดต่อระหว่างคนหลายคน ก็จำเป็นที่จะต้อง

มีรหัสครบตามจำนวนของผู้ติดต่อ รวมทั้งยังต้องคอยจดจำว่ารหัสใดใช้กับใคร ซึ่งเป็นการยุ่งยากมาก เพราะหากใช้รหัสเดียวกันทั้งหมดในการติดต่อก็จะทำให้ทุกคนอ่านข้อความของกันและกันได้หมด



รูปที่ 2.1 วิธีการเข้าและถอดรหัสที่ใช้ในการรับส่งข้อความแบบ Secret Key [1]

ข้อดีของการเข้ารหัสแบบสมมาตร

1. การเข้ารหัสและถอดรหัสข้อมูลใช้เวลาน้อย เพราะว่าอัลกอริทึมที่ใช้ไม่ได้สลับซับซ้อน

2. ขนาดของข้อมูลหลังจากทำการเข้ารหัสแล้ว มีการเปลี่ยนแปลงไม่มาก หรือ ข้อมูลหลังจากทำการเข้ารหัสแล้ว จะมีขนาดไม่ใหญ่ไปกว่าเดิมมากนัก

ข้อด้อยของการเข้ารหัสแบบสมมาตร

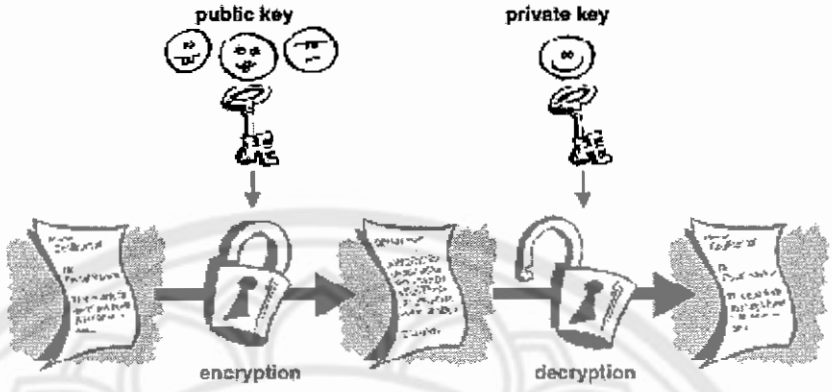
1. การจัดการกับกุญแจลับที่ยุ่งยาก เพราะต้องจำให้ได้กุญแจดอกไหนใช้ติดต่อกับใคร ซึ่งเป็นการยุ่งยากที่จะจำให้หมด

2. การกระจายกุญแจลับ เนื่องจากการเข้ารหัสวิธีนี้ต้องใช้กุญแจลับ 1 ดอกต่อผู้รับ 1 คน ดังนั้นถ้าต้องติดต่อกับคนมากๆ ก็ต้องส่งกุญแจลับที่ใช้ไปให้กับทุกคน

2.2.2 Public Key Encryption หรือ Asymmetric Cryptography

การเข้ารหัสและถอดรหัสโดย อาศัยคีย์ 2 คีย์ หรือการเข้ารหัสข้อมูลลับแบบอสมมาตร มักใช้ในสภาวะแวดล้อมที่การแลกเปลี่ยนคีย์เป็นไปได้ยาก เช่น เนตเวิร์กสาธารณะใหญ่ๆ อย่างอินเทอร์เน็ต เป็นต้น ระบบการเข้าและถอดรหัสแบบคีย์สาธารณะนี้จะใช้แนวคิดของการมีคีย์เป็นคู่ๆ ที่สามารถเข้าและถอดรหัสของกันและกันเท่านั้นได้ คีย์แรกจะทราบหรือมีอยู่ที่เฉพาะเจ้าของคีย์นั้นเอง ซึ่งจะเรียกว่า "Private Key" และจะมีคู่ของคีย์ดังกล่าวที่ส่งให้ผู้อื่นใช้ได้ เรียกว่า "Public Key" โดย Public Key นี้จะถูกแจกจ่ายให้ผู้อื่นที่ต้องการส่งข้อความถึงเจ้าของ Private Key ตัวนั้น

เพื่อที่ผู้ส่งจะสามารถใช้ Public Key ของผู้รับในการเข้ารหัส แล้วจึงส่งข้อความไปให้ ซึ่งก็จะมีเพียงเจ้าของ Private Key ซึ่งคู่กันนั้นเพียงคนเดียวที่จะสามารถถอดรหัสและอ่านข้อความนั้นได้



รูปที่ 2.2 วิธีการเข้ารหัสและถอดรหัสที่ใช้ในการรับส่งข้อความแบบ Public Key [1]

ข้อดีของระบบเข้ารหัสแบบกุญแจอสมมาตร

1. การจัดการกับกุญแจทำได้ง่าย เพราะว่าผู้ส่งไม่ต้องจำเลยว่าได้ใช้กุญแจคู่ไหนกับใคร ซึ่งผู้ส่งเพียงแค่ใช้กุญแจส่วนตัวของตัวเองทำการถอดรหัสข้อมูลที่ผู้ส่ง ส่งมาให้ หรือเอากุญแจส่วนตัวเข้ารหัสส่งไปให้ ผู้รับ ผู้รับก็สามารถอ่านข้อมูลที่ส่งให้ได้ ซึ่งวิธีนี้จะทำให้ง่ายขึ้น เพราะผู้ส่งใช้เพียงกุญแจส่วนตัวดอกเดียวก็สามารถติดต่อกับผู้รับได้

2. การกระจายกุญแจลับ เนื่องจากการเข้ารหัสโดยวิธีนี้ ใช้แค่กุญแจสาธารณะเพียงดอกเดียวในการเข้ารหัสและถอดรหัส และกุญแจสาธารณะของผู้ส่ง ก็สามารถที่จะเปิดเผยให้กับใครก็ได้ที่ต้องการจะติดต่อด้วย เพราะฉะนั้นการแจกจ่ายกุญแจสาธารณะของผู้ส่งไปให้กับหลายๆ คนได้ง่าย

ข้อด้อยของระบบเข้ารหัสแบบกุญแจอสมมาตร

1. การเข้ารหัสและถอดรหัสข้อมูลใช้เวลามาก เพราะว่าอัลกอริทึมที่ใช้ค่อนข้างจะสลับ-ซับซ้อนมาก

2. ขนาดของข้อมูลหลังจากทำการเข้ารหัสแล้ว มีการเปลี่ยนแปลงมาก หรือ ข้อมูลหลังจากทำการเข้ารหัสแล้ว จะมีขนาดใหญ่กว่าเดิมมากขึ้น เพราะฉะนั้นจึงเป็นปัญหาในการใช้งาน

2.3 ลักษณะของการเข้ารหัสข้อมูลที่ดี (Characteristics of Good Cipher)

1. ระดับความปลอดภัยของข้อมูลที่ได้ ควรจะแปรผันกับความยากของการเข้ารหัสข้อมูล นั้นคือวิธีการเข้ารหัสนั้นมีความสลับซับซ้อนมากก็ควรให้ระดับความปลอดภัยของข้อมูลที่สูงด้วย

2. ไม่ควรมีข้อจำกัดในการเลือกใช้กุญแจเข้ารหัส และในการเลือกใช้วิธีการเข้าสำหรับข้อความลักษณะใดลักษณะหนึ่ง เพราะหากการเลือกใช้นั้นมีความยากและไม่สะดวกแล้วการเข้ารหัสนั้นไม่เป็นที่นิยมใช้

3. กระบวนการนำวิธีการเข้ารหัสไปใช้จะต้องมีความสะดวกและง่าย เพราะว่าหากการเข้ารหัสยากมากเกินไปแล้วอาจทำให้เกิดความผิดพลาดในระหว่างกระบวนการพัฒนาและนำไปใช้งานได้

4. ความผิดพลาดของการเข้ารหัส ณ จุดใดจุดหนึ่งของข้อมูลจะต้องไม่ขยายไปสู่ส่วนอื่นๆ

5. เมื่อเสร็จสิ้นจากการเข้ารหัสข้อมูลแล้ว ขนาดของข้อมูล Cipher Text ต้องมีขนาดไม่ใหญ่กว่าขนาดของ Clear Text

2.4 ประโยชน์ของการเข้ารหัส

นอกจากการเข้ารหัสจะเป็นการทำให้ข้อมูลถูกสับเปลี่ยนเพื่อไม่ให้ผู้อื่นสามารถเข้าใจ และใช้ประโยชน์จากข้อมูลนั้นได้แล้ว ยังมีประโยชน์ในด้านอื่นๆ อีก เช่น สามารถนำมาประยุกต์ ใช้ในการตรวจสอบว่า ผู้ที่กำลังใช้เครือข่ายคอมพิวเตอร์ หรือทำรายการบน Webpage เป็นผู้ที่นอกจากเราต้องการติดต่อจริงไม่ใช่ผู้อื่นที่แอบอ้างเข้ามาใช้ระบบ นอกจากนี้ยังสามารถนำไปใช้เป็นลายเซ็นดิจิตอลในการระบุหรือยืนยันว่า e-mail หรือแฟ้มข้อมูลที่ส่งไปให้ผู้อื่นนั้นมาจากเราจริงๆ ได้อีกด้วย

2.5 อัลกอริทึมที่ใช้ในการเข้ารหัส

อัลกอริทึมที่ใช้ในการเข้ารหัสมีหลายแบบ แต่ละแบบมีการจัดการกับข้อมูลที่แตกต่างกัน โดยจะแบ่งเป็น 2 ประเภทหลักๆ ตามประเภทของการเข้ารหัส คือ การเข้ารหัสแบบสมมาตร โดยอัลกอริทึมที่ใช้กันอย่างแพร่หลาย เช่น DES, Triple DES, AES, Blowfish และ Twofish เป็นต้น ส่วนการเข้ารหัสแบบอสมมาตรจะมีการใช้อัลกอริทึมต่างๆ เช่น RSA, Diffie-Hellman และ ElGamal เป็นต้น ซึ่งในที่นี้จะกล่าวถึงอัลกอริทึมแบบ Twofish

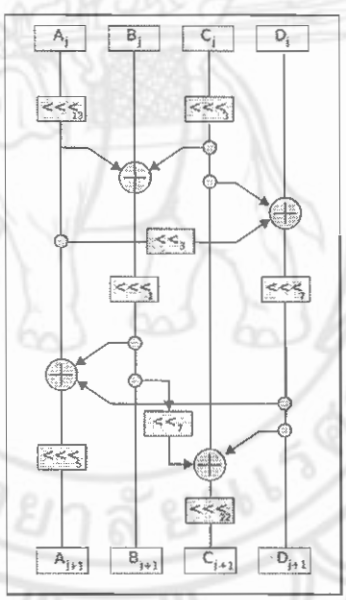
2.5.1 อัลกอริทึมแบบ AES

อัลกอริทึม Advanced Encryption Standard หรือ AES ที่ได้รับการตัดสินจาก NIST ให้ชนะเลิศกีแข่งขันในปี ค.ศ. 1998 สร้างขึ้นโดย Joan Daemen และ Vincent Rijmen ซึ่งเป็นนักวิจัยชาวเบลเยี่ยม โดยอัลกอริทึมนี้เดิมทีใช้ชื่อว่า Rijndael (Rijmen & Daemen) อัลกอริทึมนี้จะยังคงเป็นแบบ Block Cipher โดยใช้บล็อกข้อมูลขนาด 128 บิต 196 บิต และ 256 บิต โดยสามารถใช้คีย์ได้ยาวถึง 128 บิต 196 บิต และ 256 บิต โดยอัลกอริทึมนี้ได้รับการออกแบบให้มีการทำงานที่เหมาะสมกับโพรเซสเซอร์รุ่นใหม่ ๆ และสามารถใช้งานกับ Smart Card ได้ เพราะใช้หน่วยความจำ

น้อย อัลกอริทึม Rijndael จะใช้ Round Function ที่สามารถเลือกได้ว่าจะทำ 10,I2 หรือ 14 ครั้ง โดยมีการทำงานอยู่ 4 การทำงานย่อย คือ Byte Sub ก็คือการใช้ S-Boxes ในการสลับข้อมูลระหว่าง 2 บล็อก ShiftRow คือการสลับข้อมูลระหว่างแถว Mix Column คือการ Shift ข้อมูลในแต่ละ Column และสุดท้ายคือ Key Addition คือการนำมาบวกกับคีย์ ซึ่งการทำงานทั้งหมด เป็นการทำงานที่ง่าย มีจำนวนครั้งของการทำงานน้อย ทำงานได้เร็ว และใช้หน่วยความจำน้อย

2.5.2 อัลกอริทึมแบบ Serpent

สร้างขึ้นในปี ค.ศ. 1998 โดย Ross Anderson, Eli Biham และ Lars Knudsen มีแนวคิดมาจากอัลกอริทึมแบบ AES โดยมีขนาดของบล็อกเป็น 128 บิต และขนาดของคีย์เป็น 128, 192 และ 256 บิต รวมทั้งมีการเข้ารหัสทั้งหมด 32 รอบ ในการเข้ารหัสจะทำการแบ่งข้อมูลเป็น 4 ส่วน ส่วนละ 32 บิต ซึ่งจะทำใน S-boxes ไปพร้อมๆ กันดังรูปที่ 2.3 อัลกริทึมแบบ Serpent นี้จะมีความปลอดภัยมากกว่าอัลกอริทึมแบบอื่นที่พัฒนามาจาก AES เนื่องจากมีการเข้ารหัสถึง 32 รอบด้วยกัน



รูปที่ 2.3 การเข้ารหัสด้วยอัลกอริทึมแบบ Serpent [2]

2.5.3 อัลกอริทึมแบบ RC2

อัลกอริทึมแบบ RC2 พัฒนาขึ้นมาในปี ค.ศ. 1987 โดย Ron Rivest หนึ่งในผู้ร่วมคิดค้นอัลกอริทึม RSA ซึ่งใช้ขนาดของข้อมูล 64 บิต และมีขนาดของคีย์ตั้งแต่ 8 บิต ถึง 128 บิต ส่วนใหญ่จะใช้ที่ 64 บิต มีการทำงาน 18 รอบด้วยกัน โดยทำการ MXING 16 รอบ และทำการ MASHING 2 รอบด้วยกัน

2.5.4 อัลกอริทึมแบบ CAST-6 (CAST-256)

CAST-256 หรือ CAST-6 เป็นที่รู้จักในปี 1998 โดย Carlisle Adams และ Stafford Tavares ซึ่งเป็นหนึ่งในกลุ่มของอัลกอริทึมแบบ AES ที่ได้รับการยอมรับจาก NIST เป็นการเข้ารหัสแบบบล็อกข้อมูลที่พัฒนามาจากอัลกอริทึมแบบ CAST-128 รวมทั้งการใช้ S-Boxes แต่ใช้ขนาดของบล็อกข้อมูลถึง 128 บิต และสามารถใช้คีย์ได้ทั้ง 128 บิต 192 บิต และ 256 บิต เข้ารหัสทั้งหมด 48 รอบด้วยกัน

2.5.5 อัลกอริทึมแบบ Blowfish

อัลกอริทึมแบบ Blowfish พัฒนาโดย Bruce Schneier ในปี 1993 โดยเขาเป็นเป็นที่ปรึกษาอิสระและผู้เชี่ยวชาญด้านการเข้ารหัส ซึ่ง Blowfish ได้รับการต้อนรับอย่างดี ในฐานะตัวเลือกหนึ่งของ DES อัลกอริทึม Blowfish ได้รับการออกแบบมาเพื่อให้สร้างได้ง่าย และมีความเร็วในการทำงานสูง เป็นอัลกอริทึมที่ใช้พื้นที่ในการทำงานน้อยมาก เพียง 5K ก็สามารถทำงานได้ สิ่งที่น่าสนใจใน Blowfish คือ การใช้คีย์ที่มีการเปลี่ยนค่าความยาวได้ตั้งแต่ 32 บิต ถึง 448 บิต แต่ในทางปฏิบัติมักใช้กันที่ 128 บิต มีการวนรอบทั้งหมด 16 รอบ

อัลกอริทึมแบบ Blowfish มีการใช้ S-Boxes และ XOR เช่นเดียวกับ DES แต่ยังมีการใช้การบวกร่วมด้วย แต่สิ่งที่ต่างจาก DES คือ ใน DES จะใช้ S-Boxes แบบความยาวคงที่ แต่ใน Blowfish จะสามารถเปลี่ยนค่าความยาวได้ ส่วนของการสร้างคีย์ย่อยนั้น Blowfish มีการนำเอาอัลกอริทึมแบบ Blowfish เองมาใช้ในการสร้างคีย์ย่อยและ S-Boxes ด้วย โดยจะต้องมีการวนทั้งหมด 521 รอบในการทำงานเพื่อสร้างคีย์ย่อยและ S-Boxes และนั่นทำให้ Blowfish มีความปลอดภัยสูง แต่ไม่เหมาะที่จะใช้กับงานที่จะต้องมีการเปลี่ยนค่า Secret Key บ่อย ๆ

2.5.6 อัลกอริทึมแบบ Twofish

Twofish เป็นวิธีการใหม่ในการเข้ารหัสข้อมูลโดยนำข้อมูลที่สามารถอ่านแล้วเข้าใจ ได้หรือใช้งานได้ ไม่ว่าจะเป็นข้อมูลเสียง รูปภาพ อักษรมากระทำการในลักษณะแบ่งเป็นบล็อกข้อมูลแล้วปฏิบัติการ โดยวิธีการแปลงสภาพให้อยู่ในสภาพที่ไม่สามารถอ่านเข้าใจได้หรือใช้งานได้ และสามารถแปลงกลับมาเป็น ข้อมูลที่อ่านเข้าใจได้หรือใช้งานได้อีกครั้ง เป็นการเพิ่มความปลอดภัยให้กับข้อมูลที่มีความสำคัญในทุกองค์กร และกับทุกคนที่มีความสนใจนำไปใช้เข้ารหัสข้อมูลเพื่อเพิ่มความปลอดภัยของข้อมูล

ในการแข่งขันกันที่ทาง NIST ได้จัดขึ้นในปี 1997 (พ.ศ. 2540) ซึ่งมี 5 อัลกอริทึมจาก 15 อัลกอริทึมที่ถูกเลือก ได้แก่ AES, MARS, Rijndael, Serpent และ Twofish โดย Twofish โดยการออกแบบของ Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall และ Niels Ferguso ได้รับเลือกเป็นอันดับที่สองรองจาก AES เพราะการเข้ารหัสแบบ Twofish มีขนาดคีย์ในการเข้ารหัสถึง 256 บิต จึงใช้เวลานาน แต่ในความยาวของคีย์นี้ทำให้ Twofish มีความแข็งแรงในการเข้ารหัส และมีความยืดหยุ่นสูงเช่นกัน จึงได้รับเลือกเป็นอันดับที่สอง ปัจจุบันเริ่มมีคนสนใจ

และใช้กันอย่างแพร่หลายเพิ่มมากขึ้น เพราะความปลอดภัยในการเข้ารหัสที่มีอยู่มาก อีกทั้ง Twofish ยังไม่มีการจดลิขสิทธิ์ใดๆ จึงมีการนำมาใช้กันมากกว่าแบบอื่นๆ

Twofish เป็นการเข้ารหัสแบบ Symmetric key มีการทำงานเป็น Block cipher โดยแต่ละบล็อกมีขนาด 128 บิต คือ มีการเข้ารหัสและถอดรหัส 128 ชั้น และมีขนาดของคีย์เป็น 256 บิต โดย Twofish ถูกใช้กับ microprocessors ขนาดใหญ่, 8-bit smart card microprocessors และ hardware อื่นๆ Counterpane Labs ได้ทำการทดสอบความสามารถของ Twofish พบว่ามีการป้องกันการเข้าถึงข้อมูลถึง 16 รอบ ซึ่งใน 5 รอบจะมีจุดอ่อนอยู่ แต่ Twofish จะมีความปลอดภัยสูงสุดเมื่อทำงานครบ 16 รอบ ซึ่งจะเร็วกว่า AES อีกด้วย Twofish ยังสามารถจัดการกับคีย์เพื่อเพิ่ม-ลดความเร็วในการเข้ารหัสได้ มีการใช้หน่วยความจำที่น้อย และยังมีการประยุกต์อัลกอริทึมให้มีความยืดหยุ่นเพิ่มขึ้นด้วย กระบวนการเข้ารหัสแบบ Twofish มีการใช้ฟังก์ชันต่างๆ ทั้งเพื่อสร้างบล็อกข้อมูล และสร้างคีย์ ซึ่งฟังก์ชันที่ใช้มีดังนี้

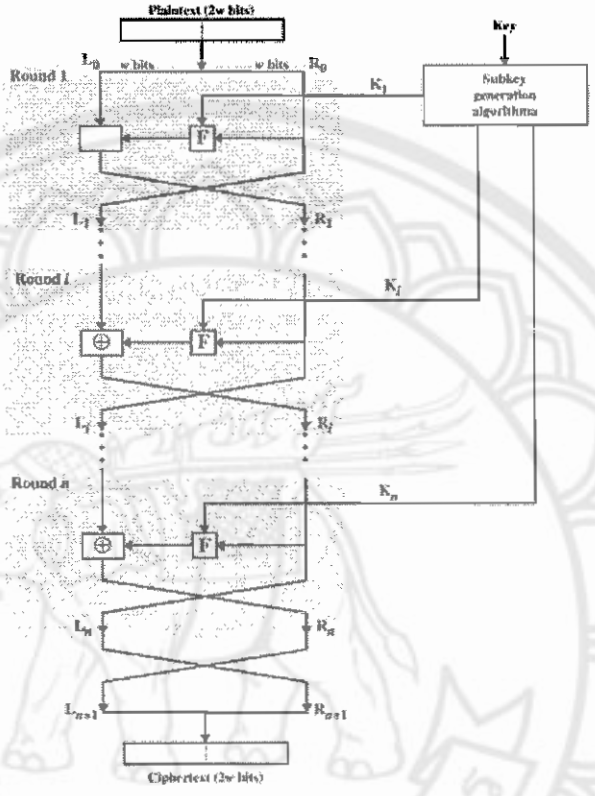
Feistel Network หรือฟังก์ชัน F สร้างโดย Horst Feistel โดยจะป้อนอินพุตเข้าสู่อัลกอริทึมในการเข้ารหัสในลักษณะที่เป็นกลุ่มของข้อมูลที่มีความยาวเท่า ๆ กัน หรือเรียกว่า บล็อก (Block) โดยจะมีความยาวของกลุ่มข้อมูลเท่ากับ 2w โดยใช้คีย์ K จากนั้นกลุ่มข้อมูลจะแบ่งออกเป็น 2 ส่วนเท่าๆ กัน คือ L_0 และ R_0 จากนั้นข้อมูลทั้ง 2 ส่วน จะป้อนเข้าสู่การประมวลผล โดยหลักของการเข้ารหัสในแบบนี้ จะอาศัยการประมวลผลซ้ำ ๆ กัน เพื่อสร้างความซับซ้อน โดยในแต่ละรอบของการประมวลผล อินพุต L_{i-1} และ R_{i-1} จะได้มาจากผลของการประมวลผลก่อนหน้า และคีย์ย่อย K_i ก็จะได้มาจากคีย์ K ซึ่งโดยทั่วไปแล้ว คีย์ย่อยที่จะใช้ในแต่ละรอบของกรประมวลผลจะต้องไม่ซ้ำกัน ดังรูปที่ 2.4

ในแต่ละรอบของการประมวลผลจะประกอบด้วยการใช้ Round Function F ทำกับข้อมูล R และนำผลลัพธ์ที่ได้ไป XOR กับข้อมูล L โดย Round Function ที่ใช้จะต้องเป็นฟังก์ชันเดียวกันทั้งหมด แต่เปลี่ยนคีย์ไปเรื่อย ๆ เท่านั้น ซึ่งโดยทั่วไปแล้ว ความซับซ้อน หรือ ความยากในการแกะรหัส จะขึ้นอยู่กับ การออกแบบในส่วนต่าง ๆ โดยมีข้อพิจารณาในการออกแบบอัลกอริทึมในแต่ละส่วนดังนี้

- ขนาดของบล็อกข้อมูล บล็อกข้อมูลที่มีขนาดใหญ่ จะมีความปลอดภัยมากขึ้น แต่จะทำให้ความเร็วในการเข้าและถอดรหัสลดลงด้วย ปกติจะถือว่าบล็อกที่มีขนาด 64 บิต ถือว่ามีความเหมาะสม
- ขนาดของคีย์ ขนาดคีย์ที่มีขนาดใหญ่ จะมีความปลอดภัยมากขึ้น แต่จะทำให้ความเร็วในการเข้าและถอดรหัสลดลงด้วย ปัจจุบันในอัลกอริทึมสมัยใหม่จะถือว่าคีย์ที่มีความยาว 128 ถือว่ามีความปลอดภัยเพียงพอ
- จำนวนครั้งของการประมวลผล ยิ่งทำมากรอบจะยิ่งถอดรหัสได้ยากขึ้น ปกติจะถือว่าประมาณ 16 รอบ มีความเหมาะสม

- อัลกอริทึมที่ใช้ในการสร้างคีย์ย่อย ยิ่งอัลกอริทึมซับซ้อนมากจะยิ่งทำให้การแกะรหัสยากยิ่งขึ้น
- ฟังก์ชัน Round ที่มีความซับซ้อนจะยิ่งทำให้การแกะรหัสยากขึ้น

สำหรับการถอดรหัสแล้ว จะมีกระบวนการเดียวกับการเข้ารหัส แต่จะใช้คีย์ย่อยย้อนกลับ



รูปที่ 2.4 การทำงานของฟังก์ชัน Feistel Network [3]

Substitution boxes (S-boxes) เป็นองค์ประกอบพื้นฐานของการเขารหัสแบบสมมาตร ในการทำงานของบล็อกข้อมูลไม่มีความชัดเจนของความสัมพันธ์ระหว่าง Plaintext และ Cipher text ซึ่งในหลายกรณีนั้นการใช้ S-boxes สามารถช่วยทำให้มีความชัดเจนขึ้นได้ S-boxes เริ่มใช้ครั้งแรกกับอัลกอริทึมแบบ DES ในส่วนของ Twofish มีการใช้งานอยู่ 4 ส่วนที่มีความแตกต่างกันได้แก่ bijective, key-dependent, 8-8 bit S-boxes โดย S-boxes ถูกสร้างเพื่อใช้ใน 8-8 bit และ key material

Maximum Distance Separable (MDS) เป็นเมตริกซ์ที่มีการนำข้อมูลของฟิลด์แบบ Linear จากฟิลด์ a ไปยังฟิลด์ b ซึ่งจะได้เป็นเวกเตอร์ที่เป็นองค์ประกอบของ $a + b$ ด้วยคุณสมบัติของตัวเลขที่น้อยที่สุดของเวกเตอร์แบบ non-zero ซึ่งมีค่าน้อยสุดเท่ากับ b+1 โดยการทำ MDS mapping สามารถแสดงโดยเมตริกซ์ MDS ขนาด a x b

Pseudo-Hadamard Transforms (PHT) เป็นกระบวนการของการ Mixing เพื่อรันโปรแกรมได้เร็วขึ้น โดยให้อินพุท 2 ตัวคือ a และ b ขนาด 32 บิต โดย

$$a' = a + b \bmod 2^{32}$$

$$b' = a + 2b \bmod 2^{32}$$

Whitening เป็นเทคนิคของการทำ XOR คีย์ ก่อนรอบแรก และหลังรอบสุดท้ายของการทำฟังก์ชัน F ถูกใช้โดย Merkle และสร้างโดย Rivest สำหรับ DES-X ซึ่งจะเป็นการเพิ่มความปลอดภัยให้แก่คีย์ โดยการ XOR คีย์ย่อย (subkey) ขนาด 128 บิต โดยจะคำนวณหมือนกับการใช้ round subkey แต่จะไม่ใช้กับทุกข้อมูล

Key Schedule เป็นค่ากลาง โดยได้จากการนำคีย์กลับเข้าสู่กระบวนการ round subkey ซึ่งข้อมูลต้นฉบับจะสามารถใช้กับคีย์ที่มีขนาดใหญ่ได้ และมีความซับซ้อน โดยความง่ายในการวิเคราะห์เกี่ยวกับ Key Schedule จะคล้ายกับพื้นฐานของการทำ Round Function

2.5.6.1 ขั้นตอนการเข้ารหัสแบบ Twofish

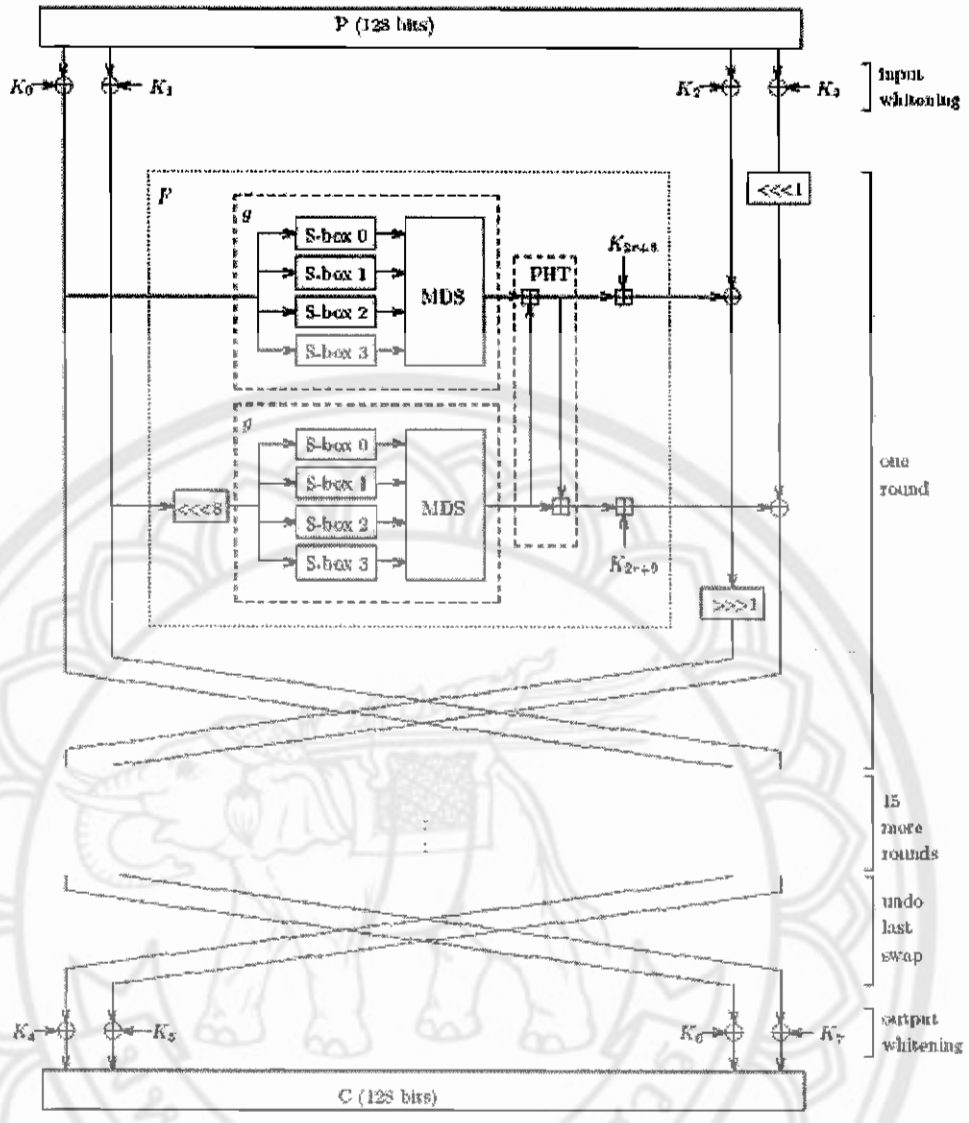
ในการเข้ารหัสแบบ Twofish จะแสดงเป็นแบบบล็อกข้อมูลตามรูปที่ 2.5 มีการเข้ารหัสทั้งหมด 16 รอบโดยมีโครงสร้างแบบ Feistel มีการเพิ่มกระบวนการ Whitening เข้าไปในส่วน input และ output แล้วจึงเข้าสู่ฟังก์ชัน F ทีละ 1 บิต แต่ต้องมีการกลับบิตก่อนการทำ whitening โดย plaintext จะถูกแบ่งเป็น 4 ส่วน ส่วนละ 32 บิต input ของกระบวนการ whitening จะถูก XOR กับคีย์ 4 ตัวด้วยกัน แล้วจึงทำการเข้ารหัสอีก 16 รอบ ในแต่ละรอบจะมี 2 ส่วนที่ถูกใช้ไปเป็น input ของฟังก์ชัน g (หนึ่งในนั้นจะถูกหมุนไป 8 บิตแรก) ฟังก์ชัน g จะประกอบไปด้วย S-boxes 4 บล็อกขนาด 4 ไบต์ แล้วจะเข้าสู่เมตริกซ์ MDS ผลลัพธ์ที่ได้จะนำมารวมกันโดยใช้ PHT และคีย์ 2 ตัวจะถูกนำมารวมกัน และผลลัพธ์ที่ได้ทั้งสองจะนำมา XOR กับคีย์ทางขวา (โดยหนึ่งในคีย์นั้นจะถูกหมุนไปทางหมุนไปทางซ้ายบิตแรก 1 บิต และบิตที่เหลือจะถูกหมุนไปทางขวาในภายหลัง) หลังจากนนั้นจะถูกสลับกันระหว่างคีย์ 4 ตัวในรอบถัดไป ทำทั้งหมด 16 รอบ หลังจากรอบสุดท้ายแล้วจสลับกลับมาอีก 1 ครั้ง แล้วคีย์ทั้ง 4 ตัวจะถูกนำมา XOR กับคีย์อีก 4 ตัวเพื่อสร้างเป็น Ciphertext

การแสดงการคำนวณจะแสดงได้โดยขั้นแรกจะมี Plaintext ขนาด 16 ไบต์ คือ $p_0, p_1, \ldots, p_{15}$ โดยจะถูกแบ่งเป็น 4 ส่วน โดยส่วนแรกมีขนาด 32 บิด คือ P_0, P_1, P_2, P_3 แสดงได้ดังนี้

$$P_i = \sum_{j=0}^{3} p_{(4i+j)} \cdot 2^{8j} \qquad i = 0,\ldots,3$$

ขั้นตอนของการ Input whitening จะทำการ XOR กับคีย์ที่มีการแบ่งแล้ว แสดงได้ดังนี้

$$R_{0,i} = P_i \oplus K_i \qquad i = 0,\ldots,3$$



รูปที่ 2.5 กระบวนการการเข้ารหัสแบบ Twofish [3]

ในแต่ละรอบของทั้ง 16 รอบในการเข้ารหัส 2 ส่วนแรกจะถูกใช้ไปเป็น input ของฟังก์ชัน F ส่วนที่ 3 จะนำไป XOR กับผลลัพธ์อันแรกที่ได้จากฟังก์ชัน F แล้วหมุนไปทางขวา 1 บิต ส่วนที่ 4 จะหมุนซ้ายไป 1 บิตแล้ว XOR กับ Output ที่ 2 ของฟังก์ชัน F แล้วจึงนำทั้งสองฝั่งมาแลกเปลี่ยนกัน ดังนี้

$$(F_{r,0}, F_{r,1}) = F(R_{r,0}, R_{r,1}, r)$$
$$R_{r+1,0} = ROR(R_{r,2} \oplus F_{r,0}, 1)$$
$$R_{r+1,1} = ROL(R_{r,3}, 1) \oplus F_{r,1}$$
$$R_{r+1,2} = R_{r,0}$$

$$R_{r+1,3} = R_{r,1}$$

โดยที่ r = 0, 1, …, 15 และโดยที่ ROR และ ROL เป็นฟังก์ชันในการหมุน argument แรกทางซ้ายหรือขวา โดยจำนวนของบิตจะแสดงโดย argument ที่สอง

Output จากขั้นตอน whitening จะทำการกลับคีย์คืนหลังรอบสุดท้าย แล้ว XOR กับกับข้อมูล 4 ส่วนซึ่งจะแสดงดังนี้

$$C_i = R_{16,(i+2) \bmod 4} \oplus K_{i+4} \qquad i = 0,\ldots,3$$

ข้อมูล 4 ส่วนหลังจากการเข้ารหัสนี้จะมีขนาด 16 ไบต์ คือ c0, c1, …, 15 คล้ายกับใช้การ little-endian กับ plaintext

$$c_i = \left\lfloor \frac{C_{\lfloor i/4 \rfloor}}{2^{8(i \bmod 4)}} \right\rfloor \bmod 2^8 \qquad i = 0,\ldots,15$$

กระบวนการทำงานของแต่ละฟังก์ชันมีดังนี้

ฟังก์ชัน F (Function F) เป็นคีย์ขนาด 64 บิต จะถูกทำให้เป็น 3 ส่วน คือ input 2 ส่วน ได้แก่ R_0, R_1 และจำนวนรอบ (r) จะใช้เพื่อเลือก subkey R_0 แล้วเข้าสู่ฟังก์ชัน g และถูกแทนที่ด้วย T_0 ส่วน R_1 จะหมุนซ้ายไป 8 บิต แล้วเข้าสู่ฟังก์ชัน g แทนที่ด้วย T_1 ผลลัพธ์จากฟังก์ชัน g จะนำไปรวมกันใน PHT แสดงได้ดังนี้

$$T_0 = g(R_0)$$
$$T_0 = g(ROL(R_1,8))$$
$$F_0 = (T_0 + T_1 + K_{2r+8}) \bmod 2^{32}$$
$$F_1 = (T_0 + 2T_1 + K_{2r+9}) \bmod 2^{32}$$

โดย F_0, F_1 เป็นผลลัพธ์จากฟังก์ชัน F โดยในการวิเคราะห์จะใช้ F' นอกจากนั้นยังไม่เป็นการเพิ่มขนาดของบล็อกข้อมูล

ฟังก์ชัน g (Function g) เป็นหัวใจหลักของการเข้ารหัสแบบ Twofish โดยที่ X จะถูกใส่เข้าไปและถูกแยกไปเป็น 4 ไบต์ แต่ละไบต์จะทำงานผ่านคีย์ย่อยของ S-boxes โดย S-boxes นี้จะเป็น bijective โดยใส่ input เข้าไป 8 บิต และได้ output ออกมา 8 บิตเช่นกัน ผลลัพธ์ทั้ง 4 จะถูกแปลงไปเป็นเวกเตอร์ของขนาดทั้ง 4 ตัวจากการเข้าฟังก์ชันทั้ง 2 มากกว่า $GF(2^8)$ และถูก multiply โดยเมตริกซ์ MDS ขนาด 4 x 4 ใช้ขนาดของฟิลด์ $GF(2^8)$ เป็นตัวคำนวณ เวกเตอร์ที่เป็นผลลัพธ์ถูกแปลงไปเป็นขนาด 32 บิต ซึ่งเป็นผลลัพธ์ของฟังก์ชัน g นั่นเอง แสดงได้ดังนี้

$$x_1 = \lfloor X / 2^{8i} \rfloor \bmod 2^8 \qquad i = 0,\ldots,3$$
$$y_1 = s_i[x_i] \qquad i = 0,\ldots,3$$

$$\begin{pmatrix} z_0 \\ z_1 \\ z \\ z_3 \end{pmatrix} = (MDS) \cdot \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$Z = \sum_{i=0}^{3} z_i \cdot 2^{8i}$$

โดย s_i เป็นคีย์ย่อยของ S-boxes และ Z เป็นผลลัพธ์จากฟังก์ชัน g ซึ่งกล่าวไว้ชัดเจนว่าค่า จะต้องอยู่ระหว่างค่าที่สอดคล้องกันนั้นกับค่าของ GF(2^8) โดย GF(2^8) คือ GF(2)[x]/v(x) ที่ซึ่ง v(x) = $x^8 + x^6 + x^5 + x^3 + 1$ เป็นองค์ประกอบหักของกำลัง 8 มี $a_i = \sum_{i=0}^{7} a_i x^i$ โดย $a_i \in GF(2)$ ที่ระบุไว้ด้วยค่าไบต์ $\sum_{i=0}^{7} a_i 2^i$ เมตริกซ์ MDS

$$MDS = \begin{pmatrix} 01 & EF & 5B & 5B \\ 5B & EF & EF & 01 \\ EF & 5B & 01 & EF \\ EF & 01 & EF & 5B \end{pmatrix}$$

Key Schedule จะมีการเตรียมคีย์ไว้ 40 ตัว คือ $K_0, K_1, \ldots, K_{39}$ และคีย์ย่อยของ S-boxes อีก 4 ตัว ที่ถูกใช้โดยฟังก์ชัน g ในการเข้ารหัสแบบ Twofish จะใช้คีย์อยู่ 3 ขนาดคือ N = 128, N = 192 และ N = 256 ส่วนคีย์ที่มีขนาดน้อยกว่า 256 บิตจะถูกเพิ่มด้วยศูนย์จนกว่าจะมีขนาดใหญ่กว่าขนาดของคีย์ที่กำหนดไว้ จะให้ k = N/64 คีย์ M ประกอบไปด้วย 8k ไบต์ คือ $m_0, m_1, \ldots, m_{8k-1}$ ไบต์แรกจะถูกแปลงไปเป็น 2k ของแต่ละ 32 บิต

$$M_i = \sum_{j=0}^{3} m_{(4i+j)} \cdot 2^{8j} \qquad i = 0,\ldots,2k\text{-}1$$

แล้วจะเข้าสู่ 2 ส่วนที่เป็นเวกเตอร์ขนาด k

$$M_e = (M_0, M_2, \ldots, M_{2k-2})$$

$$M_o = (M_1, M_3, \ldots, M_{2k-1})$$

เวกเตอร์ในส่วนที่สามที่มีความยาว k มาจากคีย์ และจะเสร็จสิ้นโดยการทำคีย์แบบไบต์ในกลุ่มของ 8 จะแสดงด้วยเวกเตอร์ GF(2^8) และนำมา multiply กับเมตริกซ์ขนาด 4 x 8 ที่มาจาก RS-code แต่ละผลลัพธ์ขนาด 4 ไบต์จะถูกแปลไปเป็น 32 บิต ซึ่งจะได้ 3 เวกเตอร์ด้วยกัน แสดงดังนี้

$$\begin{pmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \end{pmatrix} = (RS) \cdot \begin{pmatrix} m_{8i} \\ m_{8i+1} \\ m_{8i+2} \\ m_{8i+3} \\ m_{8i+4} \\ m_{8i+5} \\ m_{8i+6} \\ m_{8i+7} \end{pmatrix}$$

$$S_i = \sum_{j=0}^{3} s_{i,j} \cdot 2^{8j}$$

โดย $i = 0, \ldots, k-1$ และ $S = (S_{k-1}, S_{k-2}, \ldots, S_0)$

สำหรับเมตริกซ์ RS จะแสดง $GF(2^8)$ แทนด้วย $GF(2)[x]/w(x)$ ที่ซึ่ง $w(x) = x^8 + x^6 + x^3 + x^2 + 1$ โดยจะทำงานคล้ายกับเมตริกซ์ MDS เมตริกซ์ RS แสดงดังนี้

$$RS = \begin{pmatrix} 01 & A4 & 55 & 87 & 5A & 58 & DB & 9E \\ A4 & 56 & 82 & F3 & 1E & C6 & 68 & E5 \\ 02 & A1 & FC & C1 & 47 & AE & 3D & 19 \\ A4 & 55 & 87 & 5A & 58 & DB & 9E & 03 \end{pmatrix}$$

Additional Key Lengths ในการเข้ารหัสแบบ Twofish สามารถยอมรับขนาดคีย์ได้ถึง 256 บิต สำหรับคีย์ที่มีขนาดไม่ถึงตามกำหนดจะมีการเพิ่มศูนย์ต่อท้าย เช่นถ้าคีย์มีขนาด 80 บิต คือ m_0, m_1, …, m_9 ดังนั้นจะทำการเพิ่มคีย์โดยที่ $m_j = 0$ เมื่อ i = 10, …, 15 ซึ่งจะได้คีย์ขนาด 128 บิต

ฟังก์ชัน h (Function h) จะมีการใส่ input เข้าไป 2 input คือ X และ $L = (L_0, \ldots, L_{k-1})$ มีขนาด 32 บิต ของความยาว k และสร้าง output ออกมา 1 ตัว ฟังก์ชันนี้จะทำงานทั้งหมด k ช่วงด้วยกัน ในแต่ละช่วงจะนำ 4 ไบต์เข้าสู่การทำ S-boxes และ XOR กับ L และในช่วงสุดท้ายจะมีการทำ S-boxes อีกครั้ง แล้วนำมา multiply กับเมตริกซ์ MDS แสดงดังรูปที่ 2.6 จะอธิบายได้ดังสมการต่อไปนี้

$$l_{i,j} = \lfloor L_i / 2^{8j} \rfloor \bmod 2^8$$
$$x_i = \lfloor X / 2^{8j} \rfloor \bmod 2^8$$

สำหรับ i = 0, …, k-1 และ j = 0, …, 3 แล้วลำดับของการแทนที่และการ XOR จะถูกประยุกต์ไป

$$y_{k,j} = x_j \qquad j = 0, \ldots, 3$$

ที่ k = 4 จะได้

$$y_{3,0} = q_1[y_{4,0}] \oplus 1_{3,0}$$
$$y_{3,1} = q_0[y_{4,1}] \oplus 1_{3,1}$$
$$y_{3,2} = q_0[y_{4,2}] \oplus 1_{3,2}$$
$$y_{3,3} = q_1[y_{4,3}] \oplus 1_{3,3}$$

ที่ k = 3 จะได้

$$y_{2,0} = q_1[y_{3,0}] \oplus 1_{2,0}$$
$$y_{2,1} = q_1[y_{3,1}] \oplus 1_{2,1}$$
$$y_{2,2} = q_0[y_{3,2}] \oplus 1_{2,2}$$
$$y_{2,3} = q_1[y_{3,3}] \oplus 1_{2,3}$$

ในทุกกรณีจะได้

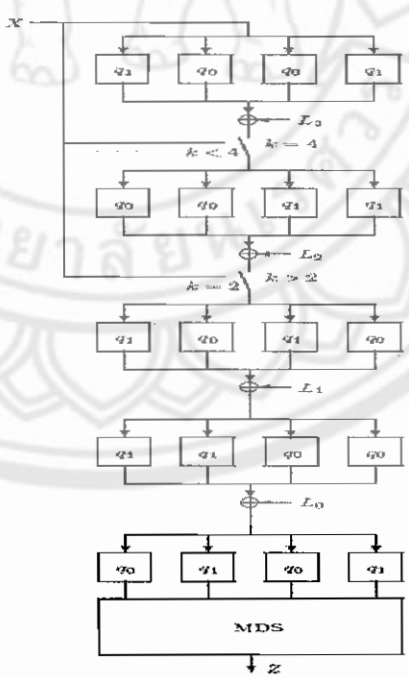
$$y_0 = q_1[q_0[q_0[y_{2,0}] \oplus l_{1,0}] \oplus l_{0,0}]$$
$$y_1 = q_0[q_0[q_1[y_{2,1}] \oplus l_{1,1}] \oplus l_{0,1}]$$
$$y_2 = q_1[q_1[q_0[y_{2,2}] \oplus l_{1,2}] \oplus l_{0,2}]$$
$$y_3 = q_0[q_1[q_1[y_{2,3}] \oplus l_{1,3}] \oplus l_{0,3}]$$

เมื่อ q_0 และ q_1 ถูกกำหนดค่าไว้แล้ว มีขนาด 8 บิต โดยเวกเตอร์ที่เป็นผลลัพธ์จากการการ multiply ระหว่าง y_i กับเมตริกซ์.MDS จะได้เป็นฟังก์ชัน g ดังนี้

$$\begin{pmatrix} z_0 \\ z_1 \\ z \\ z_3 \end{pmatrix} = (MDS) \cdot \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$Z = \sum_{i=0}^{3} z_i \cdot 2^{8i}$$

โดยที่ Z เป็นผลลัพธ์ของฟังก์ชัน h



รูปที่ 2.6 ฟังก์ชัน h [3]

Key-dependent S-boxes เราสามารถกำหนดค่า S-boxes ในฟังก์ชัน g โดย

$$g(X) = h(X, S)$$

เมื่อ i = 0, …, 3 และคีย์ย่อยของ S-boxes หรือ s_i จะอยู่ในรูปของการ mapping จาก x_i ไปยัง y_i ในฟังก์ชัน h ที่ซึ่ง L จะเท่ากับเวกเตอร์ S ที่มาจากคีย์นั้น

การคำนวณหาค่า *Kj* จะถูกแสดงดังสมการต่อไปนี้

$$\rho = 2^{24} + 2^{16} + 2^8 + 2^0$$
$$A_i = h(2i\rho, M_e)$$
$$B_i = ROL(h((2i+1)\rho, M_o), 8)$$
$$K_{2i} = (A_i + 2B_i) \bmod 2^{32}$$
$$K_{2i+1} = ROL((A_i + 2B_i) \bmod 2^{32}, 9)$$

โดย ρ ถูกใช้ในการสำเนาค่าไบต์คือจะมีคุณสมบัติเป็น i = 0, …, 255 และ *ip* ประกอบไปด้วย 4 ไบต์เท่าๆ กัน แต่ละส่วนจะมีค่า *i* สำหรับ A_i จะมีค่าเป็น 2i และ argument ที่สองของฟังก์ชัน h คือ M_e ส่วน B_i ถูกคำนวณโดยการใช้ค่าเป็น 2i + 1 คล้ายเป็นการหมุนไป 8 บิต ค่า Ai และ Bi จะมีการรวมกันกับ PHT ซึ่งผลลัพธ์ที่ได้จะเพิ่มให้มีการหมุนไปอีก 9 บิต ผลลัพธ์ทั้งสองคือผลจากการคำนวณเพื่อหาค่าคีย์

การเปลี่ยนลำดับของ q_0 และ q_1 จะกำหนดให้มีค่า 8 บิต ถูกสร้างจากสี่ส่วนที่มีความต่างกันแต่ละส่วนมีขนาด 4 บิต สำหรับค่า input x ถูกนิยามเหมือนกับ output y แสดงดังนี้

$$a_0, b_0 = \lfloor x/16 \rfloor, x \bmod 16$$
$$a_1 = a_0 \oplus b_0$$
$$b_1 = a_0 \oplus ROR_4(b_0, 1) \oplus 8a_0 \bmod 16$$
$$a_2, b_2 = t_0[a_1], t_1[b_1]$$
$$a_3 = a_2 \oplus b_2$$
$$b_3 = a_2 \oplus ROR_4(b_2, 1) \oplus 8a_2 \bmod 16$$
$$a_4, b_4 = t_2[a_3], t_3[b_3]$$
$$y = 16b_4 + a_4$$

ที่ซึ่ง ROR_4 เป็นฟังก์ชันคล้ายกับ ROR คือจะมีการหมุน 4 บิต ขั้นแรกจะมีการกระจายค่าไบต์เป็น 2 ส่วนรวมทั้งมีการรมกันในกระบวนการ bijective mixing แต่ละส่วนจะถูกผ่านเข้าสู่กระบวนการ S-boxes ไปเรื่อยๆ ขั้นสุดท้ายจะมีการรวมกันของทั้งสองส่วนกลับมาเป็นค่าไบต์ ในการเปลี่ยนลำดับของ q_0 ใน S-boxes จะแสดงดังดังนี้

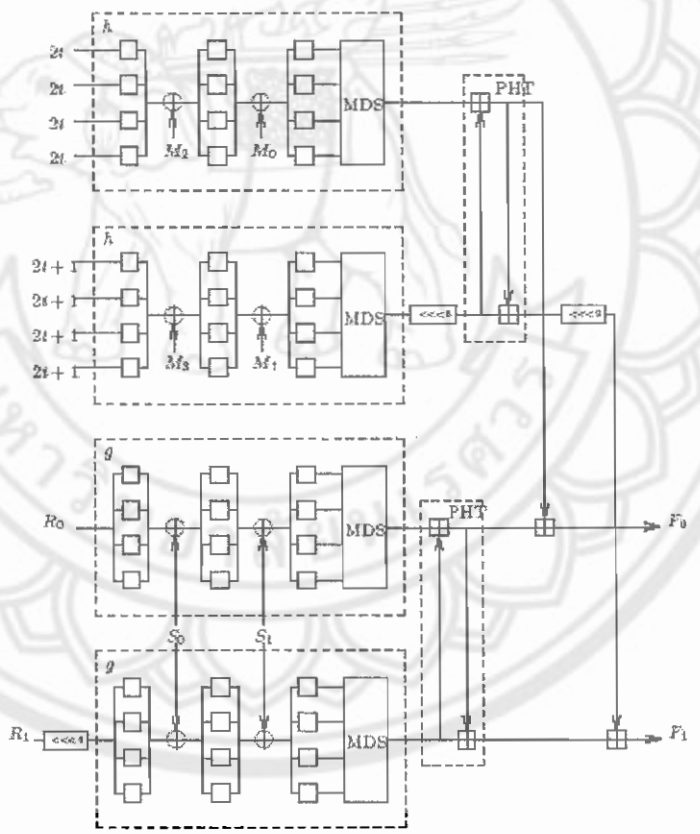
$$t_0 = [8 \quad 1 \quad 7 \quad D \quad 6 \quad F \quad 3 \quad 2 \quad 0 \quad B \quad 5 \quad 9 \quad E \quad C \quad A \quad 4]$$

$$t_0 = [E \quad C \quad B \quad 8 \quad 1 \quad 2 \quad 3 \quad 5 \quad F \quad 4 \quad A \quad 6 \quad 7 \quad 0 \quad 9 \quad D]$$
$$t_0 = [B \quad A \quad 5 \quad E \quad 6 \quad D \quad 9 \quad 0 \quad C \quad 8 \quad F \quad 3 \quad 2 \quad 4 \quad 7 \quad 1]$$
$$t_0 = [D \quad 7 \quad F \quad 4 \quad 1 \quad 2 \quad 6 \quad E \quad 9 \quad B \quad 3 \quad 0 \quad 8 \quad 5 \quad C \quad A]$$

ค่าที่แสดงจะเป็นเลขฐาน 16 สำหรับการ input i ข้ามา 0, 1, …, 15 และการเปลี่ยนลำดับของ q_1 แสดงดังนี้

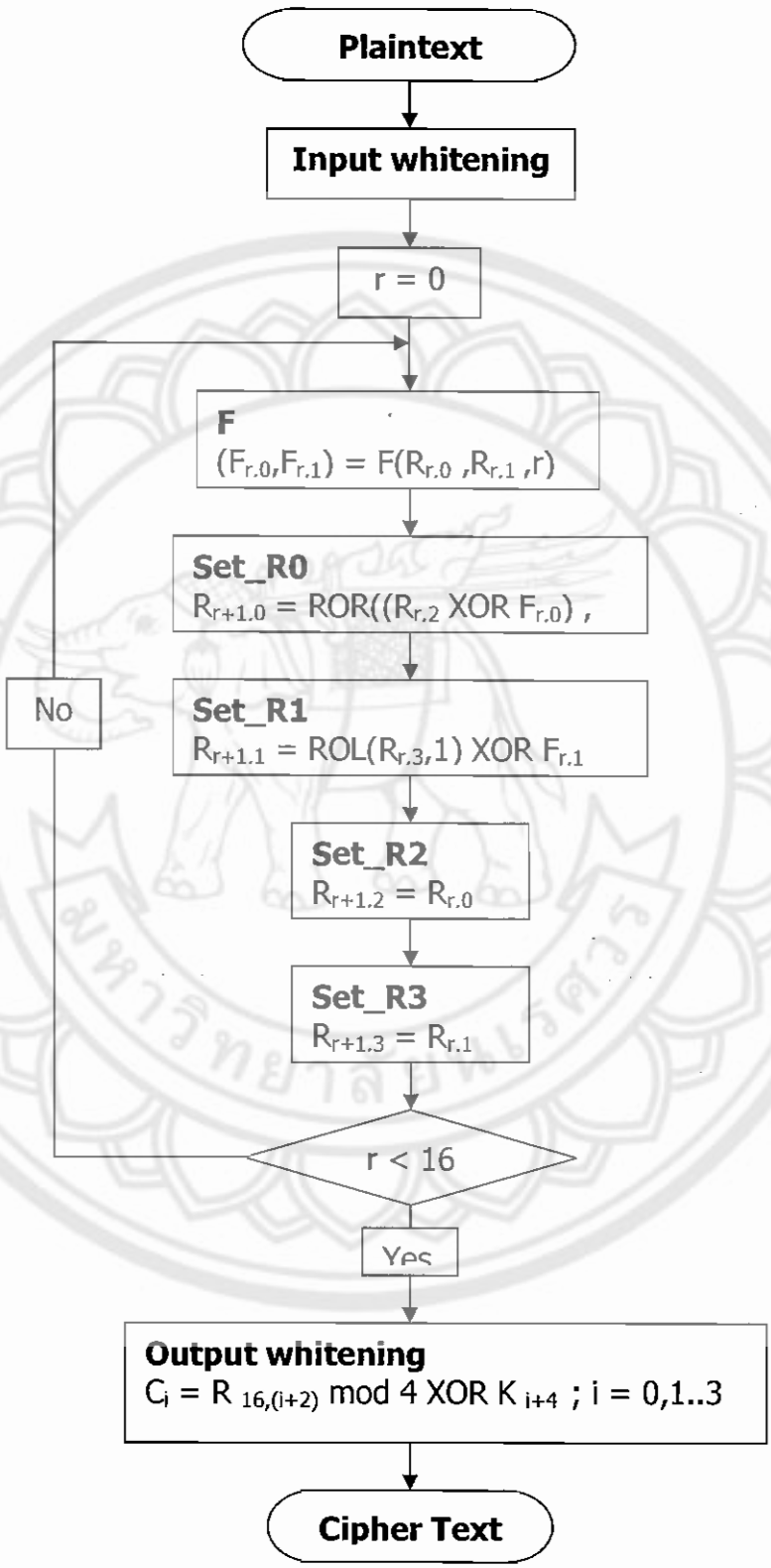
$$t_0 = [2 \quad 8 \quad B \quad D \quad F \quad 7 \quad 6 \quad E \quad 3 \quad 1 \quad 9 \quad 4 \quad 0 \quad A \quad C \quad 5]$$
$$t_0 = [1 \quad E \quad 2 \quad B \quad 4 \quad C \quad 3 \quad 7 \quad 6 \quad D \quad A \quad 5 \quad F \quad 9 \quad 0 \quad 8]$$
$$t_0 = [4 \quad C \quad 7 \quad 5 \quad 1 \quad 6 \quad 9 \quad A \quad 0 \quad E \quad D \quad 8 \quad 2 \quad B \quad 3 \quad F]$$
$$t_0 = [B \quad 9 \quad 5 \quad 1 \quad C \quad 3 \quad D \quad E \quad 6 \quad 4 \quad 7 \quad F \quad 2 \quad 0 \quad 8 \quad A]$$

Round Function เป็นการคำนวณในแต่ละรอบของฟังก์ชัน F แสดงดังรูปที่ 2.7 ซึ่งจะแสดง 1 รอบ หรือขนาด 128 บิต ดังนี้



รูปที่ 2.7 การคำนวณใน 1 รอบของฟังก์ชัน F (128 bit key) [3]

ในการเข้ารหัสในแต่ละฟังก์ชันนั้นจะมีสมการที่ใช้คำนวณตำแหน่งของข้อมูลและคีย์ต่างๆ ซึ่งแสดงได้ดังรูป 2.8



รูปที่ 2.8 แผนผังการเข้ารหัสแบบ Twofish

การเข้ารหัสในแต่ละครั้งในแบบที่ต่างกันก็จะมีวิธีการคำนวณ และการเข้ารหัสที่ต่างกัน แต่สิ่งที่เหมือนกันคือการทำให้ข้อมูลที่เข้ารหัสแล้วไม่สามารถเข้าใจได้จนกว่าจะมีการถอดรหัสกลับมาอีกครั้ง ซึ่งเป็นหลักการสำคัญในการเข้ารหัสและถอดรหัสข้อมูล

2.6 ตารางสรุปคุณสมบัติของแต่ละอัลกอริทึม

ในแต่ละอัลกอริทึมมีคุณสมบัติ และการทำงานที่แตกต่างกัน ซึ่งสามารถเปรียบเทียบได้ดังตารางต่อไปนี้

ตารางที่ 2.1 เปรียบเทียบคุณสมบัติของอัลกอริทึมในการเข้ารหัส

อัลกอริทึม	ประเภท/รูปแบบ	ขนาดข้อมูล (บิต)	ขนาดคีย์ (บิต)	Operation	จำนวนรอบ
AES	สมมาตร/บล็อกข้อมูล	128, 192, 256	128, 192, 256	XOR, addition, Multiplication	10, 12, 14
Serpent	สมมาตร/บล็อกข้อมูล	128	128, 192, 256	S-boxes, XOR, Shipping	32
RC2	สมมาตร/บล็อกข้อมูล	64	8 – 128 (64)	Addition, subtraction, XOR, rotation	18
CAST-256	สมมาตร/บล็อกข้อมูล	128	128, 192, 256	Addition, subtraction, XOR, rotation, fixed S-boxes	48
Blowfish	สมมาตร/บล็อกข้อมูล	64	32 – 448 (128)	XOR, variable S-boxes, addition	16
Twofish	สมมาตร/บล็อกข้อมูล	128	128, 192, 256	Addition, XOR, rotation, fixed S-boxes	16

จากตารางจะแสดงถึงคุณสมบัติในการทำงานของอัลกอริทึมทั้ง 6 อัลกอริทึม ซึ่งในส่วนของประสิทธิภาพในการเข้ารหัสนั้นจะแสดงการทดสอบรวมทั้งผลสรุปในบทต่อๆ ไป