โปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตร

Public Key Infrastructure and Biometric Application

นายวิภาส สุตันตยาวลี

นายวิวัฒน์ คงศิริวัฒนกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2549

โปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตร

Public Key Infrastructure and Biometric Application

โดย

นายวิภาส สุตันตยาวลี

นายวิวัฒน์ ดงศิริวัฒนกุล

อาจารย์ที่ปรึกษา

อ. อัครเดช วัชระภูพงษ์

ผศ. ธนา หงษ์สุวรรณ

อ. ธนัญชัย ตรีภาค

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2549

ปริญญานิพนธ์ปีการศึกษา 2549

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง โปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตร

Public Key Infrastructure and Biometric Application

ผู้จัดทำ

1. นายวิภาส สุตันตยาวลี รหัสนักศึกษา 46010716

2. นายวิวัฒน์ คงศิริวัฒนกุล รหัสนักศึกษา 46010725

_____อาจารย์ที่ปรึกษา

(อ. อัครเดช วัชระภูพงษ์)

_____อาจารย์ที่ปรึกษา

(ผศ. ธนา หงษ์สุวรรณ)

_____อาจารย์ที่ปรึกษา

(อ. ธนัญชัย ตรีภาค)

โปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตร

นายวิภาส สุตันตยาวลี	46010716
นายวิวัฒน์ คงศิริวัฒนกุล	46010725
อ. อัครเดช วัชระภูพงษ์	อาจารย์ที่ปรึกษา
ผศ. ธนา หงษ์สุวรรณ	อาจารย์ที่ปรึกษา
อ. ธนัญชัย ตรีภาค	อาจารย์ที่ปรึกษา
ปีการศึกษา 2549	

บทคัดย่อ

โครงงานนี้มุ่งเน้นไปที่การพัฒนาโปรแกรมรับส่งสารด่วนให้มีความปลอดภัยมากยิ่งขึ้น โดยใช้เทคโนโลยีโครงสร้างพื้นฐานกุญแจสาธารณะ และวิธีทางชีวมาตรประยุกต์ใช้ร่วมกัน โดยถือว่าเป็นการพัฒนาต้นแบบการใช้ชีวมาตรกับโปรแกรมประยุกต์ที่มีผู้ใช้งานเป็นจำนวนมาก กล่าวคือ ในการเข้าสู่ระบบรับส่งสารด่วน โปรแกรมสามารถพิสูจน์ตัวตนผู้ใช้ได้จากการอ่านลายนิ้วมือ (Fingerprint) ที่ถูกเก็บไว้ที่เครื่องแม่ข่ายที่ให้บริการไดเรกทอรีเซอร์วิซไว้ ก่อนการเข้าใช้งานระบบ และเข้ารหัสลับข้อมูลข่าวสารก่อนการส่งให้คู่สนทนา บนช่องทางที่ปลอดภัยโดยใช้โปรโตคอล SSL ร่วมกับการประยุกต์ใช้โครงสร้างพื้นฐานกุญแจสาธารณะที่ให้บริการด้านการรักษาความปลอดภัยของข้อมูลได้อย่างครบถ้วน

ABSTRACT

This project has focused on developing an instant messaging program to be more secure by using "Public Key Infrastructure" and "Biometric" technologies. This project can be considered as a prototype of using biometric with an application that many people use it. More specifically, in sign-in process, a plug-in that we developed can do two basic things; authentication and authorization by retrieving fingerprint information from server that provide directory service. After user has signed-in, another plug-in will take control in encrypting / decrypting message that sent or received by user. This plug-in uses a secure channel, which is SSL and combines with public key infrastructure that provides a complete security.

กิตติกรรมประกาศ

โครงงานโปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตร สำเร็จลุล่วงได้ด้วยดี เนื่องจากคำแนะนำ และคำปรึกษาจาก อ.อัครเดช วัชระภูพงษ์ ผศ.ธนา หงษ์สุวรรณ และ อ.ธนัญชัย ดรีภาค ข้าพเจ้ารู้สึกทราบซึ้งในความอนุเคราะห์จากท่านอาจารย์ทั้งสามท่าน และขอขอบพระคุณเป็นอย่างสูง

ขอขอบคุณห้องปฏิบัติการ ISAG ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ ที่ได้สนับสนุนสถานที่และอุปกรณ์เครือข่ายสำหรับใช้ในการพัฒนาโครงงาน

ขอขอบคุณคณาจารย์ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุก ๆ ท่านที่ได้ประสิทธิ์ประสาทวิชาให้กับข้าพเจ้า

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ในภาควิชาวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกคนที่ให้คำแนะนำต่างๆ และคอยให้กำลังใจเสมอมา

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณบิดา มารดา และครอบครัวของข้าพเจ้าที่เป็นกำลังใจและให้การสนับสนุนในทุกเรื่องๆ ทำให้ข้าพเจ้าสามารถทำปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงด้วยดี

คุณค่าและประโยชน์อันพึงมาจากปริญญานิพนธ์ฉบับนี้ ข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่าน

วิภาส สุดันตยาวลี

วิวัฒน์ คงศิริวัฒนกุล

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 2 ทฤษฎีที่เกี่ยวข้อง

สารบัญภาพ (ต่อ)

หน้า

บทที่ 1
บทนำ

1.1 บทนำ

เนื่องด้วยการติดต่อสื่อสารผ่านโปรแกรมรับส่งสารด่วนในปัจจุบันนั้นสามารถทำได้อย่างง่ายดาย ทั้งนี้เนื่องมาจาก ความสะดวกในการใช้งานของตัวโปรแกรมเอง ความรวดเร็วในการติดต่อสื่อสาร ไม่มีข้อจำกัดในเรื่องสถานที่ สามารถสื่อสารได้กับผู้คนทั่วทุกมุมโลก ทำให้ผู้คนทั่วโลกสนใจใช้โปรแกรมประเภทนี้ในการติดต่อสื่อสารระหว่างกันมากขึ้น แต่ทว่าโปรแกรมรับส่งสารด่วนที่ใช้งานในปัจจุบันนั้นยังมีปัญหาด้านการรักษาความปลอดภัยของข้อมูลอยู่มาก โดยเฉพาะการลักลอบอ่าน / เปลี่ยนแปลง / แก้ไข ข้อมูลในระหว่างทางทำให้ข้อมูลผิดไปจากความเป็นจริง อีกทั้ง การแอบอ้างเป็นผู้ใช้ ดังนั้นโครงการนี้จึงมุ่งพัฒนาให้โปรแกรมรับส่งสารด่วนนั้นมีความปลอดภัยในการใช้งานและติดต่อสื่อสารมากขึ้น

โครงการโปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตรนี้สามารถแบ่งการทำงานออกเป็นสามส่วนคือ ส่วนที่ทำหน้าที่ในการพิสูจน์ตัวตนผู้ใช้ด้วยลายนิ้วมือ โดยจะทำการอ่านลายนิ้วมือของผู้ใช้ ณ ตอนที่ผู้ใช้ต้องการ Sign-in แล้วนำลายนิ้วมือที่เก็บไว้ใน Directory Service มาเปรียบเทียบ ถ้าถูกต้องถึงจะอนุญาตให้เข้าใช้งาน ส่วนที่ทำหน้าที่ในการเข้ารหัส/ถอดรหัสข้อความ ที่จะทำหน้าที่เข้ารหัสลับ/ถอดรหัสลับข้อความที่ผู้ใช้แต่ละคนส่งหากันเพื่อป้องกันบุคคลที่สามลักลอบแอบดูข้อมูล และส่วนที่ทำหน้าที่เป็นเครื่องแม่ข่าย ที่ทำหน้าที่เป็นทั้ง Certificate Authorities (CA) และ Directory Service โดยใช้โพรโตคอล LDAP

สิ่งที่ถือว่าเป็นจุดเด่นของโครงการนี้ก็คือการนำชีวมาตรเข้ามาใช้งานกับโปรแกรมประยุกต์ทั่วไป ซึ่งถือได้ว่าเป็นก้าวแรกของการใช้งานชีวมาตรร่วมกับโปรแกรมประยุกต์พื้นฐานที่มีผู้ใช้งานเป็นจำนวนมาก ต่างจากปัจจุบันที่การใช้กระบวนการทางชีวมาตรเป็นการใช้งานที่จำกัดอยู่ในวงแคบเฉพาะกลุ่มเท่านั้น โดยโครงการนี้ได้นำกระบวนการพิสูจน์ตนโดยใช้ลายนิ้วมือ มาพัฒนาให้สามารถใช้งานร่วมกับโปรแกรม GAIM ในลักษณะเป็นส่วนขยายเพิ่มเติมจากโปรแกรมประยุกต์หลัก

1.2. วัตถุประสงค์

1.2.1 เพื่อปรับปรุงระบบรับส่งสารด่วนให้มีความปลอดภัยยิ่งขึ้น

1.2.2 เพื่อสร้างโปรแกรมที่สามารถเข้าและถอดรหัสลับข้อความ เพื่อทำให้ผู้ใช้มั่นใจได้ว่า ข้อความที่ส่งถูกปกปิดเป็นความลับ

1.2.3 เพื่อสร้างโปรแกรมที่สามารถใช้ระบบพิสูจน์ตัวตนทางชีวมาตร เพื่อทำให้ผู้ใช้แน่ใจได้ว่าบุคคลอื่นไม่สามารถใช้งานบัญชี (User Account) ที่ไม่ใช่ของตัวเองได้

1.2.4 เพื่อสร้างการเชื่อมต่อระหว่างโปรแกรมฝั่งเซิร์ฟเวอร์และฝั่งไคลเอ็นต์โดยสร้างเป็นส่วนขยายของโปรแกรมGaim โดยยังคงความสามารถตามข้อ 1.2.3 และ1.2.4

1.3. ประโยชน์ที่คาดว่าจะได้รับ

1.3.1 การปกปิดความลับของข้อมูลด้วยการเข้ารหัสลับด้วยโครงสร้างกุญแจพื้นฐานสาธารณะนั้น สามารถช่วยป้องกันไม่ให้บุคคลที่ 3 สามารถล่วงรู้ / เปลี่ยนแปลง / แก้ไข ข้อมูลของเราได้ โดยผู้ที่จะสามารถล่วงรู้ข้อมูลได้นั้น มีเพียงเราและผู้ที่เราต้องการติดต่อด้วยเท่านั้น

1.3.2 การนำทฤษฎีใบรับรองสิทธิ์มาใช้ในโครงการ ช่วยให้ผู้ใช้งานที่ติดต่อกันสามารถมั่นใจได้ว่า เครื่องคอมพิวเตอร์ที่ติดต่อด้วยนั้น สามารถเชื่อถือได้ไม่ได้มีการแอบอ้างหรือปลอมแปลงมา อีกทั้งยังใช้เป็นหลักฐานตรวจสอบในกรณีที่เกิดการปฏิเสธความรับผิดชอบได้

1.3.3 การนำเทคโนโลยีชีวมาตรเข้าประยุกต์กับการใช้รหัสผ่าน ทำให้การรักษาความปลอดภัยในการพิสูจน์ตนเพิ่มสูงขึ้น กล่าวคือ ผู้ไม่ประสงค์ดีแทบจะไม่สามารถเข้าใช้งานได้ แม้ว่าจะทราบรหัสผ่านของผู้ใช้ เพราะต้องใช้องค์ประกอบทางกายภาพของบุคคลซึ่งมีความเป็นเอกลักษณ์แตกต่างกันไปเข้าร่วมพิสูจน์ตนด้วย ยกตัวอย่างเช่น ลายนิ้วมือของแต่ละบุคคลจะไม่มีโอกาสซ้ำกันได้ เป็นต้น

1.3.4 สามารถนำโครงการโปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตร ไปพัฒนาต่อยอดและนำการพิสูจน์ตนด้วยชีวมาตรไปใช้จริงในอนาคตได้

1.4. ขอบเขตของโครงงาน

1.4.1 ได้ส่วนขยายของโปรแกรม GAIM (IsagQ) ที่สามารถเข้ารหัสลับและถอดรหัสลับข้อความที่กำลังสนทนา

1.4.2 ได้ส่วนขยายของโปรแกรม GAIM (AnubisQ) ที่สามารถใช้ในการพิสูจน์ตนด้วยการอ่านลายนิ้วมือของผู้ใช้

1.4.3 ได้โปรแกรมแม่ข่าย ที่สามารถตอบสนองต่อส่วนขยาย IsagQ และ AnubisQ

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1. บทนำ

โครงงานโปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตร มุ่งเน้นตอบสนองปัจจัยหลัก 2 ประการคือ การพิสูจน์ตนด้วยชีวมาตร และการรับส่งข้อมูลให้ปลอดภัย ซึ่งการพิสูจน์ตนด้วยชีวมาตรนั้นจะใช้การพิสูจน์ตนด้วยลายนิ้วมือ โดยมีทฤษฎีบทที่เกี่ยวข้องคือ ทฤษฎีลายนิ้วมือ วิธีการพัฒนาส่วนขยายของโปรแกรม Gaim และการใช้งานไดเรกทอรีเซอร์วิซ (OpenLDAP)

สำหรับการส่งข้อมูลให้ปลอดภัยบนเครือข่ายนั้นจะใช้การส่งแบบ SSL ร่วมกับการใช้ใบรับรองสิทธิ์ซึ่งมีทฤษฎีบทที่เกี่ยวข้อง คือ ทฤษฎีโครงสร้างพื้นฐานกุญแจสาธารณะ ทฤษฎีที่เกี่ยวข้องกับ SSL และทฤษฎีที่เกี่ยวข้องกับใบรับรองสิทธิ์ รวมถึงผู้ให้บริการใบรับรองสิทธิ์

2.2. ทฤษฎีลายนิ้วมือ

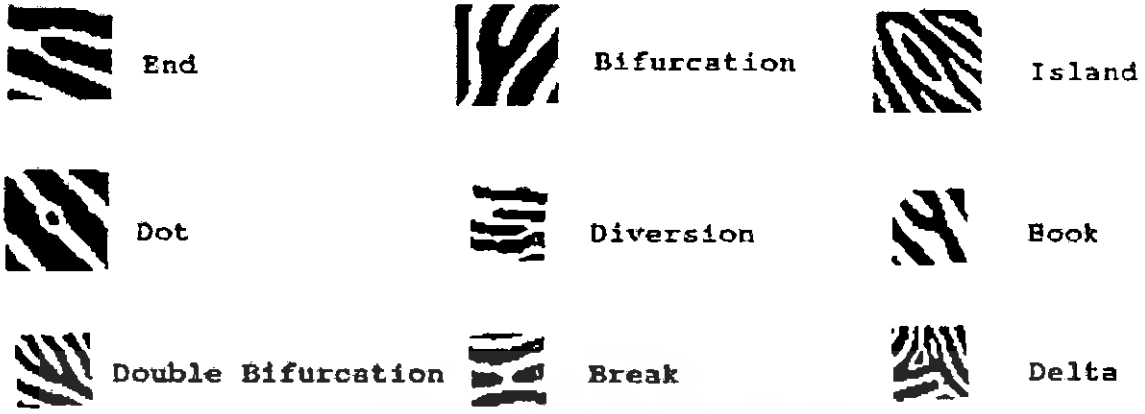
2.2.1 ความรู้เบื้องต้นของลายนิ้วมือ

บริเวณปลายนิ้วมือของมนุษย์โดยทั่วไปจะเห็นลายนิ้วมือที่มีลักษณะประกอบไปด้วยเส้นสองลักษณะคือ เส้นสันเขา (Ridges) และเส้นหุบเขา (Valleys) ซึ่งทั้งสองลักษณะจะอยู่สลับกันไปตลอด

2.2.1.1 จุดลักษณะสำคัญบนลายนิ้วมือ (Characteristics)

คือ ตำหนิต่างๆบนลายนิ้วมือ สามารถแบ่งได้เป็นสองลักษณะดังนี้

1. ตำหนิและลักษณะต่างๆของลายเส้นทั่วๆไปเช่น เส้นตรง เส้นโค้ง จุด เส้นแตก เส้นวกกลับเส้นเวียน เส้นขาด เส้นทะเลสาป และเส้นสองเส้นมาพบกัน (เส้นหักมุม)
2. ลักษณะพิเศษบางอย่าง เช่น
   - ไบเฟอร์เคชั่น คือ เส้นขอบหนึ่งที่แยกออกเป็นสองเส้นหรือมากกว่าสองเส้น
   - ไดเวอร์เจ้น คือ เส้นขอบที่วิ่งขนานกันมาหรือเกือบจะขนาน และแยกห่างออกไป
   - จุดมินูเทีย (minutiae) คือ จุดบนเส้นหยุดหรือเส้นแยก



รูปที่ 2-1 แสดงจุดลักษณะสำคัญบนลายนิ้วมือ

2.2.1.2 คำจำกัดความที่สำคัญบนลายนิ้วมือ

เป็นการอธิบายคุณลักษณะสำคัญที่ต้องศึกษาและทำความเข้าใจเพราะมีคุณประโยชน์ที่แสดงให้เห็นถึงความแตกต่างของแต่ละลายนิ้วมือซึ่งมีอยู่ 4 ข้อ ได้แก่

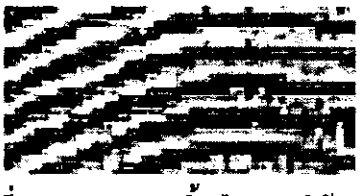
- เส้นขอบ (Type Line) คือ เส้นคู่ขนานคู่ในสุด ซึ่งคู่กันมาแล้วแยกตัวออกจากกันเพื่อจะโอบล้อมหรือพยายามโอบล้อมบริเวณลายนิ้วมือที่อยู่ภายใน
- สันดอน (Delta) คือ ลายเส้นในลายนิ้วมือซึ่งอยู่ตรงหน้าและใกล้ที่สุดกับกึ่งกลางของปากทางแยกของเส้นขอบ
- จุดใจกลาง (Core) คือ จุดใดจุดหนึ่งบนปลายเส้นหรือบนบ่าหรือไหล่ของเส้นวกกลับรูปในสุด และต้องอยู่ภายในของลายนิ้วมือ
- บริเวณลายนิ้วมือที่อยู่ภายใน (Pattern Area) คือ พื้นที่บริเวณภายในของลายนิ้วมือที่ถูกเส้นขอบโอบล้อม

2.2.1.3 รูปแบบของลายนิ้วมือ

2.2.1.3.1 เส้นโค้ง (Arch)

1. โค้งราบ (Plain Arch = PA)

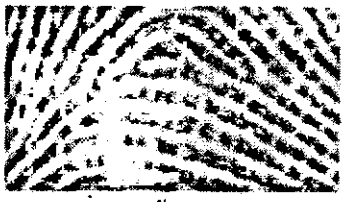
ลายเส้นวิ่งหรือไหลออกไปข้างหนึ่ง ไม่เกิดมุมแหลมหรือพุ่งขึ้นตรงกลาง



รูปที่ 2-2 แสดงลายนิ้วมือแบบโค้งราบ

2. โค้งกระโจม (Tented Arch = TA)

ลายเส้นตรงกลางเกิดเป็นลายเส้นพุ่งขึ้นจากแนวนอนเป็นมุมแหลมหรือมุมฉาก

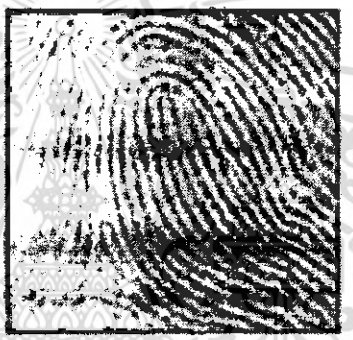


รูปที่ 2-3 แสดงลายนิ้วมือแบบโค้งกระโจม

2.2.1.3.2 ลูปหรือมัดหวาย (Loop)

1. มัดหวายปัดขวา (Right Slant Loop = RSL)

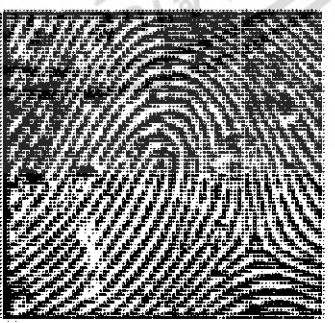
   มีสันดอนเพียงจุดเดียว มีเส้นวกหลักที่สมบูรณ์อย่างน้อยหนึ่งเส้น มีทิศทางไปด้านขวา



รูปที่ 2-4 แสดงลายนิ้วมือแบบมัดหวายปัดขวา

2. มัดหวายปัดซ้าย (Left Slant Loop = LSL)

   มีสันดอนเพียงจุดเดียว มีเส้นวกหลักที่สมบูรณ์อย่างน้อยหนึ่งเส้นมีทิศทางไปด้านซ้าย



รูปที่ 2-5 แสดงลายนิ้วมือแบบมัดหวายปัดซ้าย

3. มัดหวายคู่หรือมัดหวายแฝด (Double Loop = DL)

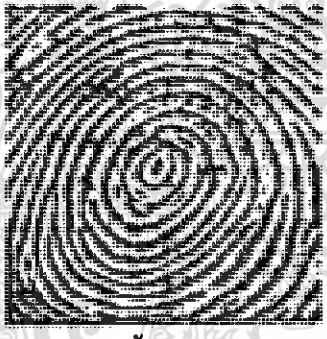
มีลักษณะคล้ายกับลายนิ้วมือแบบมัดหวายข้างบนแต่มากอดหรือกล้ำกันจนเกิดมีสันดอนสองจุด โดยไม่จำเป็นต้องมีขนาดเท่ากัน ประกอบด้วย

รูปที่ 2-6 แสดงลายนิ้วมือแบบมัดหวายคู่หรือมัดหวายแฝด

2.2.1.3.3 ก้นหอย (Whorl)

ลายนิ้วมือที่มีเส้นเวียนรอบเป็นวงจร ลักษณะเหมือนลานนาฬิกา รูปไข่ วงกลม ประกอบด้วย

1. ก้นหอยธรรมดา (Plain Whorl = W)



รูปที่ 2-7 แสดงลายนิ้วมือแบบก้นหอยธรรมดา

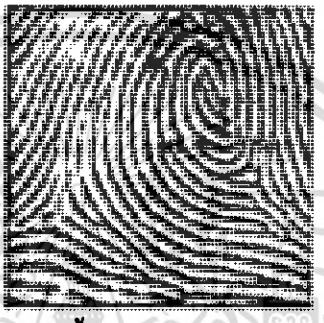
2. ก้นหอยกระเป๋ากลางปัดขวา (Right Central Pocket = RCP)

รูปที่ 2-8 แสดงลายนิ้วมือแบบก้นหอยกระเป๋ากลางปัดขวา

3. ก้นหอยกระเป๋ากลางปัดซ้าย (Left Central Pocket = LCP)

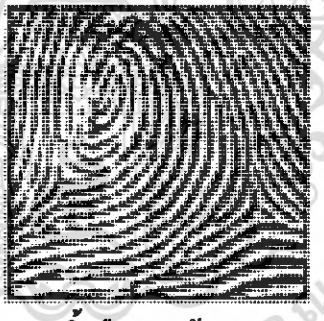
รูปที่ 2-9 แสดงลายนิ้วมือแบบก้นหอยกระเป๋ากลางปัดซ้าย

4. ก้นหอยกระเป๋าข้างปัดขวา (Right Lateral Pocket = RLP)



รูปที่ 2-10 แสดงลายนิ้วมือแบบก้นหอยกระเป๋าข้างปัดขวา

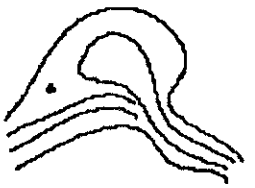
5. ก้นหอยกระเป๋าข้างปัดซ้าย (Left Lateral Pocket = LLP)



รูปที่ 2-11 แสดงลายนิ้วมือแบบก้นหอยกระเป๋าข้างปัดซ้าย

2.2.1.3.4 ซับซ้อน (Accidental Whorl = AW)

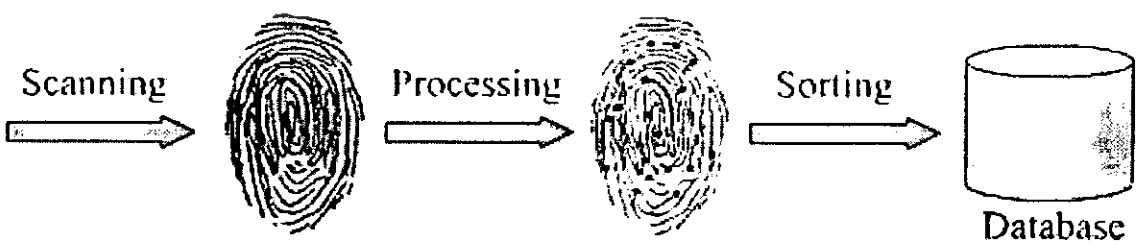
ลายนิ้วมือที่มีลักษณะพิเศษที่ไม่จัดเข้าเป็นลายนิ้วมือชนิดใดโดยเฉพาะ ประกอบด้วยลายนิ้วมือ สองแบบมาผสมกัน และมีสันดอนสองสันดอนหรือมากกว่า เช่น กรณีที่ไม่สามารถเข้ากับลายนิ้วมือกลุ่ม ที่กล่าวมาข้างต้นไม่ได้เลย โดยมีความยุ่งเหยิงและเป็นรูปแบบที่ไม่แน่นอน



รูปที่ 2-12 แสดงลายนิ้วมือแบบซับซ้อน

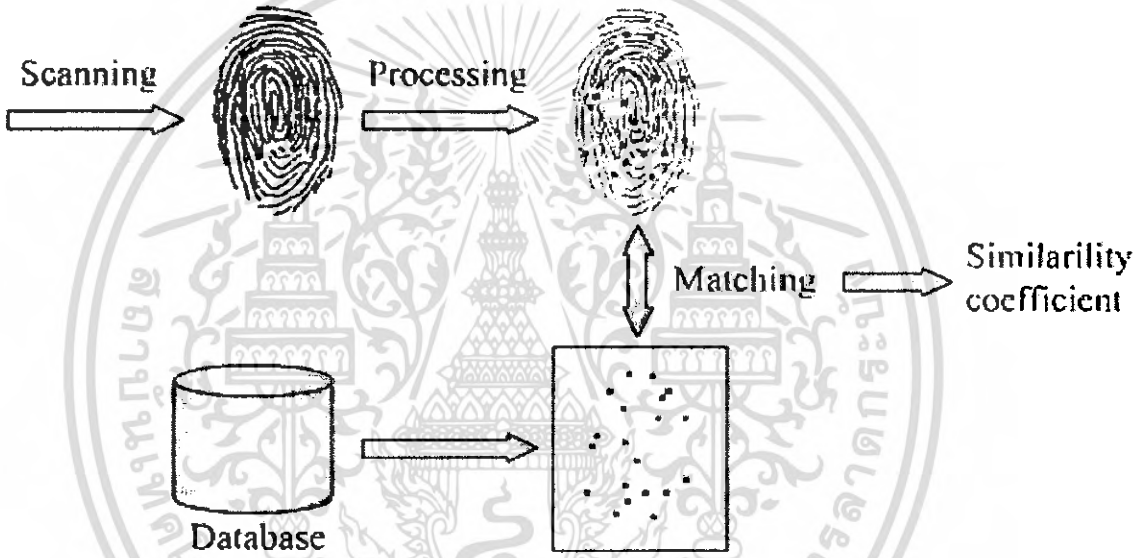
2.2.2 หลักการวิเคราะห์ลายนิ้วมือ

การวิเคราะห์ลายนิ้วมือของบุคคลโดยทั่วไปนั้น จะเริ่มด้วยการนำลายนิ้วมือของแต่ละบุคคลแต่ละนิ้วมาหาจุดลักษณะเฉพาะที่สำคัญกระบวนการแรกเริ่มของการตรวจพิสูจน์ลายนิ้วมือคือ ก ารอ่านภาพลายนิ้วมือเข้ามาเก็บไว้ในหน่วยความถาวร โ ดยข้อมูลที่อ่านหรือสแกนเข้ามานั้นจะนำมาผ่านการประมวลผลก่อนแล้วจึงเก็บข้อมูลนั้นไว้ ซึ่งข้อมูลนี้จะถูกเก็บไว้เป็นต้นแบบหรือรหัสของผู้ใช้แต่ละคนในขั้นตอนก่อนที่จะนำลายนิ้วมือเข้าไปเก็บนั้นจะต้องผ่านขั้นตอนของการประมวลผลก่อน ในกระบวนการนี้จะทำให้ภาพที่ได้รับการสแกนเข้ามาเกิดความสมบูรณ์มากขึ้นเพราะเมื่อเครื่องได้รับการสแกนภาพเข้ามาแล้ว ภาพที่อ่านได้อาจไม่ชัดเจน พร่าเลือน ก็จะทำให้การประมวลผลในขั้นตอนถัดไปทำได้ด้วยความยากลำบากหรือทำไม่ได้ ซึ่งจะทำให้ผลที่ได้ก็อาจไม่ถูกต้องตามที่ควรจะเป็น เมื่อเกิดปัญหาเช่นนี้ในกระบวนการนี้จึงได้มีการกระทำหลายกระบวนการด้วยกันคือ การกำจัดสัญญาณรบกวน การปรับความมืดสว่างและความแตกต่างของตัวภาพและฉากของภาพ การแปลงภาพเป็นภาพสองระดับ(Binary) การทำให้เส้นลายนิ้วมือบาง (Thinning) การปรับภาพหลังจากแปลงเป็นภาพสองระดับ การหาค่า Threshold ของการปรับภาพเป็นสองระดับและอื่นๆอีกมาก ซึ่งกระบวนการจะมากหรือน้อยขึ้นอยู่กับกับตัวอุปกรณ์นั้นมีการอ่านค่าลายนิ้วมือที่ได้ภาพละเอียดและสมบูรณ์แค่ไหนเมื่อได้ลายนิ้วมือที่ผ่านการประมวลผลแล้ว ก็นำข้อมูลหรือภาพนี้ไปจัดเก็บในหน่วยความจำถาวร ซึ่งสามารถลบข้อมูลใหม่ด้วยไฟฟ้า โดยภาพที่ถูกจัดเก็บไว้นี้จะถูกเก็บไว้เพื่อใช้ในการเปรียบเทียบกับลายนิ้วมือที่ได้รับการสแกนเข้ามาเมื่อนำตัวอุปกรณ์นี้ไปใช้งาน



รูปที่ 2-13 แสดงกระบวนการทำงานของการวิเคราะห์ลายนิ้วมือ

จากรูปที่ 2-13 เริ่มด้วยการสแกนลายนิ้วมือเข้ามาแล้วนำภาพที่ได้ผ่านการประมวลผลซึ่งจะได้ภาพที่มีประสิทธิภาพมากขึ้นแล้วจึงเก็บภาพนั้นไว้ หลังจากเก็บภาพไว้แล้วนั้นก็มาถึงขั้นตอนการนำไปใช้งาน เมื่อตัวอุปกรณ์ได้ถูกบันทึกหรือเก็บลายนิ้วมือของผู้ที่จะนำไปใช้แล้ว ขั้นตอนในการใช้ก็จะคล้ายกับตอนอ่านลายนิ้วมือเข้ามาเก็บไว้ เพียงแต่การอ่านเข้ามาครั้งนี้ข้อมูลที่ได้จะถูกนำเก็บไว้ที่หน่วยความจำชั่วคราว ซึ่งหลังจากสแกนเข้ามาแล้วประมวลผลแล้วก็จะทำการเก็บข้อมูลไว้ท่ส่วนของหน่วยความชั่วคราวถัดไปก็จะนำข้อมูลที่เก็บอยู่ในส่วนของหน่วยความจำถาวร กับส่วนที่เก็บอยู่ในหน่วยความจำชั่วคราวนั้นมาทำการเปรียบเทียบกัน(Matching) เมื่อได้ผลแล้วก็จะแจ้งผลให้ผู้ใช้ทราบว่ามีความเหมือนกันมากน้อยแค่ไหน



รูปที่ 2-14 แสดงกระบวนการเปรียบเทียบลายนิ้วมือ

จากรูป 2-14 แสดงให้เห็นถึงกระบวนการเปรียบเทียบลายนิ้วมือที่ได้รับการสแกนเข้ามา โดยเริ่มที่การสแกนภาพเข้ามา แล้วทำการประมวลขั้นตอนเดียวกันกับการจัดเก็บตอนแรกแล้วนำภาพที่เก็บไว้ในตอนแรกมาเปรียบเทียบกับภาพที่สแกนเข้ามา ณ ตอนนั้น เพื่อเปรียบเทียบว่ามีความเหมือนหรือแตกต่างมากเพียงใด

2.2.3 หลักการเปรียบเทียบลายนิ้วมือ

จากทฤษฎี Automated Fingerprint Identification System (AFIS) มีหลักการคือ ระบบAFIS จะตรวจสอบและค้นหา "จุดสำคัญ" บนลายนิ้วมือ และหา "ความสัมพันธ์" ระหว่างจุดต่างๆ เหล่านั้น

รูปที่ 2-15 แสดงกระบวนการวิเคราะห์ลายนิ้วมือระบบ AFIS

วิธีการเปรียบเทียบนี้เริ่มจากการรับข้อมูลลายนิ้วมือจากอุปกรณ์สแกนลายนิ้วมือโดย รูปที่ได้จากอุปกรณ์สแกนลายนิ้วมือจะประกอบไปด้วยอัตราของสีที่ไม่สม่ำเสมอ ดังนั้นก่อนส่งรูปภาพนี้ไปเก็บในไลบรารีของการพิสูจน์ตนจะต้องทำการกรองรูปภาพ (Filter) ก่อน

รูปที่ 2-16 แสดงลายนิ้วมือที่ได้จากอุปกรณ์สแกนลายนิ้วมือ

การกรองรูปภาพที่ได้รับมาจากการสแกนลายนิ้วมือนั้นเป็นการทำให้รูปภาพมีขอบเขตของสีอยู่ระหว่างช่วงสีดำ – สีขาว (0-255) โดยจะทำให้รูปภาพที่มีสีเทาเข้มกลายเป็นสีดำ และสีเทาอ่อนกลายเป็นสีขาว

กระบวนการกรองรูปภาพมีรายละเอียดดังนี้

ถ้าเป็นสีขาวหรือสีเทาอ่อน จะพิจารณาให้เป็นขอบนอกของลายนิ้วมือ และแปลงเป็นสีขาวจำนวนสีที่เกิดขึ้นในแต่ละโทนสีเทาในลายนิ้วมือจะถูกบันทึกไว้ โดยสีเทาที่เข้มที่สุดถูกพิจารณาให้เป็นเสมือนสันเขา (Ridges) และสีเทาที่อ่อนที่สุดถูกพิจารณาให้เป็นเสมือนหุบเขา (Valleys) ความแตกต่างระหว่างสันเขาและหุบเขาถูกคำนวนและแบ่งครึ่ง โดยค่าของสีเทาที่เข้มมากกว่าสีดำ (0) จนถึงค่าที่น้อยกว่าสีเทาที่เข้มที่สุดที่เป็นแนวสันเขาบวกกับค่าความแตกต่างจะต้องเปลี่ยนเป็นสีดำทั้งหมด แ ละสีเทาที่เข้มกว่าแนวหุบเขาจนถึงค่าที่เทาอ่อนบวกกับค่าความแตกต่างจะต้องเปลี่ยนเป็นสีขาวดังสมการ

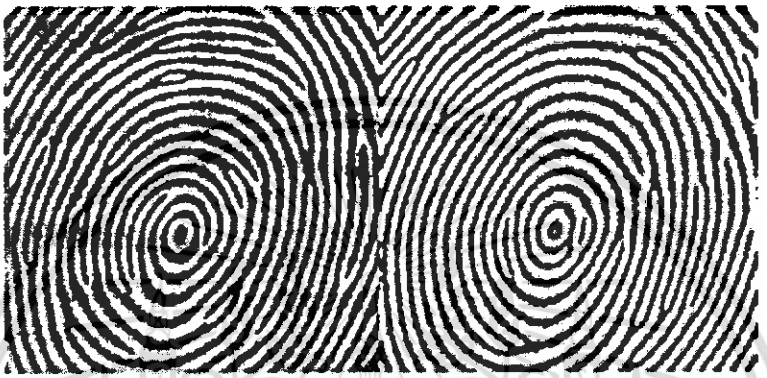
1 = สีเทาที่เข้มมากที่สุด

d = สีเทาที่เข้มน้อยที่สุด

x = (1 - d) / 2

Ridges = 0 <= 1 <= (1 + x)

Valleys = (1 + x) < d <= 255



รูปที่ 2-17 แสดงลายนิ้วมือก่อนและหลังทำการกรอง

หลังจากได้รูปภาพที่ผ่านกระบวนการกรองออกมาแล้ว ก็นำรูปภาพนั้นไปทำการเปรียบเทียบและวิเคราะห์ โดยการเปรียบเทียบจะพิจาณาในส่วนของสันเขาและหุบเขา การค้นหานั้นเริ่มต้นด้วยการเลือกจุดเริ่มต้น และทำการพิจารณาไปตามแนวสันเขาจนกระทั่งพบจุดปลายของรูปแบบลายนิ้วมือแบบBifurcation ก็จะทำเครื่องหมายเอาไว้ว่าอยู่ในตำแหน่งพิกัด x, y ที่เท่าไหร่ โดยกระบวนเช่นนี้จะทำไปเรื่อยๆจนกระทั่งหมดทั้งรูปภาพ จากนั้นพิจารณาพิกัด x, y ที่ได้ออกมาเพื่อนำไปเปรียบเทียบกับข้อมูลที่เก็บไว้ว่าถูกต้องตรงกันหรือไม่ ในการเปรียบเทียบนั้นจะทำการหมุนภาพลายนิ้วมือที่ได้มาใหม่จนกระทั่งพบว่ามันถูกต้องตรงกันหรือเกิดความล้มเหลวในการเปรียบเทียบ โดยอัตราในการเปรียบเทียบนั้นจะเปรียบเทียบภายใน 30 พิกเซลต้องมีอย่างน้อย 20 พิกเซลขึ้นไปที่ถูกต้องตรงกันจึงสามารถบอกได้ว่าถูกต้อง (โดยสามารถตั้งค่าไว้ได้ว่าต้องการความละเอียดในการตรวจสอบมากน้อยแค่ไหน และขึ้นอยู่กับคุณภาพของอุปกรณ์สแกนลายนิ้วมืออีกด้วย)

2.3. การพัฒนาส่วนขยายของโปรแกรม Gaim

เราสามารถพัฒนาส่วนขยายของโปรแกรม Gaim ด้วยภาษา C ,ภาษา Perl หรือ ภาษา Tcl ก็ได้ แต่เนื่องจากผู้พัฒนาโปรแกรม Gaim ใช้ภาษา C เป็นหลัก ในโครงงานนี้เราจึงใช้ภาษา C ในการพัฒนาส่วนขยายของโปรแกรม Gaim

โครงสร้างส่วนขยายของโปรแกรม Gaim

ในไฟล์หลักของส่วนขยายนั้นจะมีรูปแบบที่ตายตัวและจำเป็นที่จะต้องใช้ตามรูปแบบเพื่อที่จะสามารถใช้ส่วนขยายได้ ในที่นี้เราจะเริ่มศึกษาส่วนขยายของโปรแกรม Gaim ด้วยโปรแกรม helloworld.c ซึ่งเป็นส่วนขยายที่จะแสดงกล่องข้อความ เมื่อเราเปิดโปรแกรม Gaim ขึ้นมาดัง source code ด้านล่าง

ส่วนแรกเป็นการ Include file และ #define GAIM_PLUGINS

```
#define GAIM_PLUGINS                    // ต้องประกาศไว้ในไฟล์ส่วนขยายทุกไฟล์
#include <glib.h>
#include "notify.h"
#include "plugin.h"
#include "version.h"

static gboolean
plugin_load(GaimPlugin *plugin) {
  gaim_notify_message(plugin, GAIM_NOTIFY_MSG_INFO, "Hello  World!", "This is the Hello
World! plugin :)", NULL, NULL, NULL);
  return TRUE;
}
```

// เมื่อส่วนขยายถูกเรียกใช้งาน จะเรียกฟังก์ชัน plugin_load ก่อน ทำให้เราสามารถกำหนดค่าเริ่มต้นต่างๆ ให้กับส่วนขยายของเราได้ ในที่นี้เราจะเรียกฟังก์ชัน gaim_notify_message เพื่อแสดงกล่องข้อความ "Hello World!"

ส่วนที่สองคือการตั้งค่าต่างๆ ของส่วยขยาย

```
static GaimPluginInfo info = {   //struct info เป็นส่วนที่ระบุรายละเอียดต่างๆ ของส่วนขยายของเรา
(จำเป็นต้องมีในไฟล์ส่วนขยาย) ซึ่งประกอบด้วยอาร์กิวเมนต์ ต่างๆ ดังนี้
  GAIM_PLUGIN_MAGIC,                    // หากส่วนขยายไม่ได้ update อาจทำให้ส่วนขยายไม่
สามารถถูกโหลดขึ้นมาใช้งานได้ และ จะป้องกันการ crash ในขณะที่โหลดส่วนขยายเก่าขึ้นมาใช้งาน
```

```
    GAIM_MAJOR_VERSION,          // เวอร์ชัน Gaim ที่ส่วนขยายใช้งานได้
    GAIM_MINOR_VERSION,          // เวอร์ชัน Gaim ที่ส่วนขยายใช้งานได้
    GAIM_PLUGIN_STANDARD,        // ชนิดของส่วนขยาย
    NULL,                        //ui requirement
    0,                           // flag ของส่วนขยาย
    NULL,                        //plugin dependencies
    GAIM_PRIORITY_DEFAULT,       // plugin priority

    "core-hello_world",          //plugin id
    "Hello World!",              // ชื่อส่วนขยาย ซึ่งจะแสดงในกล่องข้อความ
    VERSION,                     // เวอร์ชันของส่วนขยาย

    "Hello World Plugin",        // สาระสำคัญโดยย่อของส่วนขยาย
    "Hello World Plugin",        // คำอธิบายของส่วนขยาย
    NULL,                        // ชื่อและ email ของผู้พัฒนา
    "http://www.helloworld.tld", // website ของส่วนขยายของเรา
    plugin_load,                 // ตัวชี้ไปยังฟังก์ชันที่จะถูกเรียกใช้งานเมื่อโหลดส่วนขยาย
    NULL,                        // ตัวชี้ไปยังฟังก์ชันที่จะถูกเรียกใช้งานเมื่อจบการใช้ส่วนขยาย
    NULL,                        // ตัวชี้ไปยังฟังก์ชันที่จะถูกเรียกใช้งานเมื่อGaimทำลายส่วนขยาย
    NULL,                        // ตัวชี้ไปยัง struct ui
    NULL,                        // ตัวชี้ไปยัง struct GaimPluginLoaderInfo หรือ struct
                                 GaimPluginProtocolInfo
    NULL,                        // ตัวชี้ไปยัง struct GaimPluginUiInfo เป็นการกำหนดให้แสดง
หน้าจอที่สามารถปรับแต่งค่าต่างๆได้ของส่วนขยายในหน้าต่าง Preference ของโปรแกรม Gaim
    NULL                         // ตัวชี้ไปยังฟังก์ชันที่ระบุส่วนเมนู 'Tools->Plugin Actions'
};
```

ส่วนที่สาม

```
static void
init_plugin(GaimPlugin *plugin) {
}
```

// เมื่อ Gaim ต้องการตรวจสอบส่วนขยาย จะเรียกฟังกชัน init_plugin ซึ่งส่วนขยายส่วนใหญ่จะเพิ่มส่วน preferences ไปยัง preference tree ในฟังก์ชันนี้

GAIM_INIT_PLUGIN(hello_world, init_plugin, info);

//เป็น macro จัดการค่าเริ่มต้นต่างๆ ของส่วนขยายโปรแกรม Gaim ของเรา

2.4. การพิสูจน์ตัวตนโดยการเข้ารหัสลับโดยใช้กุญแจสาธารณะ (Public-key cryptography)

เป็นการรักษาความปลอดภัยของข้อมูลระหว่างการส่งข้ามเครือข่ายวิธีหนึ่งที่นิยมใช้กันอยู่ในปัจจุบัน การเข้ารหัสลับแบบคู่รหัสกุญแจนี้ จะมีความปลอดภัยมากกว่าการเข้ารหัสข้อมูลแบบธรรมดา แต่ก็ไม่ได้หมายความว่าการเข้ารหัสลับแบบคู่รหัสกุญแจนี้จะเป็นวิธีที่เหมาะสมที่สุดของวิธีการเข้ารหัส ทั้งนี้ขึ้นอยู่กับประเภทงานของแต่ละองค์กรหรือบุคคล

2.4.1. การเข้ารหัสลับโดยใช้กุญแจสาธารณะ ประกอบไปด้วยกุญแจ 2 ชนิด

- **กุญแจสาธารณะ (public key)** เป็นกุญแจที่ผู้สร้างจะส่งออกไปให้ผู้ใช้อื่นๆ ทราบหรือเปิดเผยได้
- **กุญแจส่วนตัว (private key)** เป็นกุญแจที่ผู้สร้างจะเก็บไว้ โดยไม่เปิดเผยให้คนอื่นรู้

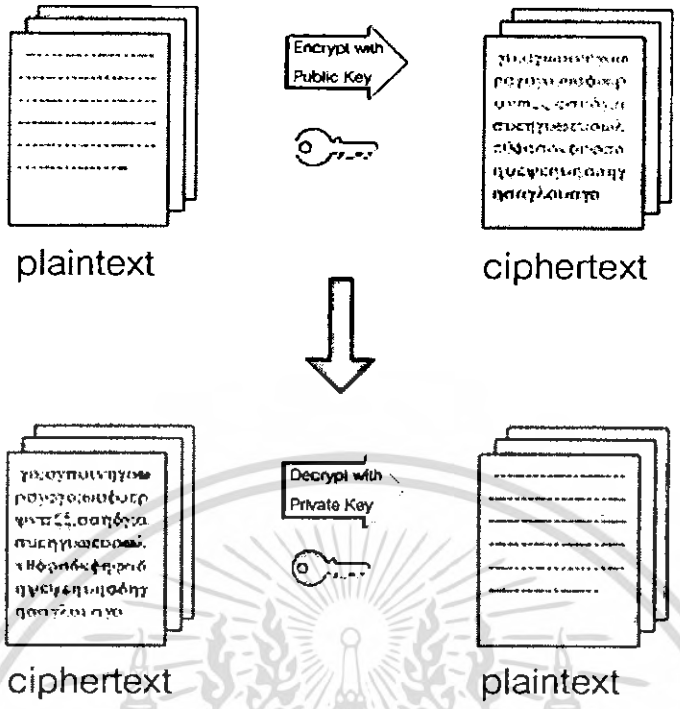
2.4.2. กระบวนการของการเข้ารหัสลับแบบคู่รหัสกุญแจ มีดังนี้

1. ผู้ใช้แต่ละคนจะสร้างคู่รหัสกุญแจของตัวเองขึ้นมา เพื่อใช้สำหรับการเข้ารหัสและการถอดรหัส
2. กุญแจสาธารณะจะถูกส่งออกไปยังผู้ใช้คนอื่นๆ แต่กุญแจส่วนตัวจะถูกเก็บที่ตนเอง
3. เมื่อจะส่งข้อมูลออกไปหาผู้ใช้คนใด ข้อมูลที่ส่งจะถูกเข้ารหัสด้วยกุญแจสาธารณะของผู้รับ ก่อนถูกส่งออกไป
4. เมื่อผู้รับได้รับข้อความแล้วจะใช้กุญแจส่วนตัวของผู้รับ ซึ่งเป็นคู่รหัสกันถอดรหัสลับออกมา

<u>หมายเหตุ</u> การเข้ารหัสโดยใช้กุญแจสาธารณะสามารถใช้ได้ทั้ง ในการเข้ารหัสลับ (Encryption) และ การพิสูจน์ตัวตน (Authentication)

2.4.3. การประยุกต์ใช้ในการเข้ารหัสลับข้อมูล (Encryption)

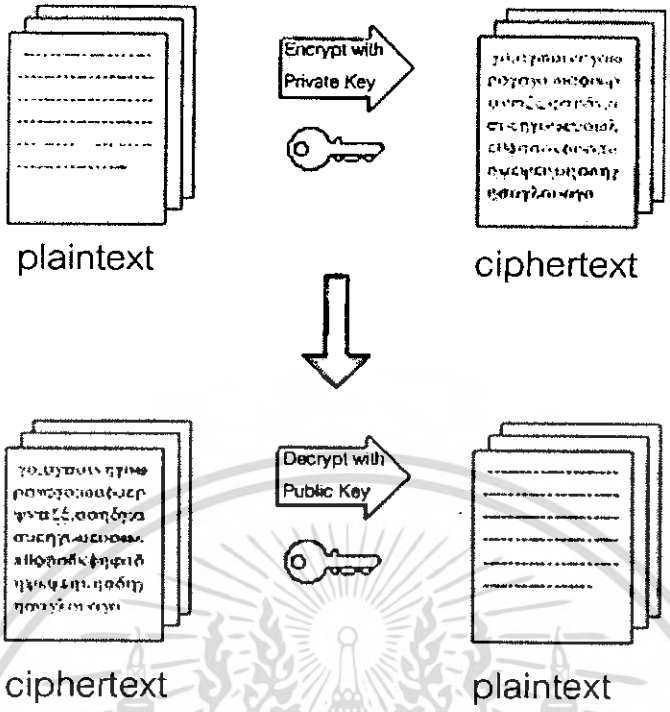
เป็นการนำข้อมูลที่จะส่งไปยังผู้รับมาเข้ารหัสลับด้วยกุญแจสาธารณะของผู้รับ และเมื่อผู้รับได้รับข้อความนั้นแล้วจะถอดรหัสลับออกมาด้วยกุญแจส่วนตัว จึงจะเห็นได้ว่ามีเพียงผู้รับเท่านั้นที่จะสามารถถอดรหัสลับออกมาได้



รูปที่ 2-18 ระบบของการเข้ารหัสลับแบบใช้คู่รหัสกุญแจ

2.4.4. การประยุกต์ใช้ในการพิสูจน์ตัวตน (Authentication)

เป็นการนำข้อมูลที่ผู้ส่งต้องการส่งมาเข้ารหัสลับด้วยกุญแจส่วนตัวของผู้ส่ง แล้วนำข้อมูลนั้นส่งไปยังผู้รับ ซึ่งผู้รับจะใช้กุญแจสาธารณะของผู้ส่งซึ่งเป็นคู่รหัสกันถอดรหัสลับออกมา ผู้รับก็สามารถรู้ได้ว่าข้อความนั้นถูกส่งมาจากผู้ส่งคนนั้นจริง หากสามารถถอดรหัสลับข้อมูลได้อย่างถูกต้อง



รูปที่ 2-19 ระบบของการเข้ารหัสลับแบบใช้คู่รหัสกุญแจเพื่อการพิสูจน์ตัวตน

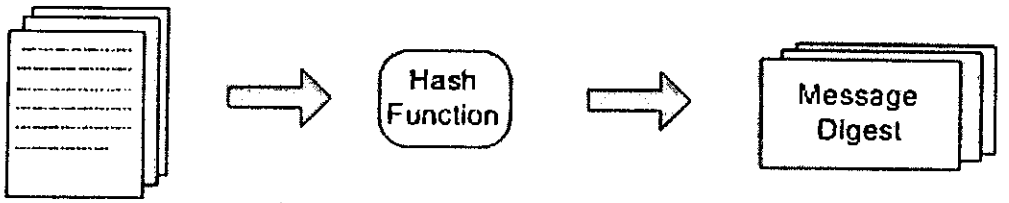
2.4.5. การพิสูจน์ตัวตนโดยการใช้ลายมือชื่อดิจิตอล (Digital Signature)

คุณสมบัติที่สำคัญของลายมือชื่อดิจิตอลที่สำคัญนั้นจะต้องประกอบด้วย 2 ประการคือ

1. สามารถยืนยันได้ว่าข้อมูลที่ได้รับมานั้นไม่มีการเปลี่ยนแปลงระหว่างการส่ง
2. สามารถยืนยันได้ว่าข้อมูลนั้นได้รับการยืนยันจากผู้ลงลายมือชื่อจริง ๆ

การพิสูจน์ตัวตนโดยการใช้ลายมือชื่อดิจิตอลเป็นการนำหลักการของการทำงานของระบบการเข้ารหัสลับแบบใช้คู่รหัสกุญแจเพื่อการพิสูจน์ตัวตนมาประยุกต์ใช้ ระบบของลายดิจิตอลสามารถแบ่งเป็นขั้นตอนได้ดังนี้

เมื่อผู้ใช้ต้องการจะส่งข้อมูลไปยังผู้รับ ข้อมูลนั้นจะถูกนำไปเข้าฟังก์ชั่นทางคณิตศาสตร์ที่เรียกว่า "แฮชฟังก์ชัน" ได้เมสเซสไดเจสต์ (Message Digest หรือ Digest) ออกมาเปรียบเสมือนตัวแทนของข้อมูล ถ้าข้อมูลที่ได้รับเข้าสู่แฮชฟังก์ชันนั้นต่างกันค่าของไดเจสต์ที่ได้รับออกมานั้นก็จะต่างกัน แต่หากว่าข้อมูลที่ได้รับเข้าสู่แฮชฟังก์ชันนั้นเหมือนกันก็จะทำให้ค่าของไดเจสต์ที่ได้ออกมาเหมือนกัน มีโอกาสน้อยมากที่จะทำให้ข้อมูลเปลี่ยนไปแล้วยังได้ไดเจสต์เหมือนเดิม

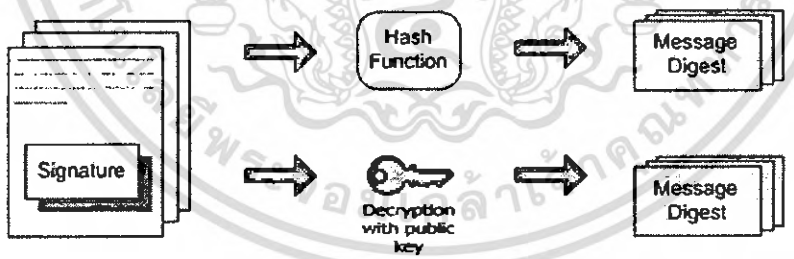


รูปที่ 2-20 การส่งข้อมูลเข้าไปใน Hash function

การใช้กุญแจส่วนตัวเข้ารหัสข้อมูล หมายถึงว่าผู้ส่งได้ลงลายมือชื่อดิจิตอล ยินยอมที่ จะให้ผู้รับ สามารถทำการตรวจสอบด้วยกุญแจสาธารณะของผู้ส่งเพื่อพิสูจน์ตัวตนของผู้ส่งได้

รูปที่ 2-21 การเข้ารหัสลับเมสเซสไดเจสต์ด้วยกุญแจส่วนตัวเพื่อเป็นการลงลายมือชื่อ

การตรวจสอบข้อมูลว่าถูกส่งมาจากผู้ส่งคนนั้นจริง โดยผู้รับจะนำข้อมูลมาผ่านแฮชฟังก์ชันเพื่อคำนวณหาค่าเมสเซจไดเจสต์ และถอดรหัสลับลายมือชื่ออิเล็กทรอนิคส์ด้วยกุญแจสาธารณะของผู้ส่ง ถ้าสามารถถอดได้อย่างถูกต้อง จะเป็นการยืนยันข้อมูลจากผู้ส่งคนนั้นจริง และถ้าข้อมูลเมสเซจไดเจสต์ที่ได้จากการถอดรหัสลับเท่ากันกับค่าเมสเซจไดเจสต์ในตอนต้นที่ทำการคำนวณได้ จะถือว่าข้อมูลดังกล่าวนั้นถูกต้อง



รูปที่ 2-22 ขั้นตอนการเปรียบเทียบความถูกต้อง

ลายเซ็นดิจิตอลนิยมนำไปใช้ในระบบรักษาความปลอดภัยในการชำระเงินผ่านระบบอินเตอร์เน็ต ซึ่งในปัจจุบันนี้การทำธุรกรรมการเงินอิเล็กทรอนิกส์ได้รับความนิยมเป็นอย่างมาก

2.5. SSL/TLS

SSL โพรโตคอล เป็นโพรโตคอลที่ได้รับการออกแบบมา เพื่อทำให้เกิดการสื่อสารอย่างปลอดภัยยิ่งขึ้น โดยโพรโตคอลสแต็กของ SSL จะอยู่ตรงกลางระหว่างชั้นแอพพลิเคชั่นเลเยอร์ (Application Layer) กับชั้นทรานสปอร์ตเลเยอร์(Transport Layer) นั่นคือ ชั้น SSL Layer จะเป็นชั้นที่เอาไว้ใช้สำหรับการสื่อสารที่ปลอดภัย ข้อดีคือเราสามารถพัฒนาแอพพลิเคชันได้อย่างเต็มที่ เนื่องจากการทำให้เกิดความปลอดภัยจะเป็นภาระหน้าที่ของชั้น SSL โดย SSL จะอาศัยทีซีพีโพรโตคอลในการรับส่งข้อมูล เพราะการส่งข้อมูลของ SSL ข้อมูลที่ถูกส่งไปนั้นเป็นข้อมูลที่ถูกเข้ารหัส ถ้ าเกิดข้อมูลส่วนใดสูญหายขึ้นมาในระหว่างการส่งข้อมูล จะทำให้ข้อมูลที่ถอดรหัสเป็นข้อมูลที่ไม่ถูกต้อง ทีซีพีจะเป็นตัวช่วยรับประกันว่าข้อมูลที่ถูกส่งไปยังผู้รับ จะต้องได้รับอย่างครบถ้วน ดังนั้นเราสามารถพัฒนาโพรโตคอลที่จะทำให้เกิดความปลอดภัยที่สูงขึ้นได้โดยไม่ต้องไปกังวลกับความถูกต้องในการส่งข้อมูล

2.5.1. บทบาทของ SSL (SSL Role)

SSL โพรโตคอลจะประกอบไปด้วยเซตของข้อความ (Message) และบทบาท (Role) ต่าง ๆ ที่เกี่ยวข้องเมื่อมีการส่งหรือรับข้อมูลซึ่งกันและกัน

SSL จะถูกกำหนดให้มีสองบทบาทสำหรับกระบวนการติดต่อสื่อสารกันระหว่างสองระบบ โดยบทบาทของระบบแรกจะต้องมีบทบาทเป็นไคลเอ็นต์ และอีกระบบจะต้องมีบทบาทเป็นเซิร์ฟเวอร์ การแยกความแตกต่างนี้มีความสำคัญเพราะว่า SSL จะปฏิบัติต่อบทบาททั้งสองนี้อย่างแตกต่างกันโดยสิ้นเชิง ไคลเอ็นต์จะเป็นผู้เริ่มต้นการติดต่อสื่อสารอย่างปลอดภัยขึ้นมา จากนั้นเซิร์ฟเวอร์จะเป็นผู้ตอบสนองต่อความต้องการของไคลเอ็นต์ ภาพที่เห็นได้ชัดของบทบาททั้งสองนี้คือ เว็บบราวเซอร์ (Web Browser) จะมีบทบาทเป็นไคลเอ็นต์ เว็บเซิร์ฟเวอร์ (Web Server) จะมีบทบาทเป็นเซิร์ฟเวอร์ นั่นคือเมื่อจะนำ SSL ไปใช้กับแอพพลิเคชั่นใด ๆ จะต้องคำนึงถึงความแตกต่างของระบบทั้งสองให้ชัดเจน

เหตุที่ SSL ต้องแยกความแตกต่างนี้เนื่องจากกระบวนการของ SSL จะต้องมีการทำการต่อรองค่าพารามิเตอร์ต่าง ๆ ในการสร้างการติต่อสื่อสารที่ปลอดภัยระหว่างกัน

ไคลเอ็นต์จะเป็นผู้เริ่มการติดต่อสื่อสาร ซึ่งเป็นค่านำเสนอค่าพารามิเตอร์ต่าง ๆ ที่ตนเองสามารถรองรับได้ส่งไปให้แก่เซิร์ฟเวอร์ และเซิร์ฟเวอร์จะเป็นผู้ทำการตัดสินใจขั้นสุดท้ายว่าทั้งสองระบบนี้จะใช้พารามิเตอร์ตัวใดในการสร้างการติดต่อสื่อสารที่ปลอดภัย ถึงแม้ว่าการตัดสินใจขั้นสุดท้ายจะอยู่ที่ฝั่งเซิร์ฟเวอร์ แต่มีข้อจำกัดอยู่ว่าเซิร์ฟเวอร์จะสามารถเลือกค่าพารามิเตอร์ต่างๆ ได้จากค่าพารามิเตอร์ที่ไคลเอ็นต์ได้ส่งมาให้แล้วเท่านั้น

2.5.2. ข้อความของ SSL (SSL Message)

เมื่อไคลเอ็นต์และเซิร์ฟเวอร์ทำการติดต่อสื่อสารกัน กระบวนการภายในการติดต่อสื่อสารกันนั้นจะเป็นการส่งข้อความของ SSL (SSL Message) ระหว่างกัน ซึ่งข้อความดังกล่าวจะมีความหมายที่แตกต่างกันไปตามเฟสในการสร้างการติดต่อสื่อสารที่ปลอดภัยโดยใช้ SSL โพรโตคอล เช่น Alert เป็นข้อความที่ทำให้ทราบว่าการติดต่อสื่อสารล้มเหลวหรือเกิดการผิดกฎความปลอดภัยของ SSL, Application Data เป็นข้อมูลจริง (Plaintext) ที่ต้องการทำการส่งให้กันซึ่งมันจะต้องถูกทำการเข้ารหัส การระบุตัวตนของผู้ส่งหรือรับและตรวจสอบความถูกต้องของข้อมูลโดยใช้กระบวนการของ SSL โพรโตคอล ฯลฯ

2.5.3 Transport Layer Security (TLS)

TLS เป็น Protocol ที่ให้บริการการติดต่อสื่อสารอย่างปลอดภัยบนเครือข่ายอินเทอร์เน็ต เช่น Web Browser, จดหมายอิเล็กทรอนิกส์, การส่งโทรสารผ่านอินเทอร์เน็ต และการส่งข้อมูลอื่นๆ ซึ่ง TLS 1.0 แตกต่างจาก SSL 3.0 เล็กน้อย แต่ส่วนหลักๆ ยังคงเหมือนกัน

โปรแกรมประยุกต์ใช้โพรโตคอล TLS ในการติดต่อสื่อสารบนช่องทางที่ถูกออกแบบไว้เพื่อป้องกันการแอบฟัง และการปลอมแปลงข้อมูล โดยใช้การพิสูจน์ตัวตน การรักษาความเป็นส่วนตัวในการติดต่อสื่อสาร และวิทยาการเข้ารหัสลับ มาจัดการ

ในปัจจุบัน เรามักใช้การพิสูจน์ตัวตนกับเครื่องแม่ข่ายเท่านั้น แต่ไคลเอนต์ไม่ใช้งานในส่วนนี้ จึงกล่าวได้ว่า ผู้ใช้งาน ซึ่งเป็นบุคคล หรือ โปรแกรมประยุกต์ต่างๆ สามารถมั่นใจได้ว่าบุคคลที่กำลังติดต่อสื่อสารด้วยนั้นเป็นบุคคลนั้นจริง แต่ในอนาคตจะใช้การพิสูจน์ตัวตนซึ่งกันและกันทั้งสองฝ่าย ซึ่งใช้โครงสร้างพื้นฐานกุญแจสาธารณะในไคลเอนต์ด้วย

TLS ประกอบด้วยขั้นตอนพื้นฐาน 3 ขั้นตอนดังนี้

1. การเจรจาระหว่างคู่สนทนาสำหรับ ข้อตกลงระหว่าง Algorithm ที่ใช้งาน
2. การเข้ารหัสลับด้วยกุญแจสาธารณะ การแลกเปลี่ยนกุญแจ และการพิสูจน์ตัวตนด้วยใบรับรองสิทธิ์
3. การเข้ารหัสลับแบบสมมาตร

ในขั้นตอนแรก การเจรจาระหว่างเครื่องแม่ข่ายและไคลเอนต์จะใช้ Algorithm ต่างๆ ได้แก่

- การเข้ารหัสลับด้วยกุญแจสาธารณะ : RSA, Diffie-Hellman, DSA หรือ Fortezza
- การเข้ารหัสลับแบบสมมาตร : RC2, RC4, IDEA, DES, Triple DES, AES หรือ Camellia
- ฟังก์ชันแฮชทางเดียว : MD2, MD4, MD5 หรือ SHA

การทำงานของ TLS

โพรโตคอล TLS แลกเปลี่ยนข้อมูล ซึ่งอาจถูกบีบอัด เข้ารหัสลับ และห่อไว้ด้วย Message Authentication Code (MAC) นอกจากนี้ข้อมูลแต่ละชุดจะมีฟิลด์ content type ซึ่งระบุว่า ใช้ Protocol ในชั้นสูงกว่า Protocol ใด

เมื่อไคลเอนต์ต้องการติดต่อสื่อสาร จะส่งและรับข้อมูลดังนี้

1. ไคลเอนต์ส่งข้อความ Client Hello ซึ่งระบุชุด Algorithm ในการเข้ารหัส วิธีการบีบอัด และเวอร์ชันโพรโตคอลสูงที่สุดที่ไคลเอนต์สามารถใช้งานได้ นอกจากนี้ยังมี ข้อความในรูปแบบไบต์แบบสุ่มซึ่งจะใช้ในภายหลังอีกด้วย
2. จากนั้นไคลเอนต์จะรับข้อความ ServerHello ซึ่งระบุพารามิเตอร์ในการติดต่อสื่อสารที่เครื่องแม่ข่าย สามารถใช้ได้จากที่ไคลเอนต์เสนอมาในขั้นตอนที่แล้ว
3. หลังจากเจรจาตกลงค่าพารามิเตอร์แล้ว ไคลเอนต์และเครื่องแม่ข่ายจะแลกเปลี่ยนใบรับรองสิทธิ์ (ขึ้นกับ Algorithm กุญแจสาธารณะที่เลือกไว้แล้ว) โดยใบรับรองสิทธิ์จะอยู่ในรูปแบบ X.509
4. เครื่องแม่ข่ายสามารถร้องขอใบรับรองสิทธิ์จากไคลเอนต์ได้ ซึ่งจะทำให้เกิดการพิสูจน์ตัวตนซึ่งกันและกันอย่างสมบูรณ์
5. เครื่องแม่ข่ายและไคลเอนต์จะเจรจาข้อความลับ เรียกว่า Master secret โดยอาจใช้ผลการแลกเปลี่ยน Diffie-Hellman หรือ การเข้ารหัสลับ secret อย่างง่ายด้วยกุญแจสาธารณะ ซึ่งสามารถถอดรหัสลับได้โดยกุญแจส่วนตัวของคู่สนทนา นอกจากนี้กุญแจอื่นๆ ยังถูกสร้างขึ้นจาก master secret และค่าที่สุ่มขึ้นมาจากไคลเอนต์และเครื่องแม่ข่าย

การประยุกต์ใช้งาน

TLS ทำงานอยู่ตรงกลางระหว่างชั้นแอพพลิเคชั่นเลเยอร์ (Application Layer) กับชั้น ทรานสปอร์ตเลเยอร์ (Transport Layer) ใน TCP/IP model ทำให้เราสามารถเพิ่มการรักษาความปลอดภัยให้ทุกโพรโตคอล ที่ต้องการการติดต่อสื่อสารอย่างปลอดภัย (เช่น TCP) ส่วนใหญ่มักใช้กับโพรโตคอล HTTP เรียกว่า HTTPS (ใช้ในการรับส่งหน้าเพจ (Webpage) เช่นในระบบธุรกิจทางอินเทอร์เน็ต) โดยใช้ใบรับรองสิทธิ์เพื่อตรวจสอบคู่สนทนา

จำนวนไคลเอนต์และเครื่องแม่ข่ายที่สนับสนุนการทำงาน TLS มีมากขึ้นเรื่อยๆ แต่ยังมีบางส่วนไม่สนับสนุน ซึ่งอาจต้องการใช้งาน TLS ในส่วนนี้แบบใช้งานคนเดียว (Standalone) เช่นโปรแกรม Stunnel ซึ่งสามารถรับการเชื่อมต่อ TLS ได้ทันที อย่างไรก็ตามในปี พ.ศ. 2540

หน่วยงาน Internet Engineering Task Force (IETF) แนะนำว่าโปรแกรมประยุกต์บางโปรแกรมที่ทำงานในลักษณะการส่งข้อมูลธรรมดา ควรเปลี่ยนมาใช้ TLS

เราสามารถใช้ TLS เป็นอุโมงค์ในการสร้าง Virtual Private Network (VPN) เช่นในกรณีโปรแกรม OpenVPN ได้ ผู้ผลิตหลายรายได้รวมการเข้ารหัสลับ และการพิสูจน์ตัวตนของ TLS เข้ากับการกำหนดระดับสิทธิ์การใช้งาน (Authorization) และตั้งแต่ปี พ.ศ. 2533 มีการพัฒนาโปรแกรมประยุกค์อย่างจริงจังในการสร้างเทคโนโลยีของไคลเอนต์ นอกจากการพัฒนา Browser นอกจากนี้ TLS มีข้อดีต่อระบบไฟร์วอลและระบบ Network Address Translation (NAT) ในการดูแลระบบขนาดใหญ่

ประวัติ

TLS ถูกพัฒนาโดย Netscapeโดยมีการออก SSL (Secure Socket Layer) เวอร์ชันที่ 3 ในปี พ.ศ.2539 ซึ่งต่อมาเป็นส่วนประกอบหลักของ TLS เวอร์ชันที่ 1.0 และในเดือนมกราคมปี พ.ศ. 2542 หน่วยงาน IETF ได้ออกมาตรฐานไว้ใน RFC 2246 ทำให้ Visa, MasterCard, American Express และสถาบันการเงินต่างๆ นำมาใช้งานในเชิงธุรกิจบนเครือข่ายอินเทอร์เน็ต

การใช้งานผิดวิธี

บางเว็บไซต์วิจารณ์การใช้งาน TLS ผิดวิธี และทำให้ความปลอดภัยลดลง อาทิเช่น

1. การรักษาความปลอดภัยเฉพาะการส่งข้อมูลส่วนตัว แต่ไม่รักษาความปลอดภัยในหน้าล็อกอิน
2. แสดงหน้าเพจที่ปลอดภัยรวมกับข้อมูลมัลติมีเดียที่ไม่ปลอดภัย

2.6. ทฤษฎีใบรับรองสิทธิ์

2.6.1. ใบรับรองสิทธิ์ (Certificate)

ใบรับรองสิทธิ์ คือ ใบที่ใช้พิสูจน์ถึงตนเองเมื่อต้องการติดต่อกับผู้อื่น เหมือนการใช้บัตรประชาชน ต่างกันตรงที่ใบรับรองสิทธิ์นี้เอาไว้รับรองสิทธิ์ในการสื่อสารในโลกดิจิตอลเท่านั้น

การที่ไคลเอ็นต์จะสามารถทำการระบุตัวตนของเซิร์ฟเวอร์ได้ เซิร์ฟเวอร์จะส่งข้อมูลต่างๆ ที่สามารถเป็นการระบุตัวตนได้เช่น ชื่อ,กุญแจสาธารณะของเซิร์ฟเวอร์ ไปให้แก่ CA Certificate Authority) ซึ่งCA จะแบ่งออกเป็นสองลักษณะคือ Internal CA และ External CA)ซึ่งจะได้กล่าวในหัวข้อที่ 3 Certificate Authority(CA)) และ CA จะทำกระบวนการที่เรียกว่า Sign() โดยจะใช้คีย์ส่วนตัวของ CA ทำการ Sign ข้อมูลที่ได้รับมาจากเซิร์ฟเวอร์ และเมื่อนำกุญแจสาธารณะของ CA กับข้อมูลที่ได้มาจากการ Sing ดังกล่าวจาก CA แล้วมาทำกระบวนการที่เรียกว่าการ Verify() ก็จะสามารถพิสูจน์ตัวตนของเซิร์ฟเวอร์ได้ จากรูปไคลเอ็นต์จะมีกุญแจสาธารณะของ CA ทีเป็น CA ที่เซิร์ฟเวอร์ได้นำข้อมูลต่างๆ ไปให้ CA นั้นทำการ Sign ให้ โดยข้อความ Certificate ของฝั่งเซิร์ฟเวอร์จะประกอบไปด้วย กุญแจสาธารณะของเซิร์ฟเวอร์และข้อมูลได้รับการ Sing มาจาก CAนั้นส่งไปให้แก่ไคลเอ็นต์

2.6.2. Certificate Authority (CA)

CA เป็นองค์กรหรือสมาคมที่จะออกใบรับรองสิทธิ์ให้แก่ผู้ขอ โดยผู้ขอใบรับรองสิทธิ์จากCA จะต้องมั่นใจในใบรับรองสิทธิ์นั้นแต่ไม่ได้หมายความว่า CA นั้นจะไม่มีความผิดพลาดเลย ตัวอย่างองค์กรที่เป็น CA เช่น Verisign รายละเอียดต่างๆ หาได้ที่ www.verisign.com

2.6.2.1. ประเภทของ CA มี 2 ชนิด

- Public CA หรือ External CA

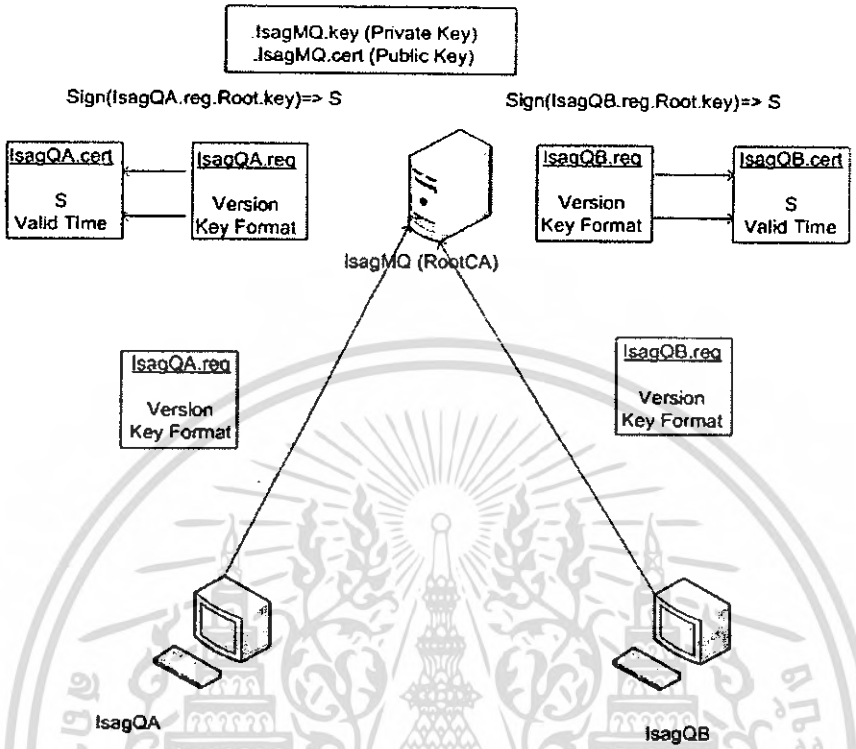
เป็นองค์กรที่ออกใบรับรองสิทธิ์ให้แก่บุคคลหรือองค์กรโดยทั่วไปเช่น Verisign จะออกใบรับรองสิทธิ์ให้กับเว็บไซต์ (web sites) หรือแอพพลิเคชั่นที่ต้องการ ให้มีการเข้ารหัสและการรับรองสิทธิ์ของผู้ใช้ในการทำกิจกรรมต่าง ๆ เช่น อีคอมเมิร์ซ (e-commerce) ให้มีระบบการรับส่งข้อมูลที่ปลอดภัยเช่น รหัสผ่านของเครดิตการ์ดของลูกค้า ซึ่งโดยมากแล้วถ้าเป็น External CA ถ้าเราไปขอใบรับรองสิทธิ์เราจะไม่ได้มาฟรี ๆ จะต้องมีค่าใช้จ่ายให้แก่ External CA นั้นๆ เสมอ

- Private CA หรือ Internal CA

เป็นองค์กรที่ออกใบรับรองสิทธิ์ให้แก่คนที่อยู่ภายในองค์กรเพื่อใช้งานภายในองค์กรเอง อ งค์กรภายนอกไม่สามารถใช้ CA และจะไม่สามารถไว้วางใจการออกแบบนี้ได้ แต่ขึ้นอยู่กับ Internal CA นั้น ๆ ว่ามีกฎระเบียบเป็นอย่างไร ซึ่ง CA ประเภทนี้จะออกใบรับรองสิทธิ์ให้ฟรีเพราะถือว่าบุคคลเหล่านั้นอยู่ภายในองค์กร

โครงงานนี้ได้ทำระบบ CA แบบ Private CA เพราะผู้จัดทำเห็นว่า เหมาะสมที่จะออกใบรับรองสิทธิ์ให้แก่ผู้ใช้ของ IsagMQ เท่านั้น เพื่อการสื่อสารอย่างปลอดภัยภายในกลุ่ม ในที่นี้จะอธิบายการสร้างและการออกแบบ Private CA เท่านั้นซึ่งจะอยู่ในส่วนการออกแบบและพัฒนา

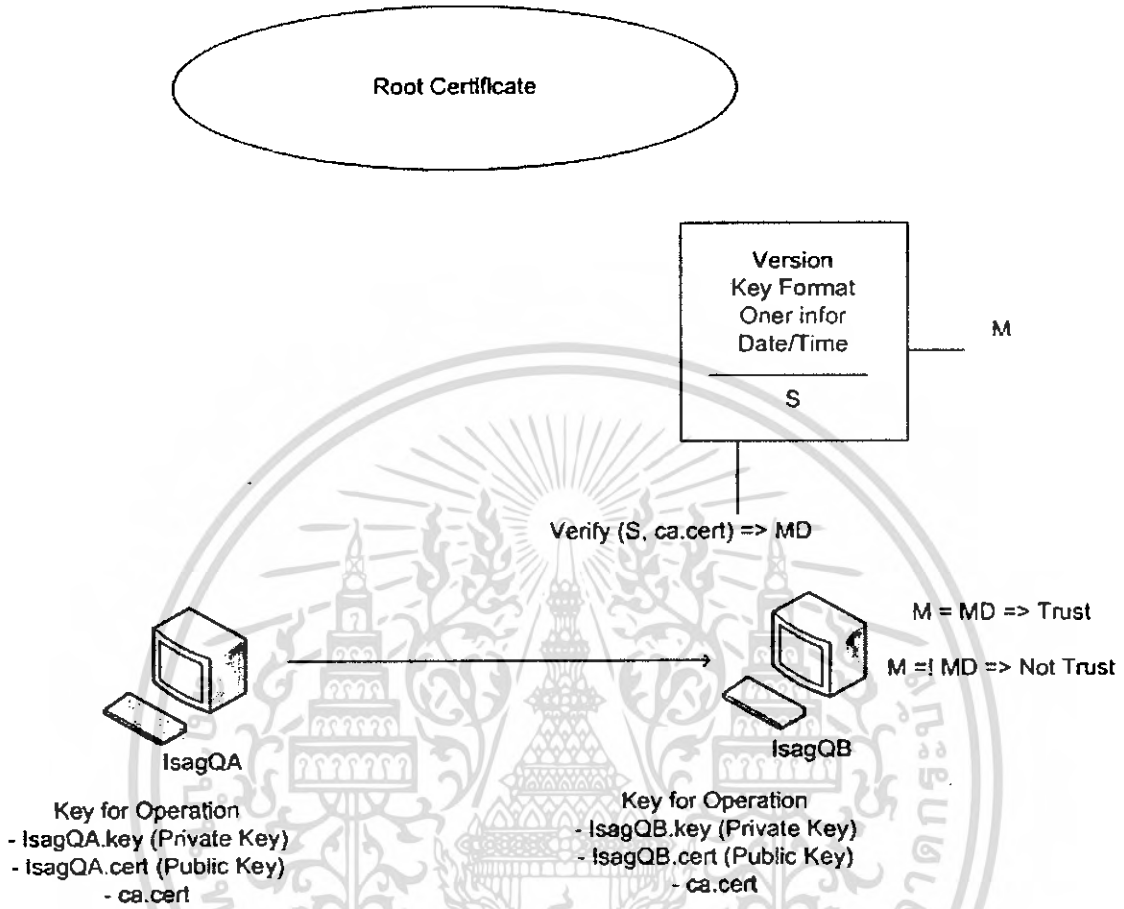
2.6.3. ขั้นตอนในการร้องขอใบรับรองสิทธิ์



รูปที่ 2-23 ขั้นตอนการร้องขอใบรับรองสิทธิ์

1. ผู้ร้องขอซึ่งในที่นี้จะประกอบด้วย IsagQA และ IsagQB จำเป็นต้องมีใบรับรองสิทธิ์ของ Root CAซึ่งก็คือ IsagMQ เพื่อเก็บไว้ใช้ในระบบพิสูจน์ตน
2. เมื่อผู้ร้องขอมีใบรับรองสิทธิ์ของ IsagMQ แล้ว ก็ให้ทำการร้องขอไปยัง Root CA ก่อนที่ผู้ร้องขอจะส่งไฟล์ใบร้องขอใบรับรองสิทธิ์ .req ออกไปผู้ร้องขอจำเป็นต้องมีไฟล์ 2 ชนิดคือ IsagQA.key หรือ IsagQB.key ซึ่งก็คือกุญแจส่วนตัวของผู้ร้องขอและ IsagQA.req หรือ IsagQB.req ซึ่งก็คือไฟล์ที่ใช้ร้องขอไปยังผู้ออกใบรับรองสิทธิ์โดยไฟล์ที่ร้องขอต้องถูกเข้ารหัสด้วย root.cert ก่อนเพื่อที่จะมั่นใจได้ว่า Root CA จะสามารถเปิดดูได้เพียงผู้เดียว
3. หลังจากที่ Root CA ได้รับ IsagQA.req หรือ IsagQB.req แล้วก็จะทำการแฮช (hash) แล้วนำค่านั้นมา sign ด้วย Root.key ผลลัพธ์ที่ได้จะได้ค่า S (Digital Signature) แล้วนำค่า S ที่ได้ไปเก็บไว้ใน IsagQA.cert หรือ IsagQB.cert
4. ผู้ออกใบรับรองสิทธิ์จะทำการส่ง IsagQA.cert หรือ IsagQB.cert คืนกลับให้ผู้ร้องขอ

2.6.4. การพิสูจน์ตนโดยการใช้ใบรับรองสิทธิ์(Certificate Authentication)



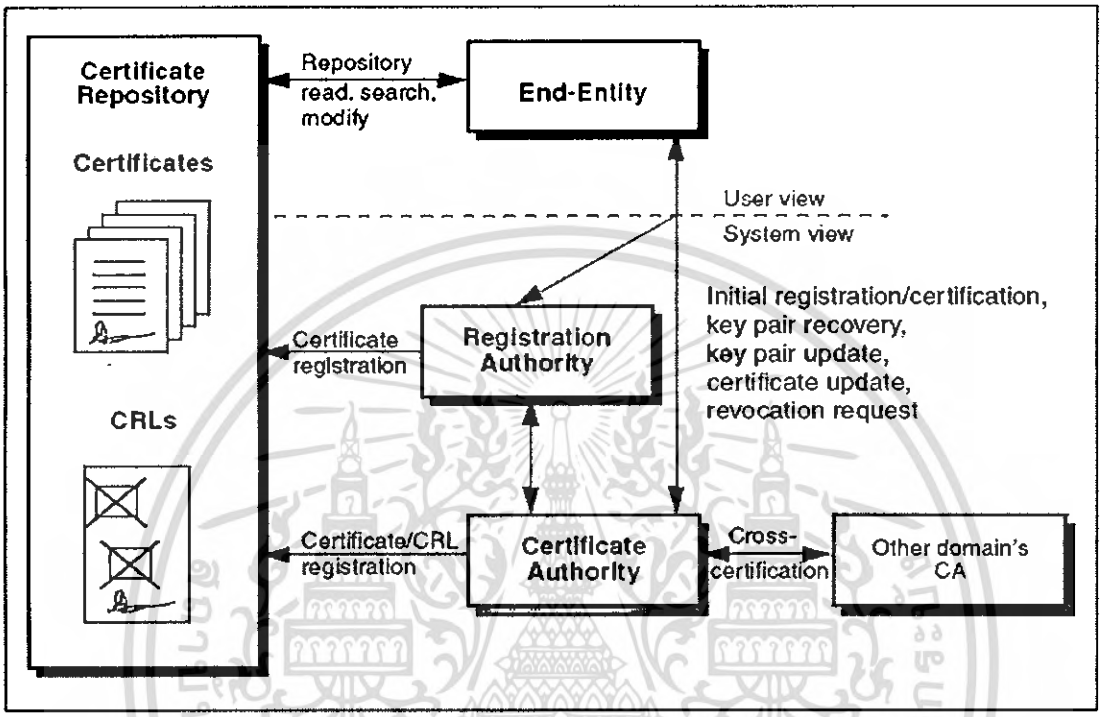
รูปที่ 2-24 รูปแสดงการพิสูจน์ตัวตนโดยใช้ใบรับรองสิทธิ์

เพื่อเริ่มทำการพิสูจน์ตัวตน IsagQA ก็จะส่ง IsagQA.cert ให้แก่ IsagQB ทำการพิสูจน์ตนของฝั่ง IsagQA เมื่อ IsagQB ได้รับไฟล์ IsagQA.cert ซึ่งไฟล์นี้ประกอบไปด้วยสองส่วนคือ ส่วนข้อมูลส่วนตัวของ IsagQA และค่า S (Digital Signature) กระบวนการพิสูจน์ตนทางฝั่ง IsagQB คือ จะนำเอาข้อมูลส่วนตัวของ IsagQA มาทำ MD5 ได้ค่าแฮช (hash) มาหนึ่งค่าคือค่า M จากนั้นนำค่า S ของIsagQA และกุญแจสาธารณะของ Root Certificate มาทำการตรวจสอบค่าที่ได้ออกมาคือ MD นำค่า M กับ MD มาเปรียบเทียบกัน ถ้าค่านั้นเท่ากันแสดงว่าการพิสูจน์ตัวตนนั้นถูกต้อง ถ้าไม่เท่ากันแสดงว่าการพิสูจน์ตนผิดพลาด

2.7 โครงสร้างพื้นฐานกุญแจสาธารณะ (Public Key Infrastructure)

2.7.1 ส่วนประกอบหลักของ PKI

จะประกอบด้วย 5 ส่วน ดังนี้



รูปที่ 2-25 แสดงภาพรวมของ PKI

- End-Entities (EE)
- The Certificate Authority (CA)
- The Certificate Repository (CR)
- The Registration Authority (RA)
- Digital Certificates (X.509 V3)

1. End-Entities (EE)

เป็นผู้ใช้ใบรับรองสิทธิ์ใน PKI และ/หรือ เป็นระบบที่ใช้บริการหรือฟังก์ชันบางอย่างของระบบ PKI ซึ่งอาจจะเป็นเจ้าของใบรับรองสิทธิ์ (บุคคล หรือ องค์กร หรือ อื่นๆ) หรือ ผู้ต้องการใช้ใบรับรองสิทธิ์ (อาจจะเป็นโปรแกรมประยุกต์ต่างๆ ได้)

2. Certificate Authority (CA)

เป็นผู้เซ็นใบรับรองสิทธิ์ จะรับผิดชอบในส่วนการสร้างใบรับรองสิทธิ์ การออกใบรับรองสิทธิ์ใหม่ และการถอดถอนใบรับรองสิทธิ์

2.7.2 การสร้างใบรับรองสิทธิ์

CA จะสร้างใบรับรองสิทธิ์โดยเซ็นด้วยลายมือชื่อดิจิตอล โดยปกติแล้วผู้ใช้งานจะสร้างกุญแจส่วนตัวและกุญแจสาธารณะเอง แล้วจึงแสดงความต้องการใบรับรองสิทธิ์แก่ CA แต่บางกรณี CA จะสร้างคู่กุญแจดังกล่าวให้แก่ผู้ใช้งาน ซึ่งกรณีหลังจะได้รับความนิยมน้อยกว่า เนื่องจากต้องส่งกุญแจส่วนตัวของผู้ใช้งานไปให้ผู้ใช้งานผ่านช่องทางที่อาจจะไม่ปลอดภัยได้

ใบร้องขอใบรับรองสิทธิ์จะประกอบด้วยกุญแจสาธารณะของผู้ใช้งาน และข้อมูลอื่นๆ เช่น ชื่อผู้ใช้งาน อีเมล์ หรือ ข้อมูลที่เกี่ยวข้อง ซึ่งผู้ใช้งานจะส่งใบร้องขอใบรับรองสิทธิ์ไปที่ CA เมื่อ CA ได้รับ จะให้ RA ตรวจสอบและยืนยันผู้ใช้งาน หลังจากนั้น CA จึงจะสร้างและเซ็นใบรับรองสิทธิ์

ในการทำงานทั่วไปของ PKI ฝ่ายใดต้องการตรวจสอบใบรับรองสิทธิ์ จะต้องเชื่อถือ CA ผู้ซึ่งเป็นคนเซ็นใบรับรองสิทธิ์นั้นๆ ด้วย เช่น นาย ก เชื่อถือ นาย ข หมายความว่า นาย ก เชื่อถือ CA ซึ่งเซ็นใบรับรองสิทธิ์ของนาย ข และ A เชื่อถือ CA หมายความว่า A มีใบรับรองสิทธิ์ของ CA อยู่ ดังจะเห็นได้จาก browser ต่างๆ จะมีใบรับรองสิทธิ์ของ CA อยู่ด้วย เมื่อ web server ใช้ใบรับรองสิทธิ์ซึ่งมี CA ที่เชื่อถือได้เซ็น ผู้ใช้งานสามารถเชื่อถือ server นั้นได้ทันที ถ้าผู้ใช้งานนั้นๆ ไม่ได้ลบใบรับรองสิทธิ์ของ CA ออกก่อน

CA สามารถสร้างใบรับรองสิทธิ์ได้หลายชนิด อาทิเช่น

User certificates

CA สร้างใบรับรองสิทธิ์แบบนี้สำหรับบุคคล หรือ entity อื่นๆ เช่น เซิร์ฟเวอร์ และโปรแกรมต่างๆ โดยใบรับรองสิทธิ์แบบนี้มักจะถูกจำกัดในการใช้งานด้าน อีเมล์, server หรือ จุดประสงค์เฉพาะอื่นๆ

CA certificates

ใบรับรองสิทธิ์CA ที่ออกโดยCA นั้นๆ เรียกว่า *self-signed certificate* หรือ *root certificate* สำหรับ CA นั้นๆ แม้แต่ใบรับรองสิทธิ์ที่ออกโดย CA เพื่อ CA อื่นๆ ก็เรียกว่า CA certificates เช่นกัน

Cross certificate

ใช้ในกรณีพิสูจน์ตัวตนข้ามขอบเขตของ CA ที่ตนเชื่อถืออยู่

2.7.3 การออกใบรับรองสิทธิ์ใหม่

เนื่องจากใบรับรองสิทธิ์ทุกใบมีช่วงเวลาที่ใช้งานได้ตามวันหมดอายุที่ได้ระบุไว้ในใบรับรองสิทธิ์ เมื่อใบรับรองสิทธิ์หมดอายุ จะเกิดกระบวนการออกใบรับรองสิทธิ์ใหม่ ทำให้มีการสร้างใบรับรองสิทธิ์ใบใหม่แก่ผู้ใช้งานต่อไป

2.7.4 การถอดถอนใบรับรองสิทธิ์

ในบางกรณีจะมีการถอดถอนใบรับรองสิทธิ์ก่อนวันหมดอายุ CA จะจดบันทึกใบรับรองสิทธิ์ ดังกล่าวไว้ใน Certificate Revocation List (CRL) เมื่อผู้ใช้งานต้องการทราบว่าใบรับรองสิทธิ์ต่างๆ ยังสามารถใช้งานได้อยู่หรือไม่ สามารถค้นหาประกาศการถอดถอนใบรับรองสิทธิ์ได้จาก CRL นี้

2.7.5 The Certificate Repository (CR)

เป็นที่เก็บใบรับรองสิทธิ์ที่ออกโดย CA และใบรับรองสิทธิ์ที่ถูกยกเลิกแล้ว CR เป็นส่วนที่ทำให้ระบบโครงสร้างพื้นฐานกุญแจสาธารณะสามารถจัดการได้ง่าย แม้ว่า CR จะไม่ได้เป็นส่วนประกอบที่จำเป็นในระบบก็ตาม

การใช้ระบบไดเรกทอรี มำการสร้างเป็น CR เป็นวิธีการที่ดีที่สุด ซึ่งเราสามารถใช้ Directory Access Protocol (DAP) อาทิเช่น *Lightweight Directory Access Protocol* (LDAP) ในการใช้งาน ไดเรกทอรี ดังกล่าวได้ โดย RFC 2587 ได้อธิบายวิธีการเข้าถึงข้อมูลในไดเรกทอรี ว่า End-Entity หรือ CA สามารถรับข้อมูลหรือ แก้ไข ใบรับรองสิทธิ์และข้อมูลใน CRL ได้อย่างไรบ้าง ดังนี้แล้ว เราสามารถใช้กระบวนการหรือคำสั่งใน LDAP เช่น bind, search หรือ modify, และ unbind เข้าถึงข้อมูลใน CR ได้

แม้ว่าจะมีวิธีการเก็บใบรับรองสิทธิ์ หรือ ข้อมูล CRL ด้วยวิธีการอื่นๆ แต่เมื่อพิจารณาถึงความต้องการบางอย่าง เช่น การเข้าถึงข้อมูลได้ง่าย, การเข้าถึงข้อมูลในรูปแบบมาตรฐานเดียวกัน, ที่เก็บข้อมูลที่มีการ update อย่างสม่ำเสมอ, ความปลอดภัย, การจัดการข้อมูล ฯลฯ สามารถสรุปได้ว่าวิธีการเก็บข้อมูลดีที่สุดคือระบบไดเรกทอรี

CR ยังทำให้การออกแบบระบบการแจกจ่าย CRL ง่ายขึ้น นอกจากนี้ ความยืดหยุ่น และความเรียบง่ายของ LDAP ทำให้เราสามารถนำไปใช้งานได้หลากหลายวัตถุประสงค์อีกด้วย

2.7.6 The Registration Authority (RA)

เป็นส่วนประกอบเสริมในโครงสร้างพื้นฐานกุญแจสาธารณะ ในบางกรณี CA จะรวมบทบาท RA ไว้ในตัวด้วย แต่ในกรณีที่มีการใช้ RA ซึ่งเป็น End-Entity ที่ CA เชื่อถือ โดย CA จะมอบหมายหน้าที่การจัดการบางอย่างให้แก่ RA เช่น RA มีหน้าที่ในการพิสูจน์ตัวตนของบุคคล,

รายงานใบรับรองสิทธิ์ที่เพิกถอนแล้ว, สร้างคู่กุญแจใหม่ ฯลฯ แต่ RA จะไม่ออกใบรับรองสิทธิ์หรือ CRL

2.7.7 X.509 certificates

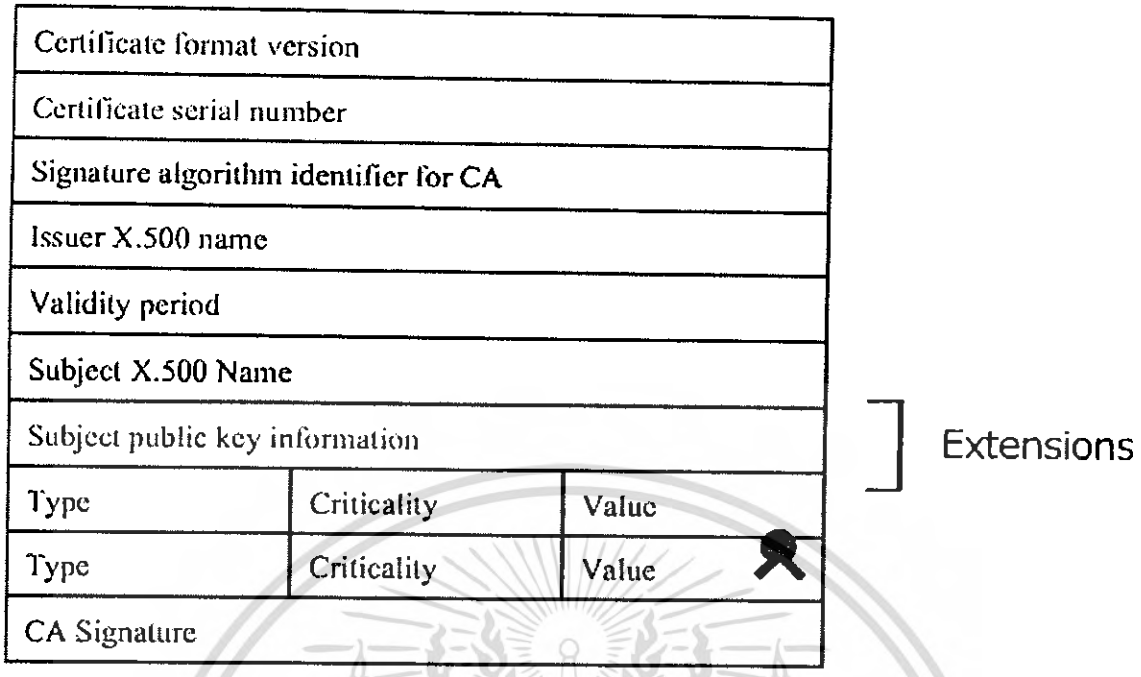
X.509 เป็นรูปแบบใบรับรองสิทธิ์ที่ใช้กันอย่างกว้างขวางมาก ในด้านโพรโตคอลและ application ที่สนับสนุนการทำงานในโครงสร้างพื้นฐานกุญแจสาธารณะ เช่น SSL, IPSec, S/MIME,Privacy Enhanced Mail (PEM), หรือ SET โดย ITU-T ได้ประกาศ X.509 version 1 ในปี ค.ศ. 1988 ต่อมา version 2 ได้เพิ่มส่วน issuer และ subject identifier เข้าไป และ X.509 version 3 ซึ่งเป็นรุ่นล่าสุด ซึ่งมีส่วน *extension* เพิ่มเติม version 2

ในปัจจุบัน แม้ว่าโปรแกรมที่ใช้ในโครงสร้างพื้นฐานกุญแจสาธารณะส่วนใหญ่จะใช้ X.509 version 1 และ 2 แต่แนวโน้มเทคโนโลยีโครงสร้างพื้นฐานกุญแจสาธารณะเริ่มหันมาใช้ version 3 กันมากขึ้น

ข้อมูลในใบรับรองสิทธิ์จะถูกเขียนในรูปแบบ *abstract syntax notation 1* (ASN.1) ดังรูปที่ หน้า ซึ่งเราสามารถแปลงเป็นข้อมูล binary ด้วย *distinguished encoding rules* (DER) ซึ่ง ASN.1 จะทำให้ข้อมูลใบรับรองสิทธิ์ไม่ขึ้นกับกฎการเข้ารหัสของแต่ละ platform

2.7.8 รูปแบบของใบรับรองสิทธิ์ (Certificate Profile)

รูปแบบของใบรับรองสิทธิ์ อิงตามมาตรฐาน X.509 Certificate ซึ่งเป็นมาตรฐานของใบรับรองสิทธิ์ที่กำหนดโดย ITU-T X.509 International Standard ที่บอกถึงรูปแบบของใบรับรองฯ โดยประกอบไปด้วยข้อมูลหลักๆ 2 ส่วน คือ ข้อมูลพื้นฐานของใบรับรองสิทธิ์ (Basic Certificate Fields) และข้อมูลเพิ่มเติมของใบรับรองสิทธิ์ (Certificate Extension) ดังรูปที่ 1



รูปที่ 2-26 ส่วนประกอบของใบรับรองสิทธิ์

1. ข้อมูลพื้นฐานของใบรับรองสิทธิ์ (Basic Certificate Fields)

ใบรับรองสิทธิ์ประกอบไปด้วยข้อมูลพื้นฐาน ดังนี้

Version

เวอร์ชันของใบรับรองสิทธิ์ที่ใช้ตามมาตรฐาน X.509 Certificate ซึ่งมีทั้งหมดด้วยกัน 3 เวอร์ชัน คือ

เวอร์ชัน 1, 2 และ 3 โดยที่เวอร์ชัน 3 จะรองรับการใช้งานใบรับรองสิทธิ์ที่มีข้อมูลเพิ่มเติม (Certificate Extension)

Serial Number

หมายเลขที่ผู้ให้บริการออกใบรับรองทำการกำหนดให้กับใบรับรองสิทธิ์แต่ละใบ เพื่อป้องกันการซ้ำกันของใบรับรองฯ ในกรณีที่ใบรับรองสิทธิ์ถูกเพิกถอน หมายเลขดังกล่าวจะปรากฏอยู่ในรายการเพิกถอนใบรับรอง โดยที่รายการเพิกถอนใบรับรองฯ นี้จะถูกรับรองโดยผู้ให้บริการออกใบรับรอง

Signature

อัลกอริทึมที่ผู้ให้บริการออกใบรับรองฯ ใช้ในการลงลายมือชื่อดิจิทัลและใช้ในการย่อยข้อมูล (Hash Function) เพื่อทำการรับรองใบรับรองสิทธิ์ เช่น sha1WithRSAEncryption, md5withRSAEncryption เป็นต้น

Issuer

ชื่อของผู้ให้บริการออกใบรับรอง ที่ทำการรับรองและออกใบรับรองสิทธิ์ ซึ่งจะใช้รูปแบบของ Distinguished Name (DN) ตามมาตรฐาน X.500 เช่น c=TH, o=GOV เป็นตัวอย่างของ DN ซึ่งหมายถึงผู้ให้บริการออกใบรับรองที่ทำการออกใบรับรองสิทธิ์ให้กับหน่วยงานที่เป็นภาครัฐในประเทศไทย

Validity

ช่วงเวลาที่สามารถใช้งานใบรับรองสิทธิ์ โดยระบุถึงวัน-เวลาเริ่มต้นและสิ้นสุดของการใช้งานใบรับรองฯ ซึ่งมี 2 มาตรฐานในการกำหนดวัน-เวลาดังกล่าว คือ

- Universal time (UTC Time) ซึ่งมีรูปแบบของวัน-เวลา ดังนี้ YYMMDDHHMMSSZ
- generalized time (Generalized Time) ซึ่งมีรูปแบบของวัน-เวลา ดังนี้ YYYYMMDDHHMMSSZ

โดยปีคริสต์ศักราชที่น้อยกว่าปีคริสต์ศักราช 2049 ใช้รูปแบบของวัน-เวลาเป็น UTC Time ส่วนปีคริสต์ศักราช 2050 เป็นต้นไป ใช้รูปแบบของวัน-เวลาเป็น Generalized Time

Subject

ชื่อของผู้ที่เป็นเจ้าของใบรับรองสิทธิ์ ซึ่งจะใช้รูปแบบของ Distinguished Name (DN) ตามมาตรฐาน X.500 เช่น c=TH, o=GOV, cn=Santipap Naraupakarn เป็นตัวอย่างของ DN ซึ่งบ่งบอกถึงชื่ออของผู้ที่เป็นเจ้าของใบรับรองฯ (สันติภาพ นราอุปการ) ที่อยู่ภายใต้หน่วยงานภาครัฐในประเทศไทย

Subject Public Key Info

กุญแจสาธารณะของผู้ที่เป็นเจ้าของใบรับรองสิทธิ์ และอัลกอริทึมที่ใช้ในการสร้างกุญแจสาธารณะดังกล่าว ตัวอย่างอัลกอริทึม เช่น RSA Encryption, Digital Signature Algorithm เป็นต้น

2. ข้อมูลเพิ่มเติมของใบรับรองสิทธิ์(Certificate Extension)

โครงสร้างของข้อมูลเพิ่มเติมในใบรับรองสิทธิ์ประกอบด้วย 3 ส่วน คือ ชนิดของข้อมูล (ExtensionType) ความจำเป็นของข้อมูล (Extension Criticality) และ ค่าของข้อมูล (Extension Value) ดังรูปที่ 2

Type	Criticality	Value

รูปที่ 2-27 โครงสร้างของข้อมูลเพิ่มเติมในใบรับรองสิทธิ์

ในฟิลด์ชนิดของข้อมูล จะบอกถึงชนิดของข้อมูลที่อยู่ในฟิลด์ค่าของข้อมูล อาทิ ข้อความ ตัวเลข วันที่ เป็นต้น โดยที่ฟิลด์ความจำเป็นของข้อมูล จะใช้ในการบ่งบอกถึงความจำเป็นของข้อมูลเพิ่มเติมในใบรับรองสิทธิ์ที่ใช้งานร่วมกับแอปพลิเคชัน (Application) ถ้าฟิลด์นี้กำหนดว่ามีความจำเป็น แสดงว่าข้อมูลเพิ่มเติมนี้มีความสำคัญ ดังนั้นแอปพลิเคชันที่มีการใช้งานใบรับรองฯ ที่มีการระบุความจำเป็น จะต้องทำการอ่านค่าและประมวลผลค่าของข้อมูลดังกล่าว เนื่องจากบางแอปพลิเคชันมีความจำเป็นที่จะต้องใช้ข้อมูลเพิ่มเติมพิเศษ ดังนั้น การกำหนดความจำเป็นสำหรับข้อมูลเพิ่มเติมที่อยู่ในใบรับรองฯ ก็เพื่อป้องกันการใช้งานในทางที่ไม่ถูกต้องและความไม่ปลอดภัยของใบรับรองฯ

ข้อมูลเพิ่มเติมในใบรับรองสิทธิ์ประกอบด้วย 2 ประเภท ดังนี้

1. Standard Extensions

ข้อมูลเพิ่มเติมที่เป็นมาตรฐานของใบรับรองสิทธิ์ ประกอบด้วยข้อมูลต่างๆ ดังนี้

Authority Key Identifier

ฟิลด์ Authority key identifier ระบุถึงกุญแจสาธารณะที่เป็นคู่กับกุญแจส่วนตัวที่ผู้ให้บริการออกใบรับรองใช้ในการลงลายมือชื่อดิจิทัลกำกับใบรับรองสิทธิ์ เพื่อช่วยในการตรวจสอบลายมือชื่อดิจิทัลในใบรับรองฯในกรณีที่ผู้ให้บริการออกใบรับรองมีคู่กุญแจหลายคู่

Subject Key Identifier

ฟิลด์ Subject key identifier ระบุถึงกุญแจสาธารณะในใบรับรองสิทธิ์ ในกรณีที่ผู้ใช้งานใบรับรองฯ มีการเปลี่ยนคู่กุญแจในการใช้งาน ดังนั้นฟิลด์นี้จะช่วยในการถอดรหัสข้อมูลที่เคยถูกเข้ารหัสด้วยกุญแจสาธารณะเดิม

Key Usage

ฟิลด์ Key usage ระบุถึงวัตถุประสงค์ในการนำกุญแจไปใช้งาน เช่น การนำไปใช้ในการลงลายมือชื่อดิจิทัล การเข้ารหัสข้อมูล เป็นต้น

Private Key Usage Period

ฟิลด์ Private key usage period ระบุถึงช่วงอายุการใช้งานของกุญแจส่วนตัวที่เป็นคู่กับกุญแจสาธารณะในใบรับรองสิทธิ์

Certificate Policies

ฟิลด์ Certificate policies ระบุถึงนโยบายของผู้ให้บริการออกใบรับรองที่ใช้ในการออกใบรับรองสิทธิ์โดยที่ฟิลด์นี้จะอยู่ในใบรับรองฯ ของผู้ใช้และผู้ให้บริการออกใบรับรอง

Policy Mappings

ฟิลด์ Policy mappings ระบุถึงนโยบายของผู้ให้บริการออกใบรับรอง โดยที่ฟิลด์นี้จะอยู่ในใบรับรองฯ ของผู้ให้บริการออกใบรับรอง

Subject Alternative Name

ฟิลด์ Subject alternative name ระบุถึงชื่อของใบรับรองสิทธิ์ในกรณีที่มีชื่อที่แตกต่างกันมากกว่า 1 ชื่อซึ่งอาจจะใช้ที่อยู่ไปรษณีย์อิเล็กทรอนิกส์ (E-mail Address) ไอพีแอดเดรส (IP Address) ชื่อโดเมน (Domain Name) ในการบ่งบอกถึงชื่อของใบรับรองฯ เป็นต้น

Issuer Alternative Name

ฟิลด์ Issuer alternative name ระบุถึงชื่อของผู้ให้บริการออกใบรับรอง ในกรณีที่มีชื่อที่แตกต่างกันมากกว่า 1 ชื่อ ซึ่งข้อมูลในฟิลด์นี้มีลักษณะเหมือนกับฟิลด์ subject alternative name

Subject ไดเรกทอรี Attributes

ฟิลด์ Subject ไดเรกทอรี attributes ระบุถึงข้อมูลเพิ่มเติมที่อยู่ในไดเรกทอรี X.500 ซึ่งเป็นส่วนหนึ่งในชื่อของใบรับรองสิทธิ์

Basic Constraints

ฟิลด์ Basic constraints ระบุถึงประเภทของใบรับรองสิทธิ์ว่าเป็นของผู้ใช้หรือผู้ให้บริการออกใบรับรองและระบุถึงจำนวนชั้นสูงสุดของห่วงโซ่ใบรับรองฯ (Certificate Chain) ที่ถูกทำการรับรองต่อกันเป็นทอดๆ

Name Constraints

ฟิลด์ Name constraints ปรากฏอยู่ในใบรับรองสิทธิ์ของผู้ให้บริการออกใบรับรอง ซึ่งทำให้ผู้ดูแลระบบสามารถที่จะกำหนดชื่อโดเมนที่ใช้ในการในการมอบความไว้วางใจกับผู้ให้บริการออกใบรับรองรายอื่น

Policy Constraints

ฟิลด์ Policy constraints ปรากฏอยู่ในใบรับรองสิทธิ์ของผู้ให้บริการออกใบรับรอง ซึ่งทำให้ผู้ดูแลระบบสามารถที่จะกำหนดชุดของนโยบายของผู้ให้บริการออกใบรับรอง สำหรับห่วงโซ่ใบรับรองฯ ที่ถูกทำการรับรองต่อกันเป็นทอดๆ โดยผู้ให้บริการออกใบรับรอง

Extended Key

ฟิลด์ Extended key ระบุถึงวัตถุประสงค์ในการนำกุญแจไปใช้งาน ซึ่งเป็นวัตถุประสงค์ที่นอกเหนือจากฟิลด์ key usage

CRL Distribution Points

ฟิลด์ CRL distribution points ระบุถึงวิธีการในการเข้าถึงรายการเพิกถอนใบรับรอง (Certificate Revocation List - CRL) ของผู้ให้บริการออกใบรับรอง

Inhibit Any-Policy

ฟิลด์ inhibit any-policy ระบุถึงนโยบายพิเศษของใบรับรองสิทธิ์ที่นอกเหนือจากนโยบายอื่นๆ ของใบรับรองฯ

Freshest CRL

ฟิลด์ Freshest crl ระบุถึงวิธีการในการเข้าถึงรายการเพิกถอนใบรับรอง (CRL) รายการล่าสุดของผู้ให้บริการออกใบรับรอง

2. Internet Certificate Extensions

ข้อมูลเพิ่มเติมที่ใช้สำหรับการเข้าถึงข้อมูลแบบออนไลน์ ประกอบด้วย 2 ส่วน ดังนี้

Authority Information Access

ฟิลด์ Authority information access ระบุถึงวิธีการในการเข้าถึงข้อมูลและบริการของผู้ให้บริการออกใบรับรองสำหรับใบรับรองสิทธิ์ของผู้ให้บริการออกใบรับรองที่มีการใช้ฟิลด์นี้

Subject Information Access

ฟิลด์ Subject information access ระบุถึงวิธีการในการเข้าถึงข้อมูลและบริการ สำหรับใบรับรองสิทธิ์ที่มีการใช้ฟิลด์นี้ ซึ่งจะรวมถึงนโยบายของผู้ประกอบรับรอง ในกรณีที่เป็นใบรับรองฯ ของผู้ให้บริการออกใบรับรอง

บทที่ 3

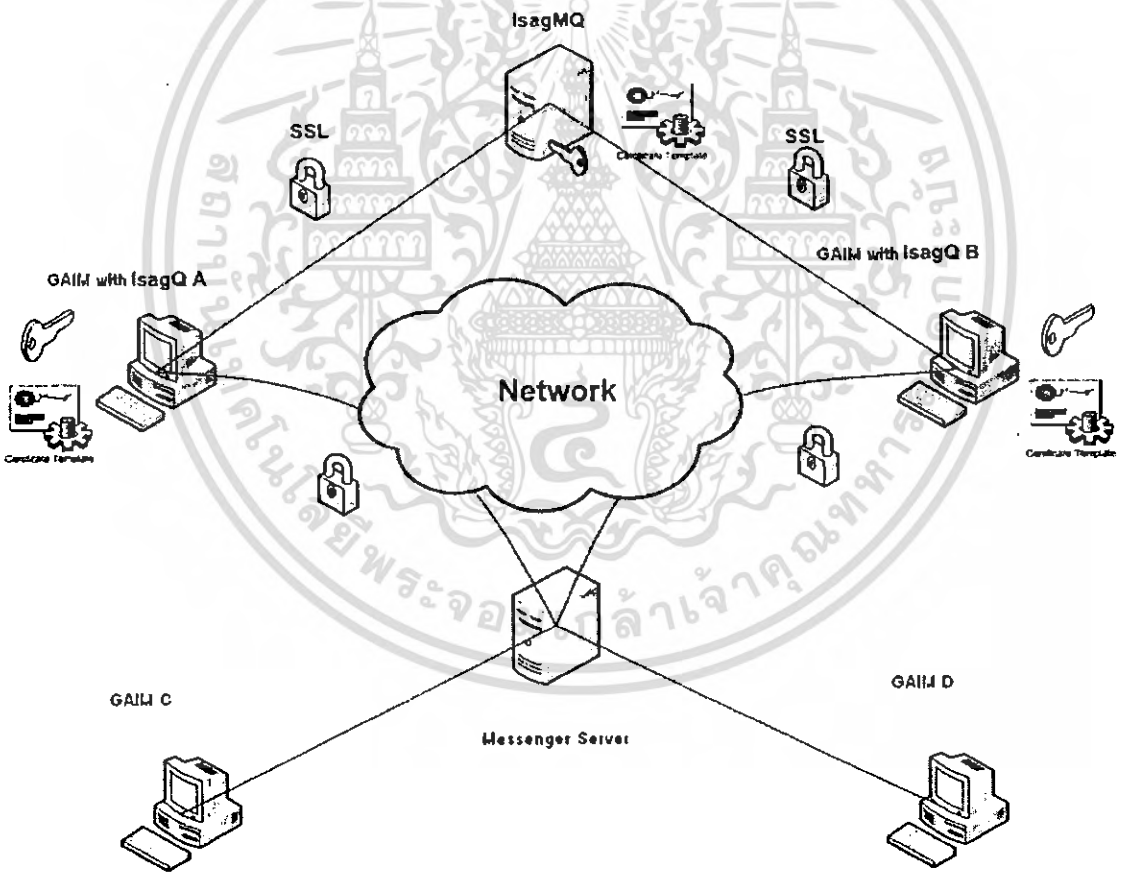
การออกแบบโครงงาน

3.1. บทนำ

ในบทนี้จะกล่าวถึงภาพโดยรวมของโครงงานที่พัฒนาในปีนี้เทียบกับโครงงานในปี 2548 รวมถึงการออกแบบโครงงานในปี 2549 นี้ โดยการเปรียบเทียบในด้านการให้บริการใบรับรองสิทธิ์ และการพิสูจน์ตัวตน

3.2 การออกแบบซอฟต์แวร์

3.2.1. การออกแบบ IsagQ และ IsagMQ ในปี 2548



รูปที่ 3-1 รูปแสดงการโครงสร้างการทำงานระหว่าง IsagQ และ IsagMQ

IsagQ และ IsagMQ เป็นเพียงผู้ให้บริการด้านความปลอดภัยเท่านั้น ส่วนการบริการด้านข้อความจะเป็นภาระหน้าที่ของโปรแกรม Gaim โดย IsagQ และ IsagMQ มีฟังก์ชันการทำงานดังนี้

IsagMQ (โปรแกรมฝั่งแม่ข่าย)

1. ออกใบรับรองสิทธิ์ ให้กับ ผู้ใช้
2. จัดทำฐานข้อมูลของผู้ใช้ที่ลงทะเบียนแล้ว
3. สามารถให้บริการลงทะเบียนและยกเลิกการลงทะเบียนได้
4.สามารถเรียกดูใบรับรองสิทธิ์ได้
5.สามารถถอดถอนใบรับรองสิทธิ์ได้

IsagQ (โปรแกรมฝั่งลูกข่าย)

1. สามารถให้บริการลงทะเบียนและยกเลิกการลงทะเบียนได้
2. ผู้ใช้ติดต่อสื่อสารระหว่างกันอย่างปลอดภัยโดยมีการเข้ารหัสข้อมูลสำหรับข้อมูลระหว่างIsagQ ด้วยกัน
3. ผู้ใช้สามารถติดตั้งบนระบบปฏิบัติการ Windows หรือ Linux ได้

3.2.2 การออกแบบโปรแกรมในปี 2549

จะทำการพัฒนาส่วนขยายของโปรแกรม Gaim ขึ้นมาอีกโปรแกรมหนึ่งที่ชื่อ AnubisQ เพื่อทำหน้าที่ในกระบวนการพิสูจน์ตนด้วยชีวมาตรเป็นหลัก โดยที่โปรแกรมส่วนขยายเดิมสามารถทำงานร่วมกับโปรแกรมส่วนขยายที่สร้างขึ้นมาใหม่ได้อย่างไม่มีปัญหา

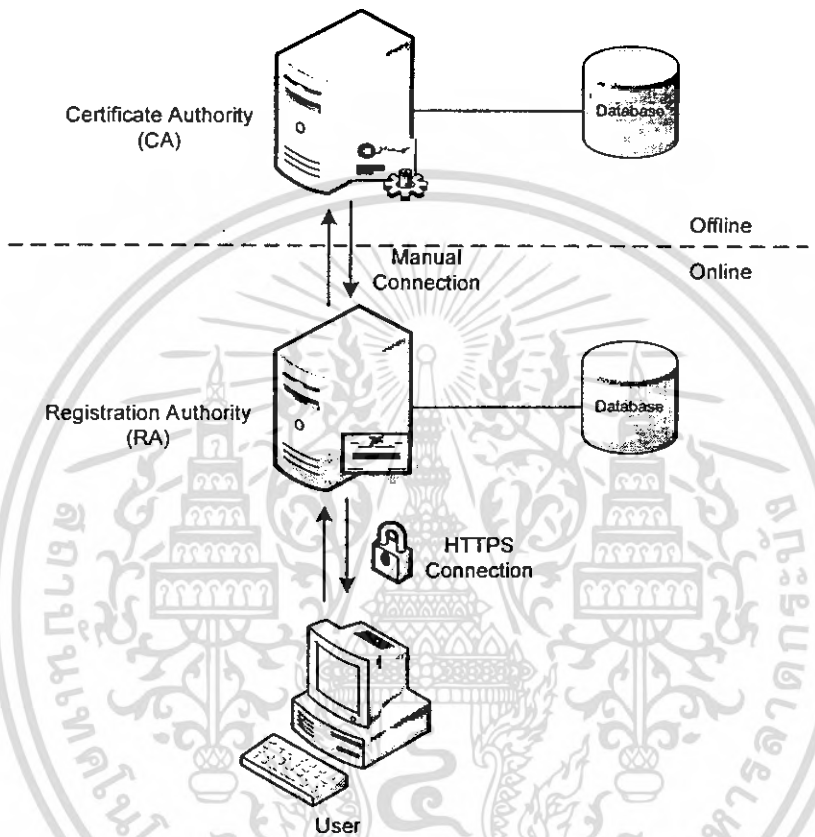
นอกจากนี้จะได้นำการใช้งาน OpenLDAP ซึ่งเป็นบริการไดเรกทอรีเซอร์วิซ และ OpenCA ซึ่งเป็นระบบ PKI เข้ามาใช้งานในโครงงานนี้ด้วย

การพัฒนาในส่วนของ AnubisQ แบ่งเป็นขั้นตอนหลักๆ ได้ดังนี้

1. ศึกษาความรู้ที่เกี่ยวข้อง
   ได้แก่ วิธีการพัฒนาส่วนขยายของโปรแกรม Gaim และ ทฤษฎีลายนิ้วมือ
2. ออกแบบโครงสร้างและส่วนติดต่อกับผู้ใช้งาน
   ดำเนินการออกแบบโครงสร้างของส่วนขยาย AnubisQ โดยประกอบด้วยสองส่วนหลักคือ ส่วนที่เป็นหน้าจอปรับแต่งค่า และ ส่วนที่ทำหน้าที่ติดต่อกับอุปกรณ์อ่านลายนิ้วมือ
3. พัฒนาสิ่งที่ได้ออกแบบด้วยภาษา C
   ทำการเขียนส่วนของหน้าจอติดต่อผู้ใช้ด้วยโปรแกม Glade และใช้ VeriFinger SDK ในกระบวนการที่เกี่ยวกับการอ่านลายนิ้วมือ
4. ทดสอบและแก้ไขข้อผิดพลาด

การออกแบบโครงสร้างพื้นฐานกุญแจสาธารณะ

เนื่องจากโครงงานนี้เป็นระบบต้นแบบในการนำโครงสร้างพื้นฐานกุญแจสาธารณะ มาใช้งานร่วมกับ ระบบรับส่งสารด่วน ผู้จัดทำจึงออกแบบโครงสร้างไว้ดังรูปที่ 3-2



รูปที่ 3-2 แสดงภาพรวมของโครงสร้างพื้นฐานกุญแจสาธารณะในโครงงานนี้

จากรูป โครงสร้างดังกล่าวจะประกอบด้วย 3 ส่วนหลักคือ

1.Certificate Authority (CA)

เป็นเวบเซิร์ฟเวอร์ทำหน้าที่เป็นผู้ให้บริการออกใบรับรองสิทธิ์

2. Registration Authority (RA)

เราจะแบ่งออกเป็น 2 องค์ประกอบย่อย คือ ส่วน RA Operator และส่วน Public (pub)

2.1 RA Operator

เป็นเวบเซิร์ฟเวอร์ทำหน้าที่ตรวจสอบใบร้องขอใบรับรองสิทธิ์ และเก็บใบรับรองสิทธิ์ที่ออกโดย CA เพื่อให้บริการแก่ผู้ใช้ นอกจากนี้ยังเป็นส่วนติดต่อกับ CA อีกด้วย

2.2 Public

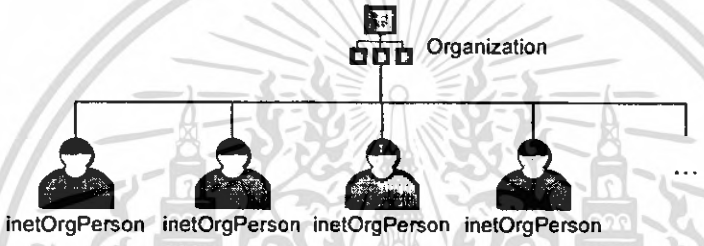
เป็นเวบเซิร์ฟเวอร์ที่ผู้ใช้ติดต่อมายัง RA Operator

3. User

เป็นผู้ร้องขอและใช้ใบรับรองสิทธิ์

จะเห็นได้ว่า ส่วน CA จะเป็นส่วนที่ไม่ได้เชื่อมต่อกับเครือข่ายภายนอก เนื่องจากเหตุผลด้านความปลอดภัยในการรักษากุญแจส่วนตัวของ CA ฉะนั้น การติดต่อระหว่าง CA และ RA จะเป็นในลักษณะการเชื่อมต่อด้วยการส่งผ่านข้อมูลด้วยผู้ดูแลระบบเอง (Manually) โดยให้ผู้ใช้งานติดต่อกับ RA เพียงส่วนเดียวผ่าน Protocol HTTPS

การออกแบบโครงสร้างของไดเรกทอรีเซอร์วิซ



รูปที่ 3-3 แสดงโครงสร้างไดเรกทอรีเซอร์วิซที่ใช้ในโครงงานนี้

โดยมีรายละเอียดแต่ละ Entry ในรูปแบบ ldif ดังนี้

Organization

dn: o=AnubisQ

objectClass: top

objectClass: inetOrgPerson

cn: admin

o: AnubisQ

inetOrgPerson

dn: cn=[ชื่อ email address], o=AnubisQ

objectClass: top

objectClass: inetOrgPerson

cn: [ชื่อ email address] #ต้องเหมือนกับชื่อในattribute dn

fno: 3 #จำนวนลายนิ้วมือใน entry

fingerprintData: [ข้อมูลลายนิ้วมือจากการอ่านครั้งที่ 1]

fingerprintData: [ข้อมูลลายนิ้วมือจากการอ่านครั้งที่ 2]

fingerprintData: [ข้อมูลลายนิ้วมือจากการอ่านครั้งที่ 3]

บทที่ 4
การพัฒนาชิ้นงานของโครงงาน

4.1 บทนำ

ในบทนี้จะกล่าวถึงการพัฒนาโปรแกรมซึ่งแบ่งออกได้เป็นสามส่วนคือ ส่วนขยาย AnubisQ ทำหน้าที่ในการพิสูจน์ตัวตนด้วยลายนิ้วมือ ส่วนขยาย IsagQ ทำหน้าที่ในการเข้ารหัสลับ / ถอดรหัสลับข้อความที่ผู้ใช้กำลังสนทนา และ เครื่องแม่ข่ายซึ่งเปิดบริการ LDAP และทำหน้าที่เป็น CA (Certificate Authority) ในขณะเดียวกัน

4.2 วิธีการพัฒนาส่วนขยายของโปรแกรม Gaim

ในการพัฒนาปลั๊กอินของGAIMแบ่งได้เป็น2ภาษาคือภาษาซีและภาษาเพิร์ลซึ่งในโครงงานนี้จะใช้ภาษาซีในการพัฒนาส่วนขยายต้องใช้ซอร์สโค้ดของ Gaim ในการคอมไพล์ โดยจะมี Gaim APIรองรับการทำงานต่างๆ และสามารถนำไลบรารีจากที่อื่นมาใช้ได้

4.2.1 การเตรียมสภาพแวดล้อมในการพัฒนาส่วนขยายของ Gaim

การเตรียมสภาพแวดล้อมในการพัฒนาส่วนขยายของ Gaim มีขั้นตอนดังนี้

1. ดาวน์โหลดซอร์สโค้ดของ Gaim เวอร์ชัน 1.5
2. เข้าไปที่โฟลเดอร์ Gaim-1.5.0 แล้วสั่ง ./configure
3. ในขั้นนี้เราจะได้ Make file ขึ้นมา
4. ถ้าต้องการincludeไลบรารีใดๆที่ต้องใช้ในส่วนขยายให้เข้าไปที่Gaim-1.5.0/plugins/ Makefile เข้าไปให้ระบุในไฟล์นี้ เช่น เราต้องการใช้ไลบรารีของ Openssl ซึ่งจะเก็บไว้ที่ /usr/local/include ก็เพิ่ม –I/usr/local/include
5. เมื่อต้องการคอมไพล์

   5.1 Window ใช้คำสั่ง make –f makefile.mingw **name**.dll

   5.2 Unix/linux ใช้คำสั่ง make **name**.so

   ซึ่ง name คือชื่อไฟล์ .c ที่ต้องการคอมไพล์
6. เมื่อเสร็จสิ้นคำสั่ง make ให้คัดลอกไฟล์ไปยังโฟลเดอร์ที่เก็บไลบรารีของ Gaim ไว้เช่น

   /usr/X11R6/lib/gaim หรือ บน Window คือ C:\Program Files\Gaim\plugins

4.2.2 ต้นแบบในการพัฒนาส่วนขยาย

ทุกๆส่วนขยายของโปรแกรม Gaim จำเป็นต้องมีส่วนของโค้ดที่เป็นแบบมาตรฐานเดียวกัน ซึ่งรายละเอียดของโค้ดส่วนดังกล่าวนั้น ได้กล่าวไว้ในบทที่ 2 หัวข้อที่ 2.3 ว่าด้วยการพัฒนาส่วนของโปรแกรม Gaim เป็นที่เรียบร้อยแล้ว

4.3 การพัฒนาส่วนขยาย AnubisQ

ส่วนขยาย AnubisQ สามารถแบ่งการพัฒนาออกเป็น 4 ขั้นตอนตามหน้าที่การทำงานได้ดังนี้

4.3.1 การอ่านข้อมูลลายนิ้วมือ

การพัฒนาในส่วนนี้จะเป็นการพัฒนาให้ส่วนขยาย AnubisQ สามารถทำการติดต่อกับอุปกรณ์อ่านลายนิ้วมือได้

- การติดต่อกับอุปกรณ์อ่านลายนิ้วมือ

  จะใช้ฟังก์ชัน init() ในการติดต่อกับอุปกรณ์อ่านลายนิ้วมือ

  ตัวอย่างการใช้งาน

```
struct scanner_info* scanner = &scanner_AFS4000;
int i = scanner -> init();
int dpi = scanner -> dpi;
char *name = scanner -> name;
char *imageFromScanner;
int w1,h1;
```

- การอ่านข้อมูลจากลายนิ้วมือ

  จะใช้ฟังก์ชัน read() ในการสั่งให้อุปกรณ์ทำการอ่านลายนิ้วมือ

  ตัวอย่างการใช้งาน

```
imageFromScanner = scanner -> read(&w1, &h1);
```

หมายเหตุ : ฟังก์ชันทั้งสองนี้สามารถเรียกใช้ได้จะต้องทำการ include ไฟล์ scanner.h เสียก่อน

4.3.2 การตรวจสอบข้อมูลลายนิ้วมือ

การพัฒนาในส่วนนี้จะเป็นการพัฒนาให้ส่วนขยาย AnubisQ สามารถทำการตรวจสอบลายนิ้วมือได้ โดยรับข้อมูลมาจากเครื่องแม่ข่ายที่เปิดบริการ LDAP

- การกำหนดค่าพารามิเตอร์ที่ใช้ในการตรวจสอบ

  ก่อนที่เราจะทำการตรวจสอบข้อมูลลายนิ้วมือ เราต้องทำการกำหนดค่าพารามิเตอร์บางอย่างเสียก่อนนั่นคือการกำหนดองศาสูงสุดของมุมที่จะหมุนภาพไปได้

  จะใช้ฟังก์ชัน VFSetParameter() ในการกำหนดค่าพารามิเตอร์

  ตัวอย่างการใช้งาน

```
ret = VFSetParameter(VFP_MAXIMAL_ROTATION, (INT)rotation,
NULL);
if (ret != VFE_OK){
    gaim_debug(GAIM_DEBUG_INFO,
                "AnubisQ","VFSetParameter error");
    return;
}
```

- การอ่านข้อมูลลายนิ้วมือ

  สำหรับการอ่านข้อมูลลายนิ้วมือในส่วนของการตรวจสอบจะเหมือนกับการอ่านข้อมูลลายนิ้วมือ ดังนั้นจะไม่ขอกล่าวถึงในส่วนนี้อีก

- การ Generalize ข้อมูลลายนิ้วมือ

  การ Generalize ข้อมูลลายนิ้วมือ คือการตรวจสอบข้อมูลลายนิ้วมือในเบื้องต้นว่าสามารถนำใช้ในกระบวนตรวจสอบได้หรือไม่ เช่น เจอจุดสำคัญที่ใช้ในการ ตรวจสอบครบหรือไม่ หรือ มีจุดสำคัญเป็นจำนวนเท่าไหร่ เป็นต้น

  จะใช้ฟังก์ชัน VFGeneralize() ในการทำกระบวนการ Generalization

  ตัวอย่างการใช้งาน

```
result = VFGeneralize (VF_GENERALIZE_COUNT,feats,
                        featuresFromLDAPServer, &size,0);
    if(result<0)
            gaim_debug(GAIM_DEBUG_INFO,
                    "AnubisQ","Generalization failed: %d\n",result);
    else
            gaim_debug(GAIM_DEBUG_INFO,
                    "AnubisQ","Generalization succeeded: %d\n",result);
```

- การตรวจสอบข้อมูลลายนิ้วมือ

  การตรวจสอบข้อมูลลายนิ้วมือในส่วนขยาย AnubisQ จะทำงานโดยการ นำข้อมูลลายนิ้วมือที่ได้มาจากเครื่องแม่ข่ายที่เปิดบริการ LDAP ไว้ แล้วนำมาเปรียบเทียบกับข้อมูลลายนิ้วมือที่ได้มาจากอุปกรณ์อ่านลายนิ้วมือ จะใช้ฟังก์ชัน VFVerify() ในการตรวจสอบ ตัวอย่างการใช้งาน

```
result = VFVerify(featuresFromLDAPServer, featuresFromScanner, &md,
0);
    switch (result) {
            case VFE_OK:
                    gaim_debug(GAIM_DEBUG_INFO,
                                    "AnubisQ","Fingerprint matched \n");
                    fingerprint_result = 1;
                    VFFinalize();
            break;
            case VFE_FAILED:
                    gaim_debug(GAIM_DEBUG_INFO,
                                    "AnubisQ","Fingerprint mismatched \n");
                    fingerprint_result = 0;
                    VFFinalize();
                    break;
            default:
                    VFFinalize();
            break;
    }
```

4.3.3 การส่งและรับข้อมูลจากเครื่องแม่ข่ายที่เปิดบริการ LDAP

การพัฒนาในส่วนนี้จะเป็นการพัฒนาให้ส่วนขยาย AnubisQ สามารถทำการส่งข้อมูลลายนิ้วมือหลังจากการอ่านลายนิ้วมือเสร็จ ไปเก็บไว้ยังเครื่องแม่ข่ายที่เปิดบริการ LDAP ในรูปแบบของไดเรกทอรีเซอร์วิซ และดึงข้อมูลลายนิ้วมือจากเครื่องแม่ข่ายกลับมาใช้ในกระบวนการตรวจสอบได้

การเตรียมความพร้อมในการใช้งานบริการ LDAP

เนื่องจากบริการ LDAP จะทำงานในลักษณะโครงสร้างแบบต้นไม้ คือ มีส่วนบนสุด (root) จากนั้นแตกย่อยลงมาเรื่อยๆ ดังนั้นเราจึงจำเป็นต้องกำหนดค่าบางอย่างให้เรียบร้อยก่อนที่จะใช้งาน

ในส่วนขยาย AnubisQ ได้ทำการ include ไฟล์ ldapValue.h ซึ่งภายในไฟล์นี้จะกำหนดค่าเริ่มต้นบางอย่างไว้ดังนี้

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <ldap.h>

 // Host name of LDAP server
#define MY_HOST          "isag23.ce.kmitl.ac.th"

//Port number where LDAP and LDAPS servers ars runninge
#define     MY_PORT              LDAP_PORT
#define     MY_SSL_PORT          LDAPS_PORT

// DN of directory manager entry.
#define MGR_DN           "cn=admin,o=AnubisQ"

// Password for manager DN.
#define MGR_PW           "secret"
```

นอกจากการกำหนดค่าเบื้องต้นแล้ว เรายังจำเป็นต้องเรียกใช้ฟังก์ชันบางฟังก์ชันก่อนที่จะทำการเพิ่ม / ดึง / ลบ ข้อมูลลายนิ้วมือที่เก็บอยู่บนเครื่องแม่ข่าย

จะใช้ฟังก์ชัน ldap_init() ในการเรียกใช้งานฟังก์ชัน LDAP ครั้งแรก

ตัวอย่างการใช้งาน

```
if ( (ld = ldap_init( MY_HOST, MY_PORT )) == NULL ) {
    perror( "ldap_init" );
    return( 1 );
  }
```

- การส่งข้อมูลลายนิ้วมือไปเก็บไว้บนเครื่องแม่ข่าย

  หลังจากที่เราทำการอ่านข้อมูลลายนิ้วมือจากผู้ใช้แล้ว ขั้นตอนต่อไปก็คือ ส่วนขยาย AnubisQ จะนำข้อมูลลายนิ้วมือดังกล่าวไปเก็บไว้บนเครื่องแม่ข่ายที่เปิดบริการ LDAP ไว้

  จะใช้ฟังก์ชัน ldapadd() ในการเพิ่มข้อมูลลายนิ้วมือลงบนเครื่องแม่ข่าย

  หมายเหตุ : สำหรับในส่วนขยาย AnubisQ จะไม่ใช้ฟังก์ชัน ldapadd() ในการเพิ่มข้อมูลลายนิ้วมือ แต่จะใช้การเรียกเชลล์ขึ้นมาทำงานโดยผ่านคำสั่ง system แทน แล้วค่อยสั่งเพิ่มข้อมูลลายนิ้วมือผ่านเชลล์อีกทีหนึ่ง

  ตัวอย่างการใช้งาน (ผ่านเชลล์)

```
sprintf(tmp,"ldapadd -D \'cn=admin,o=AnubisQ\' -w \"secret\" -x
<<EOF\ndn: %s\nobjectClass: top\nobjectClass: inetOrgPerson\ncn:
%s\nfno: 3\nfingerprintData: %s\n",dn,email_name,
base64_encode(imageFromScanner1));
  strcat(cmd,tmp);
  sprintf(tmp,"fingerprintData:
%s\n",base64_encode(imageFromScanner2));
  strcat(cmd,tmp);
  sprintf(tmp,"fingerprintData:
%s\n",base64_encode(imageFromScanner3));
  strcat(cmd,tmp);
  gaim_debug(GAIM_DEBUG_INFO, "AnubisQ", "+++++%s\n",cmd);

  system(cmd);
```

- การดึงข้อมูลลายนิ้วมือจากเครื่องแม่ข่ายมาใช้งาน

การดึงข้อมูลลายนิ้วมือจากเครื่องแม่ข่ายมาใช้งานนั้นจะปรกอบด้วยการทำงานย่อยอีก 2 อย่างด้วยกันคือ การค้นหาข้อมูลลายนิ้วมือของผู้ใช้ที่ต้องการ และ การดึงข้อมูลนั้นมาใช้งาน

จะใช้ฟังก์ชัน ldap_search_s () ในการค้นหาข้อมูลลายนิ้วมือที่ต้องการ

ตัวอย่างการใช้งาน

```
if ( ldap_search_s( ld, dn_name, LDAP_SCOPE_BASE,
        "(objectclass=*)", attrs, 0, &result ) != LDAP_SUCCESS ) {
    ldap_perror( ld, "ldap_search_s" );

    return( 1 );
}
```

จะใช้ฟังก์ชัน ldap_get_values() ในการดึงข้อมูลลายนิ้วมือ

ตัวอย่างการใช้งาน

```
if (( vals = ldap_get_values( ld, e, "fingerprintData" )) != NULL ) {
    for ( i = 0; vals[i] != NULL; i++ ) {
        gaim_debug(GAIM_DEBUG_INFO,
                "AnubisQ","\t%s\n", vals[i] );
    }
}
```

- การลบข้อมูลลายนิ้วมือที่เก็บอยู่บนเครื่องแม่ข่าย

กระบวนการทำงานในส่วนนี้จะคล้ายคลึงกับการดึงข้อมูลลายนิ้วมือมาใช้งาน ประกอบด้วยสองส่วนการทำงานย่อย คือ การค้นหาข้อมูลลายนิ้วมือที่ต้องการ และการลบข้อมูลลายนิ้วมือ

จะใช้ฟังก์ชัน ldap_delete() ในการลบข้อมูลลายนิ้วมือ โดยระบุชื่อ dn ที่ต้องการลบ

ตัวอย่างการใช้งาน

```
dn="cn=vipassu@hotmail.com,o=AnubisQ";
if (( msgid = ldap_delete( ld, dn )) < 0 ) {
    ldap_perror( ld, "ldap_delete" );
```

```
    return( 1 );
  }
```

- การเข้ารหัส / ถอดรหัสข้อมูลลายนิ้วมือในรูปแบบของ base64

เนื่องจากข้อมูลลายนิ้วมือที่อ่านมาจากอุปกรณ์นั้นอยู่ในรูปแบบของข้อมูลที่เป็นไบนารี่ แต่ข้อมูลที่จะสามารถเก็บอยู่บนเครื่องแม่ข่ายได้ต้องเป็นในลักษณะของข้อความ (Text Data) ดังนั้นเราจึงต้องแปลงข้อมูลไบนารี่เหล่านั้นเป็นข้อมูลข้อความเสียก่อน ก่อนที่จะนำไปเก็บลงเครื่องแม่ข่าย จะใช้ฟังก์ชัน base64_encode() ในการเข้ารหัสข้อมูลให้อยู่ในรูปแบบของ base64 โดยส่งพารามิเตอร์เป็นชนิดข้อมูลไบนารี่

ตัวอย่างการใช้งาน

```
base64_encode(imageFromScanner1)
```

จะใช้ฟังก์ชัน base64_decode() ในการถอดรหัสข้อมูลให้อยู่ในรูปแบบของไบนารี่ โดยส่งพารามิเตอร์เป็นชนิดข้อมูลข้อความแบบ base64

ตัวอย่างการใช้งาน

```
base64_decode(imageFromServer1)
```

- การเพิ่มความปลอดภัยในการส่งข้อมูลลายนิ้วมือไปเครื่องแม่ข่าย

เนื่องจากในการส่ง/ดึง ข้อมูลลายนิ้วมือ ไป/จาก เครื่องแม่ข่าย อาจมีผู้ไม่หวังดีทำการลักลอบดักข้อมูล แก้ไขข้อมูลลายนิ้วมือ หรือ อาจจะทำการแกล้งไม่ให้ผู้ใช้สามารถเข้าใช้งานได้ จึงมีความจำเป็นที่จะต้องพิจารณาถึงความปลอดภัยของข้อมูลลายนิ้วมือ ดังนั้นเราจึงพัฒนาให้ส่วนขยาย AnubisQ รับส่งข้อมูลลายนิ้วมือผ่านโพรโตคอล TLS (Transport Layer Security)

จะใช้ฟังก์ชัน ldap_start_tls_s() เพื่อบอกให้ LDAP รับส่งข้อมูลผ่านช่องทางการติดต่อที่ปลอดภัย (Secured Channel)

ตัวอย่างการใช้งาน

```
ldap_set_option(ld,LDAP_OPT_PROTOCOL_VERSION,&ldap_vers);    //
ldap_vers = LDAP_VERSION3
rc = ldap_start_tls_s(ld, NULL, NULL);
  if (rc != LDAP_SUCCESS) {
```

```
    fprintf(stderr, "ldap_start_tls_s(): %d: %s\n", rc, ldap_err2string(rc));
    ldap_unbind(ld);
    return;
}
```

4.3.4 การเพิ่มหน้าจอปรับแต่งค่า

การพัฒนาในส่วนนี้จะเป็นการเพิ่มหน้าจอปรับแต่งค่าของส่วนขยาย AnubisQ โดยสามารถเรียกใช้งานได้จากเมนู Preference ของโปรแกรม Gaim ดังแสดงในรูปที่ 1

รูปที่ 1 หน้าจอปรับแต่งค่าของส่วนขยาย AnubisQ

- การออกแบบหน้าจอปรับแต่งค่า

สำหรับการออกแบบหน้าจอ (User Interface) จะอาศัยโปรแกรม Glade ซึ่งเป็นโปรแกรมที่ช่วยในการสร้างหน้าจอ ดังรูปที่ 2

รูปที่ 2 โปรแกรม Glade

รูปที่ 3 การสร้างหน้าจอปรับแต่งค่า

- การเพิ่มหน้าจอปรับแต่งค่าลงในโปรแกรม Gaim

หลังจากที่ออกแบบส่วนของหน้าจอปรับแต่งค่าเสร็จสิ้นแล้ว ขั้นต่อไปเราก็จะทำการเพิ่มหน้าจอปรับแต่งค่าที่เราได้ออกแบบ ลงไปในโปรแกรม Gaim ซึ่งหน้าจอนี้จะอยู่ในเมนู Preference ของโปรแกรม Gaim

ส่วนขยายที่ต้องการให้มีหน้าจอปรับแต่งค่า จะต้องดำเนินการตามนี้

1. แก้ไขส่วนของ GaimPluginInfo

```
static GaimPluginInfo info =
{
  GAIM_PLUGIN_MAGIC,
        GAIM_MAJOR_VERSION,
        GAIM_MINOR_VERSION,
        GAIM_PLUGIN_STANDARD,
  GAIM_GTK_PLUGIN_TYPE,
        0,
        NULL,
        GAIM_PRIORITY_DEFAULT,
        "anubisq",
        "AnubisQ",
```

```
        "0.72",
        "Biometric authentication by fingerprint.",
        N_("Use your password combine with your fingerprint to sign-
in."),
        "[ CAN ] <vipassu@hotmail.com>",
        "http://webserv.kmitl.ac.th/~sapiv/",
        plugin_load,
        NULL,
        NULL,

        &ui_info,    // ใส่ชื่อฟังก์ชันที่จะเรียกใช้ฟังก์ชันสร้างหน้าจอ
ปรับแต่งค่า
        NULL,
        NULL,
        NULL//actions
};
```

2. สร้างฟังก์ชันที่จะใช้เรียกฟังก์ชันสร้างหน้าจอปรับแต่งค่า

```
static GaimGtkPluginUiInfo ui_info =
{
        get_config_frame
};
```

3. สร้างฟังก์ชันสร้างหน้าจอปรับแต่งค่า (สร้างโดยโปรแกรม Glade)

```
static GtkWidget* get_config_frame (GaimPlugin *plugin)
{

 GtkWidget *window1;
 GtkWidget *fixed1;
 GtkWidget *add_btn;
```

```
 tooltips = gtk_tooltips_new ();


 window1 = gtk_fixed_new();
 fixed1 = gtk_fixed_new ();
 gtk_widget_show (fixed1);
 gtk_container_add (GTK_CONTAINER (window1), fixed1);


 add_btn = gtk_button_new ();
 gtk_widget_show (add_btn);
 gtk_fixed_put (GTK_FIXED (fixed1), add_btn, 104, 344);
 gtk_widget_set_size_request (add_btn, 88, 32);
 gtk_tooltips_set_tip (tooltips, add_btn, _("Add a new user's fingerprint
data"), NULL);
NULL);
 // Event listener
 g_signal_connect(G_OBJECT(add_btn),                    "clicked",
G_CALLBACK(show_add_window), NULL);
 gtk_widget_show_all(window1);


 return window1;
}
```

หมายเหตุ : ซอร์สโค้ดนี้เป็นซอร์สโค้ดฉบับย่อใช้เพื่อยกตัวอย่างเท่านั้น ไม่สามารถนำไปใช้งานได้จริง เนื่องจากซอร์สโค้ดฉบับเต็มมีความยาวมาก

4.4 การนำ OpenLDAP มาใช้งาน

OpenLDAP เป็นโปรแกรมโอเพ่นซอร์สที่ให้บริการ LDAP (http://www.openldap.org) ประกอบด้วยสองส่วนที่สำคัญคือ เครื่องแม่ข่าย (LDAP Server) และเครื่องลูกข่าย (LDAP Client) โดยการทำงานจะเป็นในลักษณะที่เครื่องลูกข่าย (LDAP Client) ทำการติดต่อกับเครื่องแม่ข่ายได้ 2 วิธีเท่านั้น โปรแกรมอื่นไม่สามารถติดต่อกับเครื่องแม่ข่ายได้โดยตรง คือ การเรียกใช้คำสั่งผ่านเชลล์ หรือ การใช้งานผ่านไลบรารี่ของ LDAP เท่านั้น

สำหรับในส่วนขยาย AnubisQ จะใช้ทั้งสองวิธีการในการติดต่อกับเครื่องแม่ข่าย (LDAP Server)

4.4.1. การติดตั้ง OpenLDAP

- ส่วนของเครื่องแม่ข่ายที่เปิดให้บริการ LDAP (LDAP Server)
  จะต้องติดตั้งโปรแกรม *openldap-servers* เพื่อให้บริการ Directory Service แก่ AnubisQ และ openldap-clients เพื่อให้ผู้ดูแลระบบสามารถเพิ่ม / ลบ / แก้ไข ข้อมูลในไดเรกทอรีเซอร์วิซโดยผ่านทางเชลล์พร้อมท์ได้
- ส่วนของส่วนขยาย AnubisQ (LDAP client)
  ติดตั้งโปรแกรม *openldap-clients* เพียงอย่างเดียว

4.4.2. การกำหนดค่าก่อนการใช้งาน

- ส่วนของเครื่องแม่ข่ายที่เปิดให้บริการ LDAP (LDAP Server)
  เราสามารถกำหนดค่าต่างๆ ได้ในไฟล์ /etc/openldap/slapd.conf ซึ่งจะมีการ include ไฟล์สกีมา (schema) ที่ใช้งานเข้ามาด้วย โดยจะมีค่าเริ่มต้นบางส่วนมาให้แล้ว แต่ถ้าเราต้องการจัดเก็บข้อมูลรูปแบบใหม่ เราสามารถสร้าง เพิ่มเติม แก้ไข โครงสร้างนี้ได้ เพื่อให้เหมาะสมกับการทำงานของระบบที่สร้างขึ้น

ในโครงงานนี้เราใช้ Entry อยู่ 2 แบบคือ Entry ของ Root และ Entry สมาชิก (ดังรูปที่ 8) โดยใช้คลาส inetOrgPerson เป็นหลัก และจะเพิ่ม Attribute ที่เหมาะสมในคลาส inetOrgPerson ดังนี้

NAME	DESCRIPTION	OID	EQUALITY	SUBSTR	SYNTAX	Value
fno	Number of fingerprintData	2.16.840. 1.113730. 3.1.5	CaseIgnore Match	caseIgnore Substrings Match	1.3.6.1.4.1. 1466.115.1 21.1.15	SINGLE -VALUE
fingerprint Data	each fingerprint data	2.16.840. 1.113730. 3.1.6	CaseIgnore Match	caseIgnore Substrings Match	1.3.6.1.4.1. 1466.115.1 21.1.15	-

ตารางที่ 4-1 แสดงรายละเอียด Attribute ที่เพิ่มในคลาส inetOrgPerson

ผู้ดูแลระบบต้องเพิ่ม Entry ของ Root เพียงครั้งเดียว ก่อนที่ส่วนขยายAnubisQ จะทำการเพิ่ม Entry สมาชิก

โดยค่าเริ่มต้นแล้ว LDAP server และ LDAP client จะรับส่งข้อมูลในรูปแบบข้อความธรรมดา ผ่าน Port 389 ซึ่งผู้ไม่หวังดีอาจลักลอบดักฟังได้ ซึ่งสามารถแก้ปัญหานี้ได้ 2 วิธี คือ

1. สร้างความปลอดภัยในการรับส่งข้อมูล โดยใช้ TLS เข้ารหัสลับเป็นข้อมูลที่ไม่สามารถเข้าใจได้ก่อนรับส่งข้อมูลดังกล่าวผ่านProtocol LDAP (Port 389) ซึ่งหากทดลองดักจับข้อมูลด้วยโปรแกรม Ethereal จะแจ้งว่าเป็น Packet ที่มีความผิดพลาด
2. สร้างความปลอดภัยในการรับส่งข้อมูล โดยใช้ Protocol LDAPS (Port 636)

หมายเหตุ : สำหรับโครงงานนี้ใช้วิธีแรกในการแก้ปัญหา

หมายเหตุ : เราต้องระบุชื่อเครื่อง (hostname) ในส่วน cn (common namc) ให้ถูกต้องในการสร้างใบร้องขอรับรองสิทธิ์ เนื่องจาก LDAP server จะตรวจสอบความถูกต้องกับชื่อเครื่องจริงด้วย

- ส่วน AnubisQ (LDAP client)

การเรียกใช้คำสั่งผ่านเชลล์ เราต้องกำหนดค่าต่างๆ ที่จำเป็นในไฟล์ /etc/openldap/ldap.conf ก่อน โดยคำสั่งดังกล่าวจะเรียกใช้ฟังก์ชันการทำงานใน Library ldap.h อีกทีหนึ่ง

หมายเหตุ : หากเราไม่ได้กำหนดค่าในส่วน TLS_CACERT ข้อมูลที่รับส่งกันจะถูกส่งในลักษณะข้อความธรรมดา ซึ่งผู้ไม่หวังดีสามารถอ่านได้ง่าย

4.4.3. การเพิ่ม Entry ที่ต้องการ

การเพิ่ม Entry ของ Root จะใช้คำสั่ง ldapadd ซึ่งเป็น utility ของ openldap-client โดยมีตัวเลือกที่จำเป็นดังนี้

-D "[rootdn]"	ระบุ rootdn ของระบบ เช่น –D "cn=admin,o=AnubisQ"
-w "[password]"	ระบุรหัสผ่านของ rootdn เช่น –w "secret"
-x	ใช้การพิสูจน์ตัวตนอย่างง่ายแทน SASL
-f [path of ldif format file]	ระบุ path ของไฟล์ldif เช่น -f /etc/openldap/myOrg.ldif

ตารางที่ 4-2 แสดงรายละเอียดตัวเลือกที่สามารถใช้งานร่วมกับคำสั่ง ldapadd

ตัวอย่างการเพิ่ม Entry ของ Root

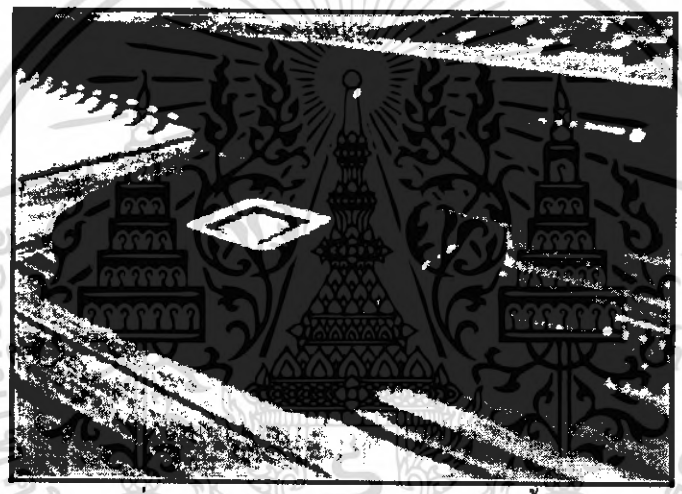
```
ldapadd –D "cn=admin,o=AnubisQ" –w "secret" –x -f /etc/openldap/myOrg.ldif
```

4.5 คุณลักษณะทั่วไปของอุปกรณ์อ่านลายนิ้วมือ

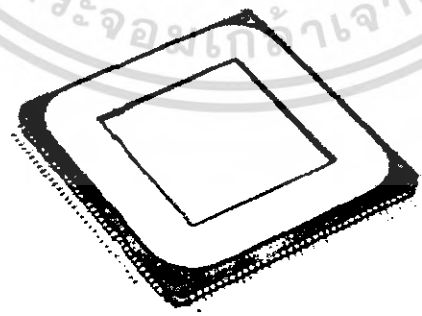
4.5.1 ซอฟต์แวร์ชุดพัฒนา (Verifinger4.2 Linux Software Develop Kit)

เป็นชุดพัฒนาสำหรับผู้พัฒนาระบบชีวมาตร เพื่อความรวดเร็วในการพัฒนาโปรแกรมประยุต์ด้วยชีวมาตร โดยเรียกใช้งานฟังก์ชันต่างๆจากไลบรารี (VeriFinger DLL) ทำให้มีความน่าเชื่อถือในการตรวจสอบลายนิ้วมือทั้งแบบ VeriFinger (1:1) และแบบ Identification (1: N) ทั้งนี้ฟังก์ชัน SDK สามารถใช้ในการติดต่อกับอุปกรณ์สแกนลายนิ้วมือใดๆ ฐานข้อมูลใดๆ และยูสเซอร์อินเทอร์เฟสใดๆ ก็ได้

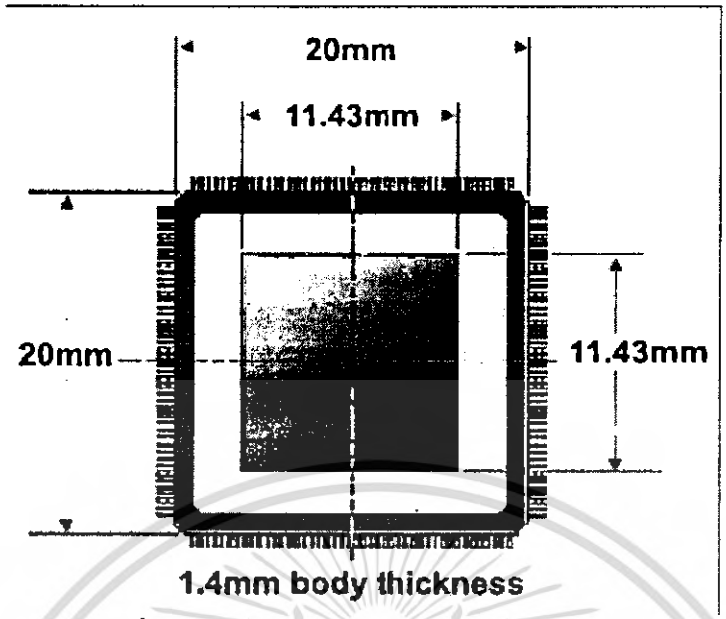
4.5.2 อุปกรณ์สแกนลายนิ้วมือ (AES4000 EntrePad (USB)



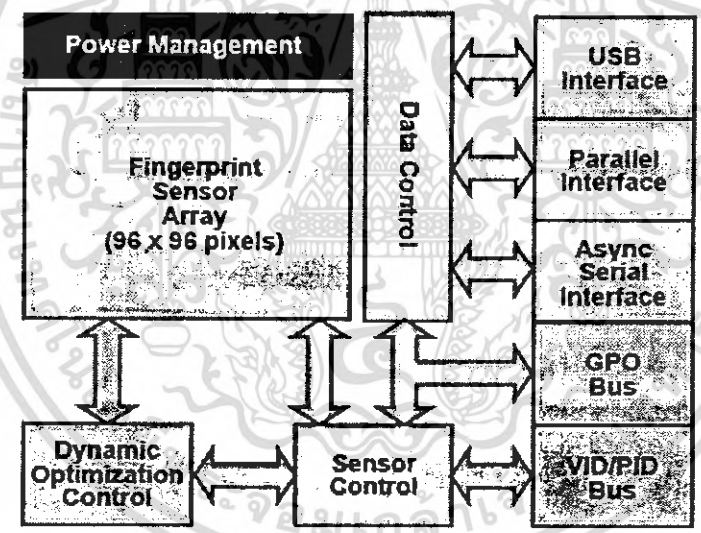
รูปที่ 4-1 แสดงอุปกรณ์การสแกนลายนิ้วมือ



รูปที่ 4-2 แสดง AES4000 Entre PAD



รูปที่ 4-3 แสดงขนาดของ Entre PAD AES4000



รูปที่ 4-4 แสดงความสัมพันธ์ระหว่าง Entre PAD AES4000 กับส่วนต่างๆ

Sensor Name	AES4000 EntréPad
Manufacturer	AuthenTec,Inc
Resolution	250 dpi
Sensor Size	20x 20x 1.5 mm (0.79"x 0.79"x 0.06")
Image capture area	11.43x 11.43 mm (0.45"x 0.45")
Supported OS	MS Windows, Linux
Operating Voltage Range	3.3V or 5.0V
Commercial Temp. Range	0°C through +70°C
High-Rate Image Capture	Up to 52 frames/second
ESD Resistance	IEC 61000-4-2 Level 3 (±8KV)

ตารางที่ 4-3 แสดงคุณสมบัติของอุปกรณ์สแกนลายนิ้วมือ

4.6 การพัฒนาส่วนขยาย IsagQ

การเขียนหน้าจอการตั้งค่าส่วนขยาย

เมื่อปลั๊กอินโหลดขึ้นมาเราสามารถที่จะสร้างหน้าจอการตั้งค่าได้โดยจะระบุไว้ในฟังก์ชัน GaimPluginInfo info ในอาร์กิวเมนต์ตัวที่ 19 โดยระบุเป็น pointer เช่น &ui_info ซึ่งเรียกฟังก์ชัน ui_info มีลักษณะดังนี้

```
static GaimGtkPluginUiInfo ui_info =
{
        get_config_frame
};
```

ซึ่งจะไปเรียกฟังก์ชัน get_config_frame ซึ่ง ui_info จะเป็นเพียงฟังก์ชันที่บอก Gaim ว่ามีการสร้างหน้าจอการตั้งค่าส่วนขยายซึ่งฟังก์ชันที่สร้างหน้าจอการตั้งค่าจริง ๆ คือ get_config_frame

การดั้งค่าการทำงานเริ่มต้นของปลั๊กอิน

ก่อนที่ตัวปลั๊กอินจะเริ่มทำงานเราสามารถสั่งให้ตัวปลั๊กอินทำงานบางอย่างก่อนได้ โดยใช้ฟังก์ชัน plugin_load (GaimPlugin *handle) ซึ่ง IsagQ จะมีการทำงาน 2 อย่างในช่วงนี้คือ

Init_pref() = สร้าง preference ที่สามารถบันทึกค่าลักษณะต่างๆของปลั๊กอินเช่น

- gaim_prefs_add_none สร้าง preference
- gaim_prefs_add_int บันทึกค่าลงไปใน preference ที่สร้างแล้ว
- gaim_prefs_get_int รับค่าจาก preference ที่บันทึกไว้

Init_cert() = สร้างกุญแจสาธารณะ,กุญแจส่วนตัว,ใบร้องขอใบรับรองสิทธิ์,ใบรับรองสิทธิ์

สัญญานบน Gaim (Gaim signal)

เมื่อผู้ใช้หรือคู่สนทนามีการตอบโต้กับโปรแกรม(event)ตัว Gaim ก็จะส่งสัญญาณ (Signal) ออกมาโดยเราสามารถจับสัญญาณเหล่านี้ได้โดยใช้ gaim_signal_connect แต่ก่อนที่จะจับสัญญาณได้จะต้องสร้าง handle สำหรับเชื่อมต่อก่อน ในตัว IsagQ จะมีการจับสัญญาณการทำงาน 2 อย่างคือ สัญญานการส่งข้อความ "sending-im-msg" และสัญญาณการรับข้อความ "receiving-im-msg" ซึ่งมีรูปแบบการทำงานดังนี้

```
gaim_signal_connect(conv_handle, "sending-im-msg", isagq_plugin_handle,
        GAIM_CALLBACK(send_msg_cb), NULL);
```

ฟังก์ชันนี้จะมีอาร์กิวเมนต์ 5 ตัวคือ

- conv_handle คือ แฮนเดิลของการสทนาเนื่องจากฟังก์ชันที่จะทำงานนี้เกี่ยวข้องกับการสนทนา ซึ่ง แฮนเดิลนี้สามารถเรียกใช้ได้จาก gaim_conversations_get_handle
- "sending-im-msg" คือ ชื่อของสัญญาณที่เราต้องการจะจับ

- isagq_plugin_handle คือ แฮนเดิลของปลั๊กอินIsagQ ซึ่งจะถูกสร้างในฟังก์ชัน plugin_load
- GAIM_CALLBACK(send_msg_cb) คือ การเรียกใช้ฟังก์ชัน send_msg_cb เมื่อมีสัญญาณการส่งข้อความ
- NULL คือ อาร์กิวเมนต์ของฟังก์ชัน send_msg_cb

การจัดการกับการส่งข้อความและการรับข้อความ

เมื่อเราสามารถจับสัญญาณของการส่งข้อความและการรับข้อความได้แล้วก็จะสามารถจัดการกับข้อความเหล่านั้นก่อนที่จะแสดงผลได้ โดยฟังก์ชันที่จะจัดการกับข้อความจะต้องมีอาร์กิวเมนต์ต่อไปนี้

- GaimAccount *account อาร์กิวเมนต์นี้จะมีรายละเอียดของผู้ใช้งานที่ได้รับหรือส่งข้อความนั้นๆ เช่น ชื่อผู้ใช้,โพรโตคอลที่ใช้
- char *who อาร์กิวเมนต์นี้คือ ชื่อของคู่สนทนา
- char **message คือ ข้อความที่ได้รับหรือส่ง

เมื่อเราจัดการกับข้อความเสร็จแล้วก็จะสามารถนำส่งหรือแสดงผลได้ โดยการส่งข้อความจะใช้ serv_send_im(server-send-im)หรือถ้าต้องการส่งอย่างอื่นก็ได้เช่นserv_send_file serv_send_typing โดย serv_send_im มีอาร์กิวเมนต์ต่างๆดังนี้

- GaimConnection * คือ ตัวชี้ไปยังการเชื่อมต่อกับผู้สนทนาซึ่งจะเป็น attribute อยู่ในตัวแปรชนิด GaimAccount ชื่อ gc
- Const char* คือ ชื่อของผู้สนทนา
- Const char* คือ ข้อความที่จะต้องการส่ง
- GaimMessageFlags คือชนิดข้อความ เช่นGAIM_MESSAGE_SEND, GAIM_MESSAGE_SYSTEM

เช่น (acct->gc, name,cipher_msg,GAIM_MESSAGE_AUTO_RESP)

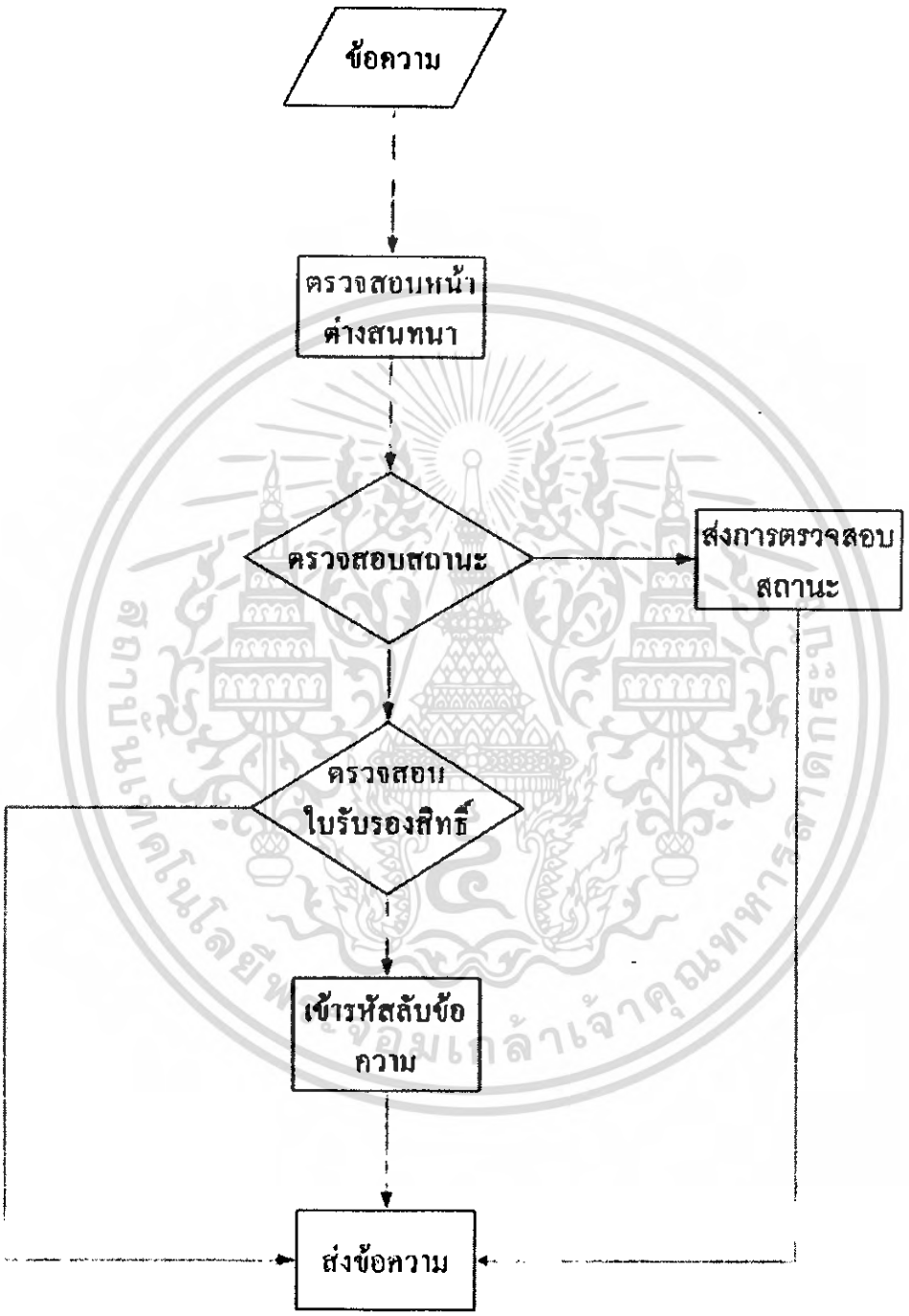
ทางด้านการแสดงผลข้อความก็จะใช้ gaim_conv_im_write โดยมีอาร์กิวเมนต์ดังนี้

- GaimConvIm คือ ตัวชี้ไปยังหน้าต่างสนทนานั้นๆ
- Const char* คือ ชื่อของผู้สนทนา
- Const char* คือ ข้อความที่จะแสดงผล
- GaimMessageFlags คือ ชนิดของข้อความ
- time_t คือ การหน่วงเวลาก่อนที่จะแสดงผล

เช่น gaim_conv_im_write(GAIM_CONV_IM(conv), *who, decrypt_msg, GAIM_MESSAGE_RECV, time((time_t)NULL))

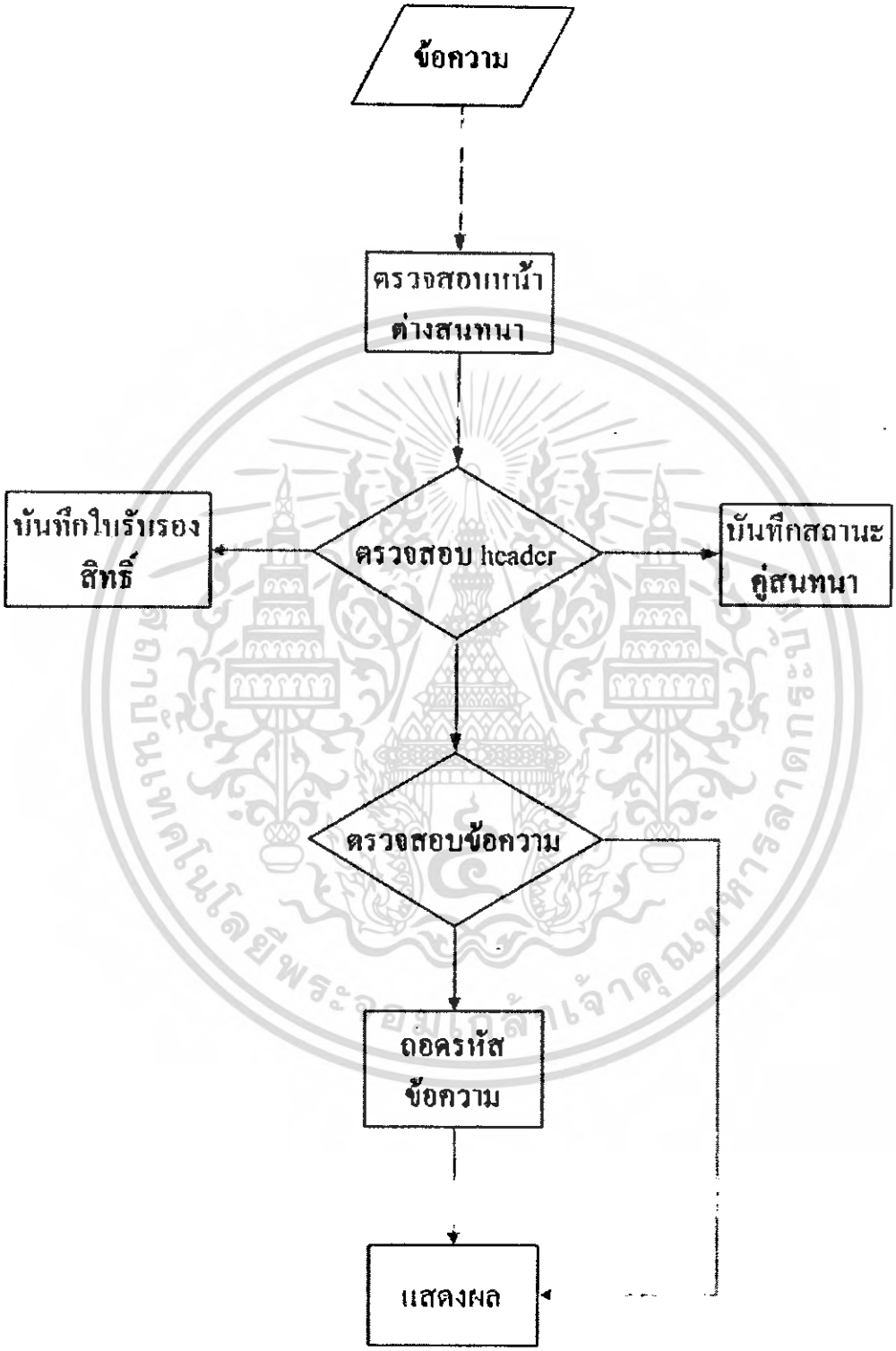
4.6.1 กระบวนการรับ-ส่งข้อความของ IsagQ

ขั้นตอนการส่งข้อความของ IsagQ



รูปที่ 4-5 ขั้นตอนการส่งการข้อความของ IsagQ

ขั้นตอนการรับข้อความของ IsagQ



รูปที่ 4-6 ขั้นตอนการรับข้อความของ IsagQ

4.7 การนำ OpenCA มาใช้งาน

OpenCA เป็นโปรแกรมโอเพ่นซอร์สที่ให้บริการโครงสร้างพื้นฐานกุญแจสาธารณะผ่าน Web Browser (http://www.openca.org) ซึ่งเราสามารถประยุกต์ใช้งานได้ตามโครงสร้างขององค์กรอย่างคล่องตัว เนื่องจากผู้พัฒนา OpenCA ออกแบบให้โปรแกรมมีความยืดหยุ่นสูง

จากโครงสร้างการทำงานในบทที่ผ่านมา เราจะแบ่งการทำงานของ OpenCA เป็นส่วน CA และ RA โดยมีขั้นตอนดังนี้

4.7.1 การติดตั้ง OpenCA

OpenCA ใช้โปรแกรม Open Source หลายโปรแกรมในการทำงาน เช่น Apache, mod_ssl, OpenSSL, OpenLDAP และ Perl ดังนั้นเราต้องติดตั้งโปรแกรมเหล่านี้ก่อน

หมายเหตุ : ในโครงงานนี้เราไม่ได้เก็บข้อมูลใบรับรองสิทธิ์ในไดเรกทอรีเซอร์วิซจึงไม่ใช้การทำงานในส่วน OpenLDAP แต่โปรแกรมอื่นๆ เราสามารถติดตั้งได้จากซีดี Fedora Core 5 ได้

4.7.2 การตั้งค่าระบบก่อนการใช้งาน

OpenCA ใช้คำสั่ง configure ในการคอมไพล์และตั้งค่าเริ่มต้นบางอย่าง รวมถึงคำสั่ง make ในการติดตั้งโปรแกรม เหมือนโปรแกรม Open Source ปกติทั่วไป แต่คำสั่ง configure ไม่ได้ตั้งค่าการทำงานของระบบที่ติดตั้งแล้ว ซึ่งการตั้งค่าการทำงานจะกล่าวในภายหลังต่อไป

ขั้นตอนการตั้งค่าระบบก่อนการใช้งาน

1. ดาวน์โหลด Source Code (openca-0.9.3-rc1.tar.gz) จาก http://www.openca.org และแตกไฟล์ดังกล่าว
2. คอมไพล์ Build และ ติดตั้ง ด้วยคำสั่ง configure [option], make และ make install-offline (สำหรับการติดตั้ง CA) หรือ make install-online (สำหรับการติดตั้ง RA)
3. กำหนดหน้า Homepage ที่ Web Server ให้เป็นหน้า Webpage ของ OpenCA

4.7.3 การตั้งค่าระบบการใช้งาน

- แก้ไขไฟล์การตั้งค่าการใช้งาน ทั้งส่วน CA และ RA ให้สามารถรับส่งข้อมูลระหว่างกันได้
- เปลี่ยนรหัสผ่านของเวบมาสเตอร์ทั้งส่วน CA และ RA
- ปิดการทำงานการล็อกเอาท์ในส่วน Public ของ RA

- สร้างไฟล์การตั้งค่าการใช้งานจากไฟล์ต้นแบบ (Template Configuration File) ทั้งส่วน CA และ RA

4.7.4 เริ่มการใช้งาน OpenCA

- รันคำสั่งให้ Apache ทำงาน
- รันคำสั่งให้ CA และ RA เริ่มทำงาน มีขั้นตอนย่อยดังนี้

กำหนดค่าเริ่มต้นให้กับ CA (Phase I)

- กำหนดค่าเริ่มต้นฐานข้อมูล
- สร้างกุญแจส่วนตัวของ CA
- สร้างใบร้องขอใบรับรองสิทธิ์ของ CA (โดยใช้กุญแจส่วนตัวในขั้นตอนที่แล้ว)
- สร้างใบรับรองสิทธิ์ของ CA โดย sign ด้วยกุญแจส่วนตัวของ CA เอง (Self Signed)
- สร้าง CA Chain

กำหนดค่าเริ่มต้นให้กับ RA (Phase II)

- สร้างใบร้องขอใบรับรองสิทธิ์ของ RA
- CA ออกใบรับรองสิทธิ์ให้กับ RA

- CA ส่งการกำหนดค่าการใช้งานของ CA ให้กับ RA
- CA ส่งข้อมูลทั้งหมดของ CA ให้กับ RA

ขั้นตอนการขอใบรับรองสิทธิ์

1. ผู้ใช้กรอกรายละเอียดการขอใบรับรองสิทธิ์ในบราวซ์เซอร์ (เช่น Mozilla Firefox หรือ Microsoft Internet Explorer)
5. Browser สร้าง Private Key และ ใบร้องขอใบรับรองสิทธิ์จากข้อมูลของผู้ใช้
6. Browser ส่งใบร้องขอใบรับรองสิทธิ์ไปยัง RA ผ่าน Protocol HTTPS
7. RA ตรวจสอบและรับรองใบร้องขอใบรับรองสิทธิ์
8. RA ส่งใบร้องขอใบรับรองสิทธิ์ให้ CA
9. CA ออกใบรับรองสิทธิ์ โดยใช้กุญแจส่วนตัวของ CA เซ็นรับรอง (sign)
10. CA ส่งใบรับรองสิทธิ์ให้ RA
11. RA ดาวน์โหลดใบรับรองสิทธิ์มาเก็บไว้ในฐานข้อมูล
12. ผู้ใช้ดาวน์โหลดใบรับรองสิทธิ์เก็บไว้ใน Browser

ขั้นตอนการนำใบรับรองสิทธิ์ไปใช้งาน

1. ผู้ใช้ Export ใบรับรองสิทธิ์ออกจาก Browser โดยกำหนดรหัสผ่านเพื่อเข้ารหัสกุญแจส่วนตัวแล้วบันทึกเก็บไว้ในไฟล์
2. ผู้ใช้เข้าโปรแกรม Gaim และเลือกไฟล์ใบรับรองสิทธิ์ดังกล่าว พร้อมทั้งระบุรหัสผ่านในขณะที่ Export ใบรับรองสิทธิ์
3. IsagQ จะแยกไฟล์ใบรับรองสิทธิ์ดังกล่าวเป็น ไฟล์กุญแจส่วนตัว และไฟล์ใบรับรองสิทธิ์ซึ่งไม่มีกุญแจส่วนตัว

บทที่ 5

การทดสอบการทำงานและผลการทดสอบ

5.1. บทนำ

สำหรับการทดสอบการทำงานจะแบ่งออกเป็นสองส่วนหลัก คือ ทดสอบการทำงานก่อนที่จะ Sign in เข้าใช้งานโปรแกรม Gaim กับ การใช้งานหลังจากการ Sign-in แล้ว

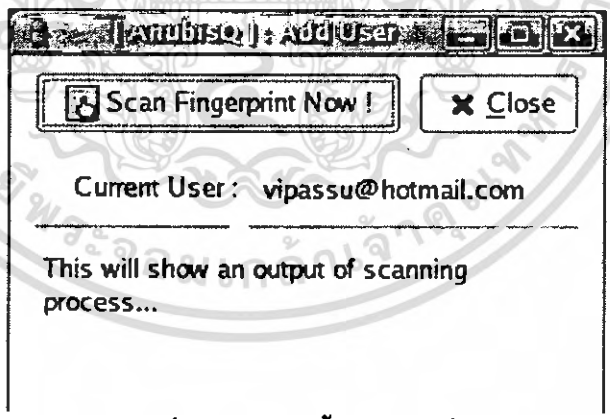
5.2. การทดสอบการทำงานก่อนการ Sign-in

จุดมุ่งหมายที่ต้องการ : สามารถ Sign-in เข้าใช้โปรแกรม Gaim โดยการใช้รหัสผ่านร่วมกับลายนิ้วมือได้

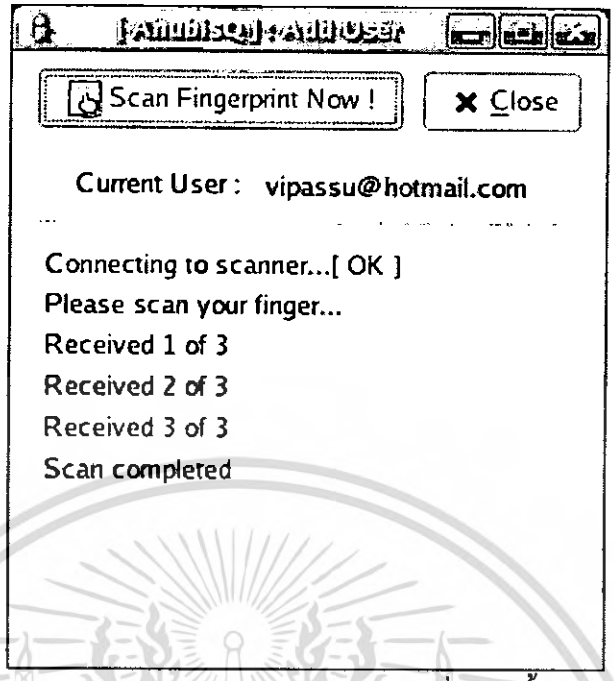
ในส่วนนี้จะเป็นการทดสอบการทำงานของส่วนขยาย AnubisQ ร่วมกับเครื่องแม่ข่ายที่เปิดบริการ LDAP แบ่งการทดสอบออกเป็น 4 ขั้นตอนดังนี้

5.2.1. การเพิ่มข้อมูลลายนิ้วมือ

การทดสอบในส่วนนี้คือ จะทดสอบว่าส่วนขยาย AnubisQ สามารถเพิ่มข้อมูลลายนิ้วมือผู้ใช้ที่ยังไม่เคยมีลายนิ้วมือมาก่อนได้ ขั้นตอนการทดสอบจะเป็นแบบตามลำดับดังรูปที่แสดง



รูปที่ 5-1 แสดงหน้าจอการเพิ่มข้อมูลลายนิ้วมือ โดยเมื่อกดปุ่ม "Scan Fingerprint Now!" โปรแกรมจะอ่านข้อมูลลายนิ้วมือเป็นจำนวนสามครั้ง



รูปที่ 5-2 แสดงหน้าจอหลังจากกระบวนการเพิ่มลายนิ้วมือเสร็จสิ้น

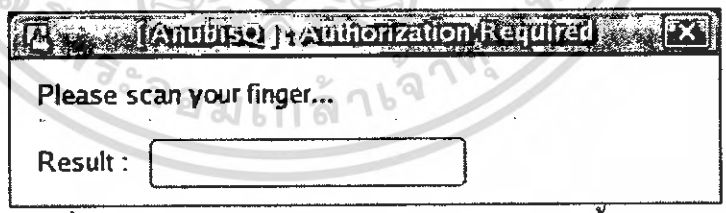
หมายเหตุ : ถ้าส่วนขยาย AnubisQ ไม่สามารถอ่านลายนิ้วมือได้ โปรแกรมจะขอให้ผู้ใช้ทำการอ่านลายนิ้วมือใหม่จนกว่าจะสามารถอ่านได้

รูปที่ 5-3 ข้อมูลที่เก็บอยู่ใน OpenLDAP เพื่อแสดงให้เห็นว่าข้อมูลลายนิ้วมือของผู้ใช้ถูกส่งไปเก็บไว้ที่เครื่องแม่ข่ายจริง

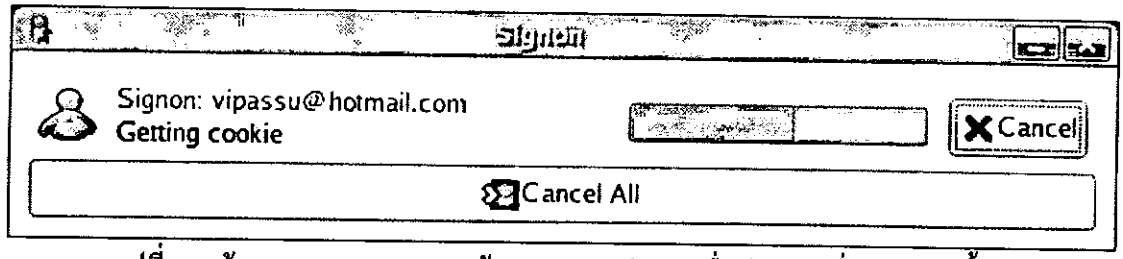
5.2.2. การตรวจสอบข้อมูลลายนิ้วมือ

การทดสอบในส่วนนี้คือ จะทดสอบว่าส่วนขยาย AnubisQ สามารถตรวจสอบข้อมูลลายนิ้วมือของผู้ใช้ได้ ในกระบวนการ Sign on ขั้นตอนการทดสอบจะเป็นแบบตามลำดับดังรูปที่แสดง

รูปที่ 5-4 หน้าจอโปรแกรม Gaim ขณะกดปุ่ม Sign on เพื่อเข้าใช้งานระบบ



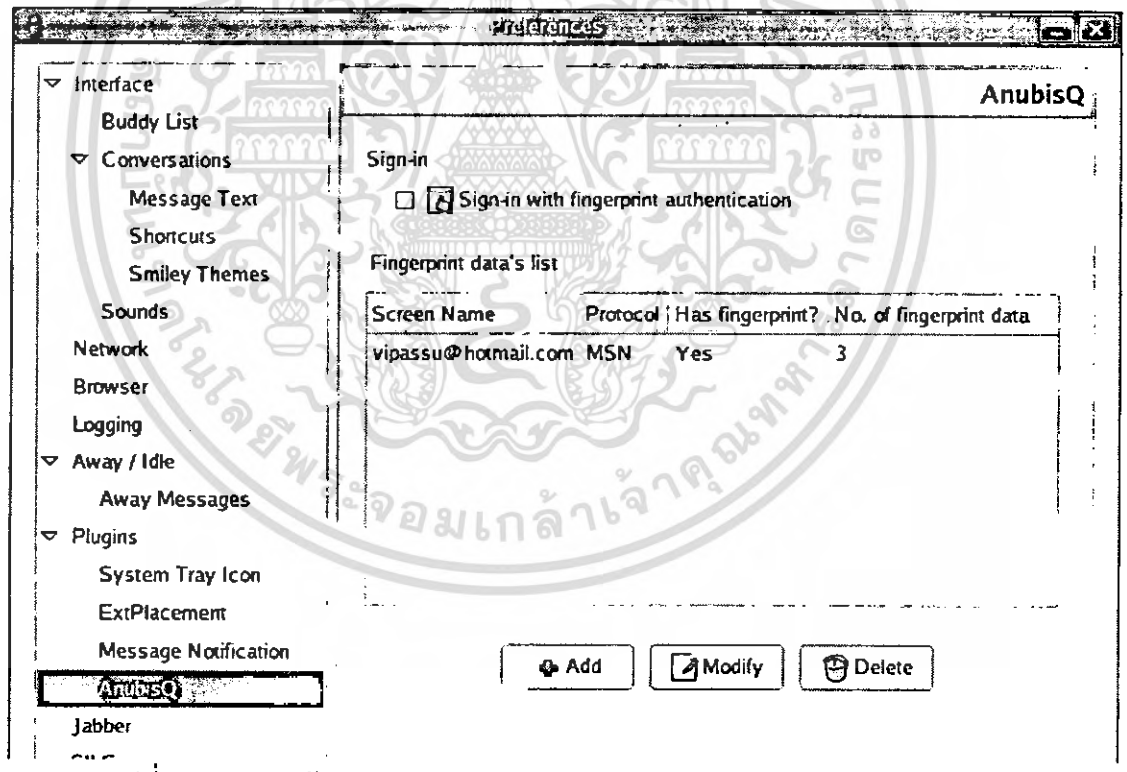
รูปที่ 5-5 แสดงหน้าจอขอให้ผู้ใช้ทำการสแกนลายนิ้วมือก่อนที่จะดำเนินการเชื่อมต่อกับเครื่องแม่ข่าย ในที่นี้คือ MSN Server



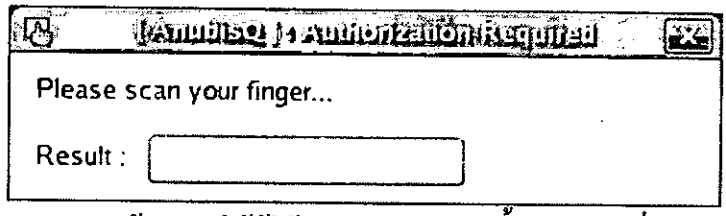
รูปที่ 5-6 ถ้าผลการตรวจสอบถูกต้องระบบจะทำการเชื่อต่อกับเครื่องแม่ข่ายนั้นๆ (ในที่นี่คือ MSN) และถ้าไม่ถูกต้องก็จะตัดการเชื่อมต่อทันที

5.2.3. การลบและแก้ไขข้อมูลลายนิ้วมือ

การทดสอบในส่วนนี้คือ จะทดสอบว่าส่วนขยาย AnubisQ สามารถแก้ไขและลบข้อมูลลายนิ้วมือของผู้ใช้ได้ ขั้นตอนการทดสอบจะเป็นแบบตามลำดับดังรูปที่แสดง

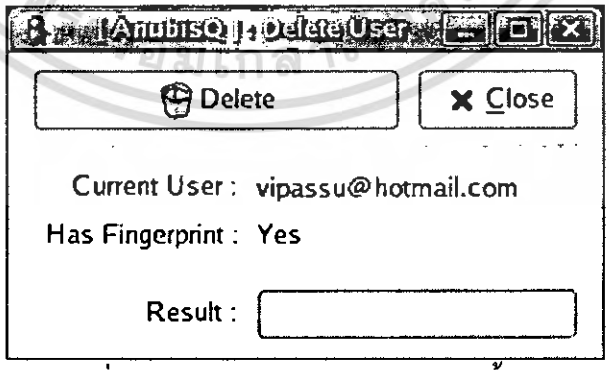


รูปที่ 5-7 แสดงหน้าจอปรับแต่งส่วนขยาย AnubisQ โดยสามารถทำการแก้ไขและลบข้อมูลลายนิ้วมือได้จากหน้าจอนี้

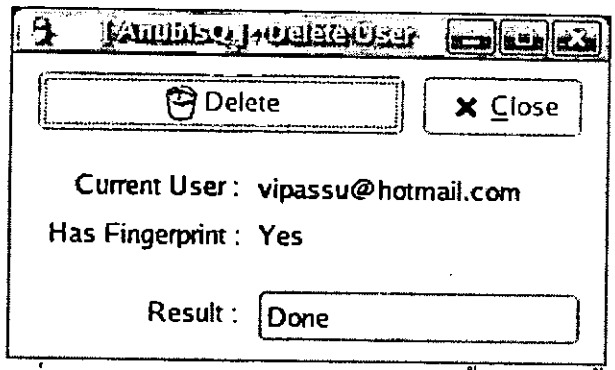


รูปที่ 5-8 แสดงหน้าจอขอให้ผู้ใช้ทำการสแกนลายนิ้วมือก่อน เพื่อป้องกันการแก้ไขจากบุคคลอื่น

รูปที่ 5-9 ถ้าผู้ใช้ทำการแก้ไขข้อมูลลายนิ้วมือ ส่วนขยาย AnubisQ จะทำการลบข้อมูลที่อยู่ในเครื่องแม่ข่ายให้แล้วจะขอให้ผู้ใช้บันทึกข้อมูลลายนิ้วมืออันใหม่



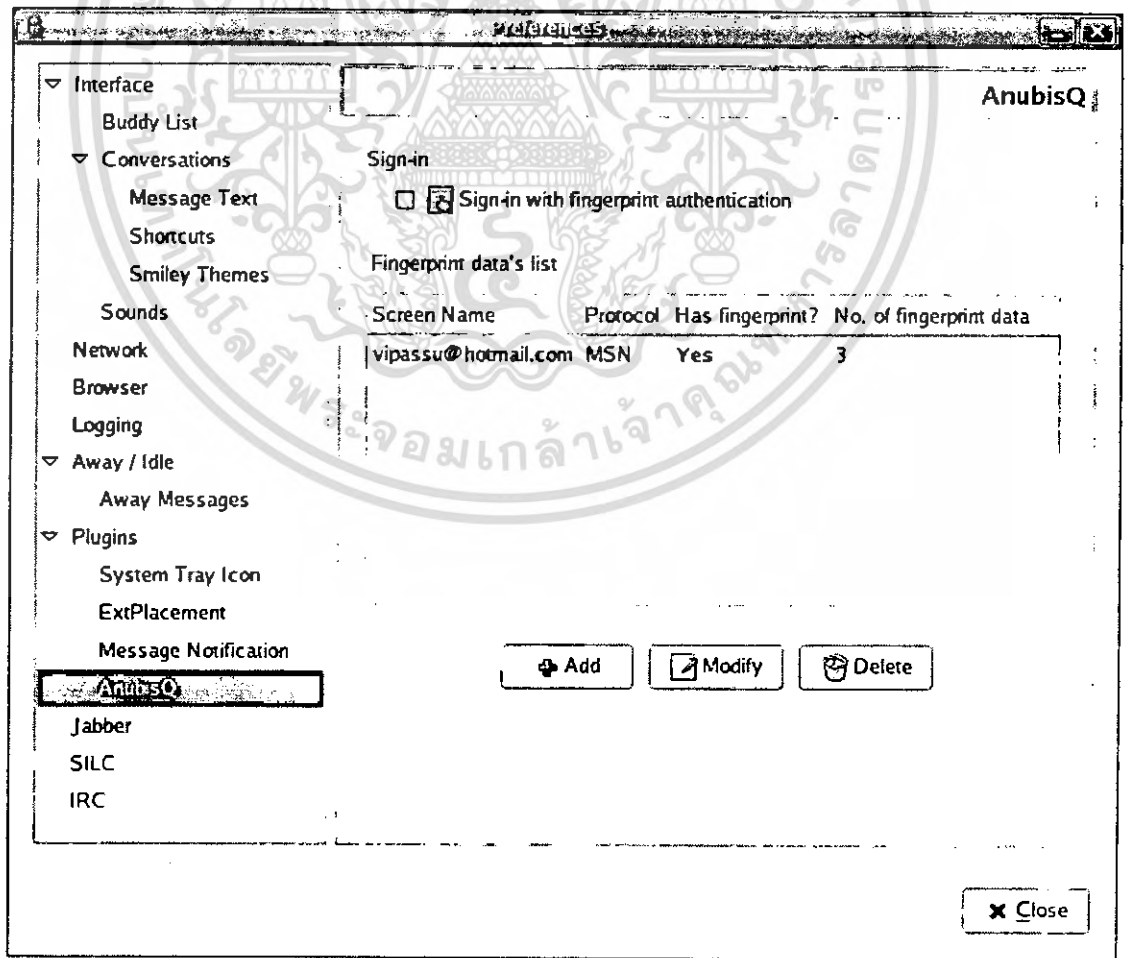
รูปที่ 5-10 ถ้าผู้ใช้ทำการลบข้อมูลลายนิ้วมือ จะปรากฏหน้าจอให้ผู้ใช้กดลบข้อมูลลายนิ้วมือ



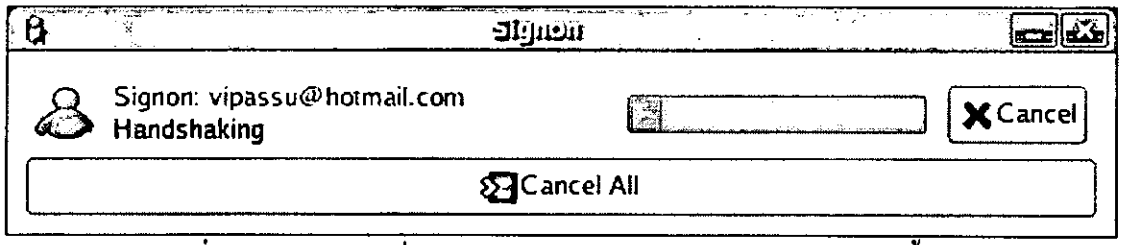
รูปที่ 5-11 หน้าจอแสดงการลบข้อมูลลายนิ้วมือเสร็จสิ้น

5.2.4. การใช้งานรหัสผ่านอย่างเดียว

การทดสอบในส่วนนี้คือ จะทดสอบว่าส่วนขยาย AnubisQ สามารถยกเลิกการใช้ลายนิ้วมือในกระบวนการ Sign on ได้ ขั้นตอนการทดสอบจะเป็นแบบตามลำดับดังรูปที่แสดง



รูปที่ 5-12 เลือกไม่ใช้การพิสูจน์ตนด้วยลายนิ้วมือ



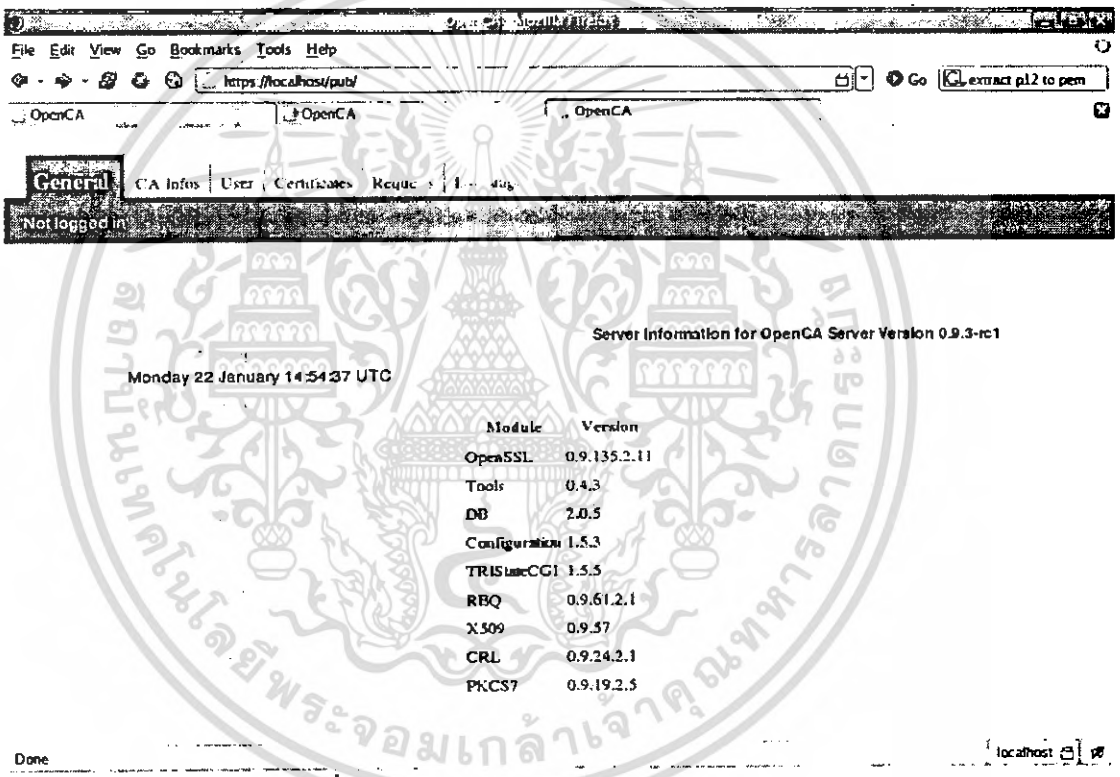
รูปที่ 5-13 แสดงการเชื่อมต่อตามปกติ ไม่ใช้การพิสูจน์ตนด้วยลายนิ้วมือ

จุดมุ่งหมายที่ต้องการ : สามารถขอใบรับรองสิทธิ์จาก OpenCA ซึ่งเป็นผู้ให้บริการ PKI ได้ และสามารถนำใบรับรองสิทธิ์นี้ไปใช้งานกับส่วนขยาย IsagQ หลังจากการ Sign-in ได้ ในส่วนนี้จะเป็นการทดสอบการทำงานของ OpenCA ร่วมกับส่วนขยาย IsagQ มีขั้นตอนดังต่อไปนี้

5.2.5 การขอใบรับรองสิทธิ์

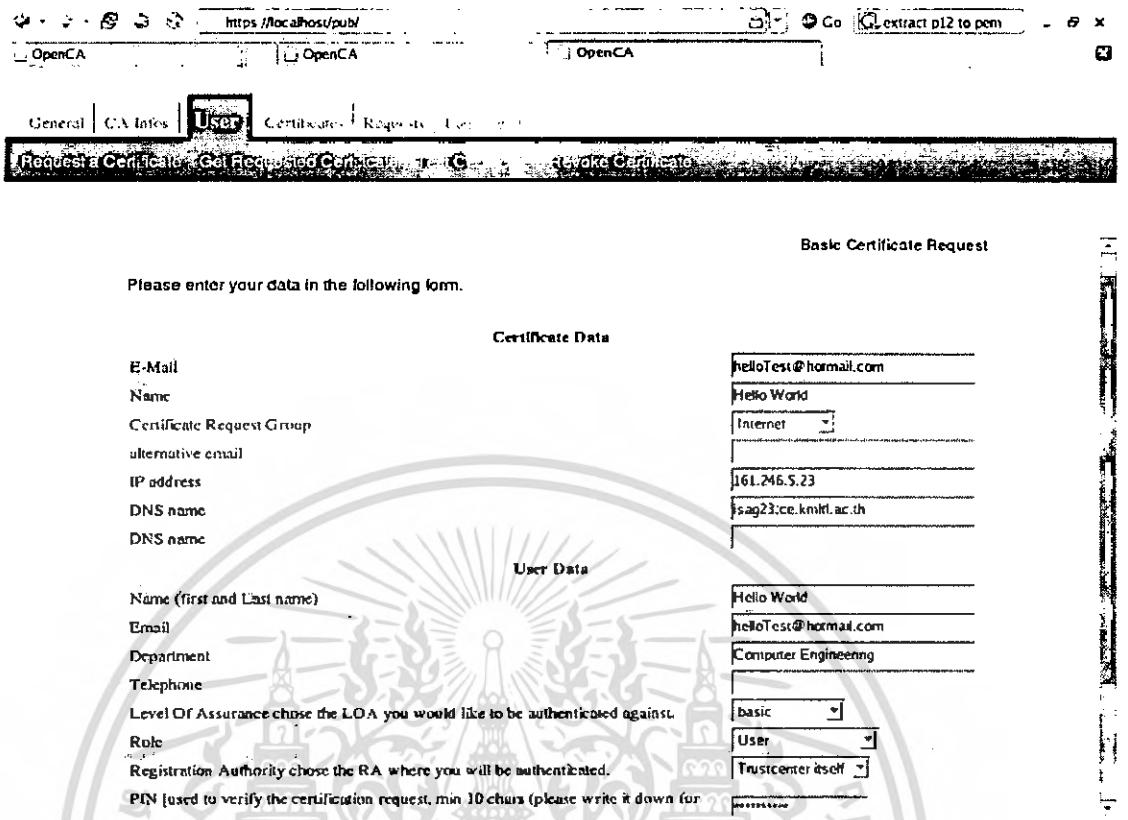
1. การสร้างกุญแจส่วนตัว กุญแจสาธารณะและใบร้องขอใบรับรองสิทธิ์

เมื่อเข้า https://[hostname]/pub จะปรากฏหน้าเพจดังรูปที่ 5-14



รูปที่ 5-14 แสดงหน้าเว็บในส่วน Public

ทำการคลิกเลือก User -> Request a Certificate -> Request a cert with automatic browser detection เพื่อกรอกข้อมูลรายละเอียดผู้ใช้งาน ดังรูปที่ 5-15



รูปที่ 5-15 แสดงแบบฟอร์มการกรอกรายละเอียดของผู้ใช้

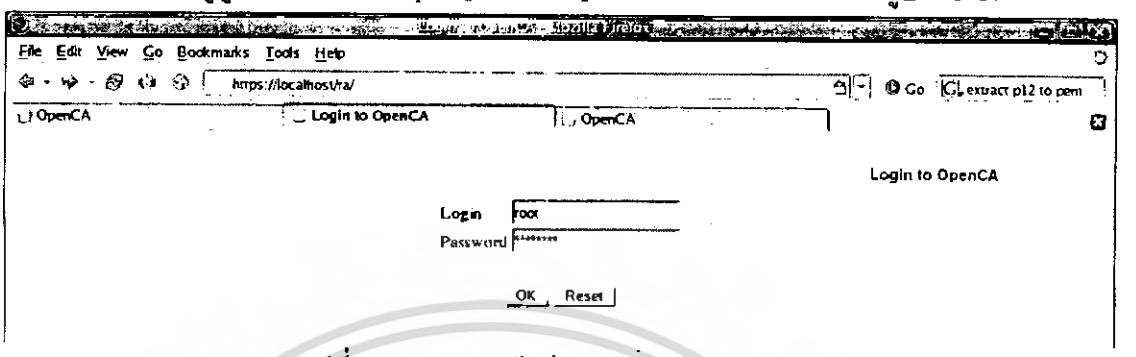
Browser จะสร้างกุญแจส่วนตัว กุญแจสาธารณะ และใบร้องขอใบรับรองสิทธิ์ ดังรูปที่ 5-16

รูปที่ 5-16 แสดงกล่องข้อความบอกสถานะของการสร้างคู่กุญแจ

ซึ่งประสบความสำเร็จในการสร้างดังรูป และถ้าไม่สำเร็จบราวซ์เซอร์จะบอกให้สร้างกุญแจส่วนตัว กุญแจสาธารณะ และใบร้องขอใบรับรองสิทธิ์ ใหม่อีกครั้งหนึ่ง

2. การตรวจสอบความถูกต้องใบร้องขอใบรับรองสิทธิ์

ผู้ดูแลระบบเข้า https://[hostnamc]/ra ระบบจะให้ล็อกอินดังรูปที่ 5-17

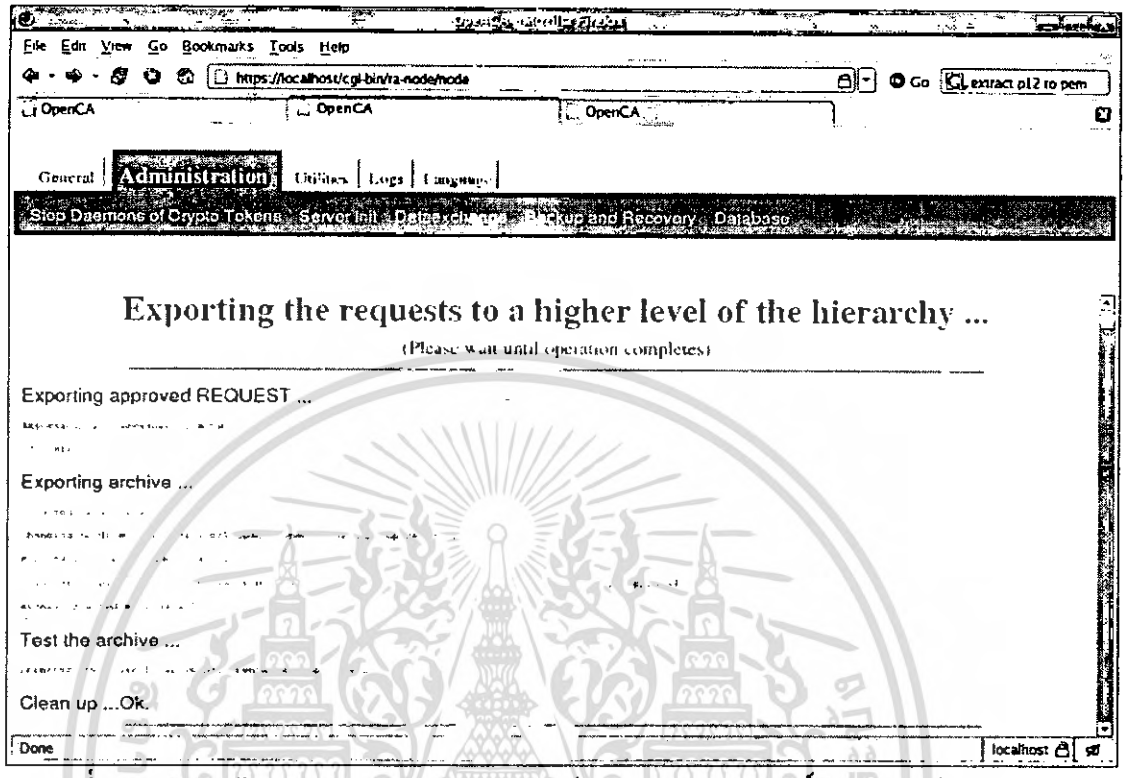


รูปที่ 5-17 แสดงหน้าเว็บของส่วน RA

เมื่อผู้ดูแลระบบตรวจสอบและรับรองใบร้องขอใบรับรองสิทธิ์แล้ว ระบบจะแสดงข้อความดังรูปที่ 5-18

รูปที่ 5-18 แสดงข้อความการรับรองใบร้องขอใบรับรองสิทธิ์เรียบร้อย

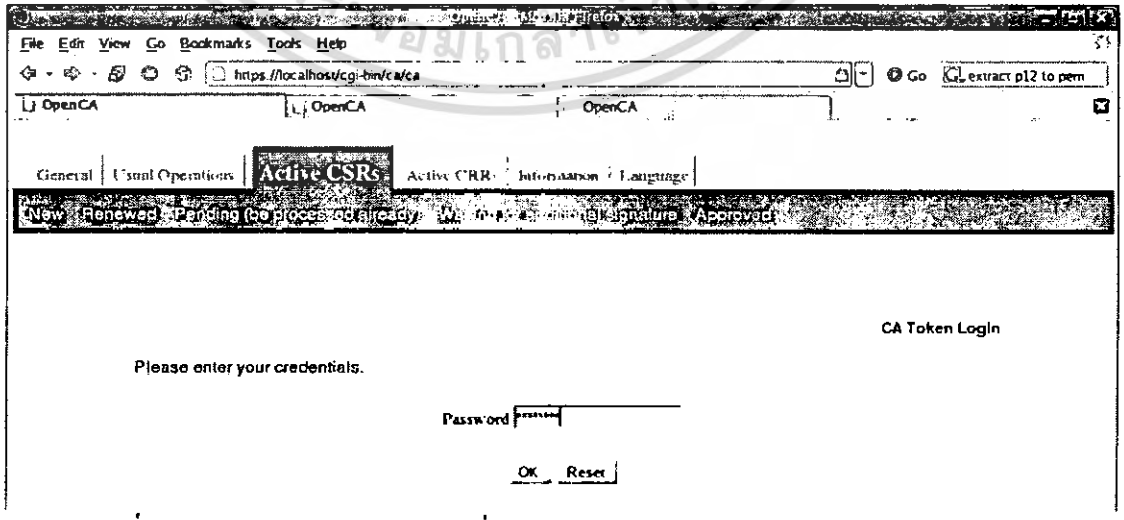
3. การส่งใบร้องขอใบรับรองสิทธิ์จาก RA ให้ CA



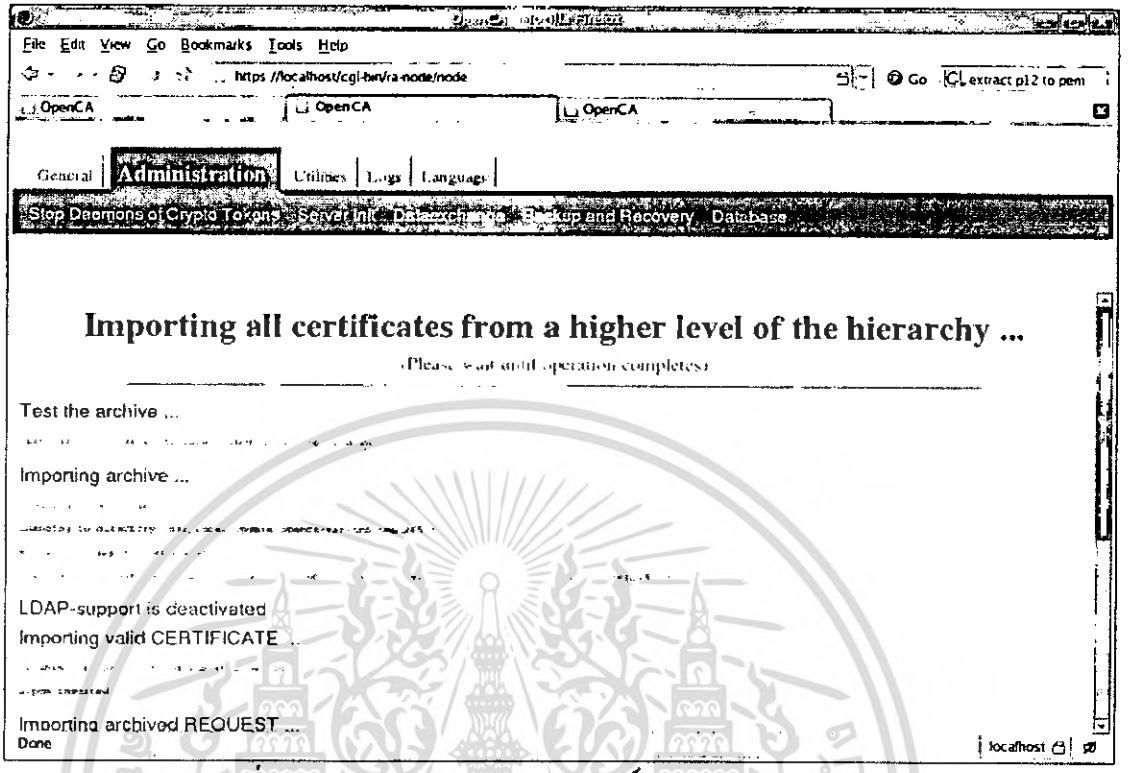
รูปที่ 5-19 แสดงข้อความรายละเอียดการส่งใบร้องขอใบรับรองสิทธิ์จาก RA ให้ CA

5.2.6 การออกใบรับรองสิทธิ์

ผู้ดูแลระบบออกใบรับรองสิทธิ์ จากใบร้องขอใบรับรองสิทธิ์ที่ตรวจสอบแล้ว โดยต้องระบุรหัสผ่านที่ใช้ในการเข้ารหัสกุญแจส่วนตัวของ CA ดังรูปที่ 5-20



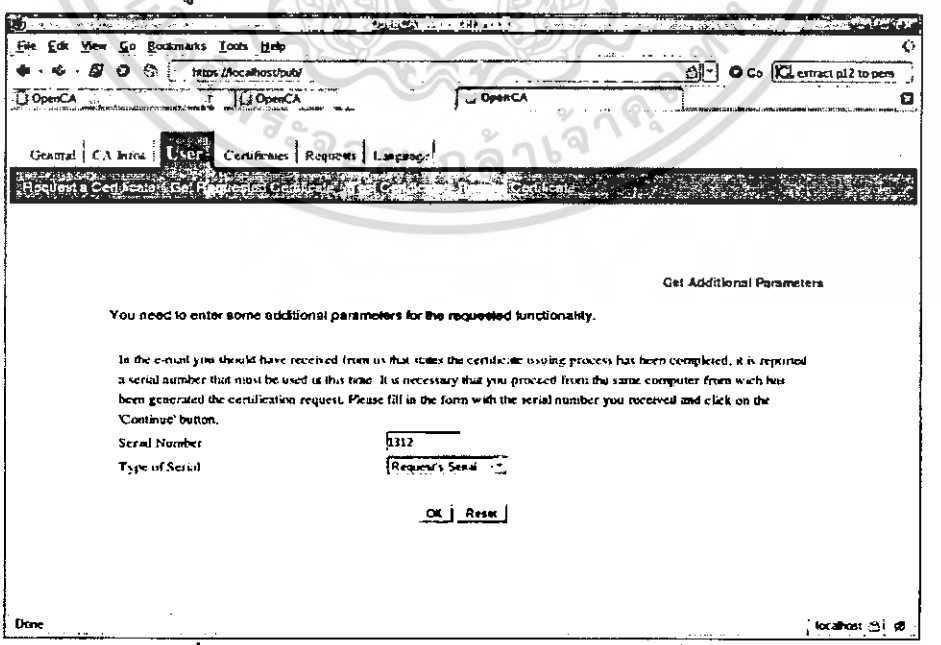
รูปที่ 5-20 แสดงการระบุรหัสผ่านที่ใช้ในการเข้ารหัสกุญแจส่วนตัวของ CA



รูปที่ 5-21 แสดงการส่งใบรับรองสิทธิ์จาก CA ให้ RA

5.2.7 การดาวน์โหลดใบรับรองสิทธิ์เก็บไว้ใน Browser

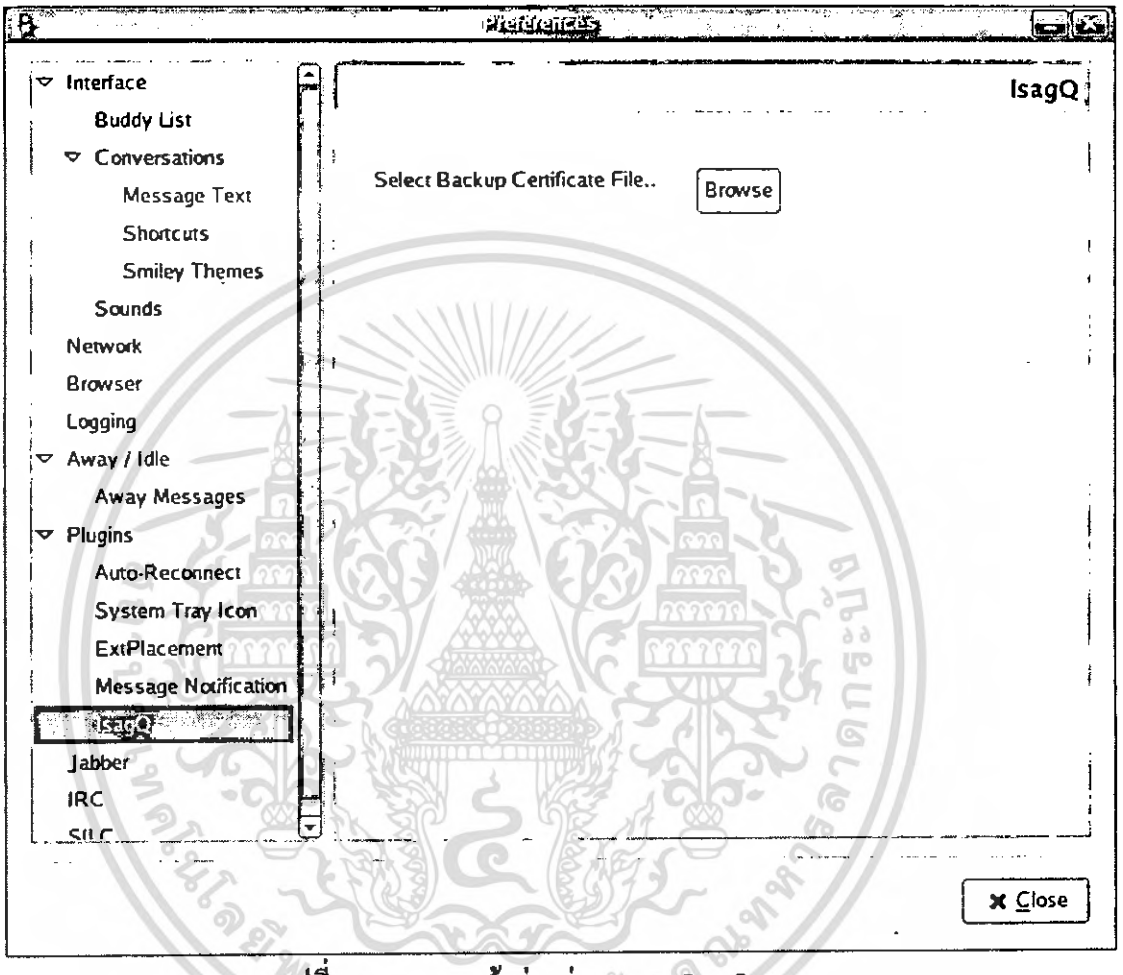
ผู้ใช้ต้องระบุหมายเลขซีเรียลของใบร้องขอใบรับรองสิทธิ์ที่ระบบออกให้ในขณะกรอกรายละเอียดดังรูปที่ 5-22



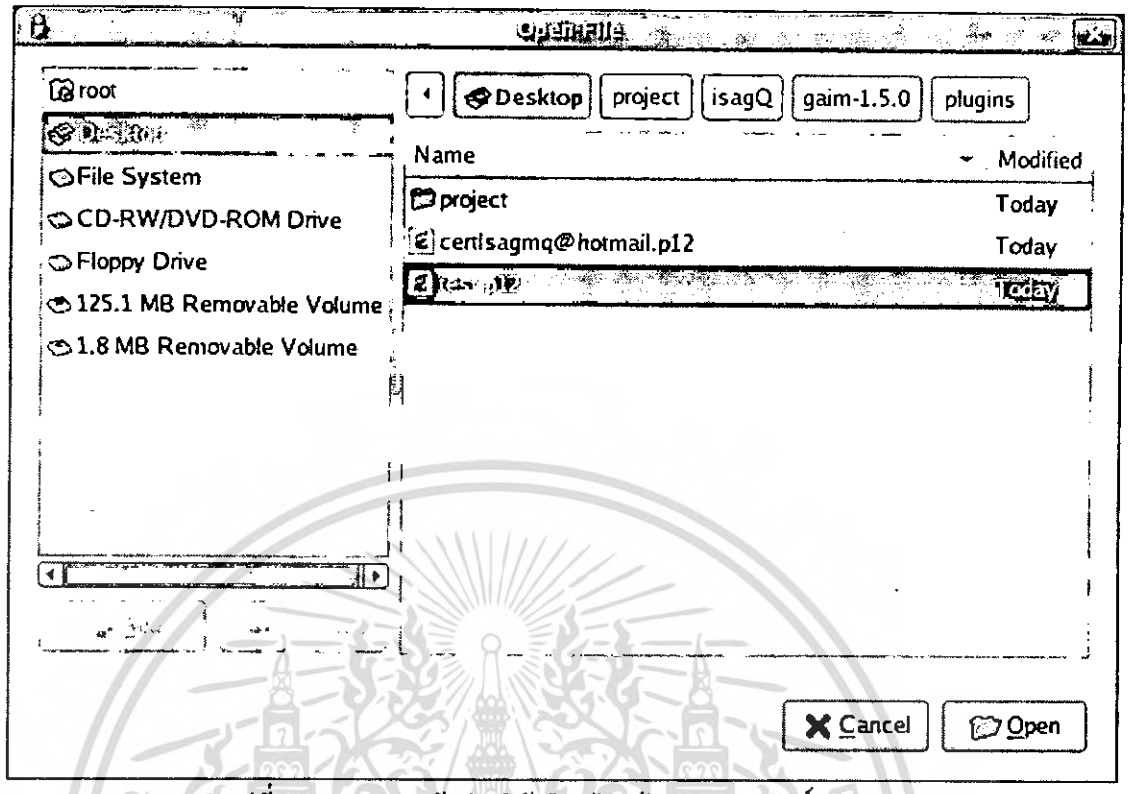
รูปที่ 5-22 แสดงการดาวน์โหลดใบรับรองสิทธิ์ของผู้ใช้

5.2.8 การนำใบรับรองสิทธิ์มาใช้ในโปรแกรม Gaim

การทดสอบในส่วนนี้จะเกิดขึ้นหลังจากผู้ใช้นำใบรับรองสิทธิ์ออก (Export) จาก Browser และบันทึกเป็นไฟล์ในรูปแบบ pkcs12 แล้ว เพื่อดูว่าสามารถนำใบรับรองสิทธิ์ที่ได้มาใช้งานได้จริง ดังรูปที่ 5-23

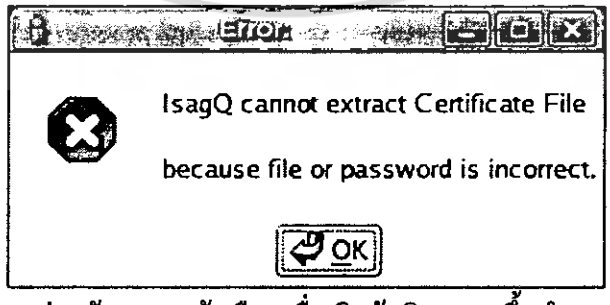


รูปที่ 5-23 แสดงหน้าต่างส่วนขยาย IsagQ



รูปที่ 5-24 แสดงหน้าต่างให้เลือกไฟล์ใบรับรองสิทธิ์

รูปที่ 5-25 แสดงหน้าต่างให้ผู้ใช้ระบุรหัสผ่านก่อนที่จะนำใบรับรองสิทธิ์มาใช้งาน



รูปที่ 5-26 แสดงกล่องข้อความแจ้งเตือนเมื่อเกิดข้อผิดพลาดขึ้น ในกรณีที่สามารถนำใบรับรองสิทธิ์มาใช้งานได้ก็จะไม่แจ้งเตือน

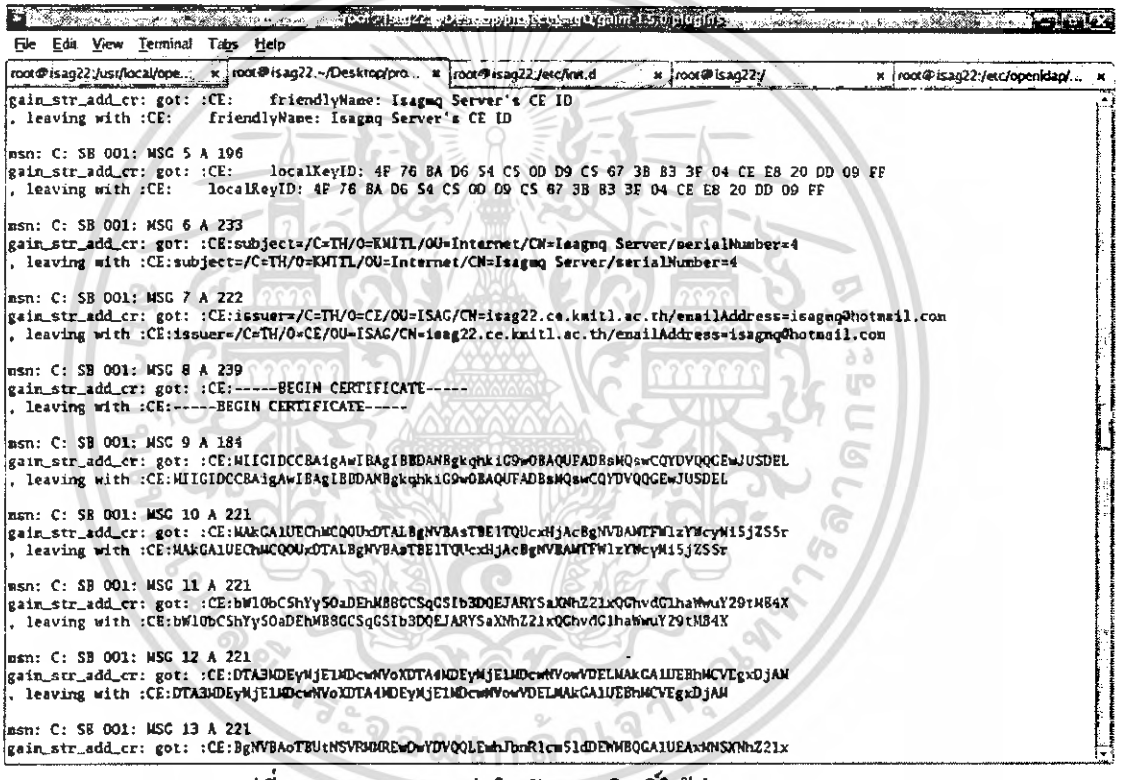
5.3. การทดสอบการทำงานหลังจากการ Sign-in

จุดมุ่งหมายที่ต้องการ : สามารถส่งข้อความที่เข้ารหัสลับไปยังปลายทาง และเครื่องปลายทางสามารถถอดรหัสลับข้อความได้

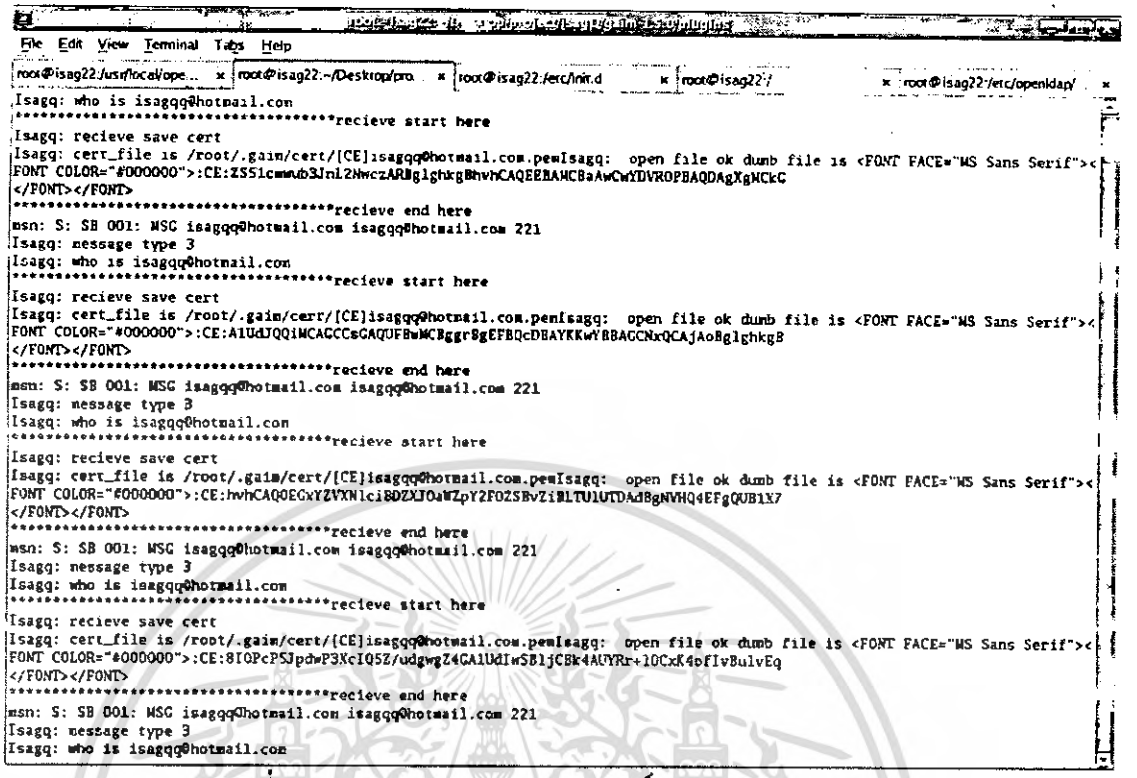
ในส่วนนี้จะเป็นการทดสอบการทำงานของส่วนขยาย IsagQ หลังจากที่ขอใบรับรองสิทธิ์เรียบร้อยแล้ว

5.3.1. การร้องขอใบรับรองสิทธิ์ของคู่สนทนา

จะทดสอบว่าเครื่องไคลเอนท์ที่ได้ลงส่วนขยาย IsagQ ไว้สามารถแลกเปลี่ยนใบรับรองสิทธิ์ระหว่างกันได้



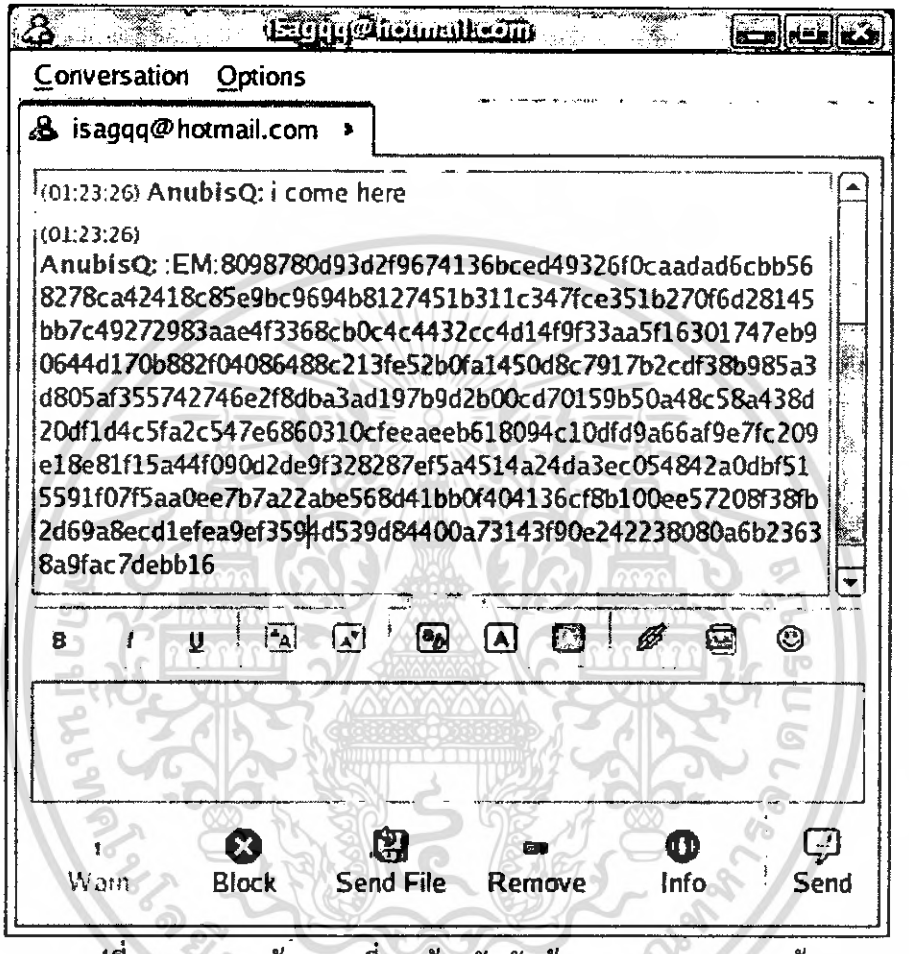
รูปที่ 5-27 แสดงการส่งใบรับรองสิทธิ์ให้คู่สนทนา



รูปที่ 5-28 แสดงการรับใบรับรองสิทธิ์ของคู่สนทนา

5.3.2. การเข้ารหัสลับข้อความก่อนส่งให้คู่สนทนา

จะทดสอบว่าเมื่อทั้งสองฝ่ายมีใบรับรองสิทธิ์ของคู่สนทนา สามารถนำกุญแจสาธารณะของคู่สนทนาเข้ารหัสลับข้อความก่อนส่งให้คู่สนทนาได้ ดังรูปที่ 5-29



รูปที่ 5-29 แสดงข้อความที่ถูกเข้ารหัสลับด้วยกุญแจสาธารณะแล้ว

บทที่ 6
บทสรุป

6.1. ภาพรวม

โปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะ และชีวมาตร เป็นโครงงานที่พัฒนามาจากโครงงานระบบไคลเอ็นต์และเซิร์ฟเวอร์สำหรับส่งสารด่วนแบบปลอดภัย และโครงงานระบบต้นแบบในการล็อกอินเข้าลีนุกซ์ด้วยลายนิ้วมือ โดยรวมการทำงานระบบรับส่งสารด่วนแบบปลอดภัยซึ่งอาศัยการเข้าและถอดรหัสลับแบบกุญแจสาธารณะ ซึ่งได้พัฒนาโปรแกรมฝั่งไคลเอ็นต์เป็นส่วนขยายของโปรแกรม Gaim เข้ากับระบบการพิสูจน์ตนด้วยชีวมาตร ซึ่งได้พัฒนาให้พิสูจน์ตนด้วยลายนิ้วมือก่อนเข้าใช้งานโปรแกรม Gaim

6.2. ปัญหาที่พบและวิธีการแก้ไข

6.2.1. ปัญหาที่พบในส่วนการทำงานของชีวมาตรด้วยลายนิ้วมือและวิธีการแก้ไข

สำหรับปัญหาที่พบในส่วนนี้ส่วนมากจะเกี่ยวข้องกับส่วนขยาย AnubisQ โดยปัญหาหลักๆที่พบคือ

- ปัญหาความไม่เข้ากันของอุปกรณ์อ่านลายนิ้วมือกับส่วนติดต่อผู้ใช้ (User Interface)

  ปัญหานี้เกิดขึ้นเมื่อเวลาเราทำการสั่งอ่านลายนิ้วมือ โปรแกรมจะต้องรอให้กระบวนการอ่านข้อมูลลายนิ้วมือเสร็จสิ้นเสียก่อน หน้าจอติดต่อผู้ใช้จึงจะแสดงขึ้นมา ซึ่งโดยความต้องการแล้ว ต้องการให้หน้าจอนี้ปรากฏขึ้นมาเสียก่อน แล้วจึงทำการอ่านลายนิ้วมือ

  วิธีการแก้ไข ได้ทำการแบ่งการทำงานสองงานดังกล่าวนี้ให้ทำงานในลักษณะของเธรด ซึ่งก็ช่วยแก้ปัญหาได้ส่วนหนึ่ง

- ปัญหาความไม่เสถียรของตัวโปรแกรม Gaim

  ปัญหานี้มักจะเกิดขึ้น เมื่อส่วนขยาย AnubisQ ที่พัฒนานั้นทำงานผิดพลาด หรือ ภายในซอร์สโค้ดของส่วนขยายมีการใช้งานพอยน์เตอร์ ซึ่งอาจทำให้เกิดปัญหาเรื่องการจัดสรรหน่วยความจำได้ โดยเมื่อเกิดความผิดพลาดขึ้น โปรแกรม Gaim จะไม่สามารถทำงานต่อได้ทันที และจะปิดตัวเองลง พร้อมกับแจ้งข้อความผิดพลาดว่า Segfaulted (Segmentation Faulted)

  วิธีการแก้ไข พยายามตรวจสอบการใช้พอยน์เตอร์ว่าใช้งานอย่างถูกต้องเหมาะสมกับชนิดของข้อมูลหรือไม่ และตรวจสอบการทำงานของแต่ละฟังก์ชัน

อยู่เสมอๆ ว่าฟังก์ชันใดทำให้เกิด Segmentation Faulted เมื่อพบก็ทำการแก้ไขฟังก์ชันนั้น หรือเขียนฟังก์ชันใหม่ขึ้นมาทดแทน

- ปัญหาในการพัฒนาส่วนขยายของโปรแกรม Gaim

ปัญหาที่พบคือ เอกสารวิธีการพัฒนาส่วนขยายของโปรแกรม Gaim ยังมีไม่มากนัก ทำให้การค้นหาข้อมูลที่ต้องการต้องใช้เวลานานเป็นพิเศษ และอีกปัญหาที่พบเกี่ยวกับส่วนขยายของโปรแกรม Gaim ก็คือ ส่วนขยายที่พัฒนาสำหรับ Gaim เวอร์ชัน 1.5 ไม่สามารถใช้งานร่วมกับโปรแกรม Gaim เวอร์ชัน 2.0 ได้อย่างสมบูรณ์

6.2.2. ปัญหาที่พบในส่วนการทำงานของโครงสร้างพื้นฐานกุญแจสาธารณะและวิธีการแก้ไข

- ปัญหาในการนำ OpenCA มาใช้งาน

ปัญหาที่พบส่วนมากจะเกี่ยวกับวิธีการปรับแต่งค่าต่างๆของ OpenCA เนื่องจาก OpenCA ประกอบด้วยส่วนการทำงานหลายส่วนๆ เช่นส่วนที่เป็น CA หรือ RA เป็นต้น

วิธีการแก้ไข พยายามศึกษาจากตัวคู่มือของ OpenCA เองและดำเนินการปรับแต่งไปทีละส่วนๆ

- ปัญหาความไม่สะดวกในการใช้ OpenCA

ปัญหานี้คือ เวลาที่ต้องการใช้ใบรับรองสิทธิ์ ผู้ใช้จะต้องมีความรู้เรื่องในส่วนของใบรับรองสิทธิ์บ้าง และ ในบางครั้งผู้ใช้ต้องเอาใบรับรองสิทธิ์ออกมาจากบราวซ์เซอร์เอง

วิธีการแก้ไข ได้พัฒนาส่วนของหน้าจอติดผู้ใช้ เพื่ออำนวยความสะดวกให้แก่ผู้ใช้ในบางขั้นตอน ซึ่งก็ช่วยได้เป็นอย่างมาก

6.3. แนวทางในการพัฒนาต่อในอนาคต

สำหรับของส่วนขยาย AnubisQ อาจพัฒนาให้สามารถใช้ชีวมาตรด้านอื่นๆได้ ไม่ว่าจะเป็นการใช้การรู้จำเสียง (Voice Recognition) หรือเป็นการรู้จำใบหน้า (Face Recognition) เป็นต้น เพื่อการพิสูจน์ตัวตนก่อนเข้าใช้งานโปรแกรม Gaim ต่อไปในอนาคต

นอกจากนี้ในส่วนของที่เกี่ยวกับโครงสร้างพื้นฐานกุญแจะสาธารณะ สามารถพัฒนาให้ระบบมีความสมบูรณ์มากยิ่งขึ้นโดยอาจพัฒนาให้ส่วนของ CRL นำมาใช้งานได้จริงและมีความสมบูรณ์มากยิ่งขึ้น

6.4. ข้อสรุปและข้อเสนอแนะ

โปรแกรมประยุกต์สำหรับโครงสร้างพื้นฐานกุญแจสาธารณะและชีวมาตรนี้สามารถทำงานได้ตรงตามจุดประสงค์ที่ได้กำหนดไว้เป็นอย่างดี คือ การยกระดับความปลอดภัยในการใช้งานโปรแกรมรับส่งสารด่วนให้มีมากยิ่งขึ้น ด้วยการอาศัยการพิสูจน์ตัวตนด้วยชีวมาตรก่อนการเข้าใช้งานโปรแกรม Gaim และการเข้ารหัสลับ/ถอดรหัสลับข้อความที่กำลังสนทนา เพื่อป้องกันไม่ให้ผู้อื่นสามารถอ่านได้

เอกสารอ้างอิง

[1] VeriFinger 4.2 Standard SDK Documentation for Linux

http://www.neurotechnologija.com/download/VF_42_SDK_Linux.pdf

[2] GAIM Documentation

http://gaim.sourceforge.net/documentation.php

[3] OpenSSL Documentation

http://www.openssl.org/docs/

[4] Paul Reid, *Biometrics for Network Security*, Prentice Hall PTR

[5] Neil Matthew / Richard Stones, *Beginning Linux Programming (Paperback)*, WROX Press