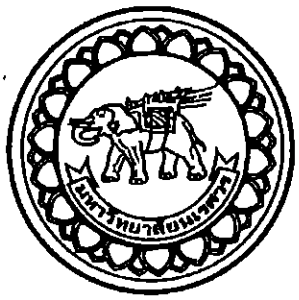


การวิเคราะห์ความปลอดภัยของโปรโตคอลยืนยันตัวตน

Security Analysis of Authentication Protocol

นายวรวิช ฮองกุล รหัส 52362861

นายศตนันท์ ศรีตันตนานนท์ รหัส 52362953

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2555

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน การวิเคราะห์ความปลอดภัยของโพรโตคอลยืนยันตัวตน

ผู้ดำเนินโครงงาน นายวรวิช ฮองกุล รหัส 52362861

นายศตนันท์ ศรีตันตนานนท์ รหัส 52362953

ที่ปรึกษาโครงงาน อ. ภาณุพงศ์ สอนคม

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2555

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

.....ที่ปรึกษาโครงงาน

(อ. ภาณุพงศ์ สอนคม)

.....กรรมการ

(ดร.พงศ์พันธ์ กิจสนาโยธิน)

.....กรรมการ

(อ.จิราพร พุกสุข)

.....กรรมการ

(อ.เศรษฐา ตั้งค้าวานิช)

ชื่อหัวข้อโครงงาน	การวิเคราะห์ความปลอดภัยของโพรโตคอลยืนยันตัวตน
ผู้ดำเนินโครงงาน	นายวรวิช ฮองกุล รหัส 52362861
	นายศตนันท์ ศรีตันตนานนท์ รหัส 52362953
ที่ปรึกษาโครงงาน	อ. ภาณุพงศ์ สอนคม
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2555

.....

บทคัดย่อ

การติดต่อสื่อสารผ่านเครือข่ายมีความสำคัญมากขึ้นเรื่อย ๆ ความปลอดภัยของข้อมูลและโปรโตคอลที่ใช้จึงเป็นเรื่องที่ต้องให้ความสำคัญเป็นอย่างยิ่ง แต่การวิเคราะห์และตรวจสอบความปลอดภัยของโปรโตคอลนั้นเป็นงานที่ยาก ต้องใช้ประสบการณ์ของผู้ที่เชี่ยวชาญเฉพาะ และยังผิดพลาดได้ง่ายหากไม่ดำเนินการอย่างเป็นระบบ โครงงานนี้จึงมุ่งเน้นที่จะศึกษาและพัฒนาวิธีการตรวจสอบความปลอดภัยของโปรโตคอล โดยใช้โปรโตคอลยืนยันตัวตน MP-Auth เป็นกรณีศึกษา ซึ่งในการตรวจสอบความปลอดภัยนี้ ผู้จัดทำโครงงานได้นำเสนอวิธีการตรวจสอบความปลอดภัยโดยใช้ Coloured Petri Nets (CPN) เป็นเครื่องมือในการสร้างแบบจำลองการทำงานของโปรโตคอล และใช้โปรแกรม CPN Tools ในการคำนวณหาความเป็นไปได้ทั้งหมดในการทำงานของแบบจำลองดังกล่าว แล้วค้นหาสถานะที่เกิดความไม่ปลอดภัยในโปรโตคอลขึ้น ผู้จัดทำโครงงานได้สร้างแบบจำลองทั้งหมด 3 ประเภท คือ 1.แบบไม่มี attacker 2. แบบมี attacker และ 3. แบบมี attacker ที่มี Oracle ช่วยถอดรหัส จากผลการทดลองพบว่า MP-Auth ทำงานได้อย่างปลอดภัย บนแบบจำลองที่ 1 และ 2 แต่ไม่ปลอดภัยในแบบจำลองที่ 3 จึงสรุปได้ว่า ความปลอดภัยของ MP-Auth ขึ้นอยู่กับความเป็นความลับของเลขสุ่มชุดหนึ่ง นอกจากนี้ ผู้จัดทำโครงงานได้เขียนโปรแกรมจำลองการเดาเลขชุดดังกล่าว เพื่อทำการทดลองหาความสัมพันธ์ระหว่างความยาวของรหัสกับเวลาที่ใช้ในการถอดรหัส เพื่อนำมาสู่ข้อเสนอแนะว่าควรใช้ความยาวของรหัสอย่างน้อยเท่าไหร่จึงจะปลอดภัย ดังนั้นวิธีการที่ผู้จัดทำโครงงานนำเสนอนั้น จึงเป็นวิธีการที่เป็นระบบสามารถทำตามได้และนำไปประยุกต์ใช้กับโปรโตคอลประเภทอื่น ๆ ได้

Project title	Security Analysis of Authentication Protocol	
Name	Mr. Worawit Hongkul	ID. 52362861
	Mr. Satanan Sritantananon	ID. 52362953
Project advisor	Ajarn. Panupong Sornkhom	
Major	Computer Engineering	
Department	Electrical and Computer Engineering	
Academic year	2012	

.....

Abstract

The communication is important. The security of information and protocol are more important but analysis and check the security of protocol be difficult. It must use specialist and error-prone if not to do it systematically. This project focuses on study and develops of check the security of protocol with example is MP-Auth protocol. The organizer present ways to check the security with Coloured Petri nets (CPN) is tools for creating and modeling of protocol system and use CPN Tools calculate all possible ways of that modeling and search for insecurity state of protocol. The organizer modeling all three models is 1. Normal system model 2. Model has an attacker 3. Model has an attacker and oracle. The results of MP-Auth system is secure on model 1 and model 2 but insecure in model 3 then concludes that the security of MP-Auth depend on the secret of secret code random. Other than the organizer had programming brute force program to guess secret code random for experiment find relationship between length of secret code and time to decrypt bring out to suggestion, how much length of secret code at least should have for secure? As a result the method is presented by organizer is systematically method can follow and apply with any protocol.

กิตติกรรมประกาศ

โครงงานนี้สำเร็จลุล่วงได้ด้วยความกรุณาจากอาจารย์ภาณุพงศ์ สอนคม อาจารย์ที่ปรึกษาโครงงานที่ได้ให้คำแนะนำ แนวคิด ตลอดจนแก้ไขข้อบกพร่องต่างๆ มาโดยตลอด จนโครงงานเล่มนี้เสร็จสมบูรณ์ ผู้ศึกษาจึงขอกราบขอบพระคุณเป็นอย่างสูง

ขอกราบขอบพระคุณพ่อ คุณแม่ และผู้ปกครอง ที่ให้คำปรึกษาในเรื่องต่างๆ รวมทั้งเป็นกำลังใจที่ดีเสมอมา

ขอกราบขอบพระคุณดร.พงศ์พันธ์ กิจสนาโยธิน อาจารย์เศรษฐา ตั้งค้าวานิช และอาจารย์จิราพร พุกสุข กรรมการทั้ง 3 ท่านที่ให้คำแนะนำ

นาย วรวิช ฮองกุล

นาย ศตนันท์ ศรีตันตนานนท์

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

ตารางที่ หน้า

สารบัญรูป

รูปที่ หน้า

สารบัญรูป(ต่อ)

รูปที่ หน้า

สารบัญรูป(ต่อ)

รูปที่ หน้า

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

โปรโตคอลการเข้ารหัสลับมีบทบาทสำคัญในการรักษาความปลอดภัยของระบบสื่อสารในปัจจุบัน มีใช้บนระบบเครือข่ายทั้งแบบระบบมีสายและไร้สาย เพื่อให้มั่นใจในในความเป็นส่วนบุคคล ความสมบูรณ์ และการพิสูจน์ตัวจริง โปรโตคอลการเข้ารหัสลับเป็นโปรโตคอลการสื่อสารที่ใช้อัลกอริทึมแบบการเข้ารหัสลับและการถอดรหัสลับเพื่อให้เกิดความสำเร็จที่แน่นอนในเป้าหมายการรักษาความปลอดภัย การจะรู้ว่าโปรโตคอลปลอดภัยหรือไม่จะต้องทำการวิเคราะห์โปรโตคอล ซึ่งการวิเคราะห์ใช้วิธีทางคณิตศาสตร์ วิธีนี้จะมีปัญหาอยู่ตรงที่ต้องมีผู้เชี่ยวชาญทางคณิตศาสตร์ถึงจะสามารถวิเคราะห์ได้ ในโครงงานนี้จึงพัฒนาวิธีวิเคราะห์โปรโตคอลที่สามารถพิสูจน์ได้ ตรวจสอบได้ มีรูปแบบที่ชัดเจน และสามารถนำไปประยุกต์กับโปรโตคอลอื่นได้ คือ วิธีใช้โปรแกรม Coloured petri net tools (CPN-Tools)

1.2 งานที่เกี่ยวข้อง

1.2.1 MP-Auth เป็นโปรโตคอลที่ใช้ในการศึกษาเกี่ยวกับระบบความปลอดภัยในการพิสูจน์ตัวตนผ่านทางโทรศัพท์มือถือ

1.2.2 CPN Tools ใช้ออกแบบ จำลองการทำงาน และ simulate เพื่อวิเคราะห์ความปลอดภัยของการทำงานของ Protocol MP-Auth

1.2.3 Java cryptography

1.3 วัตถุประสงค์ของโครงงาน

1.3.1 ศึกษาและพัฒนาวิธีการวิเคราะห์ความปลอดภัยของการยืนยันตัวตน

1.3.2 เพื่อทำการทดลองวิเคราะห์ความปลอดภัยของ Protocol MP-Auth

1.4 ขอบเขตการทำโครงงาน

1.4.1 จะใช้ Protocol MP-Auth เป็นตัวอย่างในการวิเคราะห์ความปลอดภัย

1.4.2 ใช้โปรแกรม CPN Tools ในการสร้างระบบการทำงาน พัฒนาระบบการทำงาน และวิเคราะห์ความปลอดภัยของ Protocol ตัวอย่าง

1.4.3 เขียนโปรแกรม ถอดรหัส จากภาษา Java เพื่อหาระยะเวลาในการถอดรหัสข้อมูลที่ถูกเข้ารหัส

1.4.4 ไม่ได้ทดลองจริงกับโทรศัพท์ และคอมพิวเตอร์ที่ไม่ปลอดภัย

1.5 ขั้นตอนการดำเนินงาน

รายการ	พ.ศ. 2555							พ.ศ. 2556		
	มิ.ย	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.
ขั้นวางแผนงาน (P) 1.4.1 วางแผนการดำเนินโครงงานและแบ่งหน้าที่รับผิดชอบ	/									
1.4.2 ส่งแบบฟอร์มและสอบหัวข้อโครงงาน		/								
1.4.3 ศึกษารายละเอียด Protocol ตัวอย่าง ภาษา ML และการใช้งาน CPN Tools		/	/	/						

รายการ	พ.ศ. 2555							พ.ศ. 2556		
	มิ.ย	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.
1.4.4 ศึกษา MP-Auth Protocol				/	/	/				
1.4.5 ศึกษา javax.crypto						/	/			
1.4.6 ออกแบบ MP-Auth Protocol							/	/	/	
1.4.7 เขียนโปรแกรมด้วยภาษา java									/	/
ขั้นสรุปและประเมินผล (C) 1.4.8 สรุปผลการดำเนินโครงการว่าบรรลุตามวัตถุประสงค์หรือไม่										/
ขั้นปรับปรุงตามผลการประเมิน(A) 1.4.9 นำผลสรุปการดำเนินโครงงานไปปรับปรุงแก้ไขต่อไป										/

1.6 ผลที่คาดว่าจะได้รับ

สิ่งที่คิดว่าจะได้รับคือ วิธีการที่สามารถใช้พิสูจน์ได้ว่า Protocol ที่ใช้อยู่ปลอดภัยหรือไม่ โดยเป็นวิธีที่แน่นอนและสามารถพิสูจน์ได้จริง โดยวิธีที่ใช้ในโครงงานนี้ คาดหวังว่าจะใช้เป็นวิธีที่ใช้พิสูจน์และวิเคราะห์ความปลอดภัยของ Protocol อื่นๆนอกเหนือจาก Protocol ตัวอย่างในโครงงานนี้ได้

1.7 งบประมาณของโครงงาน

ค่าถ่ายเอกสารและเข้าเล่มรายงาน	เป็นเงิน	1,000 บาท
ค่าหนังสือ	เป็นเงิน	1,000 บาท
	รวมเป็นเงิน	2,000 บาท

บทที่ 2

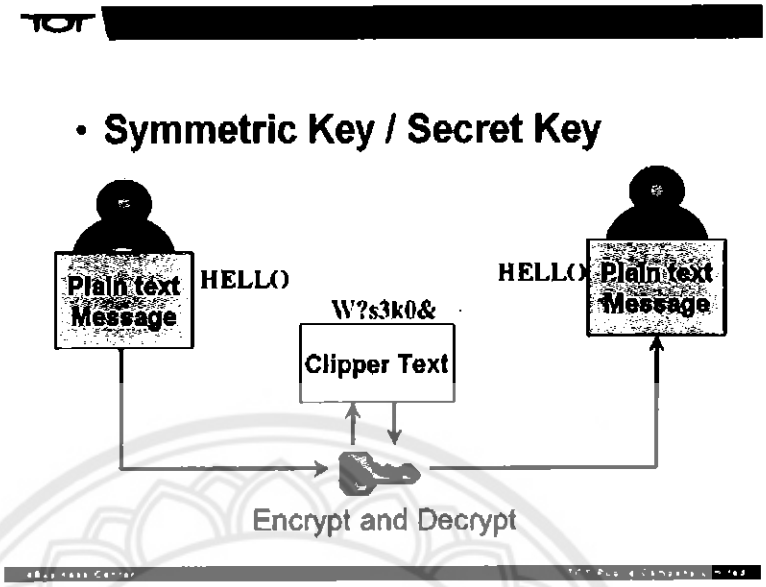
ทฤษฎีที่เกี่ยวข้อง

2.1 วิทยาการเข้ารหัสลับ (Cryptography)

วิทยาการเข้ารหัสลับ [1] คือ การแปลงข้อความปกติ (Plain Text) ให้กลายเป็นข้อความลับ (Cipher Text) โดยข้อมูลที่ถูกเข้ารหัสลับจะมีการเปลี่ยนแปลงไปทำให้เป็นข้อมูลที่ผู้อื่นไม่สามารถเข้าใจได้ นอกเหนือจากคู่สนทนา

ระบบการเข้ารหัสข้อมูล เป็นวิธีการแปลงข้อมูลอิเล็กทรอนิกส์ธรรมดาให้อยู่ในรูปแบบที่บุคคลทั่วไปไม่สามารถอ่านเข้าใจได้ โดยทั่วไปการเข้ารหัสจะทำก่อนการจัดเก็บข้อมูล หรือทำก่อนการส่งข้อมูลโดยการนำข้อมูลทางอิเล็กทรอนิกส์ธรรมดากับกุญแจ (Key) ซึ่งเป็นตัวเลขสุ่มใดๆ มาผ่านกระบวนการทางคณิตศาสตร์ ผลที่ได้คือข้อมูลที่เข้ารหัส ขั้นตอนที่กล่าวมานี้จะเรียกว่า การเข้ารหัส (Encryption) และเมื่อต้องการอ่านข้อมูลก็เอาข้อมูลที่เข้ารหัสกับกุญแจมาผ่านกระบวนการทางคณิตศาสตร์ ผลที่ได้ก็คือข้อมูลดั้งเดิม ขั้นตอนนี้จะเรียกว่า การถอดรหัส (Decryption) ระบบการเข้ารหัสสามารถแบ่งตามวิธีการใช้กุญแจได้เป็น 2 วิธี ดังนี้

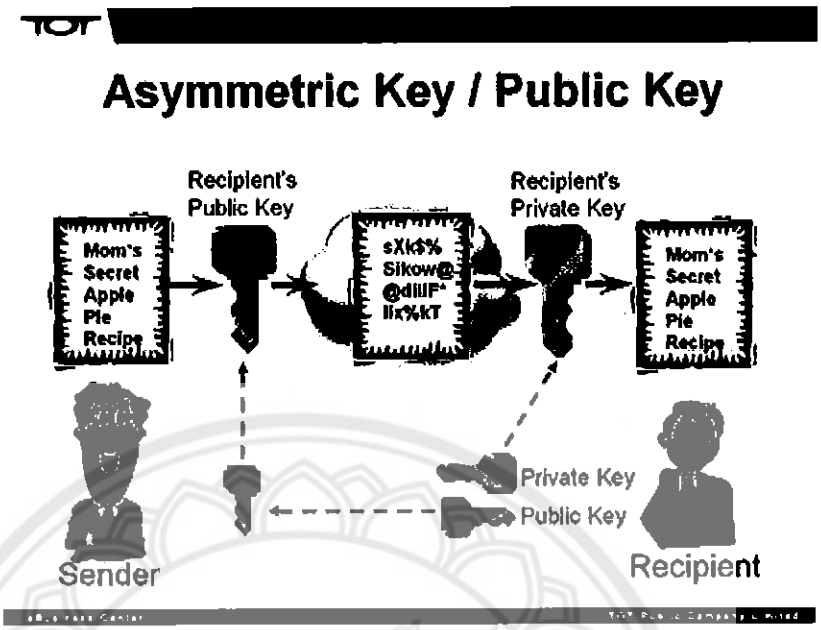
2.1.1 ระบบการเข้ารหัสแบบกุญแจสมมาตร (Symmetric-key Cryptography หรือ Secret Key) คือ การเข้ารหัสข้อมูลด้วยกุญแจเดี่ยว ทั้งผู้ส่งและผู้รับ โดยวิธีการนี้ผู้รับกับผู้ส่งต้องตกลงกันก่อนว่าจะใช้รูปแบบไหนในการเข้ารหัสข้อมูล ซึ่งรูปแบบในการเข้ารหัสข้อมูลที่ผู้รับผู้ส่งตกลงกันก็คือ กุญแจ โดยทั้งการเข้ารหัสข้อมูลก่อนส่งและการถอดรหัสข้อมูลที่ได้รับจะใช้กุญแจตัวเดียวกัน



รูปที่ 2.1 : Symmetric-key Cryptography หรือ Secret Key

ที่มา : http://www.ca.tot.co.th/Portals/456/symmetric_key.gif

2.2.2 ระบบการเข้ารหัสแบบกุญแจอสมมาตร (Asymmetric-key Cryptography หรือ Public Key) โดยการเข้ารหัสแบบนี้จะใช้หลักกุญแจคู่ทำการเข้ารหัสและถอดรหัส โดยกุญแจคู่จะประกอบไปด้วย กุญแจส่วนตัว (Private Key) และกุญแจสาธารณะ (Public Key) โดยหลักการทำงาน คือ ถ้าใช้กุญแจลูกใดเข้ารหัส ก็ต้องใช้กุญแจอีกลูกหนึ่งถอดรหัส อธิบายโดยละเอียดก็คือ เมื่อผู้ส่งต้องการส่งข้อมูลให้กับผู้รับ ผู้ส่งจะเข้ารหัสข้อมูลด้วยกุญแจสาธารณะของผู้รับ จากนั้นจึงส่งให้กับผู้รับ เมื่อผู้รับทำการถอดรหัสก็จะใช้กุญแจส่วนตัวของผู้รับในการถอดรหัส แล้วผู้รับก็จะได้ข้อมูลดั้งเดิมที่ผู้ส่งต้องการจะส่งมา สำหรับการเข้ารหัสและถอดรหัสด้วยกุญแจคู่จะใช้ฟังก์ชันทางคณิตศาสตร์เข้ามาช่วยโดยที่ฟังก์ชันทางคณิตศาสตร์ที่นำมาใช้ ได้รับการพิสูจน์มาแล้วว่าจะมีเฉพาะกุญแจคู่ของมันเท่านั้นที่จะสามารถถอดรหัสได้ ไม่สามารถนำกุญแจคู่อื่นมาถอดรหัสได้อย่างเด็ดขาด



รูปที่ 2.2 : Asymmetric-key Cryptography หรือ Public Key

ที่มา : http://www.ca.tot.co.th/Portals/456/Asymmetric_Key.gif

2.2 การพิสูจน์ตัวตน (Authentication)

การพิสูจน์ตัวตน [2] คือ กระบวนการตรวจสอบตัวตน เพื่อเป็นการพิสูจน์ว่าผู้ที่เข้าใช้งานระบบหรือรีซอร์ส (Resource) นั้นๆ เป็นผู้ที่ได้รับอนุญาตให้ใช้งานได้จริงๆ หรือเป็นบุคคลนั้นจริงๆ โดยวิธีที่พิสูจน์ตัวตนมีหลายวิธีตามความสะดวก และความเหมาะสมในการใช้งาน โดยวิธีต่างๆ เช่น การใช้ username และ password การใช้การแสกนลายนิ้วมือหรือแสกนรูปม่านตาและอื่นๆ (Biometric technology) เป็นต้น

2.3 โปรโตคอล (Protocol)

โปรโตคอล [3, 4] คือ ข้อกำหนดหรือข้อตกลงที่ใช้ในการสื่อสารระหว่างคอมพิวเตอร์ คล้ายกับเป็นภาษากลางที่ทำให้คอมพิวเตอร์ที่อยู่ในเครือข่ายสามารถสื่อสารกันได้รู้เรื่อง เหมือนกับมนุษย์ที่สามารถใช้ภาษาอังกฤษเป็นภาษากลางในการสื่อสารถึงกันได้ โปรโตคอลช่วยให้ระบบคอมพิวเตอร์สองระบบที่แตกต่างกันสามารถสื่อสารกันอย่างเข้าใจได้ ตัวอย่างของโปรโตคอล เช่น

โปรโตคอล HTTP (Hypertext Transfer Protocol) จะใช้เมื่อเรียกโปรแกรมบราวเซอร์ (Browser) เพื่อเปิดเข้าเว็บไซต์ต่างๆ ทางอินเทอร์เน็ต

โปรโตคอล SMTP (Simple Mail Transfer Protocol) เป็นโปรโตคอลมาตรฐานที่ใช้ในการรับส่ง Electronic mail (E-mail) ทางอินเทอร์เน็ต

เป็นต้น

2.4 โปรโตคอลยืนยันตน (Authentication Protocol)

โปรโตคอลยืนยันตน คือ โปรโตคอลการสื่อสารที่มีกระบวนการยืนยันตนรวมอยู่ในโปรโตคอลด้วย โดยถ้าใช้แค่โปรโตคอลปกติในการสื่อสารกันแค่ต้นทางกับปลายทางใช้โปรโตคอลเดียวกันก็สามารถสื่อสารกันได้แต่จะไม่รู้ว่าอีกด้านที่กำลังสื่อสารอยู่ด้วย ไม่ว่าจะเป็นบุคคล เซิร์ฟเวอร์ อื่นๆ นั้นเป็น บุคคลนั้นจริงๆ หรือเป็นเซิร์ฟเวอร์นั้นจริงๆหรือไม่ เพื่อการสื่อสารที่ปลอดภัยยิ่งขึ้น จึงมีการคิดค้นโปรโตคอลยืนยันตนขึ้นมาเพื่อนำมาใช้ในการสื่อสารที่ต้องการความปลอดภัยที่มากขึ้น โดยก่อนที่การสื่อสารจะเกิดขึ้นก็จะมีการใช้กระบวนการยืนยันตนว่าอีกด้านที่กำลังจะสื่อสารด้วยคือตัวจริงหรือไม่ เพื่อความปลอดภัย และความเป็นส่วนตัวในการสื่อสาร โปรโตคอลหลักที่นิยม [8] ประกอบไปด้วย Secure Socket Layer (SSL), Secure Shell (SSH), Internet Protocol Security (IPSec), และ Kerberos

2.4.1 Secure Socket Layer (SSL)

SSL เป็นโปรโตคอลความปลอดภัยที่ถูกใช้เป็นมาตรฐานในการเพิ่มความปลอดภัยให้กับการสื่อสารบนเครือข่ายอินเทอร์เน็ตที่มีการเข้ารหัสแบบใช้ Public Key

SSL คือ โปรโตคอลที่อยู่ระหว่าง Application Layer และ Transport Layer สามารถรองรับการทำงานกับ Application ต่างๆได้ เช่น HTTP, FTP (File Transfer Protocol), Telnet, SMTP เป็นต้น SSL ทำงานโดยอาศัยหลักการของการเข้ารหัสข้อมูล (Encryption), Message Digests และลายเซ็นดิจิตอล (Digital Signature) โดยแบ่งหน้าที่ออกเป็น 3 ส่วน คือ

1 การตรวจสอบ Server ว่าเป็นตัวจริง

2 การตรวจสอบว่า Client เป็นตัวจริงหรือไม่

3 การเข้ารหัสลับการเชื่อมต่อ ให้กับข้อมูลที่ รับ-ส่ง ระหว่าง Client และ Server ซึ่งมีแค่ Client และ Server เพียงสองเครื่องเท่านั้นที่จะมีกุญแจถอดรหัสข้อมูลนั้นได้

2.4.2 Secure Shell (SSH)

SSH คือ network Protocol ที่แลกเปลี่ยนข้อมูลโดยช่องทางที่ปลอดภัยระหว่างอุปกรณ์เครือข่ายสองตัว ใช้ Linux หรือ Unix เป็นระบบปฏิบัติการพื้นฐานในการเข้าถึงบัญชีผู้ใช้ SSH ได้รับการออกแบบให้มาแทนการ Telnet, Rlogin, RSH (The Remote Shell) ด้วยเหตุผลทางด้านความปลอดภัย การส่งข้อมูลจะอยู่ในรูป Plaintext ที่มีการเข้ารหัสข้อมูล เพื่อให้ข้อมูลเป็นความลับและสามารถส่งผ่านเครือข่ายอินเตอร์เน็ตได้อย่างสมบูรณ์

2.4.3 Internet Protocol Security (IPSec)

IPSec คือ ชุดโปรโตคอลเพิ่มเติมของ Internet Protocol (IP) เพื่อให้การติดต่อสื่อสารมีความปลอดภัยมากขึ้น โดยมีการเพิ่มการทำ Authentication และการ Encryption ในข้อมูล IP Package ที่รับส่งกัน IPSec ทำงานอยู่ 2 โหมด คือ

1 Transport Mode จะทำการ Encrypt หรือ Authenticate เฉพาะในส่วนข้อมูลของ IP Package ที่จะส่ง แต่จะไม่ทำในส่วนของ Header หรือ IP Header ของ IP Package

2 Tunnel Mode จะทำการ Encrypt และ Authenticate ทั้ง IP Package หรือทั้ง ส่วนข้อมูล และ Header และสร้าง IP Header ขึ้นมาใหม่

2.4.4 Kerberos

Kerberos คือ โปรโตคอลการพิสูจน์ตัวตนบนระบบเครือข่ายที่ถูกพัฒนาโดย MIT เพื่อใช้แก้ปัญหาเรื่องความไม่ปลอดภัยของการพิสูจน์ตัวตนแบบเดิมที่มีการส่งรหัสผ่านบนเครือข่ายโดยที่ไม่มีการเข้ารหัสข้อมูลทำให้ข้อมูลอาจถูกดักจับได้ โดย Kerberos มี 2 ส่วน คือ

1 Kerberos Ticket หรือ Ticket คือหลักฐานทางอิเล็กทรอนิกส์ที่ใช้ในการพิสูจน์ตัวตนตามที่ได้กล่าวอ้างจริงบนระบบ Kerberos

2 Kerberos Server หรือ Key Distribution Center (KDC) คือส่วนที่ทำหน้าที่ในการจำหน่าย Kerberos Ticket ภายใน Kerberos Server ประกอบด้วย

2.1 Authentication Services (AS) ทำหน้าที่ในการพิสูจน์ตัวตนของผู้ใช้ก่อนการเข้าใช้บริการ

2.2 Ticket Granting Services (TGS) ทำหน้าที่จำหน่าย Kerberos Ticket เพื่อให้ผู้ใช้นำไปใช้กับบริการที่ต้องการ

2.3 Kerberos Database เป็นฐานข้อมูล Kerberos อยู่บน KDC ทำหน้าที่เก็บชื่อบัญชีผู้ใช้ รหัสผ่าน และข้อมูลที่เกี่ยวกับการควบคุมดูแลของบัญชีผู้ใช้ทั้งหมดบนกลุ่มระบบเครือข่ายใน Kerberos

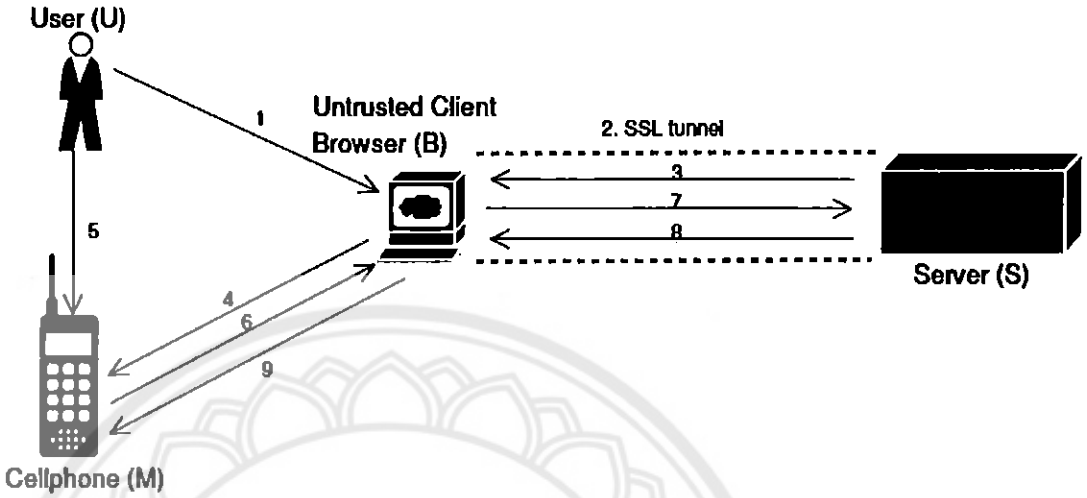
2.5 โปรโตคอล MP-Auth (Mobile Password Authentication Protocol)

โปรโตคอล MP-Auth [5] เป็นโปรโตคอลที่ใช้ปกป้อง password ของผู้ใช้ ที่ต้องใส่ผ่านคอมพิวเตอร์ที่ไม่ปลอดภัย หรือไม่น่าเชื่อถือ โดยการใช้โทรศัพท์เคลื่อนที่ ในโปรโตคอล MP-Auth จะปกป้อง password ของผู้ใช้จากวิธีการมุ่งร้ายแบบง่ายๆ เช่น การดักจับการพิมพ์คีย์บอร์ดโดยโปรแกรม keyloggers ที่ติดตั้งอยู่บนเครื่องคอมพิวเตอร์ที่ไม่น่าเชื่อถือ เป็นต้น โดยจะใช้โทรศัพท์เคลื่อนที่เข้ารหัส password ไว้และส่งไปให้เครื่องคอมพิวเตอร์ จากนั้น Browser บนเครื่องคอมพิวเตอร์ก็จะทำการเข้ารหัสด้วย Public Key และส่งไปให้เซิร์ฟเวอร์เพื่อให้เซิร์ฟเวอร์ประมวลผลต่อไป

ขั้นตอนการทำงานของโปรโตคอล MP-Auth มีดังต่อไปนี้

สัญลักษณ์ที่ใช้ใน MP-Auth

U, M, B, S	U คือ User หรือผู้ใช้ M คือ โทรศัพท์เคลื่อนที่ B คือ Browser หรือเว็บบราวเซอร์ และ S คือ Server
ID_S, ID_U	ID_S คือ ID ของ Server และ ID_U คือ ID ของ User โดยที่ ID_U จะมีแค่หนึ่งเดียวในหนึ่ง Server
P	P คือ Password ของ User ที่จะใช้เพื่อระบุตัวตนกับ Server
R_S	R_S คือ ตัวเลขแบบสุ่ม (Random) ที่สร้างขึ้นโดย Server
$\{Data\}_K$	ใช้แสดงถึงการเข้ารหัสแบบ Symmetric (Secret Key) โดย Data คือ ข้อความปกติ และ K คือ คีย์ที่ใช้เข้ารหัส
$\{Data\}_{Es}$	ใช้แสดงถึงการเข้ารหัสแบบ Asymmetric (Public Key) โดย Data คือ ข้อความปกติ และ E_S คือ คีย์ที่ใช้เข้ารหัส
X.Y	คือ การต่อกันของ X และ Y
K_{BS}	คือ คีย์แบบ Symmetric ระหว่าง Browser กับ Server
$f(\cdot)$	คือ การเข้ารหัสแบบ hash function



รูปที่ 2.3 : MP-Auth protocol steps

ขั้นตอนการทำงานของ MP-Auth

1 User เปิด Browser บนเครื่องคอมพิวเตอร์ที่ไม่น่าเชื่อถือ และเข้าเว็บธนาคาร

2 Browser และ Server ได้ติดต่อกันและสร้างช่อง SSL ขึ้น และใช้ K_{BS} เป็นคีย์ที่ใช้เข้ารหัสเพื่อใช้ติดต่อกันผ่านช่อง SSL

3 Server สร้างเลขสุ่ม หรือ R_S และส่งไปให้ Browser

$$B \rightarrow S : \{IDs.Rs\}_{KBS} \qquad (1)$$

4 Browser ถอดรหัสข้อความ (1) และส่งไปให้โทรศัพท์เคลื่อนที่ (M)

$$M \rightarrow B : IDs.Rs \qquad (2)$$

5 บนจอโทรศัพท์เคลื่อนที่จะแสดง ID ของ Server (ID_S) และจะเตือนให้ใส่ User ID (ID_U) และ Password (P) สำหรับ Server

6 โทรศัพท์เคลื่อนที่จะสร้างตัวเลขสุ่ม หรือ R_M และเข้ารหัสด้วย E_S จากนั้นโทรศัพท์เคลื่อนที่จะนำ R_S และ R_M ไปเข้าฟังก์ชัน Hash เพื่อให้ได้ K_{MS} และส่งข้อความ (4) ไปให้ Browser

$$K_{MS} = f(Rs.Rm) \qquad (3)$$

$$M \rightarrow B : \{Rm\}_{Es}.\{f(Rs).IDu.P\}K_{MS} \qquad (4)$$

7 Browser ทำการเข้ารหัสข้อความ (4) ด้วย K_{BS} เป็นการสื่อสารผ่านช่อง SSL และส่งข้อความต่อไปให้ Server

8 เมื่อถอดรหัสด้วย K_{BS} เป็นการถอดรหัสข้อความที่ออกจากช่อง SSL จากนั้น Server จะถอดรหัส R_M ด้วย Private Key แลวนำ R_M ที่ได้ไปเข้าฟังก์ชัน Hash เหมือนใน สมการ (3) แล้วจะได้ K_{MS} มาจากนั้นนำ K_{MS} ที่ได้ไปถอดรหัสที่เหลือจากข้อความ (4) จะได้ ID_U, P, และค่า Hash ของ R_S มาจากนั้น Server ก็จะนำทั้งสามค่าไปตรวจสอบ เมื่อเสร็จจะส่งข้อความ (5) สำหรับโทรศัพท์เคลื่อนที่ไปให้ Browser

$$B \rightarrow S : \{\{f(Rm)\}K_{MS}\}K_{BS} \qquad (5)$$

9 Browser ถอดรหัสด้วย K_{BS} แล้วจากนั้นก็ส่ง $\{f(Rm)\}K_{ms}$ ไปให้โทรศัพท์เคลื่อนที่ เมื่อโทรศัพท์เคลื่อนที่ได้ได้รับข้อความมาก็ทำการถอดรหัสด้วย K_{MS} แล้วจะได้ $f(Rm)$ เพื่อที่จะเอาไปตรวจสอบกับค่า R_M ที่โทรศัพท์เคลื่อนที่สร้างไว้ตอนแรกว่าเหมือนกันหรือไม่ จากนั้นบนหน้าจอโทรศัพท์เคลื่อนที่จะแสดงข้อความว่า Success หรือ Failure ให้ User เห็น

2.6 เครื่องมือและอุปกรณ์ที่ใช้ในการวิเคราะห์

2.6.1 Petri net และ Coloured Petri net

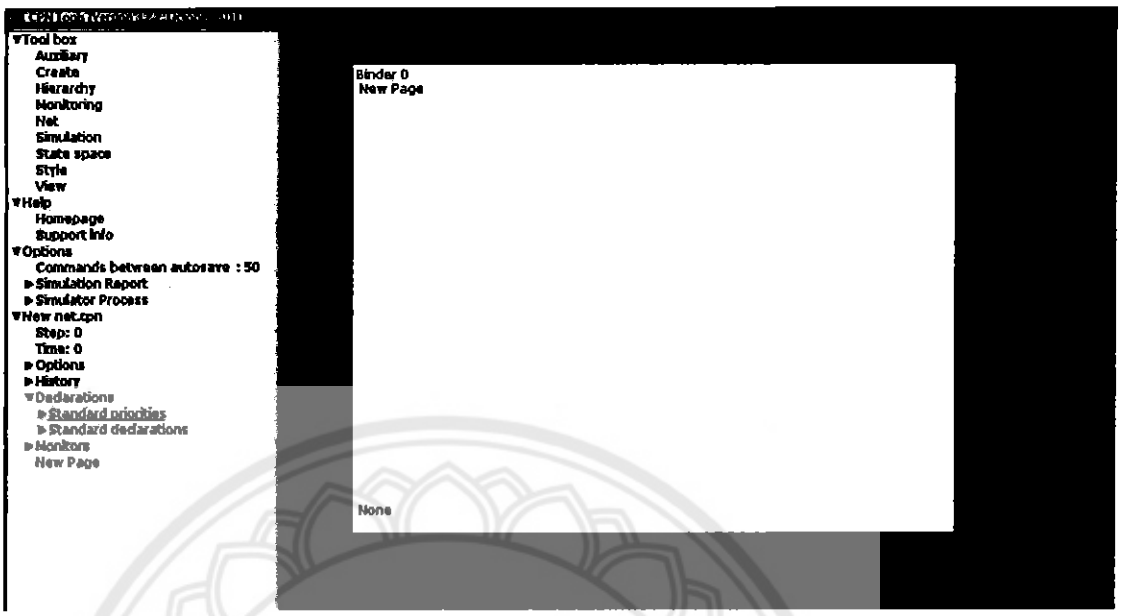
Petri net [6] ประกอบไปด้วย Place และ Transition เป็นหนึ่งในหลาย ๆ ภาษาในการสร้างแบบจำลองทางคณิตศาสตร์ในรูปแบบกราฟ Place ใน Petri net สามารถบรรจุค่าต่างๆใน marks เรียกว่า Token Token ที่แสดงอยู่เหนือ Place เรียกว่า marking Transition จะยิงออกไปเมื่อได้รับ Token มาจาก Place

Coloured Petri net (CPN) [9] เป็นภาษากราฟในการสร้าง model ของระบบที่เกิดขึ้นพร้อมกัน และใช้ในการวิเคราะห์คุณสมบัติของระบบ CPN เป็นภาษาที่ใช้ออกแบบเหตุการณ์ที่ไม่ต่อเนื่องผสมกับความสามารถของ Petri net และความสามารถของภาษาการเขียนโปรแกรมระดับสูง โดย Petri net ได้จัดสรรให้มีสัญลักษณ์พื้นฐานทางด้านกราฟ และพื้นฐานดั้งเดิมสำหรับการออกแบบ การเกิดขึ้นพร้อมกัน การติดต่อสื่อสาร และการประสานเวลา ในการใช้งาน

2.6.2 โปรแกรม CPN Tools (Coloured Petri Net Tools)

โปรแกรม CPN Tools [10] เป็นเครื่องมือที่ใช้สำหรับ แก้ไข simulate และวิเคราะห์ เกี่ยวกับ Coloured Petri Net (CPN) โดย CPN จัดภาษาคอมพิวเตอร์ที่ใช้ในการสร้างโมเดลเพื่อบรรยายพฤติกรรมการทำงานของระบบต่างๆ พัฒนามาจากการรวมกันของ Petri Nets (Place/Transition Nets) กับภาษาในระดับสูงของคอมพิวเตอร์ (High Language)

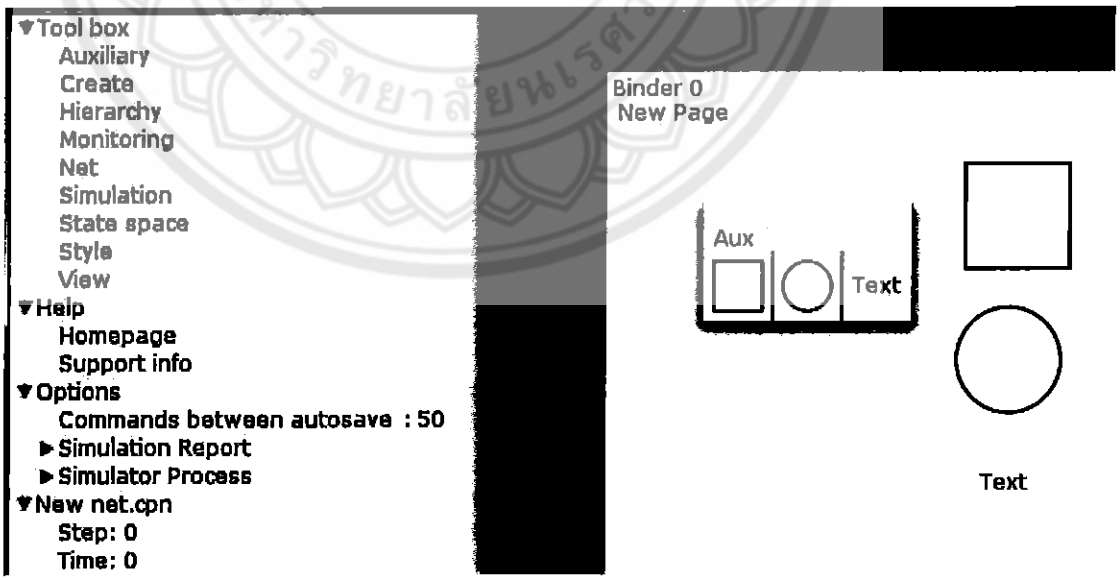
ภายใน CPN Tools มีอุปกรณ์ที่สามารถใช้สร้างแบบจำลองของระบบเพื่อดูการทำงานของระบบโดยสิ่งที่ออกแบบออกมาจะมีลักษณะเป็น State และแต่ละ State จะมี Transition ซึ่งเป็น Tools ที่ใช้แสดงเหตุการณ์ขั้นระหว่าง State เนื่องจากระบบที่ออกแบบบน CPN Tools ออกมาเป็นแบบ State ทำให้ผู้ใช้สามารถมองเห็นการทำงานทั้งหมดของระบบที่ออกแบบไว้ได้ทั้งหมด และยังสามารถปรับเปลี่ยนแก้ไขเพื่อวิเคราะห์ระบบได้โดยไม่จำเป็นต้องสร้างระบบจริงๆขึ้นมาเพื่อทำการวิเคราะห์ เนื่องจากภายในโปรแกรม CPN Tools สามารถออกแบบ ปรับเปลี่ยน แก้ไข ต่างๆได้ และสามารถสร้างเส้นทางที่สามารถเป็นไปได้ทุกกรณีที่เกิดขึ้นเหมือนในระบบจริงได้ ทำให้ประหยัดค่าใช้จ่ายในการออกแบบและทดสอบก่อนสร้างระบบจริง นอกจากนี้ยังมีอุปกรณ์ที่ใช้ Simulate เพื่อดูการไหลของข้อมูลแต่ละเส้นทางได้ และมีเครื่องมือเพื่อสร้าง State Space และทำ Query เพื่อการวิเคราะห์ได้อีกด้วย



รูปที่ 2.4 : หน้าตาของโปรแกรม CPN Tools

Tool box ในโปรแกรม CPN Tools ที่ถูกใช้งานบ่อยๆ

-Auxiliary



รูปที่ 2.5 : Auxiliary

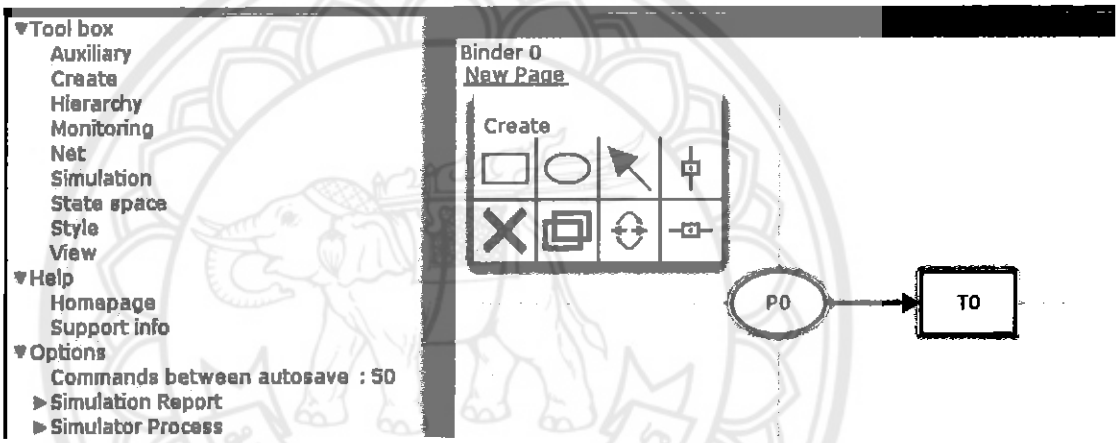
จากซ้ายไปขวารูปที่ 2.5

-ใช้วาดสี่เหลี่ยมธรรมดา

-ใช้วาดวงกลมธรรมดา

-ใช้พิมพ์ข้อความหรือ Code ใน Page

-Create



รูปที่ 2.6 : Create

แถว 1 จากซ้ายไปขวารูปที่ 2.6

-ใช้วาด Transition เปรียบเสมือนจุดเกิดเหตุการณ์ สามารถเขียนข้อกำหนดต่างๆได้โดยการกดปุ่ม Tab บนคีย์บอร์ด จะมีช่องให้ใส่ Code เพื่อกำหนดข้อตกลงต่าง เช่น If else เป็นต้น โดย Tab สามารถกดได้หลายครั้ง

-ใช้วาด Place เปรียบเสมือนที่ให้ข้อมูลอยู่ หากกดปุ่ม Tab บนคีย์บอร์ด 1 ครั้งจะให้ใส่ชนิดของขอมูล กดครั้งที่ 2 จะให้ใส่ข้อมูลโดยต้องมีชนิดข้อมูลตามชนิดที่ใส่ตอนกด Tab ครั้งแรก

-ใช้วาด เส้นที่มีหัวลูกศร เป็นเส้นการเดินทางของข้อมูล หากกดที่เส้นจะสามารถใส่ชนิดของข้อมูลที่สามารถผ่านเส้นทางนั้นๆได้

-ใช้วาด Guide line แนวแกน y

แถว 2 จากซ้ายไปขวารูปที่ 2.6

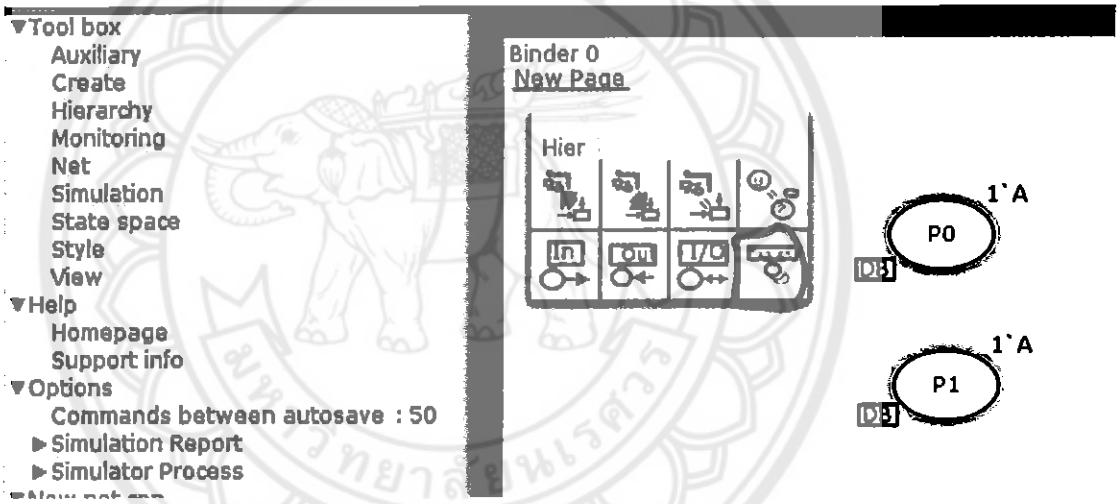
-ใช้ลบสิ่งที่วาดบน Page

-ใช้ Clone สิ่งที่อยู่บน Page

-ใช้คลิกเพื่อเปลี่ยนทิศทางของหัวลูกศร

-ใช้วาด Guide line แนวแกน x

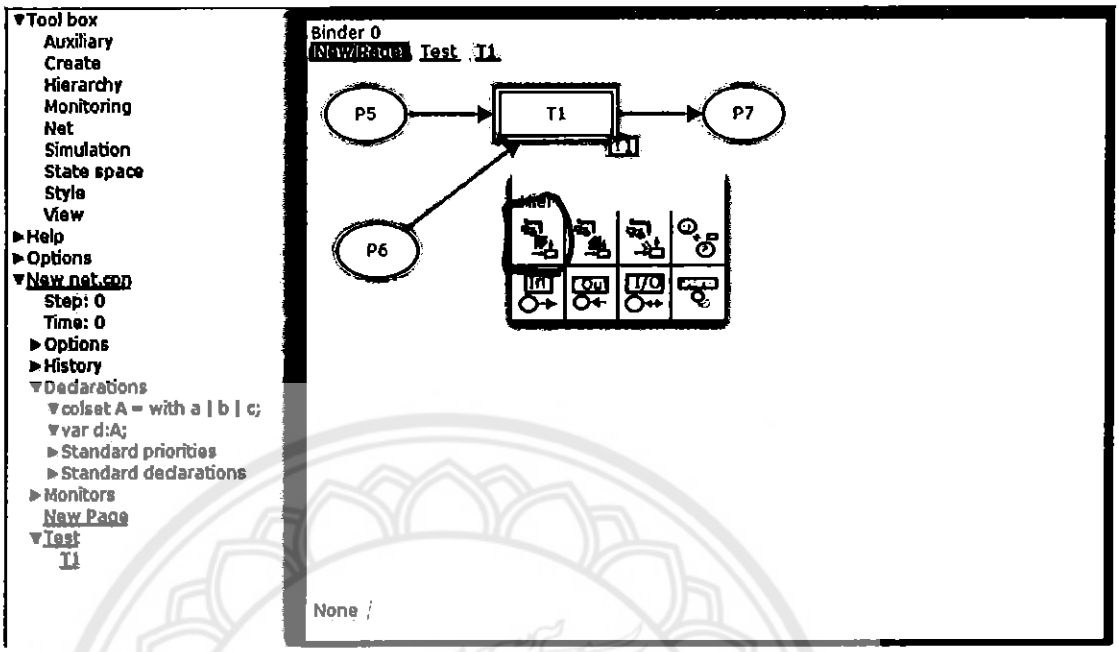
-Hierarchy



รูปที่ 2.7 : Fuses Place

จากวงกลมสีแดงรูปที่ 2.7

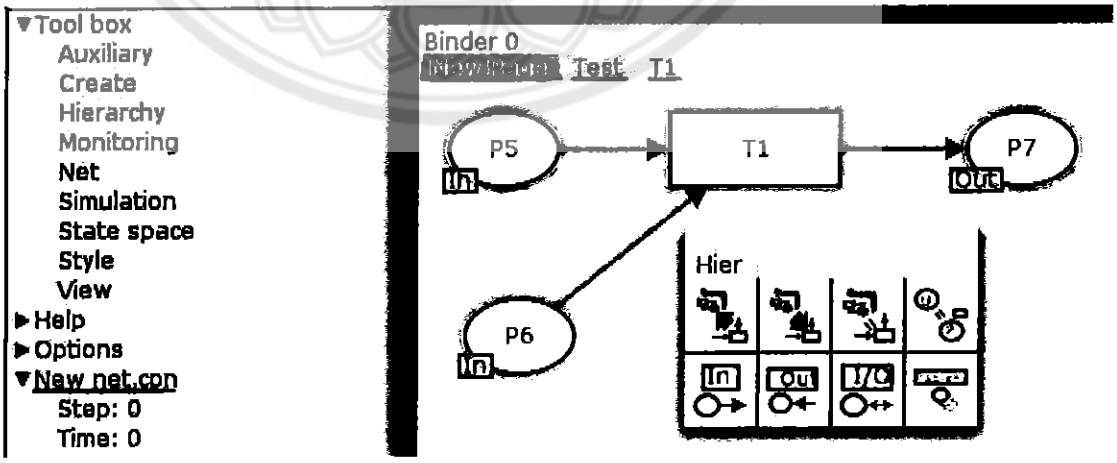
-ใช้ รวม Place ทำให้ Place 2 Place เปรียบเสมือน Place เดียวกัน



รูปที่ 2.8 : Move a transition to a subpage

จากวงกลมสีแดงรูปที่ 2.8

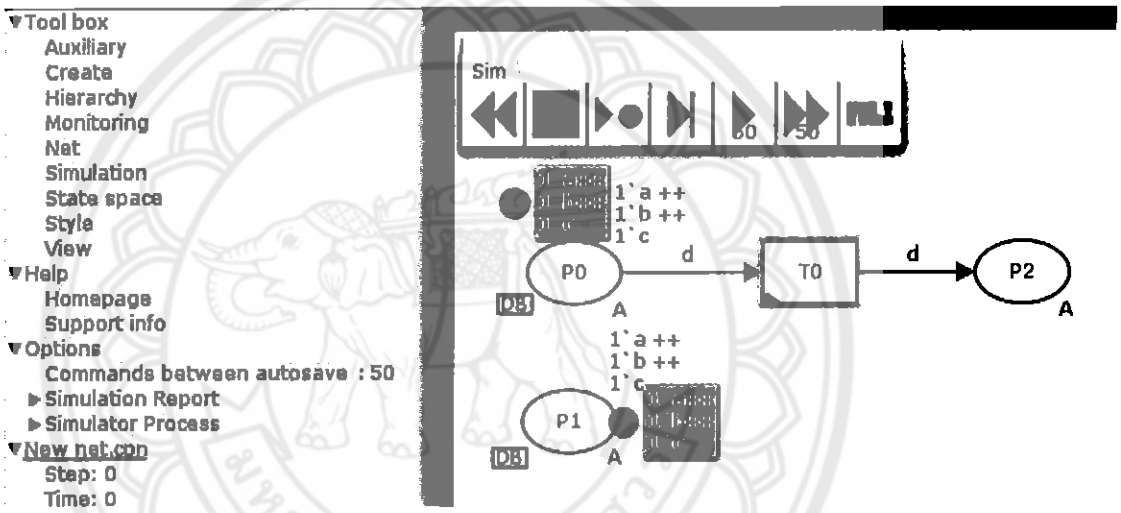
-ใช้สร้าง Subpage ให้กับ Transition เมื่อคลิกที่ Transition แล้วจะเกิด Subpage ขึ้นมาด้านซ้ายล่างของกลุ่มเครื่องมือ โดย Subpage ที่สร้างจะอยู่ต่อจาก Page หลักของ Transition และมีชื่อเหมือน Transition ที่ถูกคลิกโดยเครื่องมือนี้



รูปที่ 2.9 : Subpage

เมื่อคลิกลาก Subpage ที่ถูกสร้างขึ้น มาจากซ้ายล่างที่เกิดของแถบเครื่องมือ มาไว้ตรงแถบ Page ก็จะปรากฏหน้า Page ที่ใช้ทำงานได้ โดย In คือทางเข้าของข้อมูลที่เข้ามายัง Subpage และ Out คือทางออกของข้อมูลที่ออกจาก Subpage นี้ จากรูปที่ 2.9 เมื่อต้องการพัฒนาเพิ่มก็ให้ลบ Transition T1 ออก จากนั้นจึงเริ่มสร้างระบบที่ต้องการ โดยมี Place ที่มี In เป็นทางเข้าของข้อมูล และ Place ที่มี Out เป็นทางออกของข้อมูล

-Simulation



รูปที่ 2.10 : Simulation1

จากซ้ายไปขวารูปที่ 2.10

-ใช้เพื่อคลิกย้อนกลับไปจุดเริ่มต้น

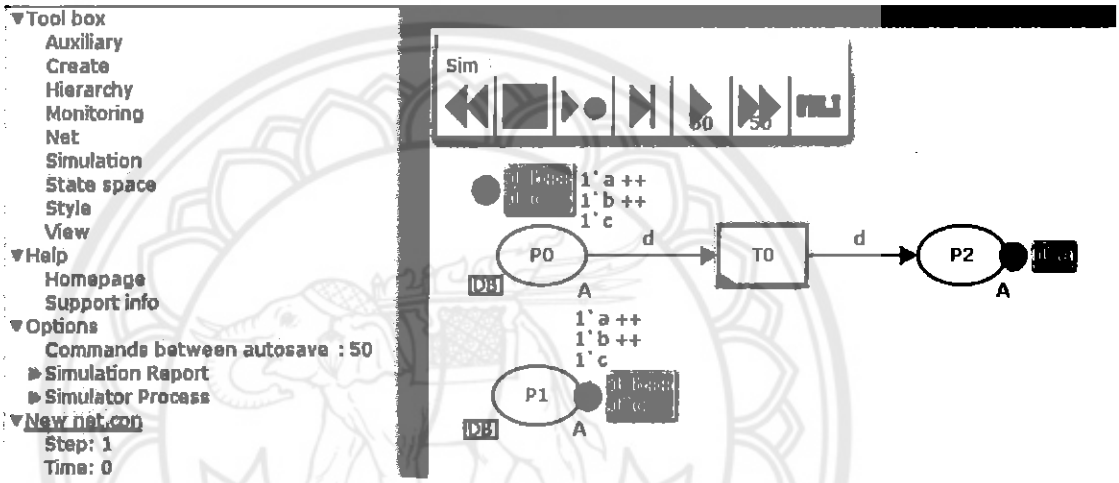
-ใช้เพื่อ หยุด การ Simulate

-ใช้เพื่อดู Transition ที่กำลังจะเกิดเหตุการณ์ว่า ขาเข้าของ Transition นั้นมีอะไรบ้าง มีกี่เส้นทางที่จะเข้า และแต่ละเส้นทางสามารถเป็นข้อมูลอะไรได้บ้าง

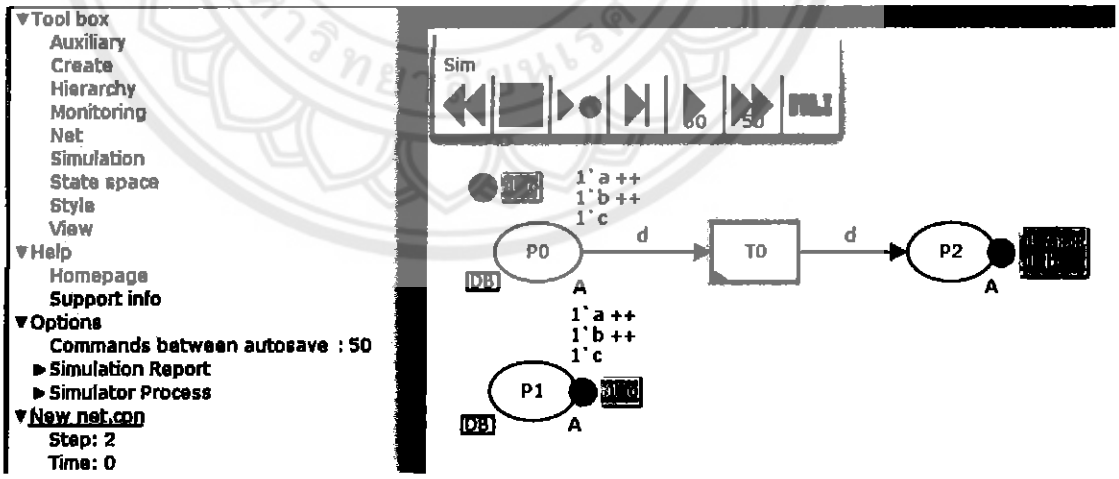
-ใช้เพื่อคลิกให้ Simulate ทีละขั้นตอน

-ใช้เพื่อให้ Simulate ไปเรื่อยๆจนกว่าการทำงานจะจบ

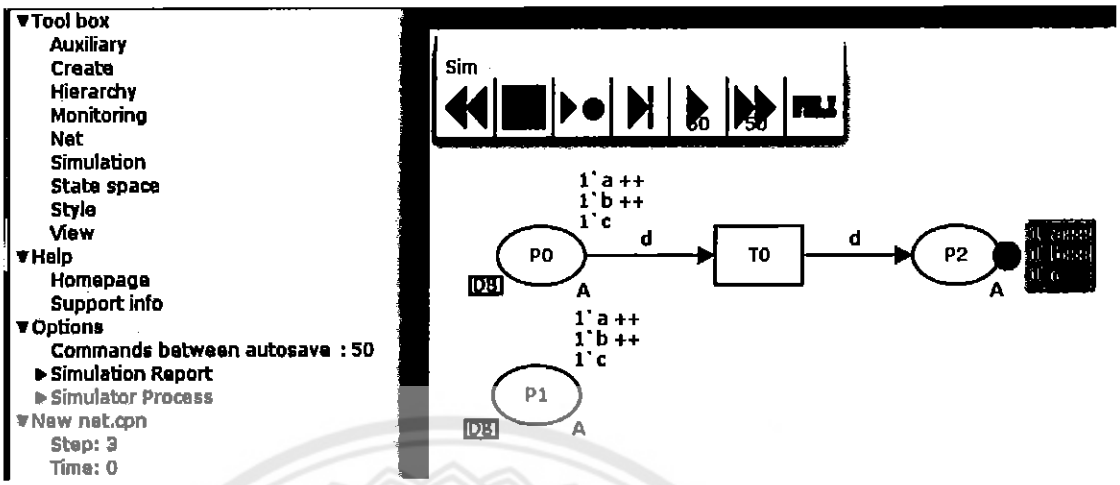
-ใช้เพื่อให้การ Simulate ไปทีละหลายๆขั้นตอน โดยอาจจะจบการทำงานเลยก็ได้ สามารถกำหนดจำนวนขั้นตอนที่ต้องการ Simulate ไปได้ โดยคลิกเปลี่ยนตัวเลขที่ด้านล่างปุ่ม คือถ้าเป็นเลข 50 แสดงว่าเมื่อคลิกแล้วหนึ่งทีขั้นตอนการทำงานจะ Simulate ไป 50 ขั้นตอน จากที่การ Simulate ปกติจะไปได้แค่ทีละ 1 ขั้นตอน หากขั้นตอนการทำงานของระบบน้อยกว่าจำนวนขั้นตอนที่ต้องการคลิกเพื่อ Simulate ไป เมื่อคลิกเพื่อ Simulate ระบบก็จะ Simulate ถึงจบการทำงานของระบบทันที



รูปที่ 2.11 : Simulation2



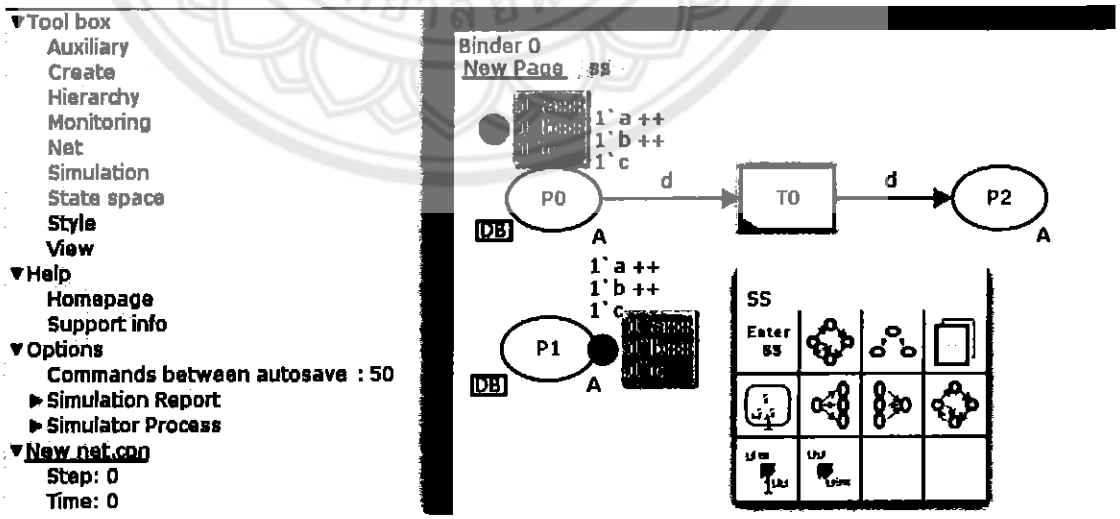
รูปที่ 2.12 : Simulation3



รูปที่ 2.13 : Simulation4

จากรูปที่ 2.11, 2.12, 2.13 คือภาพของขั้นตอนการ Simulate หากใช้ปุ่มที่ 4 คลิกก็จะไปทีละขั้นตอน โดยคลิกหนึ่งที คือไปหนึ่งขั้นตอนและจะไม่ไปต่อจนกว่าจะคลิกต่อเรื่อยๆจนจบการทำงาน แต่หากใช้ปุ่มที่ 5 ก็จะไปทีละขั้นตอนเช่นเดียวกันแต่จะไปขั้นตอนต่อไปโดยอัตโนมัติจนจบการทำงาน แต่หากใช้ปุ่มที่ 6 หากการทำงานมีขั้นตอนที่สั้นไม่ยาวมากเมื่อคลิกเพื่อ Simulate จากภาพที่ 2.10 ที่ยังไม่เริ่มทำงาน จะกลายเป็นภาพที่ 2.13 คือจบการทำงานเลย

-State space



รูปที่ 2.14 : State space

แถวที่ 1 จากซ้ายไปขวารูปที่ 2.14

-ใช้เพื่อเข้ากระบวนการทำ State space

-ใช้คำนวณเกี่ยวกับ State space เช่นจำนวน Node เป็นต้น

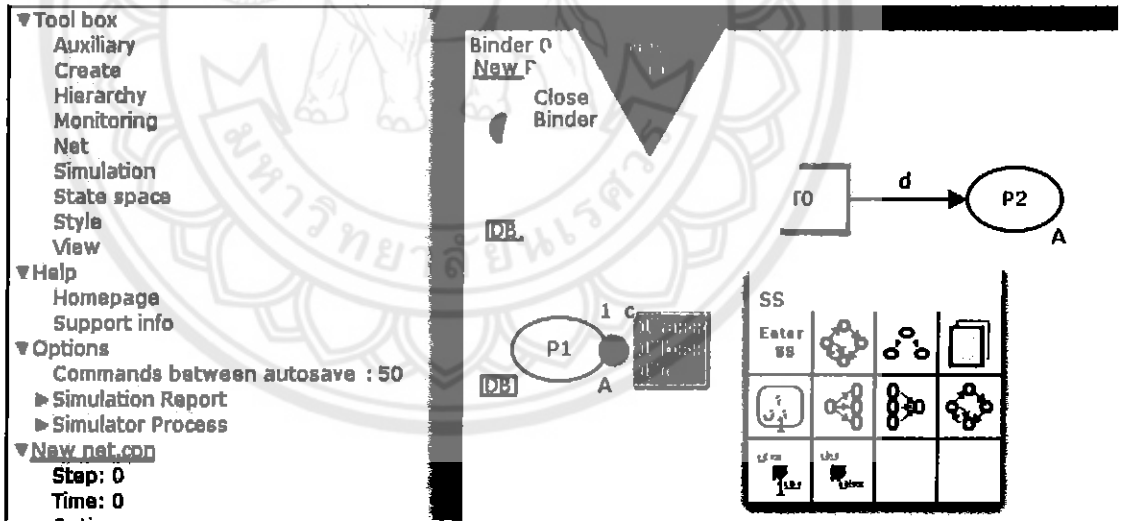
-ใช้คำนวณหากราฟ SCC

แถวที่ 2 จากซ้ายไปขวารูปที่ 2.14

-ใช้สร้างกราฟ State space โดยจะเริ่มสร้างจากจุดไหนสามารถกำหนดได้โดยเปลี่ยนตัวเลขใต้ปุ่ม

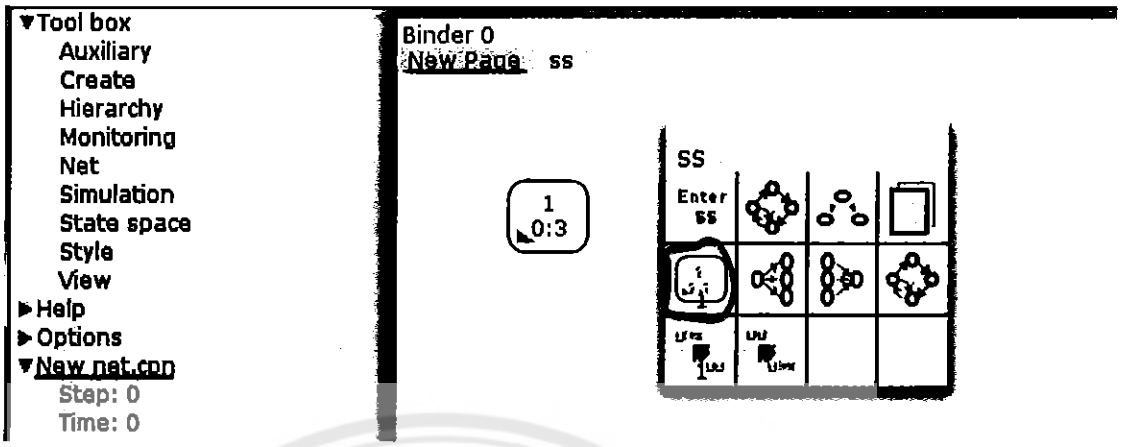
-ใช้สร้างเส้นทางจาก State ปัจจุบัน ไปยัง State ถัดไป

-ใช้สร้างเส้นทางจาก State ปัจจุบัน กลับไปยัง State ก่อนหน้า



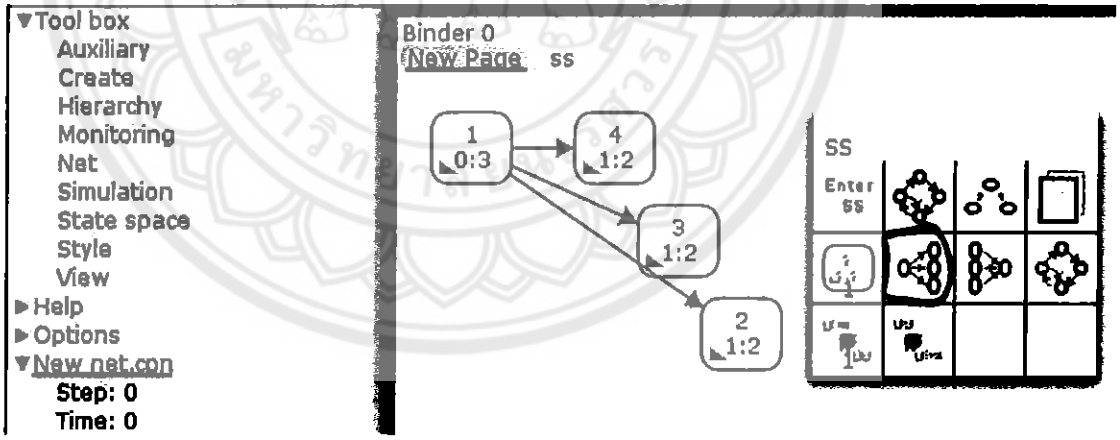
รูปที่ 2.15 : วิธีสร้างหน้าใหม่

เมื่อต้องการสร้างกราฟ State space โดยใช้หน้าใหม่ ขั้นแรกก็ทำการสร้างหน้าใหม่ โดยคลิกขวาที่แถบขาวด้านบน Page แล้วเลือก New page ดังรูปที่ 2.15



รูปที่ 2.16 : เริ่มสร้างกราฟ State space

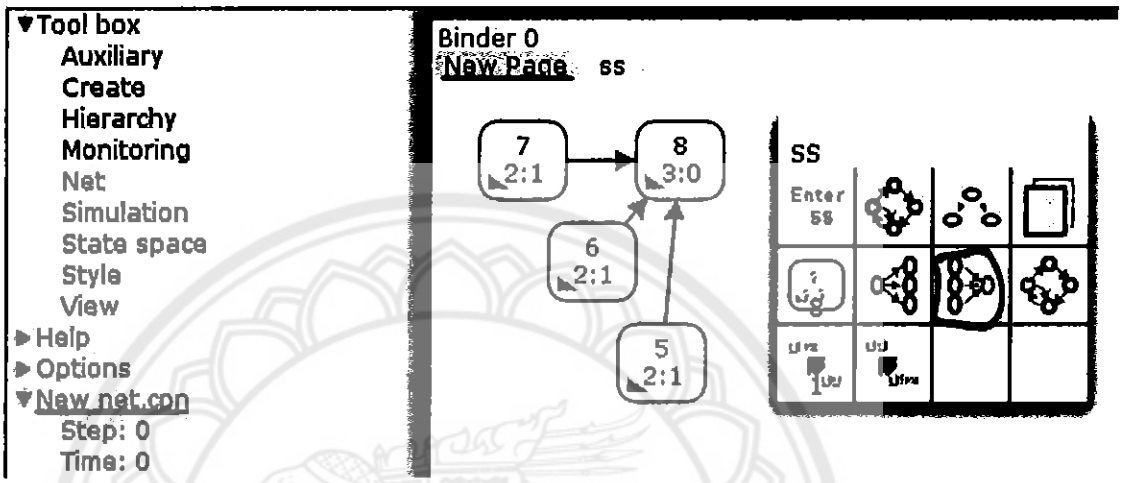
จากนั้นจะได้หน้าใหม่ดังรูปที่ 2.16 โดยสามารถเปลี่ยนชื่อ Page ได้ โดยคลิกที่ชื่อ Page ด้านบน จากนั้นใช้ State space Tool โดยคลิกที่ปุ่มในวงกลมสีแดงดังภาพที่ 2.16 โดยสามารถเปลี่ยนจุดเริ่มที่จะสร้างได้ โดยเปลี่ยนเลขด้านล่างของปุ่ม จากนั้นคลิกลงบน Page ก็จะได้จุดเริ่มที่ต้องการสร้างกราฟ State space ดังภาพที่ 2.16



รูปที่ 2.17 : Next step of State space

จากรูปที่ 2.17 หากต้องการสร้างกราฟที่เป็นเส้นทางต่อไปจากจุดปัจจุบัน ให้ใช้ปุ่มที่อยู่ในวงกลมสีแดงในรูปที่ 2.17 คลิกที่ Node ที่ต้องการ จากนั้นก็จะแสดงเส้นทางไปยัง Node ต่อไปออกมา และสามารถดูจำนวน Node ก่อนหน้าและ Node ต่อไปได้จากด้านล่าง Node โดยตัวหน้าคือ จำนวน

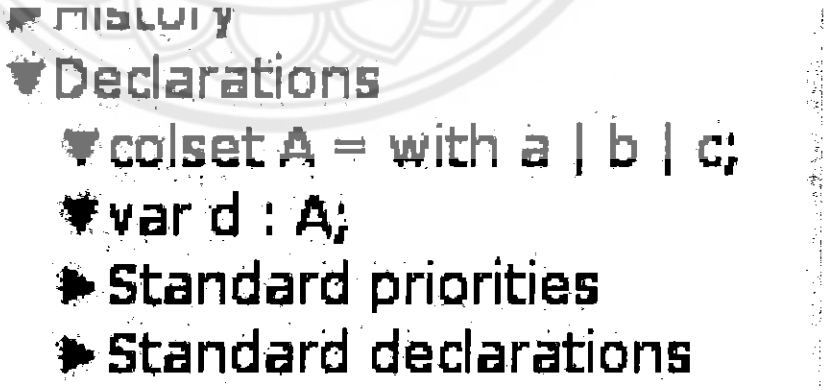
Node ก่อนหน้า และตัวหลัง คือ จำนวน Node ถัดไป ตัวอย่างในรูปที่ 2.17 จาก Node 1 คือ 0:3 หมายความว่ามี Node ก่อนหน้า 0 Node และมี Node ถัดไป 3 Node



รูปที่ 2.18 : Previous step of State space

หากต้องการแสดงเส้นทางก่อนหน้า Node ปัจจุบัน ให้ใช้ปุ่มในวงกลมสีแดงในรูปที่ 2.18 จากนั้นคลิกที่ Node ที่ต้องการ ก็จะแสดงเส้นทางก่อนหน้าของ Node ที่คลิกออกมา

-Declaration



รูปที่ 2.19 : Declaration

เป็นการประกาศตัวแปรที่ต้องการใช้ในระบบที่สร้างขึ้นมา โดยจากระบบตัวอย่างในรูปที่ 2.10 จะประกาศตัวแปรได้ดังรูปที่ 2.19 โดย สิ่งที่ประกาศเพิ่มหลังจากการสร้าง New page ก็คือ

Colset A = with a | b | c; หมายถึง ใน set A จะมีค่า a, b, และ c (บรรทัดหนึ่ง)

Var d : A; หมายถึง d มีค่าเท่ากับค่าใน set A (บรรทัดสอง)

การนำไปใช้ดังภาพที่ 2.10 คือ ใน P0, P1, และ P2 ได้ถูกกำหนดให้เป็นที่อยู่ของ set A ดังนั้นข้อมูลที่เข้าออก P0, P1, และ P2 จะต้องเป็นค่าที่อยู่ใน set A เท่านั้น และเส้นลูกศรมีการกำหนดว่าค่าที่สามารถผ่านเส้นทางนี้ได้ต้องมีค่าอยู่ใน set A เท่านั้น โดยสิ่งที่นำมาใส่เพื่อกำหนดต้องเป็นตัวแปร ไม่ใช่ set ดังนั้นจึงต้องมีการกำหนดให้ ตัวแปร d มีค่าเหมือนค่าใน set A แล้วนำ d มาใส่ตรงเส้นเพื่อกำหนดว่าข้อมูลที่จะไหลผ่านเส้นนี้ต้องเป็นข้อมูลที่อยู่ใน set A เท่านั้น

2.6.3 Paper Al-Azzani

จากวิธีการที่ใช้ในการตรวจสอบความปลอดภัยของโปรโตคอลยืนยันตน ได้อ้างอิงมาจากวิธีใน Paper Al-Azzani โดยได้ศึกษาวิธีการวิเคราะห์เพื่อพัฒนามาใช้กับโปรโตคอล MP-Auth ซึ่งเป็นโปรโตคอลตัวอย่าง เพื่อให้เห็นว่าวิธีการดังกล่าวสามารถนำมาใช้วิเคราะห์ความปลอดภัยของโปรโตคอลยืนยันตนตัวอื่นได้

2.6.4 โปรแกรมประยุกต์

โปรแกรมประยุกต์ เป็นโปรแกรมที่เขียนขึ้นโดยใช้ภาษา Java เพราะเป็นภาษาที่มีฟังก์ชันที่รองรับการเขียนโปรแกรม Cryptography โดยจะทำการเขียนโปรแกรม Encryption เพื่อนำมา Encrypt ข้อมูลที่เป็น Plaintext โดยการ Encryption จะเป็นแบบ Asymmetric Key ซึ่งในภาษา Java จะมีฟังก์ชันที่ใช้ในการสร้างคีย์คู่ ซึ่งจะได้ทั้ง Public Key และ Private Key มาใช้งานในกระบวนการของ Asymmetric Key ถ้าสมมติว่ามีการดักจับข้อมูลเกิดขึ้น และได้ข้อมูลที่ถูกเข้ารหัสมา ก็จะเอา Public Key มา Encrypt ข้อมูล Plaintext ที่ทำการ Random ขึ้นมาเพื่อเทียบกับข้อมูลที่เข้ารหัสที่ถูกดักจับได้ เพื่อหาระยะเวลาในการเดาข้อมูลที่ถูกเข้ารหัส แล้วนำไปเปรียบเทียบกับช่วงเวลาที่ข้อมูลนั้นมี

ความสำคัญอยู่ และวิเคราะห์ว่าข้อมูลนั้นมีความปลอดภัยหรือไม่เมื่อเทียบระหว่างระยะเวลาที่เคาข้อมูลที่ถูกเข้ารหัสได้สำเร็จ กับระยะเวลาที่ข้อมูลนั้นๆมีความสำคัญอยู่

ในการเขียนโปรแกรมในการสร้างคีย์ หรือกุญแจ และการเข้ารหัส ถอดรหัส จะมีฟังก์ชันจาก Package ต่างๆ ของภาษา Java ที่ช่วยในการเขียนดังนี้

-การสร้าง คีย์คู่ หรือกุญแจคู่ จะใช้ KeyPairGenerator เป็น class ในการสร้าง คีย์คู่ ซึ่งจะมี KeyPair เป็น class ใช้เป็นตัวเก็บ คีย์คู่ จาก KeyPairGenerator และเวลาเรียกใช้ คีย์คู่ ซึ่งมีทั้ง Public Key และ Private Key จาก KeyPair อีกที เวลาเขียน Public Key ลง Text ไฟล์ต้องทำข้อมูลจากชนิด Key ให้เป็นชนิด Byte Array ก่อน ที่ต้องเก็บลง Text ไฟล์เพราะการสร้างคีย์คู่โดยใช้ KeyPairGenerator จะได้คีย์คู่ที่สุ่มให้เรื่อยๆ อาจได้ Public Key เหมือนกันหรือไม่เหมือนกันก็ได้ ดังนั้นจึงต้องมีการเก็บลง Text ไฟล์เพื่อที่เวลาใช้งาน Public Key จะได้เป็นคีย์เดิม

-การเข้ารหัสต้องทำให้ข้อความที่ต้องเข้ารหัสในที่นี้คือ ตัวอักษร เป็นข้อมูลชนิด Byte Array ก่อน และ Public Key ที่อ่านจาก Text ไฟล์ต้องเรียกใช้โดย PublicKey ซึ่งเป็น interface อยู่ใน package java.security ซึ่งจะต้องใช้ KeyFactory ซึ่งเป็น class ช่วยในการสร้าง Public Key ขึ้นมาใหม่จากข้อมูลที่อ่านจากไฟล์ Text จากนั้นเมื่อได้ ตัวอักษรชนิด Byte Array และ Public Key แล้วจะใช้ Cipher ซึ่งเป็น class ที่สามารถใช้เข้ารหัสและถอดรหัสจาก Package javax.crypto มาใช้เข้ารหัสข้อความโดยจะได้ข้อมูลชนิด Byte Array เมื่อเข้ารหัสเสร็จ

-การเปรียบเทียบเพื่อทำการ Brute Force หากจะใช้ข้อความที่เข้ารหัสเปรียบเทียบต้องทำให้จากข้อมูลชนิด Byte Array เป็นชนิด String ก่อนโดยใช้ฟังก์ชัน BASE64Encoder ซึ่งได้จากการ import sun.misc.BASE64Encoder แล้วจะได้ข้อมูลที่เป็นชนิด String และใช้ compareTo ซึ่งเป็น Method ในการเปรียบเทียบ String เพื่อใช้เปรียบเทียบในการ Brute Force

บทที่ 3
การวิเคราะห์และการออกแบบ

3.1 การวิเคราะห์ปัญหา

ปัญหา คือ ต้องการวิธีที่สามารถใช้ในการพิสูจน์ว่า โปรโตคอลยืนยันตนมีความปลอดภัยจริงหรือไม่ ดังนั้นจึงมีการคิดหาวิธีที่จะใช้พิสูจน์เกี่ยวกับ โปรโตคอลยืนยันตน โดยวิธีที่เลือกมาใช้ คือ วิธีการจำลองการทำงานของโปรโตคอลยืนยันตนลงบนโปรแกรม CPN Tools และทำการสร้างเป็น State Space เพื่อทำการวิเคราะห์ว่า โปรโตคอลยืนยันตนมีความปลอดภัยหรือไม่ และอีกวิธีที่เลือกมาพิสูจน์ คือ การเขียนโปรแกรมเพื่อที่จะทำการถอดรหัสข้อมูล ถ้าในกรณีที่ข้อมูลใดๆที่มีการเข้ารหัสไว้เกิดการรั่วไหลจาก โปรโตคอลยืนยันตนเกิดขึ้น โดยจะทำการบันทึกเวลาในการถอดรหัสได้สำเร็จ และนำเวลาที่ได้ไปเปรียบเทียบกับช่วงเวลาที่ข้อมูลนั้นยังมีประโยชน์อยู่ โดยถ้ามีการถอดรหัสได้สำเร็จในช่วงเวลาที่ข้อมูลนั้นยังมีประโยชน์อยู่ แสดงว่า โปรโตคอลยืนยันตนที่ใช้อยู่นั้น มีความไม่ปลอดภัยเกิดขึ้น

3.2 การออกแบบการจัดสร้างโครงงาน

3.2.1 ศึกษาการทำงานของโปรโตคอล MP-Auth การใช้งาน CPN-Tools และ Java cryptography

3.2.2 ใช้โปรแกรม CPN-Tools ออกแบบโปรโตคอล MP-Auth มี 3 แบบ

3.2.2.1 แบบที่ 1 จะเป็นระบบการทำงานของโปรโตคอล MP-Auth แบบปกติ

3.2.2.2 แบบที่ 2 จะสมมติให้ Browser พยายามถอดรหัสข้อความเท่าที่สามารถถอดได้ที่ผ่านเข้าออก Browser และเก็บค่าเข้าฐานข้อมูลของ Browser

3.2.2.3 แบบที่ 3 จะเพิ่ม Oracle เป็นตัวช่วยถอดรหัสให้กับ Browser

3.2.3 เขียน Query และทำ State Space เพื่อหา State ที่ไม่ปลอดภัย

3.2.4 เขียนโปรแกรม Brute force จากภาษา Java เพื่อหาระยะเวลาในการเดารหัส ที่ถูกเข้ารหัสโดยใช้ Public Key โดยรหัสที่เดามีขนาดตั้งแต่ 2 ตัวขึ้นไป

3.2.5 หลังจากได้เวลามาแล้ว วิเคราะห์ว่ารหัสควรมีความยาวเท่าไหร่ถึงจะปลอดภัย

3.3 การดำเนินการสร้าง

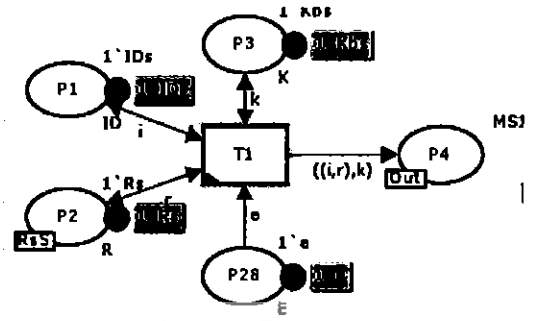
ตัวแปรกำหนดดังนี้ (ใช้ทั้ง 3 แบบ)

```
▼var k:K;
▼var i,i1,i2:ID;
▼var r,r1,r2:R;
▼var re,re1,re2,re3:RE;
▼var rr:RR;
▼var fr,fr1,fr2:FR;
▼var pa,pa1,pa2:P;
▼var m1:M1;
▼var ms1:MS1;
▼var m2:M2;
▼var cm2:CM2;
▼var cr:CR;
▼var ms3:MS3;
▼var ms4:MS4;
▼var m4:M4;
▼var ms5:MS5;
▼var db:DB;
▼fun DecryptionKey(k:K):K = case k of Kbs=> Kbs
  | Kms => Kms | PubS => PriS | PriS => PubS;
▼fun Equal(r:R):FR = case r of Rs => FRs
  | Rm => FRm;
▼fun Equal2(rr:RR):FR = case rr of RR => FRR;
▼fun GenKey(fr:FR):K = case fr of FRR => Kms;
```

```
▼MP-Auth declarations
  ▼colset ID = with IDu | IDs;
  ▼colset R = with Rs | Rm;
  ▼colset FR = with FRs | FRm | FRR;
  ▼colset P = with p;
  ▼colset K = with Kms | Kbs | PubS| PriS;
  ▼colset RE = with Success | Fail;
  ▼colset M1 = product ID*R;
  ▼colset RR = product R*R;
  ▼colset CM1 = product M1*K;
  ▼colset M2 = product FR*ID*P;
  ▼colset CM2 = product M2*K;
  ▼colset CR = product R*K;
  ▼colset M3 = product CR*CM2;
  ▼colset CM3 = product M3*K;
  ▼colset M4 = product FR*K;
  ▼colset CM4 = product M4*K;
  ▼colset MS1 = CM1;
  ▼colset MS2 = M1;
  ▼colset MS3 = M3;
  ▼colset MS4 = CM3;
  ▼colset MS5 = CM4;
  ▼colset MS6 = M4;
  ▼colset E = with e;
  ▼colset DB = union dbID:ID + dbR:R
    + dbFR:FR + dbP:P + dbK:K + dbRE:RE
    + dbM1:M1 + dbCM1:CM1 + dbM2:M2
    + dbCM2:CM2 + dbCR:CR + dbM3:M3
    + dbCM3:CM3 + dbM4:M4 + dbCM4:CM4
    + dbMS1:MS1 + dbMS2:MS2 + dbMS3:MS3
    + dbMS4:MS4 + dbMS5:MS5 + dbMS6:MS6
    + dbRR:RR;
```

รูปที่ 3.1 : การกำหนดตัวแปร

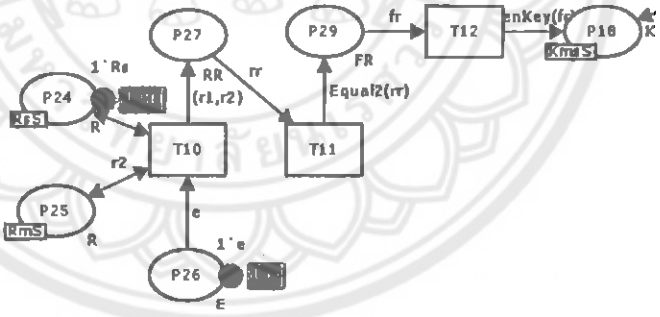
การใช้ E เพื่อทำให้ส่งข้อมูลเพียงครั้งเดียว



รูปที่ 3.2 : การใช้ E เพื่อทำให้ Transition ส่งข้อมูลแค่ครั้งเดียว

จากรูปจากเห็นว่ามีลูกศรสองหัวอยู่เมื่อทำการส่งข้อมูลไปแล้ว ข้อมูลจะ**ไม่สูญหายไปไหนอยู่**ที่ place จะเกิดการส่งข้อมูลซ้ำกันไปเรื่อย จึงต้องมี E เข้ามาช่วยเพื่อให้ส่งข้อมูลเพียง**ครั้งเดียว**

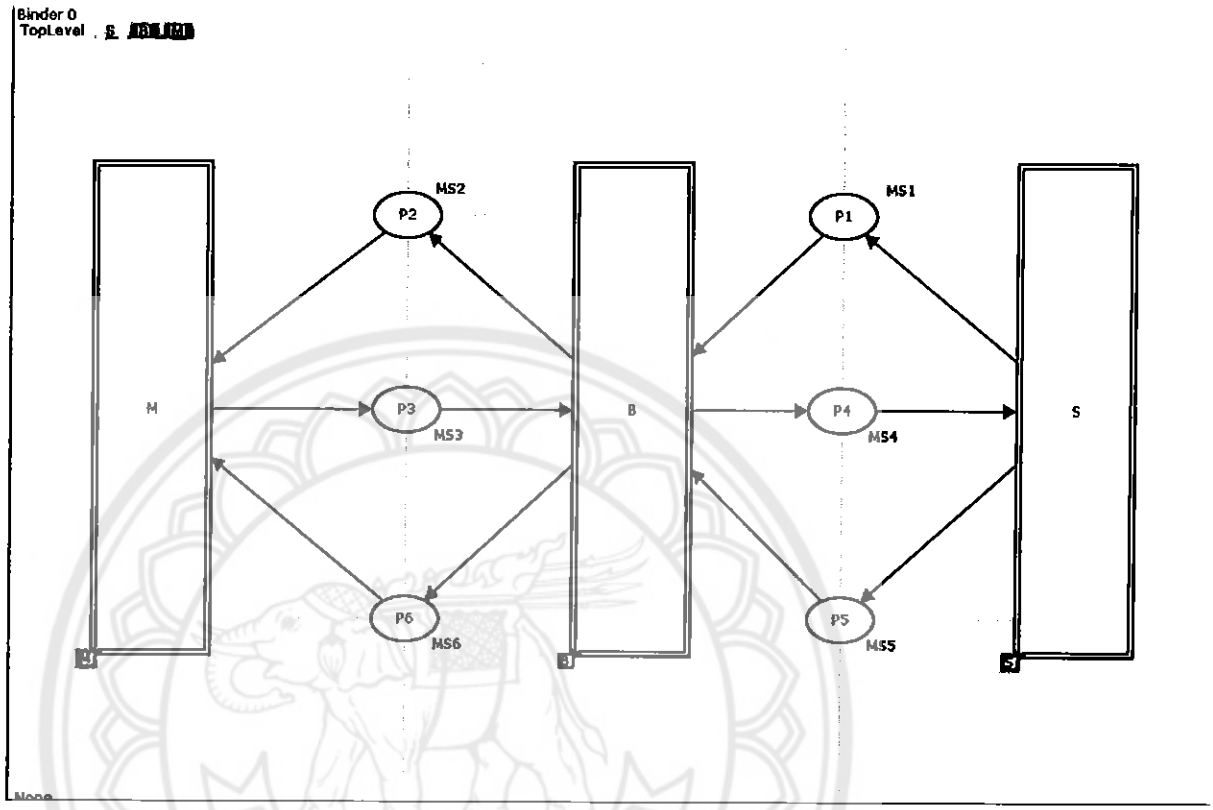
การสร้าง Kms (กุญแจที่ใช้ระหว่าง Mobile กับ Server)



รูปที่ 3.3 : การสร้าง Kms

Transition T10 จะทำการรวม R ทั้ง 2 ตัวและส่งออกไปให้ place P27 จากนั้นส่งไปให้ Transition T11 โดยผ่านเส้นทาง rr (r1,r2) Transition T11 ส่งไปให้ place 29 โดยเข้า Function เปลี่ยนจาก rr ไปเป็น fr (r ที่เข้า hash function) แล้ว นำ fr ที่ได้มา Genkey เพื่อสร้าง Kms (กุญแจที่ใช้ระหว่าง Mobile กับ Server) มาใช้

MP-Auth แบบปกติ



รูปที่ 3.4 : MP-Auth Top-Level

M หมายถึง Mobile, B หมายถึง Browser, S หมายถึง Server

P1 เป็นการส่งข้อความ (MS1) ที่ได้รับจาก Server(S) แล้วส่งไปให้ Browser(B)

P2 เป็นการส่งข้อความ (MS2) ที่ได้รับจาก Browser(B) แล้วส่งไปให้ Mobile(M)

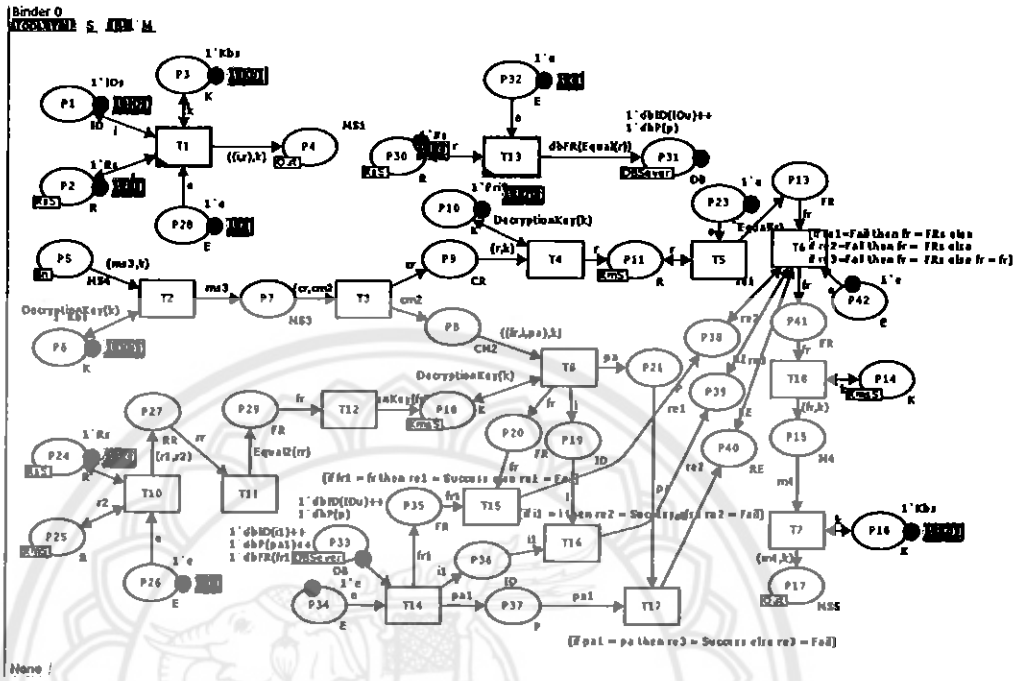
P3 เป็นการส่งข้อความ (MS3) ที่ได้รับจาก Mobile(M) แล้วส่งไปให้ Browser(B)

P4 เป็นการส่งข้อความ (MS4) ที่ได้รับจาก Browser(B) แล้วส่งไปให้ Server(S)

P5 เป็นการส่งข้อความ (MS5) ที่ได้รับจาก Server(S) แล้วส่งไปให้ Browser(B)

P6 เป็นการส่งข้อความ (MS6) ที่ได้รับจาก Browser(B) แล้วส่งไปให้ Mobile(M)

Server (S)



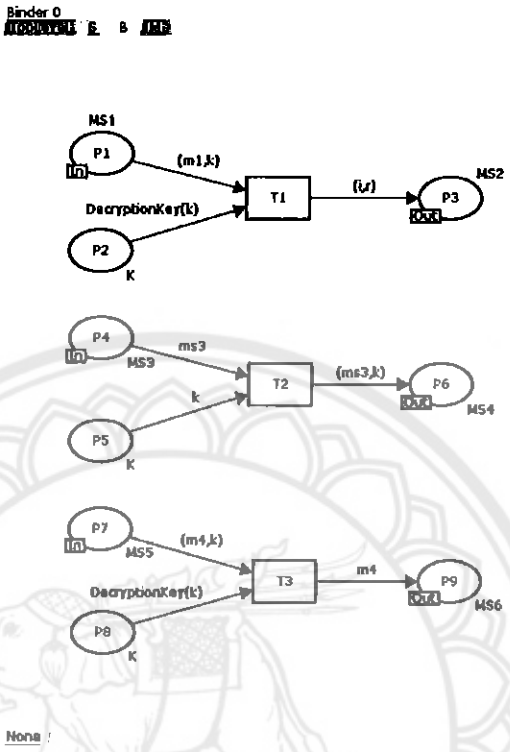
รูปที่ 3.5 : Server

Transition T1 นำ IDs (ID Server) กับ Rs (ค่า Random ของ Server) มารวมกันจากนั้นจะ Encrypt ด้วย Kbs (กุญแจที่ใช้ระหว่าง browser กับ Server) แล้วส่งไปให้ Browser ซึ่ง place P4 ของ Server จะเชื่อมกับ place P1 ของ Top-Level Transition T13 ทำการแปลง Rs ให้เป็น FRs แล้วเก็บเข้า database ของ Server

Transition T10 Transition T11 และ Transition T12 เป็นการสร้าง Kms (กุญแจที่ใช้ระหว่าง Mobile กับ Server) ซึ่งได้กล่าวไว้แล้วตอนต้น

Transition T2 รับข้อความเข้ามาแล้ว Decrypt ด้วย Kbs Transition T3 รับข้อความที่ทำ Decrypt แล้ว ข้อความที่ได้ แบ่งเป็น 2 ข้อความจึงแยกออกเป็น 2 ทางคือ cr กับ cm2 ซึ่ง cm2 ประกอบไปด้วย IDu (ID User) p (Password User) และFRs (Rs ที่เข้า hash function) ไปตรวจสอบกับ database ของ server ว่าถูกต้องหรือไม่ cr นำไป Decrypt ด้วย PriS (Private Key ของ Server) จะส่งข้อความไป Encrypt ด้วย Kms Kbs และส่งออกให้ Browser ได้นั้น IDu p และ FRs ต้องถูกต้อง ถ้าไม่ถูกต้องจะทำการจบโปรแกรม ข้อความจะค้างอยู่ที่ place P13

Browser(B)



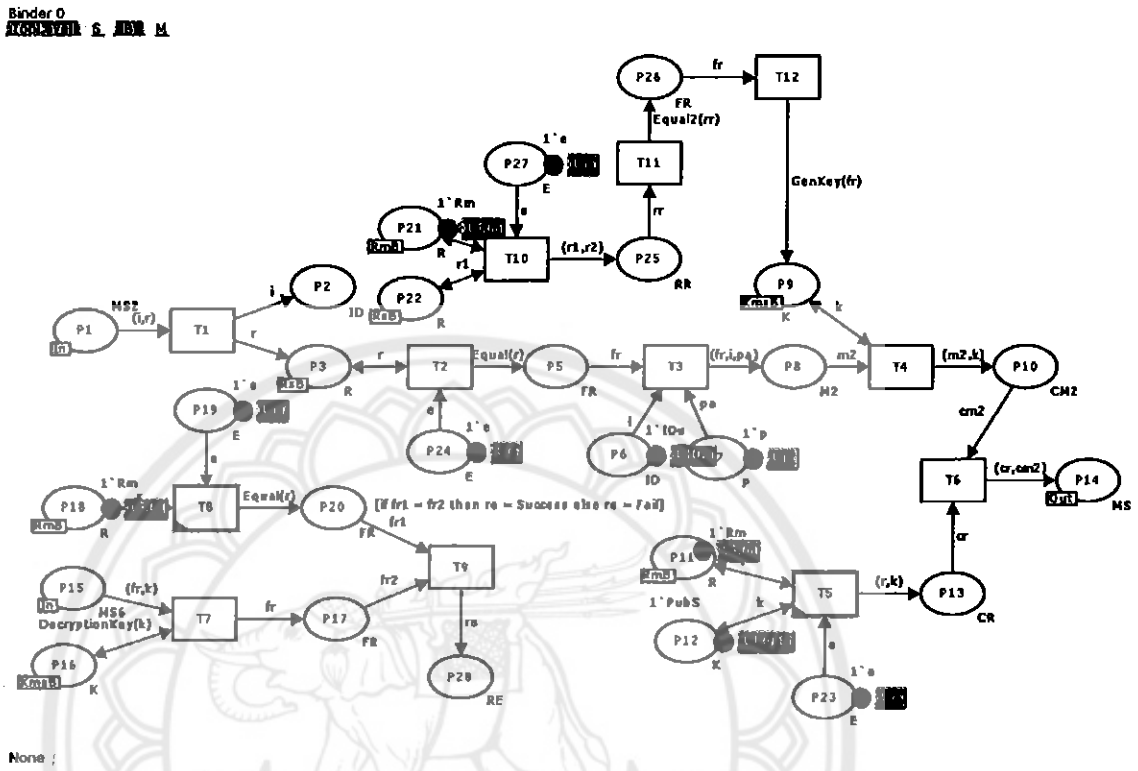
รูปที่ 3.6 : Browser

Transition T1 จะนำข้อความที่ได้รับจาก Server มา Decrypt และส่งออกไปให้ Mobile

Transition T2 จะนำข้อความที่ได้รับจาก Mobile มา Encrypt และส่งออกไปให้ Server

Transition T3 จะนำข้อความที่ได้รับจาก Server มา Decrypt และส่งออกไปให้ Mobile

Mobile(M)



รูปที่ 3.7 : Mobile

Transition T1 รับข้อความ (i,r) มาจาก Browser แล้วแยกออกเป็น 2 ทางเป็น i กับ r จากนั้นเปลี่ยน r ให้เป็น fr ทำการใส่ ID และ Password นำ fr ID และ Password มารวมกันแล้ว Encrypt ด้วย Kms ได้ข้อความที่ 1 ที่ Transition T5 Rm (Random ของ Mobile) เข้า Encrypt ด้วย PubS (Public Key ของ Server) ได้ข้อความที่ 2 Transition T6 นำข้อความที่ 1 และ 2 มารวมกันแล้วส่งไปให้ Browser

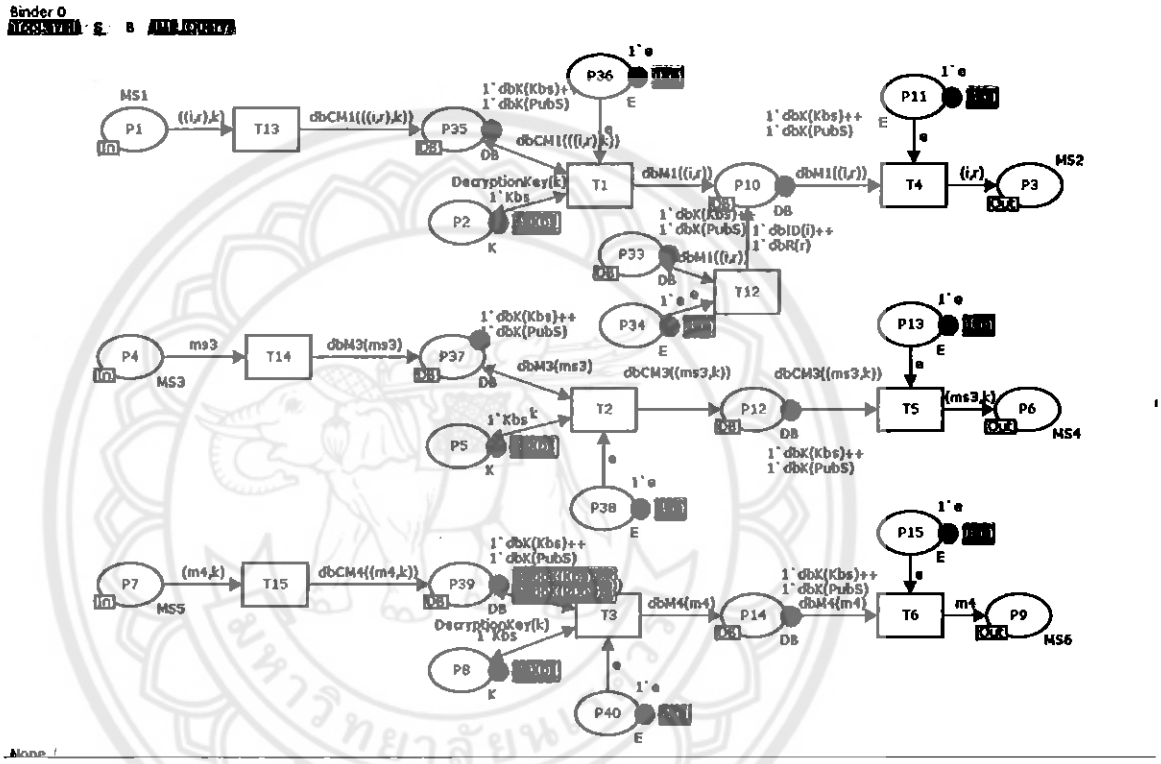
Transition T10 Transition T11 และ Transition T12 เป็นการสร้าง Kms (กุญแจที่ใช้ระหว่าง Mobile กับ Server) ซึ่งได้กล่าวไว้แล้วตอนต้น

Transition T9 ทำหน้าที่ตรวจสอบ ข้อความที่ได้รับมาแล้วทำการ Decrypt จะตรงกับของ Mobile หรือไม่ ถ้าตรงจะ Success ไม่ตรงจะ Fail

MP-Auth แบบ Browser พยายามถอดรหัส

แบบนี้ Server(S) และ Mobile(M) จะเหมือนแบบที่ 1 MP-Auth แบบปกติ เปลี่ยน Browser ให้เป็น Browser ที่พยายามถอดรหัสทุกอย่างที่เป็นไปได้

Browser(B)

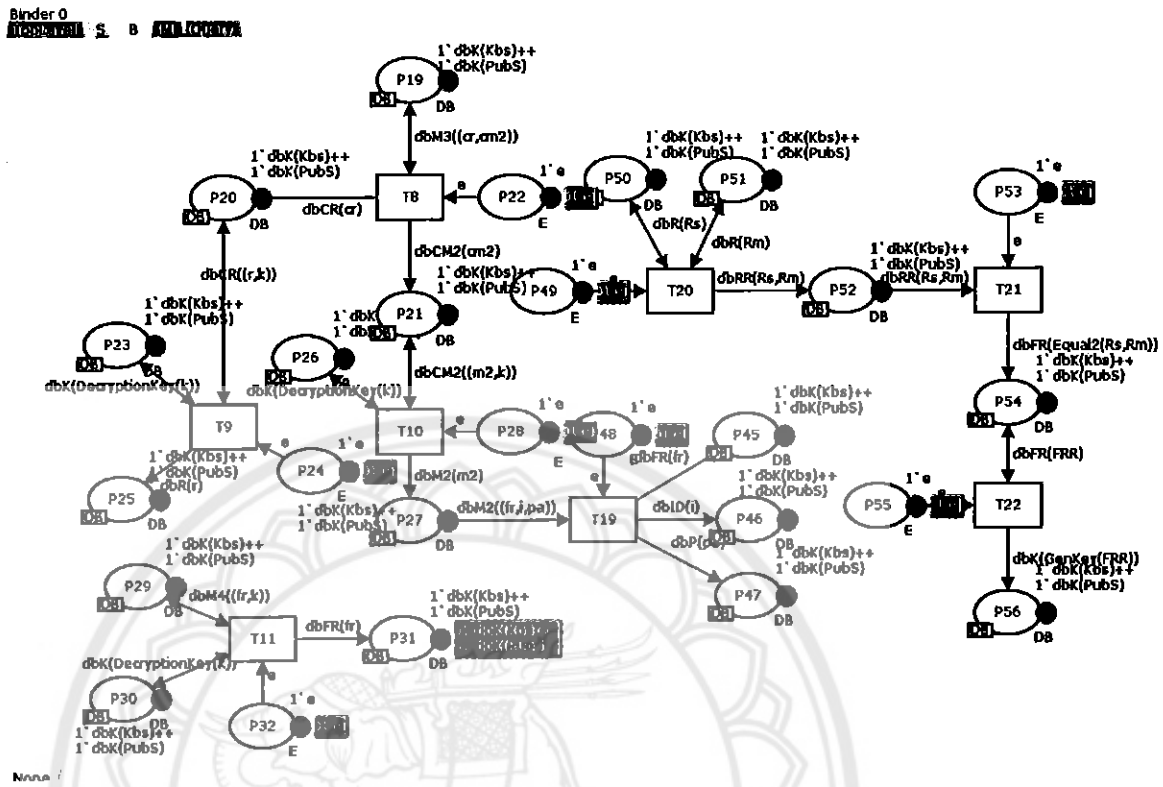


รูปที่ 3.8 : Browser แบบมี Database เก็บข้อความ

Transition T13 และ Transition T15 รับข้อความมาเก็บไว้ใน database จากนั้น Decrypt ด้วย Kbs แล้วส่งข้อความที่ Decrypt ได้ไปเก็บไว้ใน database ให้ database ส่งข้อความออกไป

Transition T14 รับข้อความมาเก็บไว้ใน database จากนั้น Encrypt ด้วย Kbs แล้วส่งข้อความที่ Encrypt ไปเก็บไว้ใน database ให้ database ส่งข้อความออกไป

Transition T12 เป็นการพยายามแยก ID กับ R ที่อยู่ใน database ออกจากกันแล้วเก็บลง database



รูปที่ 3.9 : Browser พยายามถอดรหัสทุกทางที่เป็นไปได้ใน Database

Transition T20 Transition T21 และ Transition T22 เป็นการสร้าง Kms **(กุญแจที่ใช้ระหว่าง** Mobile กับ Server) ไปเก็บไว้ใน Database

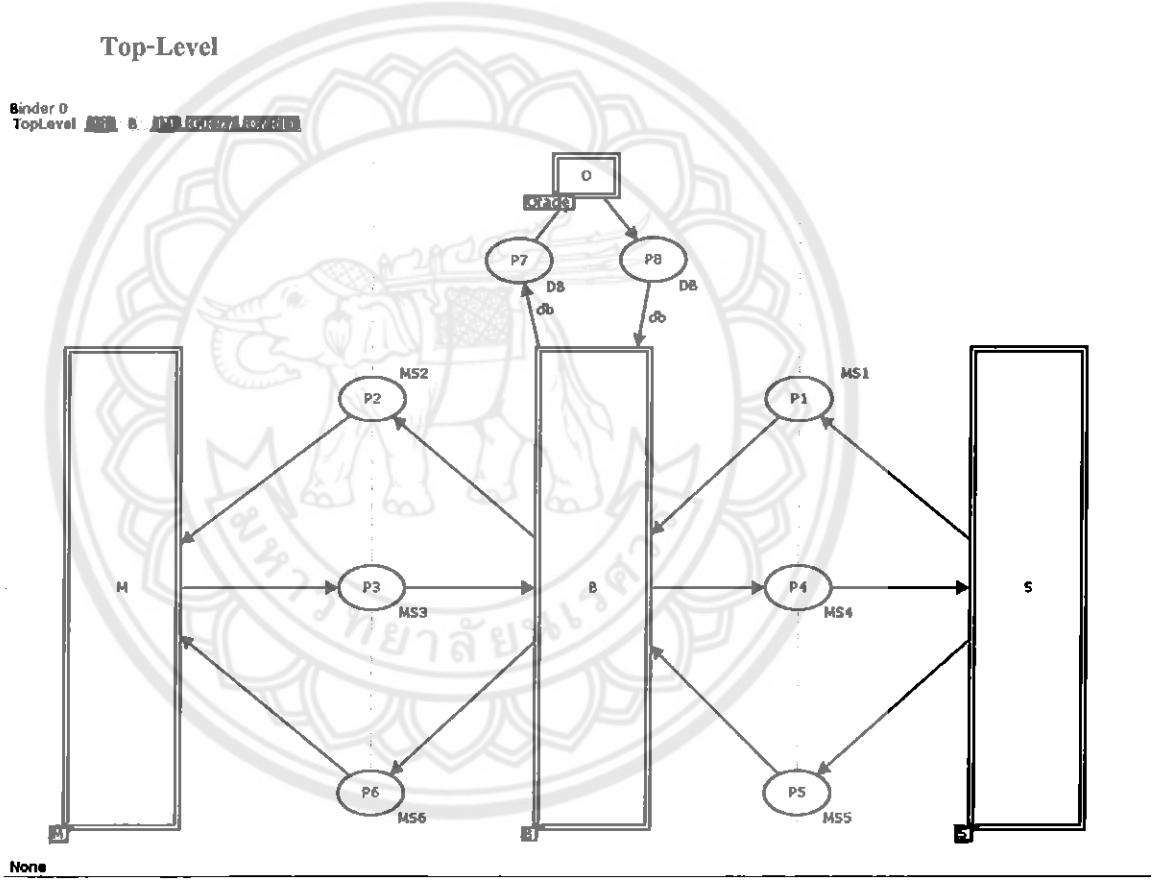
Transition T10 พยายามถอดรหัสข้อความที่ได้รับเพื่อหา fr (ตัวเลข **Random ที่เข้า hash** function) i (ID) และ pa (Password)

Transition T8 พยายามถอดรหัสข้อความที่ได้รับเพื่อหา r (ตัวเลข Random)

Transition T11 พยายามถอดรหัสข้อความที่ได้รับเพื่อหา fr (ตัวเลข Random ที่เข้า hash function)

MP-Auth แบบมี Oracle

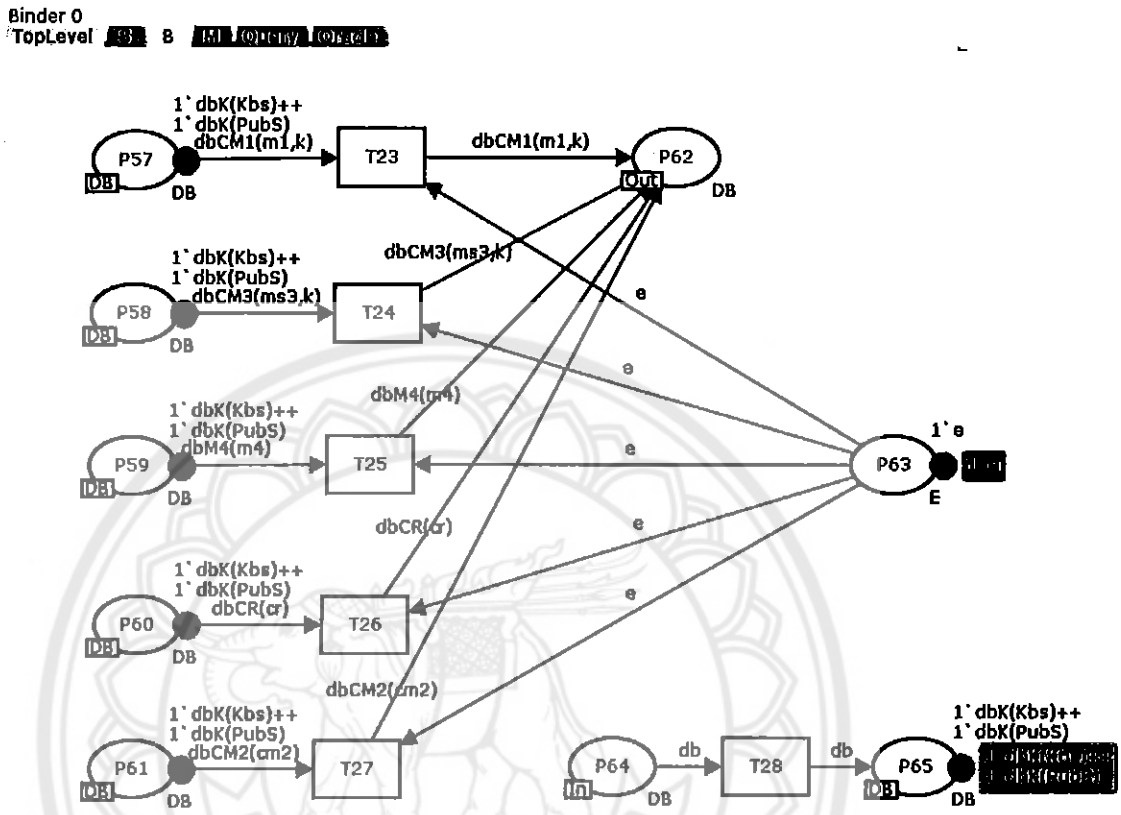
แบบนี้ Server(S) และ Mobile(M) จะเหมือนแบบที่ 1 MP-Auth แบบปกติ ส่วน Browser จะเหมือนแบบที่ 2 MP-Auth แบบ Browser พยายามถอดรหัส แต่จะเพิ่ม Oracle เข้าเพื่อช่วย Browser ถอดรหัสแต่ Oracle ช่วยได้แค่ 1 ครั้งเท่านั้น ซึ่ง Oracle จะมี Key ทุกอย่าง และ Oracle จะถอดรหัสเพียง 1 ครั้งเท่านั้น เช่น dbCM3((((Rm,PubS),((FRs,IDu,p),Kms)),Kbs)) จะถอดแค่ Kbs แล้วส่งไปให้ Browser



รูปที่ 3.10 : Top-Level เพิ่ม Oracle

เพิ่ม Transition Oracle โดยให้เชื่อมต่อกับ Browser(B)

Browser (B) เพิ่มการส่งข้อความไปให้ Oracle

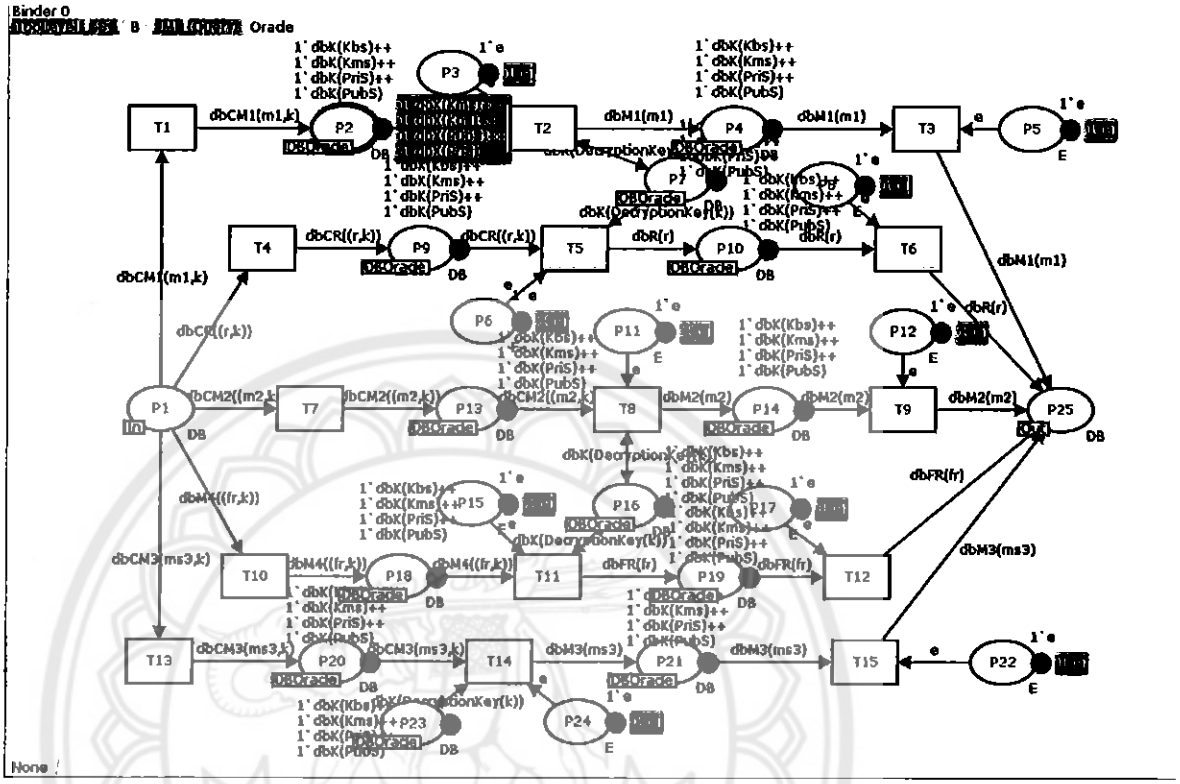


รูปที่ 3.11 : ส่งข้อความใน database ไปให้ Oracle และรับจาก Oracle เข้า database

เป็นการส่ง Cipher Text ให้ Oracle ส่งได้แค่ครั้งเดียวเพราะ E ทำให้ส่งได้**แค่ครั้งเดียว**

Transition T28 รับข้อความจาก Oracle จากนั้นส่งไปเก็บที่ database ของ Browser

Oracle (O)



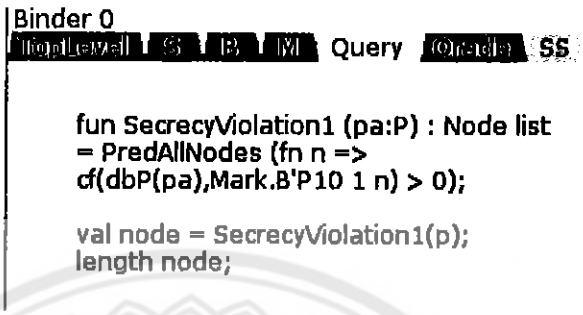
รูปที่ 3.12 : Oracle

DBOracle เป็น Database ของ Oracle เก็บ Key ที่ใช้ถอดรหัสไว้

Place P1 ส่งของมูลที่ได้รับมาจาก Browser ไปตามเส้นทางของข้อความที่ส่งมา เพื่อให้ถอดรหัสแล้วส่งกลับไปให้ Browser

Query

เป็นตัวที่ใช้ค้นหา Password(P) ใน Database ของ Browser ว่ามีหรือไม่



รูปที่ 3.13 : Code ML Language for Query

หาค่า Password ใน B ที่ Place P10 ให้หาทุก node ที่มี password อยู่ จากนั้น show ว่ามีกี่ node

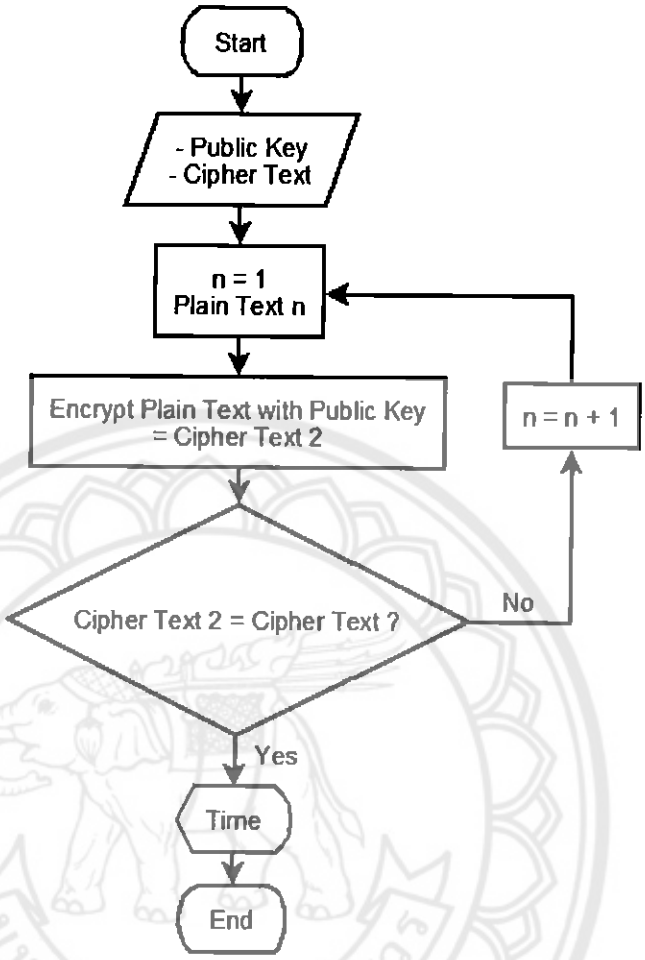
3.4 การใช้งานโปรแกรมประยุกต์

โปรแกรมประยุกต์จัดทำขึ้นเพื่อใช้สนับสนุนข้อสมมติฐานจากการวิเคราะห์ด้วยโปรแกรม CPN Tools โดยจะหาว่า หากข้อมูลที่เข้ารหัสถูกดักจับได้จะใช้เวลาขนาดไหนในการถอดรหัสข้อมูลนั้นๆ โดยจะใช้วิธี Brute Force เพื่อถอดรหัสข้อมูลหลายๆแบบ แล้วนำข้อมูลที่ได้ไปสร้างกราฟ และสมการ เพื่อที่จะได้ใช้สมการในการหาเวลาโดยประมาณตามที่ต้องการ และนำเวลาที่ได้มาเทียบกับระยะเวลาที่ข้อมูลนั้นๆมีค่าอยู่ เพื่อวิเคราะห์ความปลอดภัยของข้อมูลนั้นๆ

โปรแกรมที่ใช้จะมี 3 โปรแกรม

ตารางที่ 3.1 : โปรแกรมประยุกต์

โปรแกรมที่ 1	จะใช้สร้าง Public Key และ Private Key ซึ่งเป็นกุญแจคู่ แล้วบันทึกแค่ Public Key ลง Text ไฟล์
โปรแกรมที่ 2	โปรแกรมที่ 2 จะอ่านค่า Public Key จาก Text ไฟล์ แล้วสุ่มตัวอักษรแล้วนำตัวอักษรที่ได้เข้ารหัสด้วย Public Key แล้วนำค่าที่ได้บันทึกลง Text ไฟล์อีกไฟล์หนึ่ง ซึ่งจะสมมติว่าให้เป็นข้อมูลที่ถูกดักจับได้
โปรแกรมที่ 3	จะอ่านค่าจาก Text ไฟล์ทั้ง 2 Text ไฟล์ คือค่า Public Key และค่าตัวอักษรสุ่มที่เข้ารหัสไว้ จากนั้นจะทำการ Brute Force เอาตัวอักษรที่ได้จากการ Brute Force เข้ารหัสด้วย Public Key แล้วนำไปเทียบกับค่าตัวอักษรที่ถูกเข้ารหัสไว้ที่อ่านมาจาก Text ไฟล์ เมื่อถูกต้องโปรแกรมจะทำการแสดงเวลาที่ใช้ในการ Brute Force ให้เห็น เป็นอันจบโปรแกรม แล้วจึงทำการบันทึกเวลาไว้เพื่อใช้ในการวิเคราะห์ต่อไป



รูปที่ 3.14 : การทำงานของโปรแกรม Brute Force

จากรูปที่ 3.15 การทำงานของโปรแกรมที่ 3 จะเริ่มจากอ่านค่าของ Public Key กับ Cipher Text จาก Text ไฟล์ทั้ง 2 ไฟล์ โดย Public Key มาจากโปรแกรมที่ 1 และ Cipher Text คือตัวอักษรสุ่มที่ถูกเข้ารหัสจากโปรแกรมที่ 2 จากนั้นจะเริ่ม Brute Force โดยจะไล่ตัวอักษรไปเรื่อยๆ เช่น ต้องการตัวอักษร a-z ขนาด 3 Bytes ก็จะได้ aaa aab aac ... ไปจนถึง zzz เป็นต้น โดยจากรูปที่ 3.15 จะกำหนดให้ตัวอักษรกลุ่มแรก คือ Plain Text 1 จากนั้นจะนำ Plain Text 1 ไปเข้ารหัสด้วย Public Key และให้ค่าที่ได้เป็น Cipher Text 2 ดังรูปที่ 3.15 จากนั้นนำ Cipher Text 2 ไปเปรียบเทียบกับ Cipher Text ที่อ่านมาจากไฟล์ หากไม่ถูกต้องก็จะวนกลับไปไล่ตัวอักษรใหม่ โดยในรูปจะให้เป็น Plain Text 2 และทำการเข้ารหัสด้วย Public Key เพื่อตรวจสอบต่อไป จนกว่าจะถูกต้อง หากถูกต้องโปรแกรมจะทำการแสดงเวลาที่ใช้ในการ Brute Force และจบโปรแกรม

บทที่ 4

ผลการทดลอง

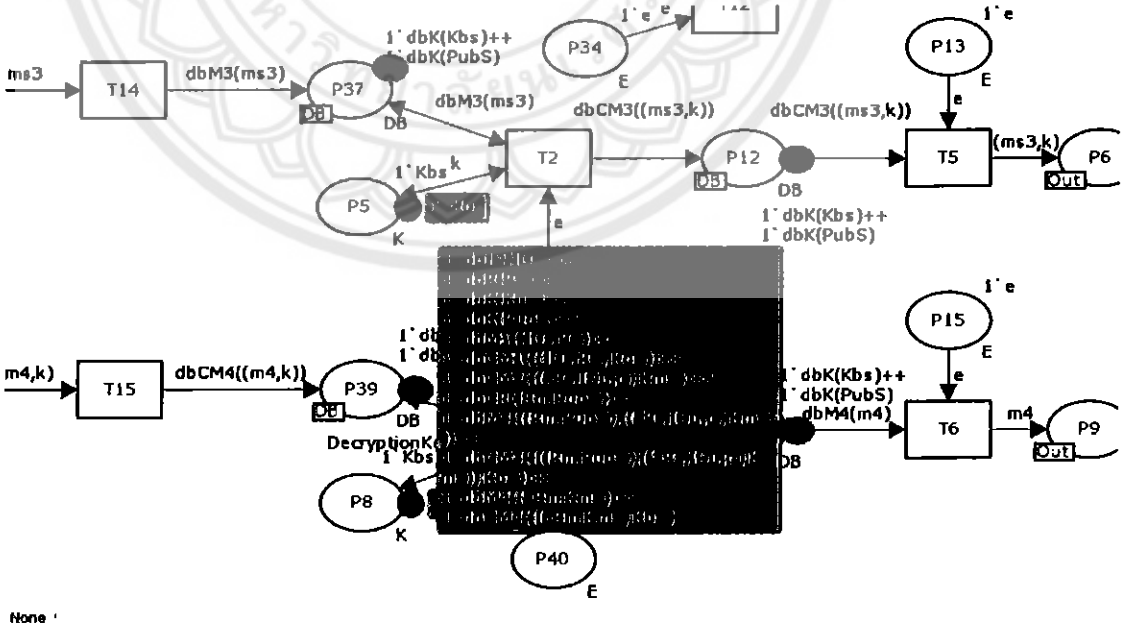
4.1 การทดลองด้วยโปรแกรม CPN Tools

จากการทดลอง ทั้ง 3 แบบ จากโปรแกรม CPN Tools มีดังนี้

4.1.1 ขั้นตอนการทำงานของโปรโตคอล MP-Auth ปกติ

จากรูปที่ 3.6, 3.7, และ 3.8 ในบทที่ 3 ซึ่งเป็นรูปการทำงานภายใน Server, Browser และ Mobile จะเห็นได้ว่าการส่งข้อมูลไม่มีจุดที่เสี่ยงต่อการรั่วไหลของข้อมูลเกิดขึ้นเนื่องจากข้อมูลทุกอย่างจะถูกเข้ารหัสไว้ก่อนที่จะส่งไปจนถึงปลายทางเสมอ ทำให้โปรโตคอล MP-Auth โดยปกติมีความเสี่ยงน้อยที่จะถูกโจมตี

4.1.2 แบบที่ Browser ทำตัวเป็นผู้บุกรุก (Intruder)



รูปที่ 4.1 : ข้อมูลภายในฐานข้อมูลของ Browser ที่ทำตัวเป็น Intruder

โดย Browser จะทำการเก็บและถอดรหัสของข้อมูลทุกอย่างที่ไหลผ่านเข้ามาที่ Browser ก่อนที่จะส่งออกไป จากรูปที่ 4.1 จะเห็นได้ว่า ภายในฐานข้อมูล หรือ Database ของ Browser โดยจะมีการสมมติให้ภายในฐานข้อมูลของ Browser มี Kbs หรือ คีย์ที่ Browser ใช้ติดต่อกับ Server และ PubS หรือ Public Key ของ Server เมื่อมีการสื่อสารโดยใช้โปรโตคอล MP-Auth Browser ก็จะเริ่มทำการเก็บข้อมูลต่างๆที่ไหลผ่านตัว Browser นอกจากนี้ Browser ยังพยายามถอดรหัสที่สามารถถอดได้อีกด้วย และเก็บเข้าฐานข้อมูลของ Browser จากรูป 4.1 จะเห็นได้ว่าแม้ Browser จะมีการพยายามถอดรหัสแต่ก็ยังไม่สามารถถอดได้มากนัก โดยหาก Browser ทำตัวเป็น Intruder จะสามารถรู้ IDs หรือ ID ของ Server, Rs หรือ เลขสุ่มที่ Server สร้างขึ้นได้

Query

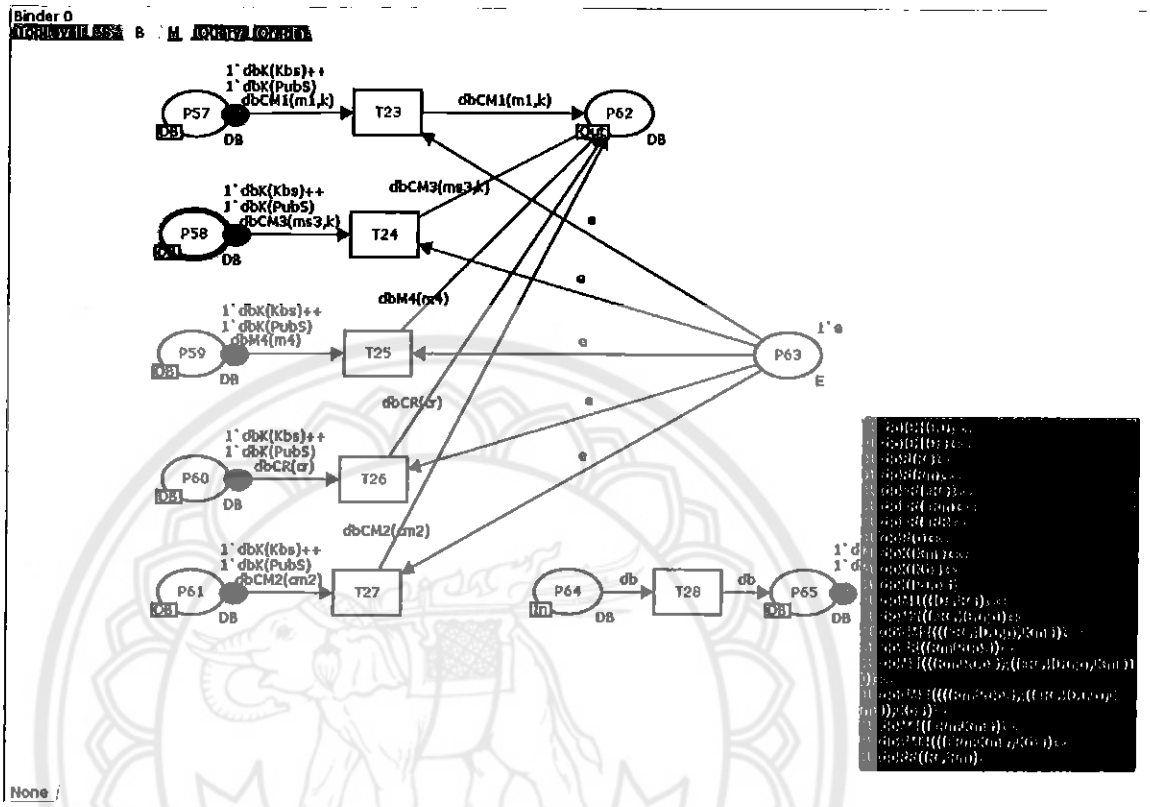
```
fun SecrecyViolation1 (pa:P) : Node list
= PredAllNodes (fn n =>
cf(dbP(pa),Mark.B'P10 1 n) > 0);

val node = SecrecyViolation1(p);
length node;
```

```
val SecrecyViolation1 = fn : P -> Node list
val node = [] : Node list
val it = 0 : int
```

รูปที่ 4.2 : Query หาโหนดที่มี Password อยู่ ของโปรโตคอล MP-Auth แบบมี Browser เป็น Intruder

จากรูปที่ 4.2 จะเห็นได้ว่าในการออกแบบให้ Browser ทำตัวเป็นเหมือน Intruder ยังไม่สามารถรู้ Password (p) ของ User ได้

4.1.3 แบบที่ Browser ทำตัวเป็น Intruder และมี Oracle เข้ามาช่วย



รูปที่ 4.3 : ข้อมูลภายในฐานข้อมูลของ Browser ที่ทำตัวเป็น Intruder และมี Oracle ช่วย

จากรูปที่ 4.3 จะเป็นรูปที่ Browser มี Oracle เป็นตัวช่วยในการถอดรหัส โดย Browser จะทำการสุ่มส่งข้อมูลที่มีอยู่ในฐานข้อมูลของ Browser ไปให้ Oracle ซึ่งเมื่อ Oracle ได้รับถ้าข้อมูลนั้นมีการเข้ารหัสอยู่ Oracle ก็จะช่วยถอดรหัสแล้วส่งข้อมูลที่ถอดได้ส่งมาให้ Browser เพื่อนำมาเก็บที่ฐานข้อมูลของ Browser ในขณะเดียวกัน Browser ก็ยังทำหน้าที่เป็น Intruder เหมือนในแบบที่ 2 เช่นเดียวกัน โดยจากในรูปค่าที่สุ่มส่งไปให้ Oracle นั้นเมื่อถูกถอดรหัสเสร็จแล้วส่งกลับมาให้ Browser ทำให้ Browser สามารถรู้ ID ของ User (IDu), ID ของ Server (IDs), รหัสสุ่มที่สร้างโดย Server (Rs) และ Mobile (Rm), ค่า Hash ของรหัสสุ่มที่สร้างโดย Server (FRs) และ Mobile (FRm), และ Password ของ User (p)

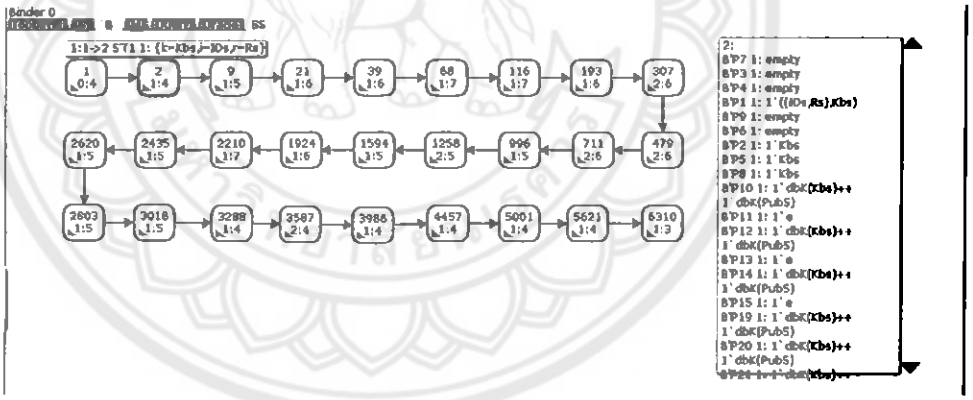
Query : Oracle

```
fun SecrecyViolation1 (pa:P) : Node list
= PredAllNodes (fn n =>
d(dbP(pa),Mark.BP10 1 n) > 0);

val node = SecrecyViolation1(p);
length node;
```

```
val SecrecyViolation1 = fn : P -> Node list
val node =
[9960,9903,9899,9844,9786,9785,9781,9778,9769,9768,9763,9759,9751,9738,9737,
 9732,9728,9720,9704,9699,9688,9684,9671,9637,9590,9533,9529,9474,9406,9253,
 9296,9292,9235,9182,9181,9176,9172,9164,9148,9143,9132,9128,9115,9093,9088,
 9077,9073,9060,9035,9024,9007,9003,8985,8951,8898,8841,8837,8780,8720,8664,
 8613,8609,8519,8508,8497,8493,8480,8455,8441,8427,8423,8405,8374,8363,8346,
 8342,8324,8292,8275,8252,8248,8186,8132,8081,8077,7977,7886,7862,7809,7792,
 7775,7771,7759,7721,7704,7681,7677,7617,7600,7577,...] : Node list
val k = 406 : int
```

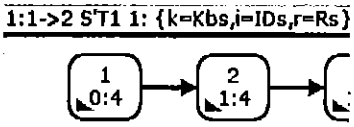
รูปที่ 4.4 : Query หาโหนดที่มี Password อยู่ ของโปรโตคอล MP-Auth แบบมี Browser เป็น Intruder และมี Oracle ช่วย

จากรูปที่ 4.4 จะพบว่ามีโหนดจำนวน 406 โหนดที่มีข้อมูลของ Password ผ่านเข้าและออก แสดงให้เห็นว่าการออกแบบโปรโตคอล MP-Auth โดยที่ Browser ทำตัวเป็น Intruder และมี Oracle เข้ามาช่วยทำให้สามารถรู้ Password ของ User ได้



รูปที่ 4.5 : ตัวอย่างเส้นทางหนึ่งของState space ที่เกิดความไม่ปลอดภัยเกิดขึ้นของโปรโตคอลMP-Auth แบบที่3

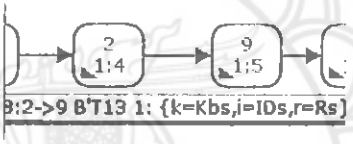
จากรูปที่ 4.5 เป็นตัวอย่างของเส้นทางจากโหนดเริ่มต้นไปหาโหนดที่มีการพบ Password ของ User หรือจะเรียกอีกอย่างได้ว่าเป็นเส้นทางที่ไม่ปลอดภัยของโปรโตคอล MP-Auth ในแบบที่ 3 โดยจะทำการเลือกโหนดจากใน List ของ Query มาแสดงให้ดูโดยไล่จากโหนดที่เลือกจาก Query ไล่ไปเรื่อยๆ จนถึงโหนดเริ่มต้น โดนภายในเส้นทางที่ผ่านนั้นก็อาจจะเจอโหนดที่มี Password อยู่เช่นกัน



รูปที่ 4.6 : เส้นทางจาก 1 ไป 2

S T1 i=IDs,r=Rs,k=Kbs

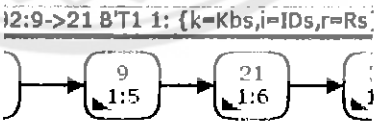
Server ที่ Transition T1 ส่ง ((IDs,Rs),Kbs) ไปให้ Browser



รูปที่ 4.7 : เส้นทางจาก 2 ไป 9

B T13 i=IDs,r=Rs,k=Kbs

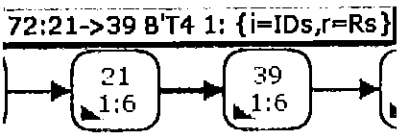
Browser ที่ Transition T13 ส่ง ((IDs,Rs),Kbs) ไปเก็บไว้ใน database



รูปที่ 4.8 : เส้นทางจาก 9 ไป 21

B T1 i=IDs,r=Rs,k=Kbs

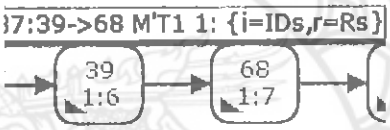
Browser ที่ Transition T1 นำ ((IDs,Rs),Kbs) มา Decrypt ด้วย Kbs แล้วส่งไปเก็บไว้ใน database



รูปที่ 4.9 : เส้นทางจาก 21 ไป 39

B T4 i=IDs,r=Rs

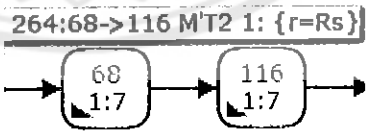
Browser ที่ Transition T4 ส่ง (IDs,Rs) ให้ Mobile



รูปที่ 4.10 : เส้นทางจาก 39 ไป 68

M T1 i=IDs,r=Rs

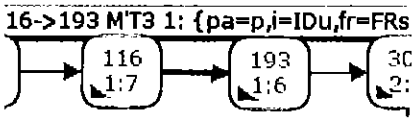
Mobile ที่ Transition T1 ส่ง (IDs,Rs) ออกไปโดยแยก IDs กับ Rs ออกจากกัน



รูปที่ 4.11 : เส้นทางจาก 68 ไป 116

M T2 r=Rs

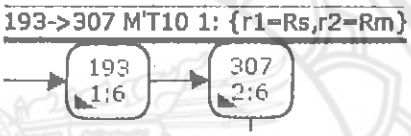
Mobile ที่ Transition T2 ส่ง Rs ออกไปเพื่อให้ Rs เข้า hash function จะได้ FRs



รูปที่ 4.12 : เส้นทางจาก 116 ไป 193

M T3 pa=p,i=IDu,fr=FRs

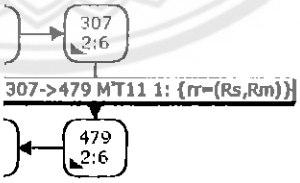
Mobile ที่ Transition T3 ส่ง (p,IDu,FRs) ออกไปเพื่อไปเตรียม Encrypt เมื่อได้ Kms มา



รูปที่ 4.13 : เส้นทางจาก 193 ไป 307

M T10 r1=Rs,r2=Rm

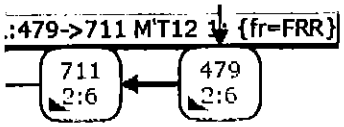
Mobile ที่ Transition T10 นำ Rs และ Rm มารวมกันจะได้ (Rs,Rm) แล้วส่งออกไป



รูปที่ 4.14 : เส้นทางจาก 307 ไป 479

M T11 rr=(Rs,Rm)

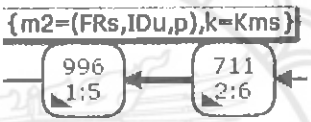
Mobile ที่ Transition T11 ส่ง (Rs,Rm) ออกไปเพื่อให้ (Rs,Rm) เข้า hash function จะได้ FRR



รูปที่ 4.15 : เส้นทางจาก 479 ไป 711

M T12 fr=FRR

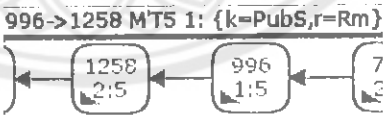
Mobile ที่ Transition T12 ส่ง FRR ออกไปเพื่อนำ FRR เข้า function GenKey จะได้ Kms



รูปที่ 4.16 : เส้นทางจาก 711 ไป 996

M T4 m2=(FRs,IDu,p),k=Kms

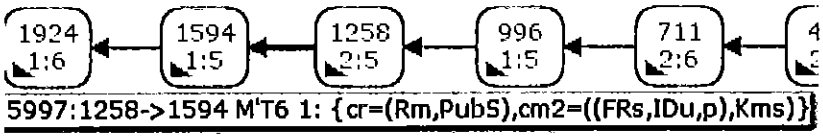
Mobile ที่ Transition T4 ส่ง m2 ที่เข้ากุญแจ Kms จะได้เป็น cm2 ออกไป



รูปที่ 4.17 : เส้นทางจาก 996 ไป 1258

M T5 r=Rm,k=PubS

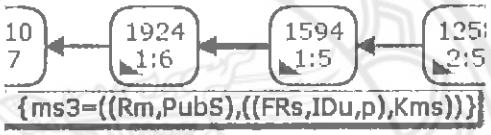
Mobile ที่ Transition T5 ส่ง Rm ที่ Encrypt ด้วย Public Key ของ Server ออกไปเพื่อเตรียมรวมกับ cm2



รูปที่ 4.18 : เส้นทางจาก 1258 ไป 1594

M T6 cr=(Rm,PubS),cm2=((FRs,IDu,p),Kms)

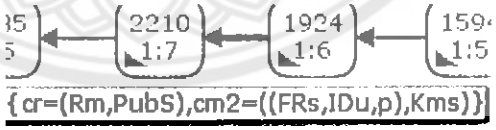
Mobile ที่ Transition T6 ส่ง cr กับ cm2 ที่รวมกันแล้วจะได้เป็น ms3 ออกไป



รูปที่ 4.19 : เส้นทางจาก 1594 ไป 1924

B T14 ms3=((Rm,PubS),((FRs,IDu,p),Kms))

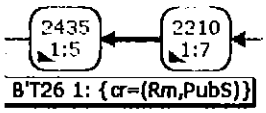
Browser ที่ Transition T14 ส่ง ms3 ไปเก็บไว้ใน database



รูปที่ 4.20 : เส้นทางจาก 1924 ไป 2210

B T8 cr=(Rm,PubS),cm2=((FRs,IDu,p),Kms)

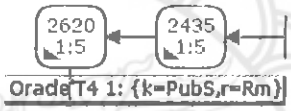
Browser ที่ Transition T8 นำ cr และ cm2 เก็บไว้ใน database เพื่อเตรียม Decrypt



รูปที่ 4.21 : เส้นทางจาก 2210 ไป 2435

B T26 cr=(Rm,PubS)

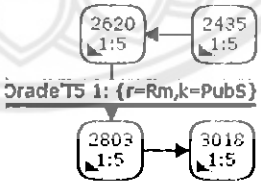
Browser ที่ Transition T26 นำ cr ที่อยู่ใน database ส่งไปให้ oracle



รูปที่ 4.22 : เส้นทางจาก 2435 ไป 2620

Oracle T4 r=Rm,k=PubS

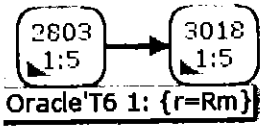
Oracle ที่ Transition T4 ส่ง cr ออกไปเพื่อเตรียม decrypt



รูปที่ 4.23 : เส้นทางจาก 2620 ไป 2803

Oracle T5 r=Rm,k=PubS

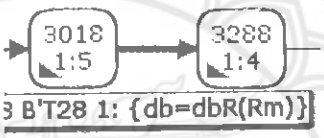
Oracle ที่ Transition T5 นำ Rm ที่ encrypt ด้วย public key ของ Server ไป decrypt ด้วย private key ที่อยู่ใน database ของ oracle



รูปที่ 4.24 : เส้นทางจาก 2803 ไป 3018

Oracle T6 r=Rm

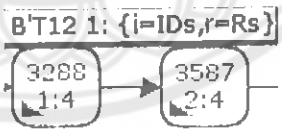
Oracle ที่ Transition T6 ส่ง Rm ออกไปให้ Browser



รูปที่ 4.25 : เส้นทางจาก 3018 ไป 3288

B T28 db=dbR(Rm)

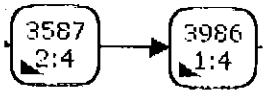
Browser ที่ Transition T28 ส่ง Rm ที่ได้จาก oracle ไปเก็บไว้ใน database



รูปที่ 4.26 : เส้นทางจาก 3288 ไป 3587

B T12 i=IDs,r=Rs

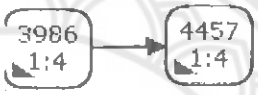
Browser ที่ Transition T12 ส่ง IDs และ Rs ไปเก็บไว้ใน database



รูปที่ 4.27 : เส้นทางจาก 3587 ไป 3986

B T20 dbR(Rs),dbR(Rm)

Browser ที่ Transition T20 นำ Rs และ Rm ที่เก็บอยู่ใน database มารวมกันแล้วส่งออกไปเก็บไว้ใน database



รูปที่ 4.28 : เส้นทางจาก 3986 ไป 4457

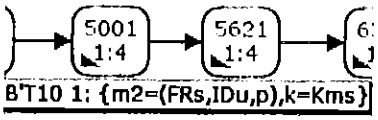
B T21 dbRR(Rs,Rm)

Browser ที่ Transition T21 ส่ง (Rs,Rm) ออกไปเพื่อให้ (Rs,Rm) เข้า hash function จะได้ FRR จากนั้นเก็บเข้า database

รูปที่ 4.29 : เส้นทางจาก 4457 ไป 5001

B T22 dbFR(FRR)

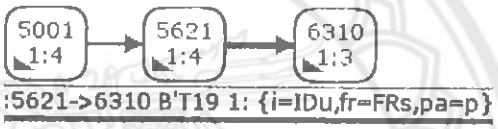
Browser ที่ Transition T22 ส่ง FRR ออกไปเพื่อนำ FRR เข้า function GenKey จะได้ Kms จากนั้นเก็บเข้า database



รูปที่ 4.30 : เส้นทางจาก 5001 ไป 5621

B T10 m2=(FRs,IDu,p),k=Kms

Browser ที่ Transition T10 นำ cm2 มา Decrypt ด้วย Kms แล้วส่ง m2 ออกไป



รูปที่ 4.31 : เส้นทางจาก 5621 ไป 6310

B T19 pa=p,i=IDu,fr=FRs

Browser ที่ Transition T19 นำ (p,IDu,FRs) มาแยกออกจากกันจะได้ ID และ Password ของ user

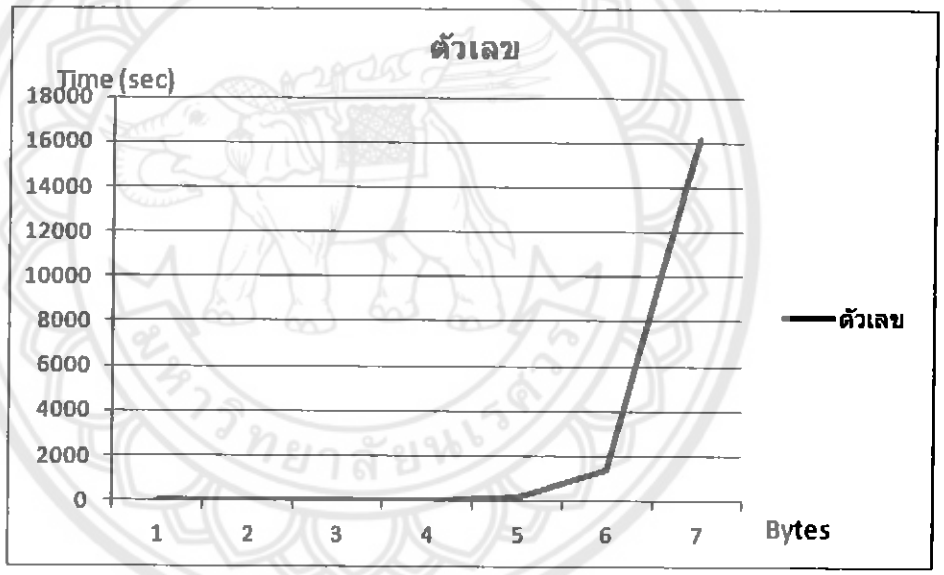
จากการทดลองสร้างระบบและทำการ Simulate ทำการ Query และทำ State space แล้วพบว่าโปรโตคอล MP-Auth จะไม่ปลอดภัยก็ต่อเมื่อ Browser ทำตัวเป็น Intruder และมี Oracle ช่วยในการถอดรหัส จาก State space ที่ได้ Browser ได้ส่ง Rm(ตัวเลขที่สุ่มมาจากโทรศัพท์มือถือ) ที่เข้ารหัสด้วย Public Key ของ Server ให้ Oracle ช่วยในการถอดรหัส เมื่อถอดรหัสเสร็จส่งกลับมาให้ Browser เพื่อนำ Rm มาสร้าง Kms หลังจากได้ Kms มาแล้วนำมาถอดรหัส ID และ Password ของผู้ใช้ที่เข้ารหัสด้วย Kms ถ้าถอดรหัสสำเร็จจะได้ ID และ Password ของผู้ใช้

4.2 การทดลองด้วยโปรแกรมประยุกต์

การทดลองด้วยโปรแกรมประยุกต์ที่เขียนจากภาษา Java สำหรับการ Brute force รหัสที่ถูกเข้ารหัสด้วย Public Key ขนาด 1024 bit เพื่อให้ได้มาซึ่งรหัสที่เกิดจากการสุ่มขึ้นมาของโทรศัพท์เคลื่อนที่ โดยใช้คอมพิวเตอร์ 1 เครื่อง และแยกชนิดของรหัสที่คาดว่าจะถูกสร้างขึ้นแบบสุ่มจากโทรศัพท์ ได้ผลดังนี้

- ตัวเลข

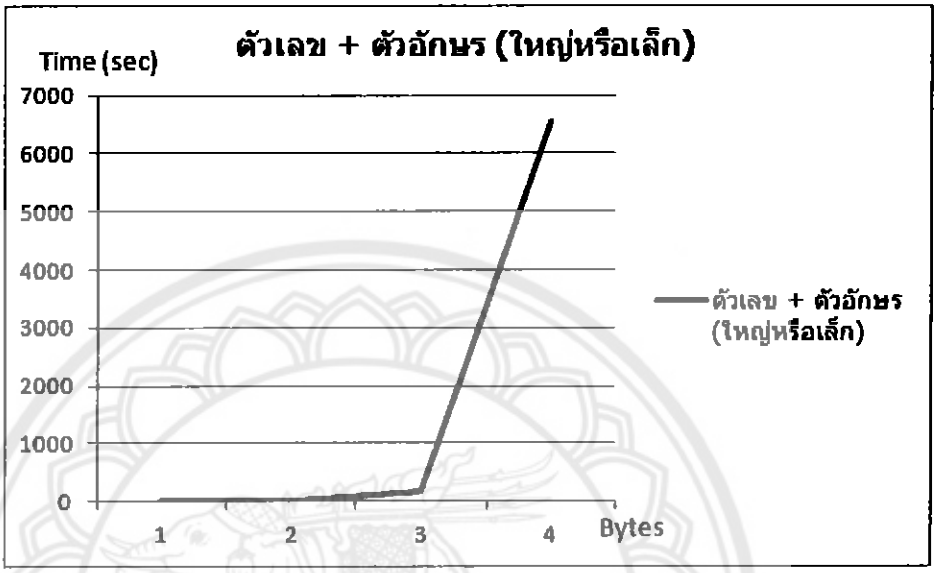
การ Brute force รหัสตั้งแต่ 2 Bytes – 8 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.32 : กราฟ Brute force ตัวเลข

- ตัวเลข + ตัวอักษร (แค่ใหญ่ หรือ เล็ก)

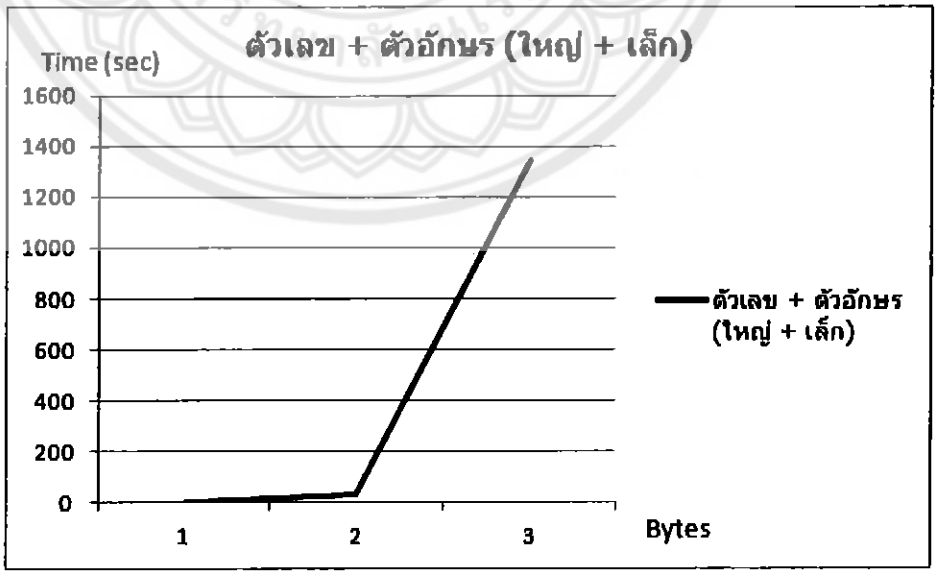
การ Brute force รหัสตั้งแต่ 2 Bytes – 5 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.33 : กราฟ Brute force ตัวเลข + ตัวอักษร (แค่ใหญ่ หรือ เล็ก)

- ตัวเลข + ตัวอักษร (ใหญ่ + เล็ก)

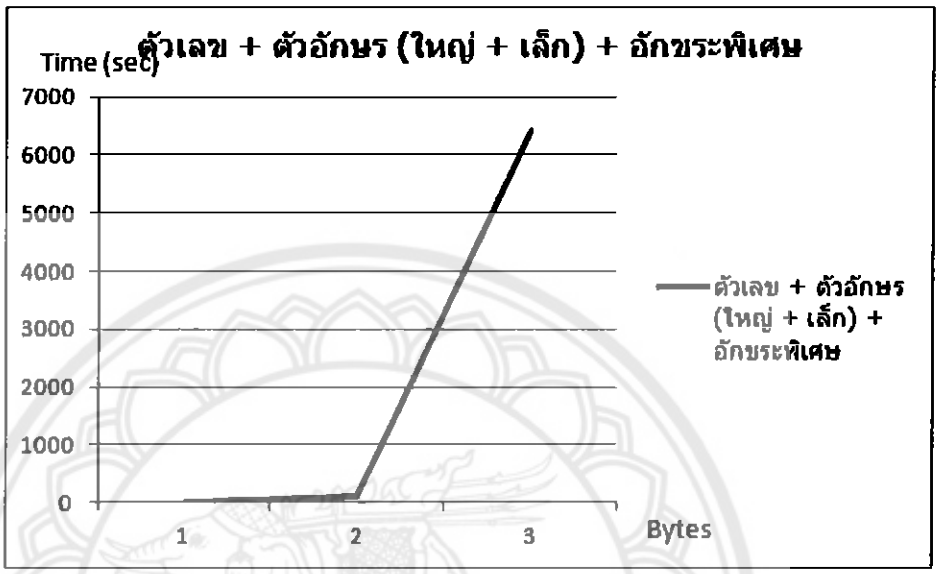
การ Brute force รหัสตั้งแต่ 2 Bytes – 4 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.34 : กราฟ Brute force ตัวเลข + ตัวอักษร (ใหญ่ + เล็ก)

- **ตัวเลข + ตัวอักษร (ใหญ่ + เล็ก) + อักขระพิเศษ**

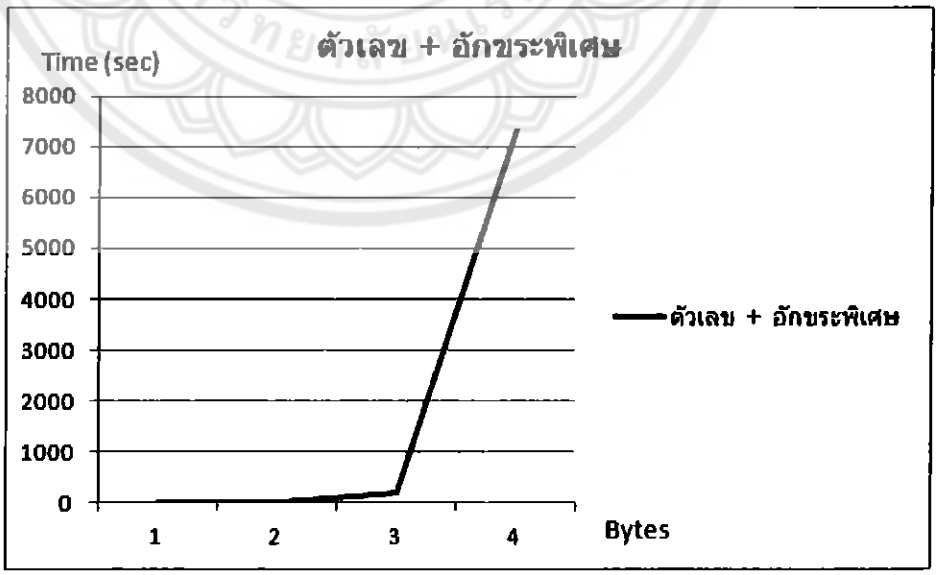
การ Brute force รหัสตั้งแต่ 2 Bytes – 4 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.35 : กราฟ Brute force ตัวเลข + ตัวอักษร (ใหญ่ + เล็ก) + อักขระพิเศษ

- **ตัวเลข + อักขระพิเศษ**

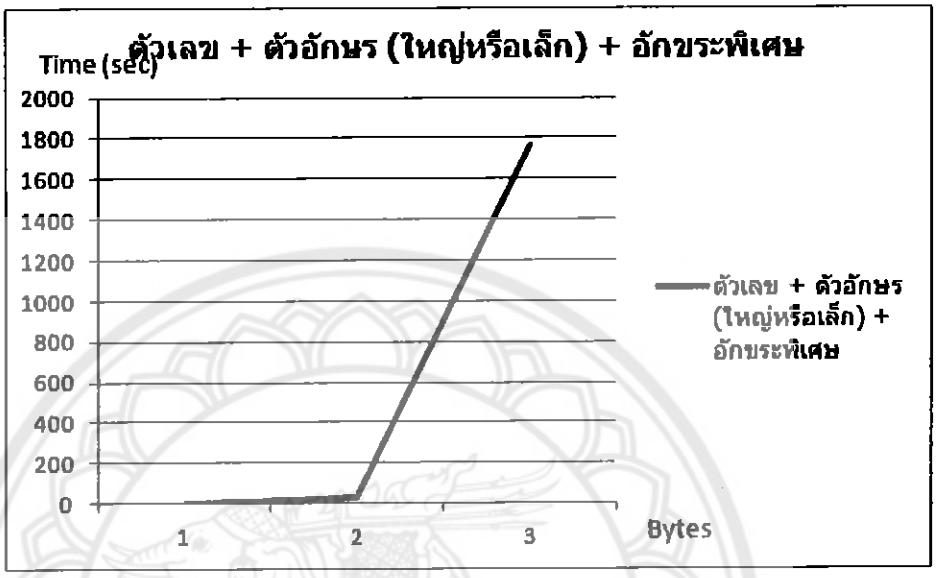
การ Brute force รหัสตั้งแต่ 2 Bytes – 5 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.36 : กราฟ Brute force ตัวเลข + อักขระพิเศษ

- ตัวเลข + ตัวอักษร (ใหญ่หรือเล็ก) + อักขระพิเศษ

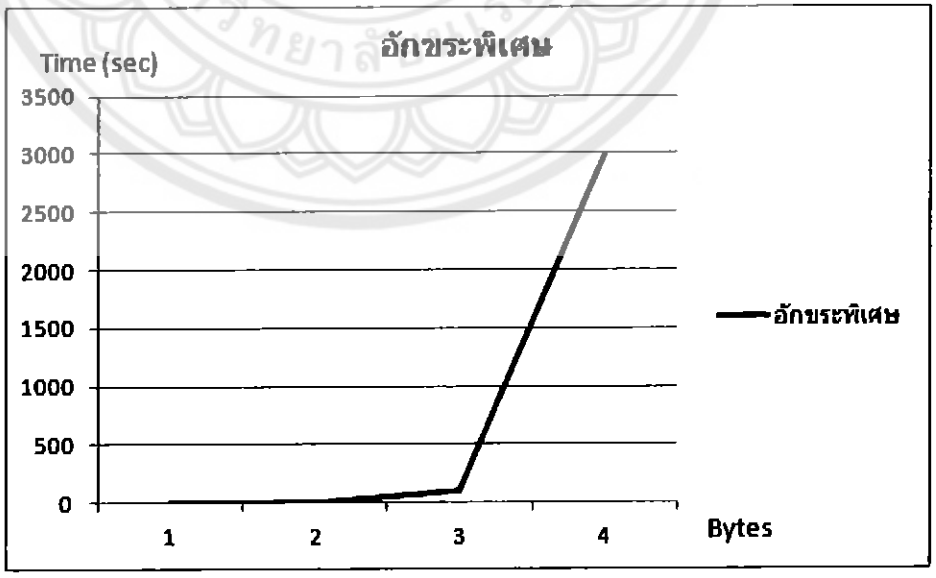
การ Brute force รหัสตั้งแต่ 2 Bytes – 4 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.37 : กราฟ Brute force ตัวเลข + ตัวอักษร (ใหญ่หรือเล็ก) + อักขระพิเศษ

- อักขระพิเศษ

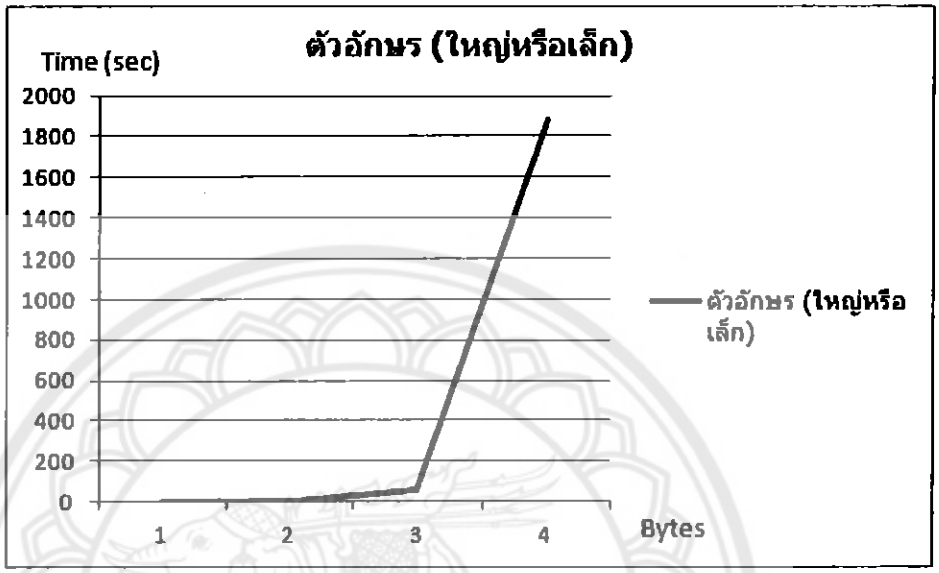
การ Brute force รหัสตั้งแต่ 2 Bytes – 5 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.38 : กราฟ Brute force อักขระพิเศษ

- ตัวอักษร (ใหญ่หรือเล็ก)

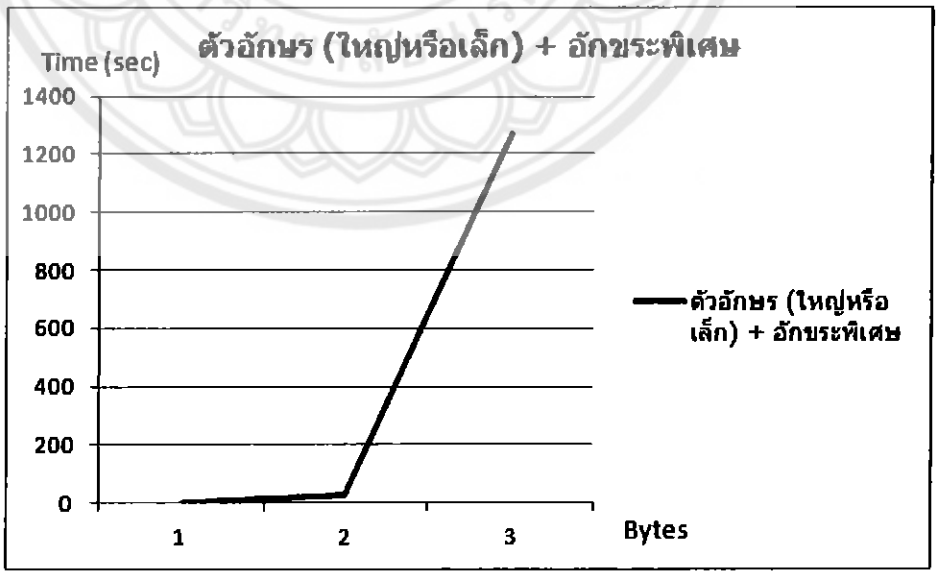
การ Brute force รหัสตั้งแต่ 2 Bytes – 5 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.39 : กราฟ Brute force ตัวอักษร (ใหญ่หรือเล็ก)

- ตัวอักษร (ใหญ่หรือเล็ก) + อักขระพิเศษ

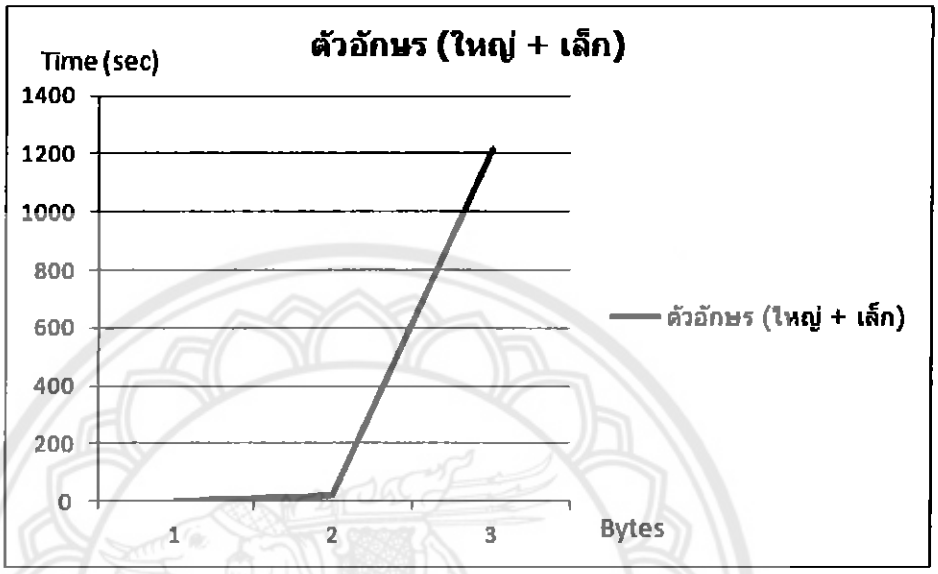
การ Brute force รหัสตั้งแต่ 2 Bytes – 4 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.40 : กราฟ Brute force ตัวอักษร (ใหญ่หรือเล็ก) + อักขระพิเศษ

- ตัวอักษร (ใหญ่ + เล็ก)

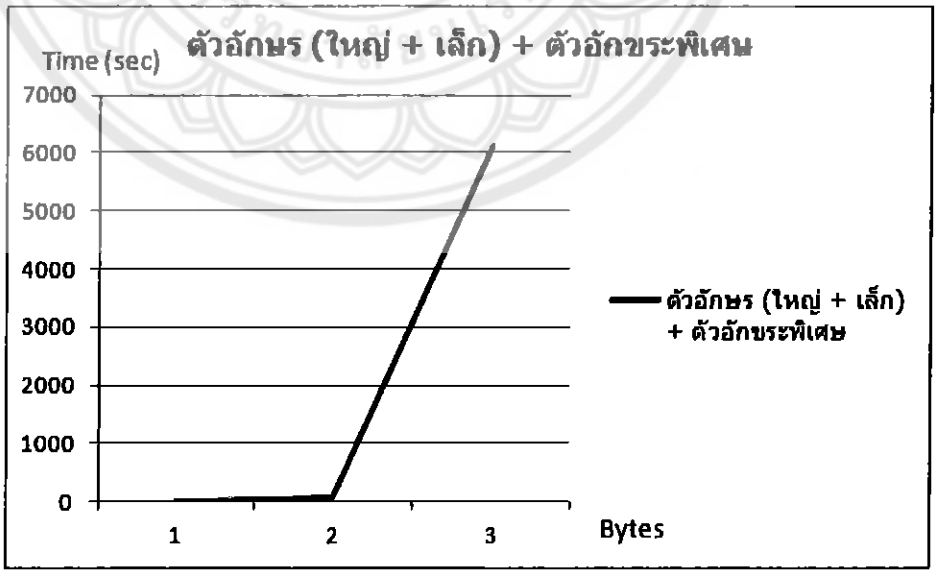
การ Brute force รหัสตั้งแต่ 2 Bytes – 4 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.41 : กราฟ Brute force ตัวอักษร (ใหญ่ + เล็ก)

- ตัวอักษร (ใหญ่ + เล็ก) + อักขระพิเศษ

การ Brute force รหัสตั้งแต่ 2 Bytes – 4 Bytes ที่เข้ารหัสด้วย Public Key



รูปที่ 4.42 : กราฟ Brute force ตัวอักษร (ใหญ่ + เล็ก) + อักขระพิเศษ

จากการทดลองการ Brute force ด้วยโปรแกรมประยุกต์ที่เขียนโดยภาษา Java และนำผลที่ได้จากการทดลองมาสร้างเป็นกราฟ โดยให้แกน Y เป็น เวลา (วินาที) และแกน X เป็น จำนวน Byte จะเห็นว่ากราฟที่ได้จะมีค่าความชันเพิ่มขึ้นเรื่อยๆ เป็นแบบกราฟ Exponential โดยเมื่อจำนวน Byte มีมากขึ้น ก็จะทำให้ใช้เวลาในการ Brute force มากขึ้นตามไปด้วย และไม่ว่ารหัสที่ต้องการเดาจะประกอบไปด้วยอะไรก็ตามกราฟที่ได้จากการทดลองก็จะมีลักษณะใกล้เคียงกัน

บทที่ 5

สรุปผลการทดลอง

5.1 การทดลองด้วยโปรแกรม CPN Tools

จากทดลองด้วย CPN-Tools ถ้าหาก Browser รู้ Rm (รหัสสุ่มของโทรศัพท์มือถือ) จะนำ Rm ไปสร้าง Kms (กุญแจเดี่ยวระหว่างโทรศัพท์มือถือกับ Server) ซึ่งจะนำไปสู่การถอดรหัสจะทำให้ MP-Auth นั่นไม่ปลอดภัย เพราะว่า Browser ที่ทำตัวเป็น Intruder (ผู้บุกรุก) จะได้ ID และ Password ของ User

5.2 การทดลองด้วยโปรแกรมประยุกต์

ตารางที่ 5.1 : สมการใช้คำนวณเวลาโดยเฉลี่ยของกราฟแต่ละกราฟ

y = เวลา (วินาที) และ x = จำนวน Byte

รหัส	สมการจากกราฟ
ตัวเลข	$y=0.0137e^{1.9076x}$
ตัวอักษร (ใหญ่ หรือ เล็ก)	$y=0.0295e^{2.6723x}$
ตัวอักษร (ใหญ่ + เล็ก)	$y=0.0297e^{3.4836x}$
อักขระพิเศษ	$y=0.0264e^{2.8475x}$
ตัวเลข + ตัวอักษร (ใหญ่ หรือ เล็ก)	$y=0.0201e^{3.1094x}$
ตัวเลข + ตัวอักษร (ใหญ่ + เล็ก)	$y=0.0282e^{3.5597x}$
ตัวเลข + อักขระพิเศษ	$y=0.0139e^{3.2336x}$
ตัวอักษร (ใหญ่ หรือ เล็ก) + อักขระพิเศษ	$y=0.0349e^{3.4561x}$
ตัวอักษร (ใหญ่ + เล็ก) + อักขระพิเศษ	$y=0.0381e^{3.9341x}$
ตัวเลข + ตัวอักษร (ใหญ่ หรือ เล็ก) + อักขระพิเศษ	$y=0.0198e^{3.7805x}$
ตัวเลข + ตัวอักษร (ใหญ่ + เล็ก) + อักขระพิเศษ	$y=0.0212e^{4.2114x}$

จากการทดลอง Brute force เดารหัสด้วยโปรแกรมประยุกต์ที่เขียนโดยภาษา Java เพื่อให้ได้มาซึ่งรหัส R_M หรือรหัสที่โทรศัพท์สุ่มสร้างขึ้นมาเพื่อที่จะใช้ติดต่อกับ Server หากรหัสมีลักษณะเป็นแค่ตัวเลข ตัวอักษร หรือแค่ชนิดเดียว หากมีความยาวของรหัสต่ำกว่า 8 Bytes หรือ 8 ตัวอักษรและไม่มีการเปลี่ยนรหัสทุก 3 เดือน จะทำให้โปรโตคอล MP-Auth ไม่ปลอดภัย เพราะถ้ามีการ Brute force โดยใช้คอมพิวเตอร์ 1 เครื่องจะสามารถถอดรหัสและได้ R_M ที่มีชนิดของรหัสชนิดเดียวและมีความยาวของรหัสต่ำกว่า 8 ตัวอักษรที่เข้ารหัสด้วย Public Key ภายในระยะเวลาไม่เกิน 3 เดือน และถ้ามีการใช้คอมพิวเตอร์ n เครื่องมาช่วยในการ Brute force ก็จะทำให้เร็วขึ้นกว่าเดิม n เท่าตามทฤษฎี ทั้งนี้ขึ้นอยู่กับประสิทธิภาพและความเร็วในการประมวลของเครื่องที่ใช้

โดยสรุปแล้วโปรโตคอล MP-Auth จะไม่ปลอดภัยก็ต่อเมื่อ Rm(การสุ่มรหัสของโทรศัพท์เคลื่อนที่) ถูกล่วงรู้ เพราะฉะนั้น Rm ควรมีความยาวของรหัส 8 Bytes หรือมากกว่า ถ้าหากจำเป็นต้องเป็นตัวอักษรชนิดเดียว ให้เพิ่มความหลายหลายของชนิดตัวอักษรให้กับ Rm และ เปลี่ยนรหัสผ่านใหม่ทุกๆ 3 เดือน โปโตคอล MP-Auth จะมีความปลอดภัยมากขึ้น

5.3 แนวทางการพัฒนาเพิ่มเติม

สามารถนำวิธีในการวิเคราะห์โปรโตคอลยืนยันตนนี้ไปปรับใช้และพัฒนาโปรโตคอลยืนยันตนตัวอื่นๆ ได้

ในส่วนของ CPN-Tools อาจจะออกแบบให้ database ใน browser ที่ทำตัวเป็น intruder ทำการแก้ไขข้อความที่ผู้ส่งส่งมา แล้วนำข้อความที่ถูกแก้ไขแล้วส่งไปให้ผู้รับ

ในส่วนของโปรแกรมประยุกต์ สามารถพัฒนาให้คอมพิวเตอร์หลายๆเครื่องช่วยกัน Brute Force ได้ อาจจะทำให้คอมพิวเตอร์หลายเครื่องมีการติดต่อกันได้ ทำให้รู้ว่า Brute force ช่วงไหนเสร็จไปแล้วบ้าง จะได้ไม่ได้ทำ Brute force ตัวเดิมซ้ำ และถ้ามีเครื่องใดเครื่องหนึ่งหาเจอให้แจ้งเครื่องอื่นทราบด้วย

5.4 สรุปวิธีที่ใช้ในการวิเคราะห์

การตรวจสอบความปลอดภัยของโปรโตคอลยืนยันตน มีด้วยกันหลายวิธี เช่น การใช้สมการทางคณิตศาสตร์ เป็นต้น แต่วิธีทางคณิตศาสตร์ต้องมีความชำนาญทางคณิตศาสตร์ด้วยจึงจะสามารถใช้ได้ ดังนั้น การใช้โปรแกรม CPN Tools เพื่อใช้วิเคราะห์ความปลอดภัยของโปรโตคอลยืนยันตนก็เป็นวิธีหนึ่งที่ผู้ที่ต้องการตรวจสอบสามารถทำได้โดยที่ไม่ต้องมีความชำนาญทางคณิตศาสตร์ หรือความชำนาญเป็นพิเศษด้านอื่นๆ แค่มีความเข้าใจเกี่ยวกับระบบก็สามารถใช้โปรแกรม CPN Tools ในการวิเคราะห์ความปลอดภัยของโปรโตคอลยืนยันตนได้

5.5 ปัญหาที่พบในการทำโครงงาน

การใช้งานโปรแกรม CPN Tools ช่วงแรกใช้เครื่องมือไม่เป็น และมีการแก้ไขตัว model หลายครั้ง ส่วนโปรแกรมประยุกต์ การเก็บข้อมูลถ้ามีตัวอักษรเยอะๆ จะใช้เวลานาน

เอกสารอ้างอิง

[1] Wikipedians. **วิทยาการเข้ารหัสลับ.** [ออนไลน์]. เข้าถึงได้จาก : http://th.wikipedia.org/wiki/%E0%B8%A7%E0%B8%B4%E0%B8%97%E0%B8%A2%E0%B8%B2%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B9%80%E0%B8%82%E0%B9%89%E0%B8%B2%E0%B8%A3%E0%B8%AB%E0%B8%B1%E0%B8%AA%E0%B8%A5%E0%B8%B1%E0%B8%9A. (วันที่ค้นข้อมูล : 9 กรกฎาคม 2555).

[2] Networker09. **AAA Protocol.** [ออนไลน์]. เข้าถึงได้จาก : http://networker09.wordpress.com/category/authentication/. (วันที่ค้นข้อมูล : 9 กรกฎาคม 2555).

[3] Wikipedians. **โพรโทคอล.** [ออนไลน์]. เข้าถึงได้จาก : http://th.wikipedia.org/wiki/%E0%B9%82%E0%B8%9E%E0%B8%A3%E0%B9%82%E0%B8%97 %E0%B8%84%E0%B8%AD%E0%B8%A5. (วันที่ค้นข้อมูล : 11 กรกฎาคม 2555).

[4] Protocol. **Protocol คืออะไร.** [ออนไลน์]. เข้าถึงได้จาก : http://www.mindphp.com/%E0%B8%84%E0%B8%B9%E0%B9%88%E0%B8%A1%E0%B8%B7%E0%B8%AD/73-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/2044-protocol-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3.html. (วันที่สืบค้นข้อมูล : 11 กรกฎาคม 2555).

[5] Mohammad Mannan and P.C. van Oorschot. **Leveraging Personal Devices for Stronger Password Authentication from Untrusted Computers.** Canada: Carleton University.

[6] Wikipedia. **Petri net.** [ออนไลน์]. เข้าถึงได้จาก : http://en.wikipedia.org/wiki/Petri_net. (วันที่สืบค้นข้อมูล : 27 พฤษภาคม 2556).

[7] Wikipedia. **Coloured petri net.** [ออนไลน์]. เข้าถึงได้จาก : http://en.wikipedia.org/wiki/Coloured_Petri_net.(วันที่สืบค้นข้อมูล : 27 พฤษภาคม 2556).

[8] Michael Westergaard and H.M.W. (Eric) Verbeek. **CPN Tools.** [ออนไลน์]. เข้าถึงได้จาก : http://cpntools.org/. (วันที่สืบค้นข้อมูล : 24 กรกฎาคม 2555).

[9] Issam Al-Azzoni, Douglas G. Down and Ridha Khedri. **Modeling and Verification of Cryptographic Protocols Using Coloured Petri Nets and *Design/CPN.*** Canada: McMaster University.

[10] สิริพร จิตต์เจริญธรรม, เสาวภา ปานจันทร์ และ เลอศักดิ์ ลิ้มวิวัฒน์กุล. **ความรู้เบื้องต้นเกี่ยวกับการพิสูจน์ตัวตน.** [ออนไลน์]. เข้าถึงได้จาก : http://www.thaipki.com/knowledge_authen.html#authen_protocol. (วันที่สืบค้นข้อมูล : 18 กรกฎาคม 2555).

[11] Issam Al-Azzoni, B.Eng(December, 2004). **THE VERIFICATION OF CRYPTOGRAPHIC PROTOCOLS USING COLOURED PETRI NETS.** Canada: McMaster University.

ภาคผนวก

Code โปรแกรมประยุกต์

genpublic.java

เป็นโปรแกรมที่ใช้สร้าง Public Key และ Private Key แล้วบันทึก Public Key ลง Text ไฟล์

```
import java.security.KeyPairGenerator;
import java.security.KeyPair;
import java.security.Key;
import java.security.PublicKey;
import java.security.PrivateKey;
import javax.crypto.Cipher;
import sun.misc.BASE64Encoder;
import java.io.*;
import java.util.*;

public class genpublic{
        public static void main (String[] args) throws Exception {
                KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");
                kpg.initialize(1024);
                KeyPair kp = kpg.genKeyPair();
                Key publicKey = kp.getPublic();
                byte[] publicKeyTest = publicKey.getEncoded();
                Key privateKey = kp.getPrivate();
                String publicString = new BASE64Encoder().encode(publicKey.getEncoded());
                System.out.println(publicString);
                FileOutputStream out = new FileOutputStream("publicKey.txt");
                byte[] pubkB = publicKey.getEncoded();
                System.out.println(pubkB);
    out.write(pubkB);
```

```
            out.close();
            System.out.println("");
        }
}
```

Key.java

เป็นโปรแกรมที่อ่านค่า Public Key จากไฟล์ Text แล้วสุ่มตัวอักษรแล้วเข้ารหัสด้วย Public Key จากนั้นบันทึกค่าตัวอักษรที่เข้ารหัสด้วย Public Key ลง Text ไฟล์อีกไฟล์หนึ่ง

```
import java.security.KeyPairGenerator;
import java.security.KeyPair;
import java.security.Key;
import java.security.PublicKey;
import java.security.PrivateKey;
import javax.crypto.Cipher;
import sun.misc.BASE64Encoder;
import java.security.KeyFactory;
import java.security.spec.X509EncodedKeySpec;
import java.io.*;
import java.util.*;
import java.util.Random;

public class key{
        public static void main (String[] args) throws Exception {
            File file = new File("publicKey.txt");
            FileInputStream fin = null;
            // create FileInputStream object
            fin = new FileInputStream(file);
            byte fileContent[] = new byte[(int)file.length()];
```

```
                // Reads up to certain bytes of data from this input stream into an array of
bytes.
                fin.read(fileContent);

                //create string from byte array
                //String s = new BASE64Encoder().encode(fileContent);
                fin.close();

                //char[] canUse =
{'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','a','b','c','
d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z','0','1','2','3','4','5','6','7','8','9','/*'
',*/',','(',')','~','!','@','#','$','%','^','&','*','_','-','+','=','{','}','[',']','|',':',';','<','>',',','/','?'};
                char[] canUse =
{'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','0','1','2','
3','4','5','6','7','8','9',',',',','(',')','~','!','@','#','$','%','^','&','*','_','-
','+','=','{','}','[',']','|',':',';','<','>',',','/','?'};
                String xform = "RSA/ECB/NoPadding";
                int value = 2;
                StringBuffer rd = new StringBuffer();

                Random r = new Random();
                for(int j = 0; j < value; j++){ // character
                        int n = r.nextInt(64); // random 0 - 90
                        //System.out.print(canUse[n]);
                        rd.append(canUse[n]);
                }
                String a = rd.toString();
                System.out.println(a);
                byte[] dataBytes = a.getBytes();
```

```
                PublicKey pubK = KeyFactory.getInstance("RSA").generatePublic(new
X509EncodedKeySpec(fileContent));
                //String publicString = new BASE64Encoder().encode(pubK.getEncoded());

                Cipher cipher = Cipher.getInstance(xform);
                cipher.init(Cipher.ENCRYPT_MODE, pubK);
                byte[] ecrypted = cipher.doFinal(dataBytes);
                String output =  new BASE64Encoder().encode(ecrypted);

                BufferedWriter out = new BufferedWriter(new FileWriter("Encode.txt"));
                BufferedWriter out1 = new BufferedWriter(new FileWriter("a.txt"));

                out.write(output);
    out.close();
                out1.write(a);
    out1.close();
        }
}
```

Key4.java

โปรแกรมนี้จะอ่าน Public Key และ Cipher text (ค่าอักษรที่เข้ารหัสด้วย Public Key) จาก Text ไฟล์ทั้ง 2 ไฟล์ แล้วทำการ Brute Force เข้ารหัสตัวอักษรที่ Brute Force ด้วย Public Key และเปรียบเทียบกับ Cipher text ที่อ่านมาจากไฟล์ Text หากตรงกันก็จะแสดงเวลาที่ Brute Force สำเร็จ

```
import java.security.KeyPairGenerator;
import java.security.KeyPair;
import java.security.Key;
import java.security.PublicKey;
import java.security.PrivateKey;
import javax.crypto.Cipher;
import sun.misc.BASE64Encoder;
import sun.misc.BASE64Decoder;
import java.security.KeyFactory;
import java.security.spec.X509EncodedKeySpec;
import java.io.*;
import java.util.*;

public class key4{
        String a;
        char[] canUse =
{'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','a','b','c','
d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z','0','1','2','3','4','5','6','7','8','9','/*'
',*/',',','(',')','~','!','@','#','$','%','^','&','*','_','-','+','=','{','}','[',']','|',':',';','<','>','.','/','?'};
        static int minlen = 1; // 2
        static int maxlen = 2;
        final double startTime = System.currentTimeMillis();

        public static void main (String[] args) throws Exception {
```

```
        key4 b = new key4();
    }

    public key4() throws Exception{
        int k = 0;
        while (k < canUse.length) {
            nextString(new Character(canUse[k]).toString());
            k++;
        }
    }

    public void nextString(String s) throws Exception{
        int i = 0;
        File file = new File("publicKey.txt");
        FileInputStream fin = null;
        String xform = "RSA/ECB/NoPadding";
        // create FileInputStream object
        fin = new FileInputStream(file);
        byte fileContent[] = new byte[(int)file.length()];
        // Reads up to certain bytes of data from this input stream into an array of
bytes.
        fin.read(fileContent);

        //create string from byte array
        fin.close();
        PublicKey pubK = KeyFactory.getInstance("RSA").generatePublic(new
X509EncodedKeySpec(fileContent));
        String publicString = new BASE64Encoder().encode(pubK.getEncoded());
        Cipher cipher = Cipher.getInstance(xform);
```

```
while (i< canUse.length) {
        a = s + new Character(canUse[i]).toString();
        byte[] dataBytes1 = a.getBytes();
        if (new String(s + new Character(canUse[i]).toString()).length() <=
maxlen-1) {
        nextString(s + new Character(canUse[i]).toString());
        }
        cipher.init(Cipher.ENCRYPT_MODE, pubK);
        byte[] ecrypted1 = cipher.doFinal(dataBytes1);
        String output1 =  new BASE64Encoder().encode(ecrypted1);

        File file1 = new File("Encode.txt");
        String output2 = new Scanner(file1).useDelimiter("\\Z").next();

        if (output2.compareTo(output1) == 0) {
                System.out.println("Success : " + a);
                final double endTime = System.currentTimeMillis();
                System.out.println("Total execution time: " + (endTime -
startTime) + " MilliSecond");
                System.out.println("Total execution time: " + ((endTime -
startTime)/1000) + " Second");
                System.out.println("Total execution time: " + (((endTime -
startTime)/1000)/60) + " Minute");
                System.out.println("Total execution time: " + ((((endTime
- startTime)/1000)/60)/60) + " Hour");
                System.out.println("Total execution time: " + (((((endTime
- startTime)/1000)/60)/60)/24) + " Day");
```

```
                    BufferedWriter out = new BufferedWriter(new
FileWriter("String.txt"));
                    out.write(a);
                    out.close();
                    System.exit(0);
                } else {
                    System.out.println("Error : " + a);
                }
                i++;
            }
        }
}
```