ระบบไคลเอ็นต์และเซิร์ฟเวอร์สำหรับรับส่งสารด่วนแบบปลอดภัย

Secure Instant Messaging Client/Server System

นายจารุ นาถบุญ
นายณราพงษ์ พิมล
นายอกนิษฐ์ ลาภวรชัย

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2547

ระบบไคลเอ็นต์และเซิร์ฟเวอร์สำหรับรับส่งสารด่วนแบบปลอดภัย

Secure Instant Messaging Client/Server System

โดย

นายจารุ นาถบุญ

นายณราพงษ์ พิมล

นายอกนิษฐ์ ลาภวรชัย

อาจารย์ที่ปรึกษา

อาจารย์อัครเดช วัชระภูพงษ์

อาจารย์ธนา หงษ์สุวรรณ

อาจารย์ธนัญชัย ตรีภาค

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

ปริญญานิพนธ์ปีการศึกษา 2547

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบไคลเอ็นต์และเซิร์ฟเวอร์สำหรับรับส่งสารด่วนแบบปลอดภัย

Secure Instant Messaging Client/Server System

ผู้จัดทำ

1. นายจารุ นาถบุญ รหัสประจำตัว 44010069
2. นายณราพงษ์ พิมล รหัสประจำตัว 45015363
3. นายอกนิษฐ์ ลาภวรชัย รหัสประจำตัว 45015395

_____อาจารย์ที่ปรึกษา

(อาจารย์อัครเดช วัชระภูพงษ์)

_____อาจารย์ที่ปรึกษา

(อาจารย์ธนา หงษ์สุวรรณ)

_____อาจารย์ที่ปรึกษา

(อาจารย์ธนัญชัย ตรีภาค)

ระบบไคลเอ็นต์และเซิร์ฟเวอร์สำหรับรับส่งสารด่วนแบบปลอดภัย

นายจารุ	นาถบุญ	44010067
นายณราพงษ์	พิมล	45015363
นายอกนิษฐ์	ลาภวรชัย	45015395
อาจารย์อัครเดช	วัชระภูพงษ์	อาจารย์ที่ปรึกษา
อาจารย์ธนา	หงษ์สุวรรณ	อาจารย์ที่ปรึกษา
อาจารย์ธนัญชัย	ตรีภาค	อาจารย์ที่ปรึกษา

ปีการศึกษา 2547

บทคัดย่อ

ระบบไคลเอ็นต์และเซิร์ฟเวอร์สำหรับรับส่งสารด่วนแบบปลอดภัย เป็นโครงงานที่ถูกพัฒนาต่อเนื่องมาจาก โปรแกรมแม่ข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย หรือ เรียกสั้นๆว่า IsagMQ ที่ใช้ภาษาซี ในการพัฒนา และ โปรแกรมรับส่งสารด่วนแบบปลอดภัย หรือ IsagQ ที่ใช้ภาษาจาวาในการพัฒนา ทั้งนี้ก็เพื่อตอบสนองด้านความปลอดภัย 2 ประการเป็นสำคัญ คือ การทำระบบพิสูจน์ตน เพื่อเสริมสร้างความมั่นใจให้กับผู้ใช้ว่ากำลังติดต่ออยู่กับบุคคลที่ต้องการจริงๆ และ การเข้าและถอดรหัสเพื่อปกปิดข้อมูลให้เป็นความลับ ไม่ถูกเปิดเผยให้บุคคลอื่นที่ไม่สมควรได้รับทราบ

ด้วยการประยุกต์ใช้ใบรับรองสิทธิ์ IsagMQ นอกจากจะให้บริการพื้นฐานแก่ IsagQ แล้ว ยังต้องทำตัวเป็น องค์กรออกใบรับรองสิทธ์ภายใน (Private Certificate Authority) แก่ ผู้ใช้อีกด้วย

รูปแบบการสื่อสารระหว่าง IsagQ กับ IsagMQ และ IsagQ กับ IsagQ จะเป็นแบบ เพียร์ทูเพียร์ ซึ่งมีความรวดเร็ว และ ไม่ทำให้เซิร์ฟเวอร์ทำงานหนัก

บริการพื้นฐานที่ IsagQ สามารถเรียกใช้ จาก IsagMQ ได้ มีดังนี้ การล๊อกอิน , การล๊อกเอาท์ , การเปลี่ยนชื่อเล่น , การเพิ่มรายชื่อคู่สนทนา , การลบรายชื่อคู่สนทนา , การค้นหาคู่สนทนา , การปฏิเสธคู่สนทนา ,การยกเลิกการปฏิเสธคู่สนทนา , การยอมรับคู่สนทนา ,การดูรายชื่อผู้รอคำยินยอม , การดูสถานะของคู่สนทนา และ การเปลี่ยนสถานะของผู้ใช้

ส่วนบริการพื้นฐานระหว่าง IsagQ กับ IsagQ ด้วยกัน คือสามารถติดต่อกับ ICQ เซิร์ฟเวอร์ได้ แต่ไม่ได้ครอบคลุมเรื่องความปลอดภัย เหมือนกันการติดต่อ ระหว่าง IsagQ ด้วยกันเอง นอกจากนี้แล้ว ยังสามารถส่งไฟล์ระหว่าง IsagQ ด้วยกันรวมถึงใช้ภาษาไทยติดต่อกันได้

Secure Instant Messaging Client/Server System

Mr. Jaru	Nartboon	44010067
Mr. Naraphong	Phimol	45015363
Mr. Akanit	Lapworachai	45015395
Mr. Akkradach	Watcharapupong	Advisor
Mr. Thana	Hongsuwan	Advisor
Mr. Thananchai	Treepak	Advisor

Academic year 2547

Abstract

Secure Instant Messaging Client/Server System is continuously developed from two projects. One is Secure Instant Messaging Server or IsagMQ using C language to develop while the last using JAVA language is Secure Instant Messenger. The purpose to develop is to serve for security enhancement . To guarantee the actual connection between each users, the authentication is required by using certificate signed by Private Cerificate Authority and to keep all conversations secret , the encryption/decryption technique comes into play.

IsagMQ is not only to serve for client services but also to be the Private Certificate Authority to handle to certificate requests.

Peer-to-peer communication is used to connect between IsagQ & IsagMQ and IsagQ & IsagQ. To speed up communication and also to decrease the load of IsagMQ.

Services served for IsagQ are Login , Logout , Change Nickname , Add Contact List ,Delete Contact List, Find Contact List , Block , Admit , Accepting Authorization , List Authorization Request , Update Status and Change User Status.

Sending file between IsagQ users , using Thai font and ICQ connection are some of IsagQ's options to serve for users.

กิตติกรรมประกาศ

โครงงานระบบไคลเอ็นต์ และ เซิร์ฟเวอร์ สำหรับรับส่งสารด่วนแบบปลอดภัย สำเร็จลุล่วงได้ด้วยดี ก็เนื่องมาจากการให้โอกาส การดูแล ให้คำแนะนำต่างๆ การสนับสนุน การให้คำสั่งสอน และให้คำปรึกษาเป็นอย่างดีเสมอมาจาก จาก อาจารย์ อัครเดช วัชระภูพงษ์ อาจารย์ ธนา หงษ์สุวรรณ และอาจารย์ ธนัญชัย ตรีภาค ซึ่งต้องขอขอบพระคุณอาจารย์ทั้ง 3 ท่านเป็นอย่างสูง ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์ และสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่ได้จัดเตรียมสิ่งอำนวยความสะดวก เพื่อให้งานวิจัยดำเนินไปได้อย่างสะดวกและรวดเร็ว ขอบคุณห้องวิจัยและพัฒนาการรักษาความปลอดภัยของข้อมูล (ISAG) ที่เป็นแหล่งประสิทธิ์ประสาทวิชาให้ความรู้ความเข้าใจ เป็นสถานที่ ที่มีความอบอุ่น ท่านอาจารย์ พี่ๆ ที่เป็นผู้ให้แนวทางแก้ไขและที่ปรึกษาของชิ้นงานจนลุลวงด้วยดี และท้ายที่สุดต้องขอขอบพระคุณบุคคลที่สำคัญที่สุดในชีวิตของข้าพเจ้าที่ทำให้ข้าพเจ้ามีทุกวันนี้คือ บิดา มารดา และบุคคลทุกคนในครอบครัวของข้าพเจ้า อันเป็นที่เคารพรัก คอยอุ้มชูเลี้ยงดู อบรมสั่งสอนข้าพเจ้ามาเป็นอย่างดี ทั้งยังให้ความรักความห่วงใย และกำลังใจที่ดีเสมอมา ข้าพเจ้าต้องขอขอบพระคุณมา ณ ที่นี้ด้วย

นายจารุ นาถบุญ

นายณราพงษ์ พิมล

นายอกนิษฐ์ ลาภวรชัย

สารบัญ

สารบัญตาราง

สารบัญภาพ

บทที่ 1
บทนำ

1. หลักการและเหตุผล

สืบเนื่องจากปัจจุบันการสื่อสารบนอินเทอร์เน็ตได้รับความนิยมเพิ่มขึ้นอย่างมาก ประกอบกับค่าใช้จ่ายในการใช้งานก็ถูกลงอันเนื่องมาจากการแข่งขันของผู้ให้บริการอินเทอร์เน็ต นับได้ว่าปัจจุบันอินเทอร์เน็ตได้กลายเป็นปัจจัยหนึ่งในชีวิตของมนุษย์ ผลลัพธ์จากการใช้บริการอินเทอร์เน็ตมีทั้งข้อดีและข้อเสีย ในทุกๆแง่มุม ในที่นี้จะกล่าวถึงข้อดีทางด้านการบันเทิง หรือ สันทนาการ

โปรแกรมรับส่งสารด่วน เป็นอีกตัวอย่างหนึ่งของความนิยมบนอินเทอร์เน็ตในขณะนี้ ซึ่งโปรแกรมดังกล่าวตอบสนองการให้บริการรับส่งสารได้อย่างทันท่วงที ได้ทุกเวลา และ ทุกที่ ทั้งที่เป็นเพื่อการบันเทิง หรือไม่ ก็ทางธุรกิจ โดยเป็นรูปแบบของการสื่อสารที่มีความประหยัดเป็นอย่างมาก ซึ่งนับได้ว่าเป็นคลื่นลูกใหม่ที่มาแรงแทนที่ความนิยมของ จดหมายอิเล็กทรอนิกส์ อย่างสมบูรณ์

แต่ในปัจจุบัน ผู้ให้บริการโปรแกรมรับส่งสารด่วน มีเป็นจำนวนมาก อาทิ MSN Messenger , ICQ , AOL และ อื่นๆ ซึ่งความสามารถของโปรแกรมดังกล่าวแตกต่างกันไปตามผู้สร้าง ซึ่งในที่นี้จะเน้นถึงเรื่องความปลอดภัยเป็นหลัก โดยปกติโปรแกรมเหล่านี้มิได้คำนึงถึงเรื่องความปลอดภัย ไม่ว่าจะเป็นเรื่องการเข้าและถอดรหัส หรือ แม้กระทั่ง การพิสูจน์ตนของผู้ใช้ และ ผู้ให้บริการ รวมถึง การขาดความเข้าใจในโปรแกรมของผู้ใช้งาน ยิ่งพิจารณาถึงปัญหา อาชญากร บนอินเทอร์เน็ต รวมด้วยแล้ว จะเห็นได้ว่า ปัญหาเรื่องความปลอดภัย เป็นอีกมุมมองหนึ่งที่ควรคำนึงถึงอย่างมาก เพื่อสนองความต้องการของผู้ใช้งานได้อย่างครบครัน

ทางห้องวิจัยและพัฒนาการรักษาความปลอดภัยของข้อมูล (Information Security Advisory Group : ISAGQ) จึงได้พัฒนาโครงงาน 2 โครงงานขึ้นมา ซึ่ง โครงงานแรกเป็นส่วนของการสร้างโปรแกรมแม่ข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย หรือ Secure Instant Messaging Server : IsagMQ ซึ่งพัฒนาด้วยภาษาซี บนระบบปฏิบัติการตระกูลยูนิกซ์ และ โครงงานที่สองเป็นส่วนของการสร้างโปรแกรมลูกข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย หรือ Secure Instant Messenger : IsagQ ซึ่งพัฒนาด้วยภาษาจาวา เพื่อตอบสนองด้านความปลอดภัย 2 ประการหลัก คือ การพิสูจน์ตน เพื่อสร้าง

ความมั่นใจให้กับผู้ใช้ว่ากำลังติดต่อยู่กับบุคคลที่ต้องการจริงๆ และ การเข้าและถอดรหัส เพื่อปิดบังข้อมูลที่ถูกส่งผ่านเครือข่าย

สำหรับโครงงานนี้จะเป็นการประยุกต์รวมความแตกต่างบนวัตถุประสงค์เดียวเข้าด้วยกัน ในชื่อ ระบบไคลเอ็นต์และเซิร์ฟเวอร์สำหรับรับส่งสารด่วนแบบปลอดภัย หรือ Secure Instant Messaging Client / Server System (IsagMQ 2547 & IsagQ 2547)

2. วัตถุประสงค์

2.1 เพื่อศึกษาการทำงานของระบบรับส่งสารด่วนทั้งฝั่งไคลเอ็นต์และเซิร์ฟเวอร์

2.2 เพื่อปรับปรุงระบบรับส่งสารด่วนให้มีความปลอดภัยยิ่งขึ้น

2.3 เพื่อสร้างโปรแกรมต้นแบบทั้งฝั่งไคลเอ็นต์และเซิร์ฟเวอร์สำหรับรับส่งสารด่วนแบบปลอดภัย

3. เป้าหมาย

3.1 สร้างโปรแกรมที่สามารถใช้ระบบพิสูจน์ตน เพื่อทำให้ผู้ใช้มั่นใจได้ว่ากำลังติดต่อกับบุคคลที่ต้องการติดต่อจริง

3.2 สร้างโปรแกรมที่สามารถเข้าและถอดรหัสลับข้อความ เพื่อทำให้ผู้ใช้มั่นใจได้ว่าข้อความที่ส่งถูกปกปิดเป็นความลับ

3.3 สร้างการเชื่อมต่อระหว่างโปรแกรมฝั่งแม่ข่าย และ โปรแกรมฝั่งลูกข่าย โดยยังคงความสามารถตามข้อ 3.1 และ 3.2 ได้

4. ผลที่คาดว่าจะได้รับ

4.1 ได้รับความรู้ความสามารถในการเข้าและถอดรหัสลับ

4.2 ได้รับความรู้ความสามารถในระบบพิสูจน์ตน

4.3 ได้รับความรู้ความสามารถในการเขียนโปรแกรมด้วยภาษาซี และ ภาษาจาวา

4.4 ได้รับความรู้ความสามารถในการใช้งานระบบปฏิบัติการตระกูลยูนิกซ์

4.5 สร้างระบบไคลเอ็นต์และเซิร์ฟเวอร์ที่ติดต่อกันโดยใช้ระบบพิสูจน์ตน รวมถึงการเข้าและถอดรหัส]y[

5. ขอบเขตการพัฒนา

Input Specification

- ผู้ใช้งานใช้บริการผ่านโปรแกรมลูกข่าย หรือ IsagQ เพื่อรับบริการต่างๆจากโปรแกรมแม่ข่าย หรือ IsagMQ
- ผู้ใช้งานใช้บริการผ่านโปรแกรมลูกข่าย หรือ IsagQ เพื่อติดต่อไปยังสมาชิกบนบัญชีรายชื่อของผู้ใช้ ที่ใช้ IsagQ อยู่ด้วยกัน

Output Specification

- ผู้ใช้งานได้รับบริการตามที่ถูกเลือก
- ผู้ใช้งานได้รับรายงานความผิดพลาดที่เกิดขึ้นขณะใช้บริการจากโปรแกรมแม่ข่าย
- ผู้ใช้งานสามารถส่งข้อความระหว่างผู้ใช้งานคนอื่นได้

Function Specification

แบ่งได้ดังนี้

IsagMQ (โปรแกรมฝั่งแม่ข่าย)

1. ออก ใบรับรองสิทธิ์ ให้กับ ผู้ใช้
2. จัดทำฐานข้อมูลของผู้ใช้ที่ลงทะเบียนแล้ว
3. สามารถรับส่งข้อมูลเกี่ยวกับสถานะ และ ชื่อของผู้ใช้ได้
4. สามารถให้บริการลงทะเบียนและยกเลิกการลงทะเบียนได้
5. สามารถให้บริการล็อกอินและล็อกเอาต์ได้
6. สามารถให้บริการเพิ่มและลบคู่สนทนาได้
7. สามารถให้บริการค้นหาคู่สนทนาได้
8. สามารถให้บริการเปลี่ยนชื่อเล่นของผู้สนทนาได้
9. สามารถให้บริการปฏิเสธ และ ยกเลิกการปฏิเสธคู่สนทนา
10. สามารถให้บริการยอมรับคู่สนทนา
11. เปลี่ยนสถานะในการสนทนา

IsagQ (โปรแกรมฝั่งลูกข่าย)

1. สามารถให้บริการลงทะเบียนและยกเลิกการลงทะเบียนได้
2. สามารถให้บริการล็อกอินและล็อกเอาต์ได้
3. สามารถให้บริการเพิ่มและลบคู่สนทนาได้
4. สามารถให้บริการค้นหาคู่สนทนาได้
5. สามารถให้บริการเปลี่ยนชื่อเล่นของผู้สนทนาได้
6. แสดงสถานะ ของ สมาชิกทั้งสองโปรแกรม
7. ผู้ใช้สามารถใช้งานรวมกับโปรแกรม ICQ ของ AOL
8. ผู้ใช้สามารถรับส่งไฟล์ข้อมูลระหว่างกันได้
9. ผู้ใช้ติดต่อสื่อสารระหว่างกันแบบ peer-to-peer
10. ผู้ใช้ติดต่อสื่อสารระหว่างกันอย่างปลอดภัยโดยมีการเข้ารหัสข้อมูลสำหรับข้อมูลระหว่าง IsagQ ด้วยกัน ส่วนข้อมูลระหว่าง IsagQ กับ ICQ จะเป็น Plain Text
11. ผู้ใช้สามารถติดตั้งบนระบบปฏิบัติการ Windows ได้
12. เปลี่ยนสถานะในการสนทนา

6. วิธีการดำเนินงาน

6.1 ศึกษาทฤษฎี หรือ แนวคิด สำหรับการเสริมสร้างความปลอดภัย ใน ตั้งแต่บทที่ 3 – 6

6.2 ศึกษาโค้ดโปรแกรมยุคก่อน เพื่อให้เกิดแนวทาง

6.3 ค้นคว้าตัวอย่างเพิ่มเติมจากอินเทอร์เน็ตในส่วนที่เกี่ยวข้องกับโครงงาน เช่น ตัวอย่างการเขียนโปรแกรมติดต่อดาตาเบส , ตัวอย่างการเขียนโปรแกรมสื่อสารด้วย SSL โดยใช้ภาษาซี เป็นต้น

6.4 ออกแบบโพรโตคอล และ ฐานข้อมูลที่เหมาะสม

6.5 พัฒนาแก้ไข และ ทดสอบโปรแกรมทางฝั่งแม่ข่าย (IsagMQ)

6.6 พัฒนาแก้ไข และ ทดสอบโปรแกรมทางฝั่งลูกข่าย (IsagQ)

6.7 จัดทำบันทึกการเหตุการณ์ ความเปลี่ยนแปลง รวมถึงปัญหาที่เกิดขึ้น

6.8 จัดทำรายงาน

บทที่ 2
โปรแกรมแม่ข่ายและลูกข่ายสำหรับรับส่งสารด่วน

เราสามารถแบ่งมุมมองในระบบรับส่งสารด่วนได้ 2 ส่วน คือ

1. โปรแกรมแม่ข่ายสำหรับให้บริการ
2. โปรแกรมลูกข่ายสำหรับรับส่งสารด่วน

โดยทั่วไปทุกครั้งก่อนที่ไคลเอ็นต์จะใช้บริการ ไคลเอ็นต์จำเป็นต้องใช้โปรแกรมลูกข่ายทำการล๊อกอิน ติดต่อไปยังโปรแกรมแม่ข่ายก่อนเสมอ เนื่องจาก โปรแกรมแม่ข่ายต้องการข้อมูลที่จำเป็นเบื้องต้นหลายๆอย่างก่อนที่ใช้เริ่มการให้บริการแก่ไคลเอ็นต์ได้ หากการ ล๊อกอินสำเร็จอย่างสมบูรณ์ โปรแกรมแม่ข่ายจะสามารถให้บริการแก่ไคลเอ็นต์

หน้าที่ของโปรแกรมแม่ข่ายจะมากหรือน้อยขึ้นอยู่กับการออกแบบรูปแบบการติดสื่อสาร รวมถึงความสามารถต่างๆที่โปรแกรมลูกข่ายต้องการ ซึ่งถ้ายิ่งมากแล้ว โปรแกรมแม่ข่ายก็ยิ่งมีหน้าที่เพิ่มมากขึ้น

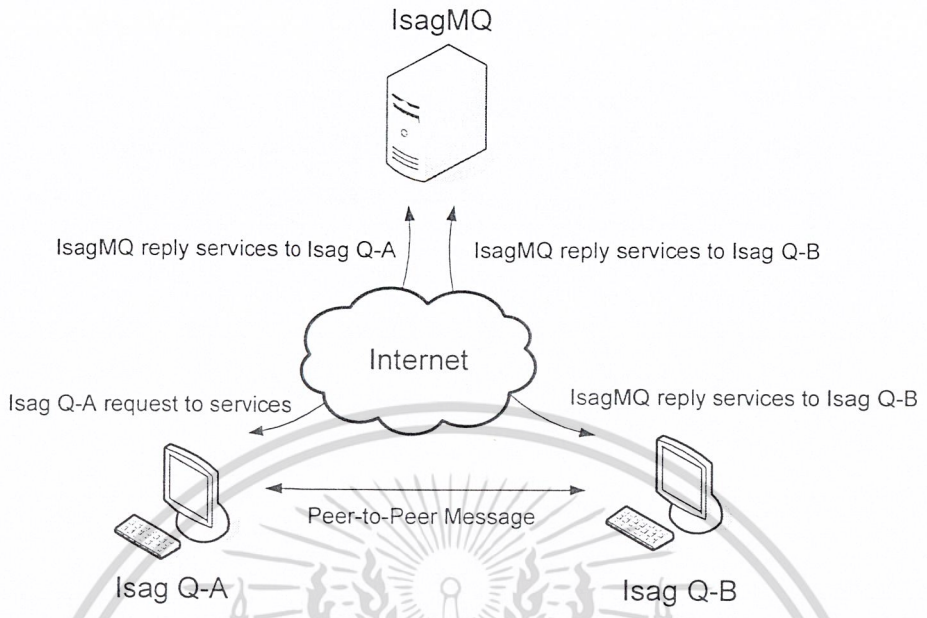
ระบบรับส่งสารด่วนในปัจจุบัน

ระบบรับส่งสารด่วนในปัจจุบันอาจจะจำแนกตามรูปแบบการติดต่อสื่อสารออกเป็น 2 แบบ ดังนี้

1. แบบ เพียร์ทูเพียร์ (Peer-to-Peer Communication)

พิจารณาจากรูปที่ 2-1 จะเห็นได้ว่าวิธีนี้จะเป็นลักษณะ การกระจายอำนาจ (Distribution) โดยเมื่อ ผู้ใช้ IsagQ-A และ IsagQ-B เป็นสมาชิกของ IsagMQ แล้ว เมื่อผู้ใช้ IsagQ-A ต้องการติดต่อกับ ผู้ใช้ IsagQ-B ผู้ใช้ทั้งสองจำเป็นที่จะต้องติดต่อไปยัง IsagMQ เสียก่อน เพื่อให้ทางเซิร์ฟเวอร์ได้ทราบว่าขณะนี้มีผู้ใช้คนใดอยู่ในสถานะที่พร้อมจะได้รับบริการ อีก ทั้งผู้ใช้แต่ละคนต้องใช้ข้อมูลรายละเอียดของกลุ่มผู้ใช้งานที่ตนเองมีสิทธิ์เข้าถึง ซึ่งข้อมูลเหล่านี้เซิร์ฟเวอร์จะเป็นคนมอบให้มาหลังจากที่ผู้ใช้งานได้ทำการติดต่อกับทางเซิร์ฟเวอร์เป็นที่เรียบร้อยแล้ว

หลังจากนั้น ผู้ใช้ IsagQ-A จะสามารถติดต่อไปยัง IsagQ-B ได้โดยตรง โดยที่ไม่ผ่าน IsagMQ ซึ่งรูปแบบนี้เป็นลักษณะแบบเพียร์ทูเพียร์ นั้นเอง ตัวอย่างของโปรแกรมรับส่งสารด่วนประเภทนี้ก็คือ ICQ

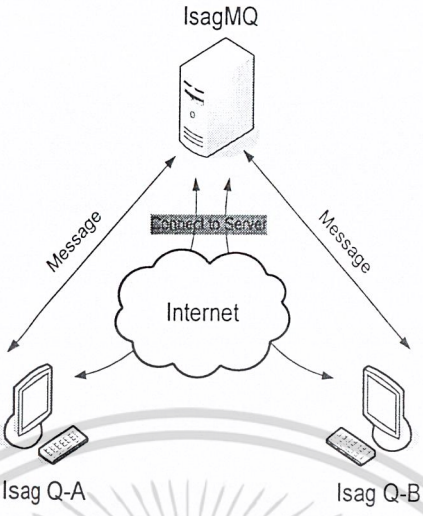


รูป 2-1 ระบบรับส่งสารด่วนแบบเพียร์ทูเพียร์

2. แบบ เซิร์ฟเวอร์ทำงานเป็นหลัก (Server-based Communication)

พิจารณารูป 2-2 แสดงการทำงานของเซิร์ฟเวอร์ จะเห็นได้ว่าวิธีนี้จะเป็นลักษณะ รวมอำนาจเข้ามาไว้ที่ศูนย์กลาง (Centralization) ผู้ใช้นอกจากจะต้อง ล๊อกอินไปยัง เซิร์ฟเวอร์แล้ว ทุกๆข้อความที่ผู้ใช้ต้องการส่งไปให้ใคร ก็จำเป็นที่จะต้องส่งไปยังเซิร์ฟเวอร์เพื่อให้เซิร์ฟเวอร์ทำการส่งข้อความต่อไปยังผู้ใช้ปลายทางอีกทีนึง

ด้วยวิธีการนี้ เซิร์ฟเวอร์จะต้องมีความสามารถนอกจากการให้บริการพื้นฐานแล้ว ยังต้องสามารถจัดลำดับการรับส่งข้อความของผู้ใช้ ณ เวลาจริง โดยข้อความใดมาก่อนก็จะได้รับการส่งไปยังปลายทางก่อน จะเห็นได้ว่า โปรแกรมทางฝั่งแม่ข่ายจะมีความซับซ้อนสูงมาก ตัวอย่างของระบบรับส่งสารด่วนประเภทนี้ก็คือ MSN Messenger



รูป 2-2 ระบบรับส่งสารด่วนแบบ Server-based

การเปรียบเทียบข้อดีข้อเสียของระบบทั้งสอง

ไม่ว่าจะเป็นระบบการส่งสารด่วนแบบเพียร์ทูเพียร์ หรือ แบบ เซิร์ฟเวอร์-เบส ต่างข้อมีข้อดีข้อเสียแตกต่างกันไปซึ่งจะสรุปดังตารางที่ 2-3

ความสามารถ	Peer-to-Peer	Server-based
เซิร์ฟเวอร์ทำงานหนัก	น้อย	มาก
ความรวดเร็วในการส่งข้อความ	มาก	น้อย
ความซับซ้อนของโพรโตคอล	น้อย	มาก
ความปลอดภัย (การเปิดเผย IP)	น้อย	มาก
ปัญหาเรื่องไฟร์วอลล์	มาก	น้อย
ค่าใช้จ่ายในการดูแลรักษา	น้อย	มาก

รูป 2-3 ตารางเปรียบเทียบข้อดีข้อเสียของระบบรับส่งสารด่วนในปัจจุบัน

ในการจะตัดสินในเลือกระบบรับส่งสารด่วนแบบใดขึ้นอยู่กับจุดประสงค์ของผู้ออกแบบ และความสามารถในการจัดการผลลัพธ์ที่เกิดขึ้น โดยในขั้นนี้ทางผู้พัฒนาได้เลือกระบบรับส่งสารด่วนแบบเพียร์ทูเพียร์ เนื่องจากมีความซับซ้อนน้อยกว่า จึงง่ายต่อการพัฒนา

ระบบรับส่งสารด่วนในอนาคต

แนวโน้มของการพัฒนาของระบบรับส่งสารด่วนเพิ่มขึ้นไปทุกวัน จากเดิมที่ผู้ใช้สามารถส่งได้เพียงข้อความที่ต้องสนทนาเท่านั้น ปัจจุบันกลับสามารถรองรับการสนทนาผ่านเสียง ผ่านภาพ รวมถึงมีลูกเล่นแตกต่างกันไปตามผู้สร้างสรรค์ สิ่งเหล่านี้ได้แทนที่ความต้องการในการใช้จดหมายอิเล็กทรอนิกส์ได้เป็นอย่างมาก นั้นก็เพราะมีความสะดวกรวดเร็ว ทันเวลา

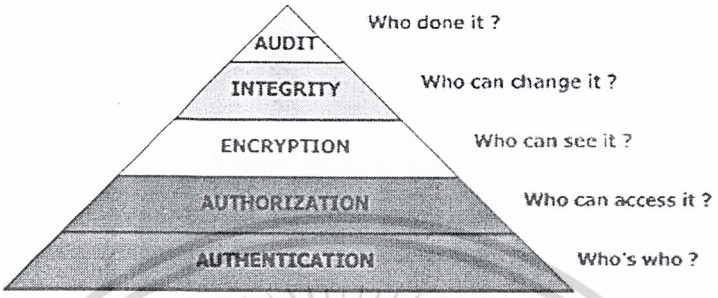
แต่ปัจจุบันด้านหนึ่งที่ระบบรับส่งสารด่วนยังไม่ได้ให้ความสำคัญมากก็คือ ด้านความปลอดภัย เช่นการเข้าและถอดรหัส รวมถึง การพิสูจน์ตน บางโปรแกรมสามารถรองรับความสามารถได้เพียงการเข้าและถอดรหัสเท่านั้น ดังนั้นหากผู้ใช้ส่งข้อความที่มีความสำคัญ และ ถูกดักจับข้อความไป ผู้ที่ดักจับข้อความก็สามารถทราบได้ว่าผู้ใช้คนดังกล่าวกำลังสนทนาอะไรกันอยู่ ทำให้เขาสามารถโขมยความลับได้ และในกรณีที่ผู้ประสงค์ร้ายปลอมแปลงข้อความส่งไปยังผู้อื่นทำให้เขาคิดว่าข้อความดังกล่าวมาจากสมาชิกของเขาเอง ผลลัพธ์ก็ยิ่งร้ายแรงขึ้นไป ด้วยเหตุนี้ หากเปรียบเทียบกับความนิยมที่ทวีความรุนแรงมากขึ้น กับ ความต้องการที่หลากหลายของผู้ใช้งาน น่าจะเป็นแรงขับดันให้ ระบบรับส่งสารด่วนในอนาคตมีความปลอดภัยมากยิ่งขึ้น ทั้งนี้มิใช่เพียงเพื่อการบันเทิงเท่านั้น แต่หากเพียงเพื่อเชิงพาณิชย์ อีกด้วย

บทที่ 3
นิยามความมั่นคงปลอดภัยคอมพิวเตอร์

ในปัจจุบันระบบคอมพิวเตอร์ได้ถูกคุกคามมากขึ้นทั้งจากไวรัสคอมพิวเตอร์หรือจากผู้ไม่ประสงค์ดี ซึ่งความมั่นคงปลอดภัยคอมพิวเตอร์ (Computer Security) ช่วยปกป้องเครื่องคอมพิวเตอร์รวมไปถึงอุปกรณ์ต่างๆที่เกี่ยวข้อง และที่สำคัญยังสามารถช่วยปกป้องข้อมูลที่ได้จัดเก็บไว้ภายในระบบหรือใช้ในความหมายความปลอดภัยทางข้อมูลสารสนเทศ (Information Security) ก็ได้ จุดประสงค์หลักของความปลอดภัยทางข้อมูลคือ ความลับ (Confidentiality) ความสมบูรณ์ (Integrity) ความพร้อมใช้ (Availability) และการห้ามปฏิเสธความรับผิดชอบ (Non-Repudiation) ของข้อมูลต่างๆภายในองค์กร (CIA-N) โดยมีรายละเอียดดังนี้

- **การรักษาความลับ (Confidentiality)** คือการรับรองว่าจะมีการเก็บข้อมูลไว้เป็นความลับ และผู้มีสิทธิเท่านั้นจึงจะเข้าถึงข้อมูลนั้นได้
- **การรักษาความสมบูรณ์ (Integrity)** คือการรับรองว่าข้อมูลจะไม่ถูกเปลี่ยนแปลงหรือทำลายไม่ว่าจะเป็นโดย อุบัติเหตุหรือ โดยเจตนา
- **ความพร้อมใช้ (Availability)** คือการรับรองว่าข้อมูลและบริการการสื่อสารต่าง ๆ พร้อมที่จะใช้ได้ในเวลาที่ต้องการใช้งาน
- **การห้ามปฏิเสธความรับผิดชอบ (Non-Repudiation)** คือวิธีการสื่อสารซึ่งผู้ส่งข้อมูลได้รับหลักฐานว่าได้มีการส่งข้อมูลแล้วและผู้รับก็ได้รับการยืนยันว่าผู้ส่งเป็นใคร ดังนั้นทั้งผู้ส่งและผู้รับจะไม่สามารถปฏิเสธได้ว่าไม่มีความเกี่ยวข้องกับข้อมูลดังกล่าวในภายหลัง

ในทางปฏิบัตินั้นสามารถกำหนดลักษณะของการควบคุมความมั่นคงปลอดภัย (Security Controls) ได้ 5 ระดับตามรูป



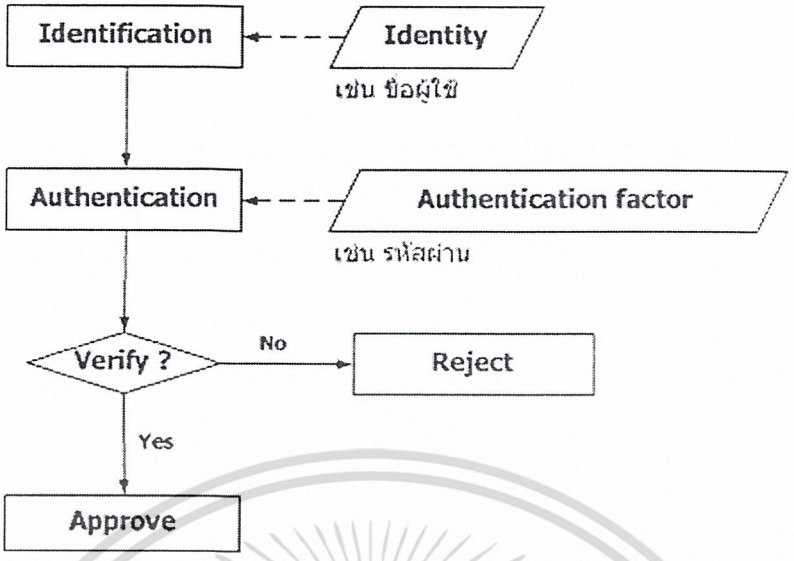
รูปที่ 3-1 แสดง Security Pyramid

และถือเป็นองค์ประกอบที่สำคัญส่วนหนึ่งของความมั่นคงปลอดภัยคอมพิวเตอร์ เพราะจัดเป็นการกำหนดและควบคุมทั้งบุคคลที่สามารถเข้าสู่ระบบและเข้าสู่ข้อมูลภายในระบบ และเพื่อกระทำการใดได้บ้าง อนุญาตตามระดับชั้นของความสำคัญของข้อมูล รวมไปถึงการจัดเก็บพฤติกรรมการใช้งานระบบของบุคคลนั้นต่อข้อมูลบนระบบทั้งหมด

1. การพิสูจน์ตัวตน (Authentication)

การพิสูจน์ตัวตน คือขั้นตอนการยืนยันความถูกต้องของหลักฐาน (Identity) ที่แสดงว่าเป็นบุคคลที่กล่าวอ้างจริง ในทางปฏิบัติจะแบ่งออกเป็น 2 ขั้นตอน คือ

- **การระบุตัวตน (Identification)** คือขั้นตอนที่ผู้ใช้แสดงหลักฐานว่าตนเองคือใครเช่น ชื่อผู้ใช้ (username)
- **การพิสูจน์ตัวตน (Authentication)** คือขั้นตอนที่ตรวจสอบหลักฐานเพื่อแสดงว่าเป็นบุคคลที่กล่าวอ้างจริง



รูปที่ 3-2 แผนผังแสดงกระบวนการการพิสูจน์ตัวตน

จากแผนผังแสดงกระบวนการพิสูจน์ตัวตน ในขั้นแรกผู้ใช้จะทำการแสดงหลักฐานที่ใช้ในการพิสูจน์ตัวตนต่อระบบ ซึ่งในขั้นนี้คือการระบุตัวตน และในขั้นตอนต่อมาระบบจะทำการตรวจสอบหลักฐานที่ผู้ใช้นำมากล่าวอ้างซึ่งก็คือการพิสูจน์ตัวตน หลังจากระบบได้ทำการตรวจสอบหลักฐานเรียบร้อยแล้วถ้าหลักฐานที่นำมากล่าวอ้างถูกต้องจึงอนุญาตให้เข้าสู่ระบบได้ หากหลักฐานที่นำมากล่าวอ้างไม่ถูกต้องผู้ใช้จะถูกปฏิเสธจากระบบ

หลักฐานที่ผู้ใช้นำมากล่าวอ้างที่เกี่ยวกับเรื่องของความปลอดภัยนั้นสามารถจำแนกได้ 2 ชนิด

- **Actual identity** คือหลักฐานที่สามารถบ่งบอกได้ว่าในความเป็นจริงบุคคลที่กล่าวอ้างนั้นเป็นใคร
- **Electronic identity** คือหลักฐานทางอิเล็กทรอนิกส์ซึ่งสามารถบ่งบอกข้อมูลของบุคคลนั้นได้ แต่ละบุคคลอาจมีหลักฐานทางอิเล็กทรอนิกส์ได้มากกว่า 1 หลักฐาน ตัวอย่างเช่น บัญชีชื่อผู้ใช้

กลไกของการพิสูจน์ตัวตน (**Authentication mechanisms**) สามารถแบ่งออกได้เป็น 3 คุณลักษณะคือ

- **สิ่งที่คุณมี** (Possession factor) เช่น กุญแจหรือเครดิตการ์ด เป็นต้น
- **สิ่งที่คุณรู้** (Knowledge factor) เช่น รหัสผ่าน (passwords) หรือการใช้พิน (PINs) เป็นต้น
- **สิ่งที่คุณเป็น** (Biometric factor) เช่น ลายนิ้วมือ รูปแบบเรตินา (retinal patterns) หรือใช้รูปแบบเสียง (voice patterns) เป็นต้น

กระบวนการพิสูจน์ตัวตนนั้นจะนำ 3 ลักษณะข้างต้นมาใช้ในการยืนยันหลักฐานที่นำมากล่าวอ้าง ทั้งนี้ขึ้นอยู่กับระบบ วิธีการที่นำมาใช้เพียงลักษณะอย่างใดอย่างหนึ่ง (Single-factor authentication) นั้นมีข้อจำกัดในการใช้ ตัวอย่างเช่น สิ่งที่คุณมี (Possession factor) นั้นอาจจะสูญหายหรือถูกขโมยได้ สิ่งที่คุณรู้ (Knowledge factor) อาจจะถูกดักฟัง เดา หรือขโมยจากเครื่องคอมพิวเตอร์ สิ่งที่คุณเป็น (Biometric factor) จัดได้ว่าเป็นวิธีที่มีความปลอดภัยสูงอย่างไรก็ตามการที่จะใช้เทคโนโลยีนี้ได้นั้นจำเป็นต้องมีการลงทุนที่สูง เป็นต้น

ดังนั้นจึงได้มีการนำแต่ละคุณลักษณะมาใช้ร่วมกัน (multi-factor authentication) ตัวอย่างเช่น ใช้สิ่งที่คุณมีกับสิ่งที่คุณรู้มาใช้ร่วมกัน เช่น การใช้ลายมือชื่อร่วมกับการใช้บัตรเครดิตหรือการใช้รหัสผ่านร่วมกับการใช้บัตร ATM เป็นต้น การนำแต่ละลักษณะของการพิสูจน์ตัวตนมาใช้ร่วมกันมากกว่า 1 ลักษณะ จะช่วยเพิ่มประสิทธิภาพในการรักษาความปลอดภัยของข้อมูล

2. การกำหนดสิทธิ์ (Authorization)

การกำหนดสิทธิ์ คือขั้นตอนในการอนุญาตให้แต่ละบุคคลสามารถเข้าถึงข้อมูลหรือระบบใดได้บ้าง ก่อนอื่นต้องทราบก่อนว่าบุคคลที่กล่าวอ้างนั้นคือใครตามขั้นตอนการพิสูจน์ตัวตนและต้องให้แน่ใจด้วยว่าการพิสูจน์ตัวตนนั้นถูกต้อง

3. การเข้ารหัส (Encryption)

การเข้ารหัส คือการเก็บข้อมูลให้เป็นส่วนบุคคลจากบุคคลอื่นที่ไม่ได้รับอนุญาต ส่วนประกอบ 2 ส่วนที่สำคัญที่จะช่วยทำให้ข้อมูลนั้นเป็นความลับได้ก็คือ การกำหนดสิทธิ์และการพิสูจน์ตัวตน เพราะว่าก่อนการอนุญาตให้บุคคลที่กล่าวอ้างเข้าถึงข้อมูลหรือถอดรหัสข้อมูลนั้นต้องสามารถแน่ใจได้ว่าบุคคลที่กล่าวอ้างนั้นเป็นใครและได้รับอนุญาตให้สามารถเข้ามาดูข้อมูลได้หรือไม่ ในการเข้ารหัส

ดังนั้นวิธีการหนึ่งที่ทำได้คือการเข้ารหัสในรูปแบบของกุญแจลับ (Secret key) ซึ่งในการใช้คีย์รูปแบบนี้ต้องเฉพาะผู้ที่มีกุญแจลับนี้เท่านั้นที่สามารถรับข้อมูลที่เข้ารหัสแล้วได้

4. การรักษาความสมบูรณ์ (Integrity)

การรักษาความสมบูรณ์ คือการรับรองว่าข้อมูลจะไม่ถูกเปลี่ยนแปลงหรือทำลายไปจากต้นฉบับ (source) ไม่ว่าจะเป็นโดยบังเอิญหรือดัดแปลงโดยเจตนาที่อาจส่งผลเสียต่อข้อมูล การคุกคามความสมบูรณ์ของข้อมูลคือการที่บุคคลที่ไม่ได้รับอนุญาตสามารถที่จะเข้าควบคุมการจัดการของข้อมูลได้

5. การตรวจสอบ (Audit)

การตรวจสอบ คือการตรวจสอบหลักฐานทางอิเล็กทรอนิกส์ ซึ่งสามารถใช้ในการติดตามการดำเนินการเพื่อตรวจสอบความถูกต้องและแม่นยำ ตัวอย่างเช่นการตรวจสอบบัญชีชื่อผู้ใช้ โดยผู้ตรวจบัญชี ซึ่งการตรวจสอบความถูกต้องของการดำเนินการเพื่อให้แน่ใจว่าหลักฐานทางอิเล็กทรอนิกส์นั้นได้ถูกสร้างและสั่งให้ทำงานโดยบุคคลที่ได้รับอนุญาต และในการเชื่อมต่อเหตุการณ์เข้ากับบุคคลจะต้องทำการตรวจสอบหลักฐานของบุคคลนั้นด้วย ซึ่งถือเป็นหลักการพื้นฐานของขั้นตอนการทำงานของการพิสูจน์ตัวตนด้วย

การพิสูจน์ตัวตนจัดเป็นการตรวจสอบหลักฐานขั้นพื้นฐานที่สำคัญที่สุดใน 5 ระดับชั้นของการควบคุมความปลอดภัย ดังนั้นการพิสูจน์ตัวตนดีจะช่วยเพิ่มความมั่นคงปลอดภัยขั้นพื้นฐานให้กับระบบมากยิ่งขึ้น

บทที่ 4
ศาสตร์ในการเข้ารหัสลับ
(Cryptography Overview)

ในส่วนนี้จะพูดถึงศาสตร์ในการเข้ารหัสลับที่สำคัญสำหรับโครงงานนี้ คือ

- การใช้ระบบกุญแจแบบไม่สมมาตร (Asymmetric Key)
- การใช้ลายมือชื่อดิจิตอล (Digital Signature)

การพิสูจน์ตัวตนโดยการเข้ารหัสโดยใช้กุญแจสาธารณะ (Public-key cryptography)

เป็นการรักษาความปลอดภัยของข้อมูลระหว่างการส่งข้ามเครือข่ายวิธีหนึ่งที่นิยมใช้กันอยู่ในปัจจุบัน การเข้ารหัสแบบคู่รหัสกุญแจนี้จะมีความปลอดภัยมากกว่าการเข้ารหัสข้อมูลแบบธรรมดา แต่ก็ไม่ได้หมายความว่าการเข้ารหัสแบบคู่รหัสกุญแจนี้จะเป็นวิธีที่เหมาะสมที่สุดของวิธีการเข้ารหัส ทั้งนี้ขึ้นอยู่กับประเภทงานของแต่ละองค์กรหรือบุคคล

การเข้ารหัสโดยใช้กุญแจสาธารณะ ประกอบไปด้วยกุญแจ 2 ชนิด

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

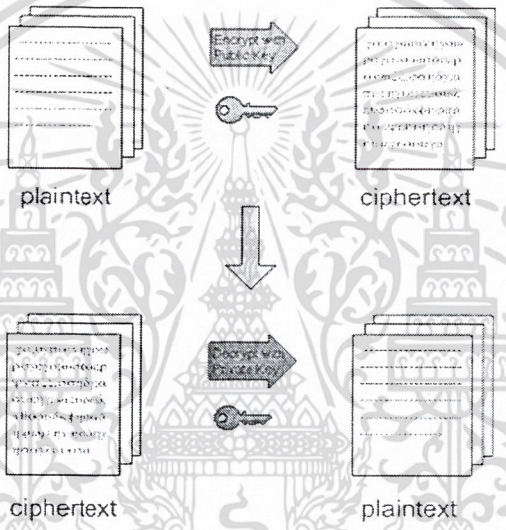
กระบวนการของการเข้ารหัสแบบคู่รหัสกุญแจ มีดังนี้

1. ผู้ใช้แต่ละคนจะสร้างคู่รหัสกุญแจของตัวเองขึ้นมา เพื่อใช้สำหรับการเข้ารหัสและการถอดรหัส
2. กุญแจสาธารณะจะถูกส่งออกไปยังผู้ใช้คนอื่นๆ แต่กุญแจส่วนตัวจะถูกเก็บที่ตนเอง
3. เมื่อจะส่งข้อมูลออกไปหาผู้ใช้คนใด ข้อมูลที่ส่งจะถูกเข้ารหัสด้วยกุญแจสาธารณะ ก่อนถูกส่งออกไป
4. เมื่อผู้รับได้รับข้อความแล้วจะใช้กุญแจส่วนตัวซึ่งเป็นคู่รหัสกันถอดรหัสออกมา

หมายเหตุ การเข้ารหัสโดยใช้กุญแจสาธารณะสามารถใช้ได้ทั้ง ในการเข้ารหัส (Encryption) และ การพิสูจน์ตัวตน (Authentication)

การประยุกต์ใช้ในการเข้ารหัสข้อมูล (Encryption)

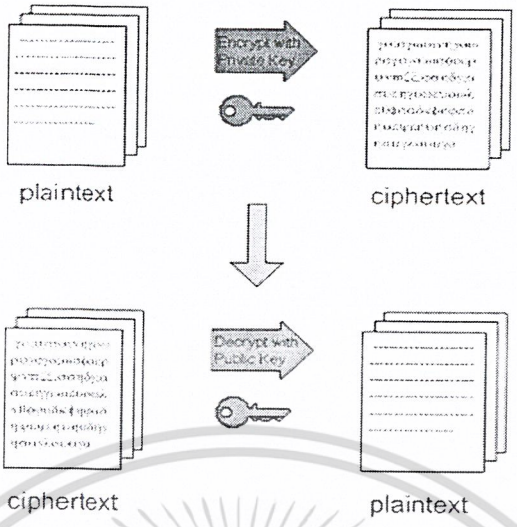
เป็นการนำข้อมูลที่จะส่งไปยังผู้รับมาเข้ารหัสด้วยกุญแจสาธารณะของผู้รับ และเมื่อผู้รับได้รับข้อความนั้นแล้วจะถอดรหัสออกมาด้วยกุญแจส่วนตัว จึงจะเห็นได้ว่ามีเพียงผู้รับเท่านั้นที่จะสามารถถอดรหัสออกมาได้



รูปที่ 4-1 ระบบของการเข้ารหัสแบบใช้คู่รหัสกุญแจ

การประยุกต์ใช้ในการพิสูจน์ตัวตน (Authentication)

เป็นการนำข้อมูลที่ผู้ส่งต้องการส่งมาเข้ารหัสด้วยกุญแจส่วนตัวของผู้ส่ง แล้วนำข้อมูลนั้นส่งไปยังผู้รับ ซึ่งผู้รับจะใช้กุญแจสาธารณะซึ่งเป็นคู่รหัสกันถอดรหัสออกมา ผู้รับก็สามารถรู้ได้ว่าข้อความนั้นถูกส่งมาจากผู้ส่งคนนั้นจริง ถ้าสามารถถอดรหัสข้อมูลได้อย่างถูกต้อง

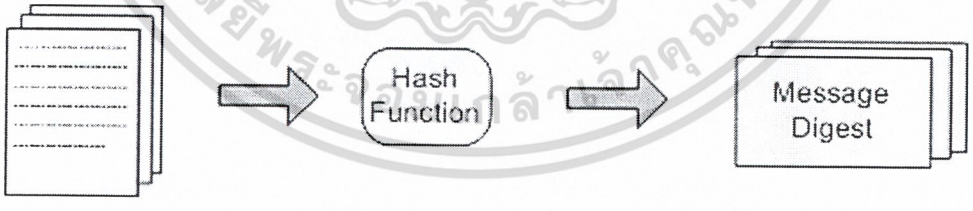


รูปที่ 4-2 ระบบของการเข้ารหัสแบบใช้คู่รหัสกุญแจเพื่อการพิสูจน์ตัวตน

การพิสูจน์ตัวตนโดยการใช้ลายอิเล็กทรอนิคส์ (Digital Signature)

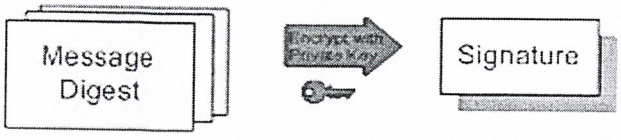
เป็นการนำหลักการของการทำงานของระบบการเข้ารหัสแบบใช้คู่รหัสกุญแจเพื่อการพิสูจน์ตัวตนมาประยุกต์ใช้ ระบบของลายดิจิตอลสามารถแบ่งเป็นขั้นตอนได้ดังนี้

เมื่อผู้ใช้ต้องการจะส่งข้อมูลไปยังผู้รับ ข้อมูลนั้นจะถูกนำไปเข้าฟังก์ชั่นทางคณิตศาสตร์ที่เรียกว่า "แฮชฟังก์ชัน" ได้เมสเซสไดเจสต์ (Message Digest) ออกมา



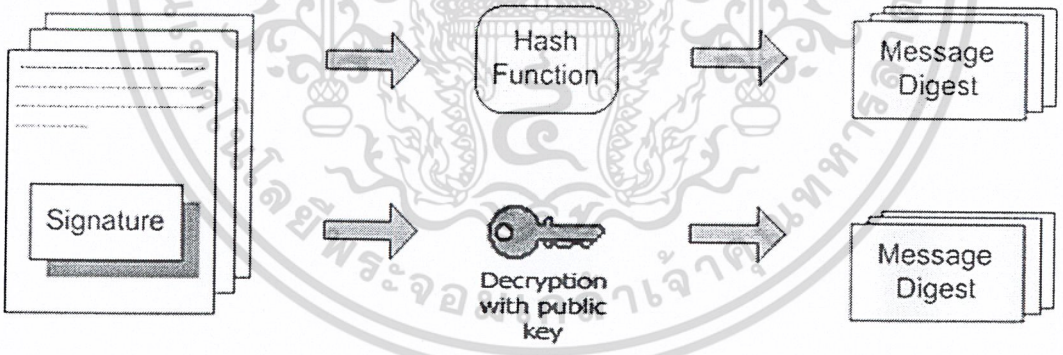
รูปที่ 4-3 การส่งข้อมูลเข้าไปใน Hash function

การใช้กุญแจส่วนตัวเข้ารหัสข้อมูล หมายถึงว่าผู้ส่งได้ลงลายดิจิตอล ยินยอมที่จะให้ผู้รับสามารถทำการตรวจสอบด้วยกุญแจสาธารณะของผู้ส่งเพื่อพิสูจน์ตัวตนของผู้ส่งได้



รูปที่ 4-4 การเข้ารหัสเมสเซสไดเจสต์ด้วยกุญแจส่วนตัวเพื่อเป็นการลงลาย

การตรวจสอบข้อมูลว่าถูกส่งมาจากผู้ส่งคนนั้นจริงในด้านผู้รับ โดยการนำข้อมูลมาผ่านแฮชฟังก์ชันเพื่อคำนวณหาค่าเมสเซจไดเจสต์ และถอดรหัสลายอิเล็กทรอนิกส์ด้วยกุญแจสาธารณะของผู้ส่ง ถ้าสามารถถอดได้อย่างถูกต้อง จะเป็นการยืนยันข้อมูลจากผู้ส่งคนนั้นจริง และถ้าข้อมูลเมสเซจไดเจสต์ที่ได้จากการถอดรหัสเท่ากันกับค่าเมสเซจไดเจสต์ในตอนต้นที่ทำการคำนวณได้ จะถือว่าข้อมูลดังกล่าวนั้นถูกต้อง



รูปที่ 4-5 ขั้นตอนการเปรียบเทียบความถูกต้อง

ลายเซ็นอิเล็กทรอนิคส์นิยมนำไปใช้ในระบบรักษาความปลอดภัยในการชำระเงินผ่านระบบอินเตอร์เน็ต ซึ่งในปัจจุบันนี้การทำธุรกรรมการเงินอิเล็กทรอนิกส์ได้รับความนิยมเป็นอย่างมาก

บทที่ 5

เอสเอสแอล ,โอเพนเอสเอสแอล และ ระบบการรับรองสิทธิ์ผู้ใช้

(SSL,OpenSSL and Certificate Authority (CA))

1. SSL/TLS

SSL โพรโตคอลเป็นโพรโตคอลที่ได้รับการออกแบบมา เพื่อทำให้เกิดการสื่อสารอย่างปลอดภัยขึ้น โดยโพรโตคอลสแต็กของ SSL จะอยู่ตรงกลางระหว่างชั้นแอพพลิเคชั่นเลเยอร์ (Application Layer) กับชั้นทรานสปอร์ตเลเยอร์(Transport Layer) นั่นคือ ชั้น SSL Layer จะเป็นชั้นที่เอาไว้ใช้สำหรับการสื่อสารที่ปลอดภัย ข้อดีคือเราสามารถพัฒนาแอพพลิเคชันได้อย่างเต็มที่ เนื่องจากการทำให้เกิดความปลอดภัยจะเป็นภาระหน้าที่ของชั้น SSL โดย SSL จะอาศัยทีซีพีโพรโตคอลในการรับส่งข้อมูล เพราะการส่งข้อมูลของ SSL ข้อมูลที่ถูกส่งไปนั้นเป็นข้อมูลที่ถูกเข้ารหัส ถ้าเกิดข้อมูลส่วนใดสูญหายขึ้นมาในระหว่างการส่งข้อมูลจะทำให้การถอดรหัสข้อมูลจะได้ข้อมูลที่ไม่ถูกต้องทีซีพีจะเป็นตัวช่วยรับประกันว่าข้อมูลที่ถูกส่งไปผู้รับจะต้องได้รับอย่างครบถ้วน ดังนั้น SSL สามารถที่พัฒนาโพรโตคอลที่จะทำให้เกิดความปลอดภัยที่สูงขึ้นได้โดยไม่ต้องไปกังวลกับความถูกต้องในการส่งข้อมูล

1.1 บทบาทของ SSL (SSL Role)

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

SSL จะถูกกำหนดให้มีสองบทบาทสำหรับขบวนการติดต่อสื่อสารกันระหว่างสองระบบ 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

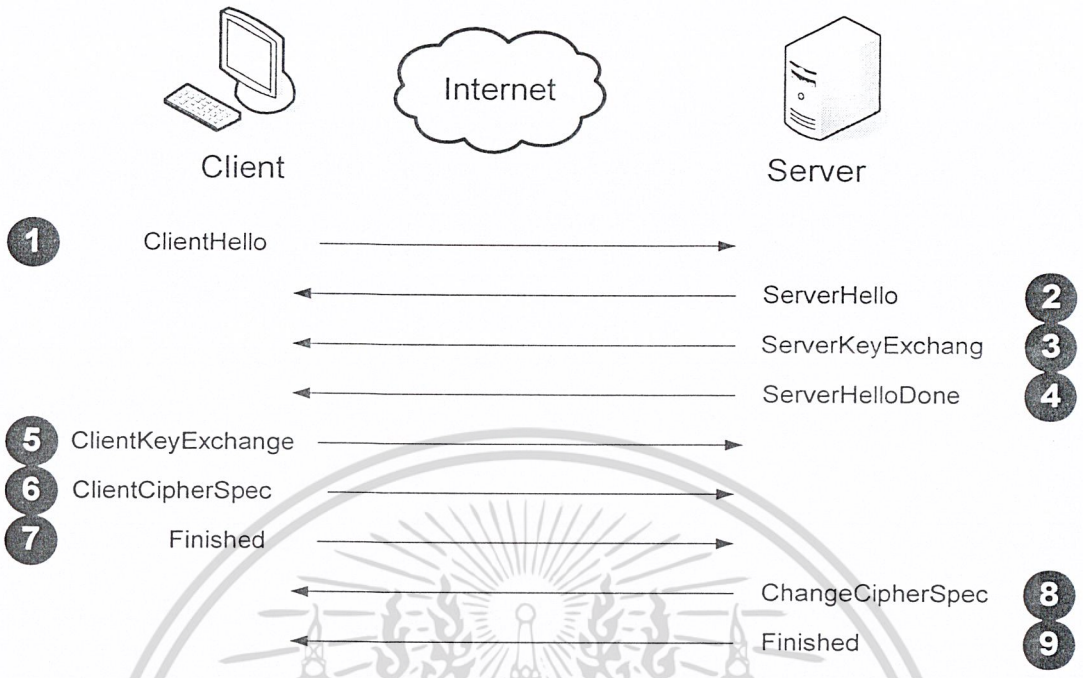
ไคลเอ็นต์จะเป็นผู้เริ่มการติดต่อสื่อสาร ซึ่งเป็นค่านำเสนอค่าพารามิเตอร์ต่าง ๆ ที่ตนเองสามารถรองรับได้ส่งไปให้แก่เซิร์ฟเวอร์ และเซิร์ฟเวอร์จะเป็นผู้ทำการตัดสินใจขั้นสุดท้ายว่าทั้งสองระบบนี้จะใช้พารามิเตอร์ตัวใดในการสร้างการติดต่อสื่อสารที่ปลอดภัย ถึงแม้ว่าการตัดสินใจขั้นสุดท้ายจะอยู่ที่ฝั่งเซิร์ฟเวอร์ แต่มีข้อจำกัดอยู่ว่าเซิร์ฟเวอร์จะสามารถเลือกค่าพารามิเตอร์ต่างได้จากค่าพารามิเตอร์ที่ไคลเอ็นต์ได้ส่งมาให้แล้วเท่านั้น

1.2 ข้อความของ SSL (SSL Message)

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

1.3 การสร้างการเชื่อมต่อแบบ SSL โพรโตคอลโดยการใช้การเข้ารหัสข้อมูล

กระบวนการพื้นฐานในการสร้างการเชื่อมต่อที่ปลอดภัยโดยใช้ SSL โพรโตคอล คือการใช้การเข้ารหัสข้อมูลโดยไคล์เอ็นต์จะรับกุญแจสาธารณะ(Public key) ที่เซิร์ฟเวอร์ได้ส่งมาให้แล้วทำการสร้างกุญแจลับ(Secret key) นำกุญแจลับที่ได้มาเข้ารหัสด้วยกุญแจสาธารณะที่ได้รับมา จากนั้นส่งกับไปให้แก่เซิร์ฟเวอร์และใช้กุญแจลับนี้ในการเข้ารหัสข้อมูลที่จะทำการส่งและถอดรหัสข้อมูลที่ได้รับ ซึ่งกระบวนการดังกล่าวจะเป็นการส่งข้อความของ SSL ตามเฟสต่าง ๆ ดังรูป 5-1 ซึ่งสามารถอธิบายได้ดังนี้



รูปที่ 5-1 จะใช้ ขบวนการสร้างการเชื่อมต่อโดย SSL แบบเข้ารหัสอย่างเดียว

1. **ClientHello**

ClientHello จะเป็นข้อความที่ไคลเอ็นต์ใช้สำหรับเริ่มต้นการสื่อสารด้วย SSL โพรโตคอลโดยข้อความนี้จะเป็นการนำเสนอค่าพารามิเตอร์ต่าง ๆ ที่ตัวเองสามารถรองรับได้ส่งไปให้แก่เซิร์ฟเวอร์ โดยองค์ประกอบที่สำคัญของ ClientHello มีดังนี้

- **Version** จะเป็นตัวบอกให้ทราบว่าเวอร์ชันสูงสุดของ SSL โพรโตคอลที่ไคลเอ็นต์สมารถรองรับได้ ซึ่ง SSL เวอร์ชันสาม เป็นเวอร์ชันที่นิยมนำไปใช้กันอย่างกว้างขวางในระบบอินเทอร์เน็ต แต่ในมุมมองของเซิร์ฟเวอร์จะมองว่าไคลเอ็นต์สามารถรองรับได้ทุกเวอร์ชัน เช่น ไคลเอ็นต์ส่ง ClientHello มาโดยระบุเป็น SSL เวอร์ชันสามแต่เซิร์ฟเวอร์สามารถรองรับได้สูงสุดเพียงเวอร์ชันสอง เซิร์ฟเวอร์จะเลือกใช้เวอร์ชันสอง ในกรณีนี้ไคลเอ็นต์สามารถที่จะทำการติดต่อสื่อสารกับเซิร์ฟเวอร์ต่อไปโดยใช้ SSL เวอร์ชันสองหรือจะยกเลิกการติดต่อสื่อสารกับเซิร์ฟเวอร์ไปเลยก็ได้
- **RandomNember** ตัวเลขสุ่มโดยฟิลด์นี้จะมีค่า 32 ไบต์โดย 4 ไบต์แรกจะเป็นค่าของวันและเวลาเหตุที่ต้องมีการระบุวันเวลาเพราะว่าเพื่อเป็นการสร้างความมั่นใจว่าในการติดต่อสื่อสารหนึ่งครั้งจะไม่มีไคลเอ็นต์คนใดที่จะใช้ตัวเลขสุ่มตัวเดียวกันถึงสองครั้ง นั่นคือมันสามารถที่จะแก้ไขปัญหาของการโจมตีแบบ

แอ็กชันรีเพลย์ (Action Replay) ได้ โดยค่าที่เหลืออีก 28 ไบต์จะเป็นตัวเลขสุ่มซึ่งจำเป็นต้องสร้างมาจากฟังก์ชันที่สามารถสร้างตัวเลขสุ่มได้อย่างปลอดภัย

- **SessionID** ในการติดต่อสื่อสารระหว่างไคล์เอ็นต์และเซิร์ฟเวอร์จะเรียกสั้น ๆ ว่า เซสชั่น (Session) ถ้าเซิร์ฟเวอร์ทำการเก็บเซสชั่นของไคลเอ็นต์เอาไว้เมื่อไคลเอ็นต์ทำการติดต่อเข้ามาใหม่เซิร์ฟเวอร์จะสามารถตรวจสอบได้จากเซสชั่นที่ได้เก็บไว้ทำให้การติดต่อสื่อสารมีความรวดเร็วขึ้นแต่การใช้เซสชั่นใน SSL นั้นมีความยุ่งยากและซับซ้อน โดยปกติแล้วฟิลด์นี้จะไม่มีความหมายใดๆ เมื่อเราใช้การติดต่อ SSL แบบปกติ
- **CipherSuite** จะเป็นฟิลด์ที่ไคลเอ็นต์จะทำการใส่ลิสต์ของอัลกอริทึมในการเข้ารหัสถอดรหัสที่ไคลเอ็นต์สามารถรองรับได้

2. **ServerHello**

เมื่อเซิร์ฟเวอร์ได้รับข้อความ ClientHello เซิร์ฟเวอร์จะทำการตอบกลับด้วย ServerHello ไคลเอ็นต์จะนำเสนอค่าพารามิเตอร์ต่างๆ ให้แก่เซิร์ฟเวอร์ เซิร์ฟเวอร์จะทำการเลือกค่าพารามิเตอร์เหล่านั้นแล้วส่งกลับไปด้วยข้อความ ServerHello โดยฟิลด์ของ ServerHello จะเป็นดังนี้

- **Version** จะเป็นฟิลด์ที่ระบุการติดต่อสื่อสารระหว่างไคลเอ็นต์และเซิร์ฟเวอร์จะใช้ SSL เวอร์ชั่น
- **RandomNumber** เซิร์ฟเวอร์จะรับตัวเลขสุ่มจากไคลเอ็นต์และนำค่าตัวเลขนั้นมาสุ่มอีกครั้งด้วยฟังก์ชั่นเดียวกับไคลเอ็นต์ ซึ่งไคลเอ็นต์จะใช้ตัวเลขสุ่มนี้ในการสร้างกุญแจลับ ซึ่งจะเป็นเซสชันคีย์ที่ใช้ในเซสชั่นของไคลเอ็นต์และเซิร์ฟเวอร์
- **SessionID** จะไม่ใช่ค่าเดียวกันกับ SessionID ของไคลเอ็นต์ ซึ่งค่านี้จะเป็นค่าที่ใช้ในการระบุถึงไคลเอ็นต์ใดๆ ที่เชื่อมต่อเข้ามายังเซิร์ฟเวอร์
- **CipherSuite** ในฝั่งของไคลเอ็นต์ค่านี้จะอยู่ในรูปพหูพจน์ เซิร์ฟเวอร์จะทำการเลือกค่าและค่าที่ได้จะอยู่ในรูปของเอกพจน์

3. ServerKeyExchange

เมื่อเซิร์ฟเวอร์ทำการส่งข้อความ ServerHello ไปแล้ว เฟสต่อไปเซิร์เวอร์จะทำการส่งข้อความ SeverKeyExchange ไปให้แก่ฝั่งไคลเอ็นต์ ในขณะที่ฟิลด์ CipherSuite จะเป็นตัวที่บอกถึงชนิดของอัลกอริทึมและขนาดของคีย์ข้อความ ServerKeyExchange จะเป็นค่าของกุญแจสาธารณะของเซิร์ฟเวอร์ ที่จะส่งไปให้ไคลเอ็นต์ใช้สำหรับเข้ารหัสกุญแจลับ ซึ่งกุญแจสาธารณะนี้จะไม่มีการเข้ารหัสใดๆ เพราะกุญแจสาธารณะมีความปลอดภัยในตัวมันเองอยู่แล้ว

4. ServerHelloDone

จะเป็นข้อความที่บอกให้ไคลเอ็นต์ทราบว่าเซิร์ฟเวอร์ได้ส่งข้อมูลที่ใช้สำหรับการเริ่มต้นการติดต่อสื่อสารเรียบร้อยแล้วซึ่งข้อความนี้จะไม่มีข้อมูลใด ๆ แต่มันเป็นสิ่งสำคัญต่อไคลเอ็นต์จะต้องได้รับข้อความนี้ก่อน ไคลเอ็นต์จึงจะสามารถกระทำเฟสต่อไปของการสร้างการเชื่อมต่อที่ปลอดภัยโดยใช้ SSL โพรโตคอลได้

5. ClientKeyExchange

เมื่อเซิร์ฟเวอร์สิ้นสุดการต่อรองค่าพารามิเตอร์ต่างๆ ที่ใช้ในการสื่อสารด้วย SSL โพรโตคอลแล้ว ไคลเอ็นต์จะตอบกลับด้วยข้อความ ClientKeyExchange ซึ่งคือการส่งกุญแจลับที่ได้เข้ารหัสด้วยกุญแจสาธารณะที่ได้รับมาจากเซิร์ฟเวอร์
เข้ารหัสถอดรหัสข้อมูลแบบกุญแจเดี่ยว (Symmetric key) จะใช้กุญแจตัวเดียวในการเข้ารหัสถอดรหัสข้อมูล ดังนั้นจึงไม่สามารถส่งกุญแจนั้นผ่านระบบเน็ตเวิร์คได้ SSL โพรโตคอลทำการแก้ปัญหาโดยให้เซิร์ฟเวอร์ส่งกุญแจสาธารณะมาให้แก่ไคลเอ็นต์ ไคลเอ็นต์จะทำการสร้างเซสชั่นคีย์แล้วนำเซสชั่นคีย์ที่ได้มาเข้ารหัสด้วยกุญแจสาธารณะที่ได้รับมาจากเซิร์ฟเวอร์ เมื่อส่งข้อมูลนี้ผ่านระบบเน็ตเวิร์คจะมีเพียงแค่เซิร์ฟเวอร์ที่ส่งกุญแจสาธารณะมาให้เท่านั้นที่จะสามารถทำการถอดรหัสข้อมูลได้ ซึ่งเป็นการทำให้ไคลเอ็นต์มั่นใจว่ามีเพียงแค่เซิร์ฟเวอร์ที่เป็นเจ้าของกุญแจสาธารณะเท่านั้นที่จะสามารถถอดรหัสข้อมูลได้ ทำให้เกิดความปลอดภัยขึ้นมาได้ในระดับหนึ่ง

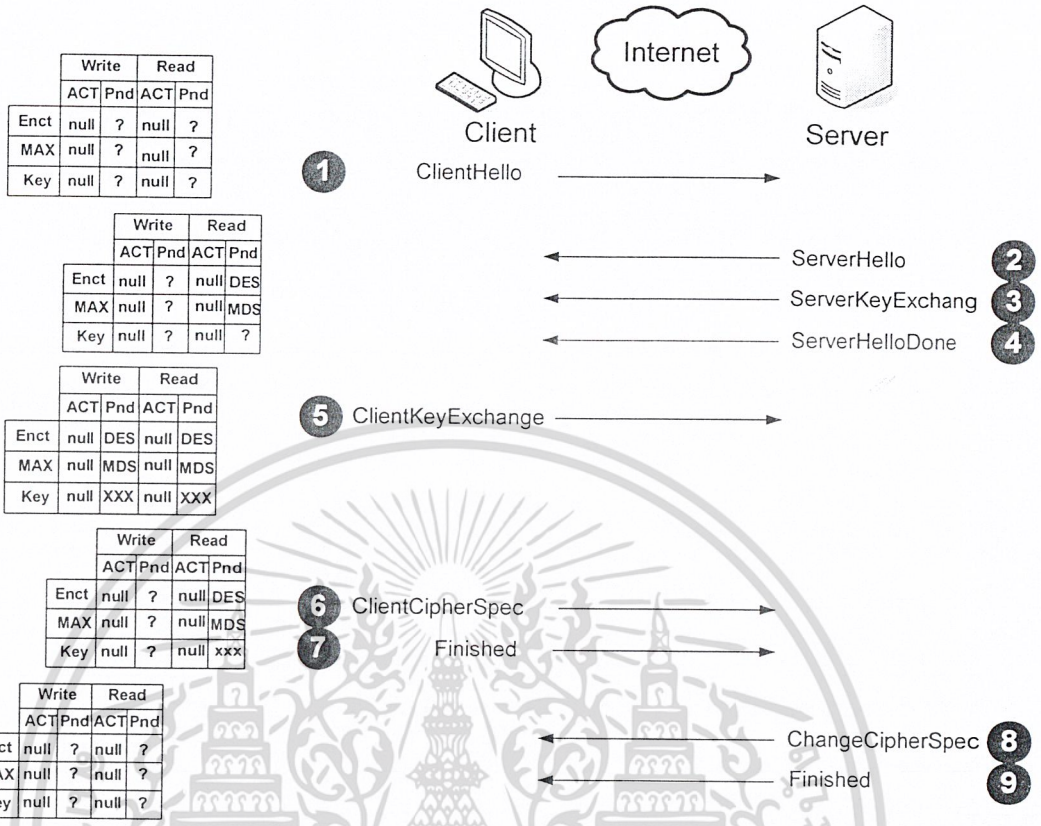
6. ChangeCipherSpec

หลังจากที่ไคลเอ็นต์ส่งข้อความ ClientKeyExchange เรียบร้อยแล้วจะถือว่าสิ้นสุดการต่อรองค่าพารามิเตอร์ต่าง ๆ ระหว่างไคลเอ็นต์กับเซิร์ฟเวอร์ ณ จุดนี้ทั้งสองระบบสามารถที่จะสื่อสารกันได้อย่างปลอดภัยโดยใช้ SSL โพรโตคอล ข้อความ ChangeCipherSpec จะเป็นข้อความที่ชี้ให้เห็นอย่างชัดเจนว่าการรับข้อมูลอย่างปลอดภัยโดยใช้ SSL โพรโตคอลสามารถทำงานได้แล้ว

กระบวนการในการรับส่งข้อมูลของ SSL โพรโตคอล ข้อมูลที่ทำการส่งไปจะต้องประกอบไปด้วยชนิดของอัลกอริทึมที่ใช้ในการเข้ารหัส,ข้อมูลที่ใช้ตรวจสอบความถูกต้องของข้อมูล(MIC),กุญแจที่ได้มาจากอัลกอริทึมดังกล่าว SSL โพรโตคอลจะต้องมีการตรวจสอบข้อมูลหลายๆ อย่างโดยเฉพาะอย่างยิ่ง เซสชั่นคีย์ โดยการตรวจสอบทิศทางระหว่างทิศทางในการรับและการส่งต้องมีความแตกต่างกัน นั่นคือ เซตของคีย์หนึ่งจะเป็นตัวใช้รักษาความปลอดภัยสำหรับไคลเอ็นต์ในการส่งข้อมูล และเซตของอีกคีย์หนึ่งจะเป็นตัวรักษาความปลอดภัยสำหรับไคลเอ็นต์ในการรับข้อมูล) ถ้ามีการแยกในความแตกต่างของอัลกอริทึมที่ใช้ทั้งในการรับและการส่งข้องมูลก็จะเป็นวิธีที่ดีแต่ SSL โพรโตคอลไม่มีตัวเลือกนี้อยู่(

ทั้งในฝั่งไคลเอ็นต์และฝั่งเซิร์ฟเวอร์ SSL โพรโตคอลจะกำหนดให้มีสภาวะในการรับข้อมูลเรียกว่า read state และในการส่งข้อมูลเรียกว่า write state จากรูปจะเห็นว่า read และ write state จะถูกแบ่งออกเป็นอีกสองสภาวะคือ active และ pending ดังนั้นจะมีทั้งหมดสี่สภาวะด้วยกันคือ active write state, pending write state, active read state, pending read state active

จากรูปใช้ตัวย่อเป็น Act และ pending จากรูปใช้ตัวย่อเป็น Pnd อัลกอริทึมในการเข้ารหัสถอดรหัสใช้ตัวย่อเป็น Encr ข้อมูลที่ใช้ตรวจสอบความถูกต้องของข้อมูลใช้ตัวย่อเป็น MAC (Message Authentication Code) และคีย์จะใช้เป็น key จากรูป 5-2 และ 5-3 ซึ่งสามารถอธิบายทั้งสี่สภาวะได้ดังนี้



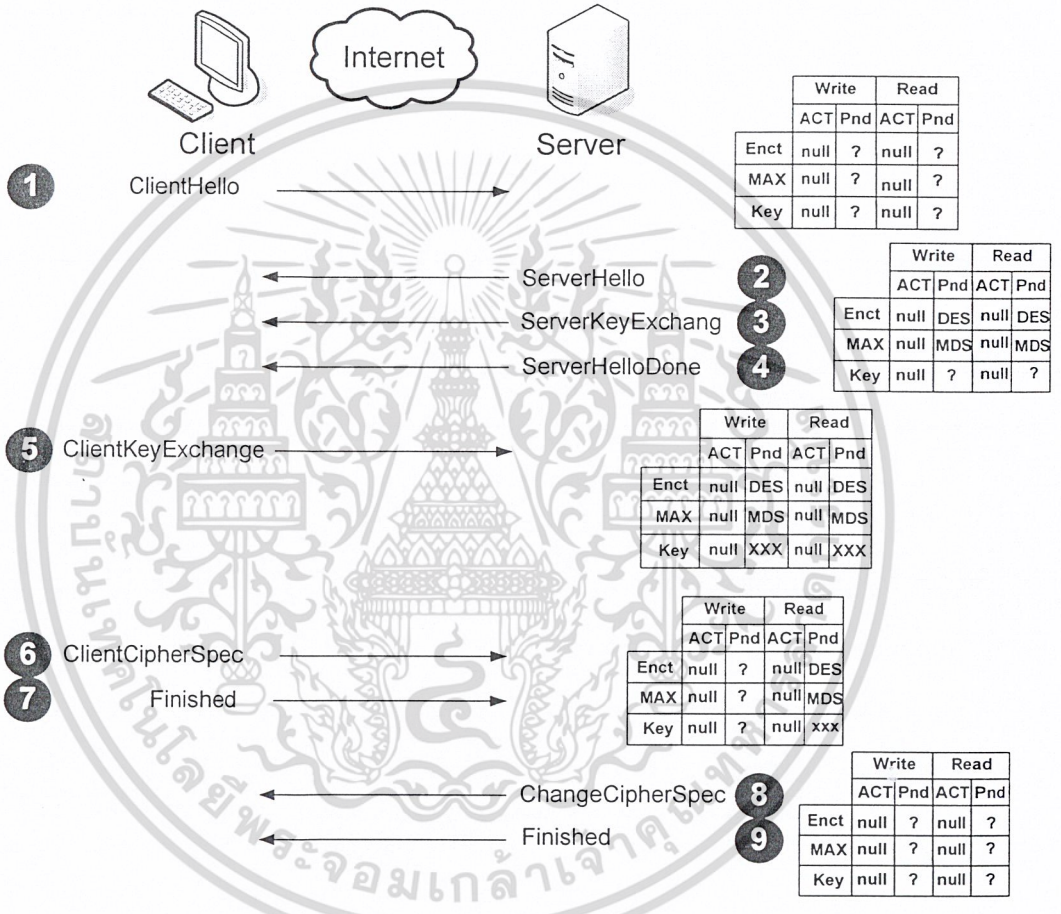
รูปที่ 5-2

แสดงสภาวะในการรับส่งข้อมูลของไคล์เอ็นต์ตามกระบวนการของ SSL โพรโตคอล

ฝั่งไคลเอ็นต์

1. เมื่อไคลเอ็นต์เริ่มต้นการสื่อสารด้วย SSL โพรโตคอลโดยทำการส่งข้อความ ClientHello จะทำการเซตสภาวะ active ทั้งสองสภาวะเป็น null สภาวะ pending จะไม่มีการทำอะไรกับมัน
2. เมื่อไคลเอ็นต์ได้รับข้อความ ServerHello ในขณะนี้ไคลเอ็นต์จะทราบว่าเซิร์ฟเวอร์ทำการเลือกอัลกอริทึมใดในการติดต่อเซสชั่นนี้ ไคลเอ็นต์ทำการปรับข้อมูลของอัลกอริทึมในการเข้ารหัส ถอดรหัส และ ข้อมูลที่ใช้ตรวจสอบความถูกต้องของข้อมูลในสภาวะ pending ทั้งสอง จากข้อมูลที่ได้รับมาดังรูป
3. เมื่อไคลเอ็นต์ได้สร้างเซสชั่นคีย์และส่งข้อความ ClientKeyExchange เรียบร้อยแล้ว จะทำให้ทราบถึงคีย์ที่จะใช้ในการติดต่อสื่อสาร จึงทำการปรับข้อมูลของคีย์ในสภาวะ pending ทั้งสอง ดังรูป

4. เมื่อไคลเอ็นต์ส่งข้อความ ChangeCipherSpec ที่สภาวะ write จะเปลี่ยนจากสภาวะ pending ไปเป็นสภาวะ active พร้อมกับปรับข้อมูลจากข้อมูลของสภาวะ pending และทำการรีเซตค่าสภาวะ pending ให้ไม่มีค่าใด ๆ ณ จุดนี้ไคลเอ็นต์จะสามารถทำการส่งข้อมูล (ciphertext) ที่ถูกเข้ารหัสด้วยอัลกอริทึม DES และ ค่าตรวจสอบความถูกต้องที่ใช้อัลกอริทึม MD5 ได้



รูปที่ 5-3

แสดงสภาวะในการรับส่งข้อมูลของเซิร์ฟเวอร์ตามขบวนการของ SSL โพรโตคอล

ฝั่งเซิร์ฟเวอร์

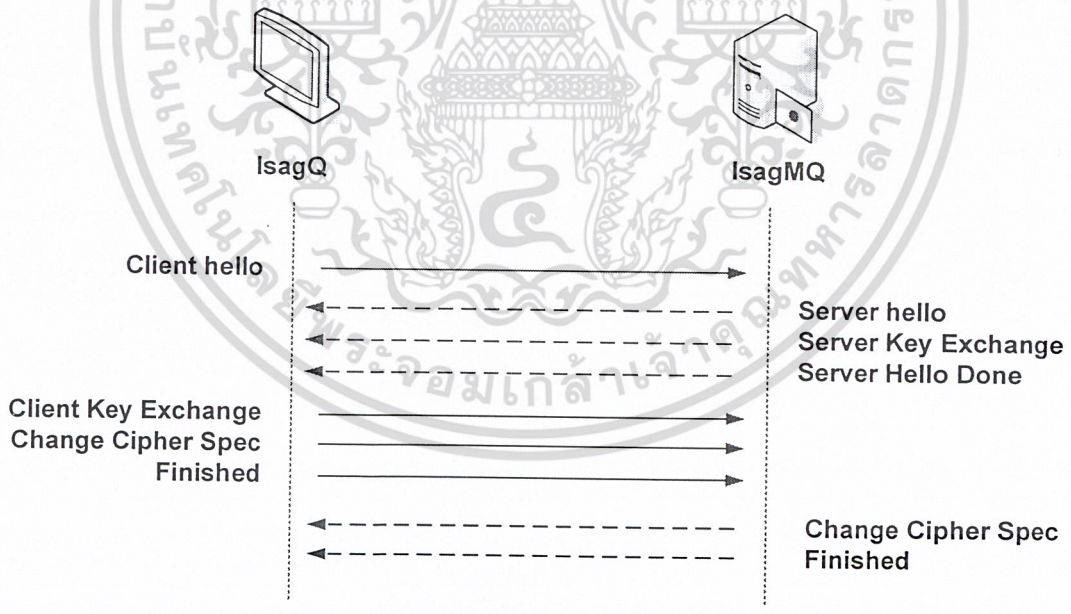
1. เมื่อเซิร์ฟเวอร์ได้รับข้อความ ClientHello มันจะทำการเซตค่าสภาวะ active ของทั้งสองให้เป็น null และสภาวะ pending จะไม่มีการทำอะไรกับมัน
2. เมื่อเซิร์ฟเวอร์ทำการส่งข้อความ ServerHello ณ จุดนี้มันจะทราบถึงอัลกอริทึมที่ใช้สำหรับเซสชั่นนี้และจะทำการปรับปรุงข้อมูลเหล่านี้ในสภาวะของ pending ดังรูป แต่ข้อมูลของคีย์ในสภาวะ pending ในตอนนี้จะอยู่ในสถาวะไม่ทราบค่า
3. เมื่อเซิร์ฟเวอร์ ได้รับข้อความ ClientKeyExchange จะทำให้มันทราบค่าของคีย์ที่จะใช้ในการติดต่อสื่อสารของเซสชั่นนี้ ดังนั้นมันจะทำการปรับปรุงข้อมูลของคีย์ในสภาวะ pending
4. เมื่อเซิร์ฟเวอร์ได้รับข้อความ ChangeCipherSpec ที่สภาวะ read มันจะทำการเปลี่ยนจากสภาวะ pending ไปเป็นสภาวะ active พร้อมกับปรับข้อมูลจากข้อมูลของ สภาวะ pending และทำการรีเซตค่าสภาวะ pending ให้ไม่มีค่าใดๆ ณ จุดนี้เซิร์ฟเวอร์สามารถที่จะรับข้อมูล (ciphertext) ที่ถูกเข้ารหัสด้วยอัลกอริทึม DES และค่าตรวจสอบความถูกต้องของข้อมูลที่ใช้อัลกอรึทึม MD5 ได้

7. **Finished**

ทันทีหลังจากที่ได้ทำการส่งขอความ ChangeCipherSpec แต่ละระบบจะทำการส่งข้อความ Finished ซึ่งเป็นข้อความที่ใช้การตรวจสอบความถูกต้องของข้อมูลที่ได้ทำการส่งไป เช่น ข้อมูลเกี่ยวกับคีย์ รายละเอียดเกี่ยวกับการต่อรองค่าพารามิเตอร์ต่างๆ ในครั้งก่อน และข้อมูลใดๆ ที่จะเป็นการตัวระบุตัวตนของทั้งเซิร์ฟเวอร์และไคลเอ็นต์ โดยค่าเหล่านี้จะต้องทำให้เป็นแฮชแวลู (hash value)
ก่อนทำการส่งออกไป

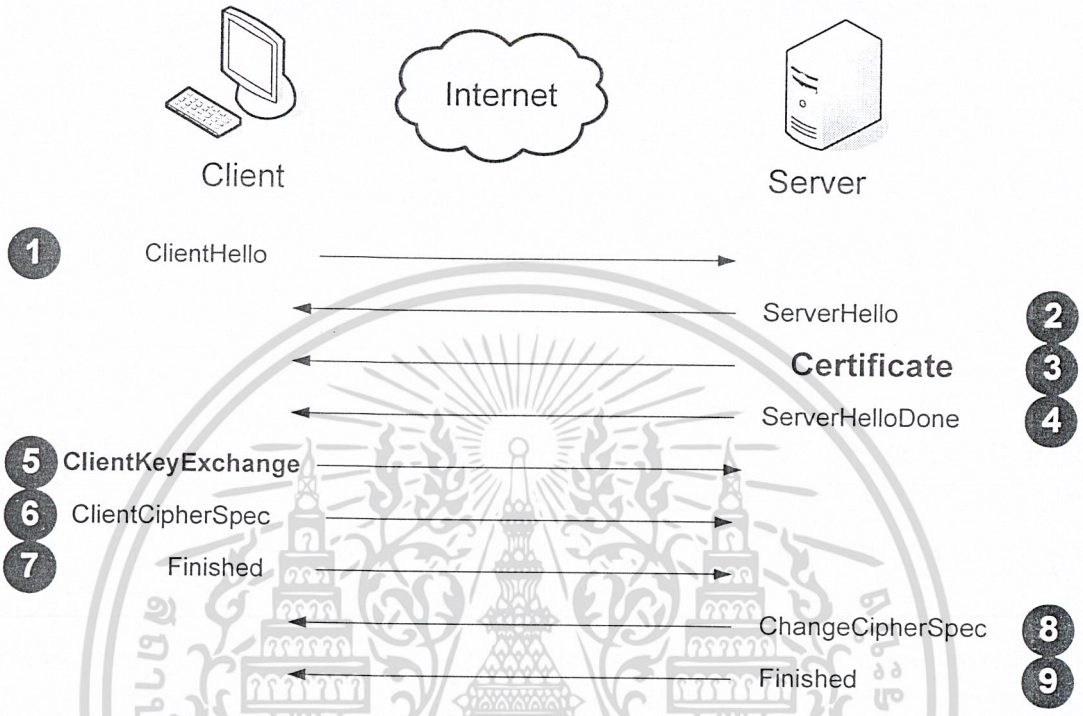
1.4 การสร้างการเชื่อมต่อแบบ SSL โพรโตคอลโดยใช้การเข้ารหัสข้อมูลและมีการพิสูจน์ตนของเซิร์ฟเวอร์

กระบวนการการทำงานของ SSL โพรโตคอลในข้างต้นที่ใช้การเข้ารหัสข้อมูลเพียงอย่างเดียวระหว่างสองระบบนั้นมันยังไม่มีความปลอดภัยเพียงพอ พิจารณาได้จากรูปที่ 5-4 เมื่อ Alice ที่มีบทบาทเป็นไคล์เอ็นต์และ Bob ที่บทบาทเป็นเซิร์ฟเวอร์ และมีบุคคลที่อยู่ตรงกลางชื่อว่า Trudy โดยขั้นแรก Trudy จะแสดงบทบาทตัวเองเป็น Alice โดยบอก Bob ว่า "I'm Alice" และ Bob จะส่งข้อความว่า "I'm Bob" พร้อมส่งกุญแจสาธารณะของตนเองให้แก่ Trudy ซึ่งในขณะนี้ได้ปลอมตัวเป็น Alice เมื่อได้กุญแจสาธารณะของ Bob Trudy จะทำการสร้างกุญแจลับขึ้นมาแล้วใช้กุญแจสาธารณะของ Bob มาเข้ารหัสกุญแจลับที่ได้สร้างขึ้นมา เสร็จแล้วก็ทำการส่งข้อมูล (Cipher text)นั้นให้แก่ Bob ณ จุดนี้ Trudy สามารถที่จะติดต่อสื่อสารกับ Bob ได้ จากนั้น Trudy จะทำการปลอมตัวเองให้เป็น Bob ที่มีบทบาทเป็นเซิร์ฟเวอร์ โดย Trudy จะรับข้อความจาก Alice ว่า "I'm Bob" พร้อมส่งกุญแจสาธารณะของตนเองให้แก่ Alice ณ ตอนนี้ Trudy ก็สามารถติดต่อสื่อสารกับ Alice ได้ เมื่อถึงจุดนี้ Trudy จะทราบข้อมูลทุกอย่างที่ Bob กับ Alice ติดต่อกัน ดังนั้น Trudy สามารถสร้างความเสียหายให้เกิดแก่ Bob และ Alice ได้ ซึ่งเราเรียกว่าปัญหาที่เกิดขึ้นนี้ว่า การโจมตีแบบแมนอินเดอะมิดเดิล (Man-in-the-middle)



รูปที่ 5-4 แสดงการโจมตีแบบแมนอินเดอะมิดเดิล (Man-in-the-middle)

โพรโตคอลได้ทำการแก้ไขปัญหาดังกล่าวโดยได้เพิ่มวิธีการที่เรียกว่าการพิสูจน์ตัวตนของเซิร์ฟเวอร์จากรูปที่ 5-5 จะเห็นได้ว่าข้อความ Certificate ถูกนำมาแทนที่ข้อความ ServerKeyExchange



รูปที่ 5-5 แสดงสองข้อความใหม่ที่ใช้สำหรับการพิสูจน์ตนของเซิร์ฟเวอร์

1.5 ใบรับรองสิทธิ์ (Certificate)

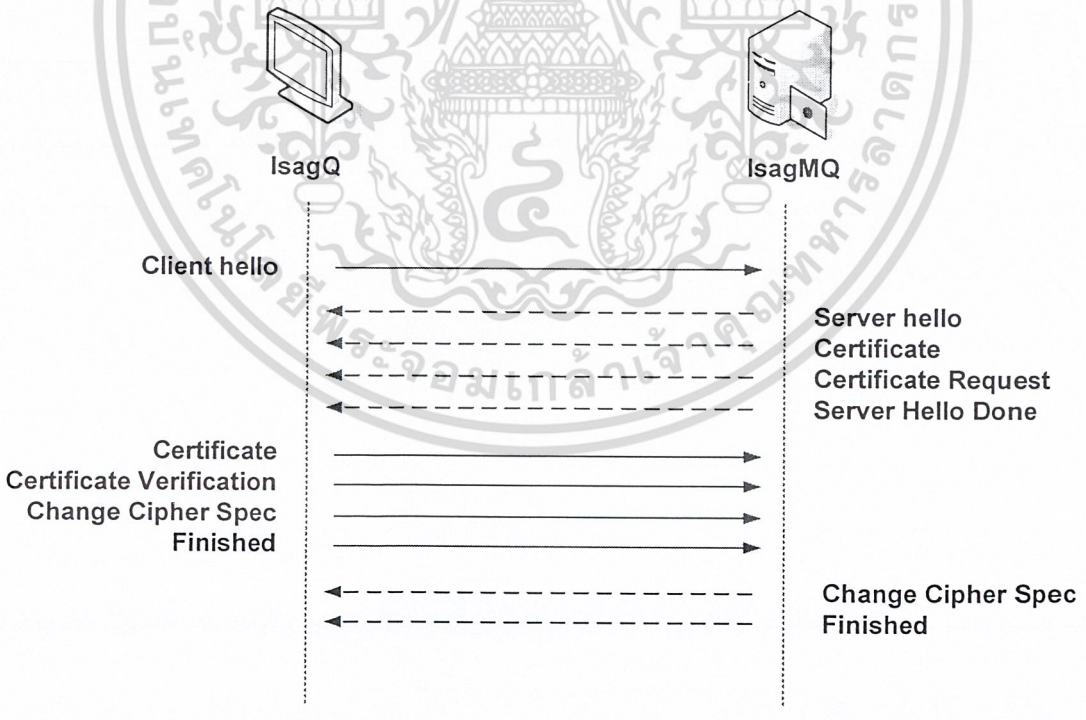
การที่ไคลเอ็นต์จะสามารถทำการระบุตัวตนของเซิร์ฟเวอร์ได้ เซิร์ฟเวอร์จะส่งข้อมูลต่างๆ ที่สามารถเป็นการระบุตัวตนได้เช่น ชื่อ,กุญแจสาธารณะของเซิร์ฟเวอร์ ไปให้แก่ CA Certificate Authority) ซึ่งCA จะแบ่งออกเป็นสองลักษณะคือ Internal CA และ External CA)ซึ่งจะได้กล่าวในหัวข้อที่ 3 Certificate Authority(CA)) และ 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 CAนั้น ส่งไปให้แก่ไคลเอ็นต์

1.6 ClientKeyExchange

เมื่อไคลเอ็นต์ได้รับข้อความ Certificate จากเซิร์ฟเวอร์ จะใช้ฟังชั่น Verify() ในการพิสูจน์ตัวตนของเซิร์ฟเวอร์ ถ้าพิสูจน์ตัวตนถูกต้อง ไคลเอ็นต์จะนำกุญแจสาธารณะของเซิร์ฟเวอร์ที่ได้ส่งมาด้วยนั้น นำมาเข้ารหัสกุญแจลับที่ได้สร้างขึ้นมาตามกระบวนการเดิม

1.7 การแยกระหว่างการพิสูจน์ตนกับการเข้ารหัสข้อมูล

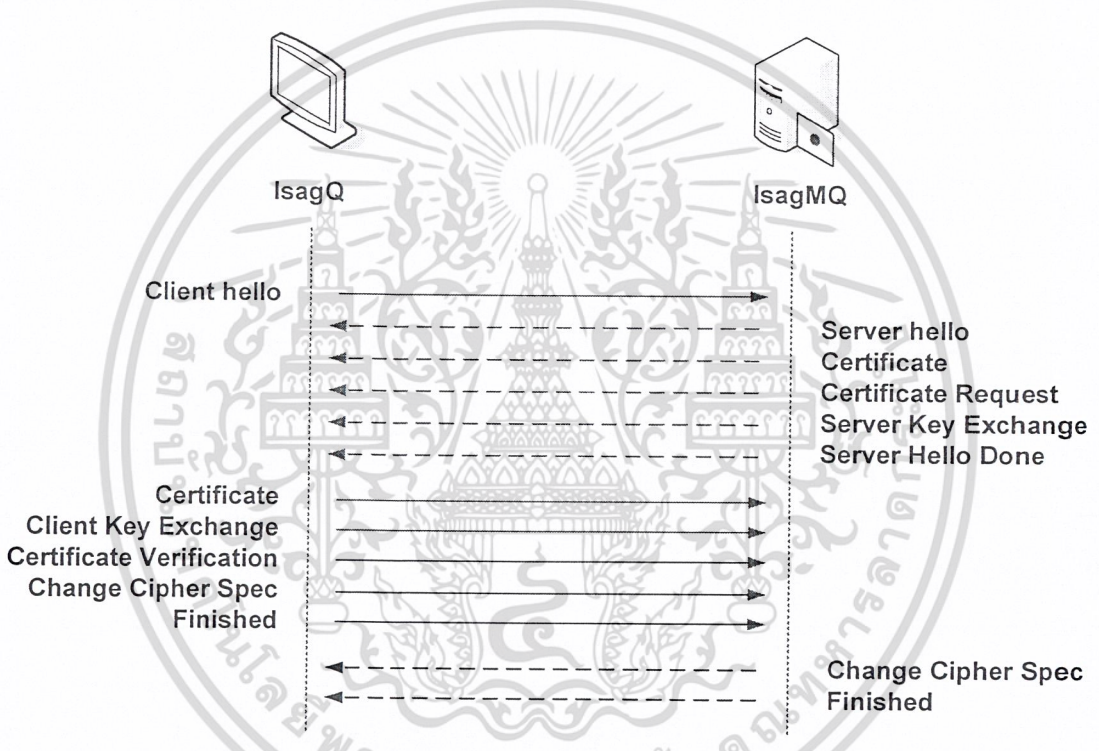
การใช้ข้อความ Certificate เพียงอย่างเดียวในการทำทั้งการพิสูจน์และการเข้ารหัสข้อมูลนั้นเป็นวิธีการที่ไม่ดีนัก เนื่องจากว่า ยกตัวอย่างเช่นมีหลาย ๆ อัลกอริทึมที่ใช้ในการสร้างกุญแจสาธารณะที่มีไว้ใช้สำหรับการ Sign ข้อมูลเท่านั้น ไม่สามารถนำมาใช้ในการเข้ารหัสข้อมูลได้ เช่น DSA (Digital Signature Algorithm) ดังนั้นจึงมีการแยกข้อความระหว่างพิสูจน์ตนกับข้อความสำหรับการเข้ารหัสออกจากกัน ดังรูปที่ 5-6 เซิร์ฟเวอร์จะให้ข้อความ Certificateสำหรับการพิสูจน์ตนของเซิร์ฟเวอร์ ส่งไปให้แก่ไคลเอ็นต์ก่อนต่อจากนั้นจึงทำการส่งข้อความ ServerKeyExchange ที่ใช้สำหรับเข้ารหัสข้อมูลให้แก่ไคลเอ็นต์ ในฝั่งของไคลเอ็นต์ถ้าการพิสูจน์ตัวตนของเซิร์ฟเวอร์ถูกต้อง ไคลเอ็นต์จะใช้กุญแจสาธารณะของเซิร์ฟเวอร์ทำการเข้ารหัสกุญแจลับที่ได้สร้างขึ้นตามกระบวนการของ SSL โพรโตคอล



รูปที่ 5-6 การใช้ระบบพิสูจน์ตนอย่างเดียว

1.8 การพิสูจน์ตัวตนของไคลเอ็นต์

เมื่อเซิร์ฟเวอร์มีความต้องการที่จะพิสูจน์ตัวตนของฝั่งไคลเอ็นต์ ไคลเอ็นต์จะต้องนำข้อมูลต่างๆ ของไคลเอ็นต์ไปให้แก่ CA ที่เซิร์ฟเวอร์เชื่อถือ ทำการ Sign ข้อมูลให้เพื่อที่ไคลเอ็นต์จะสามารถพิสูจน์ตัวตนได้ ดังนั้นไคลเอ็นต์จะต้องมีการสร้างกุญแจสาธารณะของตนเองขึ้นมาเพื่อใช้ในการพิสูจน์ตัวตนของไคลเอ็นต์ โดยกุญแจสาธารณะของไคลเอ็นต์นี้จะใช้ในการพิสูจน์ตัวตนเท่านั้นไม่ได้ใช้ในการเข้ารหัสข้อมูลใด ๆ จากรูปที่ 5.7 ข้อความที่เพิ่มขึ้นมาคือ



รูปที่ 5-7 แสดงการพิสูจน์ตนและเข้ารหัสของฝั่งไคลเอ็นต์

Certificate Request

จะเป็นข้อความจากเซิร์ฟเวอร์ที่เป็นการบอกให้แก่ไคลเอ็นต์ทราบว่าเซิร์ฟเวอร์มีความต้องการที่จะทำการพิสูจน์ตัวตนของไคลเอ็นต์ โดยจะเห็นได้ว่าข้อความนี้อยู่หลังข้อความ Certificate เนื่องจากจะต้องมีการพิสูจน์ตัวตนของเซิร์ฟเวอร์ให้ถูกต้องก่อน เมื่อการพิสูจน์ตัวตนของเซิร์ฟเวอร์ถูกต้องเซิร์ฟเวอร์จึงทำการรองขอการพิสูจน์ตัวตนของฝั่งไคลเอ็นต์

Certificate

เป็นข้อความของฝั่งไคลเอ็นต์ที่เป็นข้อมูลที่ไคลเอ็นต์ได้รับจากการส่งข้อมูลของไคลเอ็นต์ เช่น กุญแจสาธารณะของไคลเอ็นต์ ไปให้แก่ CA ที่เซิร์ฟเวอร์เชื่อถือทำการ Sign ข้อมูลดังกล่าว

Certificate Verify

เป็นข้อความที่คล้าย ๆ กับข้อความ Finished แต่สิ่งที่แตกต่างกันคือ ข้อมูลของคีย์ซึ่งถ้าเป็นข้อความ Finished จะเป็นข้อมูลของกุญแจลับ แต่ถ้าเป็นข้อความ CertificateVerify จะเป็นข้อมูลของกุญแจสาธารณะที่ไคลเอ็นต์ใช้ในการพิสูจน์ตัวตน

2 OpenSSL

2.1 OpenSSL คืออะไร

Open SSL คือโปรเจ็คของทีมงานหนึ่งที่พยายามในการพัฒนา OpenSSL ที่สามารถไปเรียกใช้การทำงานของ Secure Socket Leyer (SSL) และ Transport Layer Security (TLS) โพรโตคอลได้อย่างมีประสิทธิ์ภาพและทั้งนี้ Open Source นี้จะประกอบด้วยไลบรารีต่าง ๆ ที่เกี่ยวข้องกับการเข้ารหัสและถอดรหัสข้อมูลโดยการใช้อัลกอลิทึมต่าง ๆ ที่มีความปลอดภัยสูงเช่น RSA, DSA, 3DES ฯลฯ ไว้อย่างครบถ้วน ซึ่งข้อมูลและรายละเอียดต่าง ๆ หาได้ที่ www.openssl.org

2.2 การใช้งาน OpenSSL

ก่อนที่จะใช้งาน OpenSSL เราจะต้องทำการติดตั้ง OpenSSL เสียก่อนถึงจะมี ไลบารีและคำสั่งในการสร้างคีย์ให้ใช้งาน เริ่มต้นให้ดาวน์โหลด Source OpenSSL 0.9.7e (25-oct-2004 including important bugfixes) จาก www.openssl.org ที่เลือกเวอร์ชั่นนี้เพราะมีเสถียรภาพและปลอดภัยแล้ว วิธีการติดตั้งดูได้จาก Readme.txt ที่อยู่ในซอร์สหรือถ้าหากได้ทำการติดตั้ง Apache-ssl ไว้แล้วก็ไม่จำเป็นต้องติดตั้ง Source OpenSSL อีกเพราะมีไลบารีจาก Apache-ssl แล้วจากนั้นก็สามารถใช้คำสั่งและไลบารีต่าง ๆ ของ OpenSSL ได้ และที่สำคัญในขณะกำลังติดตั้งเราต้องสังเกตดูว่าเก็บเฮเดอร์ไฟล์และไลบารีเก็บอยู่ที่ไหนด้วยเพราะต้องใช้ตอนคอมไพล์โปรแกรม

2.2.1 ตำแหน่งเฮดเดอร์ไฟล์และไลบารี

การที่เราจะทำการคอมไพล์ โปรแกรมที่ Include เฮเดอร์ไพล์ของ Openssl เราต้องรู้จักตำแหน่งที่เก็บเฮเดอร์ไพล์และไลบารีเสียก่อน ถ้าเราทำการติดตั้ง OpenSSL แบบมาตรฐานแล้วเฮเดอร์ไฟล์และไลบารีของ OpenSSL ก็จะอยู่ที่เฮเดอร์ไฟล์และไลบารีมาตรฐานของ ลินุกซ์ คือ

-/usr/include/openssl	สำหรับเฮเดอร์ไพล์
-/usr/lib/	สำหรับไลบารี

-ไลบรารีที่จำเป็นคือ libssl.a และ lincrypto.a

ถ้าหากหาเฮเดอร์ไพล์และไลบารี ที่ตำแหน่งมาตรฐานไม่เจอหลังจากที่ได้ทำการติดตั้งไปแล้วให้ลองไปดูที่ /usr/local/ssl/ ซึ่งอาจจะเจอหรืออยู่ตามตำแหน่งที่กล่าวไว้ตอนติดตั้ง OpenSSL

2.2.2 การคอมไพล์

ถ้าหากเฮเดอร์ไพล์และไลบารีของ OpenSSL อยู่ตำแหน่งมาตรฐานของเฮเดอร์ไพล์และไลบารีของ ลินุกซ์ เราไม่ต้องอ้างถึง path เลย ทำการคอมไพล์และลิงก์ไลบารีได้ดังนี้

```
gcc -c TestOpenssl.c                                    /*Compile*/
gcc -o TestOpenssl TestOpenssl.o -lssl –lcrypto         /*Link Library*/
```

ถ้าหากเฮเดอร์ไพล์และไลบารีของ OpenSSL ไม่ได้อยู่ดังตำแหน่งมาตรฐานของเฮเดอร์ไฟล์และไลบารีของลินุกซ์ เราต้องอ้างถึง path ก่อนจะทำการคอมไพล์และลิ่งก์ไลบารี ได้ดังนี้

```
gcc -c TestOpenssl.c –L/~path header file~                          /*Compile*/
gcc -o TestOpenssl TestOpenssl.o  -L/~path library file -lssl –lcrypto /*Link Library*/
```

3. Certificate Authority(CA)

CA เป็นองค์กรหรือสมาคมที่จะออกใบรับรองสิทธิ์ให้แก่ผู้ขอ ใบรับรองสิทธิ์ก็คือ ใบที่ที่ใช้พิสูจน์ถึงตนเองเมื่อต้องการติดต่อกับผู้อื่น เหมือนการใช้บัตรประชาชน ต่างกันตรงที่ใบรับรองสิทธิ์นี้เอาไว้รับรองสิทธิ์ในการสื่อสารบนระบบ ผู้ขอใบรับรองสิทธิ์จากCA จะต้องมั่นใจในใบรับรองสิทธิ์นั้นแต่ไม่ได้หมายความว่าCAนั้นจะไม่มีความผิดพลาดเลย ตัวอย่างองค์กรที่เป็นCA เช่น Verisign รายละเอียดต่างๆ หาได้ที่ www.verisign.com

ประเภทของ CA มี 2 ชนิด

- **Public CAs หรือ External CAs**

  เป็นองค์กรที่ออกใบรับรองสิทธิ์ให้แก่บุคคลหรือองค์กรโดยทั่วไปเช่น Verisign จะออกใบรับรองสิทธิ์ให้กับเว็บไซต์ (web sites) หรือแอพพลิเคชั่นที่ต้องการให้มีการเข้ารหัสและการรับรองสิทธิ์ของผู้ใช้ในการทำกิจกรรมต่าง ๆ เช่น อีคอมเมิร์ซ (e-commerce) จะต้องมีระบบการรับส่งข้อมูลที่ปลอดภัยเช่น รหัสผ่านของเครดิตการ์ดของลูกค้า ซึ่งโดยมากแล้วถ้าเป็น External CAs ถ้าเราไปขอใบรับรองสิทธิ์เราจะไม่ได้มาฟรี ๆ จะต้องมีการจ่ายเงินให้แก่ External CAs นั้นๆ ก่อน

- **Private CAs หรือ Internal CAs**

  เป็นองค์กรที่ออกใบรับรองสิทธิ์ให้แก่คนที่อยู่ภายในองค์กรเพื่อใช้งานภายในองค์กรเอง องค์กรภายนอกไม่สามารถใช้ CA และจะไม่สามารถไว้วางใจการออก CA แบบนี้ได้แต่ขึ้นอยู่กับ Internal CAs นั้น ๆ ว่ามีกฎระเบียบเป็นอย่างไร ซึ่ง CA ประเภทนี้จะออกใบรับรองสิทธ์ให้ฟรีเพราะถือว่าบุคคลเหล่านั้นอยู่ภายในองค์กร

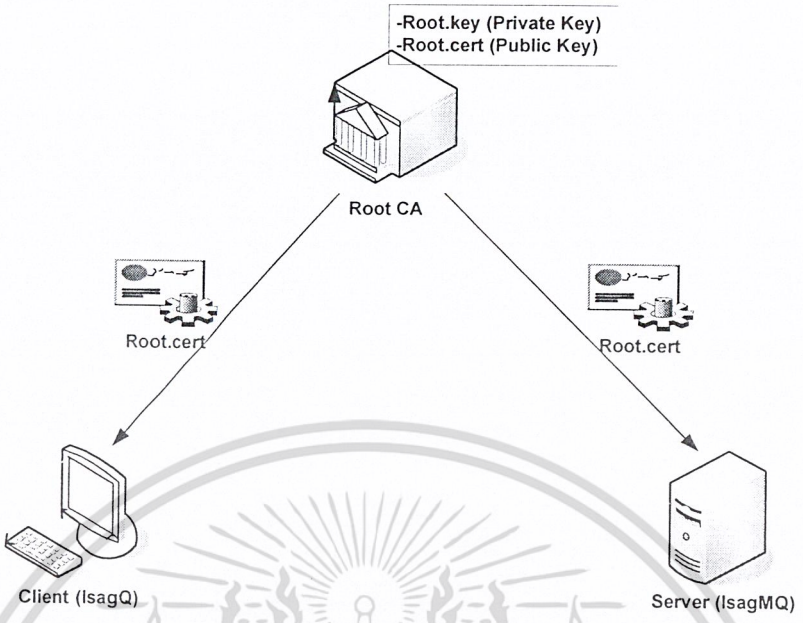
การทำโครงการนี้ได้ทำระบบ CA แบบ Private CA เพราะเห็นว่าเหมาะสมที่จะออกใบรับรองสิทธิ์ให้แก่ผู้ใช้ของ IsagMQ เท่านั้น เพื่อการสื่อสารอย่างปลอดภัยภายในกลุ่ม ในที่นี้จะอธิบายการสร้างและการออกแบบ Private CA เท่านั้นซึ่งจะอยู่ในส่วนการออกแบบและพัฒนา

3.1 Third Party

- Root CA เป็นผู้ที่จะออกใบรับรองสิทธิ์ให้แก่ผู้ร้องขอ
- Client เป็นผู้ขอใบรับรองสิทธิ์เพื่อใช้ในการพิสูจน์ตน
- Server เป็นตรวจสอบใบรับรองสิทธิ์เพื่อใช้ในการพิสูจน์ตน

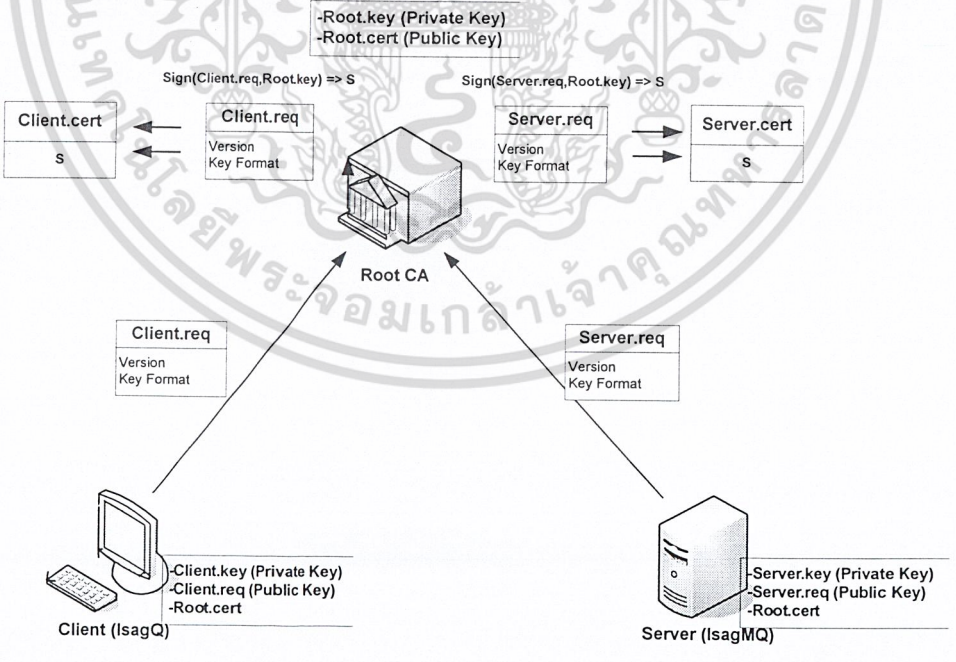
3.2 ขั้นตอนในการร้องขอใบรับรองสิทธิ์ จาก Public CA

1. ผู้ร้องขอซึ่งในที่นี้จะประกอบด้วย Client และ Server จำเป็นต้องมี ใบรับรองสิทธิ์ของ Root เพื่อเก็บไว้ใช้ในระบบพิสูจน์ตน ดังรูป 5-8



รูปที่ 5-8 ผู้ร้องขอจำเป็นต้องมี ใบรับรองสิทธิ์ของผู้ออก

2. เมื่อผู้ร้องขอมีใบรับรองสิทธิ์ของ Root CA แล้ว ก็ให้ทำการร้องขอไปยัง Root CA ดังรูป



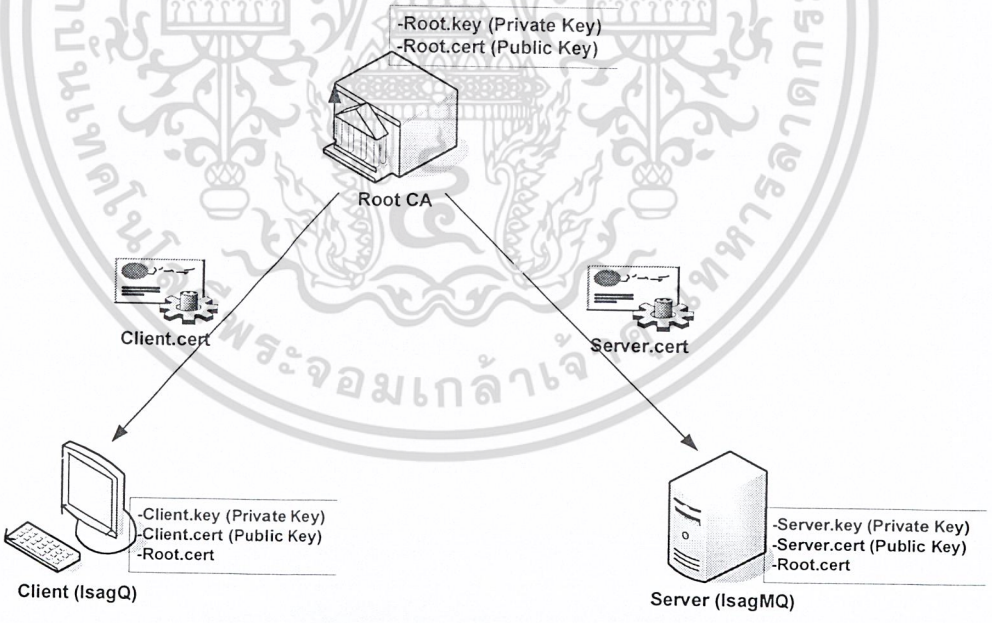
รูปที่ 5-9 ผู้ร้องขอส่งคำร้องไปยังผู้ออกเพื่อให้ผู้ออกทำการ sign ใบรับรองสิทธิ์

จากรูป 5-8 จะเห็นได้ว่า ก่อนที่ ผู้ร้องขอจะส่ง Client.req หรือ Server.req ออกไป ผู้ร้องขอจำเป็นต้องมี ไฟล์ 2 ชนิดก่อน คือ Client.key หรือ Server.key ซึ่งคือกุญแจส่วนตัวของผู้ร้องขอ (Private Key) และ Client.req หรือ Server.req ซึ่งคือไฟล์ที่ใช้ร้องขอไปยังผู้ออกใบรับรองสิทธิ์ให้

Client.req หรือ Server.req ที่ส่งไปจำเป็นต้องถูกเข้ารหัส ด้วย Root.cert ก่อนเพื่อที่จะมั่นใจได้ว่า Root CA จะสามารถเปิดดูได้เพียงผู้เดียว

หลังจากที่ Root CA ได้รับ Client.req หรือ Server.req แล้ว ก็จะทำการแฮช (hash) ซึ่งจะได้ค่ามาค่าหนึ่ง แล้วใช้ฟังก์ชั่น sign ด้วย Root.key กับ ค่าแฮช ที่ได้ ผลลัพธ์จะได้ค่า S (Digital Signature) ออกมา แล้วนำข้อมูลจาก Client.req หรือ Server.req และค่า S ที่ได้ไปเก็บไว้ใน Client.cert หรือ Server.cert

3. ผู้ออกใบรับรองสิทธ์จะทำการส่ง Client.cert หรือ Server.cert กลับมายัง ผู้ร้องขอ โดยไฟล์ดังกล่าวผู้ร้องขอเท่านั้นที่จะสามารถเปิดดูได้เช่นกัน ดังรูป 5-10

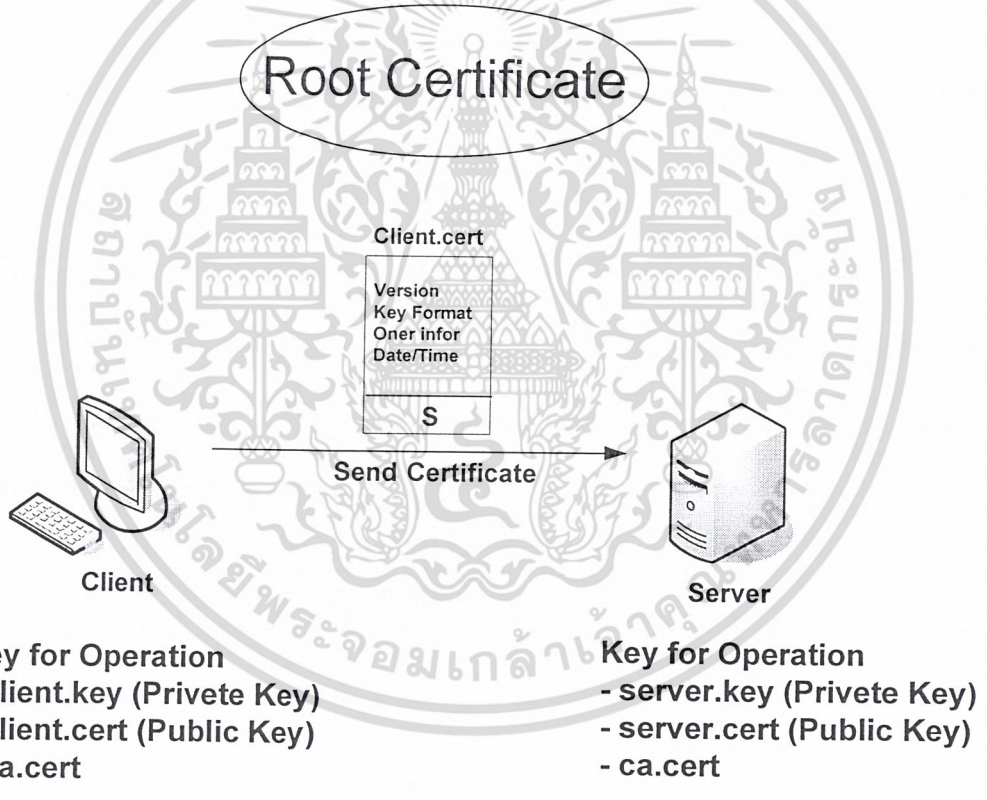


รูปที่ 5-10 ผู้ร้องขอได้รับ ใบรับรองสิทธิ์

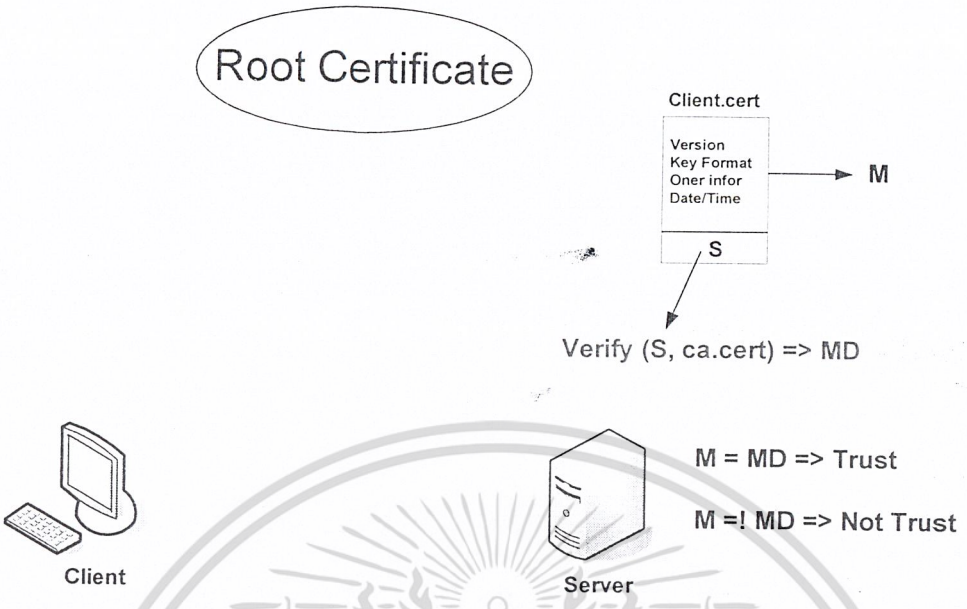
3.3 การสื่อสารกันโดยมีใบรับรองสิทธิ์ (Certificate Authentication)

จากรูป 5-11 เมื่อเซิร์ฟเวอร์รองขอการพิสูจน์ตนของฟังไคลเอ็นต์ จากนั้นไคลเอ็นต์จะส่งข้อความ Certificate (client.cert) ให้แก่เซิร์ฟเวอร์ทำการพิสูจน์ตนของฝั่งไคลเอ็นต์

จากรูปที่ 5-12 เมื่อเซิร์ฟเวอร์ได้รับไฟล์ client.cret ซึ่งไฟล์นี้ประกอบไปด้วยสองส่วนคือ ส่วนข้อมูลส่วนตัวของไคลเอ็นต์และค่า S (Digital Signature) กระบวนการพิสูจน์ตนทางฝั่งเซิร์ฟเวอร์คือ จะนำเอาข้อมูลส่วนตัวของไคลเอ็นต์มาทำ MD5 ได้ค่าแฮช (hash) มาหนึ่งค่าคือค่า M จากนั้นนำค่า S ของไคลเอ็นต์และกุญแจสาธารณะของ Root Certificate มาเข้าฟังก์ชั่น Verify ค่าที่ได้ออกมาคือ MD นำค่า M กับ MD มาเปรียบเทียบกัน ถ้าค่านั้นเท่ากันแสดงว่าการพิสูจน์ตัวตนนั้นถูกต้อง ถ้าไม่เท่ากันแสดงว่าการพิสูจน์ตนผิดพลาด



รูปที่ 5-11 แสดงการสื่อสารโดยใช้ใบรับรองสิทธิ์



รูปที่ 5-12 แสดงการพิสูจน์ใบรับรองสิทธิ์โดยฝั่งเซิร์ฟเวอร์

บทที่ 6
ภาษาจาวา กับ การประยุกต์ใช้งาน SSL
JSSE for Secure Socket

ในปัจจุบัน ความต้องการความปลอดภัยในการติดต่อสื่อสารบนอินเตอร์เน็ตเป็นเรื่องที่จำเป็นอย่างยิ่ง หลายๆ องค์กร หรือ บริษัท ต่างต้องการความเชื่อมั่นในความปลอดภัยสำหรับการทำหลากหลายธุรกรรม โดยมาตรฐานหนึ่งที่ทุกคนยอมรับโดยทั่วกันก็คือ การประยุกต์ใช้งานช่องทางการสื่อสารแบบปลอดภัย (Secure Socket) หรือที่รู้จักกันดีในนาม SSL Protocol

ในฐานะที่ภาษาจาวา เป็นหนึ่งในหลายภาษาสำหรับการเขียนโปรแกรมที่ทั่วโลกต่างยอมรับ จึงได้พัฒนาแพ็กเกจหนึ่งขึ้นมาเพื่อรองรับการทำงานดังกล่าวในชื่อ JSSE (J2SE) ซึ่งเป็น แพ็กเกจเพิ่มเติมจากที่มีอยู่ มาตรฐานนี้สามารถทำงานเข้ากันได้กับ OPENSSL ได้อีกด้วย โดยทางผู้พัฒนาได้อาศัยแพ็กเกจดังกล่าวร่วมพัฒนา โปรแกรมฝั่งลูกข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย (IsagQ) ในครั้งนี้ด้วย รายละเอียดต่อไปจะกล่าวถึง ส่วนสำคัญหลักที่ใช้ในการพัฒนาครั้งนี้

1. แพ็กเกจหลักที่สำคัญในการใช้ SSL

1.1 import javax.net.ssl.*;

1.2 import java.security.*;

1.3 import javax.security.cert.X509Certificate;

โดย 1.1 จะมีความสำคัญสำหรับการสร้างช่องทางการสื่อสารอย่างปลอดภัยโดยใช้ SSL

สำหรับ 1.2 จะมีความสำคัญเสริมในเรื่องเกี่ยวกับความปลอดภัยเช่นการสร้างคีย์ การทำเมสเสจไดเจส เป็นต้น

สำหรับ 1.3 จะจัดการเกี่ยวกับเรื่องใบรับรองสิทธิ์เพื่อให้ได้มาซึ่งข้อมูลที่จำเป็น

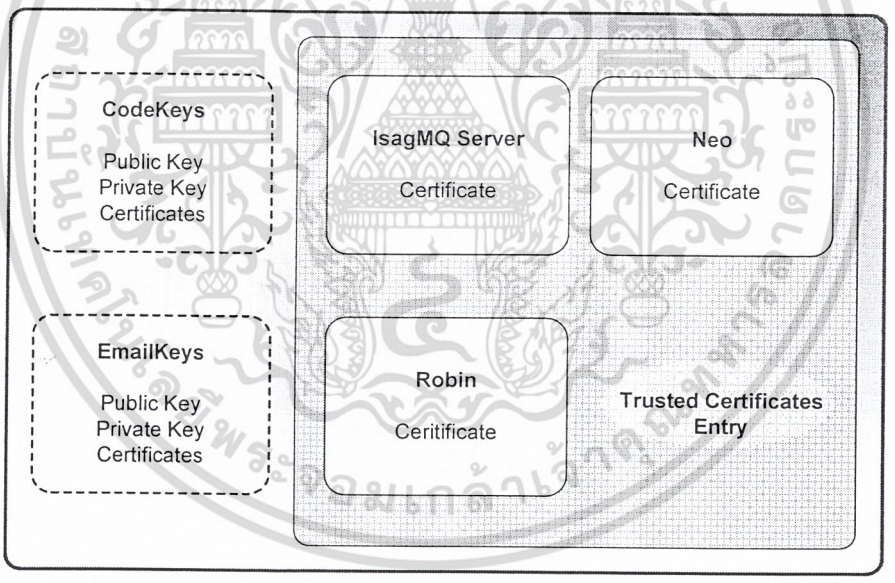
2. คลาสที่สำคัญสำหรับการประยุกต์ใช้ SSL กับ ใบรับรองสิทธ์

Class KeyStore เป็นคลาสที่ไว้ใช้จัดการเกี่ยวกับใบรับรองสิทธิ์ โดยภายในจะประกอบด้วย 2 ส่วนหลัก คือ

- กุญแจลับ **(Private Key)** และ ลำดับขั้นที่มาของใบรับรองสิทธิ์ (Chain of Certificates) ที่สัมพันธ์กันกับกุญแจสาธารณะของผู้ใช้ ซึ่งส่วนนี้ใช้ทำการเซ็นรับรองข้อมูลเพื่อแสดงถึงความเป็นเจ้าของในข้อมูลดังกล่าวจริง

- ใบรับรองสิทธิ์ **(Certificate)** ซึ่งประกอบไปด้วยใบรับรองสิทธิ์ต่างๆของผู้อื่นที่เราไว้วางใจ ซึ่งส่วนนี้อาจจะเรียกได้อีกอย่างว่า Trusted Certificate Entry ความสำคัญของส่วนนี้ก็คือ หากใบรับรองสิทธิ์ไม่ได้ถูกออกโดยรายนามของใบรับรองสิทธิ์ที่ไว้วางใจแล้วการสื่อสารก็จะไม่เกิดขึ้นอย่างแน่นอน

ข้อมูลต่างใน KeyStore จะถูกเก็บไว้เป็นกลุ่มๆ โดย alias ซึ่งเสมือนเป็นชื่อสั้นๆ ที่ใช้บ่งบอกกลุ่มข้อมูลความสัมพันธ์นั้นๆ ดังรูป 6-1



รูป 6-1 แสดงถึงความสำคัญของ alias

จากรูป 6-1 จะเห็นว่า มี คู่ของกุญแจ อยู่ 2 ชุด โดยชุดแรกใช้สำหรับเซ็นข้อมูลทั่วไป ส่วนชุดหลังสำหรับการเซ็นเฉพาะข้อมูลที่เป็นอีเมล และ มี ใบรับรองสิทธิ์ที่ไว้วางใจอยู่ 3 ชุด จะเห็นได้ว่าแต่ละชุดจะมีชื่อเฉพาะซึ่งนั้นก็คือ alias

3. Keytool

เป็นโปรแกรมที่ใช้งานผ่านคอมมานด์ไลน์ ซึ่งทำงานร่วมกันกับ คลาส java.security.KeyStore ในความเป็นจริง KeyStore ก็คือฐานข้อมูลของกลุ่ม กุญแจลับ , กุญแจสาธารณะ และ ใบรับรองสิทธิ์ที่ไว้วางใจ โดยมี alias เป็นตัวระบุกลุ่มข้อมูลเหล่านั้น

โดยปกติ KeyStore จะถูกเก็บไว้ใน ฮาร์ดดิสก์ โดย หนึ่ง KeyStore ต่อ หนึ่ง ไฟล์ โปรแกรม keytool จะช่วยจัดการในการสร้าง KeyStore ขึ้นมาได้หลายๆ KeyStore ตามต้องการโดย ไฟล์ KeyStore จะถูกเก็บไว้ใน โฮมไดเร็กทอรี (Home Directory) ที่เรียกโปรแกรมนี้ขึ้นมา

การสร้างคู่กุญแจโดยใช้ keytool

ตัวอย่าง

C:\ keytool –genkey –alias Jonathan –keyalg RSA –keysize 1024 –dname "CN=Jonathan Knudsen, OU=Technical Publications, O=Ladkrabang, C=US" –keypass buendia –storepass buendia –keystore mykeystore

รายละเอียดมีดังนี้

- genkey
  - เป็นตัวเลือกสำหรับบอก keytool ว่าต้องการจะสร้างคู่กุญแจคู่ใหม่ขึ้นมา โดยกุญแจลับจะถูกเก็บไว้ใน KeyStore ส่วน กุญแจสาธารณะจะถูกแนบไว้ในใบรับรองสิทธิ์
- alias
  - ชื่อที่บ่งบอกกลุ่มใน KeyStore
- keyalg
  - ตัวเลือกที่ใช้บอก keytool ว่าจะใช้ อัลกอริทึมใดสำหรับสร้างกุญแจ
- keysize
  - ระบุขนาดของกุญแจที่จะสร้าง
- dname
  - ย่อมาจาก Distinguished name (DN) ซึ่งประกอบไปด้วยข้อมูลเช่น ชื่อ สถานที่ ประเทศ องค์กร โดยมีรายละเอียดดังนี้
    - CN (Common name) – ชื่อ
    - OU (Oraganizational unit) – ส่วนย่อยของกลุ่มองค์กร

- O (Organization) – องค์กร
- L (Locality) – เขต แขวง อำเภอ เป็นต้น
- S (State) – จังหวัด
- C (Country) – ประเทศ

- keypass
  - passphrase ที่ใช้การเข้าถึงกุญแจลับ
- storepass
  - passphrase ที่ใช้ป้องกัน ความสมบูรณ์ (Integrity) ของ KeyStore ไฟล์
- keystore
  - ชื่อของ KeyStore ไฟล์

นอกจากนี้แล้วยังมี ตัวเลือกอื่นที่สามารถระบุลงไปได้ เช่น

- sigalg
  - เป็นตัวเลือกที่ใช้ในการสร้างใบรับรองสิทธิ์ที่เซ็นรับรองด้วยตัวเอง (self-signed certificate)
- validity
  - กำหนดอายุของใบรับรองสิทธิ์
- v
  - ใช้บอก keytool ให้แสดงผลการกระทำต่างๆออกมาทางหน้าจอ

รูป 6-2 ตารางแสดงค่าตั้งต้นของโปรแกรม keytool

ตารางแสดงค่าตั้งต้นของตัวเลือกของ keytool	
Option	**Default Value**
-alias	mykey
-keyalg	DSA
-keysize	1024
-sigalg	DSA/SHA-1
-validity	90
-keystore	Defautl.keystore file
-file	Standard input or output

การสร้าง KeyStore

ผู้ร้องขอจำเป็นต้องสร้าง KeyStore สำหรับใช้เก็บ กุญแจลับ , กุญแจสาธารณะ และ ใบรับรองสิทธิ์ที่ไว้วางใจ โดยมีวิธีการดังรูป 6-3

```
C:\test\B>keytool -genkey -alias 17 -keyalg RSA -keystore isagq
Enter keystore password:  secure!
What is your first and last name?
  [Unknown]:  17
What is the name of your organizational unit?
  [Unknown]:  CE
What is the name of your organization?
  [Unknown]:  COM
What is the name of your City or Locality?
  [Unknown]:  Ladkrabang
What is the name of your State or Province?
  [Unknown]:  Bangkok
What is the two-letter country code for this unit?
  [Unknown]:  TH
Is CN=17, OU=CE, O=COM, L=Ladkrabang, ST=Bangkok, C=TH correct?
  [no]:  yes

Enter key password for <17>
        <RETURN if same as keystore password>:  secure!
```

รูป 6-3 การสร้าง KeyStore ด้วย keytool

การตรวจสอบ KeyStore

ถ้าต้องการดูรายละเอียดภายใน KeyStore ให้ใช้คำสั่ง ดังรูป 6-4

```
C:\test\B>keytool -list -alias 17 -keystore isagg
Enter keystore password:  secure!
17, 30 ต.ค. 2548, keyEntry,
Certificate fingerprint (MD5): 9B:50:C4:D4:EC:47:E9:BD:3E:C2:97:E9:CE:67:1F:97
```

รูป 6-4 การตรวจสอบข้อมูลภายใน KeyStore ที่ถูกสร้างขึ้นด้วย keytool

ถ้าต้องการดูรายละเอียดทั้งหมดภายใน KeyStore ให้เพิ่ม –v เข้าไป ดังรูป 6-5

```
C:\test\B>keytool -list -alias 17 -v -keystore isagg
Enter keystore password:  secure!
Alias name: 17
Creation date: 30 ต.ค. 2548
Entry type: keyEntry
Certificate chain length: 2
Certificate[1]:
Owner: CN=17, OU=COM, O=CE, L=Ladkrabang, ST=Bangkok, C=TH
Issuer: CN=Root CA, O=ce.kmitl.ac.th, L=Ladkrabang, ST=Bangkok, C=TH
Serial number: 5
Valid from: Sun Jan 30 03:51:29 ICT 2005 until: Mon Jan 30 03:51:29 ICT 2006
Certificate fingerprints:
         MD5:  9B:50:C4:D4:EC:47:E9:BD:3E:C2:97:E9:CE:67:1F:97
         SHA1: 08:3A:56:1A:E3:76:07:DD:69:BD:65:9B:69:C3:F4:7B:22:4C:BD:FC
Certificate[2]:
Owner: CN=Root CA, O=ce.kmitl.ac.th, L=Ladkrabang, ST=Bangkok, C=TH
Issuer: CN=Root CA, O=ce.kmitl.ac.th, L=Ladkrabang, ST=Bangkok, C=TH
Serial number: 0
Valid from: Wed Oct 13 21:09:11 ICT 2004 until: Tue Oct 04 21:09:11 ICT 2005
Certificate fingerprints:
         MD5:  C2:81:51:63:1A:7B:E6:AF:2F:9F:9F:5B:5B:7B:61:48
         SHA1: ED:F8:7F:A4:01:1C:47:91:60:FE:6B:8B:74:BF:6E:EA:9A:9F:B0:5A
```

รูป 6-5 การตรวจสอบข้อมูลอย่างละเอียดภายใน KeyStore ที่ถูกสร้างขึ้นด้วย keytool

การนำ ใบรับรองสิทธิ์ ออกมา (Export Certificate)

โดยความเป็นจริงใบรับรองสิทธิ์ที่ดีควรจะมีบุคคลที่สามเป็นผู้รับรองให้ ซึ่งในที่นี้ก็คือ CA ดังนั้นจึงจำเป็นที่ต้องแปลงข้อมูลภายใน KeyStore ออกมาเป็นไฟล์ เพื่อนำไฟล์ดังกล่าวส่งไปยัง CA เซ็นรับรอง คำสั่งของ keytool ที่ใช้ในการทำงานดังกล่าวเป็นดังรูป 6-6

```
C:\test\B>keytool -export -alias 17 -file 17.req -keystore isagg
Enter keystore password:  secure!
Certificate stored in file <17.req>
```

รูป 6-6 การสร้างใบร้องขอใบรับรองสิทธิ์จาก KeyStore ที่ถูกสร้างขึ้นด้วย keytool

ทำการตรวจสอบ ไฟล์ที่ถูกนำออกมา โดยใช้คำสั่งดังรูป 6-7

```
C:\test\B>keytool -printcert -file 17.req
Owner: CN=17, OU=COM, O=CE, L=Ladkrabang, ST=Bangkok, C=TH
Issuer: CN=Root CA, O=ce.kmitl.ac.th, L=Ladkrabang, ST=Bangkok, C=TH
Serial number: 5
Valid from: Sun Jan 30 03:51:29 ICT 2005 until: Mon Jan 30 03:51:29 ICT 2006
Certificate fingerprints:
         MD5:  9B:50:C4:D4:EC:47:E9:BD:3E:C2:97:E9:CE:67:1F:97
         SHA1: 08:3A:56:1A:E3:76:07:DD:69:BD:65:9B:69:C3:F4:7B:22:4C:BD:FC
```

รูป 6-7 การตรวจสอบใบร้องขอใบรับรองสิทธิ์ที่สร้างด้วย keytool

การนำ ใบรับรองสิทธิ์ เข้ามา **(Import Certificate)**

โดยปกติ ในการเก็บใบรับรองสิทธิ์ที่ไว้วางใจ หรือแม้กระทั้ง ใบรับรองสิทธิ์ของตนเองที่ได้รับการเซ็นโดย CA จำเป็นที่จะต้องถูกเก็บเข้า KeyStore เพื่อแปลงไฟล์เหล่านั้นให้อยู่ในรูปแบบที่ KeyStore ต้องการ และ การนำไปใช้ รายละเอียดมีดังนี้

การนำเข้าใบรับรองสิทธิ์ที่ไว้วางใจ

อันดับแรก จำเป็นต้องมีใบรับรองสิทธิ์ที่ผู้ใช้เชื่อถืออยู่ โดยการนำใบรับรองสิทธิ์ดังกล่าวเข้ามานั้นมีวิธีดังรูป 6-8

```
C:\test\B>keytool -import -alias RootCA -keystore isagg -file root.der
Enter keystore password:  secure!
Owner: CN=Root CA, O=ce.kmitl.ac.th, L=Ladkrabang, ST=Bangkok, C=TH
Issuer: CN=Root CA, O=ce.kmitl.ac.th, L=Ladkrabang, ST=Bangkok, C=TH
Serial number: 0
Valid from: Wed Oct 13 21:09:11 ICT 2004 until: Tue Oct 04 21:09:11 ICT 2005
Certificate fingerprints:
         MD5:  C2:81:51:63:1A:7B:E6:AF:2F:9F:9F:5B:5B:7B:61:48
         SHA1: ED:F8:7F:A4:01:1C:47:91:60:FE:6B:8B:74:BF:6E:EA:9A:9F:B0:5A
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

รูป 6-8 การนำเข้าใบรับรองสิทธิ์ที่ไว้วางใจเข้ามาใน KeyStore

จากรูป เราสามารถนำเข้าใบรับรองสิทธิ์ที่ไว้วางใจได้อีก โดยทำตามขั้นตอนข้างบน

การนำเข้าใบรับรองสิทธิ์ของผู้ใช้

หลังจากผ่านขั้นตอนที่ผ่านมา ผู้ใช้จึงสามารถนำใบรับรองสิทธิ์ของผู้ใช้เข้ามาเก็บไว้ใน KeyStore โดยโปรแกรมจะทำการตรวจสอบความสัมพันธ์ระหว่าง กุญแจลับ และ ใบรับรองสิทธิ์ที่ไว้วางใจภายใน KeyStore นั้นเอง ดังรูป 6-9

```
C:\test\B>keytool -import -alias 17 -keystore isagq -file 17.der
Enter keystore password:  secure1
Certificate reply was installed in keystore
```

รูป 6-9 การนำเข้าใบรับรองสิทธิ์ที่ไว้วางใจเข้ามาใน KeyStore

บทที่ 7

การออกแบบ และ พัฒนาโปรแกรม

ส่วนที่ 1

ความแตกต่างระหว่าง

IsagMQ & IsagQ ปี 2546

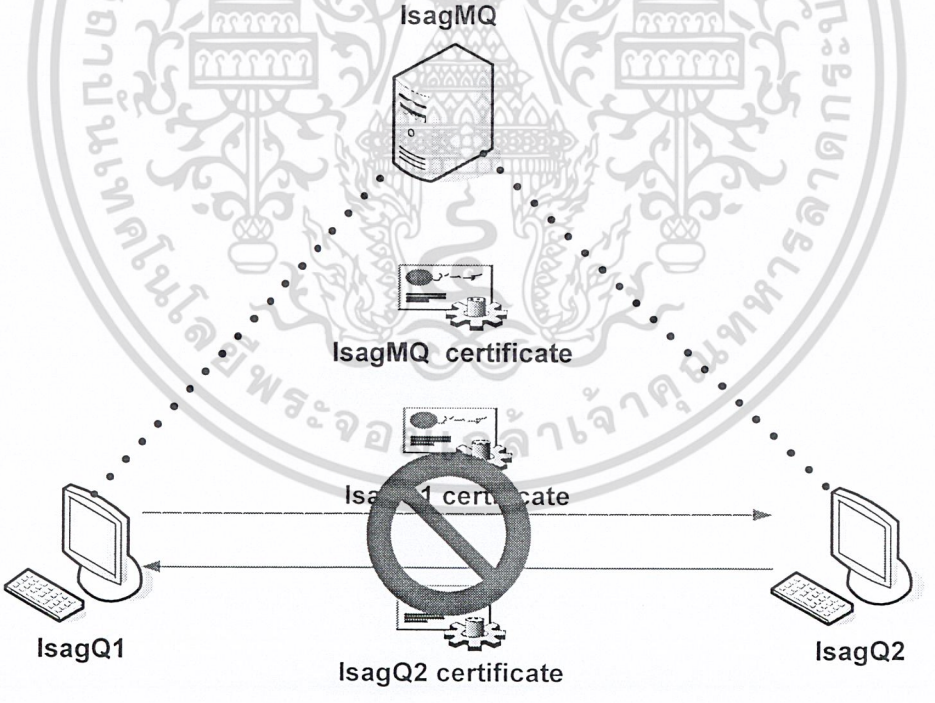
กับ

IsagMQ & IsagQ ปี 2546

ความแตกต่างระหว่าง IsagMQ & IsagQ ปีการศึกษา 2546 กับ 2547

ข้อเสียของ IsagMQ & IsagQ ในยุคก่อน

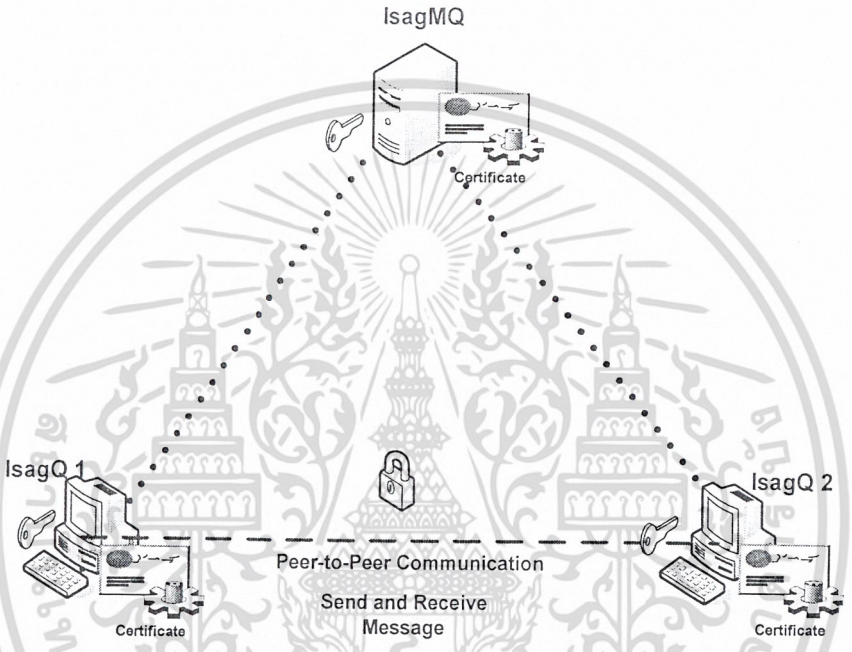
1. เนื่องจากการพัฒนา IsagMQ จะถูกพัฒนาบนระบบปฏิบัติการตระกูล Unix โดยใช้ภาษาซีในการเขียนโปรแกรม ในขณะที่ IsagQ จะถูกพัฒนาบนระบบปฏิบัติการตระกูล Windows โดยใช้ภาษา JAVA ในการเขียนโปรแกรม ผลลัพธ์ที่เกิดคือ ไม่สามารถที่จะใช้ IsagQ ติดต่อข้าม Platform ไปยัง IsagMQ ได้ สืบเนื่องจาก ปัญหาความแตกต่างในการใช้ Library SSL ของแต่ละภาษา
2. การติดต่อระหว่าง IsagQ กับ IsagQ เดิมจะใช้วิธี Diffie-Hellman ในการทำการแลกเปลี่ยนกุญแจสาธารณะ ซึ่งกลวิธีดังกล่าวมีข้อบกพร่องในปัญหาความปลอดภัยเป็นอย่างมากในที่นี้จะยกความร้ายแรงของปัญหา 2 ประการ
   2.1 ปัญหา Man in the middle ซึ่งได้กล่าวไว้ในบทที่ 5
   2.2 ปัญหาในการพิสูจน์ตน



รูปที่ 7-1 การติดต่อระหว่าง IsagQ ด้วยกันในยุคก่อน

จากรูป 7-1 จะเห็นได้ว่า การติดต่อระหว่าง IsagQ1 กับ IsagQ2 ไม่ได้มีการแลกเปลี่ยนใบรับรองสิทธิ์ซึ่งกันและกัน ผลเสียก็คือ ต่างฝ่ายไม่สามารถมั่นใจได้ว่ากำลังติดต่ออยู่กับบุคคลที่ต้องการสนทนาจริงๆ (เส้นไข่ปลา แสดงถึงไม่สามารถติดต่อไปยัง IsagMQ ได้)

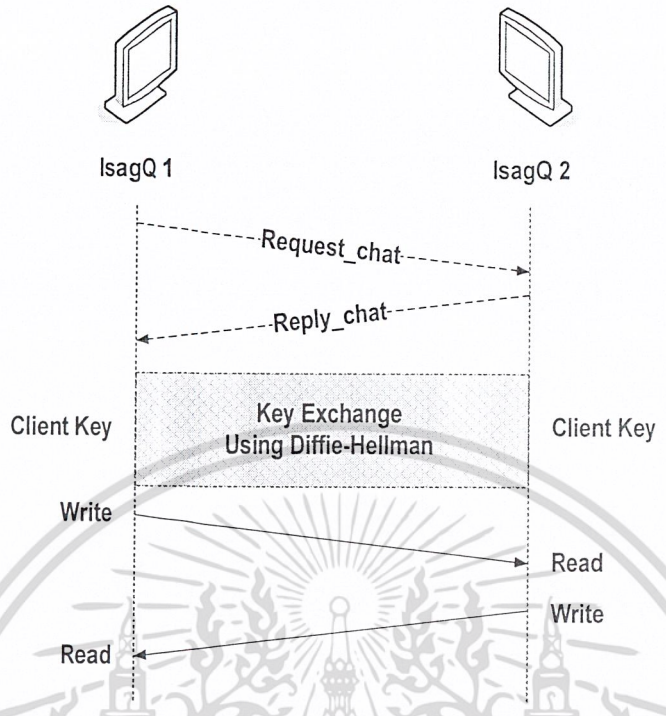
การติดต่อระหว่าง IsagQ กับ IsagQ โดยมิได้ใช้ SSL Protocol



รูปที่ 7-2 การติดต่อระหว่าง IsagQ ด้วยกัน โดยมิได้ใช้ SSL Protocol

จากรูปแสดง 7-2 ให้เห็นว่า การติดต่อระหว่าง IsagQ1 กับ IsagQ2 ซึ่งเป็นลักษณะ Peer-to-Peer Communication โดยเส้นประ คือ การติดต่อที่มิได้ใช้ SSL Protocol ทำให้ช่องทางสื่อสารดังกล่าวยังไม่ปลอดภัยเพียงพอ และ เส้นไข่ปลา แสดงถึงไม่สามารถติดต่อไปยัง IsagMQ ได้

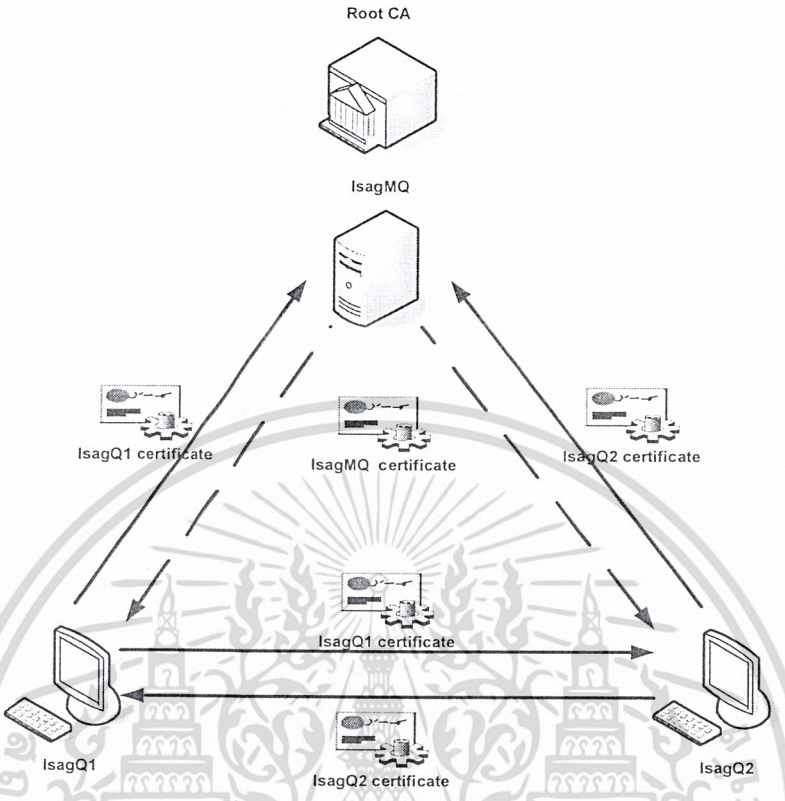
จากรูป 7-3 แสดงให้เห็นว่า มีการแลกเปลี่ยนกุญแจสาธารณะระหว่างกันเท่านั้น โดยใช้กระบวนการของ Diffie-Hellman ซึ่งก็ยังไม่ปลอดภัยต่อปัญหา Man-in-the-Middle นั้นเอง



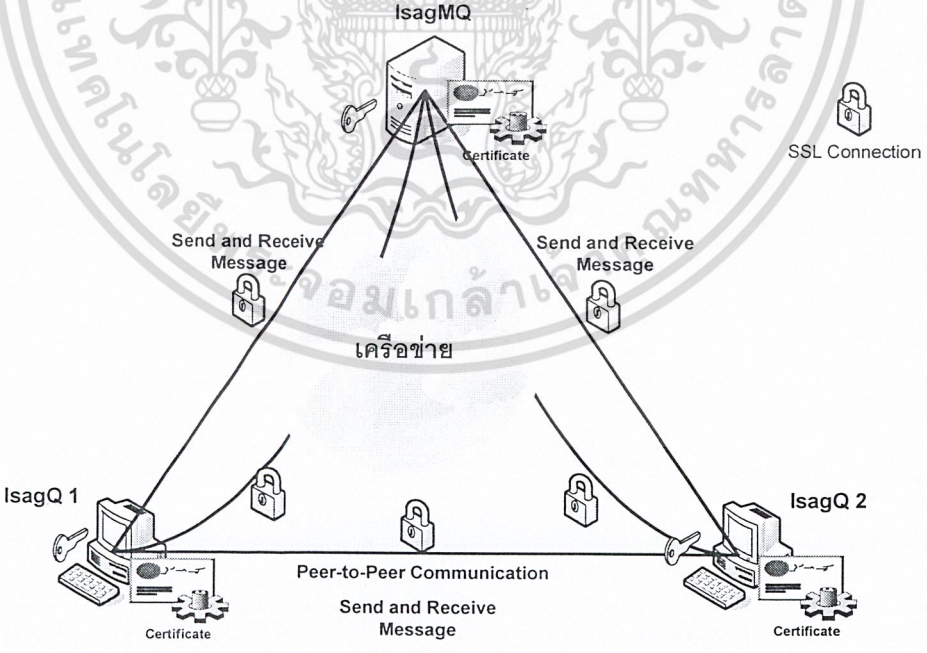
รูปที่ 7-3 การติดต่อระหว่าง IsagQ ด้วยกัน โดยใช้เฉพาะการแลกเปลี่ยนคีย์

การปรับปรุงของ IsagMQ & IsagQ ในยุคปัจจุบัน

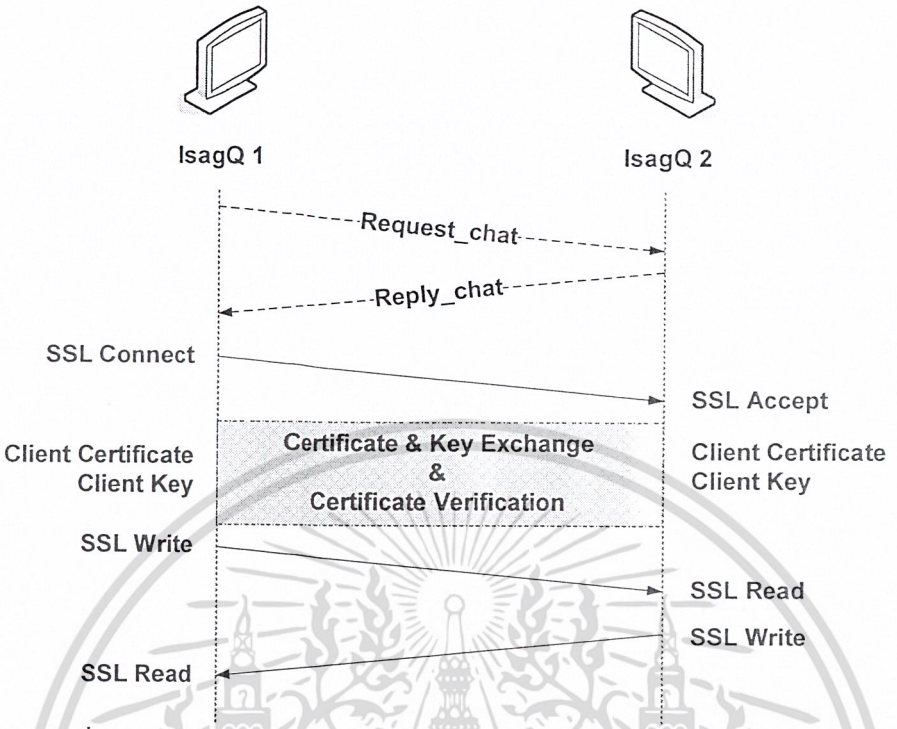
1. สามารถให้บริการโดยใช้ IsagQ ติดต่อมายัง IsagMQ ผ่านโพรโตคอล SSL ได้โดยไม่ขึ้นกับความแตกต่างของระบบปฏิบัติการที่ใช้ และ ภาษาที่ใช้เขียนโปรแกรม
2. การติดต่อระหว่าง IsagQ ด้วยกัน จะต้องติดต่อผ่าน SSL protocol เท่านั้น เพื่อ
   2.1 ไม่ให้เกิดปัญหา Man in the middle
   2.2 สามารถใช้ระบบพิสูจน์ตนเพื่อสร้างความมั่นใจให้กับผู้ใช้ทั้งสองฝั่งได้ว่ากำลังติดต่ออยู่กับบุคคลที่ต้องการจริงๆ ดังรูป7-4
3. เพิ่มการให้บริการ อาทิ การปฏิเสธ หรือ ยอมรับคู่สนทนา , การแสดงรายชื่อผู้รอคำยินยอม , การให้คำยินยอมแก่ผู้รอคำยินยอม และ การส่งไฟล์อย่างปลอดภัย ระหว่าง IsagQ ด้วยกันเอง



รูปที่ 7-4 IsagMQ & IsagQ ในยุคปัจจุบัน



รูปที่ 7-5 การติดต่อระหว่าง IsagQ กับ IsagQ โดยใช้ SSL Protocol



รูปที่ 7-6 การติดต่อระหว่าง IsagQ กับ IsagQ โดยมีการแลกเปลี่ยนคีย์และใบรับรองสิทธิ์

ความสามารถ	โครงานปี 46	โครงงานปี 47
IsagQ ติดต่อไปยัง IsagMQ ด้วย SSL Protocol	ไม่ได้	ได้
สื่อสารโดยมีการเข้ารหัสข้อมูล	ได้	ได้
ใบรับรองสิทธิ์ของ IsagQ	ไม่มี	มี
การพิสูจน์ตนระหว่าง IsagQ ด้วยกัน	Diffie-Hellman	SSL
บริการยอมรับ หรือ ปฏิเสธคู่สนทนา	ไม่มี	มี
บริการแสดงรายชื่อผู้รอคำยินยอม และ การตอบรับคำขอคำยินยอม	ไม่มี	มี
บริการลงทะเบียนและยกเลิกการลงทะเบียน	มี	ไม่มี
บริการติดต่อกับ ICQ	มี	ไม่มี
การส่งไฟล์ระหว่าง IsagQ แบบปลอดภัย	ไม่มี	มี

รูปที่ 7-7
ตารางสรุปการพัฒนาที่เกิดขึ้น

ส่วนที่ 2

การออกแบบ และ พัฒนา

IsagMQ & IsagQ ปี 2547

ระบบปฏิบัติการ และ เครื่องมือที่ใช้พัฒนา

สำหรับฝั่งโปรแกรมแม่ข่าย (IsagMQ)

1. ระบบปฏิบัติการ FreeBSD version 4.11
2. GCC Compiler – ใช้สำหรับการคอมไพล์ ภาษาซี
3. VIM – เท็กโหมด อิดิเตอร์
4. Anjuta – โปรแกรมสำเร็จรูปสำหรับการเขียนโปรแกรมขนาดใหญ่
5. VNC – ใช้สำหรับการล๊อกอินแบบกราฟฟิคโหมดจากระยะไกล ซึ่งสะดวกต่อการใช้งาน
6. Apache version 2.0.5 – เป็นโปรมแกรมสำหรับการทำเว็บเซิร์ฟเวอร์
7. PHP version 4.3.10 – เป็นภาษาโปรแกรม สำหรับการใช้แสดงผลผ่านเว็บเซิร์ฟเวอร์
8. MySQL version 4.1.8 – โปรแกรมสำหรับใช้ทำฐานข้อมูล
9. phpMyAdmin – โปรแกรมสำหรับติดต่อฐานข้อมูลผ่านเว็บเซิร์ฟเวอร์
10. OpenSSL Library – ไลบรารี สำหรับการสร้างช่องทางการสื่อสารอย่างปลอดภัย
11. Ethereal – โปรแกรมดักจับ packet

สำหรับฝั่งโปรแกรมลูกข่าย (IsagQ)

1. ระบบปฏิบัติการ Windows XP
2. Borland JBuilder – เป็นโปรแกรมสำหรับการเขียนด้วยภาษาจาวา
3. EditPlus – เป็นอิดิเตอร์สำหรับเขียนโปรแกรม
4. JDK-1.4 – แพ็คเกจ และ คอมไพเลอร์สำหรับเขียนโปรแกรมด้วยภาษาจาวา

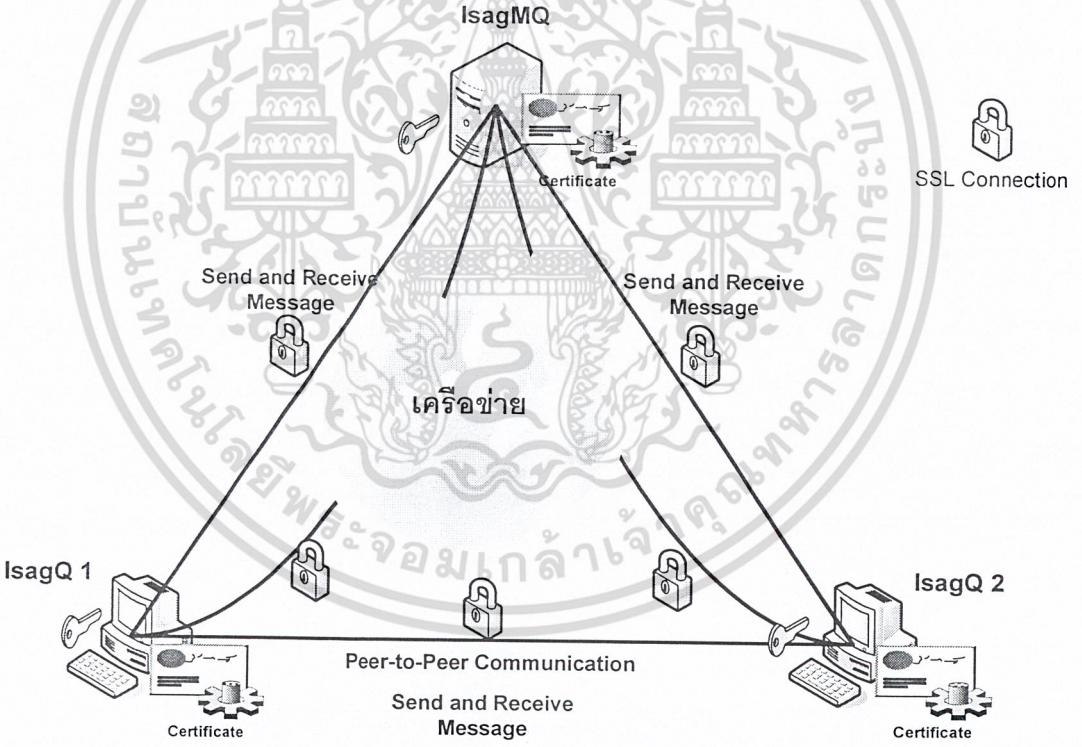
แนวคิดในการออกแบบ

โครงงานนี้มุ่งเน้นพัฒนาเพื่อสร้างความปลอดภัยดังนี้

1. โปรแกรมแม่ข่ายสามารถที่จะเข้าและถอดรหัสข้อความได้
2. โปรแกรมแม่ข่ายสามารถที่จะใช้ระบบพิสูจน์ตนเพื่อสร้างความมั่นใจในการสนทนา

1. รูปแบบการติดต่อสื่อสาร

ลักษณะการสื่อสารระหว่างโปรแกรมแม่ข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย กับโปรแกรมลูกข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย และ ระหว่างโปรแกรมลูกข่าย ด้วยกันเอง จะเป็นแบบ Peer-to-Peer ซึ่งทำให้การติดต่อสื่อสารมีความรวดเร็ว



รูปที่ 7-8 ช่องทางการสื่อสารแบบปลอดภัยของโปรแกรม

จากรูป 7-8 เป็นลักษณะการสื่อสารแบบ Peer-to-Peer ซึ่งมีการเสริมประสิทธิภาพในการเข้าและถอดรหัส รวมถึงการใช้ระบบพิสูจน์ตนด้วย

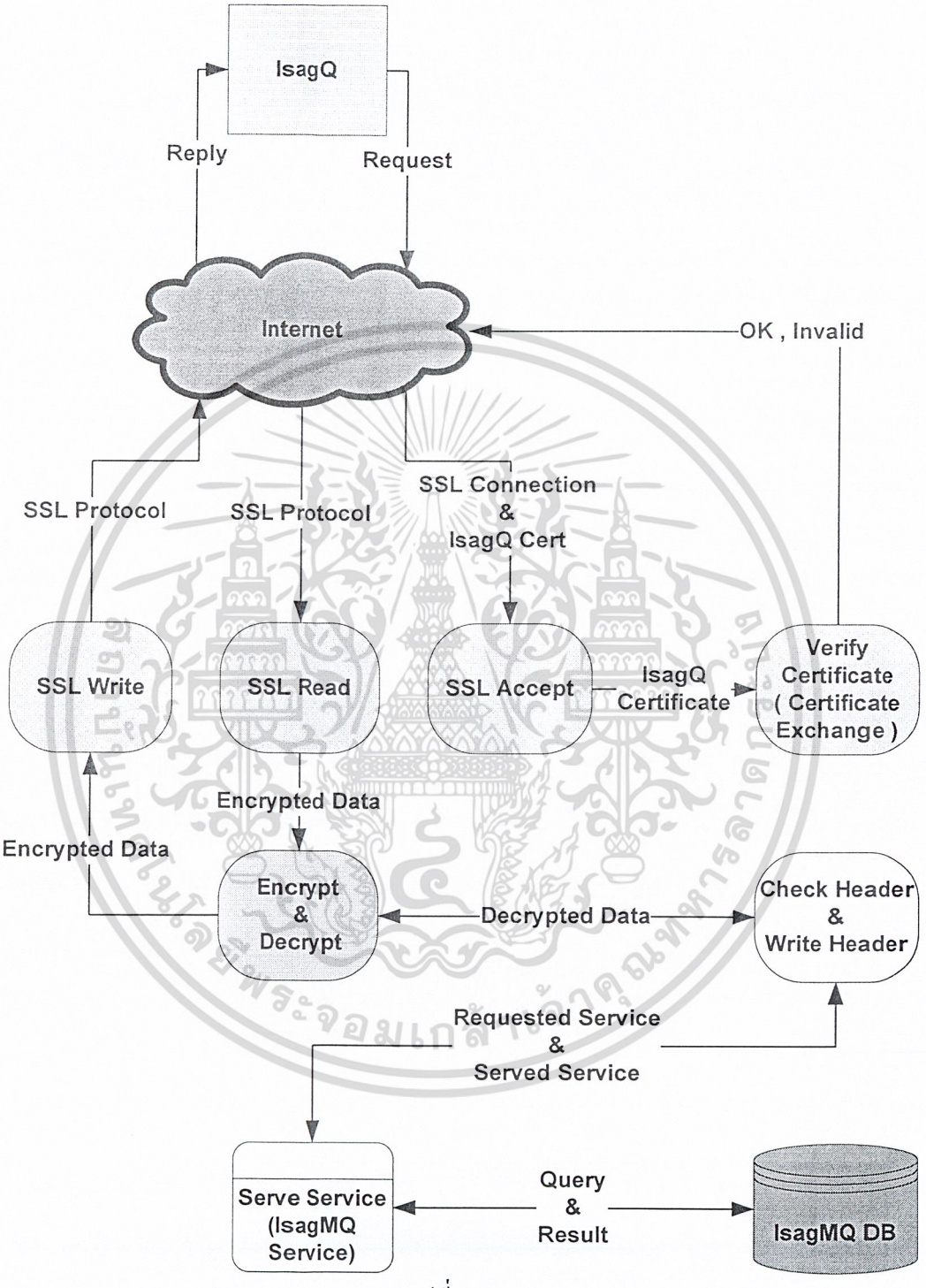
2. โครงสร้างโปรแกรม

เนื่องจากการพัฒนาโปรแกรมจะแบ่งออกเป็น 2 ส่วนด้วยกันดังนี้

- ส่วนของโปรแกรมแม่ข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย ซึ่งพัฒนาด้วย ภาษาซี บนระบบปฏิบัติการ FreeBSD 5.2.1
- ส่วนของโปรแกรมลูกข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย ซึ่งพัฒนาด้วยภาษา JAVA บนระบบปฏิบัติการ Windows XP

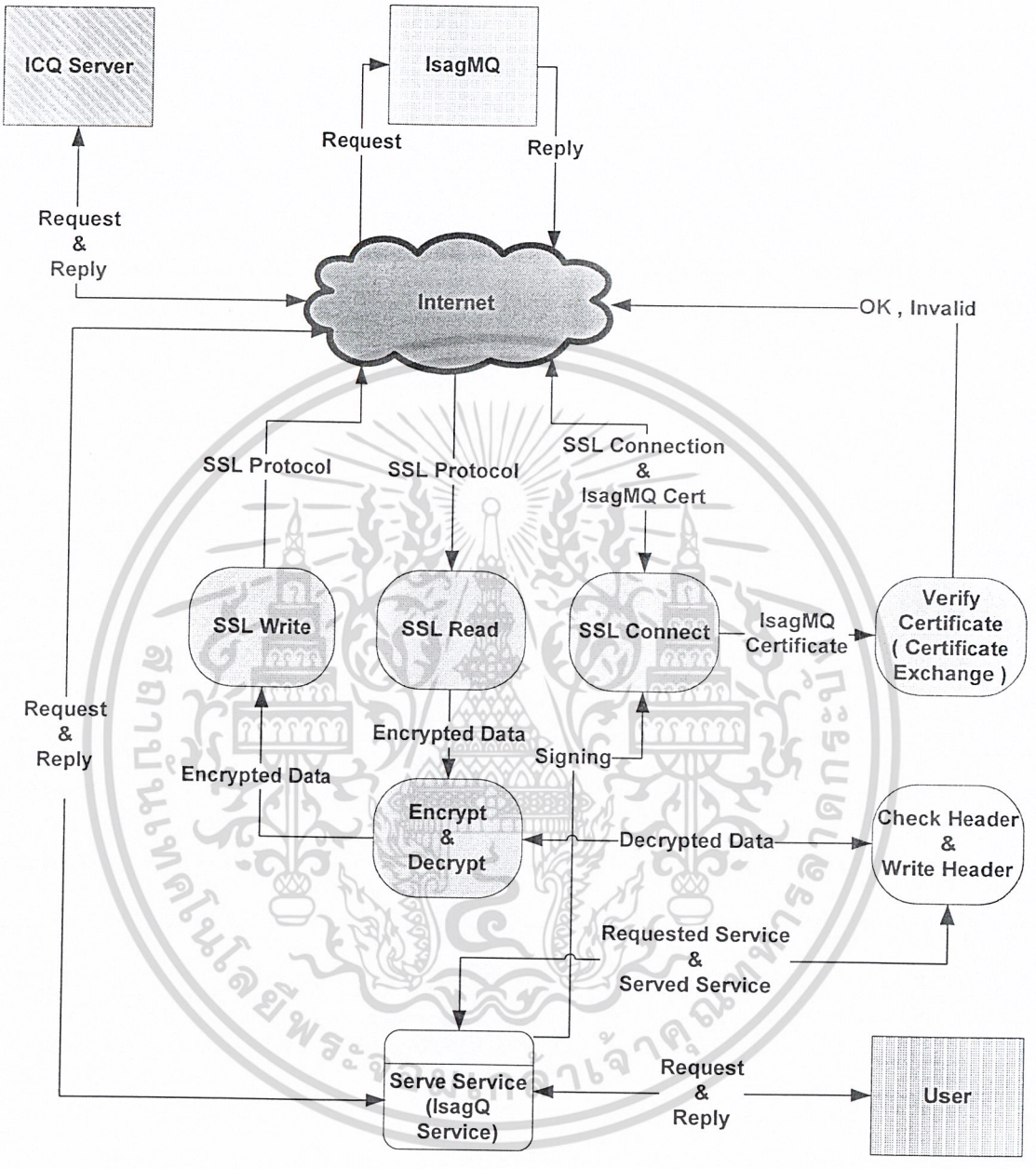
ในส่วนการติดต่อสื่อสารระหว่างโปรแกรมจะมีด้วยกัน 3 ลักษณะ

- การติดต่อสื่อสารระหว่างโปรแกรมแม่ข่ายกับโปรแกรมลูกข่าย โดยพิจารณาโปรแกรมแม่ข่ายเป็นหลัก ดังรูป 7-9
- การติดต่อสื่อสารระหว่างโปรแกรมลูกข่ายกับโปรแกรมแม่ข่าย โดยพิจารณาโปรแกรมลูกข่ายเป็นหลัก ดังรูป 7-10
- การติดต่อสื่อสารระหว่างโปรแกรมลูกข่ายด้วยกันเอง ดังรูป 7-11



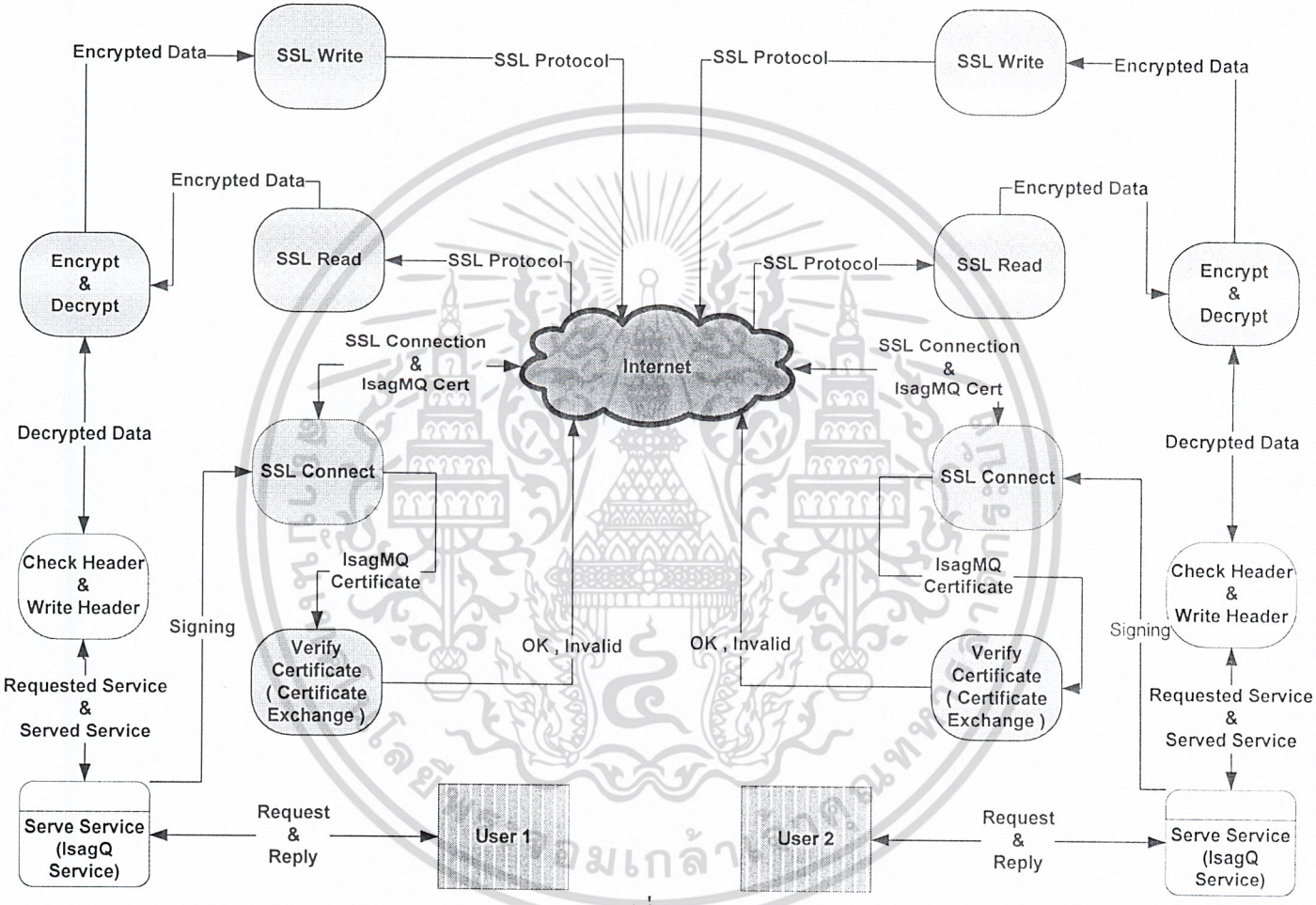
รูปที่ 7-9

โครงสร้างของโปรแกรมแม่ข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย



รูปที่ 7-10

โครงสร้างของโปรแกรมลูกข่ายสำหรับรับส่งสารด่วนแบบปลอดภัย



รูปที่ 7-11

โครงสร้างการติดต่อสื่อสารระหว่างโปรแกรมลูกข่ายด้วยกันเอง

3. คุณลักษณะของโปรแกรม (Input/Output)

IsagMQ (โปรแกรมฝั่งแม่ข่าย)

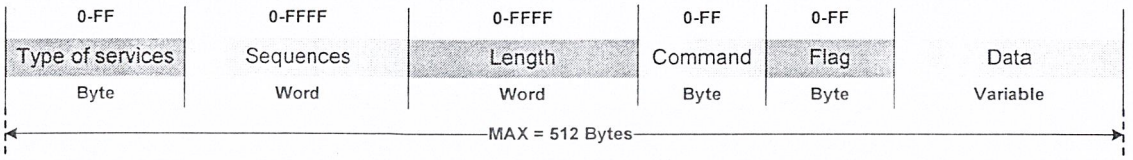
ด้านการให้บริการ

1. ออก ใบรับรองสิทธิ์ ให้กับ ผู้ใช้
2. จัดทำฐานข้อมูลของผู้ใช้ที่ลงทะเบียนแล้ว
3. สามารถรับส่งข้อมูลเกี่ยวกับสถานะ และ ชื่อของผู้ใช้ได้
4. สามารถให้บริการลงทะเบียนและยกเลิกการลงทะเบียนได้
5. สามารถให้บริการล๊อกอินและล๊อกเอาท์ได้
6. สามารถให้บริการเพิ่มและลบคู่สนทนาได้
7. สามารถให้บริการค้นหาคู่สนทนาได้
8. สามารถให้บริการเปลี่ยนชื่อเล่นของผู้สนทนาได้
9. สามารถให้บริการปฏิเสธ และ ยกเลิกการปฏิเสธคู่สนทนา
10. สามารถให้บริการยอมรับคู่สนทนา
11. เปลี่ยนสถานะในการสนทนา

IsagQ (ฝั่งไคลเอ็นต์)

1. สามารถให้บริการลงทะเบียนและยกเลิกการลงทะเบียนได้
2. สามารถให้บริการล๊อกอินและล๊อกเอาท์ได้
3. สามารถให้บริการเพิ่มและลบคู่สนทนาได้
4. สามารถให้บริการค้นหาคู่สนทนาได้
5. สามารถให้บริการเปลี่ยนชื่อเล่นของผู้สนทนาได้
6. แสดงสถานะ ของ สมาชิกทั้งสองโปรแกรม
7. ผู้ใช้สามารถใช้งานรวมกับโปรแกรม ICQ ของ Mirabilis
8. ผู้ใช้สามารถรับส่งไฟล์ข้อมูลระหว่างกันได้
9. ผู้ใช้ติดต่อสื่อสารระหว่างกันแบบ peer-to-peer
10. ผู้ใช้ติดต่อสื่อสารระหว่างกันอย่างปลอดภัยโดยมีการเข้ารหัสข้อมูลสำหรับข้อมูลระหว่าง IsagQ ด้วยกัน ส่วนข้อมูลระหว่าง ISAG Q กับ ICQ จะเป็น Plain Text
11. ผู้ใช้สามารถติดตั้งบนระบบปฏิบัติการ Windows ได้หลายเวอร์ชั่น

4. IsagMQ & IsagQ Protocol



รูปที่ 7-12

แสดงรูปแบบของโพรโตคอลที่ใช้สำหรับ IsagMQ และ IsagQ

คำอธิบาย

1. Type of services คือ การระบุว่าต้องการใช้บริการชนิดใด โดยมีรายละเอียดดังต่อไปนี้
   - 0x01 = Join
   - 0x02 = Login
   - 0x03 = Add Contact List
   - 0x04 = Change Nickname
   - 0x05 = Status
   - 0x06 = Error
   - 0x07 = Logout
   - 0x08 = Find Contact List
   - 0x0A = Disjoin
   - 0x0B = Delete Contact List
   - 0x51 = Set Status
   - 0x55 = Deny /Admit User
   - 0x5E = Authorized Request / Accept Authorized Request
2. Sequences คือ หมายเลขลำดับของแพ็กเกต
3. Length คือ ความยาวของแพ็กเกต ซึ่งจะมีความยาวสูงสุดเท่ากับ 512 ไบต์ เพื่อไม่ต้องการให้แพ็กเกตถูกแฟรกเมนต์
4. Command คือ คำสั่งย่อยของแต่ละ Type of services โดยดูได้จากรูปที่ 2
5. Flag คือ สภาวะการทำงานพิเศษนอกเหนือจากคำสั่งในข้อ 4

6. Data คือ ข้อความที่ใช้ในการรับส่ง โดยใช้ $ เป็นตัวจบข้อมูลในแต่ล่ะส่วน

รูปที่ 7-13

ตารางแสดงความสัมพันธ์ระหว่าง Type of services กับ Command

Type of Services	รหัสคำสั่ง	ชื่อคำสั่ง	รายละเอียด
0x01	0x10	Request_register	ไคลเอ็นต์ร้องขอใช้บริการลงทะเบียนสมาชิกใหม่
	0x11	Response_register	เซิร์ฟเวอร์ตอบรับการลงเบียนสมาชิกใหม่
0x02	0x21	Request_login	ไคลเอ็นต์ร้องขอใช้บริการล็อกอิน
	0x22	Response_login	เซิร์ฟเวอร์ตอบรับการล็อกอิน
0x03	0x32	Request_add_email	ไคลเอ็นต์ร้องขอใช้บริการแอดคอนแท็กลิสต์โดยใช้ Email address
	0x34	Response_add_email	เซิร์ฟเวอร์ตอบรับการแอดคอนแท็กลิสต์โดยใช้ Email address
	0x31	Request_add_userID	ไคลเอ็นร้องขอใช้บริการแอดคอนแท็กลิสต์โดยใช้ User_id
	0x33	Response_add_userID	เซิร์ฟเวอร์ตอบรับการแอดคอนแท็กลิสต์โดยใช้ User_id
0x04	0x41	Request_change_nick_name	ไคลเอ็นต์ร้องขอใช้บริการเปลี่ยนชื่อเล่นโดยส่งชื่อเล่นใหม่มาด้วย
	0x42	Response_change_nick_name	เซิร์ฟเวอร์ยืนยันว่าเปลี่ยนชื่อเล่นสมบูรณ์แล้ว
0x05	0x51	Request_status	ไคลเอ็นต์ร้องขอสถานะบนคอนแท็กลิสต์
	0x52	Response_status	เซิร์ฟเวอร์แจ้งสถานะคอนแท็กลิสต์
0x06	0x06	Error_msg	เซิร์ฟเวอ์แจ้งความผิดพลาดที่เกิดขึ้น
0x07	0x71	Request_logout	ไคลเอ็นต์จบการใช้บริการ

	0x72	Response_logout	เซิร์ฟเวอร์ยืนยันการจบการใช้บริการ
0x08	0x81	Request_find	ไคลเอ็นต์ร้องขอใช้บริการค้นหาคู่สนทนา
	0x82	Response_find	เซิร์ฟเวอร์ตอบรับการค้นหาคู่สนทนา
0x0A	0xA1	Request_delete_admin	ไคลเอ็นต์ร้องขอการยกเลิกเป็นสมาชิก
	0xA2	Response_delete_admin	เซิร์ฟเวอร์ทำการยืนยันว่าได้ทำการยกเลิกการเป็นสมาชิกแล้ว
0x0B	0xB1	Request_delete_contact	ไคลเอ็นต์ร้องขอการลบคอนแท็กพร้อมส่ง User_id ของคอนแท็กลิสต์ที่ต้องการลบ
	0xB2	Response_delete_contact	เซิร์ฟเวอร์ทากรยืนยันว่าได้ทำการลบคอนแท็กลิสต์ที่ต้องการแล้ว
0x51	0x51	Request_set_status	ไคลเอ็นต์ร้องขอการเปลี่ยนสถานะ
	0x52	Response_set_status	เซิร์ฟเวอร์ยืนยันการเปลี่ยนสถานะของไคลเอ็นต์
0x55	0x51	Request_deny_user	ไคลเอ็นต์ร้องขอการปฏิเสธคู่สนทนา
	0x52	Response_deny_user	เซิร์ฟเวอร์ยืนยันการปฏิเสธคู่สนทนา
	0x5A	Request_admit_user	ไคลเอ็นต์ร้องขอการยอมรับคู่สนทนา
	0x5B	Response_admit_user	เซิร์ฟเวอร์ยืนยันการยอมรับคู่สนทนา
0x5E	0x51	Auth_request_list	ไคลเอ็นต์ร้องขอรายชื่อผู้ที่รอคำยินยอม
	0x52	Response_auth_list	เซิร์ฟเวอร์ส่งรายชื่อผู้ที่รอคำยินยอม
	0x55	Send_accept_auth	ไคลเอ็นต์ยินยอมตามคำขอ
	0x56	Response_accept_auth	เซิร์ฟเวอร์ยืนยันผลการยินยอม

5. ฐานข้อมูล

ฐานข้อมูลของ IsagMQ ประกอบด้วย ตารางหลัก 2 ตารางดังนี้

1. ตาราง user มีรายละเอียดดังนี้
   a. User_ID
      - Primary Key
      - เป็นตัวเลข ขนาดไม่เกิน 10 หลัก
   b. FIRST_NAME
      - ห้ามเป็น NULL
      - มีความยาวไม่เกิน 64 ตัวอักษร
   c. LAST_NAME
      - ห้ามเป็น NULL
      - มีความยาวไม่เกิน 64 ตัวอักษร
   d. NICK_NAME
      - ห้ามเป็น NULL
      - มีความยาวไม่เกิน 64 ตัวอักษร
   e. SEX
      - ห้ามเป็น NULL
      - เป็น 'M' (Male), 'F'(Female) , 'N'(None)
   f. AGE
      - ห้ามเป็น NULL
      - มีค่าไม่เกิน 99
   g. EMAIL
      - ห้ามเป็น NULL
      - มีความยาวไม่เกิน 64 ตัวอักษร

2. ตาราง contact_list มีรายละเอียดดังนี้

a. User_ID

- Combined Key
- เป็นตัวเลข ขนาดไม่เกิน 10 หลัก

b. CONTACT_ID

- Combined Key
- เป็นตัวเลข ขนาดไม่เกิน 10 หลัก

c. STATUS

- ห้ามเป็น NULL
- เป็นตัวอักษร ขนาด 1 ตัวอักษร มีดังนี้

1. 'O' – Online
2. 'F' – Offline
3. 'B' – Busy
4. 'A' – Away
5. 'W' – Waiting for Authorization
6. 'D' – Block
7. 'X' – User is blocked
8. 'U' – Deny to accept Authorzation

ส่วนที่ 3

การสถาปนา Root CA
และ
การจัดการเกี่ยวกับใบรับรองสิทธิ์

IsagMQ & IsagQ ปี 2547

การสถาปนา Root CA และ การออกใบรับรองสิทธิ์

ขั้นตอนในการติดตั้ง สถาปนาเป็น ROOT CA

- **openssl** req -newkey rsa:1024 -sha1 -keyout root.key -out root.req -config root.cnf
- **openssl** x509 -req -in root.req -sha1 -extfile root.cnf -extensions certificate_extensions -signkey root.key -out root.cert
- **cat** root.cert root.key > root.pem

ขั้นตอนในการออกใบรับรองสิทธิ์ให้กับ SERVER CA

- **openssl** req -newkey rsa:1024 -sha1 -keyout serverCA.key -out serverCA.req -config serverCA.cnf
- **openssl** x509 -req -in serverCA.req -sha1 -extfile serverCA.cnf -extensions certificate_extensions -CA root.pem -CAkey root.pem -CAcreateserial -out serverCA.cert
- **cat** serverCA.cert serverCA.key root.cert > serverCA.pem

ขั้นตอนในการออกใบรับรองสิทธิ์ให้กับ SERVER

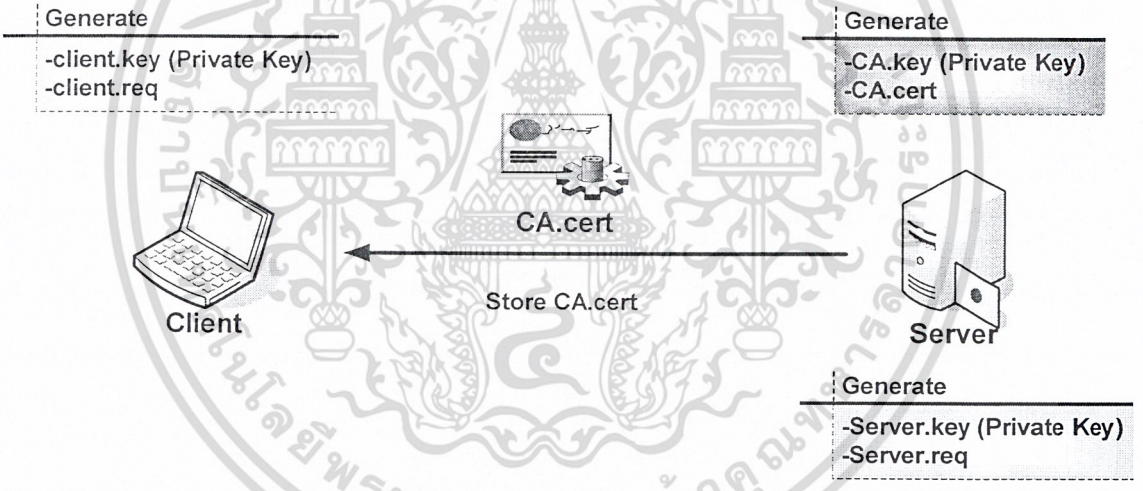
- **openssl** req -newkey rsa:1024 -sha1 -keyout server.key -out server.req -config server.cnf -reqexts req_extensions
- **openssl** x509 -req -in server.req -sha1 -extfile server.cnf -extensions certificate_extensions -CA serverCA.pem -CAkey serverCA.pem -CAcreateserial -out server.cert
- **cat** server.cert server.key serverCA.cert root.cert > server.pem

ขั้นตอนในการร้องขอใบรับรองสิทธิ์ จาก Private CA

ขั้นตอนที่ 1

เนื่องจากกรณีนี้ตัวเซิร์ฟเวอร์(IsagMQ) ทำตัวเป็น Private CA ดังนั้นที่ตัวเซิร์ฟเวอร์เองจะมี CA.key และ CA.cert อยู่ในตัว จะเห็นได้ว่า จะตัดส่วน Root CA ออกไป (ตามหัวข้อ ขั้นตอนในการขอใบรับรองสิทธิ์จาก Public CA) จากนั้นตัวเซิร์ฟเวอร์เองจะต้องสร้าง Server.key และ Server.req เพื่อไว้ใช้ในการขอใบรับรองสิทธิ์จากตัวเอง ส่วนทางฝั่งไคลเอ็นต์ก็ต้องมี Client.key และ Client.req เช่นเดียวกับเซิร์ฟเวอร์

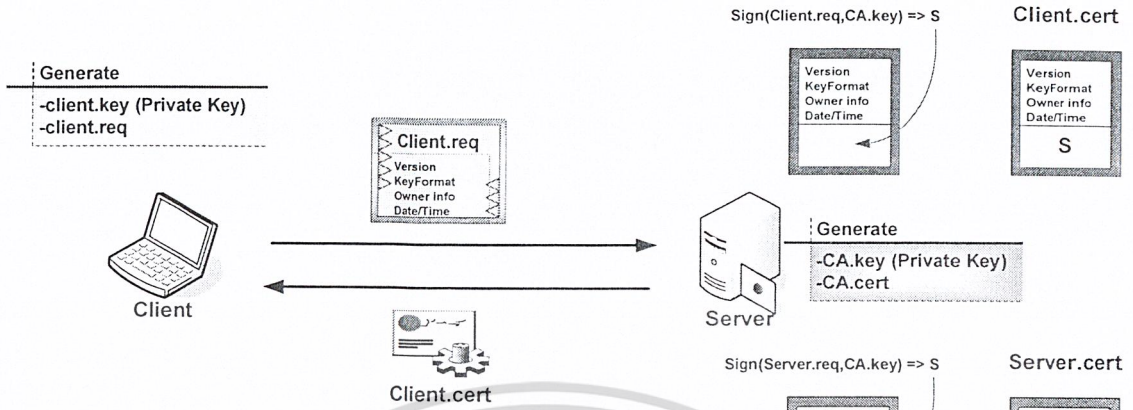
จากนั้น ผู้ร้องขอใบรับรองสิทธิ์ควรจะมี CA.cert เก็บไว้กับตัวเองเพื่อใช้ในการทำการตรวจสอบใบรับรองสิทธิ์รวมถึงการเข้ารหัสด้วย ดังรูป 7-14



รูปที่ 7-14 แสดงขั้นตอนที่ 1 ของการร้องขอใบรับรองสิทธิ์

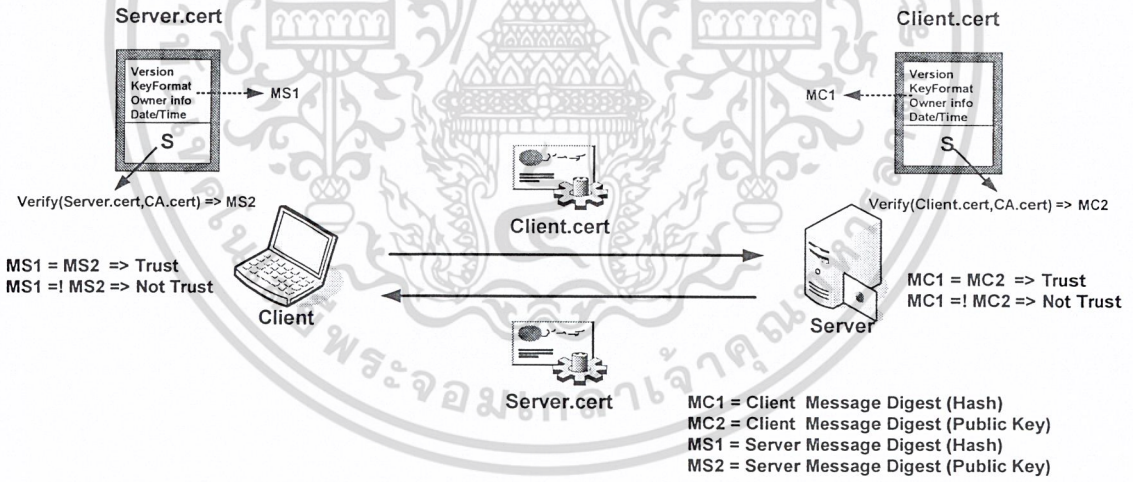
ขั้นตอนที่ 2

จากนั้น ทั้งไคลเอ็นต์ และ เซิร์ฟเวอร์จะส่ง Client.req และ Server.req ไปยังเซิร์ฟเวอร์ เพื่อให้เซิร์ฟเวอร์ออกใบรับรองสิทธิ์ให้กับตน โดยการนำ Client.req หรือ Server.req มาเข้าแฮชฟังก์ชั่น แล้วเซ็นรับรองด้วยกุญแจส่วนตัวของเซิร์ฟเวอร์นั่นคือ CA.key ดังรูป 7-15



รูปที่ 7-15 แสดงขั้นตอนที่ 2 ของการร้องขอใบรับรองสิทธิ์

ขั้นตอนในการตรวจสอบใบรับรองสิทธิ์ที่ออกโดย Private CA



รูปที่ 7-16 แสดงขั้นตอนที่ 3 ของการร้องขอใบรับรองสิทธิ์

จากรูป 7-16 ทุกครั้งที่การตรวจสอบใบรับรองสิทธิ์โดยการแลกเปลี่ยนใบรับรองสิทธิ์ของอีกฝั่งหนึ่งนั้น จะนำเอาใบรับรองสิทธิ์ดังกล่าวมาทำการตรวจสอบ (Verify) ด้วย CA.cert โดยมีขั้นตอนดังนี้

- นำเอาข้อความในใบรับรองสิทธิ์มาเข้าแฮชฟังก์ชั่น ซึ่งจะได้ Message Digest อันแรกออกมา
- นำเอาลายเซ็นอิเล็กทรอนิกส์ (Digital Signature) มาทำการตรวจสอบโดยใช้ฟังก์ชั่น Verify ด้วย CA.cert ซึ่งจะได้ Message Digest อีกอัน
- นำค่า Message Digest อันแรก กับ อันหลัง มาทำการเปรียบเทียบค่ากัน ซึ่งหากเท่ากันแสดงว่าสามารถมั่นใจได้ว่าบุคคลที่กำลังติดต่ออยู่เป็นบุคคลที่ต้องการจริง แต่หากไม่เท่ากันแสดงว่าไม่สามารถมั่นใจบุคคลดังกล่าวได้ การเชื่อมต่อก็จะสิ้นสุดลงทันที

การขอใบรับรองสิทธิ์โดย ไคลเอ็นต์ในทางปฏิบัติ แบ่งเป็น 2 แบบ

1 ขั้นตอนในการออกใบรับรองสิทธิ์ให้กับ CLIENT บน FreeBSD

- **openssl** req -newkey rsa:1024 -sha1 -keyout client.key -out client.req -config client.cnf -reqexts req_extensions
- **openssl** x509 -req -in client.req -sha1 -extfile client.cnf -extensions certificate_extensions -CA root.pem -CAkey root.pem -CAcreateserial -out client.cert
- **cat** client.cert client.key root.cert > client.pem

ขั้นตอนในการสร้าง Diffie-Hellman Key ขนาด 512 บิต
dh512.pem:

- **openssl** dhparam -check -text -5 512 -out dh512.pem

ขั้นตอนในการสร้าง Diffie-Hellman Key ขนาด 1024 บิต
dh1024.pem:

- **openssl** dhparam -check -text -5 1024 -out dh1024.pem

2. ขั้นตอนในการออกใบรับรองสิทธิ์ให้กับ CLIENT บน Windows XP ด้วย JAVA

เนื่องจากในการออกแบบ โปรแกรมลูกข่ายได้ใช้ ภาษา JAVA ในการพัฒนา และใช้ แพ็กเกจ ของ JSSE (JDK 1.4) ในการสร้างช่องทางการสื่อสารแบบปลอดภัย (SSL) ติดต่อไปยัง โปรแกรมแม่ข่าย รวมถึง โปรแกรมลูกข่ายด้วยกันเอง โดยแพ็กเกจดังกล่าวจะใช้รูปแบบเฉพาะตัวในการเก็บ และจัดการเกี่ยวกับใบรับรองสิทธิ์ ด้วย Class KeyStore ดังนั้นจึงจำเป็นที่ต้องใช้โปรแกรม Keytool ซึ่งเป็นโปรแกรมทางฝั่ง JAVA ที่ใช้ในการจัดการเกี่ยวกับใบรับรองสิทธิ์ทั้งหมด สร้าง ใบร้องขอใบรับรองสิทธิ์ แล้วจึงให้ ทาง CA เซ็นรับรองเพื่อให้ได้มาซึ่งใบรับรองสิทธิ์ โดยมีขั้นตอนดังต่อไปนี้

ขั้นตอนในการขอใบรับรองสิทธิ์

2.1 ใช้โปรแกรม keytool ออก ใบขอใบรับรองสิทธิ์ โดยผลลัพธ์คือ userID.req

- **keytool** –genkey –alias userID –keyalg RSA –keystore keystorename
- **keytool** –keystore keystorename –certreq –alias userID –file userID.req

2.2 นำ userID.req ให้ RootCA ทำการเซ็นรับรอง และแปลงให้อยู่ในรูปแบบ DER

#openssl x509 –req –in userID.req –sha1 –extensions certificate_extensions –
CA root.pem –Cakey root.pem –Cacreateserial –out userID.pem –days 365

#openssl x509 –in root.pem –out root.der –outform DER

#openssl x509 –in server.pem –out server.der –outform DER

#openssl x509 –in userID.pem –out userID.der –outform DER

2.3 ทำการนำเข้า ไฟล์ดังต่อไปนี้ root.der , server.der และ userID.der ลงใน keystore ที่สร้างเอาไว้

#keytool –keystore keystorename –alias RootCA –import –file root.der

#keytool –keystore keystorename –alias ServerCA –import –file server.der

#keytool –keystore keystorename –alias userID –import –file userID.der

หมายเหตุ -userID คือ ชื่อใบรับรองสิทธิ์ของผู้ขอ

-Keystorename คือ ชื่อที่เก็บใบรับรองสิทธิ์

-คำสั่ง keytool จะสร้าง file ที่ไดเรคทอรีที่ รัน คำสั่งดังกล่าว

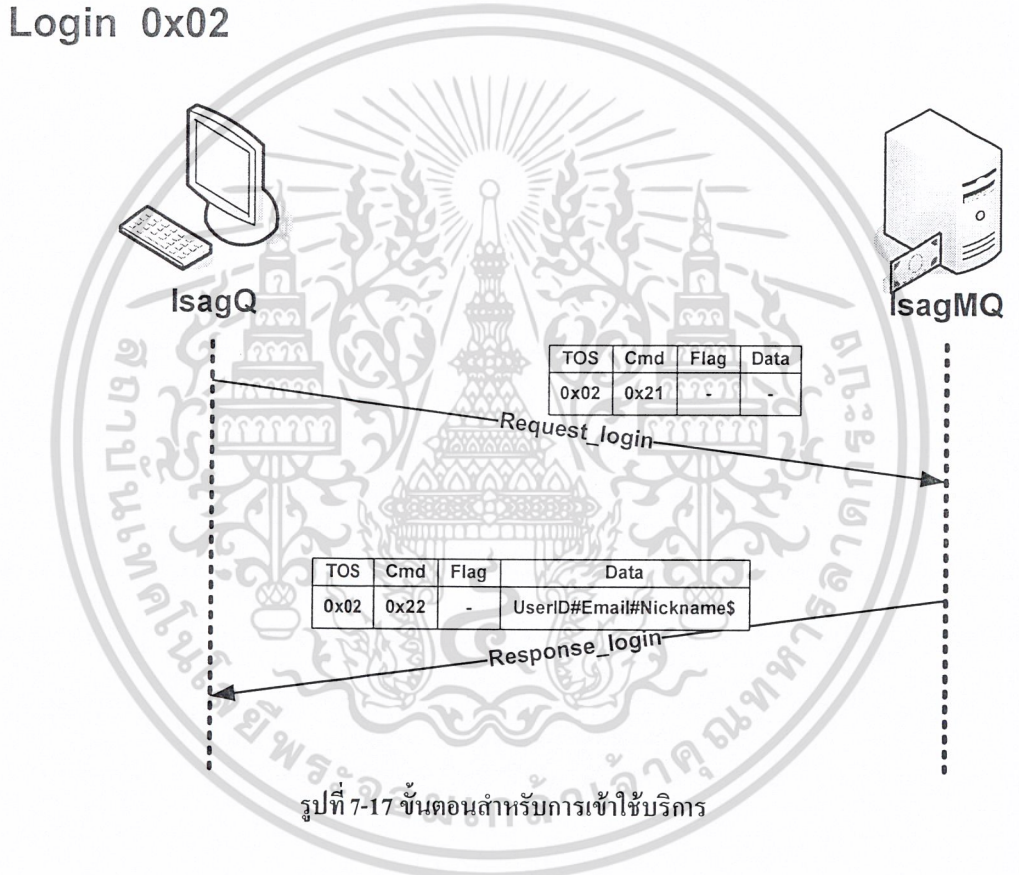
ส่วนที่ 4

รายละเอียดการให้บริการ

IsagMQ & IsagQ ปี 2547

บริการต่างๆ ของโปรแกรม

ในส่วนนี้จะกล่าวถึงบริการต่างๆที่เกิดขึ้นระหว่าง IsagMQ กับ IsagQ รวมถึง IsagQ กับ IsagQ ด้วยกันเอง โดยอธิบายผ่าน โฟลว์ไดอะแกรม เพื่อให้ง่ายต่อการเข้าใจ และ เห็นถึงความเปลี่ยนแปลงในแต่ล่ะส่วนของ แพ็กเกต (ในที่นี้ TOS = Type of Services และ Cmd = Command)โดยมีรายละเอียดดังต่อไปนี้



รูปที่ 7-17 ขั้นตอนสำหรับการเข้าใช้บริการ

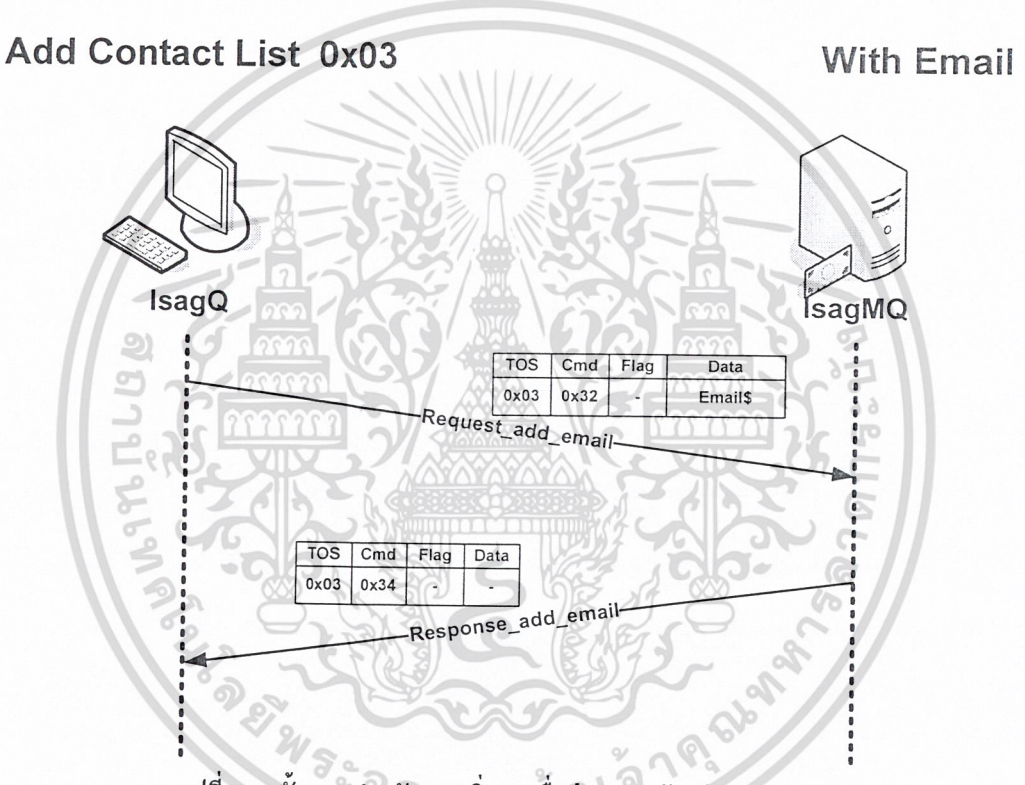
ขั้นตอนสำหรับ การเข้าใช้บริการ

(Login)

1. ในช่องของ TOS จะเป็น 0x02 และ Cmd จะเป็น 0x21 ส่งไปยังโปรแกรมแม่ข่าย
2. เมื่อโปรแกรมฝั่งแม่ข่ายเสร็จสิ้นขบวนการอย่างสมบูรณ์ ก็จะทำการเปลี่ยนเฉพาะ Cmd เป็น 0x22 พร้อมทั้งส่งข้อมูลดังนี้

- User ID - หมายเลขของผู้ใช้คนดังกล่าว
- Email - อีเมลแอดเดรสของผู้ใช้คนดังกล่าว
- Nickname - ชื่อเล่นล่าสุดของผู้ใช้คนดังกล่าว

3. เมื่อผู้ใช้ได้รับ Frame ที่มี TOS เป็น 0x02 และ Cmd เป็น 0x22 ก็จะทำการนำข้อความใน Frame ดังกล่าวมาตีความเพื่อเก็บข้อมูลทั้ง 3 อย่างนำไปใช้ต่อไป



รูปที่ 7-18 ขั้นตอนสำหรับการเพิ่มรายชื่อคู่สนทนาด้วยอีเมลแอดเดรส

ขั้นตอนสำหรับการ การเพิ่มรายชื่อคู่สนทนา ด้วย อีเมลแอดเดรส

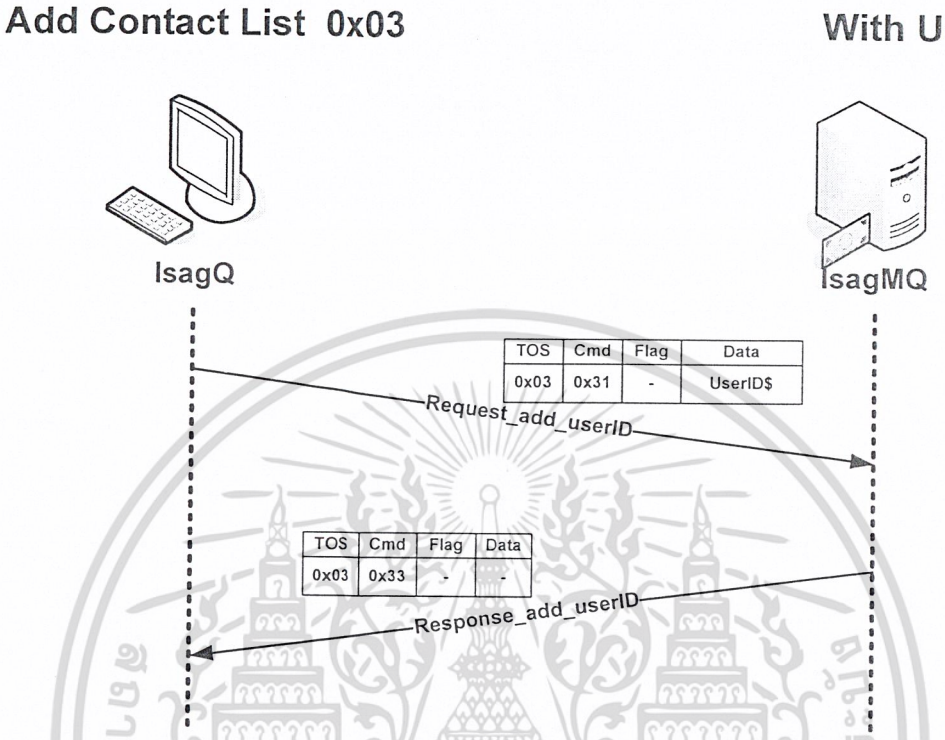
(Add Contact List with Email)

1 ในช่องของ TOS จะเป็น 0x03 และ Cmd จะเป็น 0x32 พร้อมทั้งระบุ อีเมลแอดเดรส ของผู้ที่ต้องการเพิ่มลงในช่อง Data และปิดการสิ้นสุดของข้อมูลด้วย $ ส่งไปยังโปรแกรมแม่ข่าย

- ในขั้นตอนนี้โปรแกรมฝั่งผู้ใช้จะทำการตรวจสอบว่าอีเมลดังกล่าวถูกต้องตามเงื่อนไขที่โปรแกรมฝั่งแม่ข่ายต้องการหรือไม่ ก่อนที่จะอนุญาตบรรจุข้อความลงในช่อง Data ได้
- ความยาวของ อีเมลเอดเดรส ต้องไม่เกิน 64 ตัวอักษร

2 โปรแกรมฝั่งแม่ข่ายรับ Frame ที่ผู้ใช้ส่งมาแล้วทำการถอดข้อความเพื่อเอาอีเมลแอดเดรสออกมา หลังจากนั้น จะมีลำดับขั้นตอนดังต่อไปนี้

- ตรวจสอบว่าอีเมลแอดเดรสถูกต้องทุกประการตามที่โปรแกรมฝั่งแม่ข่ายต้องการ
- ตรวจสอบว่ามีรายชื่อ อีเมลแอดเดรสดังกล่าวอยู่ในฐานข้อมูลจริง และไม่ใช่อีเมลแอดเดรสของผู้ใช้เอง ผลจากการตรวจสอบจะได้ User ID ที่สัมพันธ์กับ อีเมลแอดเดรสดังกล่าว หาก อีเมลนั้นมีอยู่จริงตามเงื่อนไข
- ทำการเพิ่มรายชื่อผู้ใช้คนดังกล่าวลงในตาราง contact_list โดยกำหนดค่าใน column STATUS เป็น 'W' ซึ่งหมายถึงกำลังรอคำยินยอมจากผู้ใช้ที่ถูกเพิ่มรายชื่อ แต่จะไม่มีการ เพิ่ม row ใดๆ ลงในฝั่งผู้ที่ถูกเพิ่มชื่อ จนกว่าผู้ที่ถูกเพิ่มชื่อจะยินยอมให้ผู้เพิ่มชื่อเสียก่อน
- เมื่อเสร็จสิ้นขบวนการดังกล่าว โปรแกรมฝั่งแม่ข่ายจะทำการเปลี่ยนเฉพาะช่อง Cmd จะเป็น 0x34 ส่งกลับไปยังฝั่งผู้ใช้เป็นการสิ้นสุดการ เพิ่มรายชื่อคู่สนทนาอย่างสมบูรณ์ และ ถูกต้อง



รูปที่ 7-19 ขั้นตอนสำหรับการเพิ่มรายชื่อคู่สนทนาด้วยรหัสผู้ใช้

ขั้นตอนสำหรับการ การเพิ่มรายชื่อคู่สนทนา ด้วย รหัสผู้ใช้

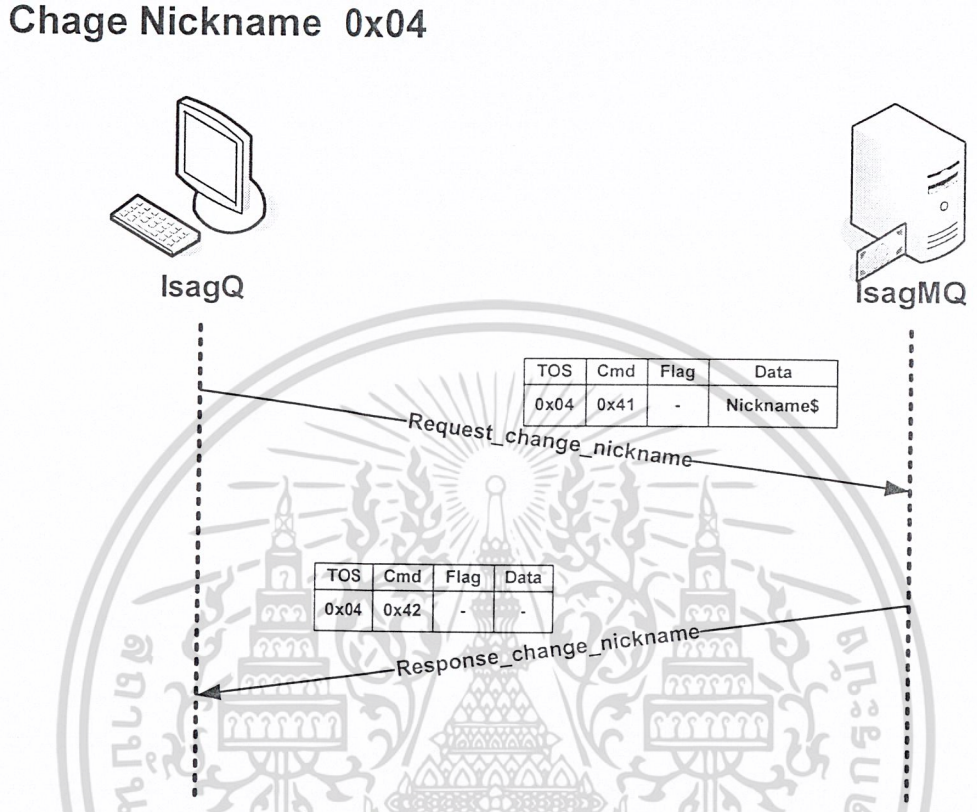
(Add Contact List with User ID)

1 ในช่องของ TOS จะเป็น 0x03 และ Cmd จะเป็น 0x31 พร้อมทั้งระบุ User ID ของผู้ที่ต้องการเพิ่มลงในช่อง Data และปิดการสิ้นสุดของข้อมูลด้วย $ ส่งไปยังโปรแกรมแม่ข่าย

- ในขั้นตอนนี้โปรแกรมฝั่งผู้ใช้จะทำการตรวจสอบว่า User ID ดังกล่าวถูกต้องตามเงื่อนไขที่โปรแกรมฝั่งแม่ข่ายต้องการหรือไม่ ก่อนที่จะอนุญาตบรรจุข้อความลงในช่อง Data ได้
- ความยาวของ User ID ต้องไม่เกิน 10 ตัวอักษร

2 โปรแกรมฝั่งแม่ข่ายรับ Frame ที่ผู้ใช้ส่งมาแล้วทำการถอดข้อความเพื่อเอา User ID ออกมา หลังจากนั้น จะมีลำดับขั้นตอนดังต่อไปนี้

- ตรวจสอบว่า User ID ถูกต้องทุกประการตามที่โปรแกรมฝั่งแม่ข่ายต้องการ
- ตรวจสอบว่ามีรายชื่อ User ID ดังกล่าวอยู่ในฐานข้อมูลจริง และ ไม่ใช่ User ID ของผู้ใช้เอง
- ทำการเพิ่มรายชื่อผู้ใช้คนดังกล่าวลงในตาราง contact_list โดยกำหนดค่าใน column STATUS เป็น 'W' ซึ่งหมายถึงกำลังรอคำยินยอมจากผู้ใช้ที่ถูกเพิ่มรายชื่อ แต่จะไม่มีการ เพิ่ม row ใดๆสำหรับผู้ที่ทำการเพิ่มชื่อ ลงในฝั่งผู้ที่ถูกเพิ่มชื่อ จนกว่าผู้ที่ถูกเพิ่มชื่อจะยินยอมให้ผู้เพิ่มชื่อเสียก่อน
- เมื่อเสร็จสิ้นขบวนการดังกล่าว โปรแกรมฝั่งแม่ข่ายจะทำการเปลี่ยนเฉพาะช่อง Cmd จะเป็น 0x33 ส่งกลับไปยังฝั่งผู้ใช้เป็นการสิ้นสุดการ เพิ่มรายชื่อคู่สนทนาอย่างสมบูรณ์ และ ถูกต้อง



รูปที่ 7-20 ขั้นตอนสำหรับการเปลี่ยนชื่อเล่นของผู้ใช้

ขั้นตอนสำหรับการ การเปลี่ยนชื่อเล่นผู้ใช้

(Change Nickname)

1 ในช่องของ TOS จะเป็น 0x04 และ Cmd จะเป็น 0x41 พร้อมทั้งระบุ ชื่อเล่นใหม่ ของผู้ใช้ลงในช่อง Data และปิดการสิ้นสุดของข้อมูลด้วย $ ส่งไปยังโปรแกรมแม่ข่าย

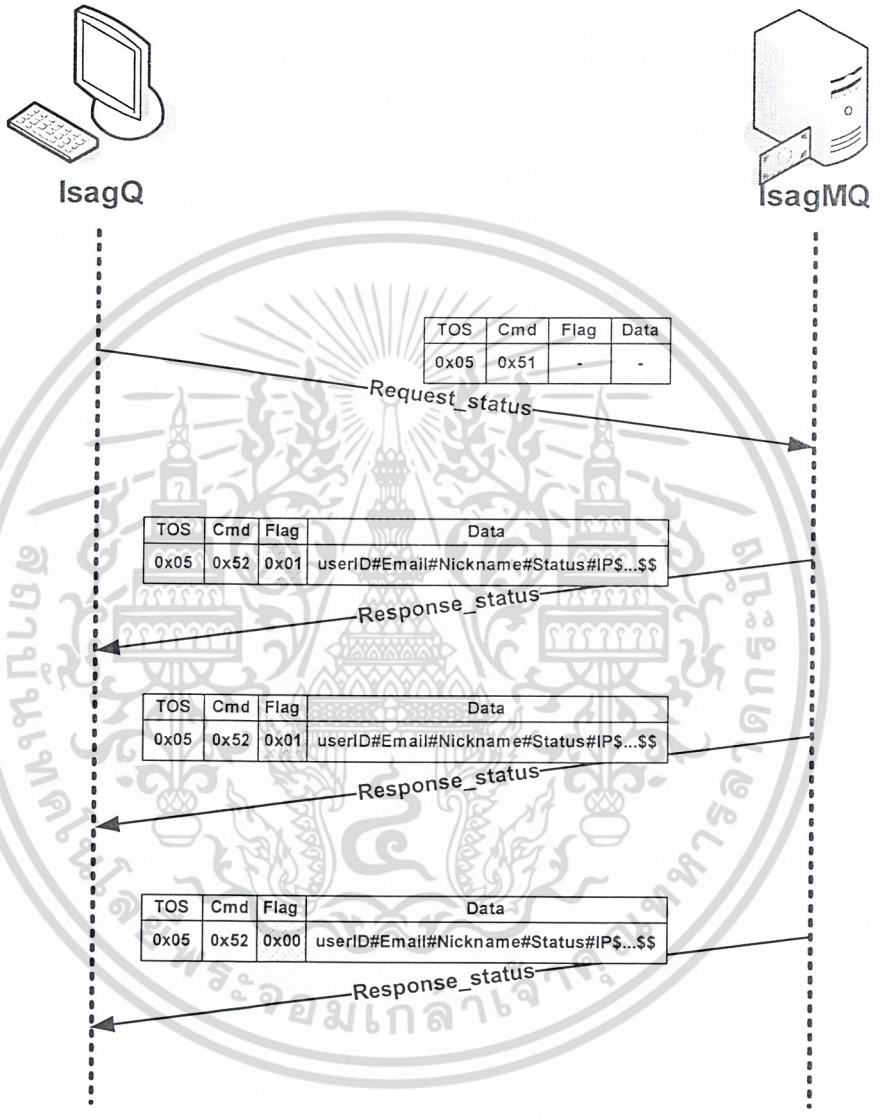
- ในขั้นตอนนี้โปรแกรมฝั่งผู้ใช้จะทำการตรวจสอบว่า ชื่อเล่นใหม่ ดังกล่าวถูกต้องตามเงื่อนไขที่โปรแกรมฝั่งแม่ข่ายต้องการหรือไม่ ก่อนที่จะอนุญาตบรรจุข้อความลงในช่อง Data ได้
- ความยาวของชื่อเล่นใหม่ จะต้องไม่เกิน 128 ตัวอักษร

2 โปรแกรมฝั่งแม่ข่ายรับ Frame ที่ผู้ใช้ส่งมาแล้วทำการถอดข้อความเพื่อเอา ชื่อเล่นใหม่ออกมา หลังจากนั้น จะมีลำดับขั้นตอนดังต่อไปนี้

- ตรวจสอบว่า ชื่อเล่นใหม่ ถูกต้องทุกประการตามที่โปรแกรมฝั่งแม่ข่ายต้องการ
- ทำการเปลี่ยนแปลงชื่อเล่นของผู้ใช้ในฐานข้อมูล ตาราง user โดยเปลี่ยนแปลง column NICK_NAME เป็นชื่อเล่นใหม่
- เมื่อเสร็จสิ้นขบวนการดังกล่าว โปรแกรมฝั่งแม่ข่ายจะทำการเปลี่ยนเฉพาะช่อง Cmd จะเป็น 0x34 ส่งกลับไปยังฝั่งผู้ใช้เป็นการสิ้นสุดการ เปลี่ยนชื่อเล่นอย่างสมบูรณ์ และ ถูกต้อง

3 เมื่อฝั่งผู้ใช้ได้รับ Frame ดังกล่าวจากโปรแกรมทางฝั่งแม่ข่ายแล้วจะทำการเปลี่ยนแปลงชื่อเล่นปัจจุบันให้เป็นดังชื่อเล่นที่ต้องการแล้ว แสดงให้ผู้ใช้เห็นว่าชื่อเล่นได้ถูกเปลี่ยนไปแล้ว

Status 0x05



รูปที่ 7-21 ขั้นตอนสำหรับการตรวจสอบสถานะ

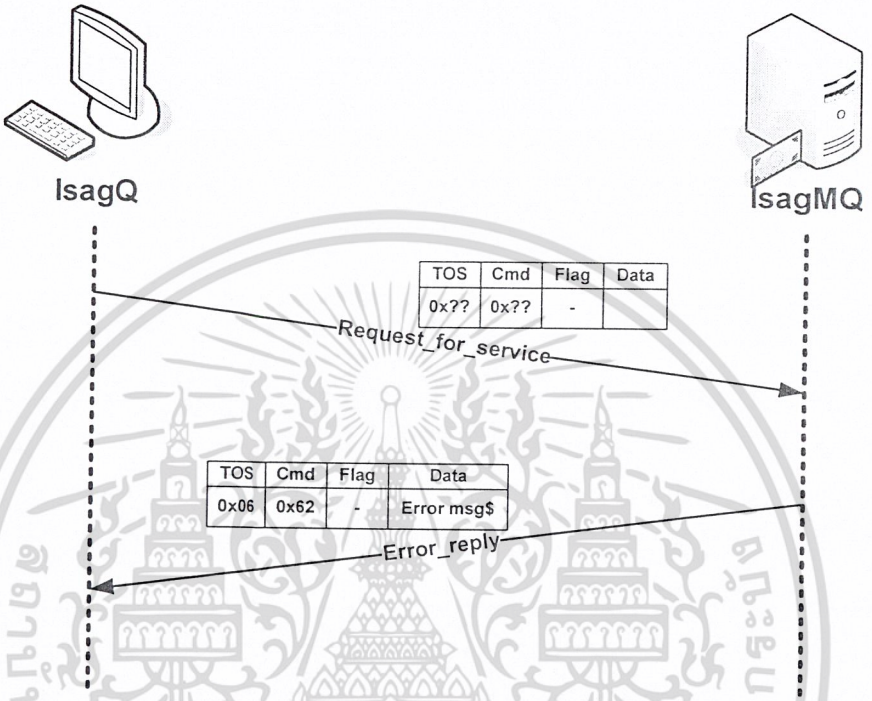
ขั้นตอนสำหรับการ ตรวจสอบสถานะบนบัญชีรายชื่อของผู้ใช้

(Status)

1. ในช่องของ TOS จะเป็น 0x05 และ Cmd จะเป็น 0x51 ส่งไปยังโปรแกรมแม่ข่าย
2. เมื่อได้รับ Frame ข้างต้นโปรแกรมฝั่งแม่ข่ายทำขบวนการดังต่อไปนี้
    1. ค้นหา User ID ของบัญชีรายชื่อทั้งหมดของผู้ใช้ จากฐานข้อมูล บนตาราง contact_list โดยจะเลือกเอาเฉพาะ row ที่ตรงตามเงื่อนไขดังต่อไปนี้คือ มี column USER_ID คือ ผู้ต้องการค้นหา และ column STATUS ไม่เป็น 'X' ผลลัพธ์จากการค้นหาจะได้ User ID และ สถานะ(Status) ปัจจุบันของบัญชีรายชื่อทั้งหมดของผู้ต้องการค้นหา
    2. ค้นหา IP Address จาก Data Structure Linklist โดยใช้ User ID ที่ได้จากการค้นหา และ พิจารณาร่วมกับ สถานะ พร้อมกันด้วย โดยจะมี IP Address จะปรากฏเฉพาะรายชื่อที่มีสถานะดังต่อไปนี้ 'O', 'B', 'D' และ 'A' ส่วนสถานะ 'W', 'X' และ 'F' จะไม่ปรากฏ IP Address
    3. ค้นหา อีเมลแอดเดรส (Email Address) และ ชื่อเล่น (Nickname) ของบัญชีรายชื่อของผู้ค้นหา จาก User ID ที่ได้มา โดยทำการค้นหาจาก ฐานข้อมูล ตาราง user ผลลัพธ์จากการค้นหาจะได้มาซึ่ง Email และ Nickname
    4. รวมรวมข้อมูลทั้งหมดที่ได้จากการค้นหา รวมกันเป็น กลุ่มข้อมูล ของ แต่ละรายชื่อ โดยลำดับ ของส่วนย่อย ภายใน แต่ละกลุ่มข้อมูล ของ แต่รายชื่อ จะถูกคั่นด้วยเครื่องหมาย '#' และ คั่นกลุ่มข้อมูล ของ แต่และรายชื่อ ด้วยเครื่องหมาย '$' และจบข้อมูลทั้งหมดของแต่ละ Frame ด้วยเครื่องหมาย '$$' โดยลำดับของส่วนย่อยของแต่ล่ะกลุ่มข้อมูลเป็นดังนี้

       UserID#Email#Nickname#Status#IP$
    5. โปรแกรมฝั่งแม่ข่ายจะเปลี่ยน Cmd จะเป็น 0x51 และ กำหนด Flag เป็น 0x01 ส่ง Frame ออกไปเรื่อยๆจน Frame สุดท้าย โปรแกรมฝั่งแม่ข่ายจะระบุ Flag เป็น 0x00 เพื่อแสดงว่าเป็นจุดสิ้นสุดของการส่ง Frame แล้ว
3. เมื่อผู้ใช้ได้รับ Frame ที่มี TOS เป็น 0x05 และ Cmd เป็น 0x51 ก็จะทำการนำ Dataใน Frame ดังกล่าวมาตีความเพื่อเก็บข้อมูลทั้ง 5 ชนิดของบัญชีรายชื่อของผู้ใช้แต่ละคน โดยจะทำการหยุดขบวนการนี้เมื่อ Flag เป็น 0x00

Error 0x06

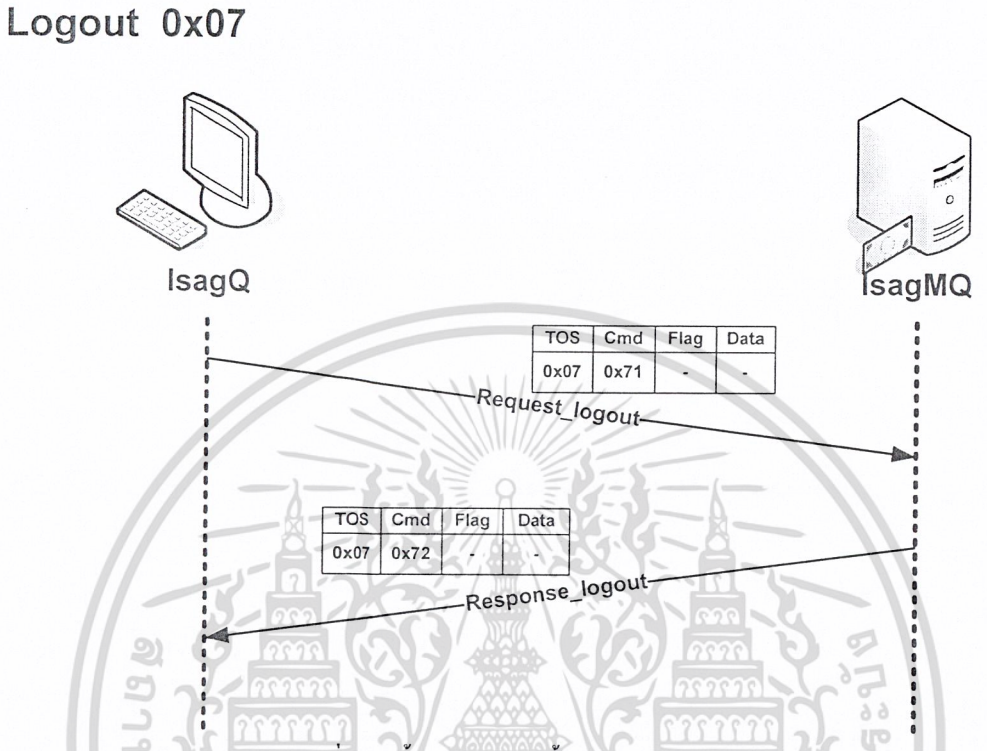


รูปที่ 7-22 ขั้นตอนสำหรับการรายงานความผิดพลาด

ขั้นตอนสำหรับการ รายงานความผิดพลาด

(Error)

1. ในช่องของ TOS และ Cmd จะเป็น บริการใดๆก็ได้ที่มีให้ ส่งไปยังโปรแกรมแม่ข่าย
2. เมื่อบริการใดๆนั้นเกิดความผิดพลาดอย่างใดอย่างหนึ่ง ก็จะรายงานความผิดพลาด โดยจะระบุในช่องของ TOS จะเป็น 0x06 และ Cmd จะเป็น 0x62 พร้อมทั้งรายงานความผิดพลาดลงในช่อง Data ส่งไปยังผู้ใช้
3. เมื่อโปรแกรมฝั่งผู้ใช้ได้รับ Frame รายงานความผิดพลาด ก็จะรายงานความผิดพลาดไปยังผู้ใช้



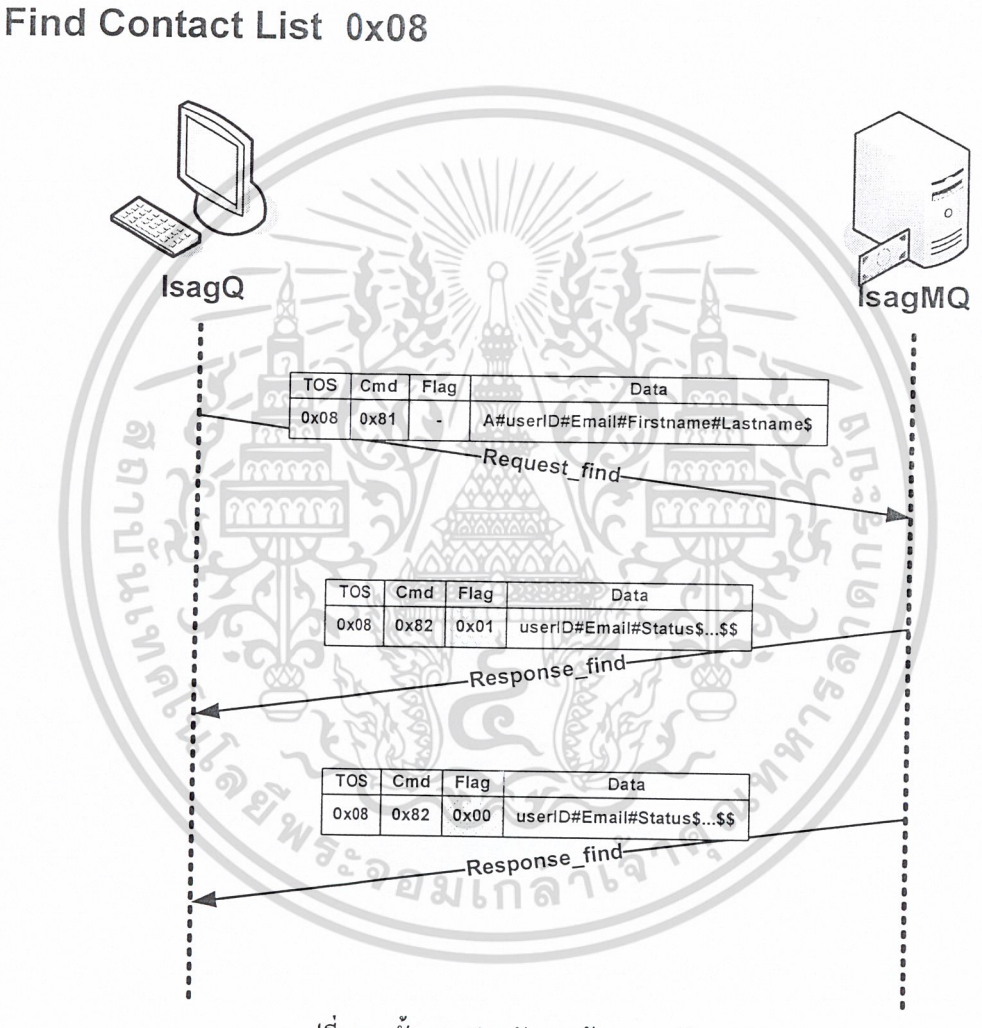
รูปที่ 7-23 ขั้นตอนสำหรับสิ้นสุดการใช้บริการ

ขั้นตอนสำหรับการ สิ้นสุดการใช้บริการ

(Logout)

1. ในช่องของ TOS จะเป็น 0x07 และ Cmd จะเป็น 0x71 ส่งไปยังโปรแกรมแม่ข่าย
2. เมื่อโปรแกรมฝั่งแม่ข่ายได้รับ Frame จะดำเนินขั้นตอนดังต่อไปนี้
   - 2.1 ทำค้นหารายชื่อผู้ที่มี User ID ของผู้ใช้ปรากฏอยู่ ในฐานข้อมูล ตาราง contact_list ที่มี column Contact_ID เป็น User_ID ของผู้ใช้ และ เปลี่ยน column Status เหล่านั้น ให้เป็น 'F' ทั้งหมด ยกเว้น ที่มี Status เป็น 'W', 'X' และ 'D'
   - 2.2 ทำการลบ Linklist ของ User ID ดังกล่าวออกไป เพื่อแสดงว่า User ID นั้นไม่ได้อยู่ในรายการผู้ที่ติดต่อมายังโปรแกรมฝั่งแม่ข่าย
   - 2.3 เมื่อเสร็จสิ้นขบวนการอย่างสมบูรณ์ ก็จะทำการเปลี่ยนเฉพาะ Cmd เป็น 0x72 และส่ง Frame กลับไป

3. เมื่อผู้ใช้ได้รับ Frame ที่มี TOS เป็น 0x07 และ Cmd เป็น 0x72 ก็จะทำการสิ้นสุดการติดต่อกับโปรแกรมแม่ข่ายโดยสมบูรณ์



รูปที่ 7-24 ขั้นตอนสำหรับการค้นหาสมาชิก

ขั้นตอนสำหรับการ ค้นหาสมาชิก

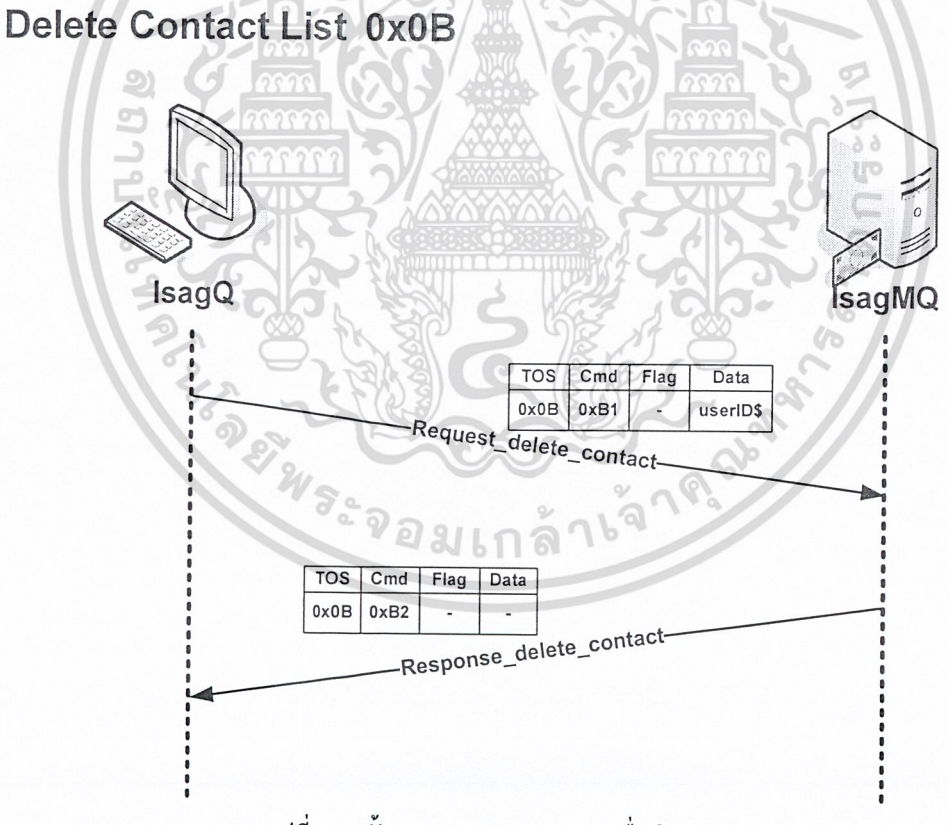
(Find Contact List)

1. ในช่องของ TOS จะเป็น 0x07 และ Cmd จะเป็น 0x71 พร้อมทั้งรายละเอียดข้อมูลของบุคคลที่ต้องการค้นหา ลงในช่อง Data ส่งไปยังโปรแกรมแม่ข่าย โดยรายละเอียดที่ผู้ใช้สามารถค้นหาได้มีดังต่อไปนี้
   - Operator And/Or สำหรับการเลือกว่าจะค้นหาแบบตรงตามรายละเอียดหรือ ค้นพาเพียงแค่มีบางส่วนของข้อมูลตรงกับความต้องการ โดยปกติเป็น And
   - User ID ต้องเป็น ตัวเลขไม่เกิน 10 โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น '_'
   - Email ต้องเป็นไปตามเงื่อนไข โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น '_'
   - First Name หรือ ชื่อ จริงต้องเป็นตัวอักษร a-z และ A-Z เท่านั้น มีความยาวไม่เกิน 64 ตัวอักษร โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น '_'
   - Last Name หรือ สกุล ต้องเป็นไปตามเงื่อนไขเช่นเดียวกับ First Name โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น '_'
2. เมื่อโปรแกรมฝั่งแม่ข่ายได้รับ Frame จะดำเนินขั้นตอนดังต่อไปนี้
   2.1 ทำแยกข้อมูลจากส่วน Data
   2.2 คำการตรวจสอบข้อมูลที่ถูกแยกออกมาว่าตรงตามเงื่อนไขของโปรแกรมฝั่งแม่ข่ายหรือไม่ มีขั้นตอนดังนี้
      - Operator ต้องเป็น A (And) หรือ O (R) เท่านั้น โดย Default แล้วเป็น And
      - User ID ต้องเป็น ตัวเลขไม่เกิน 10 โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น 0
      - Email ต้องเป็นไปตามเงื่อนไข โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น 0
      - First Name หรือ ชื่อ จริงต้องเป็นตัวอักษร a-z และ A-Z เท่านั้น มีความยาวไม่เกิน 64 ตัวอักษร โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น 0
      - Last Name หรือ สกุล ต้องเป็นไปตามเงื่อนไขเช่นเดียวกับ First Name โดยถ้าไม่มีข้อมูลส่วนนี้ค่าจะเป็น 0

2.3 ค้นหารายชื่อผู้ที่มีข้อมูลดังกล่าว ในฐานข้อมูล ตาราง user โดยหากพบตรงตามเงื่อนไขก็จะได้ผลลัพธ์คือ UserID , Email และ Status(สถานะปัจจุบัน) กลับมา

2.4 เมื่อเสร็จสิ้นขบวนการอย่างสมบูรณ์ ก็จะทำการเปลี่ยนเฉพาะ Cmd เป็น 0x82 รวม ถึง Flag เป็น 0x01 และใส่ผลลัพธ์คั่นด้วยเครื่องหมาย '#' และ '$' ตามลำดับ ลงในช่อง Data ส่ง Frame กลับไปยังผู้ใช้ จนกระทั่ง Frame สุดท้าย จะระบุลงใน Flag เป็น 0x00

3. เมื่อผู้ใช้ได้รับ Frame ที่มี TOS เป็น 0x08 และ Cmd เป็น 0x82 ก็จะทำการตีความข้อความในช่อง Data จนกว่า Flag จะเป็น 0x00 จึงถือเป็นอันเสร็จสิ้นโดยสมบูรณ์



รูปที่ 7-25 ขั้นตอนสำหรับการลบรายชื่อคู่สนทนา

ขั้นตอนสำหรับ การลบรายชื่อคู่สนทนา

(Delete Contact List)

1 ในช่องของ TOS จะเป็น 0x0B และ Cmd จะเป็น 0xB1 พร้อมทั้งระบุ User ID ลงในช่อง Data และปิดการสิ้นสุดของข้อมูลด้วย $ ส่งไปยังโปรแกรมแม่ข่าย

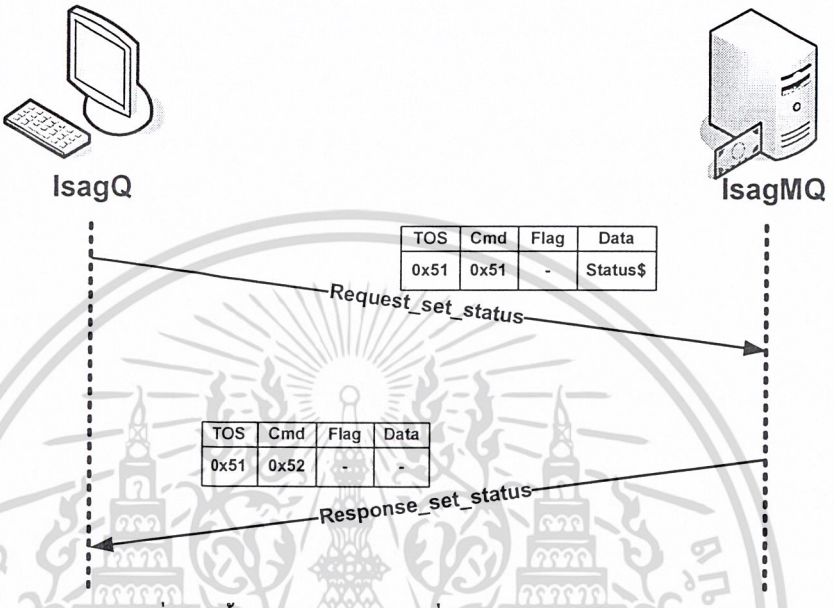
- ในขั้นตอนนี้โปรแกรมฝั่งผู้ใช้จะทำการตรวจสอบว่า User ID ดังกล่าวถูกต้องตามเงื่อนไขที่โปรแกรมฝั่งแม่ข่ายต้องการหรือไม่ ก่อนที่จะอนุญาตบรรจุข้อความลงในช่อง Data ได้
- User ID มีความยาวไม่เกิน 10 ตัวอักษร

2 โปรแกรมฝั่งแม่ข่ายรับ Frame ที่ผู้ใช้ส่งมาแล้วทำการถอดข้อความเพื่อเอา User ID ออกมา หลังจากนั้น จะมีลำดับขั้นตอนดังต่อไปนี้

- ตรวจสอบว่า User ID ถูกต้องทุกประการตามที่โปรแกรมฝั่งแม่ข่ายต้องการ
- ทำการค้นหาว่า User ID ดังกล่าวมีอยู่จริงในฐานข้อมูล ตาราง user
- ทำการลบ User ID คนดังกล่าวออกจากบัญชีรายชื่อของผู้ใช้ ในฐานข้อมูล ตาราง contact_list โดย มีเงื่อนไขว่า column USER_ID จะเป็น User ID ของผู้ที่ต้องการจะลบ และ CONTACT_ID จะต้องเป็น User ID คนดังกล่าวที่ผู้ใช้ต้องการจะลบ
- เมื่อเสร็จสิ้นขบวนการดังกล่าว โปรแกรมฝั่งแม่ข่ายจะทำการเปลี่ยนเฉพาะช่อง Cmd จะเป็น 0xB2 ส่งกลับไปยังฝั่งผู้ใช้เป็นการสิ้นสุดการลบรายชื่อคู่สนทนา

3 โปรแกรมฝั่งผู้ใช้รับ Frame ที่ TOS เป็น 0x0B และ Cmd เป็น 0xB2 แสดงว่า เสร็จสิ้นการลบรายชื่อคู่สนทนาโดยสมบูรณ์

Set Status 0x51



รูปที่ 7-26 ขั้นตอนสำหรับการเปลี่ยนสถานะของผู้ใช้

ขั้นตอนสำหรับ การเปลี่ยนสถานะของผู้ใช้

(Set Status)

1 ในช่องของ TOS จะเป็น 0x51 และ Cmd จะเป็น 0x51 พร้อมทั้งระบุสถานะใหม่ของผู้ใช้ลงในช่อง Data และปิดการสิ้นสุดของข้อมูลด้วย $ ส่งไปยังโปรแกรมแม่ข่าย

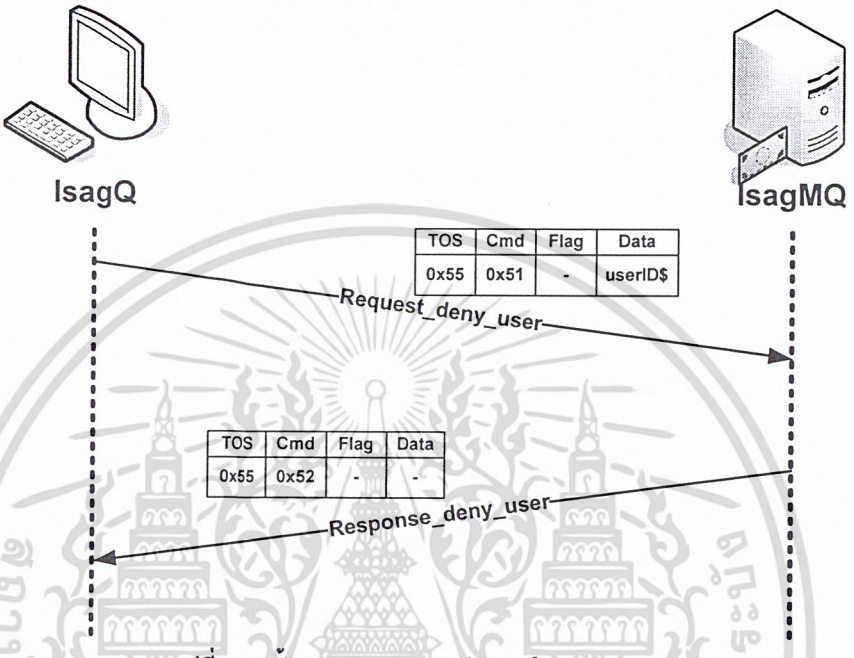
- ในขั้นตอนนี้โปรแกรมฝั่งผู้ใช้จะทำการตรวจสอบว่า สถานะใหม่ ดังกล่าวถูกต้องตามเงื่อนไขที่โปรแกรมฝั่งแม่ข่ายต้องการหรือไม่ ก่อนที่จะอนุญาตบรรจุข้อความลงในช่อง Data ได้
- สถานะที่ผู้ใช้สามารถเปลี่ยนแปลงได้มีดังนี้
  - ❖ 'A' - Away ผู้ใช้ขณะนี้ Online ไม่อยู่
  - ❖ 'B' - Busy ผู้ใช้ยณะนี้ Online อยู่แต่ไม่ว่าง
  - ❖ 'O' - Online ผู้ใช้ขณะนี้พร้อมที่จะทำการสนทนา
- ความยาวของสถานะที่จะส่งไปคือ 1 ตัวอักษรตามข้างต้น

2 โปรแกรมฝั่งแม่ข่ายรับ Frame ที่ผู้ใช้ส่งมาแล้วทำการถอดข้อความเพื่อเอา สถานะใหม่ ออกมา หลังจากนั้น จะมีลำดับขั้นตอนดังต่อไปนี้

- ตรวจสอบว่า สถานะใหม่ ถูกต้องทุกประการตามที่โปรแกรมฝั่งแม่ข่ายต้องการ คือมีแค่ 'A' , 'B' และ 'O'
- ทำการเปลี่ยนแปลงสถานะของผู้ใช้ ในฐานข้อมูล ตาราง contact_list ตามเงื่อนไข นี้คือ CONTACT_ID คือ User ID ของผู้ใช้ และ STATUS เหล่านั้นไม่ใช่ 'D' , 'X' , 'W' โดยเปลี่ยนแปลงเฉพาะ column STATUS ของทุก row ที่ตรงตามเงื่อนไข
- เมื่อเสร็จสิ้นขบวนการดังกล่าว โปรแกรมฝั่งแม่ข่ายจะทำการเปลี่ยนเฉพาะช่อง Cmd จะเป็น 0x52 ส่งกลับไปยังฝั่งผู้ใช้เป็นการสิ้นสุดการ เปลี่ยนสถานะอย่างสมบูรณ์ และ ถูกต้อง

3 เมื่อโปรแกรมฝั่งผู้ใช้ได้รับ Frame ดังกล่าวจากโปรแกรมทางฝั่งแม่ข่ายแล้วจะทำการเปลี่ยนแปลงสถานะปัจจุบันให้เห็น

Deny 0x55



รูปที่ 7-27 ขั้นตอนสำหรับการปฏิเสธคู่สนทนา

ขั้นตอนสำหรับการ การปฏิเสธคู่สนทนา

(Deny)

1 ในช่องของ TOS จะเป็น 0x55 และ Cmd จะเป็น 0x51 พร้อมทั้งระบุ User ID ที่ต้องการปฏิเสธ ลงในช่อง Data และปิดการสิ้นสุดของข้อมูลด้วย $ ส่งไปยังโปรแกรมแม่ข่าย

- ในขั้นตอนนี้โปรแกรมฝั่งผู้ใช้จะทำการตรวจสอบว่า User ID ที่ต้องการปฏิเสธดังกล่าวถูกต้องตามเงื่อนไขที่โปรแกรมฝั่งแม่ข่ายต้องการหรือไม่ ก่อนที่จะอนุญาตบรรจุข้อความลงในช่อง Data ได้
- ความยาวของ User ID ที่ต้องการปฏิเสธ ที่จะส่งไปไม่เกิน 10 ตัวอักษรตามข้างต้น

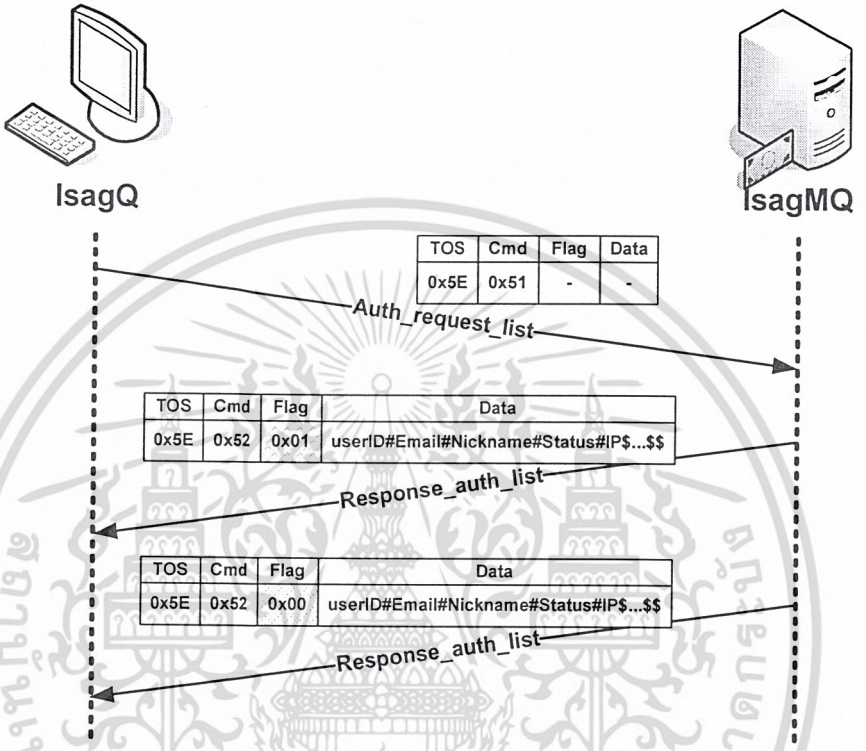
2 โปรแกรมฝั่งแม่ข่ายรับ Frame ที่ผู้ใช้ส่งมาแล้วทำการถอดข้อความเพื่อเอา User ID ที่ต้องการปฏิเสธ ออกมา หลังจากนั้น จะมีลำดับขั้นตอนดังต่อไปนี้

- ตรวจสอบว่า User ID ดังกล่าว อยู่บน บัญชีรายชื่อผู้ที่ต้องการปฏิเสธจริง

- ทำการเปลี่ยนแปลง row ของผู้ปฏิเสธ ในฐานข้อมูล ตาราง contact_list ตามเงื่อนไขนี้คือ column USER_ID คือ ผู้ที่ปฏิเสธ และ column CONTACT_ID คือ User ID ของผู้ถูกปฏิเสธ โดยเปลี่ยนแปลงเฉพาะ column STATUS เป็น 'D'
- ทำการเปลี่ยนแปลง row ของผู้ถูกปฏิเสธ ในฐานข้อมูล ตาราง contact_list ตามเงื่อนไขนี้คือ column USER_ID คือ ผู้ถูกปฏิเสธ และ column CONTACT_ID คือ User ID ของผู้ปฏิเสธ โดยเปลี่ยนแปลงเฉพาะ column STATUS เป็น 'X'
- เมื่อเสร็จสิ้นขบวนการดังกล่าว โปรแกรมฝั่งแม่ข่ายจะทำการเปลี่ยนเฉพาะช่อง Cmd จะเป็น 0x52 ส่งกลับไปยังฝั่งผู้ใช้เป็นการสิ้นสุดการ เปลี่ยนสถานะอย่างสมบูรณ์ และ ถูกต้อง

3 เมื่อฝั่งโปรแกรมฝั่งผู้ใช้ได้รับ Frame ดังกล่าวแล้ว แสดงว่าเสร็จสิ้นการปฏิเสธคู่สนทนาอย่างสมบูรณ์

Authorized Request 0x5E



รูปที่ 7-28 ขั้นตอนสำหรับการตรวจสอบบัญชีรายชื่อที่รอคำยินยอม

ขั้นตอนสำหรับ การตรวจสอบบัญชีรายชื่อที่รอคำยินยอมจากผู้ใช้

(Authorized Request)

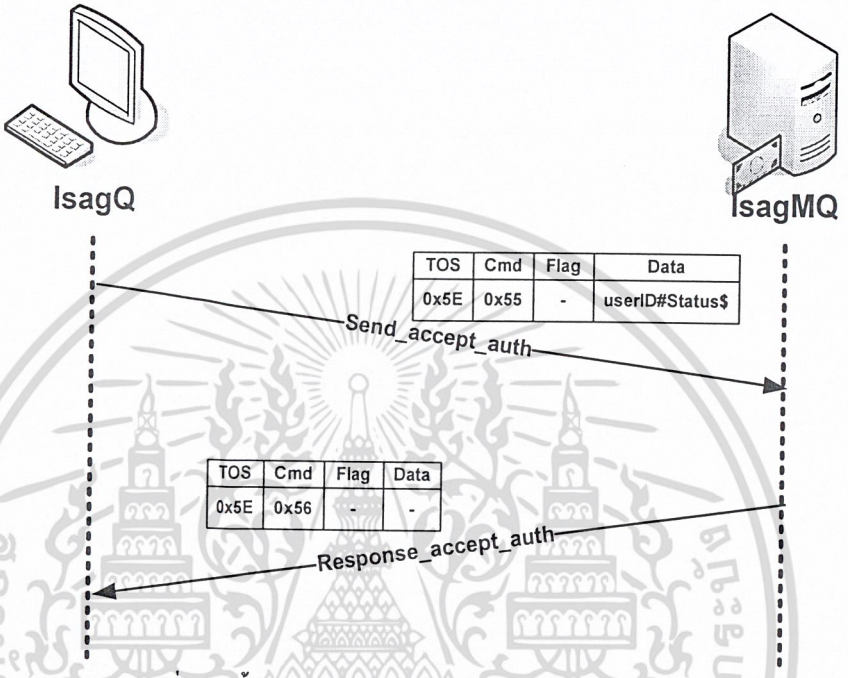
1. ในช่องของ TOS จะเป็น 0x5E และ Cmd จะเป็น 0x51 ส่งไปยังโปรแกรมแม่ข่าย
2. เมื่อได้รับ Frame ข้างต้นโปรแกรมฝั่งแม่ข่ายทำขบวนการดังต่อไปนี้
    1. ค้นหา User ID ของบัญชีรายชื่อทั้งหมดของผู้ใช้ จากฐานข้อมูล บนตาราง contact_list โดยจะเลือกเอาเฉพาะ row ตรงตามเงื่อนไขดังนี้คือ column CONTACT_ID คือ User ID ของผู้ใช้ และ column STATUS เป็น 'W' ผลลัพธ์จากการค้นหาจะได้ User ID หรือ คือผู้ที่กำลังรอคำยินยอมจากผู้ใช้นั้นเอง

2. ค้นหา IP Address จาก Data Structure Linklist โดยใช้ User ID ที่ได้จากค้นหา โดยจะมี IP Address ปรากฏเฉพาะ User ID ที่อยู่ใน Linklist เท่านั้น
3. ค้นหา อีเมลแอดเดรส(Email Address) และ ชื่อเล่น(Nickname) ของบัญชีรายชื่อของผู้ใช้ จาก User ID ที่ได้มา โดยทำการค้นหาจาก ฐานข้อมูล ตาราง user ผลลัพธ์จากการค้นหาจะได้มาซึ่ง Email และ Nickname
4. รวมรวมข้อมูลทั้งหมดที่ได้จากการค้นหา รวมกันเป็นกลุ่มข้อมูลของแต่ละรายชื่อ โดยมีลำดับของส่วนย่อยของแต่ละกลุ่มข้อมูลของแต่รายชื่อจะถูกคั่นด้วยเครื่องหมาย '#' และ คั่นกลุ่มข้อมูลของแต่และรายชื่อด้วยเครื่องหมาย '$' และจบข้อมูลทั้งหมดของแต่ละ Frame ด้วยเครื่องหมาย '$$' โดยลำดับของส่วนย่อยของแต่ล่ะกลุ่มข้อมูลเป็นดังนี้

   UserID#Email#Nickname#Status#IP$
5. โปรแกรมฝั่งแม่ข่ายจะเปลี่ยน Cmd จะเป็น 0x52 และ กำหนด Flag เป็น 0x01 ส่ง Frame ออกไปเรื่อยๆจน Frame สุดท้าย โปรแกรมฝั่งแม่ข่ายจะระบุ Flag เป็น 0x00 เพื่อแสดงว่าเป็นจุดสิ้นสุดของการส่ง Frame แล้ว

3. เมื่อโปรแกรมฝั่งผู้ใช้ได้รับ Frame ที่มี TOS เป็น 0x5E และ Cmd เป็น 0x52 ก็จะทำการนำ Dataใน Frame ดังกล่าวมาตีความเพื่อเก็บข้อมูลทั้ง 5 ชนิดของบัญชีรายชื่อของผู้ใช้แต่ละคน โดยจะทำการหยุดขบวนการนี้เมื่อ Flag เป็น 0x00 พร้อมทั้งแสดงให้ผู้ใช้เห็นว่ามี ผู้ใช้คนใดรอคำยินยอมจากเราอยู่บ้าง

Accept Authorize 0x5E



รูปที่ 7-29 ขั้นตอนสำหรับการยินยอมตามคำร้องขอ

ขั้นตอนสำหรับการยินยอมตามคำร้องขอ

(Accept Authorize)

1. ในช่องของ TOS จะเป็น 0x5E และ Cmd จะเป็น 0x55 พร้อมทั้งระบุ User ID ที่ต้องการยินยอม และ สถานะปัจจุบันของผู้ยินยอม (Status) ลงในช่อง Data และปิดการสิ้นสุดของข้อมูลด้วย $ ส่งไปยังโปรแกรมแม่ข่าย
   - ในขั้นตอนนี้โปรแกรมฝั่งผู้ใช้จะทำการตรวจสอบว่า User ID ที่ต้องการยินยอม และ สถานะปัจจุบันของผู้ยินยอม ว่าถูกต้องตามเงื่อนไขที่โปรแกรมฝั่งแม่ข่ายต้องการหรือไม่ ก่อนที่จะอนุญาตบรรจุข้อความลงในช่อง Data ได้
   - ความยาวของ User ID และ สถานะปัจจุบันของผู้ยินยอม ที่จะส่งไปไม่เกิน 10 ตัวอักษร และ 1 ตัวอักษรตามลำดับ
   - ตัวอักษรสำหรับสถานะปัจจุบันของผู้ยินยอมมีเงื่อนไขเดียวกับที่กล่าวไว้ข้างต้น

2. เมื่อได้รับ Frame ข้างต้นโปรแกรมฝั่งแม่ข่ายทำขบวนการดังต่อไปนี้
   - ค้นหา สถานะปัจจุบันของ User ID ที่ได้รับการยินยอม จากฐานข้อมูล บนตาราง contact_list
   - ค้นหา row จากฐานข้อมูล บนตาราง contact_list ที่ตรงตามเงื่อนไขดังนี้คือ column USER_ID คือ ผู้ที่ได้รับยินยอม , column CONTACT_ID คือ User ID ของผู้ยินยอม และ column STATUS เป็น 'W' และ เปลี่ยนแปลง column ดังกล่าวเป็น สถานะปัจจุบันของผู้ยินยอม
   - เพิ่มชื่อที่ผู้ที่ได้รับการยินยอมลงในบัญชีรายชื่อของผู้ยินยอม บนตาราง contact_list โดย column STATUS ของ row ที่เพิ่มเป็น สถานะปัจจุบันที่หได้จากข้างต้น
   - เมื่อเสร็จสิ้นกระบวนการทั้งหมด โปรแกรมฝั่งแม่ข่ายจะเปลี่ยน Cmd จะเป็น 0x56 ส่งกลับไปยังผู้ใช้
3. เมื่อผู้ใช้ได้รับ Frame ที่มี TOS เป็น 0x5E และ Cmd เป็น 0x56 ก็แสดงว่าตอนนี้ผู้ใช้มีรายชื่อคู่สนทนาเพิ่มในบัญชีแล้ว

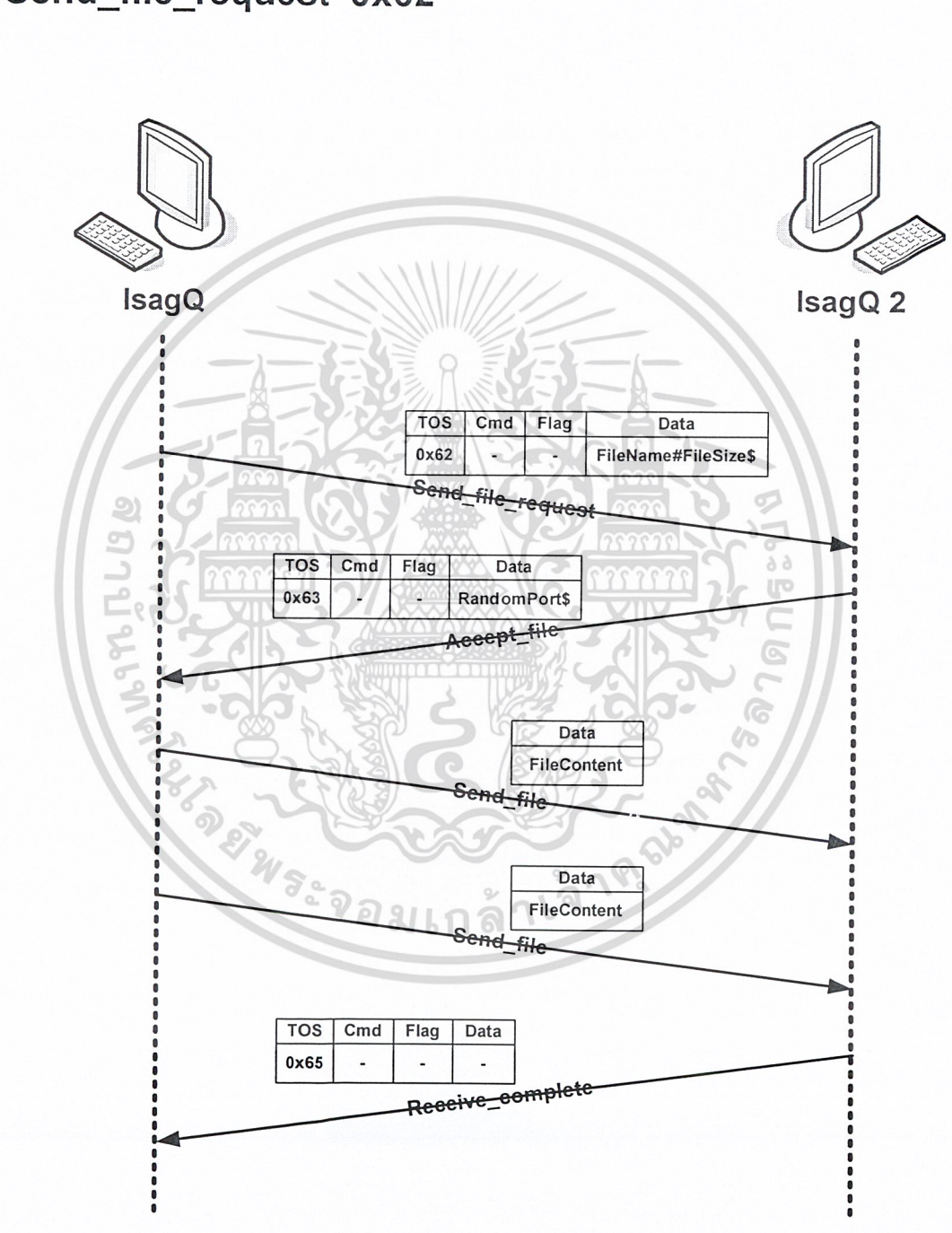
Send_msg 0x61

รูปที่ 7-30 ขั้นตอนสำหรับการส่งข้อความระหว่าง IsagQ ด้วยกัน

การส่งข้อความระหว่าง IsagQ

(Send_msg)

ในช่องของ TOS จะเป็น 0x61 พร้อมกับแนบข้อความ Msg และ ปิดท้ายด้วย $ ส่งไปยังผู้ที่สนทนาด้วยในขณะนั้น เมื่อ ผู้รับได้รับข้อความก็จะทำการตรวจสอบ TOS ว่าเป็น 0x61 หรือไม่ ซึ่งถ้าเป็นไปตามเงื่อนไขแล้วก็จะทำการดึง Msg ออกมาจาก เพ็กเกต พร้อมทั้งแสดงผลออกทางหน้าต่างสนทนา



รูปที่ 7-31 ขั้นตอนสำหรับการส่งไฟล์ระหว่าง IsagQ ด้วยกัน

การส่งไฟล์ระหว่าง IsagQ

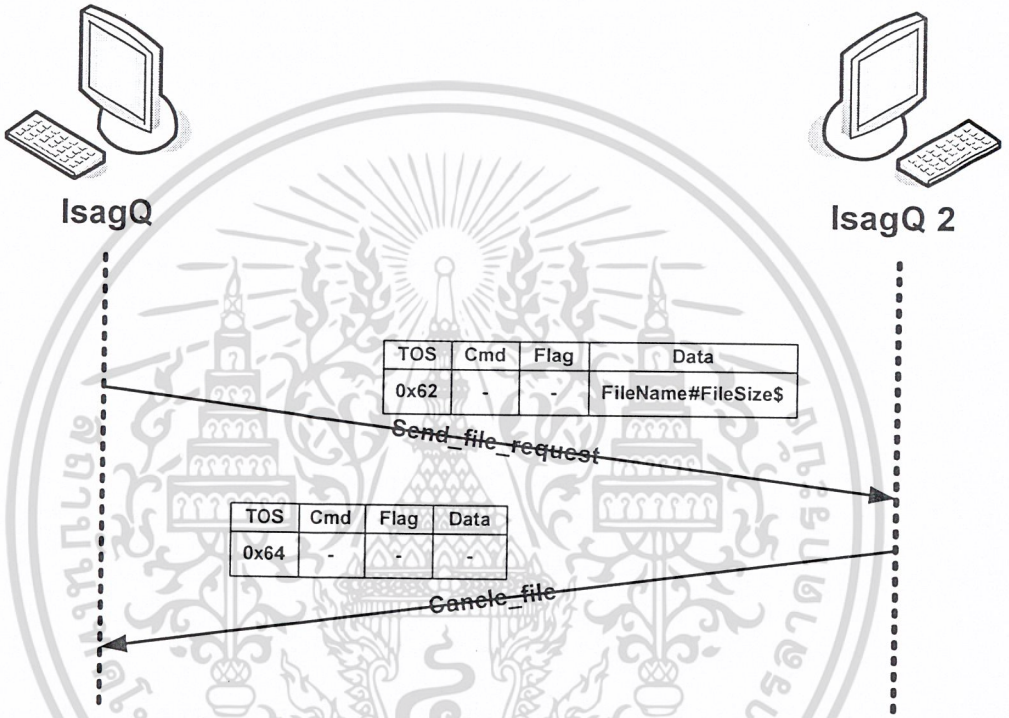
(Send_file)

1. ผู้ส่งจำเป็นต้องส่งการร้องขอไปยังผู้รับก่อนที่จะมีการส่งไฟล์ โดยการกำหนด TOS เป็น 0x62 เพื่อส่ง Send_file_request โดยในส่วนของ Data จะต้องใส่ชื่อไฟล์ และขนาดของไฟล์ที่ต้องการส่งไปยังพร้อมกับเพ็กเกตดังกล่าวด้วย เพื่อให้ผู้รับได้ทราบชื่อ , ชนิด และ ขนาดของไฟล์ เพื่อที่จะได้จองพื้นที่สำรองไว้
2. ในกรณีดังรูป ผู้รับ ประสงค์ที่จะรับไฟล์ดังกล่าว ก็จะต้องส่งข้อความแสดงการยอมรับไฟล์ดังกล่าวไปยังผู้ส่ง โดยการกำหนด TOS เป็น 0x63 พร้อมทั้งแนบหมายเลขพอร์ตที่ถูกสุ่มขึ้นมาซึ่งมีค่ามากกว่า 16002 แต่ไม่เกิน 65535 ส่งไปยังผู้ส่งเพื่อที่จะได้ส่งไฟล์ไปยังปลายทางได้ถูกต้อง
3. เมื่อผู้ส่งได้รับเพ็กเกตที่มี TOS เป็น 0x63 แล้ว ก็จะทำการนำค่าหมายเลขพอร์ตดังกล่าว พร้อมทั้ง IP ของผู้รับซึ่งมีอยู่ในระบบอยู่แล้ว ติดต่อไปยังผู้รับเพื่อทำการส่งไฟล์ โดยการนี้จะส่งไฟล์โดยใช้ SSL ด้วย
4. เมื่อผู้รับได้รับไฟล์ทั้งหมดแบบสมบูรณ์ จะส่ง Receive_complete โดยมี TOS เป็น 0x65 เพื่อบอกไปยังผู้ส่งว่าตนเองได้รับไฟล์ครบถ้วนสมบูรณ์
5. เมื่อผู้ส่งได้รับ Receive_complete ก็จะทำการแสดงข้อความสิ้นสุดการส่งไฟล์อย่างถูกต้องสมบูรณ์ให้ผู้ส่งได้รับทราบ

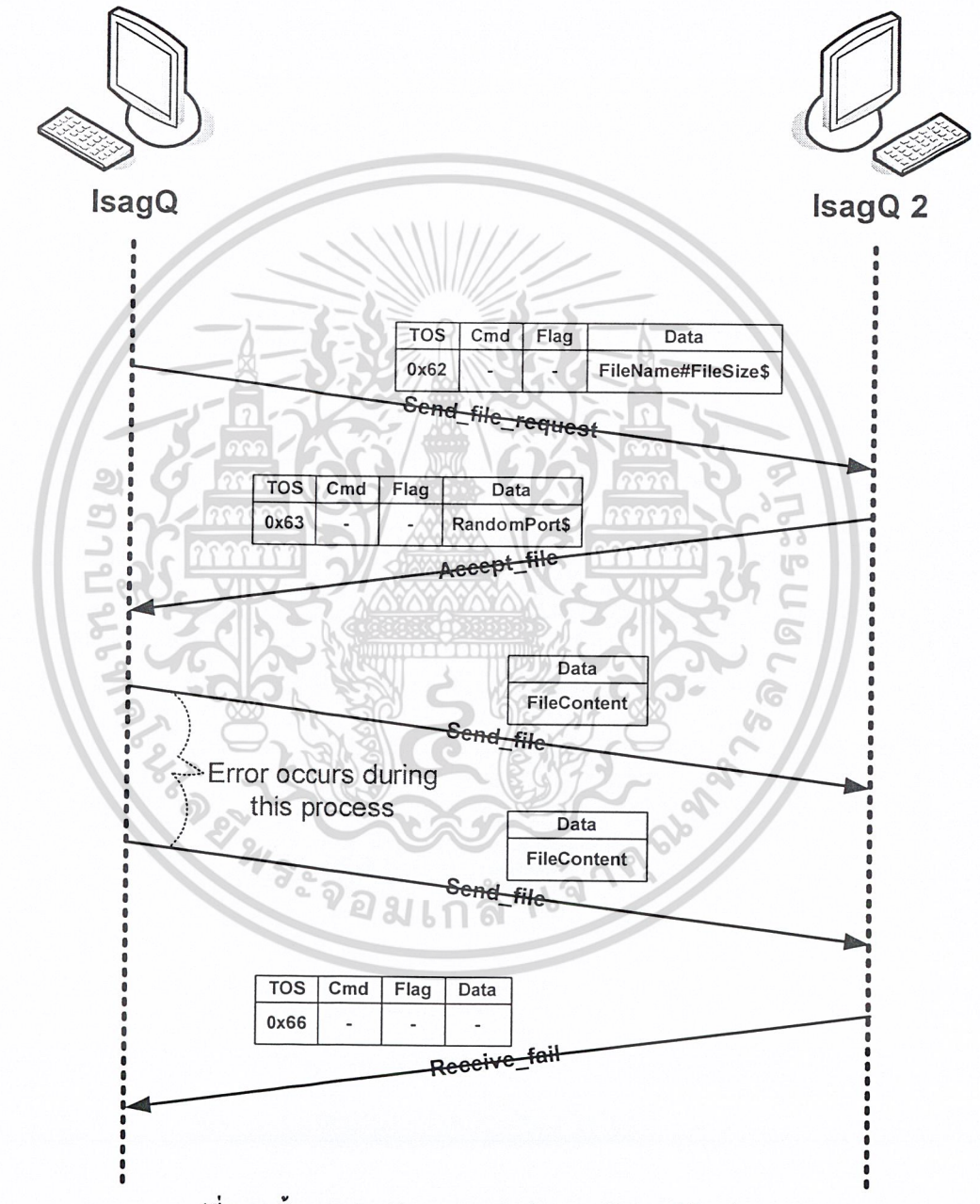
กรณีเพิ่มเติม

1. กรณีผู้รับไม่ประสงค์ที่จะรับไฟล์ดังกล่าวจะทำการส่ง Cancle_file โดยกำหนด TOS เป็น 0x64 ไปยังผู้ส่งเพื่อปฏิเสธการรับไฟล์ ซึ่งเมื่อผู้ส่งได้รับข้อความดังกล่าวก็จะสิ้นสุดการส่งไฟล์ทันที ดังรูปที่ 7-32
2. กรณีเกิดความผิดพลาดระหว่างการส่งไฟล์จนทำให้ไฟล์ที่ได้รับไม่ครบถ้วนสมบูรณ์ ผู้รับก็จะส่ง Receive_fail โดยการกำหนด TOS เป็น 0x66 ไปยังผู้ส่ง ซึ่งเมื่อผู้ส่งได้รับข้อความดังกล่าวแล้ว ก็จะแสดงผลให้ผู้ใช้ทราบเพื่อทำการส่งไฟล์ใหม่ ดังรูปที่ 7-33

Cancel_file 0x64



รูปที่ 7-32 ขั้นตอนสำหรับปฏิเสธการรับไฟล์



รูปที่ 7-33 ขั้นตอนสำหรับการแสดงผลผิดพลาดในการส่งไฟล์

บทที่ 8

การทดสอบ ISAGMQ และ ISAGQ

8.1 การติดต่อระหว่าง ISAGMQ กับ ISAGQ

ในช่วงที่มีการสถาปนาการเชื่อมต่อระหว่าง IsagMQ และ IsagQ โดยใช้โพรโตคอล SSL และใช้โปรแกรม Ether Real ดักจับPacket ที่วิ่งอยู่บนระบบ Network จะเห็นว่า ทั้งสองฝั่งได้มีการแลกเปลี่ยนคีย์ รวมถึงใบรับรองสิทธิ์ ซึ่งเป็นไปตามข้อตกลงของโครงงาน ดังรูป 8-1

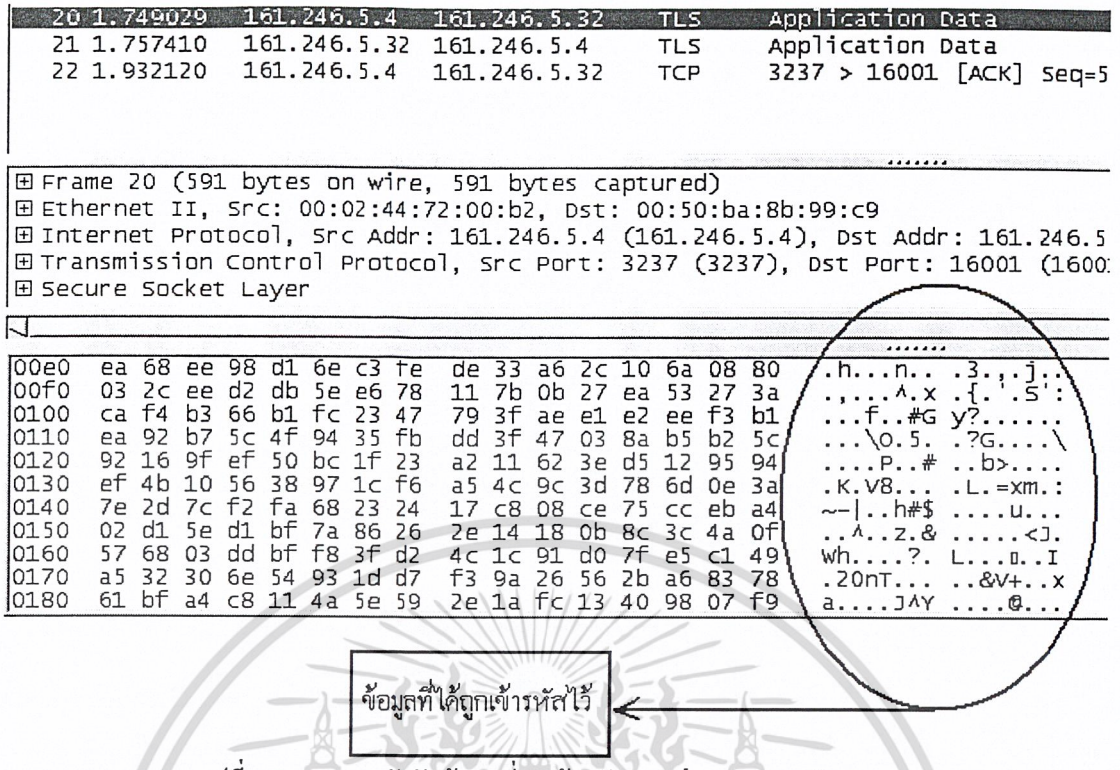
<capture> - Ethereal

e Edit Capture Display Tools

.	Time	Source	Destination	Protocol	Info
1	0.000000	161.246.5.4	161.246.5.32	TCP	3237 > 16001 [SYN] Seq=575914295 Ack=0 Win=65535 Len=0
2	0.000229	161.246.5.32	161.246.5.4	TCP	16001 > 3237 [SYN, ACK] Seq=2958061825 Ack=575914296 Win=65535 Len=0
3	0.000323	161.246.5.4	161.246.5.32	TCP	3237 > 16001 [ACK] Seq=575914296 Ack=2958061826 Win=65535 Len=0
4	0.055228	161.246.5.4	161.246.5.32	SSLv2	Client Hello
5	0.056508	161.246.5.32	161.246.5.4	TLS	Server Hello, Certificate, [Unreassembled Packet]
6	0.056584	161.246.5.32	161.246.5.4	TLS	Continuation Data, [Unreassembled Packet]
7	0.056638	161.246.5.4	161.246.5.32	TCP	3237 > 16001 [ACK] Seq=575914375 Ack=2958064145 Win=65535 Len=0
8	0.132384	161.246.5.4	161.246.5.32	TLS	Certificate, Client Key Exchange, [Unreassembled Packet]
9	0.132430	161.246.5.4	161.246.5.32	TLS	Continuation Data, [Unreassembled Packet]
10	0.133110	161.246.5.32	161.246.5.4	TCP	16001 > 3237 [ACK] Seq=2958064145 Ack=575915841 Win=64234 Len=0
11	0.203411	161.246.5.4	161.246.5.32	TLS	Certificate Verify
12	0.298278	161.246.5.32	161.246.5.4	TCP	16001 > 3237 [ACK] Seq=2958064145 Ack=575915980 Win=65535 Len=0
13	0.298359	161.246.5.4	161.246.5.32	TLS	Change Cipher Spec, Finished
14	0.299099	161.246.5.32	161.246.5.4	TLS	Change Cipher Spec, Encrypted Handshake Message
15	0.363329	161.246.5.4	161.246.5.32	TLS	Application Data

รูปที่ 8-1 แสดงการดักจับการสถาปนาการเชื่อมต่อ ด้วย SSL ระหว่าง IsagMQ กับ IsagQ

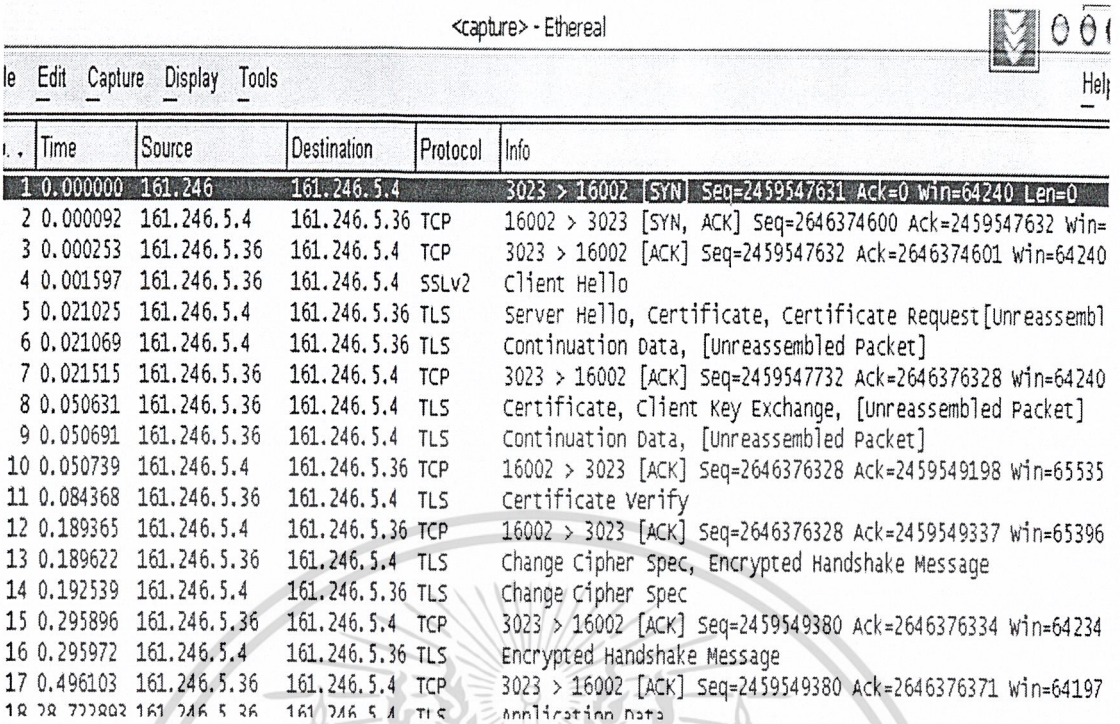
จากรูป 8-2 เมื่อมีการสถาปนาการเชื่อมต่อด้วยโพรโตคอล SSL แล้ว ข้อมูลที่ส่งผ่านระหว่างกันจะถูกเข้ารหัส ทำให้มั่นใจได้ว่าข้อมูลที่ส่งผ่านนั้นไม่ถูกเปิดเผยต่อบุคคลอื่น



รูปที่ 8-2 แสดงการดักจับข้อมูลที่ถูกเข้ารหัส ระหว่าง IsagMQ กับ IsagQ

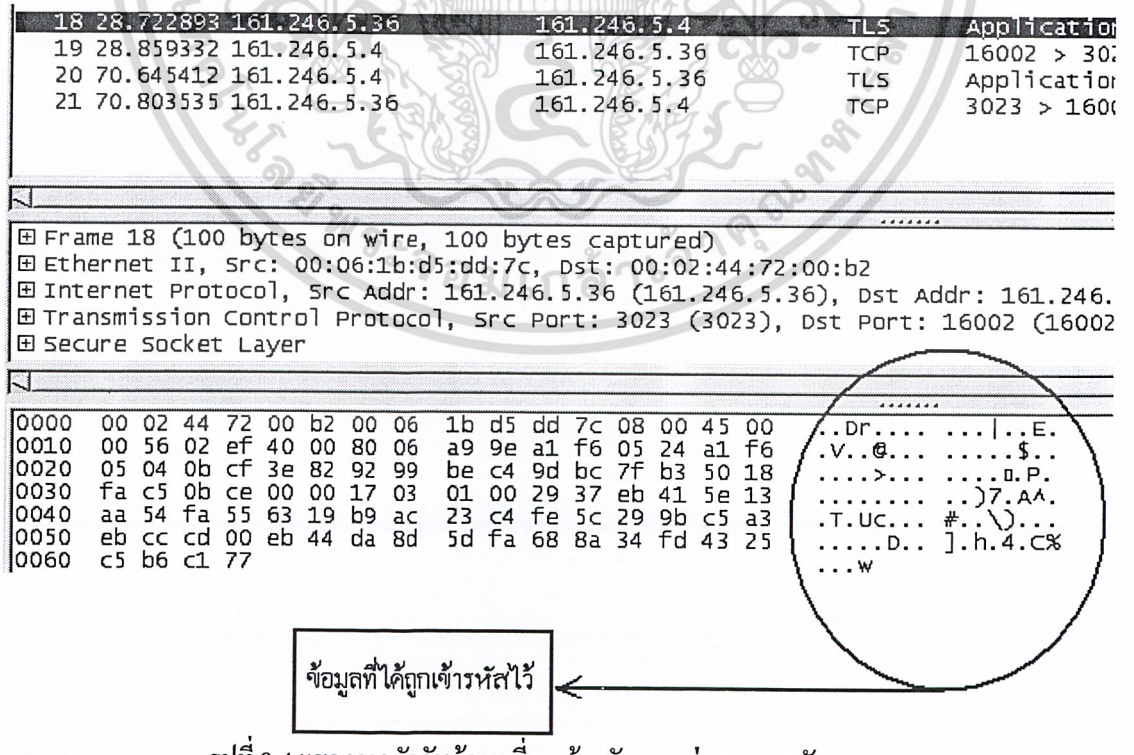
การติดต่อระหว่าง ISAGQ กับ ISAGQ

ในช่วงที่มีการสถาปนาการเชื่อมต่อระหว่าง IsagQ และ IsagQ โดยใช้โพรโตคอล SSL และใช้โปรแกรม Ether Real ดักจับPacket ที่วิ่งอยู่บนระบบ Network จะเห็นว่า ทั้งสองฝั่งได้มีการแลกเปลี่ยนคีย์ รวมถึงใบรับรองสิทธิ์ ซึ่งเป็นไปตามข้อตกลงของโครงงาน ดังรูป 8-3



รูปที่ 8-3 การดักจับการสถาปนาการเชื่อมต่อ ด้วย SSL ระหว่าง IsagMQ กับ IsagQ

จากรูป 8-4 เมื่อมีการสถาปนาการเชื่อมต่อด้วยโพรโตคอล SSL แล้ว ข้อมูลที่ส่งผ่านระหว่างกันจะถูกเข้ารหัส ทำให้มั่นใจได้ว่าข้อมูลที่ส่งผ่านนั้นไม่ถูกเปิดเผยต่อบุคคลอื่น



รูปที่ 8-4 แสดงการดักจับข้อมูลที่ถูกเข้ารหัส ระหว่าง IsagQ กับ IsagQ

รูปที่ 8-5 ตารางเปรียบเทียบความสามารถ

ความสามารถ	IsagMQ & IsagQ	ICQ	MSN
การเข้ารหัสข้อความ	มี	ไม่มี	ไม่มี
การพิสูจน์ตน	มี	ไม่มี	ไม่มี
ปัญหาเรื่องไฟร์วอลล์	มี	มี	มีน้อย
ลูกเล่น	น้อย	มาก	มาก
ความง่ายและยืดหยุ่นในการใช้งาน	น้อย	มาก	มาก
ความรวดเร็วในการส่งข้อความ	ปานกลาง	มาก	ปานกลาง
การส่งไฟล์อย่างปลอดภัย	มี	ไม่มี	ไม่มี

บทที่ 9
บทสรุปการพัฒนาโครงงาน

9.1 สรุปผลการพัฒนา

ด้านการให้บริการ

โปรแกรมฝั่งแม่ข่าย (IsagMQ) สามารถให้บริการดังต่อไปนี้

1. ออก Certificate ให้กับ ผู้ใช้ได้
2. จัดทำฐานข้อมูลของผู้ใช้ที่ลงทะเบียนแล้ว
3. สามารถรับส่งข้อมูลเกี่ยวกับสถานะ และ ชื่อของผู้ใช้ได้
4. สามารถให้บริการลงทะเบียนและยกเลิกการลงทะเบียนได้
5. สามารถให้บริการล๊อกอินและล๊อกเอาท์ได้
6. สามารถให้บริการเพิ่มและลบคู่สนทนาได้
7. สามารถให้บริการค้นหาคู่สนทนาได้
8. สามารถให้บริการเปลี่ยนชื่อเล่นของผู้สนทนาได้
9. สามารถให้บริการปฏิเสธ และ ยกเลิกการปฏิเสธคู่สนทนา
10. สามารถให้บริการยอมรับคู่สนทนา
11. เปลี่ยนสถานะในการสนทนา

โปรแกรมฝั่งลูกข่าย (IsagQ) สามารใช้บริการดังต่อไปนี้

1. สามารถให้บริการล๊อกอินและล๊อกเอาท์ได้
2. สามารถให้บริการเพิ่มและลบคู่สนทนาได้
3. สามารถให้บริการค้นหาคู่สนทนาได้
4. สามารถให้บริการเปลี่ยนชื่อเล่นของผู้สนทนาได้
5. แสดงสถานะ ของ สมาชิกทั้งสองโปรแกรม
6. ผู้ใช้ติดต่อสื่อสารระหว่างกันแบบ peer-to-peer
7. ผู้ใช้ติดต่อสื่อสารระหว่างกันอย่างปลอดภัยโดยมีการเข้ารหัสข้อมูลสำหรับข้อมูลระหว่าง IsagQ ด้วยกัน
8. ผู้ใช้สามารถติดตั้งบนระบบปฏิบัติการ Windows XP
9. ผู้ใช้สามารถสนทนาด้วยภาษาไทยได้

10. สามารถให้บริการรับส่งไฟล์ระหว่างกันได้
11. เปลี่ยนสถานะในการสนทนา

ด้านความปลอดภัย

1. การติดต่อระหว่าง IsagQ กับ IsagMQ สามารถใช้ SSL Protocol ในการสร้างช่องทางสื่อสารแบบปลอดภัย แม้จะถูกพัฒนาด้วยต่างภาษาโปรแกรมก็ตาม
2. การติดต่อระหว่าง IsagQ กับ IsagQ สามารถใช้ SSL Protocol ของภาษาจาวา (JSSE) ในการสร้างช่องทางการสื่อสารแบบปลอดภัย
3. สืบเนื่องจากข้อ 1 การติดต่อระหว่าง IsagQ กับ IsagMQ จึงประกันได้ว่า มีการพิสูจน์ตัวตน รวมถึง การเข้าและถอดรหัสข้อความทุกครั้ง
4. สืบเนื่องจากข้อ 2 การติดต่อระหว่าง IsagQ กับ IsagQ จึงประกันได้ว่า มีการพิสูจน์ตัวตน รวมถึง การเข้าและถอดรหัสข้อความทุกครั้ง

9.2 สิ่งที่ต้องพัฒนาต่อ

ในการพัฒนาโปรแกรม IsagMQ & IsagQ ครั้งนี้ทางผู้พัฒนาได้ตัดส่วนการลงทะเบียน และการติดต่อใช้งานกับ ICQ สืบเนื่องจากในส่วนของการลงทะเบียนนั้นยังหาข้อสรุปในการเขียนโปรแกรมเพื่อทำงานดังกล่าวไม่ได้ เพราะ ความแตกต่างของภาษาที่ใช้เขียน , รูปแบบในการจัดการใบรับรองสิทธิ์ รวมถึงกลไกในการส่งผ่านใบรับรองสิทธิ์

ดังนั้น ส่วน IsagQ จึงจำเป็นที่ต้องพัฒนาการติดต่อใช้งานกับ ICQ , ความสามารถ หรือ ลูกเล่นอื่นๆ และ ยูสเซอร์อินเทอร์เฟส ที่เหมาะสมและใช้งานง่าย

ในส่วน IsagMQ จะต้องปรับปรุงโครงสร้างของฐานข้อมูล รวมถึง ความสามารถในการรองรับการให้บริการใหม่ๆ ให้ได้

9.3 ข้อจำกัดในการพัฒนา

1. โปรแกรมแม่ข่ายสำหรับรับส่งสารด่วนแบบปลอดภัยถูกออกแบบมาเพื่อการใช้งานภายในองค์กรเป็นหลัก
2. การติดต่อสื่อสารระหว่างกันจำเป็นต้องอยู่บนเครือข่ายเดียวกัน และ ใช้ ควรใช้ IP จริง
3. การออกใบรับรองสิทธิ์ ผู้ดูแลโปรแกรม จะต้องเป็นคนออกให้ด้วยตัวเอง (Manually)
4. ใบร้องขอใบรับรองสิทธิ์จำเป็นต้องถูกสร้างจากโปรแกรม keytool ของ จาวา เท่านั้น

5. ใบรับรองสิทธิ์ , กุญแจสาธารณะ และ กุญแจลับ ของ IsagQ จะถูกเก็บไว้ใน KeyStore เท่านั้น
6. ใบรับรองสิทธิ์ และ กุญแจลับ จะต้องอยู่ในที่ๆปลอดภัย และ อยู่ใน Directory ที่ติดตั้งโปรแกรมเท่านั้น
7. ไม่มีการส่งใบรับรองสิทธิ์ผ่านเครือข่าย ผู้ใช้จำเป็นต้องมาทำสำเนากลับไปใส่ยัง Directory ที่ติดตั้งโปรแกรมด้วยตนเอง
8. ในการเข้าใช้งานของผู้ใช้ จะต้องเพียงแค่ ชื่อของ KeyStore ที่อยู่ใน Directory ที่ติดตั้งโปรแกรม และ Passphrase สำหรับ KeyStore ดังกล่าว
9. กำหนดให้ KeyStore 1 ไฟล์ ต่อ 1 ผู้ใช้งานเท่านั้น
10. ในส่วนของ ใบรับรองสิทธิ์ ที่ระบุเกี่ยวกับ CN (Common name) จะต้องใช้เป็น ตัวเลข UserID เท่านั้น
11. พอร์ตที่ IsagQ เปิดรอรับการเชื่อมต่อแบบ SSL จะเป็นแบบ static port คือ พอร์ตหมายเลข 16002

9.4 ปัญหาของการพัฒนา

ความแตกต่างของภาษาโปรแกรมที่ใช้พัฒนา ทำให้เกิดความซับซ้อนในการสร้างช่องทางการสื่อสารแบบปลอดภัย (SSL) จาก IsagQ ไปยัง IsagMQ ตัวอย่างเช่น JAVA จะใช้ KeyStore ในการจัดการเกี่ยวเรื่อง กุญแจสาธารณะ , กุญแจลับ และ ใบรับรองสิทธิ์ จึงจำเป็นอย่างยิ่งที่จะต้องทำตามขั้นตอน โดยต้องใช้ keytool ในการสร้างใบขอใบรับรองสิทธิ์ เพื่อส่งให้ CA ทำการเซ็น และส่งกลับมาในรูปแบบ DER จากนั้นถึงจะ import ใบรับรองสิทธิ์นั้นเข้าไปใน KeyStore ซึ่งก่อนที่จะ import เข้าไปนั้นควรต้องมี ใบรับรองสิทธิ์ที่ผู้ใช้ไว้วางใจเก็บไว้ก่อน ด้วยวิธีการ import เข้ามาในรูปแบบ DER เช่นกัน จากนั้นจะต้องทำการสร้าง Object ที่เกี่ยวข้องทั้งหมดกับการใช้ SSL ตามขั้นตอนเท่านั้น ซึ่งปัญหานี้ได้สร้างความยากลำบาก และเสียเวลา กับทางผู้พัฒนาเป็นอย่างมาก

ปัญหาการใช้งาน ไลบรารี OpenSSL ก็เป็นอีกหนึ่งในปัญหาหลักที่ทางผู้พัฒนายังขาดความเข้าใจในรายละเอียดสำคัญๆ โดยเฉพาะในส่วนที่เกี่ยวกับ API

ปัญหาในการทำ User Interface ด้วย JAVA ซึ่งยังทำออกมาได้ไม่ดีพอ

ปัญหาในเรื่องของการออกแบบโพรโตคอล ซึ่งยังออกแบบไม่ดีพอ ทำให้ยากต่อการขยายในการรองรับบริการอย่างอื่น ซึ่งอาจจะมีผลทำให้ต้องเปลี่ยนแปลงโค้ดทั้งหมดได้

9.5 ข้อเสนอแนะ

- ผู้พัฒนาควรศึกษาทฤษฎี และ ไลบรารี ของ OpenSSL ให้เข้าใจเพื่อให้ง่ายต่อการนำไปใช้
- ผู้พัฒนาควรศึกษาถึงคลาสที่จำเป็นในการใช้งาน SSL ของ JAVA
- ผู้พัฒนาควรออกแบบโปรแกรมให้เป็น โมดูล มากที่สุด เพื่อให้ง่ายต่อการค้นหาและแก้ไข รวมถึงลดความซ้ำซ้อน
- ฐานข้อมูลของผู้พัฒนาใช้เป็นแนวทางในการพัฒนาได้แต่ไม่ควรยึดแบบตาม

บรรณานุกรม

Francis J. Reiss : "Instant Messaging Security Concerns and Recommended Best Practices SANS Security Essentials GSEC Practical Version 1.4b "

J. Rosenberg , H. Sugano Fujitsu : "Request for Comments: 2778" , Network Working Group February 2000

Hiroaki Kikuchi Minako Tada Shohachiro Nakanishi : "Secure Instant Messaging Protocol Preserving Confidentiality against Administrator" , Dept. of Information Media Technology Tokai University

Stephen A. Thomas : "SSL&TLS Essentials" , Wiley Computer Publishing , United State of America 2000.

Jahn Viega and Matt Messier : "Secure Programming Cookbook" , O'REILLY & Associates,Inc. United State of America 2003.

W. Richard Stevens : "UNIX Network Programming " , Prentice-Hall International, Inc. United State of America 2001.

Jonathan Knudsen : "JAVA Cryptography" , O'REILLY & Associates,Inc. United State of America 2002.

H. M. and P. J. Deitel : "JAVA How to Program" , Deitel & Associates, Inc. United States of America

DeveloperWorks : "Using JSSE for secure socket communication" , IBM.COM