ลายมือชื่อดิจิตอล

Digital Signature

นายวิษณุ ขอบเขต
นายสุทธิพงษ์ ตันธีระธรรม

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2545

ลายมือชื่อดิจิตอล

Digital Signature

โดย

นายวิษณุ ขอบเขต

นายสุทธิพงษ์ ตันธีระธรรม

อาจารย์ที่ปรึกษา

อ.ธนา หงษ์สุวรรณ

อ.อัครเดช วัชระภูพงษ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2545

ปริญญานิพนธ์ปีการศึกษา 2545

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ลายมือชื่อดิจิตอล

Digital Signature

ผู้จัดทำ

1. นาย วิษณุ ขอบเขต รหัสประจำตัว 42010334
2. นาย สุทธิพงษ์ ตันธีระธรรม รหัสประจำตัว 42010388

_____ อาจารย์ที่ปรึกษา

(อ.ธนา หงษ์สุวรรณ)

_____ อาจารย์ที่ปรึกษา

(อ.อัครเดช วัชระภูพงษ์)

ลายมือชื่อดิจิตอล

นายวิษณุ ขอบเขต 42010334
นายสุทธิพงษ์ ตันธีระธรรม 42010388
อาจารย์ ธนา หงษ์สุวรรณ อาจารย์ที่ปรึกษา
อาจารย์ อัครเดช วัชระภูพงษ์ อาจารย์ที่ปรึกษา
ปีการศึกษา 2545

บทคัดย่อ

ลายมือชื่อดิจิตอลใช้พื้นฐานของการเข้ารหัสแบบคีย์ต่าง ซึ่งผู้ใช้ทุกคนมีคีย์ส่วนตัว และคีย์สาธารณะที่กำหนดขึ้นโดยองค์กรพิสูจน์สิทธิ์ เมื่อผู้ใช้บุคคลนั้นได้มีการลงนามลายมือชื่อดิจิตอล ลำดับของตัวเลขชุดหนึ่งจะสร้างมาจากคีย์ส่วนตัวของบุคคลนั้น ประกอบกับข้อมูลที่ต้องการลงนาม ซึ่งลำดับของตัวเลขนี้เรียกว่าลายมือชื่อดิจิตอล ลายมือชื่อดิจิตอลสามารถบ่งบอกได้ว่าผู้ลงนามเป็นผู้ใดโดยดูจากเอกสารสิทธิ์

โครงงานนี้ได้นำหลักการของลายมือชื่อดิจิตอลมาใช้กับเอกสารที่จัดทำขึ้นจากโปรแกรมไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซลให้โปรแกรมดังกล่าวมีความสามารถในการลงลายมือชื่อดิจิตอลและตรวจสอบลายมือชื่อได้ในรูปแบบของโอแอลอี (OLE – Object Linking and Embedding) ซึ่งเป็นการพัฒนาต่อจากโครงงานรุ่นก่อนหน้า โดยพัฒนาให้มีการใช้เอกสารสิทธิ์ที่รับรองโดยองค์กรพิสูจน์สิทธิ์ในการสร้างลายมือชื่อดิจิตอลเพื่อสามารถยืนยันตัวบุคคลได้ว่าใครเป็นเจ้าของลายมือชื่อนั้น โปรแกรมนี้เขียนด้วยภาษาซีพลัสพลัส (C++) และใช้ไมโครซอฟท์วิชวลซีพลัสพลัส (Microsoft Visual C++) เป็นเครื่องมือในการพัฒนา สามารถทำงานร่วมกับโปรแกรมไมโครซอฟท์เวิร์ด 97 และไมโครซอฟท์เอ็กเซล 97 ได้

Digital Signature

Widsanu Khobkhet

Suttipong Tandheeradharm

Thana Hongsuwan Advisor

Akkradach Watcharapupong Advisor

Abstract

A digital signature is based on asymmetric cryptography where every user has a unique pair of private and public keys, duly certified by a trusted Certificate Authority. When he or she signs a transaction, a unique mathematical code is created with the help of his of her private key and the actual content of the transaction. This "signature" , which is bound to the transaction, can identify the signer's identify by is relationship to the digital certificate.

This project brings the advantage of digital signature to use with Microsoft Word and Microsoft Excel. It brings more functionality to Microsoft Word to and Microsoft Excel sign and verify digital signature. Program that created in this project is using Microsoft C++. It can cooperate with Microsoft Word 97 and Microsoft Excel 97.

กิตติกรรมประกาศ

ในการทำปริญญานิพนธ์ฉบับนี้คณะผู้จัดทำขอกราบขอบพระคุณบิดามารดาที่ช่วยอบรมสั่งสอน เลี้ยงดูคณะผู้จัดทำ ส่งเสริมด้านการศึกษาหาความรู้ต่าง ๆ รวมถึงกำลังใจอันที่หาที่เปรียบมิได้จนกระทั่งช่วยให้คณะผู้จัดจบการศึกษาสำเร็จด้วยดี

ขอขอบพระคุณอาจารย์ธนา หงษ์สุวรรณและอาจารย์อัครเดช วัชระภูพงษ์ที่คอยให้คำปรึกษาและคำแนะนำต่าง ๆ เกี่ยวกับโครงงานนี้เป็นอย่างมาก ทำให้โครงงานครั้งนี้สำเร็จลุล่วงไปได้ด้วยดี ตลอดจนคณาจารย์ทุกท่านผู้ประสิทธิประสาทวิชาให้แก่คณะผู้จัดทำทุกท่าน

ขอบคุณพี่ เพื่อน และน้อง ทุกคนที่เอื้อเฟื้อน้ำใจในการแก้ไขปัญหาที่เกิดขึ้น และความคิดดี ๆ ที่ประกอบขึ้นมาเป็นโครงงานชิ้นนี้สำเร็จ

นายวิษณุ ขอบเขต
นายสุทธิพงษ์ ตันธีระธรรม

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญรูปภาพ

สารบัญรูปภาพ (ต่อ)

หน้า

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

ในปัจจุบันการส่งข้อมูลต่าง ๆ ที่สำคัญ ๆ ผ่านทางเครือข่ายอินเทอร์เน็ตหรือการส่งข้อมูลในรูปแบบอิเล็กทรอนิกส์ต่าง ๆ นั้น ประการแรกที่ผู้รับกระทำคือการตรวจสอบที่มาว่าใครเป็นผู้ส่งข้อมูลนั้นมา รวมถึงความถูกต้องของข้อมูลว่าข้อมูลดังกล่าวมีการดัดแปลงแก้ไขหรือไม่อย่างไร เพื่อเป็นการรับรองข้อมูลว่าเป็นข้อมูลที่ผู้ส่งส่งมาจริง ๆ และข้อมูลไม่มีการเปลี่ยนแปลง ลายมือชื่อดิจิตอล (Digital Signature) จึงถือกำเนิดขึ้น ลายมือชื่อดิจิตอลแตกต่างจากลายมือที่เขียนขึ้นในเอกสารทั่วไป เพราะลายมือชื่อดิจิตอลจะขึ้นอยู่กับตัวข้อมูลเป็นสำคัญและใช้หลักการของการเข้ารหัสแบบคีย์ต่าง กล่าวคือลายมือชื่อดิจิตอลสร้างขึ้นจากข้อมูลทั้งหมดโดยการเข้ารหัสทางเดียว ทำให้ลายมือชื่อดิจิตอลไม่คงที่ เปลี่ยนแปลงตามเนื้อหาของข้อมูล ดังนั้นหากมีการเปลี่ยนแปลงแก้ไขข้อมูลเพียงบิตเดียวก็จะทำให้ลายมือชื่อดิจิตอลนั้นเปลี่ยนแปลงไปด้วย จากหลักการนี้สามารถนำมาใช้ในการพิสูจน์ที่มาและความถูกต้องของข้อมูลทางอิเล็กทรอนิกส์ได้

หลักการทำงานของลายมือชื่อดิจิตอลเริ่มจากการนำข้อมูลที่ต้องการส่งมาทำการสังเคราะห์ข้อมูลให้มีขนาดเล็กลง โดยผ่านฟังก์ชันทางเดียวเรียกว่า แฮชฟังก์ชัน (Hash Function) เมื่อได้ข้อมูลที่ผ่านการสังเคราะห์ออกมาเรียกว่าเมสเซจไดเจสต์ (Message Digest) จะทำการเข้ารหัสโดยใช้คีย์ส่วนตัวของผู้ส่งข้อมูลที่ผ่านการเข้ารหัส สิ่งที่ได้จากขั้นตอนนี้เรียกว่าลายมือชื่อดิจิตอล โดยลายมือชื่อดิจิตอลของข้อมูลแต่ละชุดนั้นไม่คงที่ และจะถูกส่งไปพร้อมกับข้อมูลต้นฉบับ เพื่อเป็นการยืนยันความถูกต้องและพิสูจน์ตัวบุคคล ผู้รับเมื่อรับข้อมูลต้นฉบับและลายมือชื่อดิจิตอล จะใช้ฟังก์ชันทางเดียวแบบเดียวกันกับตอนแรกเพื่อสังเคราะห์ข้อมูลอีกชุดหนึ่งขึ้น จากนั้นจะใช้คีย์สาธารณะของผู้ส่งเพื่อถอดรหัสลายมือชื่อดิจิตอลที่ส่งมาได้ข้อมูลสังเคราะห์อีกหนึ่งชุด จากนั้นเปรียบเทียบกับข้อมูลสังเคราะห์ที่ได้จากข้อมูลต้นฉบับว่าตรงกันหรือไม่ ถ้าตรงกันแสดงว่าข้อมูลไม่มีการเปลี่ยนแปลงแก้ไขและทำให้ทราบว่าบุคคลที่เป็นเจ้าของคีย์สาธารณะเป็นผู้ส่งจริง ๆ แต่ถ้าไม่ตรงกันก็แสดงว่าข้อมูลที่รับมามีการเปลี่ยนแก้ไข ในกรณีที่ไม่สามารถทำการถอดรหัสลายมือชื่อดิจิตอลได้ก็แสดงว่าเอกสารไม่ใช่ของผู้ส่งจริง ๆ ทำให้สามารถตรวจสอบบุคคลอื่นที่ปลอมเอกสารนี้ได้

ในการที่จะสามารถยืนยันได้ว่าลายมือชื่อดิจิตอลนั้นเป็นของบุคคลใด จะมีหน่วยงานที่ทำหน้าที่ในการรับรองการเป็นบุคคลคนนั้นและสร้างคู่คีย์ต่างให้กับบุคคลนั้นนั่นก็คือองค์กรพิสูจน์สิทธิ์ (Certificate Authorities - CA) โดยคู่คีย์ที่ได้รับรองจากองค์กรพิสูจน์สิทธิ์นี้จะอยู่ในรูปของเอกสารสิทธิ์ (Digital Certificate) ซึ่งประกอบด้วยข้อมูลอื่น ๆ อีก นอกเหนือจากคีย์ดังกล่าว

1.2 วัตถุประสงค์ของปริญญานิพนธ์

เพื่อศึกษาถึงการสร้างลายมือชื่อดิจิตอล และสามารถนำความรู้ที่ศึกษามาได้ นำมาพัฒนาโปรแกรมที่สามารถลงนามลายมือชื่อดิจิตอลได้ เพื่อความถูกต้องของข้อมูลคือ ข้อมูลจะต้องไม่มีการเปลี่ยนแปลงระหว่างการส่ง และทำให้สามารถยืนยันได้ว่าบุคคลที่ส่งข้อมูลนั้นเป็นบุคคลคนนั้นจริง ๆ โดยศึกษาการใช้เอกสารสิทธิ์ที่รับรองโดยองค์กรพิสูจน์สิทธิ์

1.3 ขอบเขตของปริญญานิพนธ์

ในงานวิจัยนี้จะทำการสร้างโปรแกรมลงลายมือชื่อดิจิตอลและตรวจสอบลายมือชื่อดิจิตอลในรูปของโปรแกรมสนับสนุนการทำงานแบบโอแอลอี ซึ่งการทำงานร่วมกับไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซล ลายมือชื่อที่สร้างขึ้นจะสร้างจากข้อมูลที่อยู่ในโปรแกรมไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซล โดยพิจารณาที่เนื้อความของข้อมูลเป็นสำคัญ โดยแบ่งการดำเนินงานออกเป็น 4 ส่วนใหญ่ ๆ คือ

1. ส่วนโปรแกรมเข้ารหัส/ถอดรหัส การทำแฮชชิ่งข้อมูลและสร้างลายมือชื่อดิจิตอลโดยใช้เอพีไอของไมโครซอฟท์วินโดว์สที่ชื่อว่าคริปโตเอพีไอ (CryptoAPI)
2. ส่วนที่ติดต่อกับที่เก็บเอกสารสิทธิ์ โดยมีการนำคีย์ที่ได้รับการรับรองจากองค์กรพิสูจน์สิทธิ์มาใช้ในสร้างและตรวจสอบลายมือชื่อดิจิตอล
3. ส่วนโปรแกรมที่เป็นรูปแบบโอแอลอี ติดต่อกับไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซลเพื่อนำข้อมูลมาทำการสร้างและตรวจสอบลายมือชื่อดิจิตอล
4. สร้างส่วนติดต่อกับผู้ใช้ให้ง่ายต่อการใช้งานโดยใช้ไมโครซอฟท์ฟาวเดชั่นคลาส (Microsoft Foundation Class – MFC)

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1. ความเข้าใจในเรื่องการเข้ารหัสทั้งแบบคีย์เหมือนและคีย์ต่าง
2. สามารถเขียนโปรแกรมสร้างและตรวจสอบลายมือชื่อดิจิตอลได้
3. สามารถเขียนโปรแกรมที่เป็นรูปแบบโอแอลอีได้

1.5 ขั้นตอนการดำเนินงาน

โครงงานนี้จะใช้ภาษาซีพลัสพลัส (C++) เป็นหลักในการพัฒนาโปรแกรมและใช้ไมโครซอฟท์วิชวลซีพลัสพลัส (Microsoft Visual C++) เป็นเครื่องมือที่ใช้ในการพัฒนาโปรแกรมและศึกษาเรื่องการเข้ารหัสถอดรหัสทั้งแบบคีย์เหมือนและแบบคีย์ต่าง หัวข้อของเรื่องที่ศึกษามีดังนี้

1. ศึกษาการทำแฮชข้อมูลโดยใช้แฮชอัลกอริทึมแบบต่าง ๆ
2. ศึกษาวิธีการสร้างและตรวจสอบลายมือชื่อดิจิตอล

3. ศึกษาวิธีการเขียนโปรแกรมแบบโอแอลอีเซิร์ฟเวอร์และโอแอลอีออโตเมชันไคลเอ็นต์
4. ศึกษาการสร้างแถบเครื่องมือในไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซล

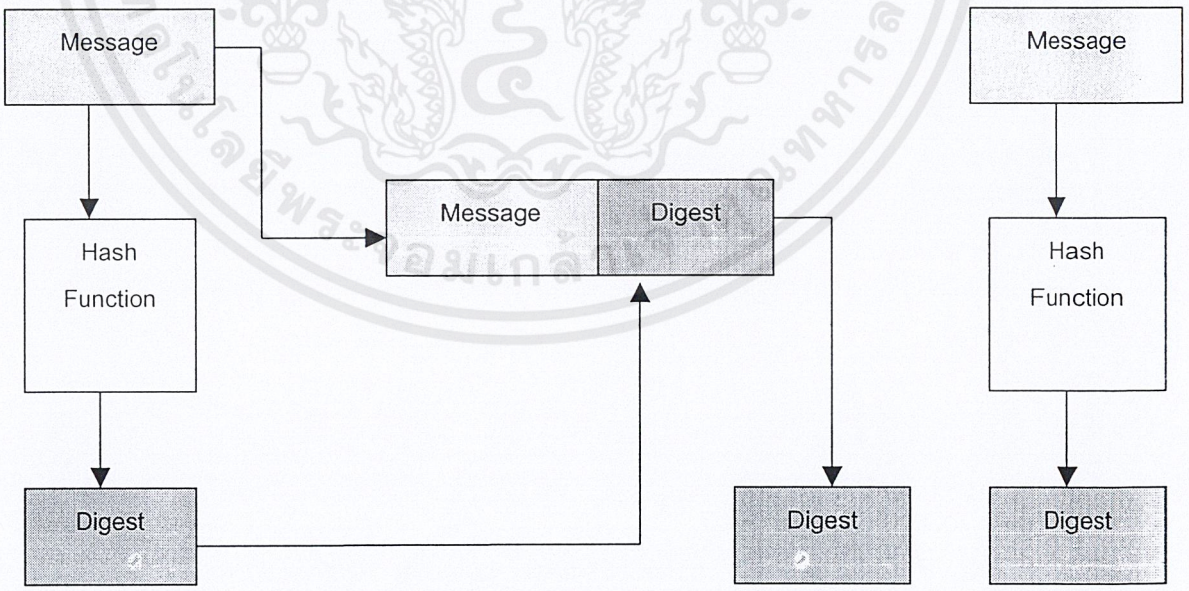
บทที่ 2
ลายมือชื่อดิจิตอล

2.1 การทำงานของลายมือชื่อดิจิตอล

ลายมือชื่อดิจิตอลใช้เมื่อต้องการความมั่นใจในแหล่งที่มา ของเอกสารเปรียบเหมือนลายมือชื่อซึ่งเฉพาะเจ้าของจริงที่สามารถคำนวณขึ้นมาได้ แต่ลายมือชื่อนี้สามารถพิสูจน์ได้กล่าวคือบุคคลอื่นสามารถตรวจสอบได้ว่าลายมือชื่อนั้นมาจากผู้สร้างจริง ๆ วิธีธรรมดาทั่วไปที่จะคำนวณลายมือชื่อดิจิตอลก็คือการเข้ารหัสแบบคีย์ต่างหรือคีย์สาธารณะ เช่น ผู้ลงนามคำนวณค่าลายมือชื่อโดยใช้คีย์ส่วนตัว (Private key) และคนอื่นสามารถใช้คีย์สาธารณะ (Public Key) พิสูจน์ได้ว่าลายมือชื่อมาจากคีย์ส่วนตัวที่ตรงกัน คุณสมบัติที่สำคัญของลายมือชื่อดิจิตอลที่สำคัญนั้นจะต้องประกอบด้วย 2 ประการคือ

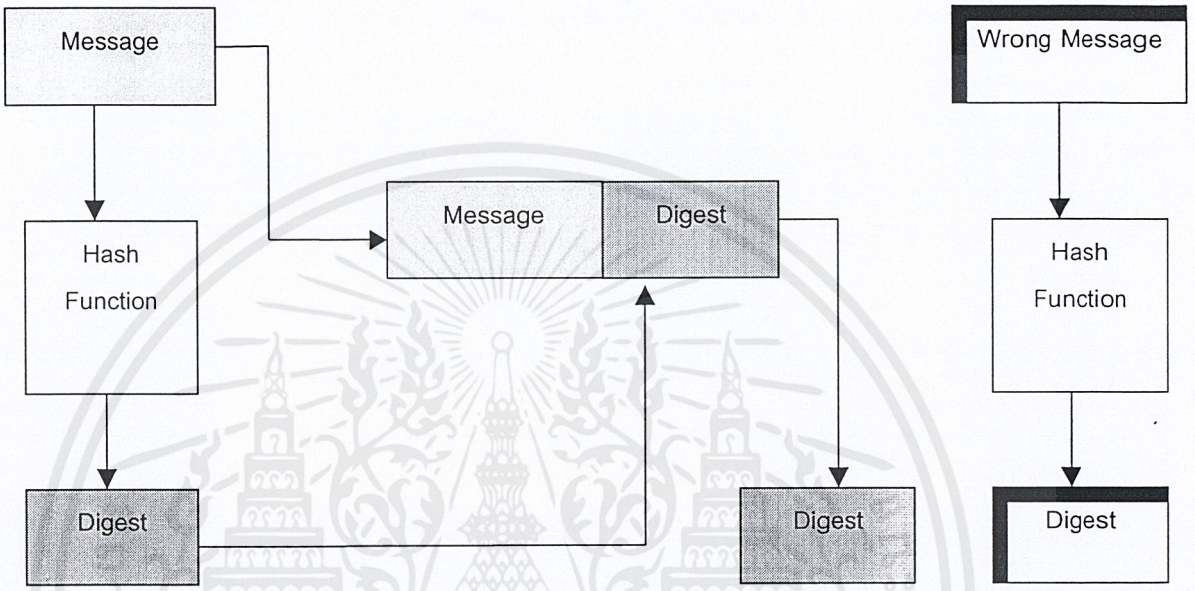
1. สามารถยืนยันได้ว่าข้อมูลที่ได้รับมานั้นไม่มีการเปลี่ยนแปลงระหว่างการส่ง
2. สามารถยืนยันได้ว่าข้อมูลนั้นได้รับการยืนยันจากผู้ลงลายมือชื่อจริง ๆ

วิธีการยืนยันว่าข้อมูลที่ได้รับมานั้นมีความถูกต้อง และไม่ได้รับการเปลี่ยนแปลงข้อมูลระหว่างการส่งนั้นสามารถทำได้โดยวิธีง่าย ๆ คือ การหาค่าแฮช (Hash Value) ดังรูปที่ 2.1



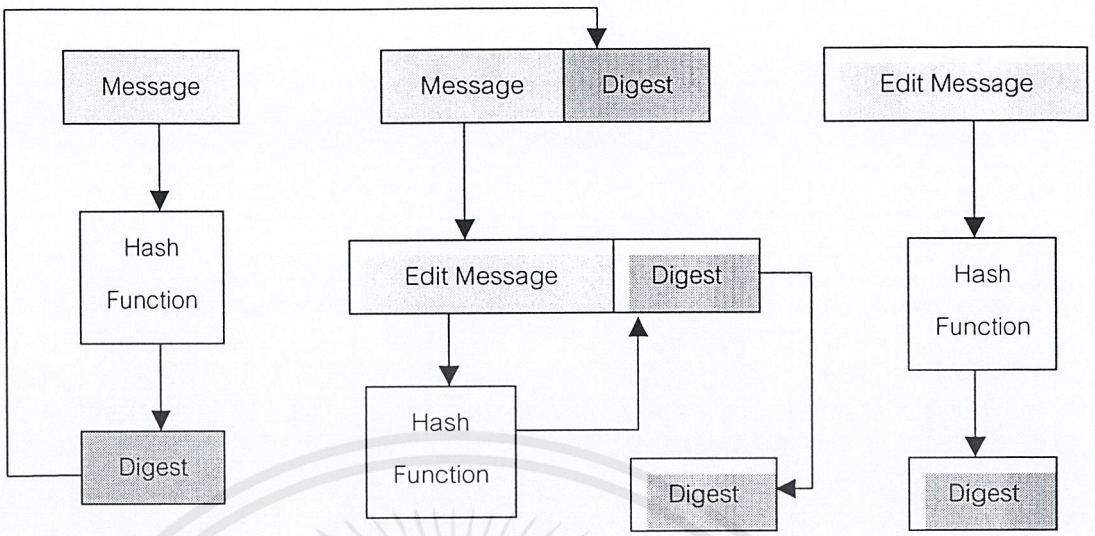
รูปที่ 2.1 วิธีการตรวจสอบความถูกต้องของข้อมูลแบบพื้นฐาน

จากรูปที่ 2.1 จะเป็นวิธีการที่ใช้ในการยืนยันความถูกต้องของข้อมูลอย่างง่าย โดยการหาค่าแฮช หรือจะเรียกว่าเมสเซจไดเจสต์ (Message Digest หรือ Digest) ก็ได้ซึ่งเมสเซจไดเจสต์ จะเปรียบเสมือนตัวแทนของข้อมูล ถ้าข้อมูลที่ได้รับเข้าสู่แฮชฟังก์ชันนั้นต่างกันค่าของเมสเซจไดเจสต์ที่ได้รับออกมานั้นก็จะต่างกัน แต่หากว่าข้อมูลที่ได้รับเข้าสู่แฮชฟังก์ชันนั้นเหมือนกันก็จะทำให้ค่าของเมสเซจไดเจสต์ที่ได้ออกมาเหมือนกัน มีโอกาสน้อยมากที่จะทำให้ข้อมูลเปลี่ยนไปแล้วยังได้เมสเซจไดเจสต์เหมือนเดิม



รูปที่ 2.2 เมื่อข้อมูลเกิดการผิดพลาด

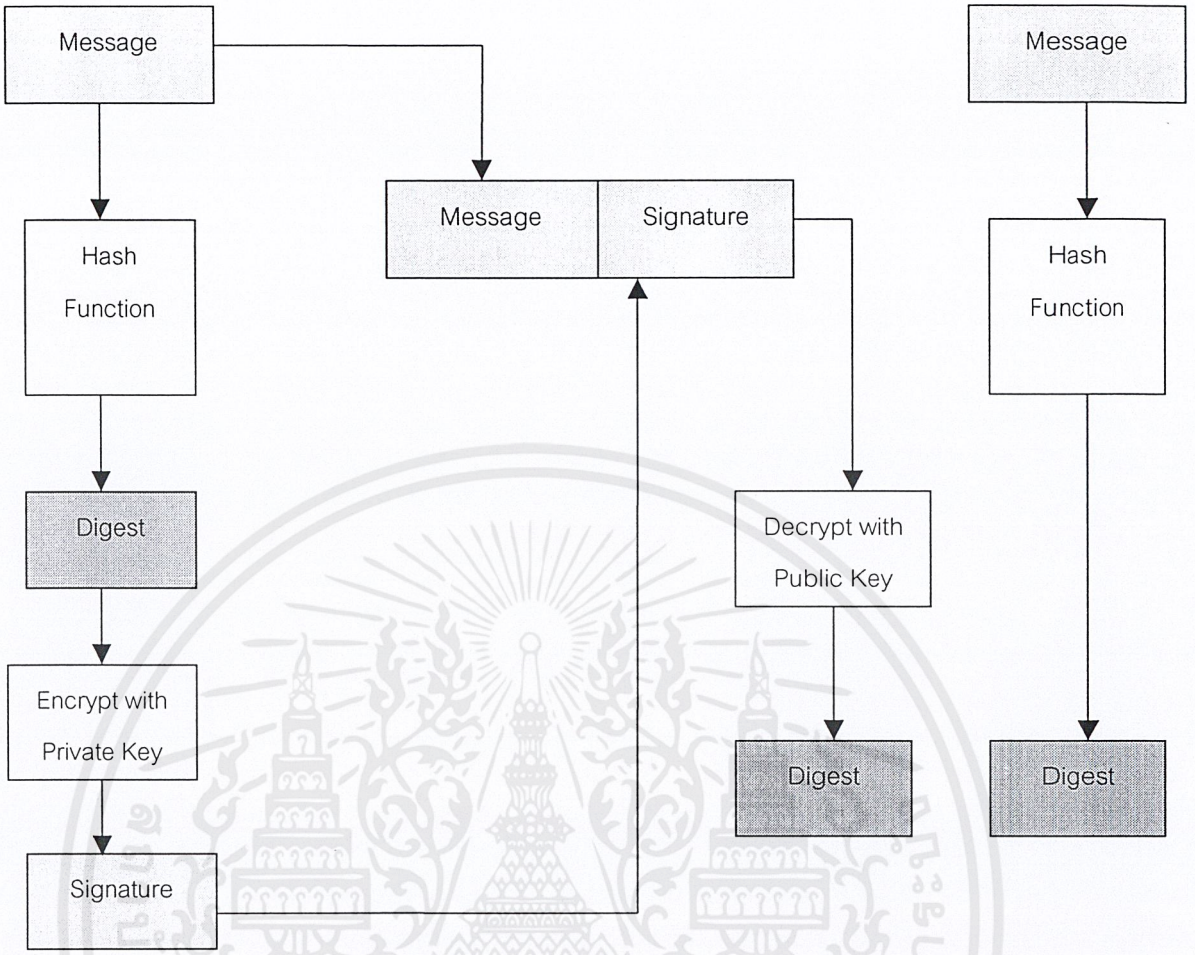
จากรูปที่ 2.2 แสดงให้เห็นทางด้านซ้ายมือเป็นของผู้ส่งซึ่งจะมีทั้งข้อมูลและเมสเซจไดเจสต์ส่งมาให้ ทางด้านผู้รับเกิดได้รับข้อมูลที่มีความผิดพลาดเกิดขึ้น (Wrong) เมื่อด้านผู้รับทำการแฮชซึ่งได้เมสเซจไดเจสต์ออกมา เมื่อนำไปเปรียบเทียบกับเมสเซจไดเจสต์ที่ได้รับ จะเห็นว่าไม่เหมือนกัน ทำให้ฝั่งผู้รับทราบได้ทันทีว่าข้อมูลที่ได้รับมานั้นมีความผิดพลาด แต่การทำวิธีนี้อาจไม่ปลอดภัยนักเนื่องจากหากผู้ต้องการปลอมแปลงทราบว่าผู้ส่งใช้แฮชฟังก์ชันอะไร จะทำให้สามารถทำการเปลี่ยนแปลงข้อมูลนั้นได้ดูได้จากรูปที่ 2.3



รูปที่ 2.3 เมื่อมีผู้อ้างตัวเป็นผู้ส่ง

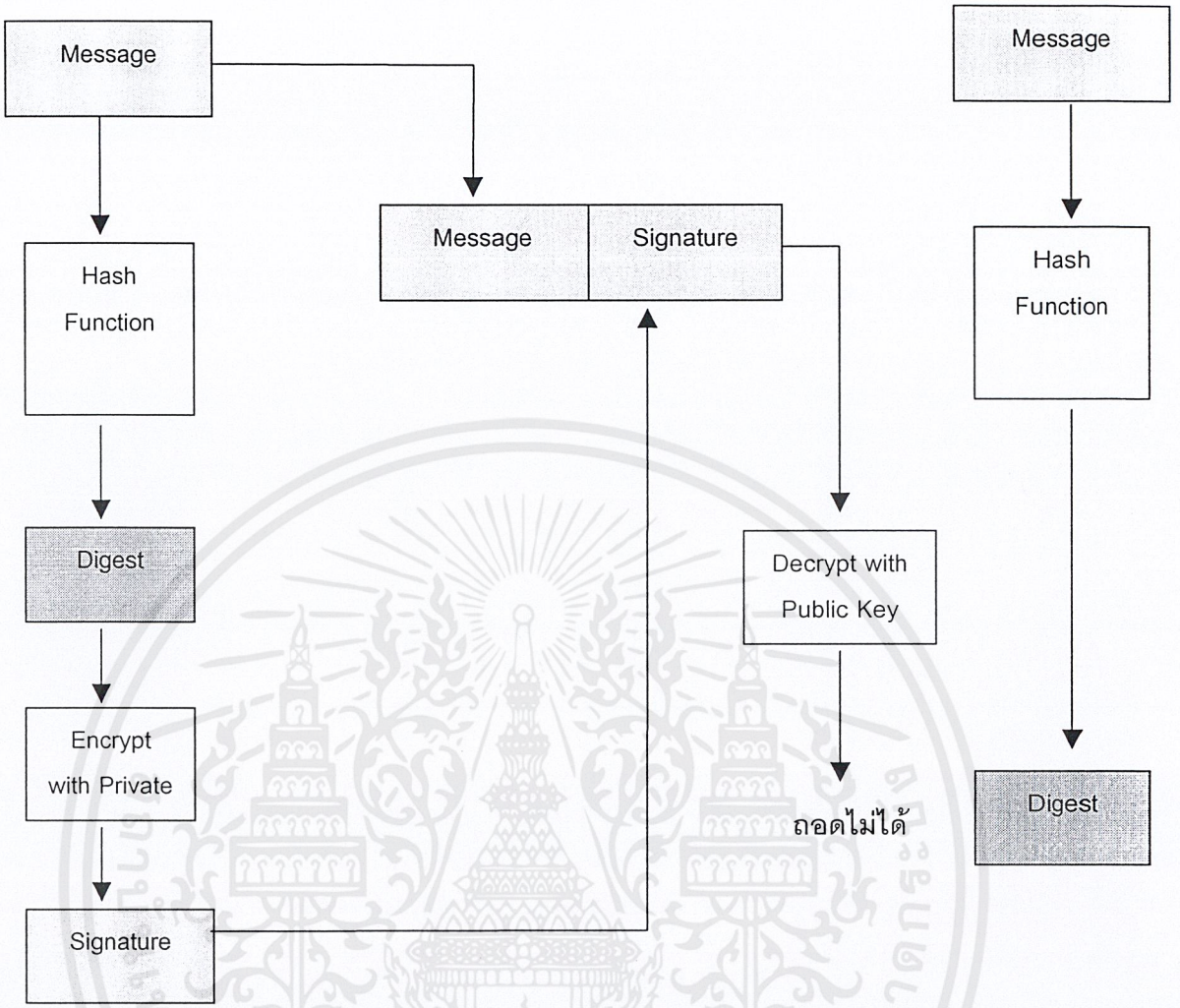
จากรูปที่ 2.3 เมื่อมีผู้อ้างตัวเป็นผู้ส่งนำข้อมูลที่ผู้ส่งส่งมาดัดแปลงแก้ไข แล้วใช้แฮชฟังก์ชันซึ่งเป็นอันเดียวกับผู้ส่งและผู้รับใช้จะทำให้ด้านผู้รับไม่อาจรู้ได้ว่าข้อมูลนั้นมีการเปลี่ยนแปลง ทำให้ไม่ปลอดภัยดังนั้นจึงได้มีการใช้การเข้ารหัสเข้ามาช่วย

ในการยืนยันบุคคลผู้ส่งนั้นเป็นสิ่งที่จะต้องทำตามมาหลังจากการยืนยันความถูกต้องของข้อมูล เทคนิคที่นิยมใช้ในการทำก็คือ ใช้เทคนิคคีย์ต่าง เช่น อาร์เอสเอ (RSA) โดยข้อความแรกเริ่มจะผ่านแฮชฟังก์ชันแล้วได้เป็นเมสเซจไดเจสต์ เพื่อเป็นการยืนยันความถูกต้องของข้อมูลก่อน จากนั้นจะนำเมสเซจไดเจสต์ ที่ได้ส่งไปเข้ารหัสกับคีย์ส่วนตัวของผู้ส่งเพื่อเป็นการยืนยันว่าเป็นเจ้าของเอกสารนั้นจริง และจะได้สิ่งที่เรียกว่าลายมือชื่อดิจิตอลออกมาส่งไปพร้อมกับข้อมูลด้วย ทางด้านรับนั้นจะเป็นวิธีการที่คล้ายกับด้านส่งโดยจะต้องเริ่มจากการหาเมสเซจไดเจสต์ของข้อมูลที่ได้รับมาเก็บไว้ก่อน จากนั้นจะนำลายมือชื่อดิจิตอลที่ได้รับมาด้วย มาทำการถอดรหัสด้วยคีย์สาธารณะของผู้ส่ง ในขั้นตอนนี้เองที่จะสามารถยืนยันบุคคลผู้ส่งได้ เพราะถ้าหากผู้ที่ลงลายมือชื่อไม่ได้เป็นบุคคลที่เราคาดว่าเป็นเจ้าของ ก็จะไม่สามารถถอดลายมือชื่อดิจิตอลได้ ถ้าหากการยืนยันบุคคลผู้ส่งสำเร็จแล้วจะได้รับเมสเซจไดเจสต์ของข้อมูลก่อนจะส่งออกมา ด้านรับจะต้องทำการตรวจสอบความถูกต้องของข้อมูลด้วยโดยการนำเมสเซจไดเจสต์ที่คำนวณได้ในตอนรับข้อมูล กับเมสเซจไดเจสต์ที่ได้หลังจากการถอดรหัสลายมือชื่อดิจิตอลออกมา นำมาเปรียบเทียบกันถ้าหากเมสเซจไดเจสต์ที่ได้ออกมามีค่าเท่ากันนั่นแสดงว่าข้อมูลที่ได้รับนั้นเป็นข้อมูลเดียวกันจากทางด้านส่ง และไม่มีการเปลี่ยนแปลงข้อมูลระหว่างการส่ง ขั้นตอนการส่งและการรับของวิธีการนี้ แสดงไว้ในรูปที่ 2.4



รูปที่ 2.4 วิธีการสร้างลายมือชื่อดิจิตอลโดยใช้แฮชฟังก์ชันและคีย์ต่าง

จากรูปที่ 2.4 จะเห็นว่ามีความปลอดภัยมากยิ่งขึ้นในกรณีที่มีการปลอมแปลงจะเป็นดังรูปที่ 2.5



รูปที่ 2.5 ในกรณีมีผู้แอบอ้าง

จากรูปที่ 2.5 เมื่อมีผู้แอบอ้าง ผู้แอบอ้างไม่รู้คีย์ส่วนตัวของผู้ส่งจริง ๆ จึงใช้คีย์ของตัวเอง ซึ่งไม่ได้เข้าคู่กับคีย์สาธารณะทางฝั่งผู้รับ ทำให้ทางฝั่งผู้รับไม่สามารถถอดรหัสลายมือชื่อดิจิตอลได้ จึงรู้ว่าไม่ใช่ผู้ส่งตัวจริงเป็นคนส่ง

บทที่ 3
การเข้ารหัสและการถอดรหัส

3.1 แฮชฟังก์ชัน (Hash function)

การแฮชเป็นวิธีการที่ใช้ในการตัวสอบการเปลี่ยนแปลงของข้อมูล และใช้ในวงการอุตสาหกรรมคอมพิวเตอร์มาตั้งแต่ปี ค.ศ. 1970 หลักการทำงานจะคล้าย ๆ กับการเช็คซัม (Checksum) ของการส่งข้อมูลผ่านระบบเครือข่ายคือใช้ตรวจสอบว่าข้อมูลที่ส่งมานั้นมีความถูกต้องหรือไม่ ถ้าการส่งข้อมูลถูกรบกวนด้วยสัญญาณรบกวนจะทำให้ค่าของบิตข้อมูลมีการเปลี่ยนแปลงก็จะสามารถตรวจสอบเปรียบเทียบกับค่าของเช็คซัมที่ส่งมาพร้อมกับข้อมูลนั้น ค่าเช็คซัมที่ได้จากข้อมูลที่รับมากับค่าเช็คซัมที่ติดมากับข้อมูลจะไม่เท่ากันทำให้ต้องมีการส่งข้อมูลใหม่ ภายในแฮชฟังก์ชันจะมีแฮชอัลกอลิทึมรูปแบบต่าง ๆ ที่ใช้ในการลดขนาดของข้อมูลให้มีขนาดเล็กลงและสิ่งที่ได้จากแฮชฟังก์ชันเรียกว่า เมสเซจไดเจสต์ (Message Digest) แฮชฟังก์ชันนั้นมีชื่อเรียกหลายชื่อด้วยกัน เช่น เมสเซจไดเจสต์ (Message digest), เช็คซัม (Checksum), คอนแทร็กชันฟังก์ชัน (Contraction function), ดาต้าอินทิกริตีเช็ค (Data integrity check) เป็นต้น ลักษณะที่สำคัญโดยทั่วไปของแฮชฟังก์ชันมีดังนี้

1. การทำงานของแฮชฟังก์ชันเป็นฟังก์ชันทางเดียว เมื่อเรามีข้อมูล (M), แฮชฟังก์ชัน ($H(\ )$) และเมสเซจไดเจสต์ (d) จะเขียนเป็นสมการได้ดังนี้

$$d = H(M)$$

2. ผลลัพธ์หรือเมสเซจไดเจสต์ที่ได้จากการแฮชจะมีขนาดคงที่คือ 128 บิต หรือ 160 บิต
3. การแฮชสามารถกระทำการบนอุปกรณ์ฮาร์ดแวร์หรือซอฟต์แวร์ก็ได้
4. เมื่อมีเมสเซจไดเจสต์ (d) และแฮชฟังก์ชัน ($H(\ )$) เป็นการยากที่จะหาข้อมูลต้นฉบับ (M)
5. เมื่อมีข้อมูลชุดแรก (M) และข้อมูลชุดที่สอง (N) ซึ่งไม่เหมือนกับชุดแรกเป็นการยากที่จะทำให้

$$H(M) = H(N)$$

ต่อไปนี้คณะผู้จัดทำขอยกตัวอย่างแฮชอัลกอริทึมโดยย่อที่นำมาใช้ในโครงงานดังนี้

3.1.1 เมสเซจไดเจสต์ 4 (Message Digest 4 – MD4)

MD4 ถูกพัฒนาขึ้นโดยไรเวส (Rivest) ในปี ค.ศ. 1990 เป็นแฮชอัลกอลิทึมที่ความปลอดภัยน้อยในปัจจุบันนี้ไม่ค่อยนำมาใช้ในการสังเคราะห์มากนัก ข้อมูลที่จะทำการแฮชซึ่งโดยใช้อัลกอริทึมแบบ MD4 นี้จะต้องมีการเพิ่มจำนวนบิตให้เป็นจำนวนเท่าของ 512 บิต ในแต่ละบล็อก การสังเคราะห์ข้อมูลทำการสังเคราะห์ทั้งหมด 3 รอบ ผลลัพธ์ที่ได้จะเป็นเลขฐานสอง 128 บิต

3.1.2 เมสเซจไดเจสต์ 5 (Message Digest 5 – MD5)

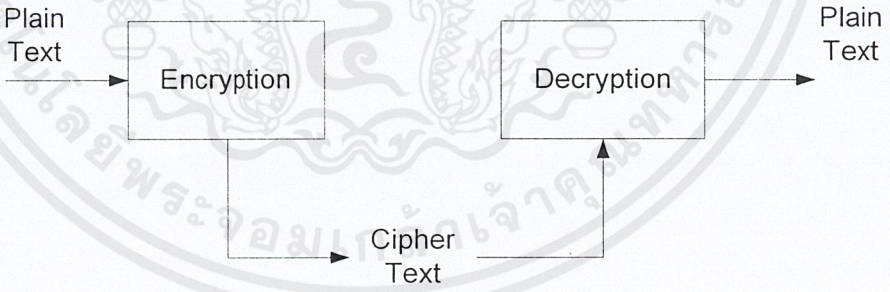
MD5 ถูกพัฒนาขึ้นโดยไรเวส (Rivest) เช่นเดียวกับ MD4 ในปี ค.ศ. 1991 อัลกอริทึมแบบ MD5 นี้จุดประสงค์ที่ทำขึ้นก็เพื่อให้มีการชนกันของข้อมูลน้อยกว่าแบบ MD4 แต่จะใช้เวลาในการสังเคราะห์มากกว่า การสังเคราะห์ข้อมูลจะทำการสังเคราะห์ทั้งหมด 4 รอบและอัลกอริทึมโดยทั่วไปลักษณะจะเหมือนกับ MD4 ผลลัพธ์ที่ได้จะเป็นเลขฐานสอง 128 บิต

3.1.3 ซีเคียวริตี้แฮชอัลกอริทึม (Secure Hash Algorithm – SHA)

SHA ถูกพัฒนาขึ้นโดย National Institute of Standards and Technology (NIST) และเผยแพร่เป็น Federal Information Processing Standard (FIB PUB 180) ในปี ค.ศ. 1994 มีการพิจารณาแก้ไขปรับปรุงแฮชอัลกอลิทึมแบบ SHA ใหม่โดยตั้งชื่อว่า SHA – 1 ผลลัพธ์ที่จากการสังเคราะห์ข้อมูลจะมีขนาด 160 บิต ใช้เวลาในการสังเคราะห์มากกว่าแบบ MD5 แต่ความปลอดภัยก็เพิ่มขึ้นตามไปด้วย

3.2 พื้นฐานการเข้ารหัสและถอดรหัส

การเข้ารหัส (Encryption) คือ กระบวนการในการเปลี่ยนข้อมูลต้นฉบับให้อยู่ในอีกรูปแบบหนึ่งที่ไม่สามารถเข้าใจได้โดยง่าย ผลลัพธ์ที่ได้จากการเข้ารหัสจะสามารถกลับไปเป็นข้อมูลต้นฉบับได้นั้นต้องใช้การถอดรหัส (Decryption) โดยเราจะเรียกข้อมูลต้นฉบับที่จะทำการเข้ารหัสว่า เคลียร์เท็กซ์ (Clear text) หรือ เพลนเท็กซ์ (Plain text) และเราจะเรียกข้อมูลที่ทำการเข้ารหัสเรียบร้อยแล้วว่า ไซเฟอร์เท็กซ์ (Cipher text), โค้ดเท็กซ์ (Code text) หรือ ไซเฟอร์ (Cipher) การเข้ารหัสและถอดรหัสสามารถเขียนไดอะแกรมแสดงได้ดังนี้



รูปที่ 3.1 แสดงการเข้ารหัสและถอดรหัส

ข้อมูลเพลนเท็กซ์ = P จะแสดงอยู่ในรูปอนุกรมดังนี้

P = [P1, P2, …, Pn] และเมื่อเข้ารหัสแล้วจะเปลี่ยนเป็น C = [C1, C2, …, Cn]

เขียนให้อยู่ในอีกรูปแบบ C = E(P), P = D(C) หรือ P = D(E(P))

C = Cipher text

P = Plain text

E = Encryption algorithms

D = Decryption algorithms

หลักการของการเข้ารหัสโดยทั่วไปมี 2 ประเภทคือ

1. การแทนที่ (Substitution) เป็นการแทนที่บิตใด ๆ ด้วยข้อมูลอื่น ทำให้ข้อมูลมีความสับสนยากต่อการถอดรหัส เช่น เรามีข้อความว่า PRIVATE แล้วใช้หลักการเพิ่มค่ารหัสแอสกี (ASCII) ของตัวอักษรแต่ละตัวในข้อความไปอีก 3 แล้วแทนที่ในข้อความต้นฉบับจะได้ข้อความออกมาเป็น SULYDWH เป็นต้น

2. การสับเปลี่ยนตำแหน่ง (Permutation) เป็นการสับเปลี่ยนตำแหน่งใด ๆ ของข้อมูล เมื่อมีการสับเปลี่ยนตำแหน่งมาก ๆ ทำให้ข้อมูลมีความซับซ้อนยากต่อการถอดรหัส เช่น เรามีข้อความว่า PRIVATE จะได้ข้อความที่จากการเข้ารหัสเป็น VRIPTEA เป็นต้น

3.3 การเข้ารหัสและถอดรหัสโดยใช้คีย์ (Encryption and Decryption with key)

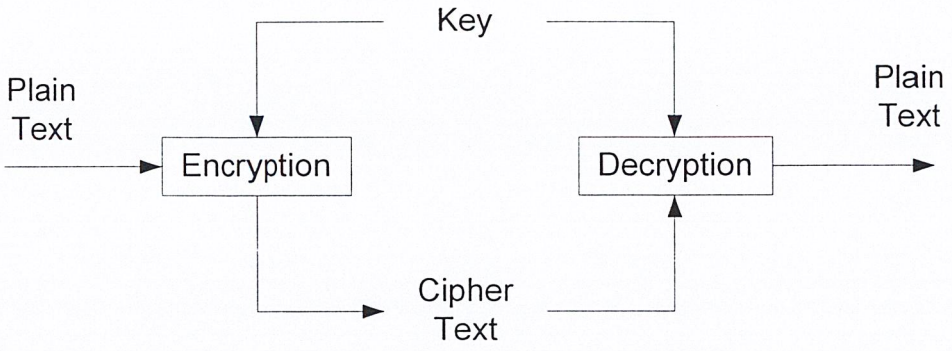
ระบบการเข้ารหัสและถอดรหัสโดยใช้คีย์นั้น การเข้ารหัสและถอดรหัสข้อมูลจะเปลี่ยนแปลงไปตามคีย์นอกจากอัลกอริทึมที่ใช้แล้ว ถึงผู้อื่นรู้อัลกอริทึมในการเข้ารหัสแต่ไม่รู้คีย์ก็ไม่สามารถถอดรหัสออกมาได้ กระบวนการเข้ารหัสและถอดรหัสแบบนี้มีอยู่ 2 ประเภทคือ

1. ระบบการเข้ารหัสแบบสมมาตร (Symmetric Cryptosystem) เป็นระบบที่การเข้ารหัสและการถอดรหัสใช้รูปแบบและคีย์เดียวกัน เช่น DES (Data Encryption Standard), 3DES (Triple DES), IDEA (International Data Encryption), CDMF (Commercial Data Masking Facility) เป็นต้น

2. ระบบการเข้ารหัสแบบไม่สมมาตร (Asymmetric Cryptosystem) เป็นระบบการเข้ารหัสที่ใช้ 2 คีย์ที่มีความเกี่ยวข้องกันทางคณิตศาสตร์ โดยประกอบด้วยคีย์สาธารณะ (Public Key) และคีย์ส่วนตัว (Private Key) โดยถ้าการเข้ารหัสทำโดยการใช้คีย์สาธารณะการถอดรหัสต้องใช้คีย์ส่วนตัวที่เป็นคู่ของมันในการถอดรหัสเท่านั้น ในทางตรงกันข้ามถ้าใช้คีย์ส่วนตัวในการเข้ารหัสต้องใช้คีย์สาธารณะในการถอดรหัสเท่านั้นเช่นกัน เช่น RSA, DH, DSA

3.4 การเข้ารหัสแบบสมมาตร

การเข้ารหัสแบบสมมาตรทำงานโดยการเข้ารหัสจะใช้คีย์เข้าร่วมกับอัลกอริทึมในการเข้ารหัสและเมื่อทำการถอดรหัสจะใช้คีย์เดียวกับที่ใช้ในการเข้ารหัสเข้าร่วมกับอัลกอริทึมแบบเดียวกัน แสดงการทำงานได้ดังรูป



รูปที่ 3.2 แสดงการเข้ารหัสและถอดรหัสแบบสมมาตร

การเข้ารหัสแบบนี้อาจเรียกว่า การเข้ารหัสแบบคีย์เดี่ยว (Single Key Encryption) หรือการเข้ารหัสแบบซีเครทคีย์ (Secret Key Encryption) เพราะว่าทั้งการเข้ารหัสและการถอดรหัสจะใช้คีย์เดียวกัน

สามารถแสดงด้วยสมการได้ดังนี้

$C = E(K, P)$ และ $P = D(K, C)$ หรือได้ว่า $P = D(K, (K, C))$

โดย P = เพลนเท็กซ์

C = ไซเฟอร์เท็กซ์

E = อัลกอริทึมการเข้ารหัส

D = อัลกอริทึมการถอดรหัส

K = ซีเครทคีย์ที่ใช้ในการเข้ารหัสและการถอดรหัส

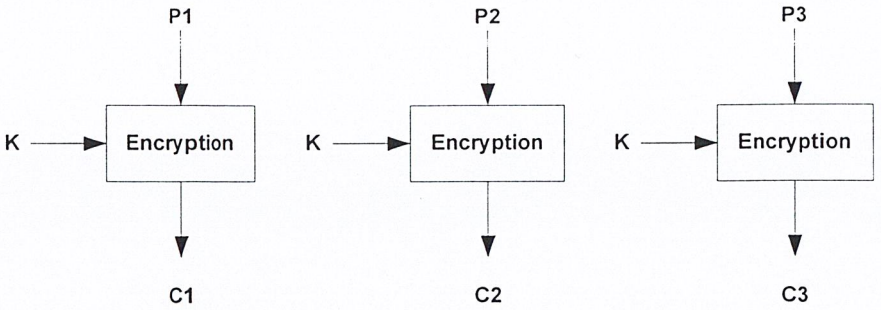
3.4.1 ลักษณะที่สำคัญของการเข้ารหัสแบบสมมาตร

3.4.1.1 การเข้ารหัสแบบสมมาตรเพื่อความลับ จะต้องมีอัลกอริทึมการเข้ารหัสที่แข็งแกร่งเพียงพอ กล่าวคือไม่ว่าผู้ใดรู้ถึงอัลกอริทึมการเข้ารหัสจะต้องไม่สามารถหาเพลนเท็กซ์และซีเครทคีย์จากไซเฟอร์เท็กซ์ ทำให้การส่งข้อมูลจะต้องมีเฉพาะผู้รับและผู้ส่งเท่านั้นที่รู้ซีเครทคีย์ ถ้าเมื่อใดที่มีผู้อื่นรู้ซีเครทคีย์นี้และรู้อัลกอริทึมที่ใช้ในการถอดรหัสก็จะสามารถหาเพลนเท็กซ์ได้ การแปลงเพลนเท็กซ์เป็นไซเฟอร์เท็กซ์ ใช้การกระทำ 2 รูปแบบคือ ทั้งการแทนที่และการเปลี่ยนตำแหน่ง

3.4.1.2 การเข้ารหัสและการถอดรหัสใช้คีย์เดียวกัน

3.4.1.3 การเข้ารหัสสามารถใช้บล็อกไซเฟอร์ในการเข้ารหัสได้ ซึ่งลักษณะของบล็อกไซเฟอร์แบบต่าง ๆ มีดังต่อไปนี้

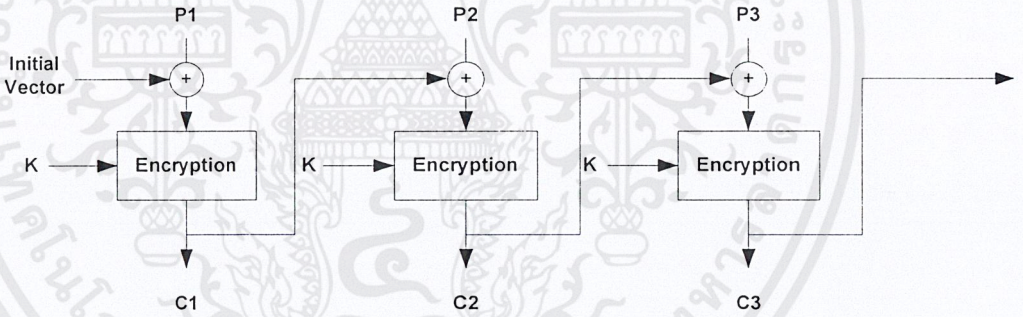
1. อิเล็กทรอนิกส์โค้ดบุ๊ก (Electronic Codebook : ECB) บล็อกไซเฟอร์แบบนี้จะแบ่งเพลนเท็กซ์ออกเป็นบล็อก บล็อกละเท่า ๆ กันใช้คีย์และอัลกอริทึมเดียวกันดังรูป



รูปที่ 3.3 แสดงการเข้ารหัสแบบอิเล็กทรอนิกส์โค้ดบุ๊ก

ข้อดีสำหรับบล็อกไซเฟอร์แบบอิเล็กทรอนิกส์โค้ดบุ๊ก คือสำหรับข้อมูลสั้น ๆ จะได้ไซ-เฟอร์เท็กซ์ที่ดีและสามารถทำที่ละหลายบล็อกทำให้เร็ว แต่ถ้าข้อมูลยาวมาก ๆ อาจมีข้อมูลซ้ำกัน เมื่อทำการเข้ารหัสจะทำให้ได้ไซเฟอร์เท็กซ์ที่มีลักษณะซ้ำกันได้

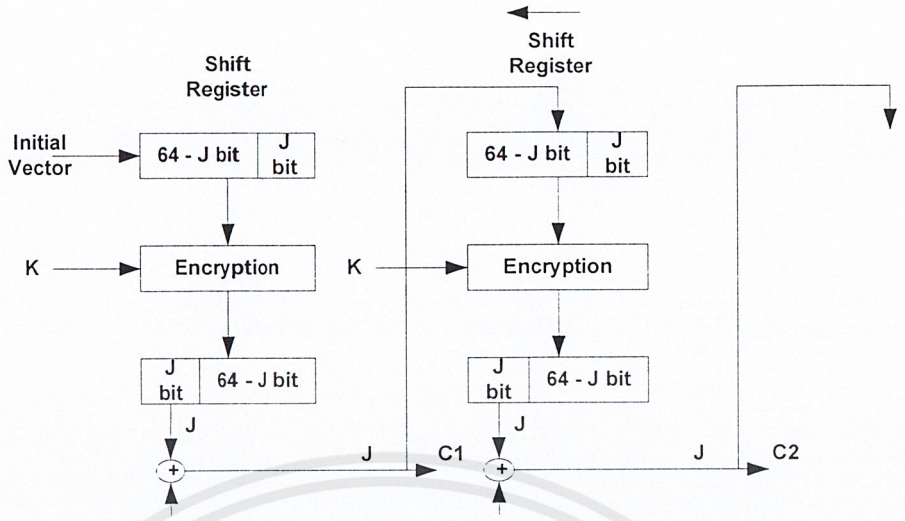
2. ไซเฟอร์บล็อกเชนนิ่ง (Cipher Block Chaining : CBC) แบ่งเพลนเท็กซ์ ออกเป็น บล็อก ๆ ละเท่า ๆ กัน โดยก่อนการเข้ารหัสนำเพลนเท็กซ์มาเอ็กซ์คลูซีฟออร์ (XOR) กับไซเฟอร์เท็กซ์ก่อนหน้ามัน โดยเพลนเท็กซ์แรกจะทำการ XOR กับเวคเตอร์เริ่มต้น (Initial Vector : IV) ดังรูป



รูปที่ 3.4 แสดงการเข้ารหัสแบบไซเฟอร์บล็อกเชนนิ่ง

ข้อดีของบล็อกไซเฟอร์ลักษณะนี้คือสามารถแก้ปัญหาซ้ำกันของข้อมูลก่อนเข้ารหัสเพราะต้องนำเพลนเท็กซ์มา XOR กับไซเฟอร์เท็กซ์ก่อนหน้ามันก่อนที่จะนำมาเข้ารหัส แต่มีข้อเสียเนื่องจากการเข้ารหัสแต่ละบล็อกต้องรอบล็อกข้างหน้ามันก่อนจึงทำการเข้ารหัสได้จึงไม่สามารถทำพร้อมกันได้จึงทำให้ช้า

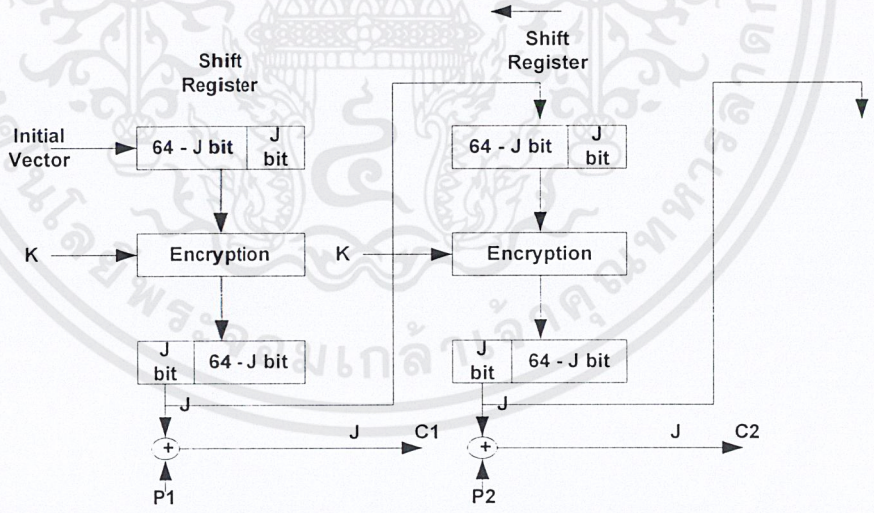
3. ไซเฟอร์ฟีดแบ็ก (Cipher Feed Back : CFB) ขั้นแรกมี ชิฟต์รีจิสเตอร์ (Shift Register) ขนาด 64 บิตโดยกำหนดค่าเริ่มต้นเป็นเวคเตอร์เริ่มต้น (Initial Vector) แล้วนำมาเข้ารหัสกับคีย์ (K) จากนั้นจำ J บิตหน้าสุดนำมา XOR เพลนเท็กซ์แล้วนำไซเฟอร์เท็กซ์ที่ได้ไปใช้ใน ชิฟต์รีจิสเตอร์ J บิตท้าย โดยเลื่อนไปด้านซ้าย ไป J บิต เพื่อนำข้อมูลไซเฟอร์เท็กซ์เข้ามาดังรูป



รูปที่ 3.5 แสดงการเข้ารหัสแบบไซเฟอร์ฟีดแบ็ก

โดยที่ส่วนใหญ่จะทำทีละ 8 บิต ซึ่งคือ 1 ตัวอักษร (Character) โดยสามารถทำงานแบบ เวลาจริงได้ (Real Time) ได้

4. เอาต์พุตฟีดแบ็ก (Output Feedback : OFB) เป็นบล็อกไซเฟอร์ที่ทำงานคล้ายกับไซเฟอร์ฟีดแบ็กเพียงแต่นำข้อมูลที่ได้จากการเข้ารหัส J บิตแรกไปใส่ในชิฟต์รีจิสเตอร์ J บิตท้ายแทนดังรูป



รูปที่ 3.6 แสดงการเข้ารหัสแบบเอาต์พุตฟีดแบ็ก

ข้อดีของเอาต์พุตฟีดแบ็กความผิดพลาดจะไม่แพร่ไปยังบล็อกถัดไป คือถ้าเกิดความผิดพลาดจากไซเฟอร์เท็กซ์บล็อกแรกบล็อกต่อมาจะไม่ได้รับผลกระทบจากบล็อกแรกเพราะนำข้อมูล J บิตก่อนทำการ XOR กับข้อมูลเพลนเท็กซ์มาใส่ในชิฟต์รีจิสเตอร์แทน

3.4.2 DES (Data Encryption Standard)

3.4.2.1 ประวัติและที่มาของ DES

ในปลายทศวรรษที่ 1960 บริษัท IBM ได้จัดตั้งโครงงานวิจัยทางด้านการเข้ารหัสด้วยคอมพิวเตอร์ (Computer Cryptography) ซึ่งนำโดยฮอสท์ เฟสเทล (Horst Feistel) ซึ่งโครงงานนี้เสร็จสิ้นในปี 1971 ซึ่งผลงานวิจัยของโครงงานนี้คือลูซิเฟอร์ (LUCIFER[FEIS73])โดยมีลักษณะเป็นการเข้ารหัสข้อมูลเป็นบล็อกขนาด 64 บิตและใช้คีย์ขนาด 128 บิต ซึ่งต่อมาได้ถูกพัฒนาขนาดของคีย์ให้ลดลงเหลือขนาด 56 บิต

โดยอัลกอริทึมของการเข้ารหัสข้อมูลของลูซิเฟอร์ได้ถูกพัฒนาโดย IBM สำหรับ NBS (National Bureau of Standards) อัลกอริทึมนี้ได้เป็นที่รู้จักในนามของ DES (Data Encryption Standard) ถึงแม้ว่าชื่อจริงของมันคือ DEA (Data Encryption Algorithm) ในสหรัฐและ DEAI (Data Encryption Algorithm-1) ในอีกหลาย ๆ ประเทศ

3.4.2.2 รายละเอียดของ DES

เป็นวิธีการเข้ารหัสที่ใช้กันอย่างแพร่หลายที่เป็นพื้นฐานบน Data Encryption Standard (DES) ที่ได้พัฒนาขึ้นในปี 1977 โดย National Bureau of Standards ซึ่งปัจจุบันคือ Federal Information Processing Standard 46 (FIPS PUB46) สำหรับ DES ข้อมูลจะถูกเข้ารหัสเป็นบล็อกขนาด 64 บิต ซึ่งใช้คีย์ขนาด 56 บิต โดยวิธีการจัดการกับข้อมูล 64 บิตที่เข้ามาเพื่อแปลงเป็นข้อมูล 64 บิตออกไป และใช้คีย์ตัวเดียวกันนี้ในการถอดรหัส

แม้ว่า DES ถูกนำมาใช้ตั้งแต่ช่วงทศวรรษที่ 70 (ค.ศ.1960-1970) และได้รับการตอบรับอย่างดีจากเหล่านักวิเคราะห์รหัส (Cryptanalysis) อย่างแพร่หลาย แต่ก็เป็นข้อถกเถียงกันเป็นอย่างมากถึงเรื่อง DES นั้นจะปลอดภัยหรือไม่และมีความปลอดภัยมากน้อยแค่ไหน แต่จนถึงปัจจุบันเราก็ยังไม่พบช่องโหว่ของ DES ตามเอกสารที่ตีพิมพ์เป็นสาธารณะ แม้ว่าจะใช้คีย์เพียงไม่กี่บิตก็ตาม ในทางตรงกันข้ามแนวความคิดแบบ IDEA กลับใช้คีย์แบบ 128 บิต(ซึ่งมีขนาดกว่า 2 เท่าของ DES) และได้รับการตอบรับจากสาธารณะตั้งแต่ทศวรรษที่ 90 (ค.ศ.1980-1990) (แต่ดีไม่เท่าตอนประกาศใช้ DES) IDEA มีความปลอดภัยมากกว่า DES และสามารถประมวลผลได้เร็วกว่า DES อย่างไรก็ตาม IDEA ยังต้องรอการตรวจสอบจากผู้เชี่ยวชาญอีกมากถึงเรื่องช่องโหว่ของความปลอดภัย

3.4.2.3 อัลกอริทึมของการเข้ารหัสแบบ DES

โดยพิจารณาออกเป็น 2 ส่วนเพื่อให้ง่ายแก่การทำความเข้าใจคือ ส่วนที่เป็นคีย์ที่จะใช้ในการเข้ารหัสและส่วนที่เป็นข้อมูลที่จะนำมาทำการเข้ารหัส

อัลกอริทึมเป็นผลมาจากทฤษฎีของแชนนอน (Shannon) ซึ่งเกี่ยวกับการปิดบังข่าวสาร ซึ่งแนะนำ 2 วิธีในการปกปิดข่าวสารนั่นคือคอนฟิวชัน (Confusion) และดิฟฟิวชัน (Diffusion)

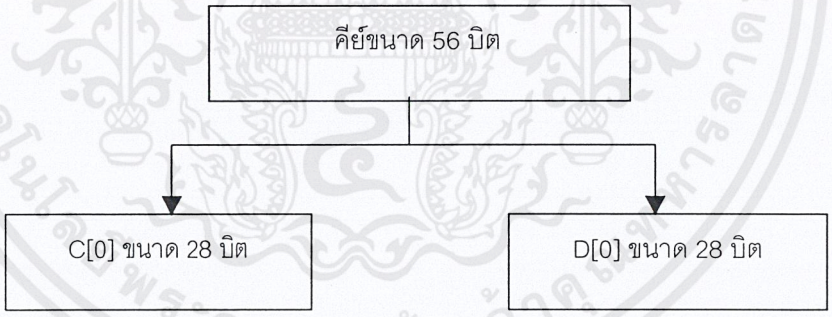
คอนฟิวชัน คือการทำให้ชิ้นส่วนของข่าวสารถูกเปลี่ยนไป ดังนั้นเอาต์พุตบิตจะสังเกตไม่เห็นความสัมพันธ์กับอินพุตบิต

ดิฟฟิวชัน จะทำการกระจายเพลนเท็กซ์บิตไปที่บิตอื่นในไซเฟอร์เท็กซ์ มีผลทำให้ข้อมูลต่าง ๆ มีความซับซ้อนมากขึ้น เพราะข้อมูลจะถูกกระจายไปในตำแหน่งอื่นด้วย

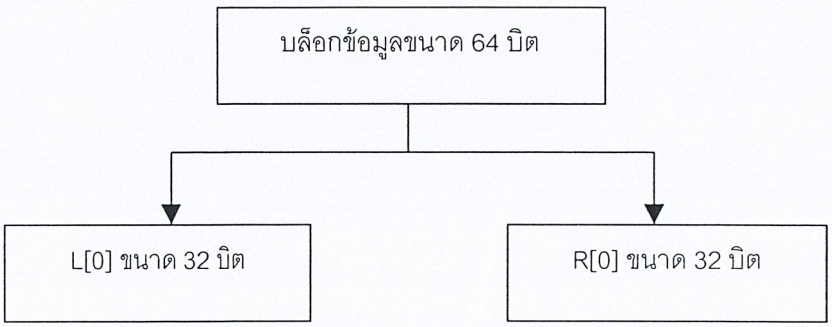
อัลกอริทึม DES ที่กระทำบนบล็อกของข้อมูล บล็อกของข้อมูลจะถูกแยกออกเป็น 2 ส่วน แต่ละส่วนจะแยกออกเป็นอิสระต่อกัน จากนั้นจะทำการรวมคีย์กับส่วนใดส่วนหนึ่งของข้อมูลและก็สลับกัน 2 ส่วนโปรเซส (Process) นี้จะทำซ้ำ 16 ครั้ง อัลกอริทึมในการทำซ้ำจะใช้เทเบิลลุกอัพ (Table lookup) และการคำนวณบิตแบบง่าย ๆ (simple bit) ถึงแม้ว่าการจัดการระดับบิตของอัลกอริทึมจะยุ่งยากซับซ้อน

อินพุตที่เข้ามา DES จะแบ่งอินพุตออกเป็นบล็อก ๆ ละ 64 บิต ซึ่งจะถูกเปลี่ยนไปใช้คีย์ขนาด 64 บิต ข้อมูลขนาด 64 บิต จะถูกสับเปลี่ยนตำแหน่งโดยการสับเปลี่ยนตำแหน่งเริ่มต้น (Initial Permutation) และคีย์จะถูกลดลงจาก 64 บิต เหลือ 56 บิต โดยการทิ้งบิตที่ 8, 16, 24, ..., 64 ซึ่งบิตเหล่านี้จะถูกกำหนดเป็นพาริตีบิต โดยแสดงไว้ในรูปที่ 3.1

ข้อมูล 64 บิตที่ถูกสับเปลี่ยนตำแหน่งแล้วจะถูกแบ่งเป็นครึ่งซ้ายและครึ่งขวา (แต่ละครึ่งมีขนาด 32 บิต) แสดงไว้ในรูปที่ 3.2 คีย์จะถูกชิฟต์ไปทางซ้ายโดยการกำหนดที่จำนวนบิตและจะทำการสลับตำแหน่ง ต่อไปคีย์จะถูกรวมกับครึ่งขวา หลังจากนั้นก็จะมารวมกับครึ่งซ้ายใหม่อีกครั้ง ผลลัพธ์ของการรวมนี้จะเปลี่ยนเป็นครึ่งด้านขวาใหม่ ส่วนครึ่งขวาเก่าจะกลายมาเป็นครึ่งซ้ายใหม่ กิจกรรมเหล่านี้จะทำเป็นวัฏจักร (Cycle) วัฏจักรจะถูกทำซ้ำ 16 ครั้ง หลังจากวัฏจักรสุดท้ายซึ่งเป็นการสับเปลี่ยนครั้งสุดท้ายซึ่งจะถูกสับเปลี่ยนตำแหน่งบิตผกผันกับแบบเริ่มต้น (Inverse Initial Permutation - IP) หรือการสับเปลี่ยนตำแหน่งบิตผกผันกับแบบเริ่มต้น ตามตารางจะได้ผลลัพธ์สุดท้ายออกมา คือข้อมูลที่ถูกทำการเข้ารหัสแล้ว



รูปที่ 3.7 แสดงการแบ่งคีย์ 56 บิตที่เรียงลำดับบิตแล้วออกเป็น 2 ส่วน



รูปที่ 3.8 แสดงบล็อกข้อมูล 64 บิต ที่แบ่งออกเป็น 2 ส่วน หลังจากทำการจัดเรียงบิตแล้ว

3.4.3 จุดเด่นและจุดด้อยของการเข้ารหัสแบบสมมาตร

การเข้ารหัสแบบแบบสมมาตรถูกกระทำในสองทิศทางระหว่างผู้ส่งและผู้รับ คือ ผู้ส่งและผู้รับสามารถที่จะเข้ารหัสข้อมูลแล้วส่งไปหาอีกคนได้ และในขณะเดียวกันนั้นก็สามารถที่จะถอดรหัสข้อมูลนั้นมาดูโดยการใช้คีย์เดียวกันนี้ แต่มักจะมีจุดเด่นและจุดด้อยดังนี้

1. อัลกอริทึมที่ใช้นั้นง่าย ไม่ค่อยยุ่งยากซับซ้อนแต่การเข้ารหัสและถอดรหัสสามารถทำได้รวดเร็วซึ่งเป็นข้อได้เปรียบของวิธีการเข้ารหัสและถอดรหัสแบบนี้

2. การเก็บคีย์ที่ใช้ในการเข้ารหัสเป็นเรื่องที่สำคัญ คีย์ที่ใช้นั้นต้องเก็บเป็นความลับถ้าคีย์ไม่เป็นความลับหรือถูกขโมย บุคคลที่รู้คีย์นั้นสามารถที่จะถอดรหัสข้อมูลที่เข้ารหัสนั้นได้ นอกจากนี้ยังอาจปลอมแปลงข้อมูลเดิมนั้นใหม่ แล้วเข้ารหัสข้อมูลที่ปลอมแปลงนั้นด้วยคีย์เดียวกัน แล้วส่งไปที่ผู้รับตัวจริง ดังนั้นเพื่อความปลอดภัยควรเลือกคีย์ที่ยากแก่การเดาและเก็บไว้อย่างปลอดภัย รวมทั้งไม่ควรใช้คีย์เดียวกันนี้ซ้ำกันหลาย ๆ ครั้ง

3. การส่งคีย์ไปพร้อมกับข้อมูลที่เข้ารหัสนั้น อาจทำให้เกิดปัญหาได้ ซึ่งถ้าทำอย่างนั้นแล้วคีย์ที่ส่งไปด้วยนั้นจะต้องถูกส่งไปด้วยความปลอดภัยที่สูงมาก ยิ่งเป็นการส่งไปในระยะทางไกล ๆ เช่น คนละเครือข่าย ซึ่งจะทำได้ยากมากเลยทีเดียว วิธีที่ง่ายก็คือให้ส่งคีย์ให้กับผู้รับด้วยมือของคนส่งเอง แต่อาจจะทำได้ไม่สะดวกและเสียเวลาอีกวิธีหนึ่งก็คือการส่งคีย์ไปพร้อมกับข้อมูลนั้น แต่แบ่งคีย์นั้นออกเป็นส่วน ๆ ก่อนแล้วจึงส่งไปตามเส้นทางที่ต่าง ๆ กัน ซึ่งถึงแม้ว่าคีย์บางส่วนจะถูกดักได้แต่ก็ไม่สามารถรู้ถึงตัวคีย์ที่สมบูรณ์จริง ๆ ได้

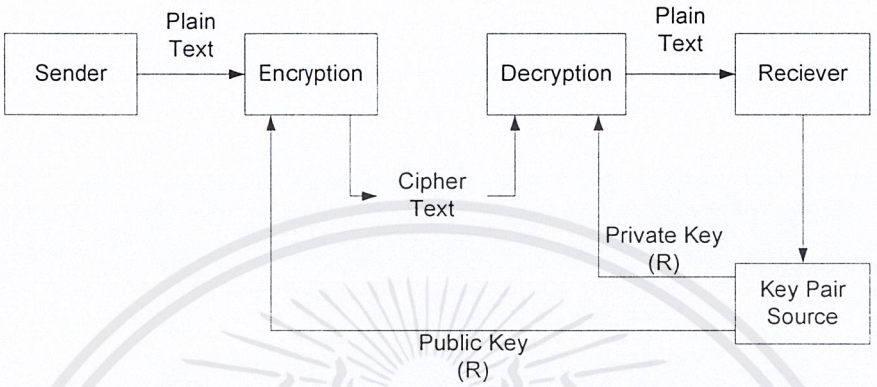
3.5 การเข้ารหัสแบบไม่สมมาตร

แนวความคิดของการเข้ารหัสแบบไม่สมมาตรหรือการเข้ารหัสแบบคีย์สาธารณะ (Public Key Encryption) ได้เกิดขึ้นจากการพยายามแก้ไขปัญหาของการเข้ารหัสแบบสมมาตร 2 ข้อด้วยกันคือ การจัดการคีย์ (Key Management) และการลงนามรับรองข่าวสารทางดิจิตอล (Digital Signature) ใช้เพื่อการพิสูจน์สิทธิ์ ระบบการเข้ารหัสเป็นตัวอย่างหนึ่งของการป้องกันไม่ให้มีการเปิดเผยข้อมูลแก่ผู้ที่ไม่มีสิทธิ์

3.5.1 ลักษณะที่สำคัญของการเข้ารหัสแบบไม่สมมาตร

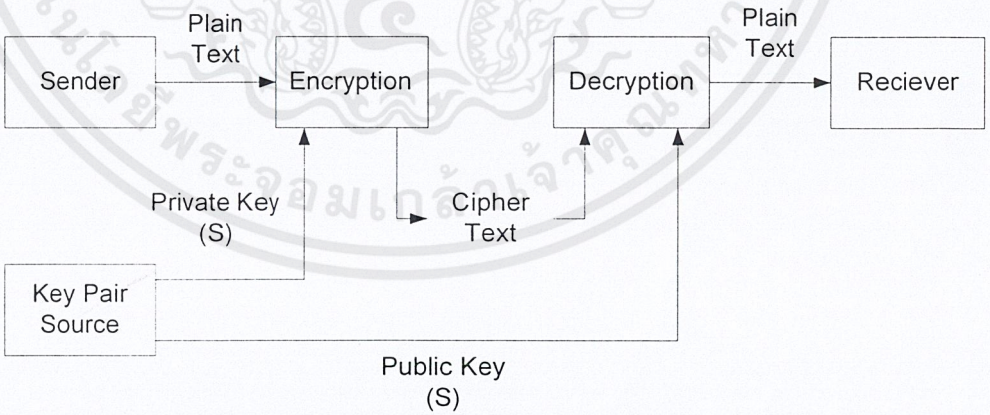
วิธีการนี้เป็นการเข้ารหัสและการถอดรหัสที่ใช้คีย์คนละดอกกันโดยคีย์ทั้งสองต้องเป็นคู่คีย์ (Key Pair) กัน คือ คีย์สาธารณะ (Public Key) ซึ่งเป็นคีย์ที่ทำการแจกให้ผู้อื่น และคีย์ส่วนตัว (Private Key) เป็นคีย์ที่เก็บไว้เป็นความลับ ความสามารถของการเข้ารหัสแบบไม่สมมาตรมีอยู่ด้วยกัน 3 ข้อ คือ

1. ความลับ (Secrecy) หมายถึง ไม่ยอมให้มีบุคคลที่ไม่มีสิทธิ์เข้ามาดูข้อมูลได้ ซึ่งสามารถทำได้โดยผู้ส่งเข้ารหัสโดยใช้คีย์สาธารณะของผู้รับ ซึ่งทำให้มีแต่ผู้รับที่มีคีย์ส่วนตัวที่เป็นคู่คีย์ของมันเท่านั้นที่สามารถทำการถอดรหัสได้ ถึงแม้คีย์สาธารณะมีบุคคลอื่นรู้ก็ไม่สามารถทำการถอดได้ดังรูป



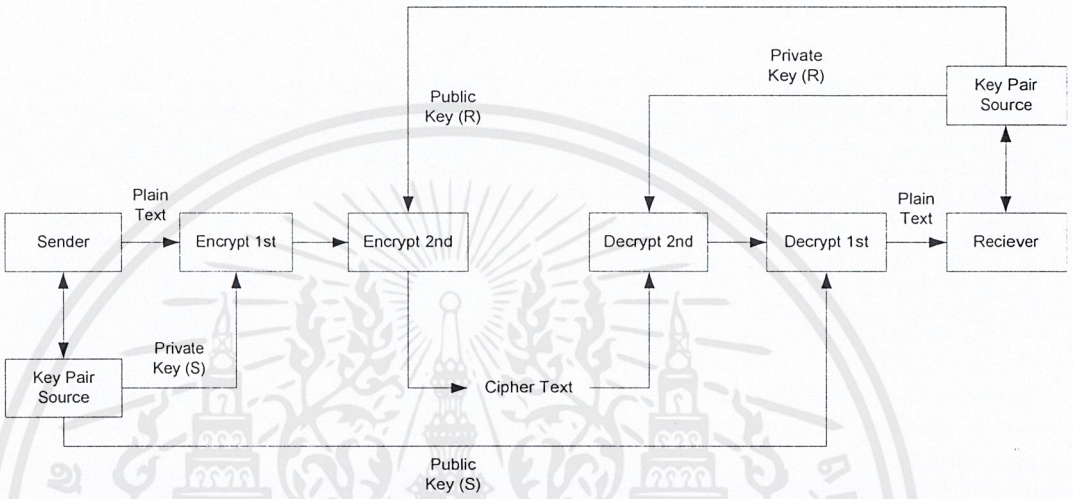
รูปที่ 3.9 แสดงการเข้ารหัสแบบไม่สมมาตรแบบเป็นความลับ

2. การพิสูจน์บุคคล (Authenticity) หมายถึง การตรวจสอบที่มาของข้อมูล ว่าถูกส่งมาจากผู้ส่งคนนั้นจริงหรือไม่ ซึ่งทำโดยเข้ารหัสข้อมูลโดยใช้คีย์ส่วนตัวของผู้ส่ง การตรวจสอบทำได้โดยใช้คีย์สาธารณะที่เป็นคู่คีย์ทำการถอดรหัส ซึ่งผู้ที่จะสามารถเข้ารหัสได้นั้นต้องเป็นผู้ที่เป็นเจ้าของคีย์ส่วนตัวนั้นเท่านั้น ส่วนมากการพิสูจน์ตัวบุคคลนั้นผู้ส่งจะทำการส่งข้อมูลที่เป็นข้อมูลส่วนตัวเข้ารหัสโดยคีย์ส่วนตัวไปให้ผู้รับ โดยผู้รับนำเอาข้อมูลมาทำการถอดรหัสแล้วตรวจดูว่าข้อมูลส่วนตัวของผู้ส่งเป็นจริงหรือไม่ โดยการเข้ารหัสเพื่อพิสูจน์ตัวบุคคลแสดงดังรูป



รูปที่ 3.10 แสดงการเข้ารหัสแบบไม่สมมาตรแบบพิสูจน์บุคคล

3. การพิสูจน์บุคคลและความลับ (Authenticity and Secrecy) หมายถึง การตรวจสอบที่มาของข้อมูลและจำกัดสิทธิ์ข้อมูลให้เพียงแต่ผู้รับเท่านั้นที่สามารถอ่านข้อมูลได้ สามารถทำได้โดยผู้ส่งทำการนำข้อมูลมาเข้ารหัสครั้งแรกด้วยคีย์ส่วนตัวของตนเพื่อเป็นการพิสูจน์ตัวบุคคลจากนั้นนำมาเข้ารหัสโดยคีย์สาธารณะของผู้รับเพื่อเป็นการรักษาความลับให้ผู้รับที่มีคีย์ส่วนตัวที่เป็นคู่คีย์เท่านั้นที่สามารถเปิดอ่านได้ เมื่อผู้รับได้รับไซเฟอร์เท็กซ์มาก็นำมาถอดรหัสโดยใช้คีย์ส่วนตัวของตัวเองเปิด จากนั้นนำมาถอดรหัสโดยใช้คีย์สาธารณะที่เป็นคู่คีย์ของผู้ส่ง ดังแสดงในรูป



รูปที่ 3.11 แสดงการเข้ารหัสแบบไม่สมมาตรแบบพิสูจน์บุคคลและความลับ

3.5.2 อาร์เอสเอ (RSA)

3.5.2.1 หลักการทำงานของอาร์เอสเอ

วิธีการของระบบ RSA ใช้ประโยชน์จากรูปแบบสมการเอ็กซ์โปเนนท์เชียล (Exponential) โดยเพลนเท็กซ์จะถูกเข้ารหัสเป็นบล็อก (Ciphertext block) แต่ละบล็อกมีค่าเป็นเลขฐานสองซึ่งมีค่าน้อยกว่าค่า n สำหรับบล็อกเพลนเท็กซ์ M บล็อกไซเฟอร์เท็กซ์ C การเข้ารหัสและถอดรหัสจะอยู่ในรูปแบบดังต่อไปนี้

$C = M^e \bmod n$

$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$

ทั้งผู้ส่งและผู้รับต้องรู้ค่าของ n ผู้ส่งรู้ค่า e และมีผู้รับเพียงคนเดียวที่รู้ค่า d โดยกำหนดให้การเข้ารหัสประกอบด้วย KU = (e, n) และในการถอดรหัสต้องมี KR = (d, n) สิ่งสำคัญที่ต้องการสำหรับระบบนี้มีอยู่ 3 ประการก็คือ

1. ต้องสามารถหาค่า e, d และ n ที่ทำให้ $M^{ed} = M \bmod n$ สำหรับ M ทุกค่าที่ $M < n$
2. ค่า C^d และ M^e ต้องสามารถคำนวณได้โดยง่ายสำหรับค่า M ทุก ๆ ค่าที่ $M < n$
3. ระบบต้องใช้ปัญหาทางคณิตศาสตร์ที่ยากพอที่จะไม่ให้สามารถคำนวณค่า d จากค่า e และ n ได้

วิธีในการสร้างคีย์มีดังนี้

1. เลือกเลขจำนวนเฉพาะ p, q (เลือกคีย์ส่วนตัว)
2. คำนวณหาค่า $n = p \times q$ (คำนวณหาคีย์สาธารณะ)
3. เลือกเลขจำนวนเต็ม d เมื่อ ห.ร.ม. ($\Phi^{(n)}$, d) โดยที่ $1 < d < \Phi^{(n)}$ (คำนวณหาคีย์ส่วนตัว)
4. คำนวณหาค่า e เมื่อ $e = d^{-1} \bmod \Phi^{(n)}$ (เลือกคีย์สาธารณะ)
5. กำหนดให้คีย์สาธารณะเป็น KU = (e, n)
6. กำหนดคีย์ส่วนตัวเป็น KR = (d, n)

คีย์ส่วนตัวประกอบด้วย (d, n) และคีย์สาธารณะประกอบด้วย (e, n) นาย ก. ประกาศคีย์สาธารณะของเขาออกไป เมื่อนาย ข. ต้องการส่งข่าวสาร (สมมติให้เป็น M) ที่เป็นความลับให้แก่นาย ก. นาย ข. ต้องใช้คีย์สาธารณะของนาย ก. เพื่อนำมาใช้ในการคำนวณหรือเข้ารหัสออกมาเป็นไซเฟอร์เท็กซ์ $C = M^e \bmod n$ แล้วส่ง C ไปให้กับนาย ก. และนาย ก. จึงทำการคำนวณหรือถอดรหัสให้กลับเป็นข่าวสาร M เหมือนเดิมโดย $M = C^d \bmod n$

3.5.2.2 การคำนวณการเข้ารหัสและถอดรหัสของอาร์เอสเอ

การคำนวณการเข้ารหัสและการถอดรหัสจะเกี่ยวกับการเพิ่มเลขจำนวนเต็มให้เป็นเลขจำนวนเต็มยกกำลังแล้วนำมามอดูลโลด้วย n การลดเวลาของการทำงานกับเลขจำนวนเต็มที่มีค่ามาก ๆ เราสามารถใช้คุณสมบัติของการมอดูลโลดังนี้ $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

ขั้นตอนการสร้างคีย์นั้นต้องหาเลขจำนวนเฉพาะ p และ q ที่มีขนาดมาก ๆ และเลือกค่า e และ d ซึ่งเราอาจนำค่า e และ d ที่เลือกมานี้ไปคำนวณอย่างอื่น เนื่องจากค่า $n = pq$ ใช้สำหรับป้องกันการโจมตีโดย วิธีบรูทฟอร์ท (Brute Force) ที่ใช้การลองทุกค่าที่เป็นไปได้ทีละค่าในการแยกค่า p และ q ออกมา จึงต้องเลือกค่าของ p และ q จากเซตขนาดใหญ่และต้องมีค่ามาก ๆ

วิธีหนึ่งที่จะหาเลขจำนวนเฉพาะขนาดใหญ่จริง ๆ ก็คือการเลือกเลขจำนวนคี่ที่ได้จากการสุ่มลำดับของค่าที่ต้องการแล้วทดสอบว่ามันเป็นจำนวนเฉพาะหรือไม่ ถ้าใช่ก็ดำเนินขั้นตอนต่อไป แต่ถ้าไม่ใช่ก็เลือกจำนวนคี่มาอีกตัวหนึ่งแล้วทดสอบ ต่อไปนี้เป็นตัวอย่างง่าย ๆ

1. เลือก p = 11 และ q = 13 ดังนั้น n = 143
2. $\Phi(n) = (p-1) \times (q-1)$, $\Phi(n) = 120$ แล้วให้ e = 11
3. ดังนั้น d = 11 จาก $d = e^{-1} \bmod 120$
4. ให้ M = 7 (Plain text)
5. สร้างคีย์สาธารณะ KU = (11, 143)
6. สร้างคีย์ส่วนตัว KR = (11, 143)
7. แปลงเพลนเท็กซ์ให้เป็นไซเฟอร์เท็กซ์โดยคำนวณ $7^{11} \bmod 143 = 106$
8. เราคำนวณหาไซเฟอร์เท็กซ์ C ได้ออกมาเท่ากับ 106

9. นำมาคำนวณกลับแปลงจากไซเฟอร์เท็กซ์เป็นเพลนเท็กซ์ $106^{11} \bmod 143 = 7$

10. เป็นค่าของเพลนเท็กซ์เริ่มต้น = M

3.5.3 จุดเด่นและจุดด้อยของการเข้ารหัสแบบไม่สมมาตร

จุดเด่นและจุดด้อยของการเข้ารหัสแบบไม่สมมาตรมีดังนี้

1. การเข้ารหัสค่อนข้างช้า และต้องใช้การคำนวณอย่างมาก

2. สามารถเข้ารหัสข้อมูลให้เป็นความลับได้ นอกจากนี้ยังสามารถตรวจสอบที่มาของข้อมูลบุคคลโดยใช้ร่วมกับลายมือชื่อดิจิตอลได้อีกด้วย

จากการเข้ารหัสและการถอดรหัสที่ใช้คีย์ ส่วนที่สำคัญนอกจากอัลกอริทึมที่ใช้ก็คือ คีย์ หากคีย์ที่ใช้มีความยาวมากก็จะทำให้ยากแก่การที่บุคคลอื่นจะคาดเดาเพื่อทำการถอดรหัสได้ ตารางต่อไปนี้แสดงระยะเวลาที่ใช้ในการที่ถอดรหัสโดยมีความยาวคีย์แบบต่าง ๆ

ค่าใช้จ่าย	ความยาวของคีย์ (บิต)				
	40	56	64	80	128
$ 100,000	2 วินาที	35 ชั่วโมง	1 ปี	70,000 ปี	10^{19} ปี
$ 1 ล้าน	0.2 วินาที	3.5 ชั่วโมง	37 วัน	7,000 ปี	10^{18} ปี
$ 100 ล้าน	2 มิลลิวินาที	2 นาที	9 ชั่วโมง	70 ปี	10^{16} ปี
$ 1,000 ล้าน	0.2 มิลลิวินาที	13 วินาที	1 ชั่วโมง	7 ปี	10^{15} ปี
$ 100,000 ล้าน	2 ไมโครวินาที	0.1 วินาที	32 วินาที	24 วัน	10^{13} ปี

ตารางที่ 3.1 แสดงระยะเวลาที่ใช้ในการที่ถอดรหัสโดยมีความยาวคีย์แบบต่าง ๆ

บทที่ 4
เอกสารสิทธิ์

4.1 ลักษณะของเอกสารสิทธิ์ดิจิตอล (Digital Certificate)

การรับรองและพิสูจน์ตนในการติดต่อสื่อสารนั้นเป็นสิ่งสำคัญอย่างยิ่ง เมื่อมีผู้ส่งข้อมูลมายังผู้รับ ผู้รับต้องตรวจสอบว่าผู้ส่งคือใครและสามารถเชื่อถือข้อมูลที่ส่งมาได้มากน้อยเพียงใด การรับรองและพิสูจน์ตนในการส่งข้อมูลแบบอิเล็กทรอนิกส์นั้นยิ่งเป็นเรื่องที่ซับซ้อนเพราะว่าเราไม่สามารถรู้ได้เลยว่าใครเป็นผู้ส่งที่แท้จริง เพราะสามารถทำการดักจับข้อมูลมาแก้ไขดัดแปลงเนื้อหาของข้อมูลได้ รวมถึงการแอบอ้างสิทธิ์ของผู้ส่งเอง

รูปที่ 4.1 แสดงตัวอย่างของเอกสารสิทธิ์ที่ได้รับการรับรองจากองค์กรพิสูจน์สิทธิ์

เอกสารสิทธิ์เป็นเหมือนกับบัตรประจำตัวของบุคคลนั้น ซึ่งจะบ่งบอกรายละเอียดของบุคคล เราสามารถส่งเอกสารสิทธิ์ไปพร้อมกับลายมือชื่อดิจิตอลเพื่อใช้ในการอ้างถึงผู้ส่ง การสร้างเอกสารสิทธิ์ทำโดยผู้ใช้ทุกคนทำการขอเอกสารสิทธิ์กับองค์กรพิสูจน์สิทธิ์ (Certificate Authority) โดยการส่งคีย์สาธารณะและข้อมูลตามที่องค์กรพิสูจน์สิทธิ์นั้นกำหนดไปและทำการขอเอกสารสิทธิ์มา การติดต่อต้องทำเป็นการส่วนตัวหรือติดต่อผ่านระบบการพิสูจน์บุคคลที่ปลอดภัย เราสามารถกำหนดสิ่งที่จำเป็นในการสื่อสารดังต่อไปนี้

1. ผู้ใช้ทุกคนสามารถคำนวณหาชื่อและคีย์สาธารณะของเจ้าของเอกสารสิทธิ์ได้
2. ผู้ใช้ทุกคนสามารถตรวจสอบได้ว่าเอกสารสิทธิ์มาจากองค์กรพิสูจน์สิทธิ์จริง ๆ ไม่ได้ถูกปลอมแปลงมา
3. ผู้ใช้สามารถตรวจสอบได้ว่าเอกสารสิทธิ์นั้นหมดอายุหรือไม่
4. ผู้ที่สามารถสร้างและอัพเดตเอกสารสิทธิ์ได้มีเพียงองค์กรผู้มีอำนาจในการรับรองสิทธิ์เท่านั้น

5. ผู้ใช้ทุกคนสามารถตรวจสอบเอกสารสิทธิ์ได้เป็นประจำ

4.2 ความสำคัญของเอกสารสิทธิ์ดิจิตอล

ในระบบการเข้ารหัสแบบคีย์ต่างนั้นเราจะต้องมีการสร้างทั้งคีย์ส่วนตัวและคีย์สาธารณะ ซึ่งโดยทั่วไปการสร้างคีย์จะทำโดยโปรแกรมที่จะใช้คีย์นั้น เช่น โปรแกรมเว็บบราวเซอร์หรือโปรแกรมติดต่ออิเล็กทรอนิกส์เมล์ หลังจากที่สร้างคีย์ทั้งสองเรียบร้อยแล้วการเก็บรักษาคีย์เป็นเรื่องที่สำคัญ เราจะต้องเก็บรักษาคีย์ส่วนตัวไว้ให้ดีอย่าให้ใครมาแอบเห็นหรือขโมยไปได้ จากนั้นจะเป็นการตัดสินใจว่าจะทำการแจกจ่ายคีย์สาธารณะของเราไปสู่ผู้อื่นด้วยวิธีใด เช่น อาจแจกคีย์โดยส่งอิเล็กทรอนิกส์เมล์ไปให้เพื่อนหรือบุคคลที่ติดต่อกับเรา แต่วิธีนี้เราอาจส่งคีย์ไปให้ไม่ครบทุกคน และยังต้องเป็นภาระคอยจัดการส่งคีย์ให้กับบุคคลใหม่ ๆ ที่ต้องการติดต่อด้วย นอกจากนี้ยังไม่สามารถทำให้ผู้รับมั่นใจได้ว่าคีย์ที่ส่งไปให้นั้นเป็นคีย์ของเราจริง ๆ เนื่องจากอาจมีผู้อื่นแอบสร้างคีย์โดยใช้ชื่อเราและแอบอ้างส่งคีย์ดังกล่าวให้กับผู้อื่นเพื่อให้เข้าใจว่าเป็นคีย์ของเราก็ได้

วิธีที่ดีกว่าและใช้อยู่ในปัจจุบันก็คือการใช้ระบบแจกจ่ายคีย์ที่เชื่อถือได้โดยจะมีองค์กรที่ทำหน้าที่เฉพาะเป็นองค์กรที่สาม (Third Party) ในการรับรองและระงับการรับรองคีย์ที่เรียกว่า องค์กรพิสูจน์สิทธิ์ (Certificate Authority - CA) โดยองค์กรพิสูจน์สิทธิ์นี้จะตรวจสอบคีย์สาธารณะของเราด้วยหลักฐานว่าคีย์นั้นเป็นของเราจริง ๆ พร้อมทั้งตรวจสอบข้อมูลส่วนตัวของเรา (ข้อมูลที่ตรวจสอบจะมากน้อยแค่ไหนขึ้นอยู่กับระดับชั้นของการรับรอง) เมื่อผู้อื่นได้รับคีย์ของเราก็สามารถที่จะตรวจสอบกับผู้ออกเอกสารสิทธิ์นี้ว่าคีย์ที่ได้รับเป็นของเราจริงหรือไม่ ซึ่งตัวเอกสารสิทธิ์นี้จะเปรียบเสมือนบัตรประชาชนดิจิตอลของเราที่ใช้บอกได้ว่าเราเป็นบุคคลที่อ้างจริง ๆ ในระบบเครือข่ายหรือการส่งข้อมูลทางอิเล็กทรอนิกส์

ในปัจจุบันบริษัทหลัก ๆ ที่ออกเอกสารสิทธิ์ดิจิตอล (Digital Certificate) คือ บริษัท Verisign, Cyvertrust, Global Sign และ Nortel โดยในเอกสารสิทธิ์ดิจิตอลจะประกอบด้วยข้อมูลต่าง ๆ ดังนี้ ชื่อของผู้ถือเอกสารสิทธิ์ ชื่อของบริษัทที่ออกเอกสารสิทธิ์ คีย์สาธารณะของผู้ถือเอกสารสิทธิ์ วันหมดอายุของเอกสารสิทธิ์ (โดยทั่วไปจะกำหนดระยะเวลา 6 เดือนหรือหนึ่งปี) ระดับชั้นของเอกสารสิทธิ์ และเลขหมายของตัวเอกสารสิทธิ์ดิจิตอลนั้นเอง

เอกสารสิทธิ์ดิจิตอลแบ่งออกได้เป็นสี่ระดับชั้นตามระดับการตรวจสอบข้อมูลของเจ้าของเอกสารสิทธิ์

1. ระดับชั้นที่หนึ่งเป็นชั้นที่ออกเอกสารสิทธิ์ได้ง่ายที่สุดเนื่องจากมีการตรวจสอบน้อยที่สุด โดยจะตรวจสอบแค่ชื่อผู้ถือเอกสารสิทธิ์ และที่อยู่อิเล็กทรอนิกส์เมล์ (e-mail address) ว่าถูกต้องจริงเท่านั้น

2. ระดับชั้นที่สองจะตรวจสอบเลขประจำตัวประชาชน เลขประจำตัวของระบบสวัสดิการหรือประกันสังคม (Social Security Number) และวันเดือนปีเกิด

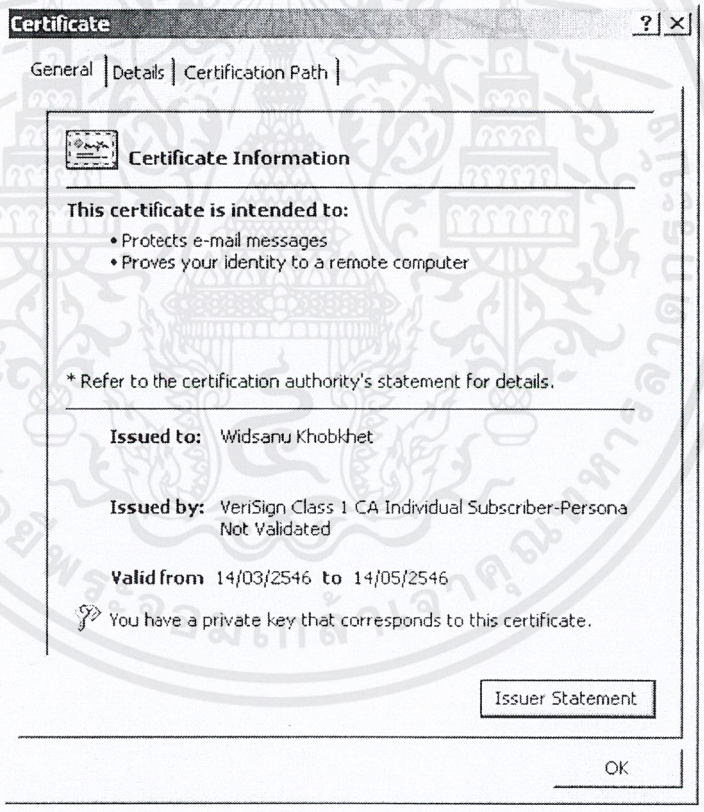
3. ระดับชั้นที่สามจะมีการตรวจสอบเพิ่มเติมเกี่ยวกับประวัติการใช้เครดิตและการชำระเงิน

4. ระดับชั้นที่สี่นั้นยังไม่มีการออกมาเป็นมาตรฐานอย่างแน่ชัด แต่จะเป็นการตรวจสอบข้อมูลเพิ่มเติมเกี่ยวกับตำแหน่งงานในองค์กรด้วย

ในการขอเอกสารสิทธิ์นั้นจะต้องกระทำการบนกระบวนการที่ปลอดภัย ซึ่งในปัจจุบันมี 2 วิธีคือ

1. การของเอกสารสิทธิ์ผ่านโปรแกรมประเภทบราวเซอร์ เช่น อินเตอร์เน็ตเอ็กซ์พลอเรอร์, เนสเคปท์ เป็นต้น โดยผู้ใช้เข้าไปยังโฮสต์ที่ให้บริการการขอเอกสารสิทธิ์หรือเว็ปไซต์ที่เปิดให้บริการดังกล่าวอยู่ ตัวโปรแกรมประเภทบราวเซอร์จะทำการสร้างคู่คีย์ คีย์ส่วนตัวจะเก็บไว้ที่เครื่องของเรา จากนั้นจะส่งคีย์สาธารณะและข้อมูลส่วนตัวของเราไปยังโฮสต์หรือไซต์ที่เป็นขององค์กรพิสูจน์ จากนั้นองค์กรพิสูจน์สิทธิ์จะทำการรับรองข้อมูลแล้วจึงส่งเอกสารสิทธิ์กลับมาที่เครื่องของเรา

2. การขอเอกสารสิทธิ์อีกรูปแบบหนึ่งคือการส่งเอกสารสิทธิ์ที่ไม่ผ่านระบบเครือข่าย กล่าวคือผู้ขอต้องไปทำการขอที่สำนักงานขององค์กรพิสูจน์สิทธิ์โดยตรง โดยทำเรื่องขอเอกสารสิทธิ์และกรอกข้อมูลส่วนตัว องค์กรพิสูจน์สิทธิ์จะทำการรับรองและบันทึกข้อมูลของผู้ขอแล้วทำการส่งเอกสารสิทธิ์พร้อมทั้งคีย์ส่วนตัวให้กับผู้ขอในรูปของไฟล์ในแผ่นดิสก์ให้ผู้ขอนำไปบันทึกในเครื่องของตนเองต่อไป



รูปที่ 4.2 แสดงตัวอย่างเอกสารสิทธิ์ที่ได้รับจากองค์พิสูจน์สิทธิ์โดยใช้โปรแกรมประเภทบราวเซอร์

4.3 บริการพิสูจน์สิทธิ์แบบ X.509 (X.509 Authentication Service)

ระบบ X.509 เป็นระบบพิสูจน์สิทธิ์ที่สำคัญมากในระบบเครือข่าย โดย X.509 เป็นอนุกรมย่อยของ X.500 ซึ่งกำหนดมาตรฐาน ITU – T โดยขณะที่ X.500 เป็นตัวกำหนดโครงสร้างในลักษณะที่เป็นได

เรกทอรีหรือโครงสร้างนั้น X.509 จะทำหน้าที่ในการพิสูจน์สิทธิ์ให้กับส่วนต่าง ๆ ของไดเรกทอรีนั้น สำหรับรูปแบบการใช้งานจะเน้นไปที่การพิสูจน์บุคคล เพื่อยืนยันการติดต่อเป็นสำคัญ

การทำงานของ X.509 จะมีโครงสร้างการทำงานที่เป็นไดเรกทอรีโดยในที่นี้ไดเรกทอรีจะทำหน้าที่เป็นที่เก็บข้อมูลที่ใช้ในการยืนยัน ซึ่งโดยทั่วไปจะอยู่ในรูปของเอกสารสิทธิ์ซึ่งในเอกสารสิทธิ์จะบรรจุคีย์สาธารณะของผู้ใช้ที่เข้ารหัสโดยคีย์ส่วนตัวขององค์กรที่จ่ายใบเอกสารสิทธิ์มาให้ สำหรับการทำงานของ X.509 นั้นจะมีขอบเขตการนำไปใช้งานที่กว้างขวางมาก เช่น ใช้ในการทำ Mail Security ใช้ในการทำ IP Security ใช้ในการทำ Web Security หรือหากจะกล่าวว่าเมื่อใดที่ต้องการพิสูจน์บุคคลหรือยืนยันเครื่องคอมพิวเตอร์แล้ว ก็มักจะอยู่ในขอบข่ายการทำงานของ X.509 เสมอ

X.509 ได้ถูกนำเสนอเมื่อปี 1988 จากนั้นได้ผ่านการปรับปรุงเป็นลำดับขั้น ในประเด็นต่าง ๆ รวมทั้งเรื่องความปลอดภัยด้วย จากนั้นก็ได้ออกมาเป็นข้อเสนอที่ปรับปรุงแล้วในปี 1993 และปรับปรุงอีกครั้งในปี 1995 โดยการทำงานของ X.509 จะใช้การเข้ารหัสแบบคีย์สาธารณะและใช้มาตรฐานลายมือชื่อดิจิตอลในการรับรองข้อมูล สำหรับอัลกอริทึมนั้นไม่ได้ระบุแน่นอนโดยสามารถเลือกใช้ได้หลายตัวแต่ที่แนะนำคืออาร์เอสเอ
รูปแบบทั่วไปของเอกสารสิทธิ์มีส่วนประกอบดังนี้

1. เวอร์ชัน (Version) แสดงหมายเลขเวอร์ชันเพราะในแต่ละเวอร์ชันจะมีรูปแบบของข้อมูลที่ไม่เหมือนกันก็ได้ โดยปกติจะเป็นเวอร์ชัน 1 แต่หากในเอกสารสิทธิ์มีการใช้

2. หมายเลขลำดับ (Serial Number) เป็นเลขจำนวนเต็ม โดยจะต้องไม่ซ้ำกันในองค์กรที่ออกเอกสารสิทธิ์ โดยเลขนี้จะเป็นเลขที่จะใช้อ้างถึงแต่ละเอกสารสิทธิ์ที่สร้างขึ้นมา

3. อัลกอริทึมที่ใช้สร้าง (Signature Algorithm Identifier) เป็นฟิลด์ที่ระบุอัลกอริทึมที่ใช้ในการสร้างเอกสารสิทธิ์

4. ชื่อผู้ออกเอกสารสิทธิ์ (Issue Name) เป็นชื่อขององค์กรที่ออกเอกสารสิทธิ์

5. ช่วงเวลาที่รับรองเอกสารสิทธิ์ (Period of Validity) เป็นตัวบอกว่าให้ใช้เอกสารสิทธิ์นี้ตั้งแต่วันที่เท่าไรและสิ้นสุดวันที่เท่าไร

6. ชื่อเจ้าของเอกสารสิทธิ์ (Subject Name) เป็นชื่อของบุคคลที่เอกสารสิทธิ์ใบนี้อ้างถึงหรือแทนตัวบุคคลนั้น

7. ข้อมูลของคีย์สาธารณะ (Subject's Public Key Information) เป็นฟิลด์ที่เก็บคีย์สาธารณะและระบุถึงอัลกอริทึมที่ใช้กับคีย์นี้ขึ้นมา รวมถึงพารามิเตอร์อื่น ๆ ด้วย

8. ตัวระบุผู้ออกเอกสารสิทธิ์ (Issuer Unique Identifier) เป็นฟิลด์ออปชันที่ใช้ในการระบุถึงองค์กรที่ออกเอกสารสิทธิ์ ในกรณีที่ชื่อ X.500 มีการนำไปใช้กับส่วนอื่น ๆ

9. ตัวระบุผู้ขอเอกสารสิทธิ์ (Subject Unique Identifier) เป็นฟิลด์ที่ใช้ในการระบุถึงตัวบุคคลที่เป็นเจ้าของเอกสารสิทธิ์ ในกรณีที่ชื่อ X.500 มีการนำไปใช้กับส่วนอื่น ๆ

10. ส่วนขยาย (Extension) เป็นกลุ่มของฟิลด์ที่เพิ่มเติมข้อมูลอื่น ๆ เข้ามาด้วย

11. ลายมือชื่อ (Signature) จะบรรจุเมสเซจไดเจสต์ของข้อมูลในทุกฟิลด์ที่เข้ารหัสด้วยคีย์ส่วนตัวขององค์กรพิสูจน์สิทธิ์เพื่อเป็นการยืนยันว่าเอกสารสิทธิ์นี้สร้างมาจากองค์กรดังกล่าว จริง ๆ โดยจะมีข้อมูลที่ระบุวิธีการแฮชและวิธีการเข้ารหัสด้วย

ในการใช้งานเอกสารสิทธิ์จะมีช่วงเวลาใช้งานที่จำกัดแน่นอน ดังนั้นหากผู้ใช้ต้องการใช้เอกสารสิทธิ์ต่อไปก็ต้องขอต่ออายุเอกสารสิทธิ์ก่อนที่จะหมดอายุ แต่หากมีการหมดอายุโดยที่ไม่ขอต่อหรือมีการเลิกใช้เอกสารสิทธิ์อาจจะเนื่องมาจากพนักงานลาออก หรืออาจจะเนื่องจากเอกสารสิทธิ์นี้ไม่ปลอดภัยแล้วก็ต้องทำการเรียกคืน (Revoke) และองค์กรพิสูจน์สิทธิ์จะต้องมีการจัดทำรายการเอกสารสิทธิ์ที่ถูกเรียกคืน (Certificate Revocation List – CRL) ซึ่งจะเก็บไว้ในไดเรกทอรีและรับรองโดยองค์กรพิสูจน์สิทธิ์ ซึ่งผู้ที่ต้องการตรวจสอบเอกสารสิทธิ์ว่าเป็นเอกสารสิทธิ์ที่ไม่ใช้งานแล้วหรือไม่ ก็ต้องขอรายการเอกสารสิทธิ์ที่ถูกเรียกคืนไปตรวจสอบ

และเนื่องจากเอกสารสิทธิ์ไม่สามารถปลอมได้ ดังนั้นการเก็บเอกสารสิทธิ์ไว้ที่องค์กรพิสูจน์สิทธิ์จึงไม่ต้องมีกลไกพิเศษมาป้องกันแต่อย่างใด กล่าวคือผู้ใช้คนใดที่เป็นสมาชิกขององค์กรพิสูจน์สิทธิ์ก็สามารถเข้าถึงเอกสารสิทธิ์ของผู้ใช้คนอื่น ๆ ได้ทุกคน โดยเอกสารสิทธิ์นี้จะเก็บอยู่ในไฟล์เพียงไฟล์เดียวที่มีขนาดเล็ก นอกจากจะสามารถขอเอกสารสิทธิ์จากองค์กรพิสูจน์สิทธิ์แล้วผู้ใช้ยังสามารถส่งเอกสารสิทธิ์ไปให้กันเองได้อีกด้วยโดยผ่านทางสื่อต่าง ๆ เช่น จดหมายอิเล็กทรอนิกส์, ส่งผ่านแผ่นดิสก์เก็ตเป็นต้น

อย่างไรก็ตามเนื่องจากระบบเครือข่ายในปัจจุบันมีขนาดใหญ่โตกว้างขวางมากและการติดต่อสื่อสารก็ไม่ได้มีลักษณะเฉพาะกลุ่มอีกแล้ว ดังนั้นการที่จะให้ผู้ใช้ทุกคนมาใช้เอกสารสิทธิ์ที่รับรองโดยองค์กรพิสูจน์สิทธิ์เดียวกันทั้งหมดก็อาจเป็นเรื่องยาก หากผู้ใช้ 2 คนใช้เอกสารสิทธิ์ที่รับรองจากองค์กรพิสูจน์สิทธิ์คนละแห่งก็จะไม่สามารถตรวจสอบเอกสารสิทธิ์ของอีกฝ่ายได้ว่าเป็นฉบับจริงหรือไม่ ซึ่งในกรณีเช่นนี้ก็อาจจะใช้วิธีสำเนาคีย์สาธารณะขององค์กรพิสูจน์สิทธิ์ของผู้ใช้อีกคนหนึ่งมาทำการตรวจสอบเองก็สามารถทำได้ เช่น กำหนดให้มี CA – A และ CA – B โดยให้บริการกับผู้ใช้ A และ B แต่เนื่องจากในครั้งแรกที่ A สำเนาคีย์สาธารณะของ CA – B มานั้นอาจเกิดการปลอมได้ เพราะสิ่งที่เรามีอยู่ก็คือคีย์สาธารณะของ CA – A ของเรา แต่เอกสารสิทธิ์ของ CA – B ซึ่งบรรจุคีย์สาธารณะของ CA – B นั้นจะรับรองการเข้ารหัสด้วยคีย์ส่วนตัวของ CA – B ทำให้เราไม่สามารถตรวจสอบว่าเอกสารสิทธิ์ที่ได้รับมานั้นเป็นฉบับที่ถูกต้องหรือไม่ ดังนั้นวิธีดังกล่าวจึงถือว่ามีความปลอดภัยไม่เพียงพอ

สำหรับอีกวิธีการอีกแบบ คือให้ CA – A เก็บเอกสารสิทธิ์ของ CA – B เอาไว้ด้วยและ CA – B ก็เก็บเอกสารสิทธิ์ของ CA – A เอาไว้เช่นกัน ด้วยวิธีนี้เราก็สามารถให้องค์กรพิสูจน์สิทธิ์ตรวจสอบเอกสารสิทธิ์ได้ไม่ว่าเอกสารสิทธิ์นั้นจะรับรองจาก CA – A หรือ CA – B ก็ตาม เช่น ผู้ใช้ A ต้องการตรวจสอบเอกสารสิทธิ์ที่รับรองจาก CA – A ก็สามารถทำได้เลยเพราะรู้คีย์สาธารณะของ CA – A อยู่แล้วเนื่องจากเป็นสมาชิกของ CA – A และหากผู้ใช้ A ต้องการตรวจสอบเอกสารสิทธิ์ที่รับรองโดย CA – B ผู้ใช้ A ก็ขอเอกสารสิทธิ์ของ CA – B จาก CA – A โดยเอกสารสิทธิ์ดังกล่าวจะรับรองโดย CA – A ดังนั้นจึงแน่ใจได้ว่าเอกสารสิทธิ์ของ CA – B ที่ได้รับนั้นเป็นของจริงและคีย์สาธารณะของ CA – B ก็เป็นของจริง

จากนั้นจึงนำเอาคีย์สาธารณะของ CA – B ไปตรวจสอบเอกสารสิทธิ์อีกทีก็จะทราบได้ว่าเอกสารสิทธิ์นั้นเป็นของจริงหรือไม่

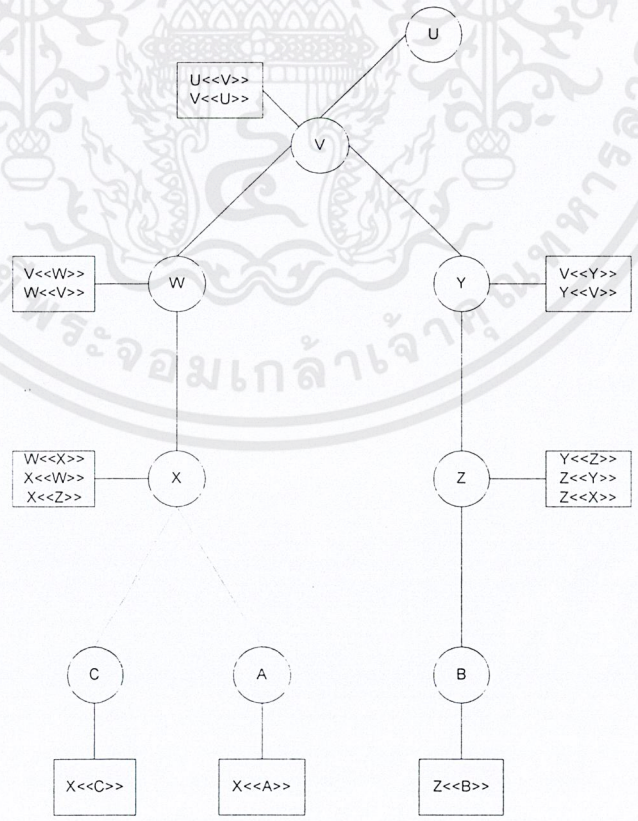
และนอกเหนือจาก CA – A และ CA – B แล้วการเชื่อถือ (Trust) กันเช่นนี้ ยังสามารถกระทำกับองค์กรพิสูจน์สิทธิ์อื่น ๆ ไปเรื่อย ๆ อย่างไรก็ตาม หากองค์พิสูจน์สิทธิ์มีจำนวนมาก ๆ แล้ว ก็มีความจำเป็นที่จะต้องจัดโครงสร้างการเชื่อถือกันขององค์กรพิสูจน์สิทธิ์ให้เป็นระบบ ไม่เช่นนั้นก็อาจมีการเชื่อถือกันแบบยุ่งเหยิงและทำให้การทำงานเป็นไปอย่างไม่มีประสิทธิภาพได้ ซึ่งมาตรฐาน X.509 ก็ได้แนะนำให้มีการใช้ระบบเชื่อถือกันในรูปของความสัมพันธ์แบบระดับชั้น (Hierarchical) ซึ่งรูปที่ 4.3 จะแสดงความสัมพันธ์แบบระดับชั้นขององค์กรพิสูจน์สิทธิ์ต่าง ๆ โดยส่วนที่เป็นวงกลมจะหมายถึงความสัมพันธ์ระหว่างองค์กรพิสูจน์สิทธิ์ และในกล่องจะหมายถึงเอกสารสิทธิ์ที่เก็บไว้ในไดเรกทอรีของแต่ละองค์กรพิสูจน์สิทธิ์แบ่งออกเป็น 2 แบบ คือ

1. Forward Certificate หมายถึง เอกสารสิทธิ์ของ X ที่สร้างโดย CA อื่น ๆ

2. Reverse Certificate หมายถึง เอกสารสิทธิ์ที่สร้างโดย X ซึ่งก็จะกลายเป็นเอกสารสิทธิ์ขององค์กรพิสูจน์สิทธิ์อื่น ๆ ด้วย

จากตัวอย่างผู้ใช้ A ได้ขอเอกสารสิทธิ์จาก CA – B จะได้เส้นทางของเอกสารสิทธิ์ดังนี้

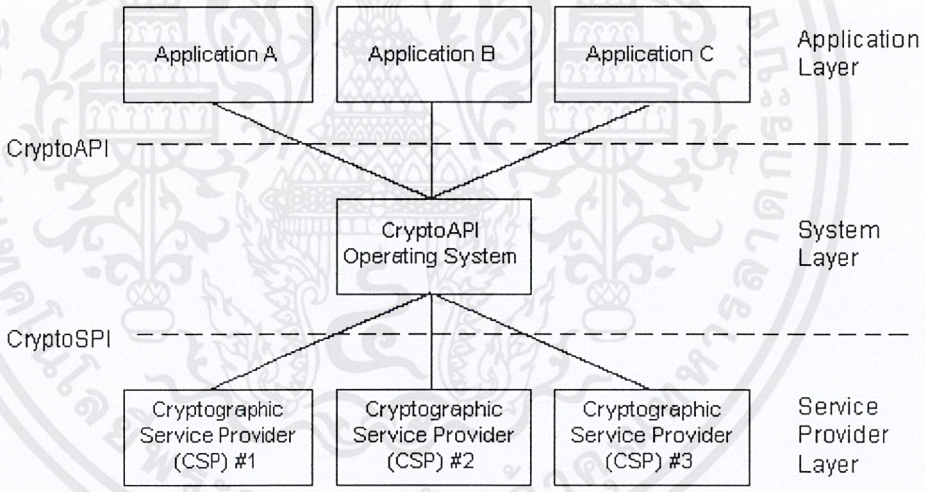
X << W >> W << V >> V << Y >> Y << Z >> Z << B >>



รูปที่ 4.3 จะแสดงความสัมพันธ์แบบระดับชั้นขององค์กรพิสูจน์สิทธิ์ต่าง ๆ ตามรูปแบบ X.509

บทที่ 5
คริปโตเอพีไอ
(Cryptographic Application Programming Interface)

ในการการพัฒนาตัวโปรแกรมคณะผู้จัดทำได้ใช้เอพีไอ (API – Application Programming Interface) ของไมโครซอฟท์วินโดวส์ที่ชื่อว่าคริปโตเอพีไอ ซึ่งเป็นเอพีไอตัวหนึ่งที่มีความสามารถในการทำงานในด้าน การเข้ารหัส ถอดรหัสและสามารถสร้างลายมือชื่อดิจิตอลได้ โดยเอพีไอตัวนี้จะทำการติดต่อกับส่วนที่เรียกว่า ซีเอสพี (CSP – Cryptographic Service Provider), ระบบปฏิบัติการร่วมถึงที่เก็บเอกสารสิทธิ์ (Certificate Store) ภายในเครื่องคอมพิวเตอร์ โครงสร้างความสัมพันธ์ระหว่างคริปโตเอพีไอและส่วนต่าง ๆ นั้นมีลักษณะดังรูป



รูปที่ 5.1 แสดงโครงสร้างความสัมพันธ์ของคริปโตเอพีไอและซีเอสพี

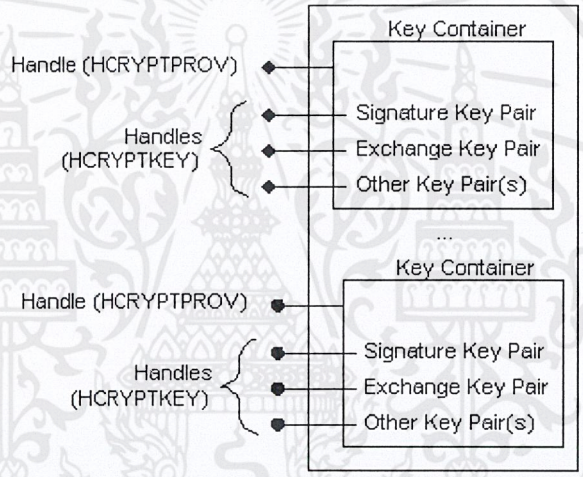
ภายในซีเอสพีจะมีวัตถุต่าง ๆ ที่ใช้ในการทำงาน วัตถุที่อยู่ภายในซีเอสพี ได้แก่ คีย์คอนเทนเนอร์ (Key Container), วัตถุแฮช (Hash Object), วัตถุเซสชันคีย์ (Session Key Object) และ วัตถุคีย์ส่วนตัว – คีย์สาธารณะ (Private – Public Key Object) ตัวโปรแกรมจะทำการติดต่อกับวัตถุต่าง ๆ ภายในซีเอสพีโดยผ่านแฮนเดิล (Handle) ที่ผูกกับวัตถุที่ต้องการติดต่อด้วย

5.1 โครงสร้างของซีเอสพี

5.1.1 คีย์คอนเทนเนอร์ (Key Container)

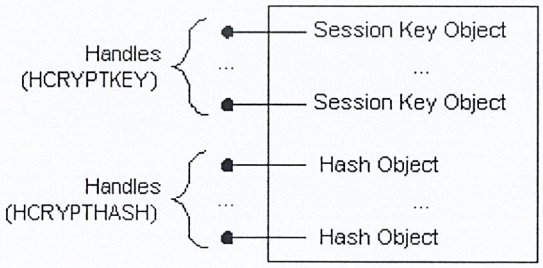
แต่ละซีเอสพีจะเก็บคู่คีย์ต่าง ๆ ภายในภายในคีย์คอนเทนเนอร์ ซึ่งเปรียบเสมือนฐานข้อมูลที่ใช้เก็บคู่คีย์ต่าง ๆ โดยคีย์คอนเทนเนอร์จะมีชื่อเรียกเฉพาะเพื่อใช้ในการระบุและจัดการกับคู่คีย์ภายในคีย์คอนเทนเนอร์นั้น ๆ ได้ แต่ละซีเอสพีนั้นจะเป็นผู้จัดการเองว่าให้คีย์คอนเทนเนอร์นั้นไปเก็บไว้ที่ใด อาจเก็บไว้ในฮาร์ดแวร์เฉพาะ (Tamper - resistant hardware), เก็บไว้ในรีจีสทรี (Registry) หรือเก็บไว้ในไฟล์ซิสเต็ม เป็นต้น การเก็บวัตถุต่าง ๆ ภายในซีเอสพี ออกเป็น 2 ประเภทดังนี้

1. เก็บไว้ในหน่วยความจำถาวร (Persistant Data Objects) ซีเอสพีจะเก็บคีย์ส่วนตัวและคีย์สาธารณะไว้ในหน่วยความจำถาวรในรูปแบบที่ทำการเข้ารหัสไว้



รูปที่ 5.2 แสดงโครงสร้างที่เก็บข้อมูลแบบถาวรของคีย์คอนเทนเนอร์

2. เก็บไว้ในหน่วยความจำชั่วคราว ซีเอสพีจะทำการเก็บวัตถุเซสชันคีย์ และ วัตถุแฮชในหน่วยความจำชั่วคราวเมื่อมีการเรียกในงานวัตถุนั้น ๆ ซึ่งวัตถุต่าง ๆ เหล่านี้จะถูกทำลายเมื่อมีการคืนหน่วยความจำที่เก็บแฮนเดิลของคีย์คอนเทนเนอร์และซีเอสพี



รูปที่ 5.3 แสดงโครงสร้างที่เก็บข้อมูลแบบชั่วคราวของคีย์คอนเทนเนอร์

5.1.2 วัตถุคีย์เซสชัน (Session Key Object)

วัตถุคีย์ที่ใช้ในการเข้ารหัส/ถอดรหัสแบบสมมาตร โดยทั่วไปคีย์นี้จะมีขนาด 40 ถึง 2,000 บิต

5.1.3 วัตถุคีย์ส่วนตัวและคีย์สาธารณะ (Private and Public Key Object)

วัตถุคีย์ที่ใช้ในการเข้ารหัส/ถอดรหัสแบบไม่สมมาตร โดยทั่วไปซีเอสพีจะแบ่งคีย์ประเภทนี้ออกเป็น 2 ประเภทคือ คีย์ลายมือชื่อ (Signature Key Pair) และ คีย์แลกเปลี่ยน (Exchange Key Pair)

5.1.4 คีย์ลายมือชื่อ (Signature Key Pair)

เป็นคู่คีย์ที่มีไว้ในการพิสูจน์ตน จะถูกใช้ในการลงนามลายมือชื่อดิจิตอล การเข้ารหัสแบบไม่สมมาตร

5.1.5 คีย์แลกเปลี่ยน (Exchange Key Pair)

เป็นคู่คีย์ที่มีไว้สำหรับการทำการเข้ารหัส/ถอดรหัสเซสชันคีย์ มีไว้เพื่อประโยชน์ในการกระจายคีย์

5.1.6 วัตถุแฮช (Hash Object)

วัตถุที่เก็บเมสเซจไดเจสต์ของข้อมูลที่ผ่านแฮชฟังก์ชัน

ในการพัฒนาโปรแกรมให้มีความสามารถอย่างเต็มที่การเขียนโปรแกรมให้สามารถที่จะเรียกใช้คริปโตเอพีไอให้ถูกต้องได้นั้นจำเป็นที่จะต้องรู้ถึงคำสั่งต่าง ๆ ในการเรียกเอพีไอให้เป็นลำดับที่ถูกต้อง คำสั่งต่าง ๆ จะถูกกำหนดไว้ในไฟล์ที่ชื่อ Wincrypt.h ซึ่งผู้ที่พัฒนาโปรแกรมจำเป็นต้องรู้ถึงคำสั่งพื้นฐานในการใช้งานของเอพีไอ แต่ก่อนอื่นคณะผู้จัดทำจึงขอทำการอธิบายคำศัพท์ต่าง ๆ ที่สำคัญในการเรียกคริปโตเอพีไอก่อนดังนี้

5.2 ชนิดของซีเอสพี

เป็นชนิดของซีเอสพีนั้นมีได้หลายชนิดและแต่ละชนิดจะมีรูปแบบของข้อมูล (Data Format) ที่ไม่เหมือนกันหรืออาจเป็นฟังก์ชันการทำงานใช้อัลกอริทึมในการทำงานไม่เหมือนกัน ในชนิดของโพรไวเดอร์หนึ่ง ๆ นั้น จะมีซีเอสพีหลายตัวที่เป็นแบบเดียวกัน เราสามารถแบ่งชนิดของโพรไวเดอร์ได้ดังนี้

1. PROV_DSS Provider Type เป็นชนิดของโพรไวเดอร์ที่สามารถสร้างลายมือชื่อดิจิตอลและทำการหาค่าแฮชได้ ซึ่งภายในจะประกอบไปด้วย อัลกอลิทึมการลงลายมือชื่อ การทำแฮชชิ่งแบบ MD5 และการทำแฮชชิ่งแบบ SHA – 1

2. PROV_DSS_DH Provider Type เป็นชนิดของโพรไวเดอร์ที่มีความสามารถในการทำคีย์แลกเปลี่ยนคีย์ลายมือชื่อและการทำแฮชชิ่งซึ่งจะคล้ายกับ PROV_DSS

3. PROV_FROTEZZA Provider Type เป็นโพรไวเดอร์ที่มีความสามารถในการทำคีย์แลกเปลี่ยนคีย์ลายมือชื่อ การเข้ารหัสและการทำแฮชซึ่งซึ่งในชนิดนี้อัลกอริทึมต่าง ๆ จะถูกกำหนดโดยสถาบัน National Institute of Standards and Technology (NIST)

4. PROV_MS_EXCHANGE Provider Type จะใช้สำหรับโปรแกรมไมโครซอฟท์เอ็กซ์เชนจ์ (Microsoft Exchange) หรือว่าโปรแกรม (Application) ที่ทำหน้าที่คล้ายไมโครซอฟท์เมล์ (Microsoft Mail) ซึ่งชนิดโพรไวเดอร์ชนิดนี้จะสามารถทำการแลกเปลี่ยนคีย์ได้ ทำการลงนามลายมือชื่อดิจิตอลได้ เข้ารหัสข้อมูล และทำการแฮชซิ่ง

5. PROV_RSA_FULL Provider Type เป็นโพรไวเดอร์ที่ทั้งไมโครซอฟท์ (Microsoft) และอาร์เอสเอดาต้าซีเคียวริตี (RSA Data Security) ช่วยกันจัดทำขึ้นมาถือเป็นชนิดโพรไวเดอร์ที่ทำหน้าที่ได้หลากหลาย เช่น การทำแลกเปลี่ยนคีย์ การลงนามลายมือชื่อดิจิตอล การเข้ารหัสข้อมูล และการทำแฮชซิ่ง โดยการทำการต่าง ๆ ที่เกี่ยวข้องกับการเข้ารหัส/ถอดรหัสนั้นจะใช้ฟังก์ชันการทำงานของอาร์เอสเอเป็นหลัก

6. PROV_RSA_SIG Provider Type เป็นชนิดโพรไวเดอร์อีกชนิดหนึ่งที่ถูกจัดทำโดยไมโครซอฟท์และอาร์เอสเอดาต้าซีเคียวริตี ซึ่งชนิดโพรไวเดอร์ชนิดนี้เป็นส่วนที่แตกออกมาจาก PROV_RSA_FULL แต่ว่าในชนิดนี้จะสามารถทำได้แค่การลงนามลายมือชื่อดิจิตอลและการทำแฮชซิ่งเท่านั้น

7. PROV_SSL Provider Type เป็นชนิดโพรไวเดอร์ที่มีความสามารถในการทำตามมาตรฐานเอสเอสแอล (SSL – Secure Sockets Layer) ซึ่งชนิดของโพรไวเดอร์ชนิดนี้สามารถที่จะทำคีย์แลกเปลี่ยน คีย์ลายมือชื่อ คีย์การเข้ารหัสข้อมูลและการทำแฮชซิ่ง

5.3 ชื่อของซีเอสพี

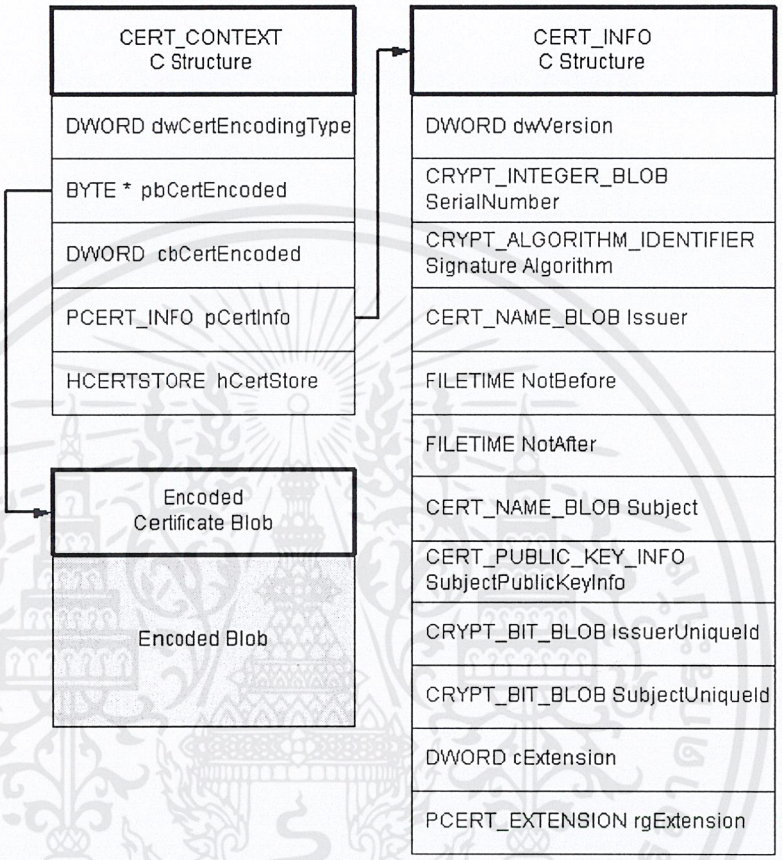
เป็นชื่อที่ใช้บ่งบอกว่าเป็นซีเอสพีตัวใด มีตัวอย่างรายชื่อของไมโครซอฟท์ดังต่อไปนี้

Defined Name	Value
MS_DEF_PROV	"Microsoft Base Cryptographic Provider v1.0"
MS_ENHANCED_PROV	"Microsoft Enhanced Cryptographic Provider "
MS_DEF_RSA_SIG_PROV	"Microsoft RSA Signature Cryptographic Provider"
MS_DEF_RSA_SCHANNEL_PROV	"Microsoft RSA Schannel Cryptographic Provider"
MS_DEF_DSS_PROV	"Microsoft Base DSS Cryptographic Provider"
MS_DEF_DSS_DH_PROV	"Microsoft Base DSS and Diffie-Hellman Cryptographic Provider"
MS_ENH_DSS_DH_PROV	"Microsoft Enhanced DSS and Diffie-Hellman Cryptographic Provider"
MS_DEF_DH_SCHANNEL_PROV	"Microsoft DH Schannel Cryptographic Provider"

ตารางที่ 5.1 ตารางแสดงชื่อของซีเอสพี

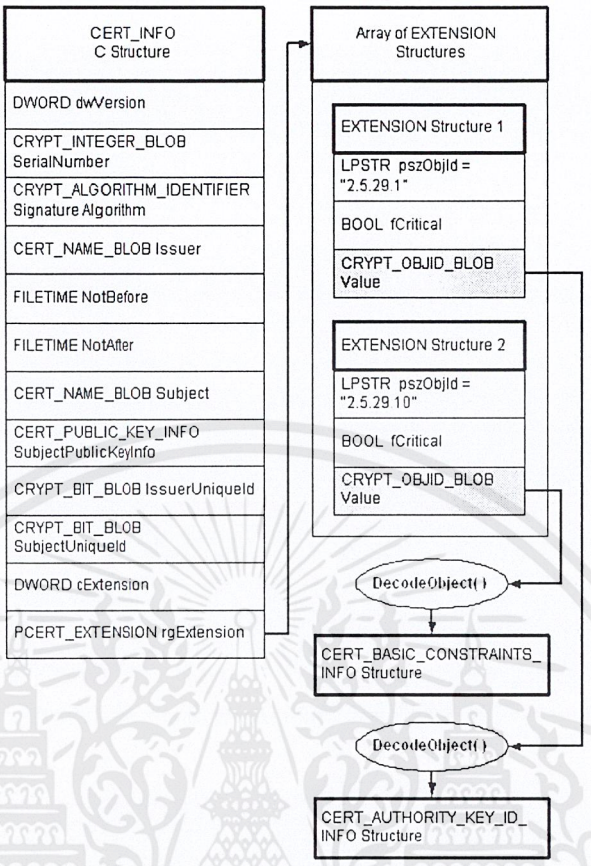
5.4 โครงสร้าง CERT_CONTEXT และ CERT_INFO

CERT_CONTEXT โครงสร้างโดยทั่วไปของ CERT_CONTEXT คือ เป็นที่เก็บเอกสารสิทธิ์ (CERT_INFO ซึ่งจะกล่าวถึงต่อไป) ความยาวของเอกสารสิทธิ์, รูปแบบการเข้ารหัสข้อมูลของเอกสารสิทธิ์



รูปที่ 5.4 โครงสร้างของ CERT_CONTEXT

CERT_INFO เป็นส่วนที่เก็บข้อมูลหลักของเอกสารสิทธิ์ ซึ่งจะมีรายละเอียดข้อมูลเกี่ยวกับเอกสารสิทธิ์นี้ เช่น หมายเลขของเอกสารสิทธิ์ (Serial Number), ชื่อเจ้าของเอกสารสิทธิ์ (Subject name), องค์กรที่เป็นผู้ออกเอกสารสิทธิ์ (Issuer name), ระดับของความปลอดภัยของคู่คีย์, วันที่ขอเอกสารสิทธิ์ (Issue Date), วันที่เอกสารสิทธิ์หมดอายุหรือไม่ถูกรับรอง (Revocation Date) เป็นต้น โดยส่วนที่เก็บข้อมูลจะมีทั้งที่ทำการเข้ารหัสและไม่ได้เข้ารหัสของเอกสารสิทธิ์



รูปที่ 5.5 โครงสร้างของ CERT_INFO

การที่เราจะแสดงข้อมูลในฟิลด์ที่เข้ารหัสไว้ให้ผู้ใช้ดูและตรวจสอบค่าต่าง ๆ ต้องทำการถอดรหัสฟิลด์นั้น ๆ ก่อน วิธีการที่คณะผู้จัดทำใช้ในการถอดรหัสนั้นมีสองวิธีคือ

1. ใช้ฟังก์ชัน CryptDecodeObject ในการถอดรหัสฟิลด์ใด ๆ ที่ทำการเข้ารหัสไว้
2. ใช้ฟังก์ชัน CertNameToStr ในการถอดรหัสฟิลด์ที่เก็บชื่อเจ้าของเอกสารสิทธิ์และชื่อขององค์กรที่ออกเอกสารสิทธิ์

5.5 การติดต่อและการใช้คริปโตเอพีในการติดต่อกับเอกสารสิทธิ์

5.5.1 คำศัพท์ที่เกี่ยวข้อง

CRL – Certificate Revocation List คือ รายชื่อของเอกสารสิทธิ์ที่ถูกนำกลับมาใช้ใหม่ เช่น ในกรณีที่ เอกสารสิทธิ์นั้น เจ้าของอาจจะไม่ต้องการใช้อีกต่อไปก็จะสามารถนำกลับไปให้ผู้อื่นใช้ต่อไปได้

CTL – Certificate Trust List คือ รายชื่อของเอกสารสิทธิ์ที่สามารถเชื่อถือได้

คำสั่งต่าง ๆ ที่ใช้ในการติดต่อกับเอกสารสิทธิ์สามารถแบ่งคำสั่งต่าง ๆ ที่สำคัญออกเป็นกลุ่มได้ดังนี้

5.5.2 ฟังก์ชันติดต่อกับที่เก็บเอกสารสิทธิ์ (Certificate Store Functions)

มีฟังก์ชันดังต่อไปนี้

1. CertOpenStore ใช้เพื่อทำการเปิดที่เก็บเอกสารสิทธิ์ (Certificate Store) โดยจะต้องกำหนดชนิดผู้ให้บริการที่เก็บเอกสารสิทธิ์ด้วย

2. CertRegisterSystemStore ใช้เพื่อทำการลงทะเบียน (Register) ที่เก็บเอกสารสิทธิ์โดยเราสามารถระบุได้ว่าจะให้เก็บไว้ตรงส่วนใด

3. CertSetStoreProperty ใช้เพื่อทำการตั้งค่าต่าง ๆ ให้กับที่เก็บเอกสารสิทธิ์

5.5.3 ฟังก์ชันติดต่อกับเอกสารสิทธิ์ (Certificate Functions)

มีฟังก์ชันดังต่อไปนี้

1. CertAddCertificateContextToStore ใช้เพื่อเพิ่มเอกสารสิทธิ์อันใหม่เข้าไปในที่เก็บเอกสารสิทธิ์

2. CertFindCertificateInStore ใช้สำหรับการหาเอกสารสิทธิ์ที่อยู่ในที่เก็บเอกสารสิทธิ์โดยในการหาครั้งแรกนั้น จะได้เอกสารสิทธิ์ตัวแรกออกมา และในครั้งต่อไปจะได้ตัวถัดไปเรื่อย ๆ ออกมา

3. CertDuplicateCertificateContext ใช้เมื่อต้องการที่จะทำสำเนาของเอกสารสิทธิ์ไว้

5.5.4 ฟังก์ชันติดต่อกับที่เก็บรายชื่อของเอกสารสิทธิ์ที่ถูกนำกลับมาใช้ใหม่ (Certificate Revocation List Functions)

มีฟังก์ชันดังต่อไปนี้

1. CertAddCRLContextToStore เป็นการเพิ่ม CRL Context ลงไปในที่เก็บเอกสารสิทธิ์

2. CertAddCRLLinkToStore ใช้เมื่อต้องการเพิ่มลิงค์ (Link) ให้กับ CRL ในที่เก็บเอกสารสิทธิ์อันหนึ่งเพื่อชี้ไปยัง CRL อื่นในที่เก็บเอกสารสิทธิ์อื่น

3. CertDuplicateCRLContext เป็นฟังก์ชันที่ถูกเรียกใช้เมื่อต้องการทำสำเนาของ CRL

4. CertFindCRLInStore ใช้เมื่อต้องการหา CRL ที่มีอยู่ในที่เก็บเอกสารสิทธิ์

5.5.5 ฟังก์ชันติดต่อกับที่เก็บรายชื่อของเอกสารสิทธิ์ที่สามารถเชื่อถือได้ (Certificate Trust List Functions)

มีฟังก์ชันดังต่อไปนี้

1. CertAddCTLContextToStore ใช้เมื่อต้องการเพิ่ม CTL Context ลงในที่เก็บเอกสารสิทธิ์

2. CertDuplicateCTLContext ใช้เมื่อต้องการจะทำสำเนา CTL Context

3. CertFindCTLInStore ใช้เมื่อต้องการหา CTL ที่อยู่ภายในที่เก็บเอกสารสิทธิ์

5.6 ขั้นตอนการสร้างลายมือชื่อดิจิตอล

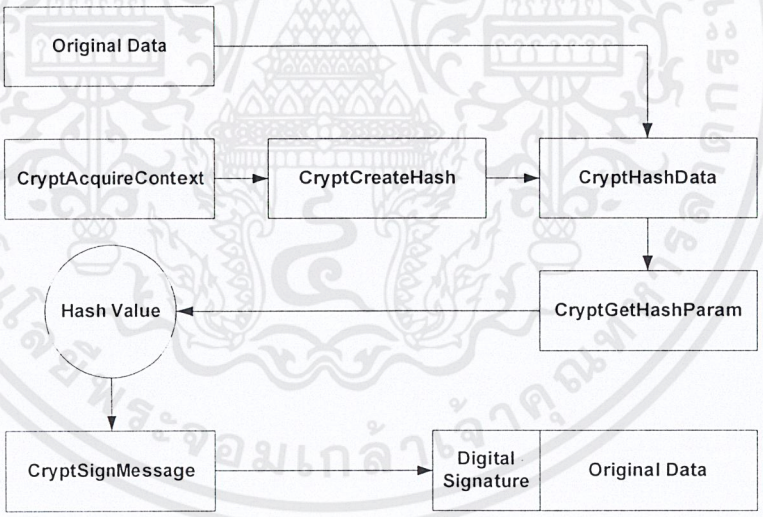
การสร้างลายมือชื่อดิจิตอลมีขั้นตอนดังนี้

1. ขั้นตอนแรกในการติดต่อกับซีเอสพีนั้นจะเริ่มจากการที่โปรแกรม เรียกใช้ฟังก์ชัน CryptAcquireContext เพื่อที่จะใช้จัดการกับซีเอสพีใด ๆ โดยเมื่อเรียกใช้คำสั่งนี้แล้วจะเป็นการบอกถึงชนิดของซีเอสพี ซีเอสพีที่ต้องการติดต่อ และชื่อของคีย์คอนเทนเนอร์ที่เก็บอยู่ภายในซีเอสพีนั้นด้วย

2. เมื่อเราทำการติดต่อกับซีเอสพีและคีย์คอนเทนเนอร์ได้แล้วเราจะทำการสร้างแฮนเดิลของวัตถุแฮชขึ้นมาก่อน โดยใช้ฟังก์ชัน CryptCreateHash

3. จากนั้นทำการแฮชข้อมูลแล้วเก็บค่าเมสเซจไดเจสต์ไว้ที่ตำแหน่งที่แฮนเดิลที่ชี้วัตถุแฮชอยู่โดยใช้ฟังก์ชัน CryptHashData

4. ขั้นตอนต่อมาคือการนำค่าเมสเซจไดเจสต์ที่ได้มาทำการเข้ารหัสด้วยคีย์ส่วนตัวที่ได้รับการรับรองโดยองค์กรพิสูจน์สิทธิ์ ด้วยฟังก์ชัน CryptSignMessage



รูปที่ 5.6 บล็อกไดอะแกรมแสดงการใช้คริปโตเอพีไอในการสร้างลายมือชื่อดิจิตอล

5.7 ขั้นตอนการตรวจสอบลายมือชื่อดิจิตอล

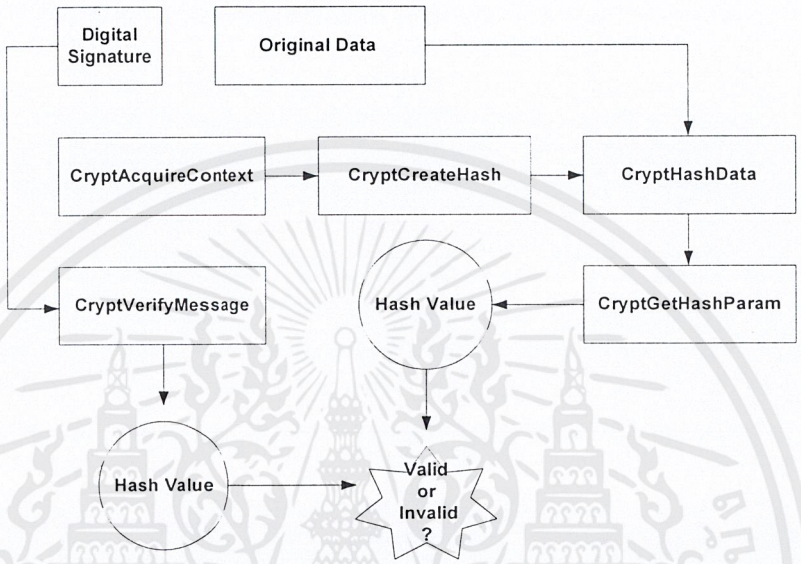
การตรวจสอบลายมือชื่อดิจิตอลมีขั้นตอนดังนี้

1. แยกส่วนที่เป็นข้อมูลต้นฉบับและลายมือชื่อดิจิตอล

2. ในส่วนที่เป็นข้อมูลต้นฉบับ หาค่าเมสเซจไดเจสต์ของข้อมูลต้นฉบับตามลำดับขั้นตอน 1 - 4 เช่นเดียวกับขั้นตอนการสร้างลายมือชื่อดิจิตอล

3. ในส่วนที่เป็นลายมือชื่อดิจิตอลใช้ฟังก์ชัน CryptVerifyMessage ในการถอดรหัสลายมือชื่อดิจิตอล ได้เป็นเมสเซจไดเจสต์อีกชุดออกมา

4. ทำการตรวจสอบว่าเมสเซจที่ได้ขั้นตอนที่ 3 และ 4 เหมือนกันหรือไม่ ถ้าเหมือนกันแสดงว่าเอกสารไม่มีการเปลี่ยนแปลง ถ้าไม่เหมือนกันแสดงว่าเอกสารมีการเปลี่ยนแปลง



รูปที่ 5.7 บล็อกไดอะแกรมแสดงการใช้คริปโตเอพีไอในการตรวจสอบลายมือชื่อดิจิตอล

บทที่ 6

โอแอลอี (OLE – Object Linking and Embedding)

6.1 โอแอลอี (OLE – Object Linking and Embedding)

โอแอลอี เป็นลักษณะการเขียนโปรแกรมอย่างหนึ่ง โดยใช้รูปแบบของคอมออปเจ็กต์ (COM Object) โดยจะแบ่งการทำงานออกเป็น 2 ส่วนสำคัญคือโอแอลอีเซิร์ฟเวอร์ (OLE Server) และส่วนที่เป็นโอแอลอีไคลเอ็นต์ (OLE Client) โดยส่วนเซิร์ฟเวอร์ทำหน้าที่ให้บริการแก่ส่วนไคลเอ็นต์ ซึ่งส่วนไคลเอ็นต์นี้จะทำการฝังตัวลงบนโปรแกรม แล้วเรียกใช้การทำงานของส่วนเซิร์ฟเวอร์ เราจะเรียกโปรแกรมที่สามารถนำวัตถุที่เป็นโอแอลอีไคลเอ็นต์ไปฝังนั้นว่า โอแอลอีคอนเทนเนอร์ (OLE Container)

การทำงานของโอแอลอีนั้นจะมีลักษณะดังนี้

1. การเชื่อมโยงและการฝัง (Linking and Embedding) การเชื่อมโยงและการฝังเป็นวิธีการเชื่อมต่อหรือฝังตัวของวัตถุที่เป็นโอแอลอีไคลเอ็นต์ที่สร้างโดยโปรแกรมโอแอลอีเซิร์ฟเวอร์ลงในเอกสาร

2. อินเพซเอกติเวชัน (In - Place Activation (Visual Editing)) เมื่อเราทำการเรียกตัววัตถุที่เป็นโอแอลอีไคลเอ็นต์ที่ฝัง (Embedding) ในโอแอลอีคอนเทนเนอร์ขึ้นมา การทำงานส่วนติดต่อกับผู้ใช้ของโอแอลอีคอนเทนเนอร์จะเปลี่ยนเป็นรูปแบบของโอแอลอีเซิร์ฟเวอร์เพื่อเรียกฟังก์ชันของโอแอลอีเซิร์ฟเวอร์ขึ้นมาทำงาน แต่ถ้าโอแอลอีไคลเอ็นต์นั้นเป็นวัตถุที่แสดงเพียงแค่การเชื่อมโยง (Linking) ส่วนติดต่อกับผู้ใช้ของโอแอลอีคอนเทนเนอร์จะไม่เปลี่ยนเป็นลักษณะดังกล่าว แต่จะเป็นการเรียกการทำงานโปรแกรมที่เป็นผู้สร้างวัตถุนั้นขึ้นมาทำงานแทน

3. ออโตเมชัน (Automation) เป็นการทำงานในลักษณะที่โปรแกรมหนึ่งสามารถเรียกฟังก์ชันการทำงานของอีกโปรแกรมหนึ่งได้ โปรแกรมที่เป็นผู้เรียกฟังก์ชันจะเรียกว่าโอแอลอีออโตเมชันไคลเอ็นต์ (OLE Automation Client) หรือ โอแอลอีออโตเมชันคอนโทลเลอร์ (OLE Automation Controller) ส่วนโปรแกรมที่เป็นผู้ถูกเรียกนั้นเราจะเรียกว่า โอแอลอีออโตเมชันเซิร์ฟเวอร์ (OLE Automation Server) หรือโอแอลอีออโตเมชันคอมโพเนนต์ (OLE Automation Component)

4. คอมพาวด์ไฟล์ (Compound Files) คอมพาวน์ไฟล์เป็นลักษณะที่ไฟล์หนึ่งไฟล์ประกอบด้วยวัตถุต่าง ๆ ในรูปแบบที่เป็นมาตรฐานประกอบกันเป็นไฟล์นั้น เช่น ภายในเอกสารของไมโครซอฟท์เวิร์ดจะประกอบด้วยวัตถุตัวอักษร, วัตถุรูปภาพ, วัตถุกราฟ, วัตถุลายมือชื่อ เป็นต้น

5. รูปแบบการส่งข้อมูล (Unifrom Data Transfer) การส่งข้อมูลระหว่างโปรแกรมที่สนับสนุนการทำงานแบบโอแอลอีนั้นจะมีรูปแบบที่เป็นมาตรฐาน ได้แก่ ใช้หลักการทำงานของคลิปบอร์ดหรือไดนามิกดาต้าเอ็กซ์เชนจ์ ซึ่งเป็นหลักการสำคัญในการนำข้อมูลมาสร้างเป็นโปรแกรมลายมือชื่อดิจิตอลของคณะผู้จัดทำ

6. ลากและวาง (Drag and Drop) คือการทำงานในรูปแบบลากแล้ววางวัตถุจากโปรแกรมหนึ่งไปอีกโปรแกรมหนึ่ง เป็นรูปแบบการส่งข้อมูลระหว่างโปรแกรมที่ใช้มากในปัจจุบัน

จากหลักการดังกล่าว คณะผู้จัดทำจะนำหลักการของโอแอลอีออโตเมชันและโอแอลอีเซิร์ฟเวอร์นำมาใช้ในโครงงานดังนี้

6.1.1 โอแอลอีออโตเมชัน (OLE Automation)

โอแอลอีออโตเมชันเป็นวิธีการในการจัดการกับวัตถุของโปรแกรม (Application) ที่สนใจ โดยสามารถควบคุมได้จากนอกโปรแกรมนั้น ๆ ซึ่งในการที่จะสามารถใช้งานได้นั้น โปรแกรมที่เขียนขึ้นมาจำเป็นที่จะต้องติดต่อกับโปรแกรมที่สนใจโดยการใช้คอมอินเทอร์เฟซ (COM interface) ซึ่งการใช้อินเทอร์เฟซก็จะประกอบไปด้วยพรอพเพอร์ตี (Property) และเมธอด (Method) ต่าง ๆ เราสามารถเรียกโปรแกรมที่เขียนขึ้นมาเพื่อใช้จัดการกับโปรแกรมอื่นว่าออโตเมชันไคลเอ็นต์ (Automation Client) เพราะเราได้เรียกใช้เมธอดจากโปรแกรมอื่น และในแนวคิดเดียวกันเราสามารถเรียกโปรแกรมที่เราไปติดต่อด้วยว่า ออโตเมชันเซิร์ฟเวอร์ (Automation Server) เพราะสามารถให้บริการต่าง ๆ กับเรา

อินเทอร์เฟซต่าง ๆ จะถูกประกาศไว้ในรูปแบบที่เรียกว่า ODL (Object Description Language) ซึ่งไคลเอ็นต์สามารถเรียกใช้ได้จากไฟล์ชนิดของไลบรารี (Type Library) โปรแกรมต่าง ๆ ที่สามารถเรียกใช้ชนิดของไลบรารีได้จะทำการสร้างซอร์สโค้ดขึ้นมาเพื่อใช้กับออโตเมชันไคลเอ็นต์

การสร้างโปรแกรมให้ทำงานกับโอแอลอีออโตเมชันโดยใช้เอ็มเอฟซี (MFC) และชนิดของไลบรารี (Type Library) มีขั้นตอนดังนี้

1. ทำการนำชนิดของไลบรารีเข้า (Import Type Libray) จากคลาสวิซาร์ด (Class Wizard) โดยการเลือกหัวข้อออโตเมชันภายในคลาสวิซาร์ด

2. ทำการเพิ่มคลาส (Add Class) จากชนิดของไลบรารีจากนั้นเลือกไฟล์ไลบรารีของโปรแกรมที่ต้องการจะติดต่อด้วย ในที่นี้จะเลือกไฟล์ชนิดของไลบรารีของโปรแกรมไมโครซอฟท์เวิร์ด (Microsoft Word) โดยต้องเลือกไฟล์ C:\Program Files\Microsoft Office\Office\Msword8.olb

3. จากนั้นโปรแกรมจะทำการสร้างซอร์สโค้ด (Source Code) ของคลาสที่เราเลือกให้สร้าง ผลที่ได้ออกมาจะเป็นไฟล์ .cpp และ .h ของออโตเมชันเซิร์ฟเวอร์

4. ทำการอินคลูด (include) ไฟล์เฮดเดอร์ของออโตเมชันเซิร์ฟเวอร์

5. เริ่มการติดต่อกับออโตเมชันเซิร์ฟเวอร์ โดยการสั่งคำสั่ง AfxOleInit() เพื่อเป็นการจัดตั้งค่าเริ่มต้นต่าง ๆ

6. หลังจากนั้นจะต้องใส่คำสั่ง AfxEnableControlContainer(); เพื่อที่จะทำให้โปรแกรมสามารถเข้าควบคุมออโตเมชันได้

7. สร้างวัตถุของโปรแกรมที่เป็นออโตเมชันเซิร์ฟเวอร์ โดยการประกาศ _Application app;

8. ทำการกำหนดโปรแกรมที่เราต้องการให้เป็นออโตเมชันเซิร์ฟเวอร์ให้กับวัตถุของออโตเมชันเซิร์ฟเวอร์ โดยการใช้คำสั่ง GetActiveObject();

9. สามารถเริ่มการติดต่อกับโปรแกรมออโตเมชันเซิร์ฟเวอร์ได้

6.1.2 โอแอลอีเซิร์ฟเวอร์ (OLE Server)

โปรแกรมแบบโอแอลอีเซิร์ฟเวอร์เป็นโปรแกรมที่ทำงานเกี่ยวกับวัตถุต่าง ๆ ซึ่งวัตถุเหล่านี้จำเป็นจะต้องมีคลาสที่ใช้ในการสร้างวัตถุโดยโปรแกรมทั้งหมดที่ทำงานบนไมโครซอฟท์วินโดว์สจะรู้จักคลาสเหล่านี้ โดยการไปดูที่รีจีสทรีในส่วนของคีย์ HKEY_CLASSES_ROOT\CLSID ซึ่งเปรียบเสมือนกับตัวเลขที่ใช้บ่งบอกถึงคลาสทั้งหมดที่วินโดว์รู้จักโอแอลอีเซิร์ฟเวอร์จะต้องเป็นโปรแกรมที่ให้บริการในเรื่องของการเพิ่มวัตถุลงในเอกสารของโปรแกรมที่สนับสนุนการทำงานแบบโอแอลอีและต้องสามารถให้บริการในเรื่องของการเชื่อมโยงและฝังตัววัตถุได้ด้วย

ขั้นตอนการสร้างโปรแกรมแบบโอแอลอีเซิร์ฟเวอร์มีดังนี้

1. กำหนดให้โปรแกรมทำงานแบบ SDI (Single Document Interface)

2. คลาสที่ใช้สำหรับแสดงผล หน้าจอของโปรแกรมโอแอลอีเซิร์ฟเวอร์จะสืบทอดมาจากคลาส CView เพื่อใช้ในการแสดงรูปวัตถุลายมือชื่อ (Signature Object)

3. ในเมธอด InitInstance() จะต้องเรียกใช้คำสั่ง AfxOleInit() เพื่อเริ่มต้นการทำงานของโอแอลอี

4. ทำการเพิ่มคลาสไอดี (Class ID) เพื่อเป็นการบอกระบบให้รู้ว่ามีคลาสใหม่ที่เป็นของโอแอลอีเซิร์ฟเวอร์

5. จะต้องทำการแก้ไขคลาสวิว (Class view) ที่ใช้ในการแสดงผล ในกรณีที่มีการแทรกวัตถุของลายมือชื่อลงในเอกสารใด ๆ เพื่อใช้แสดงผลของวัตถุนั้น

คลาสต่าง ๆ ที่เกิดจากการสร้างโปรแกรมแบบโอแอลอีเซิร์ฟเวอร์จะสืบทอดมาจากคลาสต่าง ๆ ดังนี้

1.CFrameWnd เป็นคลาสที่จะใช้เก็บหน้าต่างหลักของโปรแกรมในขณะที่โปรแกรมทำงานแยกต่างหากจากโอแอลอีไคลเอ็นต์

2.CWinApp เป็นคลาสที่ใช้สำหรับโปรแกรมหลัก เพื่อเป็นการกำหนดการทำงานของโปรแกรมหลัก

3.COleServerDoc เป็นคลาสที่ใช้สำหรับเก็บข้อมูลของโปรแกรมหลักเพื่อนำไปบันทึกเป็นไฟล์ได้

4.COleServerItem เป็นคลาส ที่ใช้เก็บวัตถุของไอเท็ม (Item) ต่าง ๆ ที่ถูกใช้โดยโอแอลอีไคลเอ็นต์

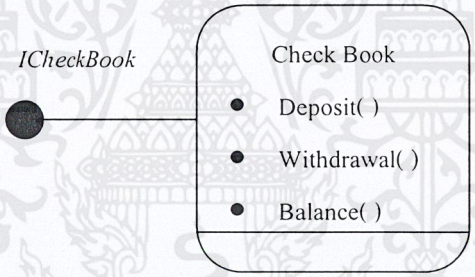
5.CView เป็นคลาสที่ใช้เก็บส่วนของการแสดงผลของหน้าจอหลักของโปรแกรม

6.COleIPFrameWnd เป็นคลาสที่ใช้จัดการกับวิชวลอีดิตติง (Visual Editing) ที่เกิดจากโอแอลอีไคลเอ็นต์

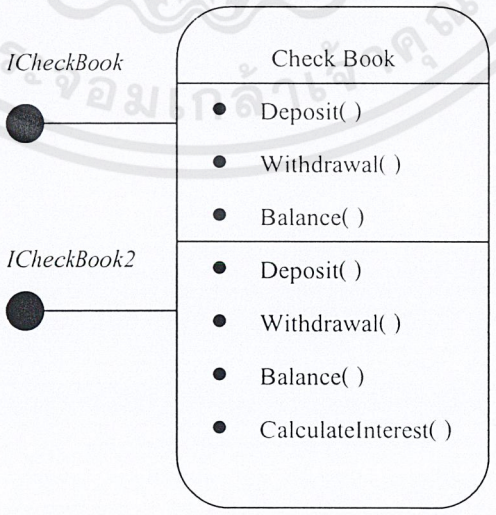
6.2 อินเทอร์เฟซ (Interface)

รูปแบบการเขียนโปรแกรมแบบคอมออปเจ็กต์ (COM – Object) นั้นจะประกอบไปด้วยอินเทอร์เฟซ ซึ่งทำหน้าที่ติดต่อกันระหว่างสองวัตถุ (Object) ใด ๆ โดยเป็นกระบวนการที่นำเมธอด (Method) ที่เราต้องการใช้จากวัตถุ อินเทอร์เฟซต่าง ๆ จะสืบทอด (Derive) มาจากอินเทอร์เฟซ IUnknown ซึ่งจากหลักการของโอแอลอีนั้นเมื่อไคลเอ็นต์ทำการติดต่อกับอินเทอร์เฟซได้ก็จะสามารถเรียกใช้เมธอดของทางเซิร์ฟเวอร์ได้โดยผ่านทางอินเทอร์เฟซนั้น ๆ เมธอดการทำงานต่าง ๆ ของอินเทอร์เฟซนั้นจะถูกเก็บไว้ในตารางวีเทเบิล (V Table) ในรูปแบบของพอยน์เตอร์ชี้ไปที่เมธอดของเซิร์ฟเวอร์

เมื่อตัววัตถุมีการเปลี่ยนแปลงโดยการสร้างหรือเพิ่มเมธอดเข้าไปใหม่ต้องมีการสร้างอินเทอร์เฟซใหม่ซึ่งสืบทอดมาจากอินเทอร์เฟซเดิมเพื่อให้อินเทอร์เฟซใหม่สามารถเห็นเมธอดที่ทำการเพิ่มเข้าไปใหม่ได้ แต่ถ้ามีการปรับปรุงเมธอดที่มีอยู่แล้วต้องทำการเตรียมอินเทอร์เฟซใหม่ที่บรรจุเมธอดที่ปรับปรุงแล้วแต่เราไม่สามารถสืบทอดมาจากอินเทอร์เฟซเดิมได้ เพราะชื่อของเมธอดซ้ำกัน การทำงานไม่เหมือนกัน



รูปที่ 6.1 ตัวอย่างอินเทอร์เฟซของคอมออปเจ็กต์



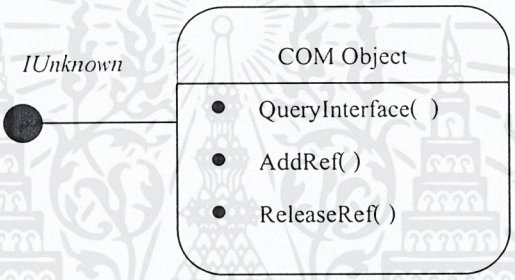
รูปที่ 6.2 ตัวอย่างแสดงการสร้างอินเทอร์เฟซใหม่เมื่อมีเมธอดใหม่

6.2.1 อินเทอร์เฟซ IUnknown

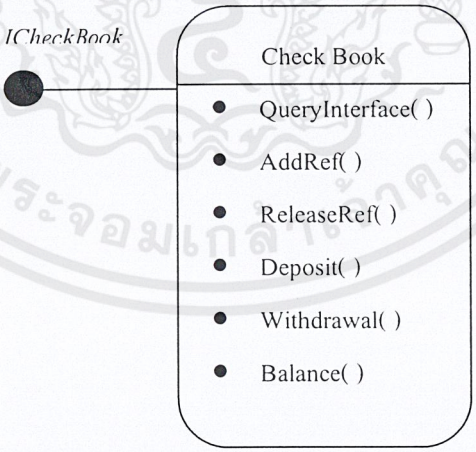
ประกอบด้วยเมธอดหลัก 3 เมธอดด้วยกันคือ

1. QueryInterface() ใช้เพื่อดูว่าวัตถุมีอินเทอร์เฟซที่เราต้องการหรือไม่
2. AddRef() ทำการเพิ่มค่าเรเฟอเรนต์เคาน์ติง
3. ReleaseRef() ทำหน้าที่ลดค่าเรเฟอเรนต์เคาน์ติง

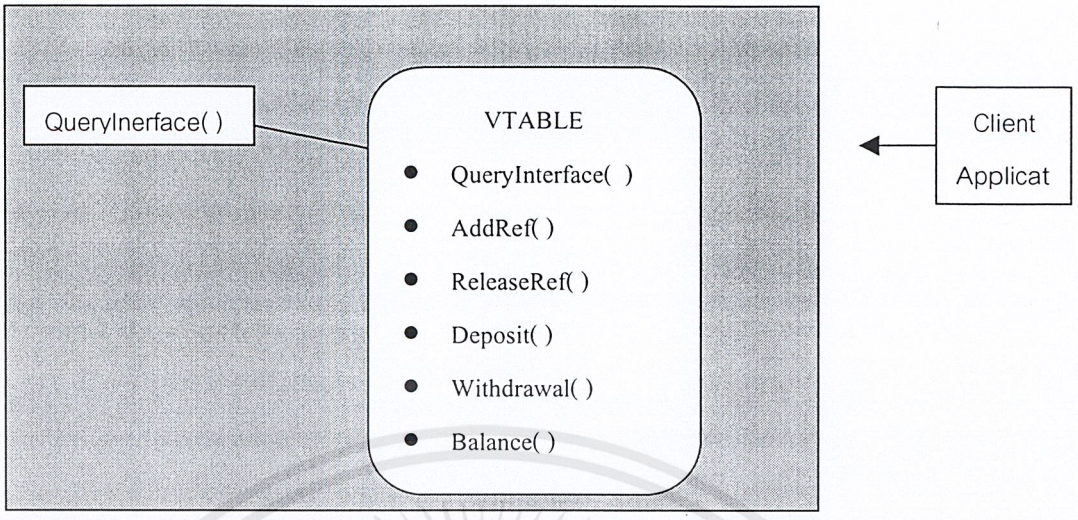
โดยเมื่อเซิร์ฟเวอร์ให้ค่าพอยน์เตอร์ที่ชี้ไปยังอินเทอร์เฟซออกมา ต้องมีการเพิ่มเรเฟอเรนต์เคาน์ติงโดยเรียกผ่านเมธอด AddRef() เมื่อไคลเอ็นต์ทำงานกับเซิร์ฟเวอร์เสร็จสิ้นก็จะลดค่าเรเฟอเรนต์เคาน์ติงลงโดยเรียกผ่านเมธอด ReleaseRef() โดยค่าของเรเฟอเรนต์เคาน์ติงมีไว้เพื่อระบุว่ามีไคลเอ็นต์จำนวนเท่าใดที่ใช้อินเทอร์เฟซนี้บ้าง เมื่อค่าเรเฟอเรนต์เคาน์ติงเป็นศูนย์ก็ไม่สามารถใช้เซิร์ฟเวอร์นั้นได้อีก



รูปที่ 6.3 แสดงอินเทอร์เฟซ IUnknown



รูปที่ 6.4 แสดงอินเทอร์เฟซที่สืบทอดมาจาก IUnknown



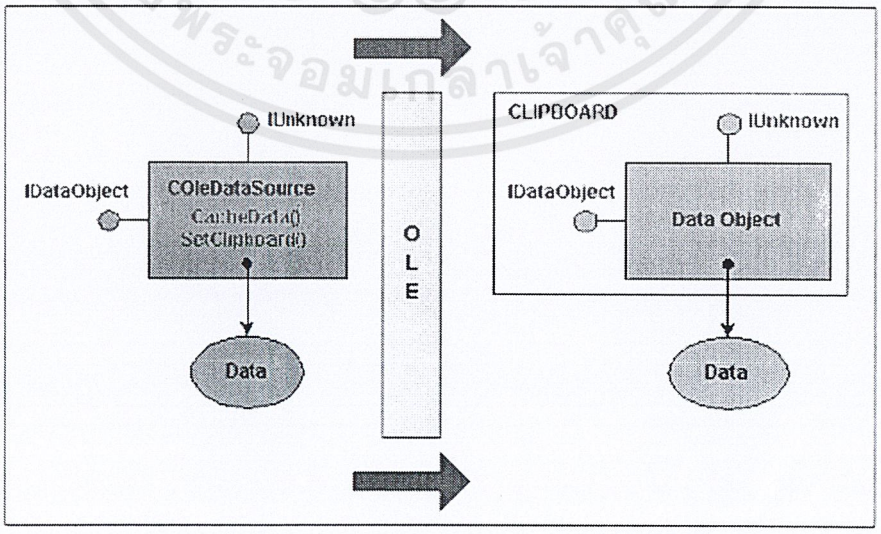
รูปที่ 6.5 แสดงลักษณะ VTABLE

6.2.2 อินเทอร์เฟซ IDataObject

วัตถุข้อมูล (Data Object) หมายถึง วัตถุต่าง ๆ ที่สนับสนุนอินเทอร์เฟซ IDataObject อินเทอร์เฟซ IDataObject เป็นอินเทอร์เฟซที่มีเมธอดในการส่งข้อมูลและบอกถึงการเปลี่ยนแปลงข้อมูลของวัตถุ โดยในการส่งข้อมูลจะต้องมีการระบุรูปแบบของข้อมูล และตัวกลางที่ใช้ในการส่งข้อมูลด้วย

6.3 คลาส COleDataObject

คลาส COleDataObject เป็นคลาสที่ใช้ในการส่งผ่านข้อมูลในหลายรูปแบบระหว่างโปรแกรม ซึ่งข้อมูลอาจมาจากคลิปบอร์ด (Clipboard) ผ่านทางการลากและวาง (Drag and Drop) โดยในความเป็นจริงข้อมูลนั้นจะได้มาจากการสร้างคลาส COleDataSource หรือ ผ่านทางอินเทอร์เฟซ IDataObject ก็ได้



รูปที่ 6.6 แสดงการทำงานของคลาส COleDataObject

ภายในคลาส COleDataObject จะมีฟังก์ชันที่ใช้ในการรับส่งข้อมูลระหว่างโปรแกรมที่สนับสุนนการทำงานในแบบโอแอลอีอยู่ โดยเราสามารถที่จะระบุประเภทของข้อมูลที่เราต้องการผ่านฟังก์ชันในคลาส COleDataObject นี้ได้ประเภทของข้อมูลดังกล่าวคณะผู้จัดทำได้อธิบายไว้ในบทคลิปบอร์ด และการส่งผ่านข้อมูลใด ๆ ก็ตามโดยใช้คลาส COleDataObject นี้เราจำเป็นต้องระบุประเภทของตัวกลางที่ใช้ในการส่งข้อมูลด้วย ตัวกลางดังกล่าวอาจเป็นไฟล์โดยใช้คลาส CFile, การใช้โกลบอลเมมโมรี (HGLOBAL) หรือใช้โครงสร้าง STGMEDIUM เป็นต้น ซึ่งฟังก์ชันที่คณะผู้จัดทำนำมาใช้มีดังต่อไปนี้

1. void COleDataObject::Attach(LPDATAOBJECT lpDataObject, BOOL bAutoRelease = NULL) เป็นฟังก์ชันที่กำหนดจุดเริ่มต้นให้กับตัวแปรของคลาส COleDataObjet ว่ารับข้อมูลในโปรแกรมที่สนับสนุนการทำงานในแบบโอแอลอีตัวใด โดยผ่านทางอินเทอร์เฟซ IDataObject ซึ่ง lpDataObject ชี้อยู่

2. void COleDataObject::BeginEnumFormats() เป็นฟังก์ชันเพื่อเตรียมการใช้งานฟังก์ชัน GetNextFormat เพื่อเก็บค่าตำแหน่งของรูปแบบแรก

3. BOOL COleDataObject::GetNextFormat(LPFORMATETC lpFormatEtc) เป็นฟังก์ชันที่เอาตัวแปรพอยน์เตอร์ชี้ไปยังข้อมูลรูปแบบต่อไป ในกรณีที่รูปแบบของข้อมูลนั้นเป็นรูปแบบสุดท้ายจะคืนค่าออกมาเป็นเท็จนอกนั้นจะค่าออกมาเป็นจริง

4. BOOL ColeDataObject::IsDataAvailable(CLIPBOARD cfFormat, LPFORMATETC lpFormatEtc = NULL) เป็นฟังก์ชันที่ใช้เพื่อตัวสอบว่ามีข้อมูลประเภทคลิปบอร์ดที่ระบุตาม cfFormat ภายในคลาส COleDataObject หรือไม่ ในกรณีที่มีรูปแบบข้อมูลดังกล่าว ฟังก์ชันคืนค่าออกมาเป็นจริงนอกนั้นจะค่าค่าออกมาเป็นเท็จ

5. HGLOBAL COleDataObject::GetGlobalData(CLIPBOARD cfFormat, LPFORMATETC lpFormatEtc = NULL) เป็นฟังก์ชันกำหนดโกลบอลเมมโมรีเพื่อรับค่าข้อมูลประเภทคลิปบอร์ดที่ระบุตาม cfFormat ในกรณีที่กำหนดค่าโกลบอลเมมโมรีสำเร็จจะคืนค่าจริง นอกนั้นคืนค่าเท็จ

6.4 ฟอร์แมตอีทีซี (FORMATETC)

เป็นโครงสร้างข้อมูลที่ระบุรูปแบบของข้อมูลในการส่งข้อมูลของโปรแกรมที่สนับสนุนการทำงานแบบโอแอลอี โครงสร้างของฟอร์แมตอีทีซีโดยย่อมีดังนี้

```
typedef struct tagFORMATETC
{
    CLIPFORMAT          cfFormat;    // รูปแบบของข้อมูลที่ต้องการ เช่น CF_TEXT
    DVTARGETDEVICE      *ptd;        // รายละเอียดโครงสร้างของอุปกรณ์เป้าหมาย
    DWORD               dwAspect;    // ตัวระบุรายละเอียดของวัตถุในคลิปบอร์ด
    LONG                lindex;      // ตัวระบุขอบเขตของข้อมูลที่ต้องการ
```

```
    DWORD           tymed;          // ประเภทของตัวกลางในการส่งข้อมูล
} FORMATETC,  *LPFORMATETC;
```

บทที่ 7
คลิปบอร์ด

7.1 วินโดว์สคลิปบอร์ด

วินโดว์สคลิปบอร์ด เป็นตัวแทนการแลกเปลี่ยนข้อมูลระหว่างแอพพลิเคชันต่าง ๆ ก่อนที่จะมีวิธีลากแล้วปล่อยจากการฝังตัวของโปรแกรมแบบโอแอลอี ผู้ใช้ต้องใช้คลิปบอร์ดเช่น การตัด การคัดลอก การวาง เพื่อส่งข้อมูลจากแอพพลิเคชันหนึ่งไปยังอีกแอพพลิเคชันหนึ่ง หรือย้ายข้อมูลในแอพพลิเคชันเดียวกัน

ปัจจุบันนี้คลิปบอร์ดได้ถูกลืมไปแล้วเพราะถูกแทนที่โดยโอแอลอี ไม่ได้หมายความว่าแอพพลิเคชันจะไม่สนับสนุนการทำงานของคลิปบอร์ด แต่แนวคิดเกี่ยวกับคลิปบอร์ดก็ยังมีอยู่ถึงแม้ว่าแอพพลิเคชันจะแลกเปลี่ยนข้อมูลด้วยวิธีการขั้นสูงกว่าก็ตาม เช่น การคัดลอกโดยกดคีย์ลัด (Ctrl-C) หรือเมนูคัดลอก

จริง ๆ แล้วคลิปบ อร์ดคืออะไร อาจจะเป็นสิ่งอำนวยความสะดวกของ Win32 ซึ่งแอพพลิเคชันสามารถวางข้อมูลไว้ได้ เช่น ข้อมูลสามารถเข้าถึงได้โดยทุกแอพพลิเคชัน ข้อมูลสามารถเป็นได้หลายรูปแบบ บางชนิดสนับสนุนโดยระบบปฏิบัติการ บางชนิดสนับสนุนโดยแอพพลิเคชัน

7.2 คลิปบอร์ดฟอร์แมต

แอพพลิเคชันเก็บข้อมูลลงในคลิปบอร์ดโดยใช้ฟังก์ชัน SetClipboardData ทั้งยังจัดการควบคุมวัตถุที่เป็นข้อมูล ฟังก์ชันนี้สามารถรับพารามิเตอร์ที่เจาะจงรูปแบบของข้อมูลได้ด้วย ทำให้แอพพลิเคชันต่าง ๆ สามารถจัดการข้อมูลในหลาย ๆ รูปแบบได้ ตัวอย่างเช่น เวิร์ดโปรเซสเซอร์สามารถวางข้อมูลบนคลิปบอร์ดโดยใช้ทั้งรูปแบบส่วนตัวและรูปแบบเพลนเท็กซ์ที่เป็นรูปแบบตัวอักษรธรรมดา ซึ่งเพลนเท็กซ์สามารถถูกใช้ได้โดยโปรแกรมอื่น ๆ เช่น Notepad เป็นต้น

มีรูปแบบของคลิปบอร์ดอยู่สามอย่างที่สามารถรวมเข้าไปในแอพพลิเคชันได้คือ

1. รูปแบบมาตรฐาน
2. รูปแบบที่ขึ้นทะเบียนไว้
3. รูปแบบส่วนตัว

7.2.1 สแตนดาร์ดคลิปบอร์ดฟอร์แมต (รูปแบบคลิปบอร์ดมาตรฐาน)

รูปแบบคลิปบอร์ดมาตรฐานมากมายที่มีอยู่ถูกนิยามโดยสัญลักษณ์ที่คงที่ รูปแบบเหล่านี้ถูกรวบรวมไว้ในตาราง ในกรณีที่แอพพลิเคชันถูกกำหนดให้หาตัวควบคุมของรูปแบบเฉพาะขณะเรียกฟังก์ชัน SetClipboardData รูปแบบการควบคุมจะถูกบ่งชี้ว่าข้อมูลที่จะถูกส่งไปในคลิปบอร์ดเป็นข้อมูลชนิดใด

ชนิดของรูปแบบ	คำบรรยาย
Text Formats	
CF_OEMTEXT	เท็กซ์ที่มีอักขระที่มาจากชุดอักขระ OEM
CF_TEXT	เท็กซ์ที่มีอักขระที่มาจากชุดอักขระ ANSI
CF_UNICODETEXT	เท็กซ์ที่มีอักขระแบบ Unicode
Bitmap formats	
CF_BITMAP	Device-dependent bitmap (HBITMAP)
CF_DIB	Device independent bitmap (HBITMAPINFO)
CF_TIFF	Tagged Image File Format
Metafile formats	
CF_ENHMETAFILE	Enhanced metafile (HENHMETAFILE)
CF_METAFILEPICT	Windows Metafile (METAFILEPICT)
Substitute formats for private formats	
CF_DSPBITMAP	บิตแมบที่แสดงข้อมูลเฉพาะของโปรแกรมนั้น
CF_DSPENHMETAFILE	Enhanced metafile ที่แสดงข้อมูลเฉพาะของโปรแกรมนั้น
CF_DSPMETAFILEPICT	Metafile ที่แสดงข้อมูลเฉพาะของโปรแกรมนั้น
CF_DSPTEXT	เท็กซ์ที่แสดงข้อมูลเฉพาะของโปรแกรมนั้น
Sound formats	
CF_RIFF	Resource Interchange File Format
CF_WAVE	ข้อมูลที่เป็นรูปแบบมาตรฐานของแฟ้มประเภทเวฟ
Special formats	
CF_DIF	Data Interchange Format from Software Arts
CF_OWNERDISPLAY	Data displayed by the owner of the clipboard data
CF_PALETTE	Color palette (HPALETTE)
CF_PENDATA	Microsoft Pen Extensions data
CF_PRIVATEFIRST through CF_PRIVATELAST	ข้อมูลเฉพาะโปรแกรมนั้น ๆ
CF_SYLK	Microsoft Symbolic Link format
Windows 95 only formats	

CF_GDIOBJFIRST through CF_GDIOBJLAST	Application-defined GDI objects
CF_HDROP	List of files (HDROP)
CF_LOCALE	Locale information for CF_TEXT data

ตารางที่ 7.1 แสดงรูปแบบคลิปบอร์ดมาตรฐาน

วินโดว์สสามารถสร้างข้อมูลในรูปที่ไม่เฉพาะเจาะจงสำหรับโปรแกรมใดโปรแกรมหนึ่งตัวอย่าง เช่น ถ้าแอพลิเคชันเก็บข้อมูลในรูปแบบ CF_TEXT วินโดว์สสามารถแปลงข้อมูลในรูปแบบ CF_OEMTEXT ไปเป็นรูปแบบที่โปรแกรมอื่นต้องการได้ วินโดว์สสามารถแสดงการเปลี่ยนแปลงของรูปแบบระหว่างรูปแบบเท็กซ์ CF_TEXT, CF_OEMTEXT และ (ภายใต้วินโดว์สเอ็นที) CF_UNICODETEXT รูปแบบบิตแมบ CF_BITMAP และ CF_DIB และรูปแบบเมตาไฟล์ CF_ENHMETAFILE และ CF_METAFILEPICT ท้ายที่สุดวินโดว์สสามารถสร้างรูปแบบ CF_PALETTE จากรูปแบบ CF_DIB ได้

7.2.2 รีจิสเตอร์ฟอร์แมต (รูปแบบที่ถูกขึ้นทะเบียน)

แอพพลิเคชันอื่น ๆ ซึ่งต้องการวางข้อมูลลงบนคลิปบอร์ดในรูปแบบอื่น ๆ นอกเหนือจากรูปแบบมาตรฐาน สามารถขึ้นทะเบียนรูปแบบคลิปบอร์ดใหม่โดยใช้ฟังก์ชัน RegisterClipboardFormat ตัวอย่าง เช่น แอพพลิเคชันต้องการที่จะวางรูปแบบ RTF Text บนคลิปบอร์ดจะต้องทำการขึ้นทะเบียนดังนี้ cfRTF = RegisterClipboardFormat("Rich Text Format") ถ้าหลาย ๆ แอพพลิเคชันเรียก RegisterClipboarFormat ด้วยชื่อรูปแบบเดิม รูปแบบจะถูกขึ้นทะเบียนได้ครั้งเดียวเท่านั้น

มีหลายรูปแบบในคลิปบอร์ดถูกขึ้นทะเบียนโดยวินโดว์ส ตัวอย่างเช่น รูปแบบที่ถูกขึ้นทะเบียนบางอันเกี่ยวข้องกับโอแอลอี บางรูปแบบเกี่ยวข้องกับเชลล์ของวินโดว์ส 95 ชื่อของรูปแบบที่ถูกขึ้นทะเบียนสามารถเรียกดูได้โดยเรียกใช้ฟังก์ชัน GetClipboardFormatName

7.2.3 ไพรเวทฟอร์แมต (รูปแบบส่วนตัว)

บางครั้งแอพพลิเคชันไม่จำเป็นต้องขึ้นทะเบียนรูปแบบคลิปบอร์ดใหม่ กรณีนี้เมื่อคลิปบอร์ดเคยถูกใช้ ตัวอย่าง เช่น การส่งข้อมูลภายในแอพพลิเคชันเดียวกันและข้อมูลไม่ได้ถูกใช้โดยแอพพลิเคชันอื่น แอพพลิเคชันนั้นกำหนดให้เป็นรูปแบบส่วนตัว แอพพลิเคชันนั้นสามารถใช้ CF_PRIVATEFIRST จนถึง CF_PRIVATELAST

ตามระเบียบการทำให้ตัวดูคลิปบอร์ดแสดงข้อมูลที่ถูกเก็บไว้ในรูปแบบส่วนตัวได้นั้น เจ้าของคลิปบอร์ดต้องแสดงข้อมูลในรูปแบบที่แสดงออกมาได้เหล่านี้ CF_DSPBITMAP, CF_DSPTEXT, CF_DSPMETAFILEPICT หรือ CF_DSPENHMETAFILE รูปแบบเหล่านี้เป็นตัวบอกถึงส่วนพื้นฐาน

(CF_BITMAP, CF_TEXT, CF_METAFILEPICT, และ CF_ENHMETAFILE) นอกจากว่าจะถูกใช้เพื่อแสดงผลและไม่ใช่เพื่อการวาง

7.3 การทำงานของคลิปบอร์ด

ตามระเบียบการใช้งานของคลิปบอร์ด แอพพลิเคชันต้องทำงานหลาย ๆ อย่าง ได้แก่ การสร้างข้อมูลคลิปบอร์ด, การขอสิทธิ์การใช้ของคลิปบอร์ด, การส่งผ่านข้อมูล, และการตอบสนองเหตุการณ์ที่เกี่ยวเนื่องกับคลิปบอร์ด แอพพลิเคชันต้องจัดการกับส่วนติดต่อผู้ใช้ จัดการกับคำสั่งของผู้ใช้ที่เจาะจงเกี่ยวกับคลิปบอร์ด (เช่นคำสั่ง ภายใต้เมนูแก้ไข)

7.3.1 การส่งผ่านข้อมูลไปยังคลิปบอร์ด

ก่อนที่ข้อมูลจะถูกส่งไปยังคลิปบอร์ด แอพพลิเคชันต้องทำสองสิ่งคือ

1. วัตถุข้อมูลต้องถูกกำหนดไว้
2. สิทธิ์การใช้คลิปบอร์ดต้องใช้ได้

วัตถุข้อมูลต้องมีตัวควมคุม ตัวควบคุมนี้สามารถอ้างถึงกลุ่มของหน่วยความจำที่ถูกจองโดยการใช้ฟังก์ชัน GlobalAlloc ด้วยแฟลก GMEM_MOVEABLE และ GMEM_DDESHARE (สถานะของแฟลก GMEM_DDESHARE ไม่ได้แสดงถึงกลุ่มของหน่วยความจำที่ใช้ร่วมกันระหว่างแอพพลิเคชัน) แอพพลิเคชันได้รับสิทธิ์การเปิดคลิปบอร์ดโดยการเรียกใช้ฟังก์ชัน OpenClipboard และทำให้คลิปบอร์ดว่างเปล่าโดยเรียกใช้ฟังก์ชัน EmptyClipboard ตัวควบคุมทั้งหมดที่ถูกส่งไปยังคลิปบอร์ดก่อนหน้านี้จะถูกปล่อย ขั้นต่อไปแอพพลิเคชันส่งข้อมูลไปยังคลิปบอร์ดโดยใช้ฟังก์ชัน SetClipboardData และ ปิดคลิปบอร์ดโดยเรียกฟังก์ชัน CloseClipboard แอพพลิเคชันสามารถเรียกฟังก์ชัน SetClipboardData ได้หลาย ๆ ครั้งถ้ามีข้อมูลอยู่หลาย ๆ รูปแบบ ตัวอย่างเช่นแอพพลิเคชันอาจจะเรียก SetClipboardData โดยใช้รูปแบบ CF_DIB และ CF_ENHMETAFILE เพื่อจัดการรูปภาพทั้งในรูปแบบบิตแมบ และรูปแบบเมตาไฟล์

7.3.2 การวางข้อมูลจากคลิปบอร์ด

แอพพลิเคชันสามารถใช้ฟังก์ชัน IsClipboardFormatAvailable เพื่อตรวจสอบดูว่าข้อมูลในรูปแบบที่กำหนดสามารถใช้ได้บนคลิปบอร์ดหรือไม่ ถ้าต้องการที่จะได้รับสำเนาของข้อมูลบนคลิปบอร์ด แอพพลิเคชันสามารถเรียก OpenClipboard, ตามด้วยการเรียก GetClipboardData แอพพลิเคชันควรจะคัดลอกข้อมูลที่ตัวควบคุมนั้นจัดการอยู่ทันที โดยต้องทำก่อนการเรียกฟังก์ชัน CloseClipboard หลังจากการเรียกฟังก์ชัน CloseClipboard การส่งค่าต่าง ๆ ผ่านฟังก์ชัน GetClipboardData จะไม่มีผลทันทีและแอพพลิเคชันอื่นสามารถทำให้คลิปบอร์ดว่างได้

ฟังก์ชัน IsClipboardFormatAvailable สามารถใช้เพื่ออัพเดตไอเทมที่ใช้ปรับแต่งเมนูได้ด้วย ตัวอย่าง เช่น ถ้าฟังก์ชัน IsClipboardFormatAvailable บอกว่าไม่มีข้อมูลคลิปบอร์ดในรูปแบบที่ใช้งานได้ แล้วแอพพลิเคชันจะเข้าใจแอพพลิเคชันจะทำการปิดคำสั่งวางในเมนู (ที่เมนูวางจะเป็นสีเทา)

แอพพลิเคชันสามารถรับข้อมูลเกี่ยวกับรูปแบบข้อมูลที่ใช้ได้ในคลิปบอร์ดโดยเรียกฟังก์ชัน CountClipboardFormats หรือ EnumClipboardFormats

7.3.3 คลิปบอร์ดเมสเสจ

มีหลายวินโดว์สเมสเสจที่ถูกรวมเข้าไปไว้ในคลิปบอร์ดเมสเสจ WM_DESTROYCLIPBOARD ถูกส่งไปยังเจ้าของคลิปบอร์ดเมื่อสารบัญของคลิปบอร์ดถูกทำลาย ในการตอบรับเมสเสจนี้ แอพพลิเคชันอาจจะปลดปล่อยทรัพยากรที่นำไปใช้ในการจัดการหรือการวาดไอเทมของคลิปบอร์ด

กลุ่มของเมสเสจที่ถูกส่งไปยังแอพพลิเคชันที่ข้อมูลอยู่บนคลิปบอร์ดโดยใช้รูปแบบ CF_OWNERDISPLAY ประกอบด้วย WM_ASKCBFORMATNAME, WM_DRAWCLIPBOARD, WM_HSCROLLCLIPBOARD, WM_VSCROLLCLIPBOARD, และ WM_PAINTCLIPBOARD ส่วนกลุ่มของเมสเสจอื่นถูกส่งหรือถูกใช้โดยโปรแกรมคลิปบอร์ดวิวเวอร์

7.3 คลิปบอร์ดวิวเวอร์

คลิปบอร์ดวิวเวอร์ คือ โปรแกรมที่สามารถแสดงข้อมูลปัจจุบันของคลิปบอร์ดได้ ตัวอย่างของคลิปบอร์ดวิวเวอร์ได้แก่โปรแกรม Windows Clipboard Viewer

คลิปบอร์ดวิวเวอร์มีไว้เพียงเพื่อความสะดวกสบายของผู้ใช้เท่านั้นและไม่ทำลายหรือเปลี่ยนแปลงการทำงานของคลิปบอร์ด

7.4 การใช้รูปแบบ RTF Text ในการตรวจสอบการเปลี่ยนแปลงของเอกสาร

ปัญหาที่เกิดขึ้นในการตรวจสอบเอกสารบนโปรแกรมไมโครซอฟท์เวิร์ดคือไม่สามารถดึงข้อมูลประเภท CF_BITMAP และ CF_DIB ซึ่งเป็นข้อมูลประเภทรูปภาพออกมาเพื่อหาค่าแฮชพร้อมกับข้อมูลประเภท CF_TEXT ได้จึงต้องใช้รูปแบบ RTF Text ซึ่งมีข้อมูลหลายประเภทในการหาค่าแฮชแทน

ข้อมูลที่เก็บอยู่ในรูปแบบ RTF Text ที่สำคัญมีดังนี้

1. Font Table เก็บข้อมูลเกี่ยวกับฟอนต์ที่ใช้ในเอกสารทั้งหมด
2. Color Table เก็บข้อมูลเกี่ยวกับสีที่ใช้ในเอกสารทั้งหมด
3. Style Sheet เก็บข้อมูลเกี่ยวกับรูปแบบของคาแร็กเตอร์และพารากราฟ
4. Pictures เก็บข้อมูลเกี่ยวกับลักษณะต่าง ๆ ของรูปภาพที่อยู่ในเอกสาร
5. Objects เก็บข้อมูลเกี่ยวกับวัตถุต่าง ๆ ที่อยู่ในเอกสาร

นอกจากที่กล่าวมานี้ยังมีส่วนประกอบอีกมากที่อธิบายถึงลักษณะของเอกสารฉบับนั้น

RTF Text มีข้อมูลเกี่ยวกับวัตถุต่าง ๆ ที่อยู่ในเอกสารนั้นซึ่งรวมถึงลายมือชื่อดิจิตอลของเอกสารนั้นด้วย ลายมือชื่อดิจิตอลนั้นจะเก็บค่าแฮชที่หาได้เอาไว้ ดังนั้นถ้าทำการพิสูจน์เอกสารที่ลงรายมือชื่อดิจิตอลเอาไว้แล้วโดยการหาค่าแฮชจากข้อมูลประเภท RTF Text ที่ได้ทั้งหมดจะทำให้เกิดข้อผิดพลาด เพราะค่าแฮชที่ได้จะไม่ตรงกับค่าแฮชเดิมถึงแม้ว่าเอกสารไม่มีการเปลี่ยนแปลงก็ตาม

การหาค่าแฮชจึงไม่ควรรวมวัตถุในส่วนที่เก็บค่าแฮชเข้าไปด้วย ใน RTF Text จะมีคีย์เวิร์ด \object บอกว่าเป็นส่วนที่เก็บวัตถุและ *\objclass เก็บชื่อของวัตถุไว้ ทำให้ทราบว่าลายมือชื่อดิจิตอลเก็บอยู่ในส่วนไหนและสามารถแยกออกไปในขั้นตอนการหาค่าแฮชได้

บทที่ 8
การออกแบบโปรแกรม

8.1 หลักการและแนวคิดการออกแบบ

คณะผู้จัดทำเลือกใช้ภาษาซีพลัสพลัส (C++) และใช้โปรแกรมไมโครซอฟท์วิชวลซีพลัสพลัส (Microsoft Visual C++) ในการพัฒนาตัวโปรแกรม จากแนวคิดเชิงวัตถุ (Object) ของภาษาซีพลัสพลัส และการเขียนโปรแกรมแบบโอโอพี (Object Oriented Programming) มีความสามารถในแยกส่วนที่เป็นข้อมูลและการดำเนินงานของวัตถุต่าง ๆ ออกจากกัน โดยเมื่อมีการอ้างถึงข้อมูลภายในวัตถุจะกระทำผ่านเมธอดของวัตถุแทน (Encapsulation) และยังมีการสืบทอดคุณสมบัติของวัตถุหนึ่งให้กับวัตถุอื่น ๆ ได้อีกด้วย (Inheritance) เป็นการประหยัดระยะเวลาในการพัฒนาโปรแกรมและทำให้ประสิทธิภาพในการออกแบบโปรแกรมดีขึ้น

นอกจากนี้ในส่วนติดต่อกับผู้ใช้คณะผู้จัดทำได้ใช้ไลบรารีที่ชื่อว่า เอ็มเอฟซี (MFC - Microsoft Foundation Class) เป็นไลบรารีที่ทางบริษัทไมโครซอฟท์สร้างขึ้นเพื่อช่วยให้นักพัฒนาโปรแกรมประยุกต์เขียนโปรแกรมได้ง่ายขึ้น ซึ่งภายในตัวเอ็มเอฟซีเองจะประกอบด้วยคลาสพื้นฐานต่าง ๆ ที่ต้องใช้ในการสร้างหรือแสดงผลในระบบวินโดว์ โดยจะช่วยให้โปรแกรมประยุกต์ที่เขียนขึ้นนั้นมีขนาดเล็กลงและไม่มีความซับซ้อนมาก ทำให้การเขียนโปรแกรมประยุกต์ง่ายขึ้น

หลักการออกแบบโปรแกรมเมื่อพิจารณาการใช้งานโปรแกรมนั้น ผู้ใช้ต้องสามารถกระทำการต่าง ๆ ได้ดังนี้

การใช้งาน	คำอธิบาย
Sign Siganature	เป็นการลงลายมือชื่อดิจิตอลแล้วฝังตัวในเอกสาร
Verify Signature	เป็นการตรวจสอบการเปลี่ยนแปลงและพิสูจน์ที่มาของเอกสารว่ามีความถูกต้องหรือไม่
Delete Signature	เป็นการลบลายมือชื่อดิจิตอลที่สร้างขึ้น
Export Digital Certificate	เป็นการส่งเอกสารสิทธิ์ของคนอื่น ๆ ให้อยู่ในรูปของไฟล์ขนาดเล็ก ซึ่งจะกล่าวถึงต่อไป
Import Digital Certificate	เป็นการนำเอกสารสิทธิ์ของคนอื่น ๆ ในรูปของไฟล์มาเก็บไว้ในที่เก็บเอกสารสิทธิ์
Delete Digital Certificate	เป็นการลบเอกสารสิทธิ์ออกจากที่เก็บเอกสารสิทธิ์

ตารางที่ 8.1 ตารางแสดงการใช้งานโปรแกรมของผู้ใช้

การทำงานของโปรแกรมแบ่งออกเป็น 4 ส่วนดังที่แสดงในรูปที่ 8.1

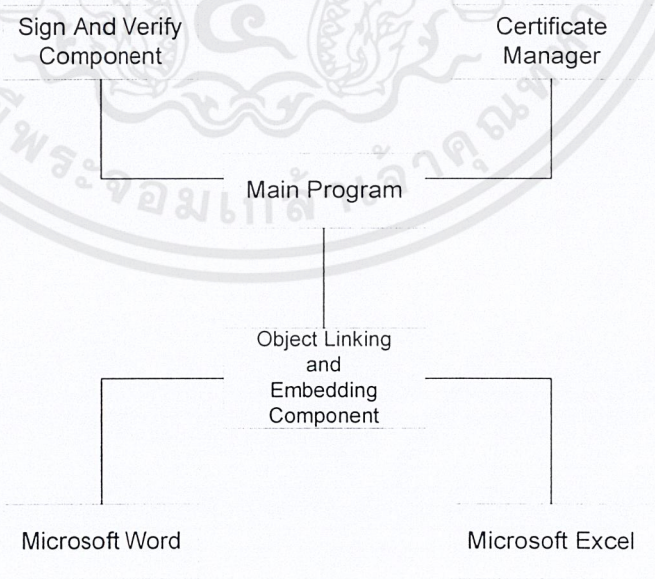
1. SignAndVerify เป็นส่วนที่ทำหน้าที่ในการสร้างและตรวจสอบลายมือชื่อดิจิตอล ฟังก์ชันการทำงานที่อยู่ในส่วนนี้จะใช้คริปโตเอพีไอที่กล่าวมาเป็นหลัก

2. CertificateManagement เป็นส่วนที่ทำหน้าที่ติดต่อกับที่เก็บเอกสารสิทธิ์ (Certificate Store) ในการลบ, ดูข้อมูลของเอกสารสิทธิ์ รวมถึงการนำเข้าและส่งออกเอกสารสิทธิ์จากที่เก็บเอกสารสิทธิ์ให้อยู่ในรูปของไฟล์ที่เก็บเอกสารสิทธิ์พร้อมทั้งเข้ารหัสไว้ด้วย ไฟล์ที่ส่งออกมามีหลายรูปแบบ ได้แก่

1. DER encoded binary X.509 (.CER)
2. Base – 64 encoded binary X.509 (.CER)
3. Cryptographic Message Syntax Standard – PKCS #7 Certificates (.P7B)

3. Object Linking and Embedding เป็นส่วนที่ทำหน้าที่ในการติดต่อกับโอแอลอีทั้งหมดของโปรแกรม ได้แก่ คลาสในการสร้างโอแอลอีเซิร์ฟเวอร์และโอแอลอีไคลเอ็นต์, คลาสที่ติดต่อกับโปรแกรมไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซลในการนำข้อมูลมาหาค่าเมส-เซจไดเจสต์โดยใช้ฟังก์ชันในคลาส COleDataObject โดยใช้หลักการของโอแอลอีออโตเม-ชันตามที่กล่าวมาแล้ว

4. MainProgram เป็นส่วนที่ควบคุมการทำงานทั้งหมดของโปรแกรม ได้แก่ ส่วนติดต่อกับผู้ใช้, ส่วนควบคุมการดำเนินไปของโปรแกรมต่าง ๆ , ส่วนควบคุมอีเวนต์ (Event) ต่าง ๆ ที่กระทำบนตัวโปรแกรม



รูปที่ 8.1 บล็อกไดอะแกรมแสดงโครงสร้างของโปรแกรม

การออกแบบโปรแกรมนั้นจะออกแบบในลักษณะด็อกคิวเมนต์/วิว (Document/View) คือ แยกส่วนที่เก็บข้อมูลและส่วนที่แสดงผลของวัตถุลายมือชื่อดิจิตอลออกจากกัน ส่วนที่เก็บข้อมูลคือส่วนด็อกคิวเมนต์จะเก็บข้อมูลของวัตถุลายมือชื่อที่โปรแกรมสร้างขึ้นแล้วฝังในเอกสาร เมื่อวัตถุฝังในเอกสารแล้ว ส่วนวิวจะทำหน้าที่ในการแสดงผลตัววัตถุ

8.2 การทำงานของคลาสที่สำคัญ

8.2.1 CSignAndVerify

ทำหน้าที่ในการสร้างและตรวจสอบลายมือชื่อดิจิตอล ในการสร้างลายมือชื่อดิจิตอลนั้นผู้ใช้ต้องทำการเลือกคีย์ส่วนตัวที่สอดคล้องกับเอกสารสิทธิ์ที่ได้รับการยืนยันจากองค์กรพิสูจน์สิทธิ์เสียก่อน ลำดับแรกภายในคลาสนี้จะทำการสร้างแฮนเดิลที่ติดต่อกับ CSP ขึ้นมาก่อน โดยใช้ฟังก์ชัน Crypt AcquireContext จากนั้นจะทำการสร้างแฮนเดิลติดต่อกับวัตถุต่าง ๆ ที่อยู่ภายใน CSP ผ่านฟังก์ชันต่อไปนี้

1. FindHashValue ฟังก์ชันในการหาค่าเมสเซจไดเจสต์ ซึ่งมีให้เลือกด้วยกัน 3 อัลกอริทึม ได้แก่ MD4, MD5 และ SHA – 1
2. SignSignature ฟังก์ชันในการสร้างลายมือชื่อดิจิตอล
3. VerifySignature ฟังก์ชันในการตรวจสอบลายมือชื่อดิจิตอล
4. GetHashValue ฟังก์ชันในการส่งค่าเมสเซจไดเจสต์ออกมา

8.2.2 CCertStore

ทำหน้าที่ในการติดต่อกับที่เก็บเอกสารสิทธิ์ที่อยู่ภายในเครื่องคอมพิวเตอร์เครื่องนั้น ได้แก่ ทำหน้าที่ในการติดต่อกับส่วนนำเข้าและส่งออกเอกสารสิทธิ์จากเครื่องคอมพิวเตอร์, การลบเอกสารสิทธิ์, การดูข้อมูลต่าง ๆ ของเอกสารสิทธิ์ โดยเรียกใช้ฟังก์ชัน CertMgr ของไมโครซอฟท์อินเตอร์เน็ตเอ็กซ์-พลอเรอร์ ในการใช้งานเอกสารสิทธิ์ที่อยู่ในเครื่องคอมพิวเตอร์นั้นเพื่อสร้างลายมือชื่อดิจิตอล โปรแกรมจะพิจารณาเอกสารสิทธิ์ที่อยู่ในช่วงเวลาที่องค์กรพิสูจน์สิทธิ์รับรองเท่านั้น ถ้า ณ เวลาปัจจุบันเอกสารสิทธิ์ใดไม่ถูกรับรองโปรแกรมจะไม่อนุญาตให้ผู้ใช้ใช้เอกสารสิทธิ์นั้นในการสร้างลายมือชื่อดิจิตอล ฟังก์ชันที่ใช้มีดังนี้

1. ReadCertificateInStore อ่านค่าของเอกสารสิทธิ์ที่เก็บอยู่ในที่เก็บเอกสารสิทธิ์ออกมาเก็บไว้ในรายการเพื่อแสดงให้ผู้ใช้เลือกในการลงนาม
2. HaveIssueName เป็นฟังก์ชันตรวจสอบว่าเอกสารสิทธิ์ดังกล่าวว่ามีการรับรองโดยองค์กรพิสูจน์สิทธิ์หรือไม่
3. GetTime เป็นฟังก์ชันในการแปลงค่าฟิลด์เวลาในเอกสารสิทธิ์ในรูปวันเดือนปี เพื่อใช้ในการตรวจสอบว่าเอกสารสิทธิ์อยู่ในช่วงเวลาที่ใช้ได้หรือไม่

8.2.3 CSignatureInformation

เป็นคลาสที่เก็บข้อมูลเกี่ยวกับลายมือชื่อดิจิตอล ได้แก่

1. ชื่อเจ้าของเอกสารสิทธิ์
2. องค์กรที่รับรองเอกสารสิทธิ์นั้น
3. วันและเวลาที่ลงนามลายมือชื่อดิจิตอลซึ่งอยู่ในรูปเวลาสากล (UTC - Universal Coordinate Time)
4. แฮชอัลกอริทึมที่ใช้ในการหาเมสเซจไดเจสต์
5. ลายมือชื่อดิจิตอลซึ่งอยู่ในรูปของบิตสตรีม
6. ความยาวของลายมือชื่อดิจิตอล

8.2.4 CISAGSignApp

เป็นคลาสที่ทำหน้าที่หลักของโปรแกรม ควบคุมการติดต่อระหว่างคลาสต่าง ๆ ได้แก่ ส่วนเก็บข้อมูลลายมือชื่อดิจิตอล, ส่วนติดต่อกับผู้ใช้, ส่วนติดต่อกับที่เก็บเอกสารสิทธิ์, ส่วนที่ใช้ในการสร้างและตรวจสอบลายมือชื่อดิจิตอล, ส่วนติดต่อกับโปรแกรมไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซล ตลอดจนส่วนที่จัดการกับความผิดพลาดต่าง ๆ ของโปรแกรม

8.2.5 COleGetData

เป็นคลาสที่ทำหน้าที่ในการติดต่อกับโปรแกรมไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซล ภายในคลาสจะมีฟังก์ชันทำงานแบบโอแอลอีโอโตเมชั่นเพื่อดึงข้อมูลออกมาให้กับคลาส CSignAndVerify ในการหาค่าเมสเซจไดเจสต์และสร้างลายมือชื่อดิจิตอลต่อไป

8.2.6 CISAGSignDoc

คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส COleDocument ซึ่งสืบทอดมากจากคลาส CDocument อีกต่อหนึ่ง เป็นคลาสที่เก็บข้อมูลของลายมือชื่อดิจิตอลที่สร้างขึ้นในเอกสาร ได้แก่ ตัวแปรที่เป็นชนิดของคลาส CSignatureInformation การทำงานในคลาสนี้จะมีฟังก์ชันต่าง ๆ ที่สำคัญดังนี้

1. Serialize() ฟังก์ชันทำหน้าที่ในการนำข้อมูลจากด็อกคิวเมนต์วัตถุลายมือชื่อและบันทึกข้อมูลของวัตถุลายมือชื่อดิจิตอลลงด็อกคิวเมนต์
2. OnNewDocument() ฟังก์ชันในการสร้างด็อกคิวเมนต์หรือที่เก็บข้อมูลลายมือชื่อขึ้นมาใหม่

8.2.7 CISAGSignView

คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส CView เป็นคลาสที่ทำหน้าที่ในการแสดงรูปร่างลักษณะตัวโปรแกรมที่สร้างลายมือชื่อดิจิตอล คือเมื่อผู้ใช้ทำการดับเบิลคลิกที่ตัววัตถุลายมือชื่อดิจิตอล ตัวโปรแกรมโอแอลอีเซิร์ฟเวอร์จะถูกเรียกขึ้นมาส่วนที่เป็นเมนูบาร์ของโปรแกรมโอแอลอีคอนเทนเนอร์จะเปลี่ยนไปเป็นแบบโปรแกรมโอแอลอีเซิร์ฟเวอร์ซึ่งในส่วนนี้จะเป็นหน้าที่ของคลาส IpFrame ซึ่งจะกล่าวถึงต่อไป ส่วนที่เป็นพื้นที่แสดงผลจะเป็นส่วนของโปรแกรมโอแอลอีเซิร์ฟเวอร์เช่นกันคลาสนี้จะดูแลการแสดงผลในส่วนนี้นั่นเอง

8.2.8 CISAGSignSrvrItem

เป็นคลาสที่ทำหน้าที่ในการดูแลการแสดงผลรูปวัตถุลายมือชื่อดิจิตอล ที่แทรกอยู่ในเอกสาร

นอกจากนี้ยังมีคลาสที่เป็นส่วนติดต่อกับผู้ใช้อีกดังนี้

8.2.9 CSignStep01Dlg

คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส CDialog เป็นไดอะล็อกการสร้างลายมือชื่อดิจิตอลขั้นตอนที่หนึ่ง คือการเลือกคีย์ส่วนตัวที่สอดคล้องกับเอกสารสิทธิ์ที่อยู่ภายในที่เก็บเอกสารสิทธิ์

8.2.10 CSignStep02Dlg

คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส CDialog เป็นไดอะล็อกการสร้างลายมือชื่อดิจิตอลขั้นตอนที่สอง คือการป้อนข้อมูลเกี่ยวกับการสร้างลายมือชื่อ ได้แก่ แฮชอัลกอลิทึมที่ใช้, ระยะเวลาที่ลายมือชื่อรับรอง, รูปภาพที่แสดงวัตถุลายมือชื่อเมื่อถูกฝังลงในเอกสาร

8.2.11 CSignStep03Dlg

คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส CDialog เป็นไดอะล็อกการสร้างลายมือชื่อดิจิตอลขั้นตอนที่สาม ไดอะล็อกแสดงให้ผู้ใช้รอ ขณะทำการสร้างลายมือชื่อดิจิตอล

8.2.12 CVerifySignatureDlg

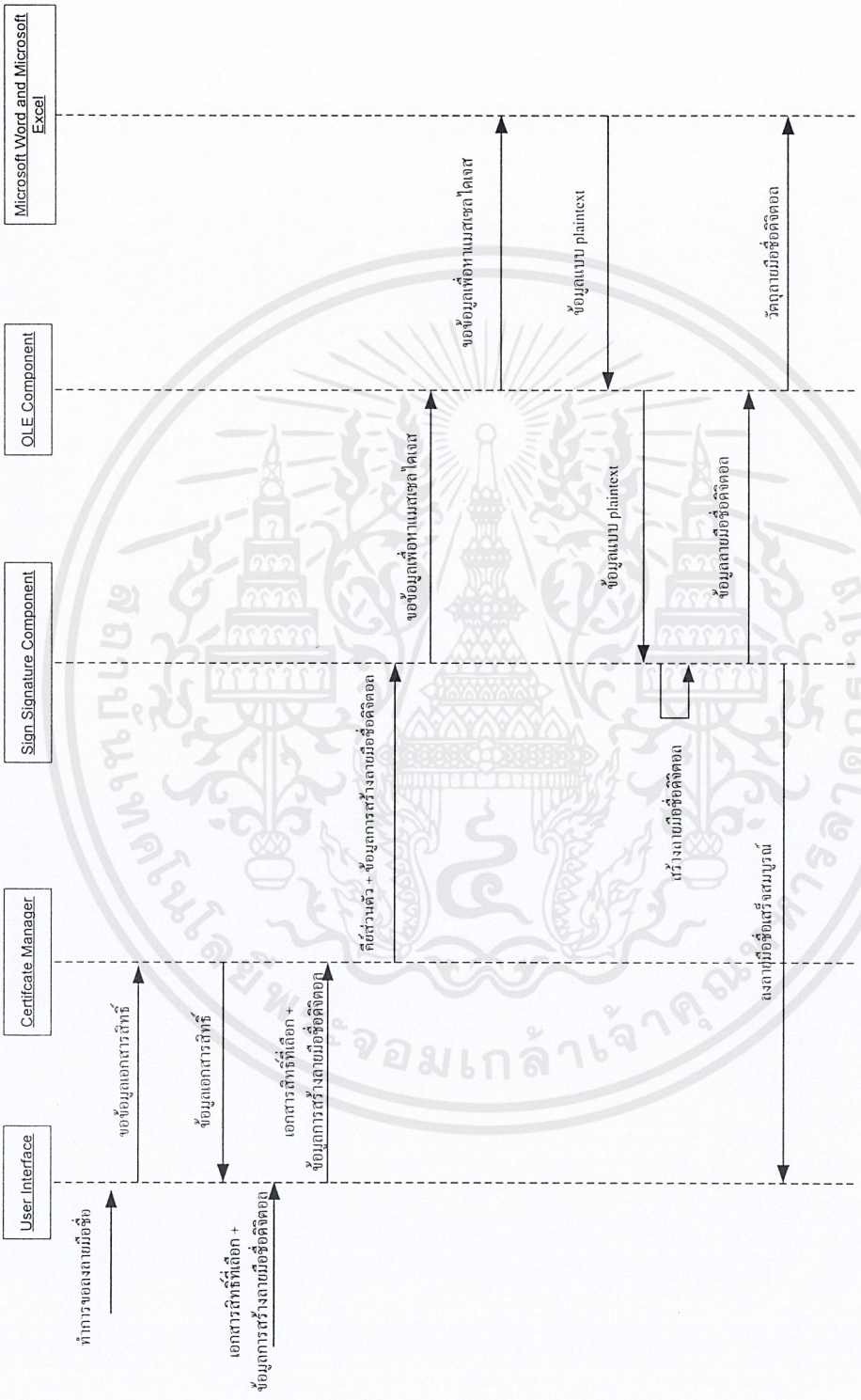
คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส CDialog เป็นไดอะล็อกแสดงผลการตรวจสอบลายมือชื่อดิจิตอลว่า เอกสารมีการเปลี่ยนแปลงหรือไม่

8.2.13 CMainFrame

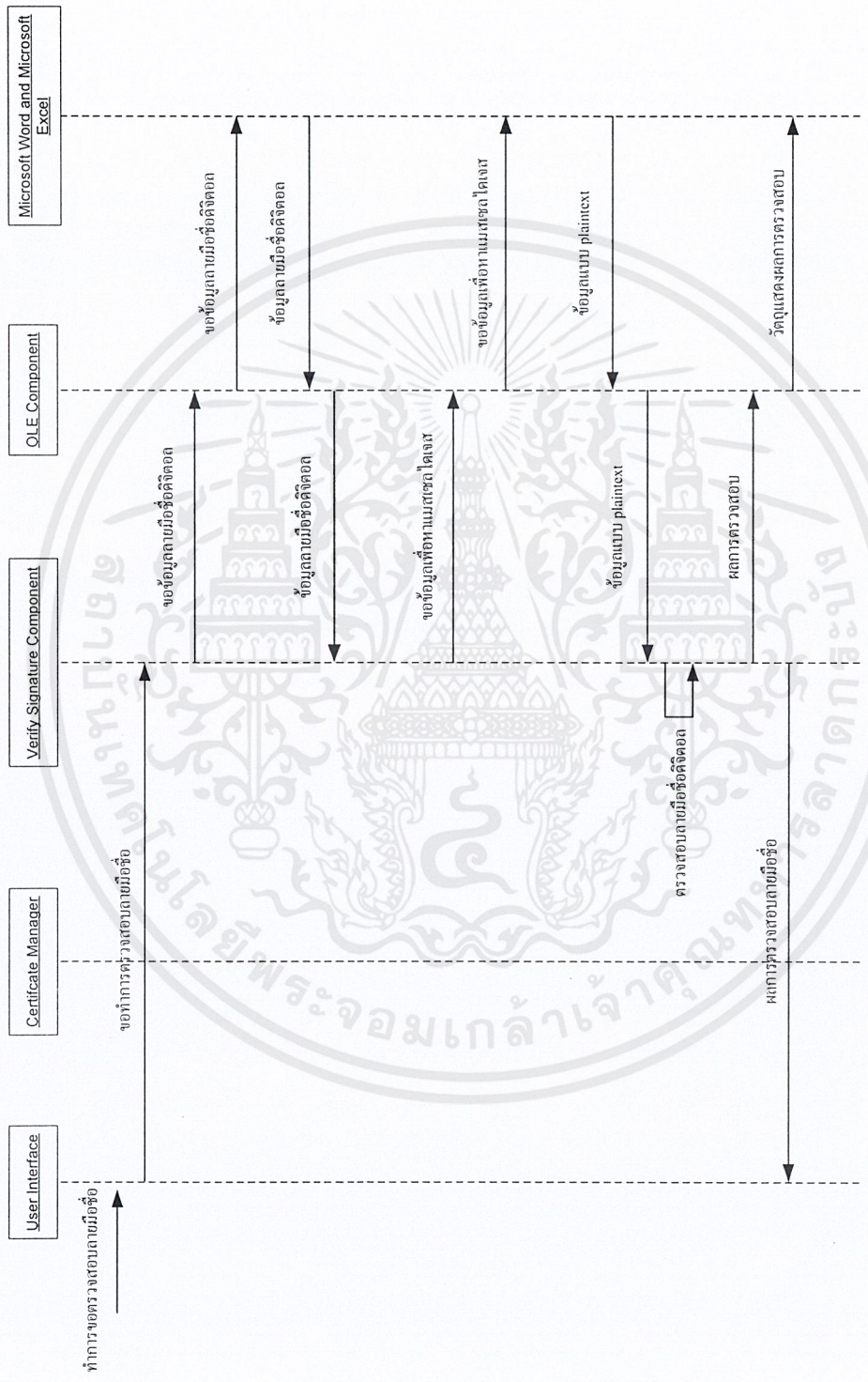
คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส CFrameWnd เป็นคลาสเกี่ยวกับส่วนติดต่อกับผู้ใช้หรือตัววินโดว์หลักของโปรแกรม ได้แก่ แถบเครื่องมือ (Tool bar), แถบบอกสถานะ (Status bar) และเมนูบาร์ (Menu bar)

8.2.14CIpFrame

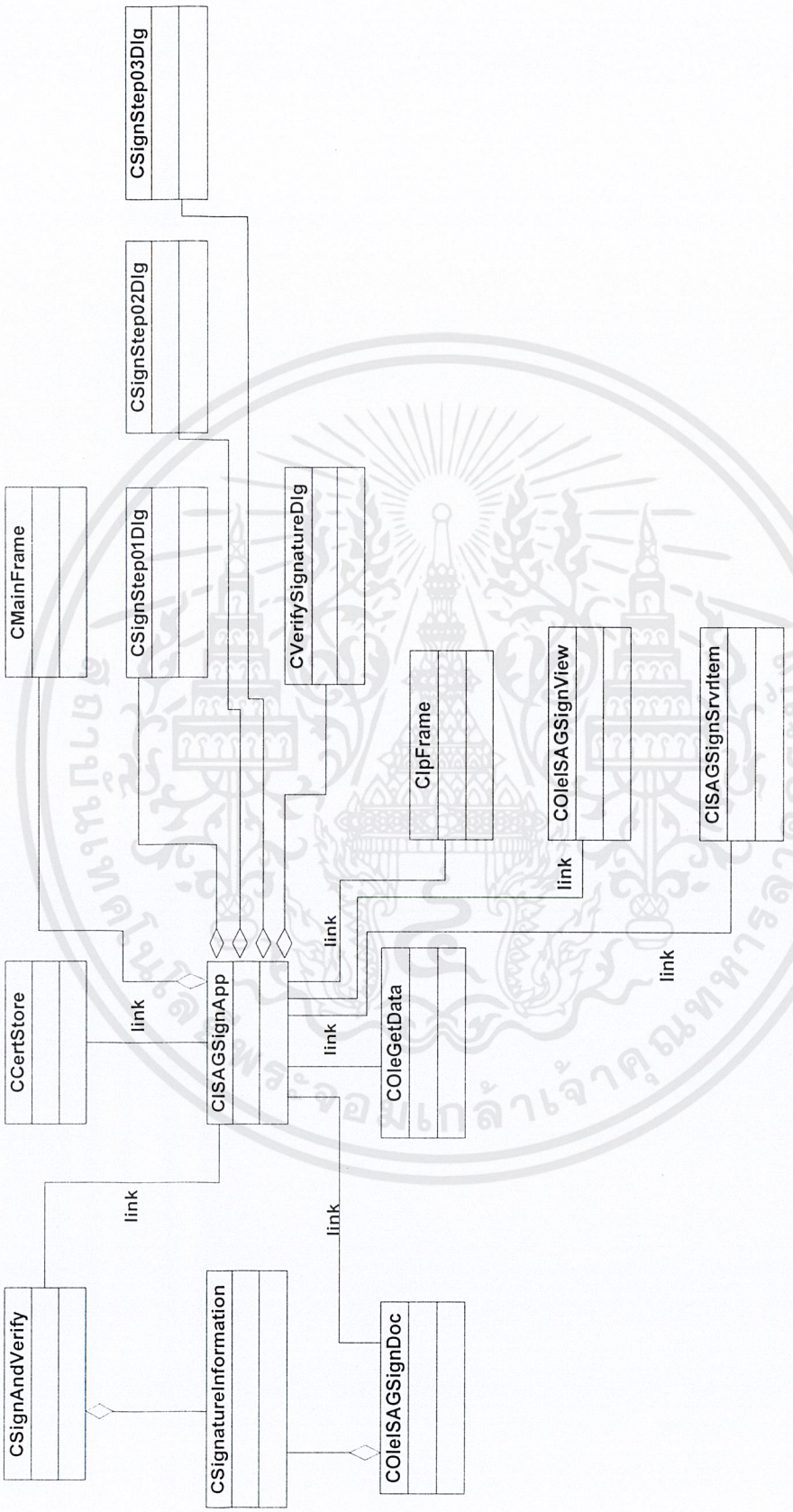
คลาสนี้เป็นคลาสที่สืบทอดมาจากคลาส COleIPFrameWnd เป็นคลาสเกี่ยวกับแสดงส่วนของเมนูบาร์ของโปรแกรมที่เป็นโอแอลอีเซิร์ฟเวอร์ เมื่อตัววัตถุลายมือชื่อมีการเรียกใช้งาน



รูปที่ 8.2 แสดงซีเควนซ์ไดอะแกรมในการลงลายมือชื่อดิจิตอล



รูปที่ 8.3 แสดงซีเควนซ์ไดอะแกรมในการตรวจสอบลายมือชื่อดิจิตอล



รูปที่ 8.4 แสดงคลาสไดอะแกรมของโปรแกรม

บทที่ 9

การทดลองและผลการทดลอง

9.1 ความต้องการของระบบ

ระบบที่จะใช้โปรแกรม ISAGSign ได้จะต้องมีคุณสมบัติดังนี้

1. เครื่องที่ได้ลงระบบปฏิบัติการวินโดวส์ 95 OSR 2.0 ขึ้นไป

2. ในเครื่องจะต้องรองรับการทำงานของ CryptoAPI 2.0 หรือได้ทำการลง Internet Explorer 4.0 ขึ้นไปเนื่องจากถ้าเป็น Internet Explorer เวอร์ชันที่ต่ำกว่า 4.0 แล้วจะทำให้การสร้างคีย์ไม่สามารถกำหนดรหัสผ่านได้

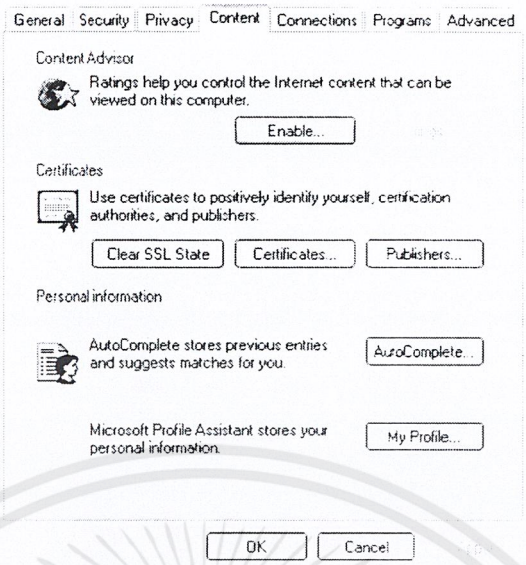
3. จะต้องมีโปรแกรมไมโครซอฟท์เวิร์ด 97 อยู่ในเครื่อง

9.2 ระบบที่ใช้ทดสอบ

1. ระบบปฏิบัติการวินโดวส์ 2000
2. รองรับการทำงานของ CryptoAPI 2.0
3. มีโปรแกรมไมโครซอฟท์เวิร์ด 97 ไว้

9.3 ก่อนการทดสอบโปรแกรม

ก่อนที่จะใช้งานโปรแกรมลงลายมือชื่อดิจิตอลนั้น ผู้ใช้จะต้องมีเอกสารสิทธิ์ก่อน การทดลองนี้ได้ทดลองขอเอกสารสิทธิ์แบบทดลองใช้ผ่านทางเว็บขององค์กรพิสูจน์สิทธิ์ เมื่อได้เอกสารสิทธิ์มาแล้วจึงเพิ่มเอกสารสิทธิ์เข้าไปในระบบโดยเปิด Control Panel > Internet Options เลือกหัวข้อ Content แล้วคลิกที่ปุ่ม Certificates จะปรากฏรายชื่อของเอกสารสิทธิ์ที่มีอยู่ในระบบอยู่แล้ว จากนั้นคลิกที่ปุ่ม Import แล้วทำตามขั้นตอนที่ระบุไว้ เอกสารสิทธิ์ที่ต้องการจะเข้าไปอยู่ในระบบ

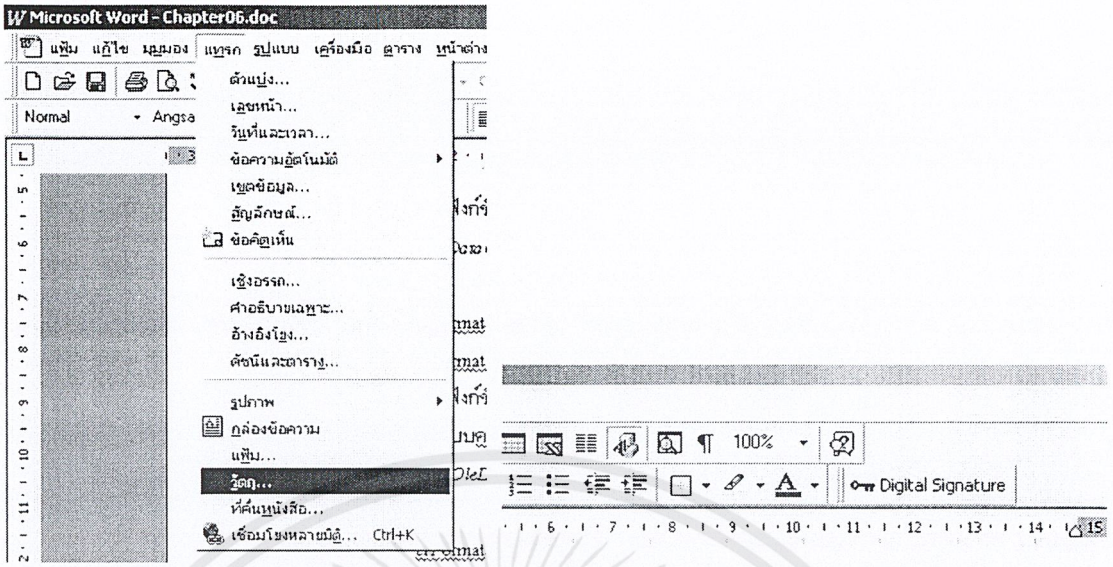


รูปที่ 9.1 หน้าจอของ Internet Options

รูปที่ 9.2 หน้าจอของ Certificates

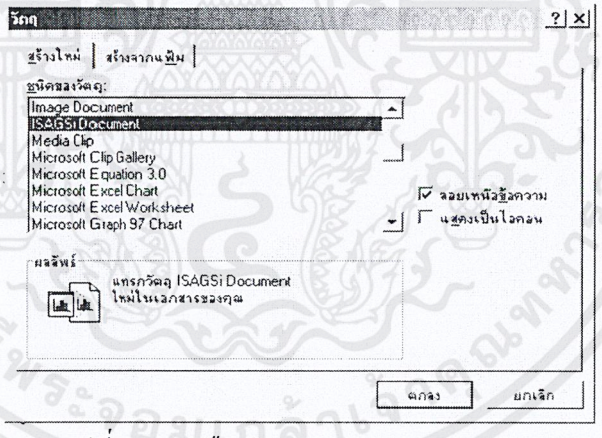
9.4 การทดสอบโปรแกรม ISAGSign กับไมโครซอฟท์เวิร์ด 97

1. เปิดเอกสารขึ้นมา 1 ฉบับด้วยโปรแกรมไมโครซอฟท์เวิร์ด 97
2. เลือกจุดที่จะลงลายมือชื่อดิจิตอลในเอกสารแล้วเลือกเมนู แทรก > วัตถุ หรือแถบเครื่องมือ

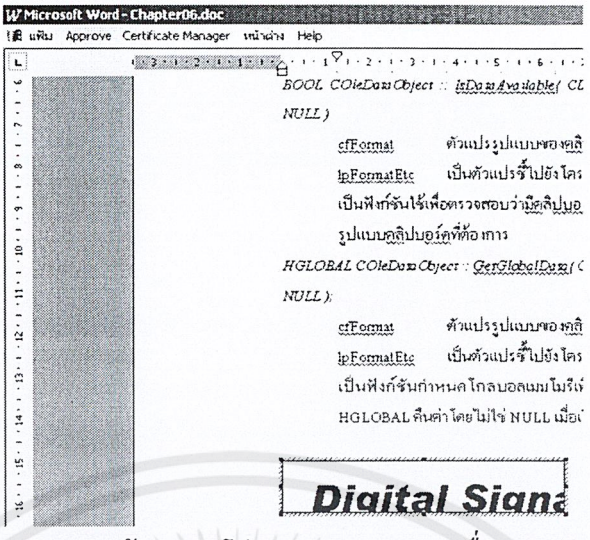


รูปที่ 9.3 การแทรกวัตถุในโปรแกรมไมโครซอฟท์เวิร์ด 97

3. เลือกวัตถุ ISAGSign Document จะปรากฏวัตถุชื่อ Digital Signature ในเอกสารและเมนูของโปรแกรมไมโครซอฟท์เวิร์ด 97 จะเปลี่ยนไปเป็นเมนูของโปรแกรม ISAGSign



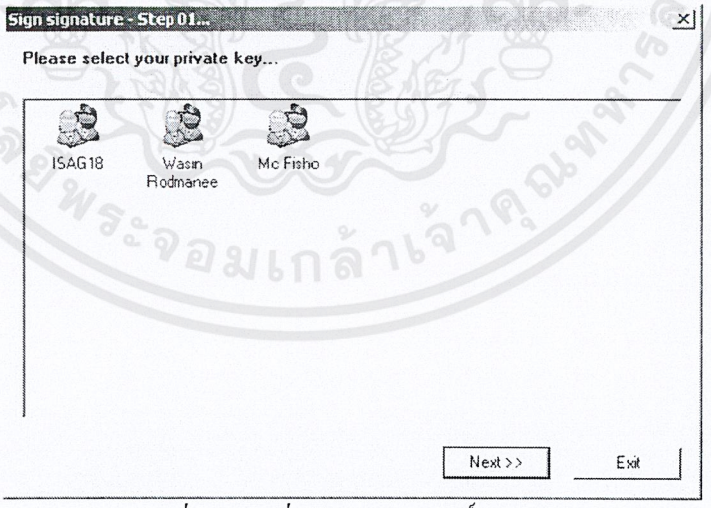
รูปที่ 9.4 การเลือกวัตถุ ISAGSign Document



รูปที่ 9.5 แสดงหน้าจอของโปรแกรม ISAGSign เมื่อแทรกวัตถุแล้ว

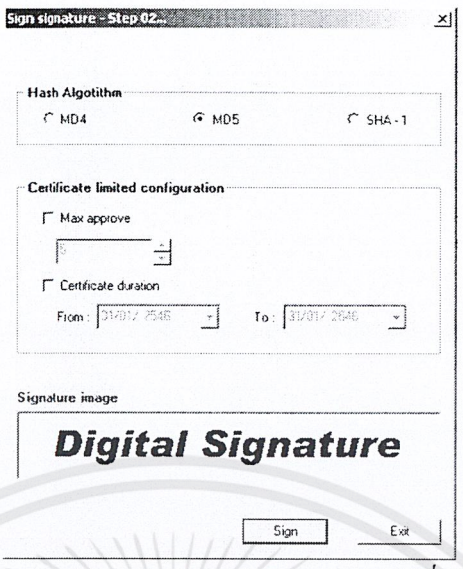
4. ทดลองการลงลายมือชื่อดิจิตอลก่อน โดยเลือกเมนู Approve > Sign Signature จะปรากฏรายชื่อผู้มีเอกสารสิทธิ์อยู่ในระบบ เลือกชื่อที่ต้องการแล้วกดปุ่ม Next

รูปที่ 9.6 การเลือกเมนู Sign Signature



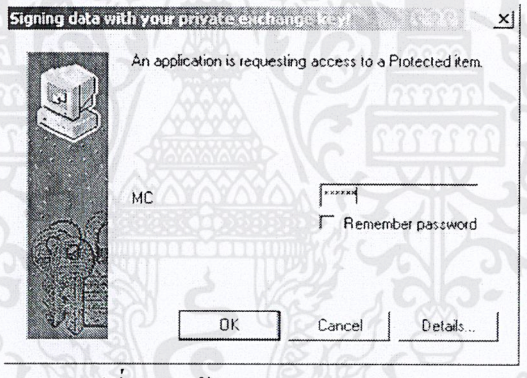
รูปที่ 9.7 รายชื่อผู้มีเอกสารสิทธิ์ในระบบ

5. ขั้นตอนที่ 2 เป็นการเลือกอัลกอริทึมของการหาค่าแฮชและข้อจำกัดต่าง ๆ ของลายมือชื่อดิจิตอลนี้ การทดลองนี้ได้เลือกแฮชอัลกอริทึมแบบ MD-5 จากนั้นคลิก Sign



รูปที่ 9.8 การเลือกคุณสมบัติต่าง ๆ ของลายมือชื่อดิจิตอล

6. กรอกรหัสผ่านแล้วคลิก OK



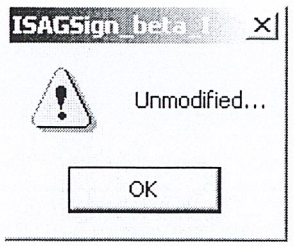
รูปที่ 9.9 หน้าจอการกรอกรหัสผ่าน

7. เมื่อการลงลายมือชื่อดิจิตอลเสร็จสมบูรณ์แล้ว คลิกบริเวณเอกสารที่ไม่ใช่ส่วนของลายมือชื่อดิจิตอลก็จะกลับสู่หน้าจอปกติของโปรแกรมไมโครซอฟท์เวิร์ด 97

8. ทดลองพิสูจน์เอกสารที่ไม่มีการเปลี่ยนแปลงโดยดับเบิลคลิกที่ลายมือชื่อดิจิตอล จะปรากฏเมนูของโปรแกรม ISAGSign

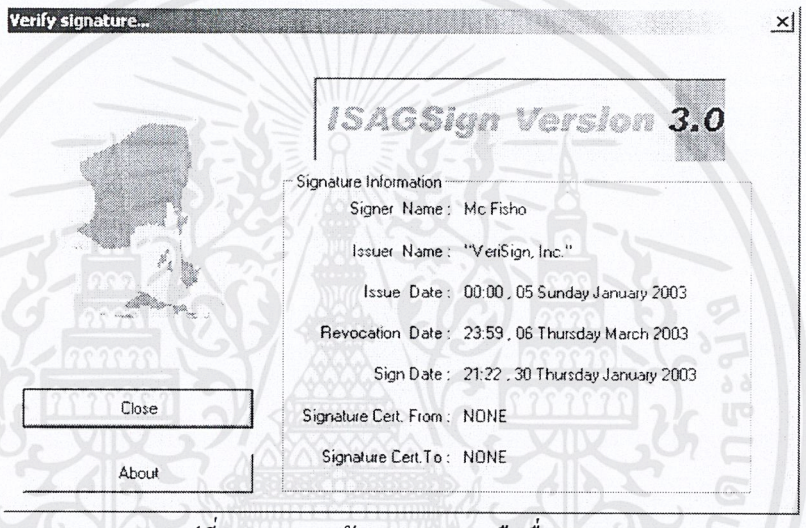
9. เลือกเมนู Approve > Verify Signature จะปรากฏข้อความว่า Unmodified แสดงว่าเอกสารไม่มีการเปลี่ยนแปลง

รูปที่ 9.10 การเลือกเมนู Verify Signature



รูปที่ 9.11 ข้อความแสดงว่าเอกสารไม่มีการเปลี่ยนแปลง

10. เมื่อคลิก OK จะปรากฏข้อมูลของลายมือชื่อดิจิตอลนี้

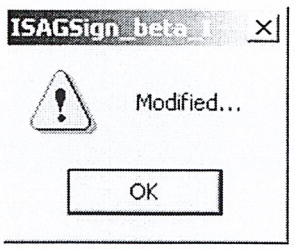


รูปที่ 9.12 แสดงข้อมูลของลายมือชื่อดิจิตอล

11. เมื่อคลิก Close จะพบว่าลายมือชื่อดิจิตอลจะเปลี่ยนเป็นเป็นคำว่า Unmodified Document

รูปที่ 9.13 สัญลักษณ์ของลายมือชื่อดิจิตอลแสดงการไม่เปลี่ยนแปลงของเอกสาร

12. ทดลองพิสูจน์เอกสารที่มีการเปลี่ยนแปลงโดยแก้ไขเอกสารบางส่วนแล้วทำตามขั้นตอนเดิม จะพบว่าลายมือชื่อดิจิตอลเปลี่ยนเป็นคำว่า Modified Document แสดงว่าเอกสารมีการเปลี่ยนแปลง

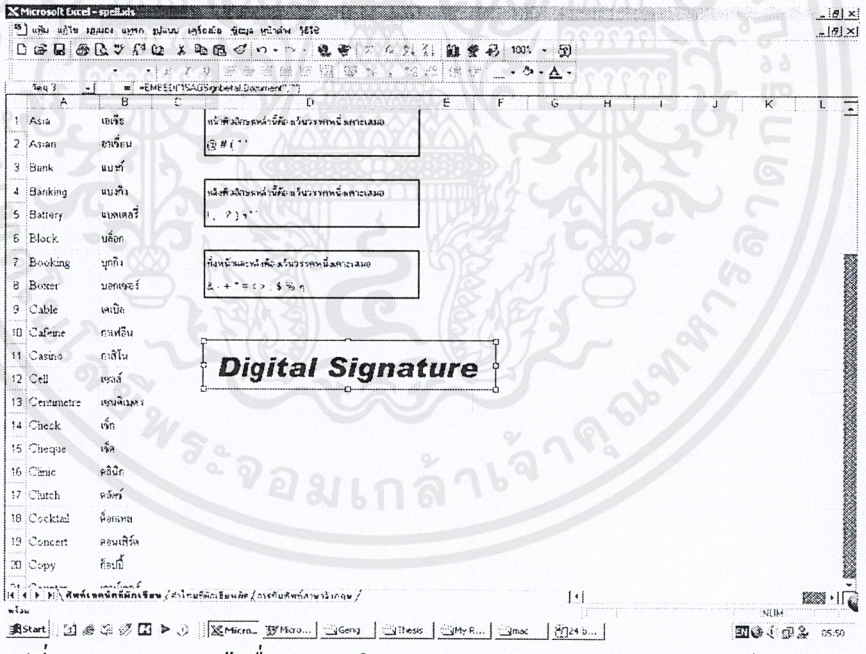


รูปที่ 9.14 ข้อความแสดงว่าเอกสารมีการเปลี่ยนแปลง

รูปที่ 9.15 สัญลักษณ์ของลายมือชื่อดิจิตอลแสดงการเปลี่ยนแปลงของเอกสาร

9.5 การทดสอบโปรแกรม ISAGSign กับไมโครซอฟท์เอ็กเซล 97

การทดลองกับไมโครซอฟท์เอ็กเซล 97 ทำตามขั้นตอนเช่นเดียวกับการทดลองกับไมโครซอฟท์เวิร์ด 97 ผลการทดลองที่ได้เหมือนกัน



รูปที่ 9.16 การลงลายมือชื่อดิจิตอลในเอกสารบนโปรแกรมไมโครซอฟท์เอ็กเซล 97

บทที่ 10
วิจารณ์และสรุป

10.1 บทวิจารณ์

จากโปรแกรมที่ได้พัฒนามาเพื่อใช้สำหรับการลงลายมือชื่อดิจิตอล ผลที่ได้ออกมาสามารถที่จะลงลายมือชื่อดิจิตอลได้ แต่ตัวโปรแกรมมีข้อจำกัดอยู่ที่การใช้งานได้ในโปรแกรมไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซลโดยมีข้อจำกัดอีกว่าใช้ได้กับข้อมูลประเภท CF_TEXT เท่านั้นไม่สามารถใช้กับข้อมูลประเภทอื่น ๆ ที่อยู่ในเอกสารได้ จากการทดสอบเมื่อลงลายมือชื่อดิจิตอลในเอกสารต้นฉบับ หลังจากนั้นตรวจสอบลายมือชื่อดิจิตอล จะได้ผลลัพธ์ว่าการตรวจสอบลายมือชื่อดิจิตอลนั้นถูกต้อง หลังจากได้ทดลองแก้ไขข้อมูลที่อยู่ในเอกสาร จากนั้นตรวจสอบลายมือชื่อดิจิตอล จะได้ว่าการตรวจสอบลายมือชื่อดิจิตอลผิดพลาด ซึ่งเป็นผลลัพธ์ที่ออกมาตรงตามทฤษฎีของลายมือชื่อดิจิตอล

10.2 แนวทางในการพัฒนาโปรแกรม

เนื่องจากโปรแกรมจะสามารถตรวจสอบได้เฉพาะในโปรแกรมไมโครซอฟท์เวิร์ดและไมโครซอฟท์เอ็กเซลอีกทั้งยังมีข้อจำกัดในเรื่องประเภทของข้อมูลที่นำมาสร้างลายมือชื่อดิจิตอลคือใช้กับข้อมูลประเภท CF_TEXT ได้อย่างเดียว ดังนั้นแนวทางในการพัฒนาต่อ จะต้องทำให้โปรแกรมสามารถตรวจสอบข้อมูลได้ในเอกสารที่หลากหลายรวมถึงประเภทของข้อมูลที่นำมาสร้างลายมือชื่อดิจิตอลโดยศึกษาข้อมูลประเภท RTF (Rich Text Format), ข้อมูลประเภท CF_BITMAP เป็นต้น

10.3 บทสรุป

การทำลายมือชื่อดิจิตอลนั้นมีความสำคัญกับการยืนยันบุคคล ถ้าเอกสารดิจิตอลใด ๆ ที่มีลายมือชื่อดิจิตอลอยู่ ผู้ที่เป็นเจ้าของลายมือชื่อดิจิตอลจะไม่สามารถปฏิเสธความรับผิดชอบต่อเอกสารนั้น ๆ ได้ และในด้านความถูกต้องของข้อมูลก็สามารถยืนยันได้โดยลายมือชื่อดิจิตอลเช่นกัน ว่าจะไม่ถูกเปลี่ยนแปลง หรือแก้ไขหลังจากการลงนามเรียบร้อยแล้ว สามารถนำไปใช้ในการทำธุรกรรมต่าง ๆ ผ่านคอมพิวเตอร์ เพราะเอกสารทางคอมพิวเตอร์นั้นเป็นสิ่งที่ทุกคนสามารถเข้าไปแก้ไขได้ ลายมือชื่อดิจิตอลจึงมีบทบาทสำคัญเป็นอย่างยิ่งต่อการพัฒนาการติดต่อสื่อสาร และการทำธุรกรรมในเครือข่ายคอมพิวเตอร์

บรรณานุกรม

[1] Dorothy E. Denning (1982) : "*Cryptography and Data Security*" , Massachusettesa, Addison-Wesley.

[2] Eugene Olafsen, Kenn Scribner, K. David White (1999) : "*MFC Programming with Visual C++ 6*" , Sams Publishing

[3] Vijay Ahuja : *"Network & Internet Security"* , AP Professional

[4] John Toohey : "*Using OLE 2.x in application development*" , Tndianapolis, IN

เว็บไซต์อ้างอิง

[1] http://www.microsoft.com

[2] http://www.codeproject.com

[3] http://www.codeguru.com

[4] http://www.rsa.com

[5] http://www.verisign.com