ปีการศึกษา 2539

การพัฒนาระบบความปลอดภัยของข้อมูลบนเครือข่ายคอมพิวเตอร์ในสภาพแวดล้อมแบบไคล์เอนท์/เซิร์ฟเวอร์

(CLIENT/SERVER NETWORK WITH SECURITY SYSTEM DEVELOPMENT)

โดย

นายชัยรัตน์ บุหงาวงษ์

นายดุลยพินิจ พอสอน

อาจารย์ที่ปรึกษา

ดร. วรวัฒน์ ลิ้มโภคา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามปริญญา

วิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง

ปริญญานิพนธ์ปีการศึกษา 2539

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะ วิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การพัฒนาระบบต้นแบบของแอพพลิเคชั่นความปลอดภัยของข้อมูลบนเครือข่ายคอมพิวเตอร์แบบไคล์เอนท์/เซิร์ฟเวอร์

ผู้จัดทำ

1. นายชัยรัตน์ บุหงาวงษ์ รหัส 37013289
2. นายคุลยพินิจ พอสอน รหัส 37013292

อาจารย์ที่ปรึกษา

(ดร.วรวัฒน์ ลิ้มโภคา)

การพัฒนาระบบต้นแบบของแอพพลิเคชั่นความปลอดภัยของข้อมูลบนเครือข่ายคอมพิวเตอร์แบบไคล์เอนท์/เซิร์ฟเวอร์

นายชัยรัตน์ บุหงาวงษ์
นายดุลยพินิจ พอสอน
ดร. วรวัฒน์ ลิ้มโภคา

ปีการศึกษา 2539

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้เรียบเรียงขึ้นจากขั้นตอนการทำงานทั้งหมดในการสร้างระบบงานประยุกต์เพื่อใช้ในการจัดการเข้ารหัสแบบ RSA บนเครือข่ายคอมพิวเตอร์แบบ ไคลเอนท์/เซิร์ฟเวอร์ (Client/Server Network with RSA Security) โดยใช้วิธีการของ อัลกอริทึม RSA เพื่อพัฒนาระบบดังกล่าวขึ้นมาเป็นระบบต้นแบบ (Prototype) เริ่มจากการศึกษาพื้นฐานในการเข้ารหัสถอดรหัส,ทฤษฎีของการเข้ารหัสถอดรหัสแบบอาร์เอสเอ อัลกอริทึมและทฤษฎีของระบบไคล์เอ็นท์เซิร์ฟเวอร์ขั้นต้นจากนั้นทำการวิเคราะห์และออกแบบระบบโดยอ้างอิงกับทฤษฎีข้างต้นที่ศึกษา และเขียนโปรแกรมโดยวิธีของออบเจค-ออเรนท์เตท โดยให้สัมพันธ์กับระบบไคล์เอนท์เซิร์ฟเวอร์เพื่อสร้างโปรแกรมประยุกต์ขึ้นมาใช้งาน ในการเขียนโปรแกรมนั้นเราจะใช้ Delphi ของบริษัท Borland เป็นตัวพัฒนาโปรแกรม

CLIENT/SERVER NETWORK WITH SECURITY SYSTEM DEVELOMENT

Chairat Bunghaworng

Doonyapinit Posorn

Dr. Vorawat Limpoka

1996

Abstrack

This Thesis is based on the development of Client/Server Network with RSA Security System to be a prototype. First is study Basic of Encryption/Decryption Data Security,RSA Algorithm and Data Information of Client/Server,After that analysis and assign design by reference theory.By means of Object-Oriented Methodlogy to design and implement a application on a Client/Server database.The product that use to design and deverlopment by Delphi from Borland Internation Inc..

สารบัญ

สารบัญรูปภาพ

สารบัญตาราง

บทที่ 1

บทนำ

ปริญญานิพนธ์ฉบับนี้เขียนขึ้นเพื่อประกอบโครงงานของนักศึกษาชั้นปีที่ 3 ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง ซึ่งในโครงงานนี้เป็นการพัฒนาระบบต้นแบบของแอพพลิเคชั่นการเข้ารหัสข้อมูลบนฐานข้อมูลแบบไคล์เอนท์/เซิร์ฟเวอร์

ซึ่งในการพัฒนาระบบการรักษาความปลอดภัยข้อมูลมีความจำเป็นอย่างมากในปัจจุบัน เนื่องจากข้อมูลแต่ละองค์กรมีความสำคัญมาก โอกาสที่ข้อมูลจะถูกโจรกรรมมีความเสี่ยงสูง ถ้าของมูลรั่วไหลจาก องค์กร ทำให้เกิดผลเสียหายต่อองค์กร ดังนั้นโครงงานนี้จึงได้เลือก การเข้ารหัสจากข้อมูลแบบอาร์เอสเออัลกอริทึม(RSA Algorithms) ที่มีการทำงานภายใต้ ฐานข้อมูลแบบไคล์เอน/เซิร์ฟเวอร์(Client/Server Environment)

ซึ่งรายละเอียดของปริญญานิพนธ์ฉบับนี้ จะประกอบไปด้วย

บทที่ 2 เป็นรายละเอียดเกี่ยวกับทฤษฎีที่ใช้ในการพัฒนาระบบ ได้แก่ ความรู้พื้นฐานของการเข้ารหัสและถอดรหัส,หลักการของไคล์เอนท์/เซิร์ฟเวอร์ และการสนับสนุนของเดลไฟล์ Delphi ในการพัฒนาแอพพลิเคชั่นฐานข้อมูล

บทที่ 3 เป็นรายละเอียดของผลที่ได้จากการใช้ความรู้จากทฤษฎีในการพัฒนาระบบมาใช้ในการวิเคราะห์และออกแบบระบบซึ่งได้แก่ อัลกอริทึมต่าง ๆ เช่น อัลกอริทึมการหาค่าจำนวนเฉพาะ,อัลกอริทึมการหาค่ายกกำลังแล้วมอด เป็นต้น

บทที่ 4 เป็นผลการทดลองใช้งานและวิธีการใช้งานที่ได้จากการพัฒนาระบบด้วยเดลไฟล์ (Delphi)จะแสดงระบบการทำงานในส่วนหลัก ๆ และหน้าจอประกอบ

บทที่ 5 เป็นข้อจำกัด,ขอบเขต,ปัญหาและแนวทางในการแก้ไข,พัฒนาระบบนี้ต่อไป

บทที่ 2

ทฤษฎีที่ใช้ในการพัฒนาระบบ

2.1 พื้นฐานการเข้ารหัส,ถอดรหัส(Basic Encryption and Decryption)

ในบางครั้งในการติดต่อส่งข่าวสาร ข่าวสารของเราอาจจะไปไม่ถึงมือผู้รับ โดยอาจมีผู้อื่นมาเอาข่าวสารของเราไป ดังนั้นเราอาจจะใช้การเข้ารหัส(Encryption) เพื่อปลอมแปลงข่าวสารของเราขณะส่งไป การ เข้ารหัส(Encryption) เป็นการรักษาความปลอดภัยของข้อมูล ที่อยู่ในสภาพแวดล้อมที่ไม่ปลอดภัย

ซึ่งการเข้ารหัส(Encryption) มีพื้นฐานด้วยกันอยู่ 2 อย่างคือ การแทนที่ (substitution) และ การเปลี่ยนที่(transpasition)

2.1.1 ศัพท์เทคนิคและพื้นฐานขั้นต้น(Terminalogy and Background)

ถ้า S ต้องการส่งข่าวสารให้ R เรียก S ว่าผู้ส่ง(Sender) และเรียก R ว่าผู้รับ (Receiver) โดย S จะส่งข่าวสารโดยใช้ T เป็นตัวกลางส่งไปให้ผู้รับ Rเรียก T ว่า ตัวกลางการสื่อสาร(Transmission mediam)แต่ถ้ามี O ซึ่งต้องการข่าวสารนี้เช่นกัน และมาดึงข่าวสารนี้ไป เราเรียก O ว่าผู้กีดกัน(intercept หรือ intruder) เพราะในการส่งข่าวสารของ S ผ่านตัวกลาง T นั้น เป็นข่าวสารที่เปิดเผย ดังนั้น O อาจจะพยายามที่จะนำข่าวสารนี้ไปใช้โดยใช้วิธี

- interrupt โดยจะป้องกันไม่ให้ R ติดต่อข่าวสารนี้ไปใช้ประโยชน์
- intercept หาช่องทางในการติดต่อ และทำการปิดบังข่าวสารนั้น
- modify เปลี่ยนแปลงข้อมูลและดึงข้อมูลเข้ามา
- fabricate ทำการแปลงข้อมูลให้เหมือนกับข้อมูลที่ส่งมาจาก S

จากส่วนนี้จะเป็นปัญหาในการติดต่อส่งข้อมูลทำให้ไม่สำเร็จ การเข้ารหัส (Encrytion) เป็นเทคนิคในการแก้ปัญหา

ศัพท์เทคนิค(Terminalogy)

Encryption เป็นการเข้ารหัสข้อมูล ซึ่งจะทำให้ผู้อื่นไม่สามารถใช้ได้ Decryption เป็นการถอดรหัสข้อมูล เพื่อที่จะให้สามารถใช้งานข้อมูลนี้ได้ จะเรียกระบบนี้ว่า Cryptosystem

อัลกอริทึมการเข้ารหัส(Encryption Algorithms)

เริ่มต้นข้อมูลจะอยู่ในรูปของข้อความ(plaintext) เมื่อเข้ารหัสแล้วจะเรียกว่าข้อความที่เข้ารหัสแล้ว(Ciphertext)

ข้อมูล plaintext = P จะแสดงในรูปอนุกรมดังนี้

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

เขียนให้อยู่ในอีกรูปแบบ C = E(P) และ P = D(C)

C = Ciphertext , P = Plaintext , E = Encryption algorithms ,

D = Decryption algorithms

แต่ในระบบความปลอดภัย(Cryptosyste) จะได้สมการ

$$P = D(E(P))$$

รูปที่ 1 แสดงบล็อคการเข้ารหัส/ถอดรหัส(Encryptino/Decryption)

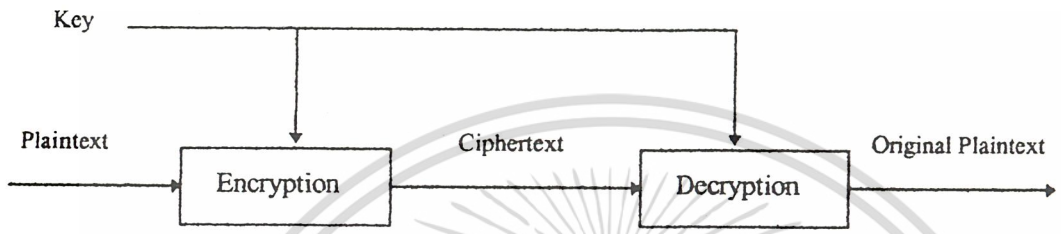
บางอัลกอริทึมการเข้ารหัส(Encryption algorithms) ใช้กุญแจ(Key K)ดังนั้นข้อมูล C จะสัมพันธ์กับข้อมูล P และ K จะได้ C = E(K,P)

E = Set ของ Encryption algarithms

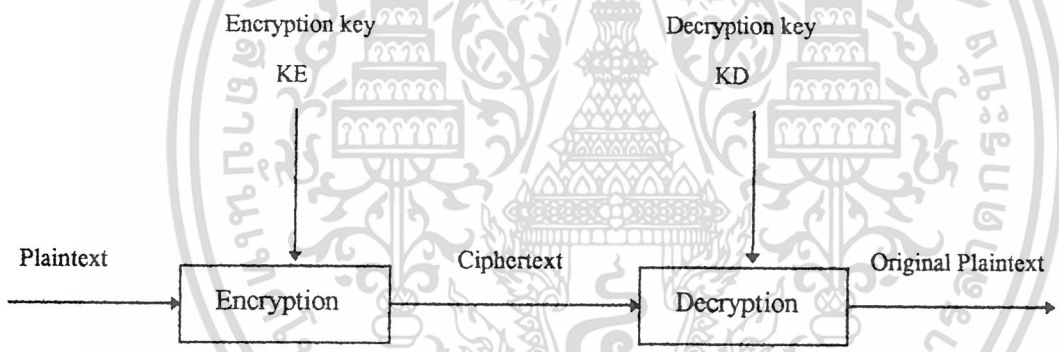
K = สมาชิกหนึ่งของ Algorithms

ดังนั้นในบางครั้งการการเข้ารหัส(Encryption) กับการถอดรหัส(Decryption) จะใช้กุญแจ K ที่เหมือนกัน จะได้ P = D(K,E(K,P)) ในบางครั้งจะใช้คู่กัน จะได้

$$P = D(Kp,E(Ke,P))$$



รูปที่ 2 แสดงบล็อคการเข้ารหัสถอดรหัสโดยใช้กุญแจตัวเดียว(Single-key Crytosystem)



รูปที่ 3 แสดงบล็อคการเข้ารหัสถอดรหัสโดยใช้กุญแจที่ต่างกัน(Two-key Cryptosystem)

การเข้ารหัสของข้อมูลหนึ่ง (Plaintext) อาจเปลี่ยนได้โดยการเปลี่ยนกุญแจ(Key) ถ้าผู้อื่นทราบอัลกอริทึมการเข้ารหัส(Encryption algorithms) ก็จะไม่สามารถถอดรหัสนำข้อมูลไปใช้ได้ เพราะไม่ทราบว่ากุญแจ(Key) ที่ใช้ ไซท์เฟอร์เท็ค(Ciphertext)ที่ไม่ต้องใช้กุญแจ (Key) ในการเข้ารหัสถอดรหัส เรียกคีย์เลส ไซท์เฟอร์(Keyless Cipher)

ระบบความปลอดภัย(Cryptography)เป็นวิธีการเข้รหั สกรใช้ ตรเข้ รหั ส (Encryption) ในการซ้อนข้อมูล(พยายามส่งข้อมูลไปให้จากผู้ส่งไปยังผู้รับ) ตามกฎของมันการวิเคราะห์ระบบความปลอดภัย(Cryptanalyst) การหาความหมายของข้อมูลที่ถูก

ซ้อนวิเคราะห์(Code) ทั้ง 2 อย่างนี้จะพยายามแปลงโค้ด(Code) ไปสู่รูปแบบ(form) เริ่มต้น(ป้องกันไม่ให้ผู้อื่นเข้ามายุ่ง)

การวิเคราะห์ระบบ(Cryptanalysis)

เป็นการหยุด(break) การเข้ารหัส (Encryption) คือ มันจะพยายามแปลความหมายของข้อมูลไซท์เฟอร์เท็ด(Ciphertext) หรือคือการเลือก Decrypting algorithms ให้ match กับ Encrypting algorithms ซึ่งทำได้ 3 ทางคือ

1. พยายามตีความหมายที่ละส่วน (break a single message)

2. พยายามจำแพทเทอร์น(patterns) ของข้อมูลที่เข้ารหัสแล้ว

โดยในลำดับสามารถตีความหมาย(break) งานถัดไป โดยทำการประยุกต์(applying) โดยตรงของการถอดรหัส(Decryption algorithms) ได้

3. พยายามหาจุดด้อยของการเข้ารหัส (Encryption algorithms) แล้วพยายามตีความ

การวิเคราะห์(Analyst) จะทำงานกับข้อมูลที่ทำการเข้ารหัส(Encryption algorithms) แล้ว รู้อัลกอริทึม(algorithms) ป้องกันข้อมูลรู้โค้ด(code) ของข้อมูล

การตีความหมายของการเข้ารหัส(Breakable Encryption)

การหยุดการเข้ารหัส(Encryption algorithms) คือการวิเคราะห์อัลกอริทึม (Algorithms) ที่จะใช้ในการตีความหมายและเวลาที่เพียงพอในการทำงาน จากตัวอย่าง การตีความหมายต้องการใช้ 10^{30} คำสั่ง(operations) ในเทคโนโลยีในปัจจุบัน(a current-technology computer) จะทำคำสั่ง 10^{10} คำสั่ง/วินาที ดังนั้นการแปลความหมายจะต้องใช้เวลา 10^{20} วินาที หรือ $\approx 10^{20}$ ปี

ใน Case นี้ จะไม่สนใจว่าทำได้หรือไม่ได้

ข้อสังเกต 2 ข้อ ของการตีความหมายของการเข้ารหัส (Break of Encryption algorithms)

1. การวิเคราะห์การเข้ารหัส (Cryparalyst) ไม่สามารถทำงานในระยะเวลาที่ยาวนานได้ ในตัวอย่าง การถอดรหัสจะต้องการ 10^{30} คำสั่ง แต่มากที่สุดที่เราอาจจะทำได้

เพียง 10^{15} คำสั่ง ที่ความเร็ว 10^{10} คำสั่ง/วินาที จะใช้เวลามากกว่า 1 วัน ซึ่งเป็นงานที่หนัก เราจะไม่เน้นตรงส่วนนี้

2. เวลาในการพัฒนาพื้นฐานทางเทคโนโลยี เริ่มเมื่อ≈ 40 ปีที่แล้ว 1940 หรือ กลาง 1950

ลักษณะของตัวอักษร(Representation of Characters)

การเข้ารหัสของคอมพิวเตอร์จะอ้างถึงภาษาอังกฤษซึ่งอาจจะแทนด้วยรหัสแอสกี้(ASCII)

- plaintext จะใช้ตัวพิมพ์ใหญ่
- ciphertext จะใช้ตัวพิมพ์เล็ก

การเข้ารหัสจะใช้ฟังก์ชั่นทางคณิตศาสตร์ ในการปแลงกลับไปมาระหว่างตัวพิมพ์ใหญ่และรหัสตัวเลข(numeric code)

Letter: A,B,C,D,...,W,X,Y,Z

Code: 0,1,2,3,...,22,23,24,25

ซึ่งในการกระทำเราจะกระทำกับรหัสของมันซึ่งจะเป็นรอบคือเมื่อถึง Z ซึ่งมีรหัส 25 จากนั้นก็จะกลับมาเริ่มต้น 1 ใหม่ เช่น A+3 = D หรือ K-1 = J และ Y+3 = B . ซึ่ง

คำตอบจะอยู่ระหว่าง 0-25

การหาเศษของการหารผลลัพธ์(modular arithmetic) การกระทำทางคณิตศาสตร์ เช่น การ mod n และจะอยู่ระหว่าง 0-n เช่น 95 mod 26 จะได้ 17 จาก 95-26-26-26 = 17 ∴ 0 คือค่าของ A ∴ คำตอบก็จะดูที่ค่า รหัส 17 = R ∴ 95 จะมีค่า = R

ในบทนี้เราจะสนใจอยู่ 2 อย่างคือ การแทนที่ (Substitutions) และ การเปลี่ยนที่ (transpotions)

2.1.2 การแทนที่ข้อมูลแบบตัวเดียว(Monomlphabetic Ciphers)เป็นการแทนที่ (Substitutions)

ซีซ่าไซท์เฟอร์(The Caesar Cipher)

จะใช้การเลื่อน(Shift)ไป 3 ตำแหน่ง โดย ข้อความ(Plaintext)คือ Pi และตัวที่ทำการเข้ารหัสแล้ว คือ Ci โดยใช้กฎ

$$Ci = E(Pi) = Pi + 3$$

∴ การเข้ารหัสโดย ซีซ่า ไซท์เฟอร์(Caesar Cipher) จะเป็นลักษณะดังนี้

Plaintext Letter = A,B,C,D,E,...,X,Y,Z

Ciphertext Letter = d, e, f, g,h,...,a, b,c

เช่น

ข้อมูล TREATY IMPOSSIBLE

เมื่อเข้ารหัสจะได้ wuhd wb l p sr vvl eoh

ข้อดีและข้อเสีย

ข้อดี คือ ทำให้ผู้อื่นสับสนไม่สามารถเข้าใจความหมายกับข้อมูลที่ทำการเข้ารหัส

ข้อเสีย คือ การเข้ารหัสแบบนี้ มีตัวอักษรที่บางครั้งผู้ต้องการข้อมูลอาจทำนายข้อมูลที่เราเข้ารหัสแล้วได้ เพราะการเข้ารหัสแบบนี้อยู่ใน form ที่ไม่ซับซ้อน

การวิเคราะห์ซีซ่า ไซเฟอร์(Cyptamatysis of The Caesar Cipher)

เป็นการง่ายที่จะทำการแปลความหมาย เช่น SS ก็จะถูกแปลงเป็น VV

เช่น มีข้อความที่เข้ารหัสแล้ว(Ciphertext) คือ

wklv phvvdjh lv qrw wrr kdug wr euhdn

การเข้ารหัสจะใช้ทั้งหมด 27 ตัว จาก A-Z และช่องว่าง(Blank) เป็นตัวแยกระหว่างคำช่องว่าง(Blank) อาจจะแปลงโดยอัตโนมัติหรือถูกยกเว้นไม่ต้องมีการแปลง ในการเข้ารหัสบางครั้งจะถูกลบทิ้งไป ซึ่งในการแปลความหมายสามารถเดาได้และเข้าใจได้เอง

ในภาษาอังกฤษ(English) คำสั้นๆจะใช้อักษรเพียงไม่กี่ตัว เช่น am,is,to,be,ฯลฯ เพราะฉะนั้น การที่จะแปลให้ความหมายเหมาะสม จะต้องทดลองนำค่าต่างๆมาแทนดูว่าคล้ายคลึงกัน(matching) กับคำในข้อมูลที่เข้ารหัส(Ciphertext) หรือไม่ เช่น wrr อาจจะเป็นได้ คือ see,too,add,odd,off ต้องใช้ sense

ตัวอย่าง ถ้า wrr เป็น too,wr จะเป็น to ∴ ลองแทนค่าในประโยค

wklv phvvdjh lv qrw wrr kdug wr euhdn

T--- ---------- -- -OT TOO ----- TO ------

ความยุ่งยากของการแทนที่ตัวอักษรตัวเดียวในการเข้ารหัสและถอดรหัส (Complexity of Monoalphabetic Enceryption and Decryption)

การเข้ารหัส(Eneryption) และการถอดรหัส(Decryption) จากอัลกอริทึม (algorithms) นี้สามารถกระทำได้โดยดูจากตารางที่เป็นสัดส่วนกัน

การวิเคราะห์การแทนที่ตัวอักษรตัวเดียวของระบบความปลอดภัย (Cryptanalysis of Monoalphabetic Ciphers)

การติดตามของซีซ่า Caesar Cipher สามารถใช้การแทนที่ (Monoalphabetic Ciphers)สังเกตจากคำที่สั้นๆที่คุ้นเคย ซึ่งทำให้เราสามารถเดาตัวที่จะมาแทนในข้อมูลได้ แต่ถ้าเป็นงานที่มีข้อมูลมากๆ ก็ควรใช้วิธีอื่น เพราะวิธีนี้จะทำให้ใช้เวลานาน และก็น่าเบื่อ

ความถี่ของการกระจายของตัวอักษร(Frequency Distributions)

แบ่งการวิเคราะห์จากความถี่ เช่น ในภาษาอังกฤษ ตัว E,T และ A จะถูกใช้มากกว่า J,Q และ Z อีกอย่างคือ เช่น ถ้าเป็นข้อมูลทางการแพทย์ X-ray ก็จะถูกใช้บ่อย ซึ่งจะไม่เกี่ยวกับตัวอักษร X

จากตาราง(Table) เป็นการแสดงความสัมพันธ์ความถี่ของตัวอักษร ของหนังสือบทหนึ่งที่เกี่ยวกับ คอมพิวเตอร์(Computing)และรหัส(Code) ของภาษาปาสคาล(Pascal) ซึ่งจะมีความถี่ใกล้เคียงกัน เพราะปาสคาล(Pascal) ใช้ คีย์เวิร์ด(Keywords) ภาษาอังกฤษ (English) ซึ่งตารางนี้สามารถใช้วิเคราะห์ข้อมูลที่เข้ารหัสแล้ว(Ciphertext) ได้

ตารางที่ 1 แสดงการจัดกลุ่มวัดความถี่ของแต่ละตัวอักษรในภาษาอังกฤษและที่ใช้ในภาษาปาสคาล (Pascal)

	English		Pascal	
Letter	Count	Percent	Count	Percent
a	3312	7.49	664	4.70
b	573	2.29	197	1.39
c	1568	3.54	878	6.22
d	1602	3.62	511	3.61
e	6192	14,00	1921	13.60
f	966	2.18	504	3.57
g	769	1.74	294	2.08
h	1869	4.22	478	3.39
i	2943	6.65	1215	8.60
j	119	0.27	6	0.04
k	206	0.47	87	0.61
l	1579	3.57	722	5.11
m	1500	3.39	270	1.91
n	2892	6.74	1157	8.19
o	3261	7.37	835	5
p	1074	2.43	340	2.41
q	116	0.26	12	0.08
r	2761	6.14	1147	8.21
s	3072	6.95	594	4.21
t	4358	9.85	1311	9.28
u	1329	3.00	377	2.66
v	512	1.16	127	0.89
w	748	1.69	193	1.36
x	123	0.28	139	0.98
y	727	1.64	137	0.96
z	16	0.04	5	0.03
ALL	44232		14121	

หากว่าข้อมูลมีน้อย ก็ไม่สามารถที่จะบอกได้ว่า ตัวอักษรนั้นเป็นตัวอักษรที่เราคาดไว้หรือไม่เนื่องจากข้อมูล เช่น ถ้า V มีมากในข้อมูลที่เข้ารหัสแล้ว(Ciphertext) ก็ไม่ใช่ว่า V จะ = E (E มีความถี่มากในภาษา English) เพราะข้อมูลมีน้อยเกินไปที่จะใช้ตัดสินได้

ถ้าข้อมูลยาวเพียงพอ ก็จะทำให้ตีความได้รวดเร็ว การใช้วิธีนี้และวิธีอื่นๆ การวิเคราะห์ที่ดีจะทำให้สามารถแก้ปัญหาได้ง่ายขึ้น แต่การเข้ารหัสที่ซับซ้อนมากๆ อาจจะทำให้เกิดปัญหาที่ยุ่งยากมากๆ

ระบบการเข้ารหัสของดิเลมนา(The Cryptographer's Dilemna)

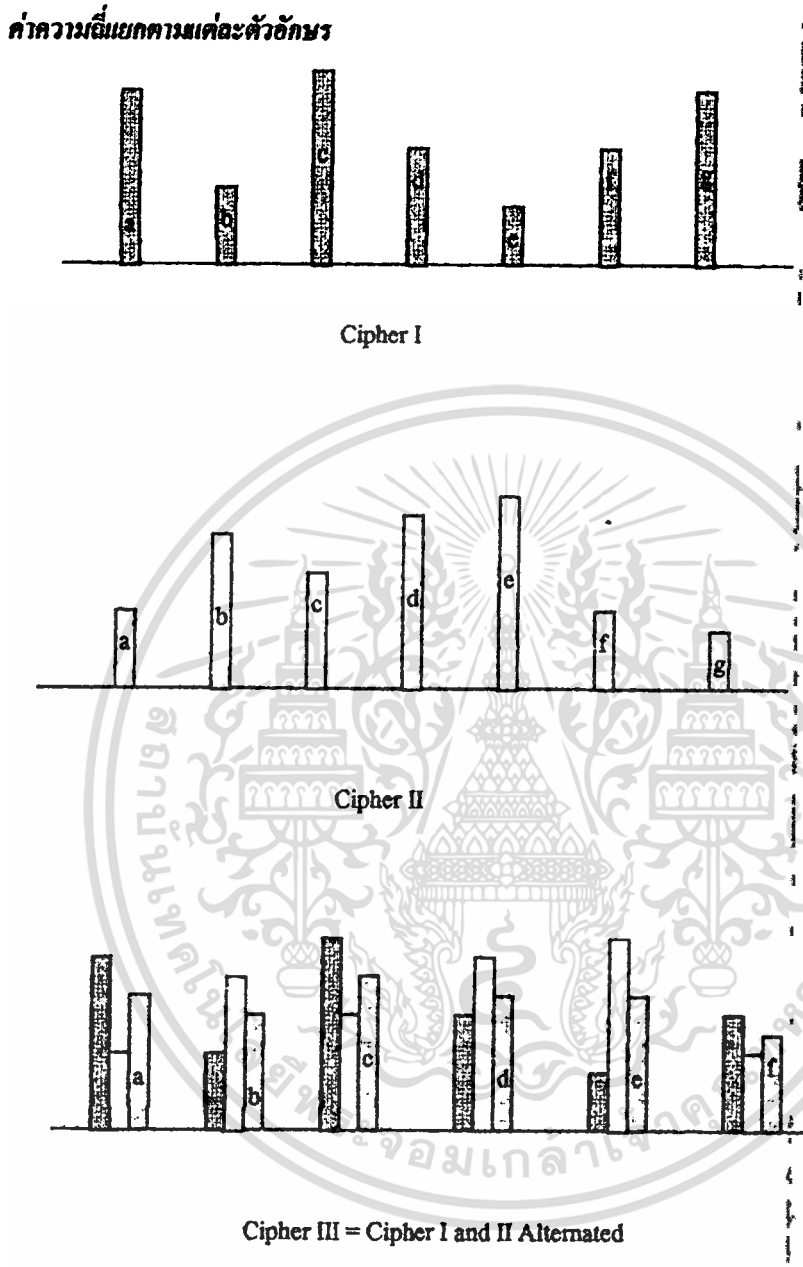
มีหลายเทคนิคที่จะเข้ารหัสข้อมูลและมีบางส่วนของข้อมูลที่เข้ารหัสแล้ว (Ciphertext) ที่ขัดขวางการใช้ประโยชน์ของเทคนิค(Technique) นั้น

การเข้ารหัสโดยใช้การแทนที่(Monoalphabetic) จะมีลำดับขั้นตอน ซึ่งจะพอทำให้ทราบร่องรอยการวิเคราะห์โค้ด(Code) ที่เข้ารหัส

ระบบการเข้ารหัส(Cryptographer) จะประดิษฐ์การเข้ารหัสขึ้นใหม่(วิธีใหม่)และจะดูว่าดีหรือไม่ จะใช้ความปลอดภัยของข้อมูลเป็นตัววัดว่าใช้ได้หรือไม่ ซึ่งเราจะต้องตรวจดูจุดอ่อนของมันดู

2.1.3 การแทนที่ของตัวอักษรหลายตัว(Polyalphabetic Substitution Ciphers)

จุดอ่อนของ การแทนที่ตัวอักษรตัวเดียว(Monoalphabetic Ciphers) คือ ความถี่ของมันมีผลกับข้อมูลจริง ซึ่งอาจจะไม่ปลอดภัย มันการจะไม่ให้ข้อมูลเพื่อให้วิเคราะห์ได้



รูปที่ 4 แสดงการแทนที่ในลักษณะตัวอักษรหลายตัว

เราจะใช้ทางใหม่โดยการรวมความถี่ที่มีค่าสูงและต่ำเข้าด้วยกัน เช่น ถ้า T อาจเป็น a และบาวครั้งอาจเป็น b และก็มี X มีลักษณะเช่นเดียว T คืออาจเป็นได้ทั้ย a และ b ดังนั้น เราจะเอาความถี่สูงสุดของ T ที่อาจจะเป็น a ผสมกับตวามถี่ต่ำสุดที่ X มีโอกาสเป็น a

จากรูป E1 (T) = a และ E2 (T) = b ขณะที่ E1 (X) = b และ E2 (X) = a

เราจะรวม 2 ส่วนเข้าด้วยกันโดยแยกเป็นส่วนของอักษาที่เข้ารหัส ส่วนแรกจะใช้อักษรที่ตำแหน่งคี่ของข้อความ(plaintext) ส่วนที่ 2 ถืออักษรที่อยู่ในตำแหน่งคู่ โดยจะสลับกันระหว่าง 2 ตารางตามตัวอย่างรูป

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 d g j m p s v y b e h k n q t w z c f i l o r u x

Table for Odd Positions

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

n s x c h m r w b g l q v a f k p u z e j o t y t i

Table for Even Positions

ตารางแรกเป็นการเข้ารหัส(การแทนที่) โดยใช้ฟังก์ชั่น (Function) ; $\pi 1(a) = (3*a) \bmod 26$ ขณะที่ ตารางที่สองจะใช้ฟังก์ชั่น(Function); $\pi 2(a) = ((5*a) + 13) \bmod 26$

ตัวอย่างการใช้งาน Table ทั้งสองในการเข้ารหัสข้อมูล TREATY IMPOSSIBLE

Data : TREAT YIMPO SSIBL E

เข้ารหัสโดย ตำแหน่งคี่ของข้อมูลให้ดูจากตารางคี่ (Table Odd) และ ตำแหน่งคู่ของข้อมูลให้ดูจากตารางคู่ (Table Even)

เมื่อทำการเข้ารหัสจะได้ดังนี้

Ciphertext : fumnf duvtf czysh h

จากตัวอย่าง S เป็นได้ทั้ง c และ z และ E เข้ารหัสได้ทั้ง m,hและ T เป็น f ทั้ง 2 ครั้ง I = y L,E =h ข้อมูลที่ได้อาจเป็นตัวเดียวกันหรือต่างกันก็ได้ขึ้นอยู่กับตำแหน่งของมัน

การเข้ารหัสโดยใช้ตัวอักษรหลายตัว โดยพิจารณาความถี่ของข้อความ(plaintext)

ตาราง เป็นการวิเคราะห์ความถี่ของดิคเคน(Dicken's)

ตารางที่ 2 แสดง ความถี่ของสองตัวอักษรในการเข้ารหัส

Letter	Count	Percent	Letter	Count	Percent
a	14	2.95	n	20	4.22
b	4	0.48	o	25	5.06
c	3	0.42	p	7	1.27
d	17	3.38	q	0	0.00
e	35	7.17	r	39	8.02
f	14	2.95	s	27	5.49
g	31	6.33	t	30	6.33
h	43	8.86	u	10	2.11
i	33	6.75	v	39	8.02
j	0	0.00	w	18	3.80
k	5	0.84	x	3	0.42
l	31	6.33	y	5	0.84
m	7	1.27	z	14	2.95
ALL	474				

จากตัวอย่างจะใช้การแทนตัวอักษรตัวเดียว 2 วิธี

1. $\pi 1(a) = a$ ถ้าเป็น E ก็จะ = e

2. $\pi 2(a) = 25 - a$ ถ้าเป็น E ก็จะ = v

จะใช้ 2 อักษรนี้ในการจำแนกสังเกตว่าe จะใช้ น้ อถ้ าv ชี้ ยก จะ ที g h จะ ช่วยให้ s และ t กระจายมากขึ้น

จาก 2 ข้อนี้ บังเอิญ 2 อักษรที่มีความถี่ต่ำเกิดชนกัน เช่น J และ Q ซึ่งทั้งคู่ เทียบ (map) เป็น j และ q

การชนกันมีผลต่อความถี่

การเลือกแก้สถานะนี้ให้เลือกสลับแบบ $\pi 1$ และถ้าระมัดระวังเลือก $\pi 2$ เป็นส่วนประกอบ $\pi 1$ ด้วย ถ้า $\pi 1$ เทียบ(map) แล้วมีความถี่ของตัวอักษร(letter) สูง เช่น E ไป เป็น x ดังนั้น $\pi 2$ การเทียบ(map) ให้ x มีความถี่ต่ำ $\pi 2$ จะทำให้ต่ำลงไม่ได้

ขยายการสลับที่

การใช้ลักษณะอื่นในการสลับที่ มี 3 ขั้น

1. ใช้การหมุน (rotation)
2. เพิ่มโอกาสในการแบ่งจำแนก

ตารางเวอร์เนีย (Vigenere tableaux)

ข้อเสียคือมีการเปลี่ยนตำแหน่งมาก การเข้ารหัสจะนำกุญแจคำ(Keyword) มาใช้เลือกในแนวตั้ง(Column) สำหรับเข้ารหัส

ตารางที่ 3 แสดงตารางเวอร์เนีย(TABLE VIGENERE TABLEAU)

	0	1	2
	0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5		
	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	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		
B	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		
C	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		
D	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		
E	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		
F	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		
G	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		
H	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		
I	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		
J	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		
K	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		
L	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		
M	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		
N	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		
O	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		
P	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		
Q	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		
R	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		
S	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		
T	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		
U	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		
V	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		
W	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		
X	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		
Y	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		
Z	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		

ตัวอย่าง ถ้าต้องการเข้ารหัส

BUT SOFT,WHAT LIGHT THROUGH YONDER WINDOW BREAKS

โดยใช้กุญแจคำ (Keyword) ว่า **juliet** วางไว้เหนือตัวอักษร(Character) ของข้อมูลแต่ละตัวอักษรและ ใช้กุญแจคำ(Keyword) วนตามตัวอย่างไปเรื่อยๆ

j u l i e t j u l i e t j u l i e t j u l i e t j u l i e t j u l i e t j u l i e

BUTSO FTWHA TLIGH TTHRO UGHYO NDERW INDOW BREAK S

เราจะกำหนดตัวอักษร(Charaters) ของกุญแจ(Keys) เป็นตัวเลข K1,K2,...,Kn เพื่อใช้อ้างข้อความ(Plaintext) และแต่ละข้อความ(Plaintext) จะเป็น Pi แนวตั้ง(Column) Ki ของตาราง จากตัวอย่างอักษรแรก (B) ถูกแปลงไปเป็นข้อมูลที่เข้ารหัสแล้ว (Ciphertext) ในแถวนี้ 1(B) แนวตั้ง(Column) 9(j) จากตารางนี้จะตรงกับตำแหน่งของ k จะได้

j u l i e t j u l i e t j u l i e t j u l i e t j u l i e t j u l i e t j u l i e

BUTSO FTWHA TLIGH TTHRO UGHYO NDERW INDOW BREAK S

k o e as yc q s i .....

การวิเคราะห์ระบบการแทนที่ของตัวอักษรหลายตัว

(Cryptanalysis of Polyalphabetic Substitutions)

วิธี ของ คาร์ซิสกี้(Kasiski Method Reapeated Patterns)

เป็นวิธีที่จะใช้กลุ่มและคำเต็มที่มีซ้ำๆ มาวิเคราะห์ เช่น ส่วนท้ายของคำ -th,-ing,ฯลฯ ตัวเริ่มต้น im-,in- และภายในคำ -eek-,-oat-,-our-,ฯลฯ และคำที่ใช้บ่อย of,and,to,with,are,is,ฯลฯ

ถ้าข้อมูลถูกเข้ารหัสโดยกุญแจคำ(keyword) ด้วยจำนวน n ตัวอักษรที่ใช้ในการหมุน และคำคำนั้นหรือกลุ่มคำนั้นปรากฏ k ครั้งในข้อความ (Plaintext) เพราะฉนั้น จะใช้ตัวอักษรชุดเดิมในการเข้ารหัส k/n ครั้ง

<u>ตัวอย่าง</u> ถ้ามี Keyword = 6 ตัวอักษร ซึ่งไม่ซ้ำกัน ให้ไปวางเหนือข้อมูล(Plaintext); ถ้าข้อความ(Plaintext Word) มีความยาวปรากฏมากกว่า 6 ครั้งต้องเข้ารหัสกุญแจคำ(keyword) เดิม อย่างน้อย 2 ครั้ง โดยตำแหน่งของกุญแจคำ(Keyword) จะกลับไปเหมือนเดิม(ใช้รหัส ของ keyword ที่หมุนซ้ำเดิม) เช่น ตัวอย่างถัดไป

ใช้กุญแจ(keyword) เท่ากับ dickens

d i c k e n s d i c k e n s d i c k e***n*** ***s d i c k*** ***e n***s di ckens dicke
ITWAS THEBE STOFT IMES***I*** ***TWAST*** ***HE***WOR STOFT IMESI
n s d i c k e n s d i c k e n s d i c k e n s d i c k e n s d i c k e n s d i c
TWAST HEAGE OFWIS DOMIT WASTH EAGEO FFOOL ISHNE
k e***n s d*** ***i c k e n*** s d i c k e n s d i c k e***n s*** ***d i c k e*** ***n***s d i c k e n s d
SS***ITW*** ***ASTHE*** EPOCH OFBEL IEF***IT*** ***WASTH*** ***E***EPOC HOFIN

คำว่า IT WAS THE เข้ารหัสด้วยวิธีของดิกเคน(nsdicken) ครั้งที่ 1 บรรทัดแรก ครั้งที่ 2 บรรทัดที่ 3 ซึ่งข้อความนี้มี 3 ครั้ง ใช้อักษรเข้ารหัส 8 ตัว

คำเดิมกลับมาใช้รหัสเดิม ซึ่งการจะวนกลับมาใช้รหัสเดิมก็ขึ้นกับความยาวของกุญแจคำ(Keyword)

การใช้วิธีของแคสซิสกี้(Kasiski) : การที่คำเดิมจะใช้รหัสเดิม ก็ส่วนมากจะเป็นคำที่สั้นๆ ดังนั้นการทำการคำนวนระยะทางตั้งแต่ต้นจนจบว่าควรจะใช้กุญแจ(Keyword) ยาวเท่าใด ซึ่งจะเหมาะสม เป็นดังนี้

คุณจะต้องตรวจดูว่ามีคำที่ซ้ำกันมากน้อยเพียงใดตั้งแต่ต้นของข้อมูลถึงส่วนท้ายของข้อมูลจากนั้นดูว่าคำที่ซ้ำกันมีระยะห่างแต่ละส่วนเท่าไร เพื่อทำการคำนวณหาความยาวของกุญแจ(Keyword) ที่เหมาะสม

จากตัวอย่างมี คำที่ซ้ำกันอยู่ 3 ส่วน เราจะเขียนเป็นตารางได้ดังนี้

Starting Position	Distance from Previous	Factors(ความยาวที่เป็นไปได้)
20	-	-
83	63(83-20)	3,7,9,21,63
104	21(104-83)	3,7,21

จากตาราง เราจะคาดได้ว่ากุญแจ(Keyword) ยาว 21,63 ตัวอักษรไม่น่าจะเป็นไปได้ เพราะตำแหน่งซ้ำเริ่มต้นที่ 20 และห่างจากคำซ้ำที่ 2 ไม่เป็นตามแฟคเตอร์ (Factors) ที่ได้ เราจึงคาดว่ากุญแจ(Keyword) น่าจะเป็น 3 หรือ 7 ซึ่งเราจะนำไปใช้ต่อไป

สรุปวิธีของแคสซิสกี้(Kasiski Method)

1. หาข้อความ(patterns) ที่ซ้ำกัน 3 ครั้งหรือมากกว่า
2. เขียนตำแหน่งเริ่มต้นของแต่ละข้อความ(patterns)
3. ตรวจดูความแตกต่างของระยะห่างตั้งแต่ต้นจนจบ
4. หาแฟคเตอร์(factors) ที่เป็นไปได้ของความห่าง(ความยาวของ Keyword ที่เป็นไปได้)
5. ความยาวของกุญแจ(Key) ที่จะใช้จะเป็นตัวใดตัวหนึ่งที่หาได้จากข้อ 4

การชี้ร่วมกันของอินเด็ก(Index of Coincidence)

จากตัวอย่างที่แล้ว ความยาวกุญแจ(Key) ที่อาจเป็นได้คือ 3 หรือ 7 ขั้นตอน ต่อไปให้ลองแบ่งข้อมูลเพื่อเข้ารหัสตามขนาดที่ได้ เช่น ถ้าลองกำหนดให้กุญแจ(Key) ยาว = 3 จากตัวอย่าง คุณก็จะได้กลุ่มของตัวอักษรที่เข้ารหัสแล้ว(Ciphertex) ห่างกัน = 3

S1 = {C1,C4,C7,C10,...} , S2 = {C2,C5,C8,C11,...} , และ

S3 = {C3,C6,C9,C12,...}

สังเกตตัวอักษรในกลุ่มที่มีการเข้ารหัสที่ใช้อักษรเดียวกัน ตัวใดตัวหนึ่ง จากนั้นเราจะสังเกตในอีกกลุ่มหนึ่งเช่นกัน แล้วนำมาเปรียบเทียบกัน ว่าอักษรที่เราสังเกตในแต่ละกลุ่มมีโอกาสเป็นตัวใดได้บ้าง

การเข้ารหัสโดยใช้การแทนที่ตัวอักษรตัวเดียว(Monoalphabetic) นั้นความถี่ของตัวอักษรภาษาอังกฤษ(English) กับความถี่ของอักษรที่ใช้เข้ารหัสที่ถูกต้อง ควรจะเหมือนกัน การเกิดร่วมกันของอินเด็ก(The Index of Coincidence)เป็นเครื่องมือในการวัดความแตกต่างของความถี่ที่จำแนกไว้ เราจะจำแนกข้อความ(Plaintext) ด้วยการแทนที่แบบตัวอักษรตัวเดียว (Monoalphabetic Substitution)จะทำให้ Ciphertext เมื่อจำแนกก็จะคล้ายๆกัน เราจะนำ 2 ตัวอักษรที่มีความถี่สูงและต่ำมาผสมกัน คือนำมาเฉลี่ยให้มีความน่าจะเกิดใกล้เคียงกัน ในการแบ่งจำแนกที่ดี เมื่อผสมกันแล้วทุกตัวอักษรควรจะปรากฏ ในความถี่ที่เท่ากันหรือใกล้เคียงกัน

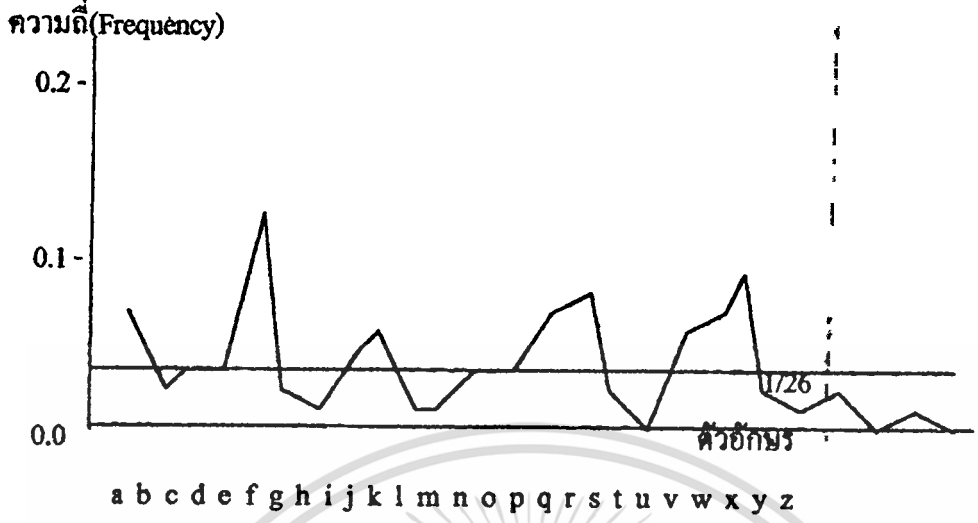
การเกิดร่วมกันของอินเด็ก(Index of Coincidence) เป็นตัวบอกว่าควรเข้ารหัสด้วย 1 ตัวอักษร(Monoalphabetic Substitution) หรือ 2,3 ตัวอักษร แต่จะวัดต่ำ(low)สูง (high) ของการจำแนกเท่านั้น จะไม่บอกว่าตัวอักษรของที่เข้ารหัสแล้วเหมือน (Ciphertex Matches) กับอักษรของข้อความ(Plaintext) ตัวใด

เราจะใช้ความน่าจะเป็นของแต่ละตัวในตารางการจำแนกความถี่ของตัวอักษรในภาษาอังกฤษ(English) มาใช้ เช่น ความน่าจะเกิดของตัวอักษร t = 0.0985 หรือ ความน่าจะเกิดของตัวอักษร x = 0.0028 ดังนั้นการสุ่มตัวอักษร(Ranom Leter) a-z จะเป็นดังนี้

$$Prob_a + Prob_b + Prob_c + ... + Prob_z = \sum_{i=a}^{i=z} Prob_i = 1$$

แต่ละตัวอักษรน่าจะมีโอกาสที่จะเกิดได้เท่ากัน จะได้

$$Prob_a = Prob_b = Prob_c = ... = Prob_z = 1/26 \cong 0.0384$$



รูปที่ 5 แสดงความยุ่งยากของการกระจายของตัวอักษรในภาษาอังกฤษ

มีค่าเฉลี่ยของกราฟ คือ ในแนวนอน (Horizental Straight) = 0.0384 ชินคอฟ(Sinkov) ได้กำหนดเส้นมาตราฐาน(Line base) = 0.0384 ซึ่งจะใช้เปรียบเทียบความถี่ของแต่ละตัวอักษร การหาค่าความแตกต่างของ ค่าความถี่ของแต่ละตัว ($Rfreq_i$) เช่น

ความสัมพันธความถี่ของตัวอักษร a(Relative Frequency of a) = $Rfreq_a$ - (1/26) จะทราบว่าเป็นค่าสูงสุด(peak) ถ้าผลเป็น + และถ้าเป็นค่า - จะเป็นค่าวัลเลย์(valley) ของอักษร a

เราจะใช้การยกกำลังสอง เพื่อป้องกันการหักล้างกันในขณะรวมกันจะได้

$$(Prob_a-(1/26))^2$$

การวัดประสิทธิภาพในการเข้ารหัสจะใช้

$$var = \sum_{i=a}^{i=z} Prob_i^2 - 0.0384$$

; var = ค่าความแตกต่างกับเส้นมาตราฐาน(Line base)

ถ้าคำนวณออกมาแล้วได้ค่าวา(var) = 0 แสดงว่าการเข้ารหัสสามารถทำได้ดีมีการใช้อักษรในการเข้ารหัสเฉลี่ยทุกตัวเท่าหรือใกล้เคียงกัน ทำให้ผู้ขัดขวางหรือผู้อื่นที่ต้องการข้อมูลถอดรหัสได้ยาก เทอม(term) ในแต่ละผลรวมของ $Prob_i^2$ อาจเกิดได้ 2 ตัวอักษรในเวลาหนึ่ง ซึ่งเราจะหาได้จากการนับจำนวนคู่และหารด้วย ผลรวมของคู่นี้เป็นไปได้ แต่เราจะไม่รู้ว่าอักษรนี้เกิดบ่อยแค่ไหนเพราะเราไม่ทราบอัลกอริทึม (Algorithm) ของการใช้(generate)แต่ละตัวอักษร แต่เราจะประมาณความน่าจะเป็นโดยสังเกตจากความถี่

สังเกตตัวอักษรในที่เข้ารหัส(Ciphertext Letters) สมมุติว่า $Freq_i$ เป็นความถี่ของอักษร i เราต้องการทราบว่าการจะสุ่มอักษร i ขึ้นมา 2 ครั้งเป็นเท่าไร ดังนั้น $Freq_i$ เป็นโอกาสที่จะเลือก i ครั้งแรก และ$(Freq_i-1)$ เป็นโอกาสที่จะเลือกอักษร i เป็นครั้งที่ 2 ดังนั้น $Freq_i*(Freq_i-1)$ เป็นโอกาสที่จะเลือกอักษร i 2 ครั้งติดกัน ถ้าคู่ของ (a,b) เหมือนกับคู่ (b,a) $\therefore$ เราจะนับทั้งคู่นี้ มีโอกาสเกิดขึ้นเป็น $Freq_i*(Freq_i-1)/2$ ของการเลือกคู่หนึ่งคู่ ดังนั้นรวมๆจะเป็น $n*(n-1)/2$ ของคู่อักษรที่เข้ารหัสแล้ว(Ciphertext) ของ n ตัวอักษร(Characters)

จะได้ $Freq_i*(Freq_i-1) / n*(n-1)/2$

; n = จำนวนตัวอักษรทั้งหมดที่ใช้ในอักษรที่เข้ารหัส(Ciphertext)

แต่ถ้าเราสุ่มเอาอักษร i ขึ้นมา 2 ครั้งจะมีโอกาสประมาณ $Prob_i^2$

การเกิดร่วมกันของอินเด็ก(Index of Coincidence(IC)) เป็นการประมาณตามแตกต่างจากการสังเกตข้อมูล

$$IC = \sum_{i=a}^{i=z} Freq_i*(Freq_i-1) / n*(n-1)/2$$

การเกิดร่วมกันของอินเด็ก(IC) เป็น rang ของ 0.0384 ซึ่งเป็นการแทนที่แบบหลายตัวอักษร(Polyalphabetic) และขอบเขต(rang)0.068 เป็นของการแทนที่โดยใช้ตัวอักษรตัวเดียว(Monoalphabetic) จากตัวร่วมในภาษาอังกฤษ(English)

การเกิดร่วมกันของอินเด็ก(IC) สามารถใช้ทำนายจำนวนตัวอักษรที่มีโอกาสจะเกิดขึ้นได้ ตาราง 2.6 แสดงค่า การเกิดร่วมกันของอินเด็ก(IC)ของตัวอักษรที่ใช้ในการแทนที่แบบตัวอักษรหลายตัว(Polyalphabetic Substitution) สังเกตว่าจะใช้ได้ดีกับการใช้ตัวอักษรจำนวนน้อยๆ

ตารางที่ 4 แสดงจำนวนอักษรในการเข้ารหัสและค่าการเกิดร่วมกันของอินเด็ก (Index of Coincidence)

alphabets	1	2	3	4	5	10	large
IC	.068	.052	.047	.044	.044	.041	.038

จากตัวอย่างก่อนกุญแจ(Key) อาจจะเป็น 3 หรือ 7 การเกิดร่วมกันของอินเด็ก (IC) จะไม่แตกต่างระหว่าง 3 หรือ 7 ให้หาการเกิดร่วมกันของอินเด็ก(IC) ของแต่ละกลุ่ม เช่น ใช้กุญแจ(Keyword) ยาว 3 จะได้ S1 = {C1,C4,C7,...} S2 = {C2,C5} S3 = {C3,C6,...} จากนั้นหาค่าการเกิดร่วมกันของอินเด็ก(IC) ของแต่ละกลุ่ม คือ IC(S1),IC (S2) และ IC(S3) ถ้าได้ = 0.068 ก็แสดงว่าถูกต้อง แต่ถ้าไม่ใช่ให้ลองเปลี่ยนไปใช้กุญแจ (Key) ยาว = 7 ดู และหาการเกิดร่วมกันของอินเด็ก(IC) ของแต่ละกลุ่ม ซึ่งจะได้ = 0.068 ถ้ากุญแจ(Keyword) นี้ ถูกต้อง

สรุปการเข้ารหัสแบบการแทนที่โดยใช้ตัวอักษรหลายตัว(Polyalphabetic Ciphers)

ขั้นตอนการวิเคราะห์ข้อมูล

1. ใช้วิธีแคสซิสกี้(Kasiski) ทำนายจำนวนตัวอักษร
2. หาค่าการเกิดร่วมกันของอินเด็ก(IC) จากขั้นที่ 1
3. แยกอักษรที่เข้ารหัสแล้ว(Ciphertext) เป็นกลุ่มย่อย(subset) แล้วตรวจสอบการเกิดร่วมกันของอินเด็ก(IC)แต่ละกลุ่มย่อย(subset)

การเข้ารหัสแบบเวอร์แนม(The Vernam Cipher)

เป็นวิธีการเข้ารหัสอีกแบบหนึ่งที่ใช้ได้อย่างมีประสิทธิภาพ วิธีการก็คือจะทำการสุ่มตัวเลขขึ้นมาโดยที่ตัวเลขที่สุ่มขึ้นมานั้นจะไม่ซ้ำกัน จานั้นนำตัวเลขเหล่านั้นมาทำการรวมกัน(Combine) กับข้อความ(Plaintext) เมื่อได้ผลรวมแล้วก็จะทำการมอด(mod) 26 ซึ่งก็จะได้ค่ารหัสของตัวอักษร(Ciphertext)

การเข้ารหัสแบบนี้สามารถป้องกันผู้ที่ไม่เกี่ยวข้องกับข้อมูลได้ดี เนื่องจากว่าข้อมูลที่เข้ารหัสแล้ว(Ciphertext) จะไม่แสดงข้อความ(Pattern) ของกุญแจ(Key) ซึ่งทำให้ถอดรหัสได้ยาก เช่นตัวอย่าง

มีข้อมูลที่จะทำการเข้ารหัสคือ

ข้อมูล	V	E	R	N	A	M	C	I	P	H	E	R
ค่าของอักษร	21	4	17	13	0	12	2	8	15	7	4	17

เมื่อทำการเข้ารหัสจะได้ดังนี้

Plaintext	V	E	R	N	A	M	C	I	P	H	E	R
numeric equivalent	21	4	17	13	0	12	2	8	15	7	4	17
+ random number	76	48	16	82	44	3	58	11	60	5	48	88
= sum	97	52	33	95	44	15	60	19	75	12	52	105
= mod 26	19	0	7	17	18	15	8	19	23	12	0	1
Ciphertext	T	A	H	R	S	P	I	T	X	M	A	B

ดังนั้นจะได้ข้อมูลที่เข้ารหัสแล้ว

VERNAM CIPHER = tahrspitxmab

จากตัวอย่างจะเห็นได้ว่า อักษรตัวเดียวกัน คือ E จะใช้ตัวเลขที่สุ่มขึ้นมาเหมือนกัน คือ 48 และเมื่อเข้ารหัสก็จะได้อักษรตัวเดียวกัน คือ A อีกกรณีหนึ่ง คือ ตัวอักษร

ข้อมูลที่เข้ารหัสแล้ว(Ciphertext) T เกิดขึ้นซ้ำกัน แต่มาจากข้อความ(Plaintext) คนละตัว คือมาจาก V และ I ซึ่งผู้อื่นจะถอดรหัสได้ยาก

การเข้ารหัสข้อมูล 2 ข้อมูลด้วยครั้งเดียว (Dual-Massage Entrapment)

การเข้ารหัสข้อมูล 2 ข้อมูลด้วยครั้งเดียว ผู้อื่นจะไม่ทราบข้อแตกต่าง เรียกว่า ดัมมี่(dummy) เป็นกุญแจ(Key) การเข้ารหัสของ X ด้วย Y หรือกับเข้ารหัส Y ด้วย X เช่น ตัวอย่าง

จากข้อมูล d i sr eg a rd t his mes s age กุญแจ (key(dummy))

ข้อมูลอีกส่วน THIS MASSAGEIS CRUCIAL ข้อความ(message)

เมื่อทำการเข้ารหัสจะใช้ตารางเวอร์เนีย(VIGENERE TABLEAU) จะได้

Ciphertext : wpajqejvdzlqkovvmuigp

การเข้ารหัสแบบนี้เป็นการใช้ข้อมูลที่มีอยู่แล้วมาทำการเข้ารหัสซึ่งกันและกัน โดยไม่ต้องหากุญแจ(keyword) จากที่อื่น ทำให้เกิดความสับสนในการเข้ารหัส

สรุปการแทนที่(Substitutions)

มีผลกับการเขียนรหัสลับ เป็นการใช้อักษรตัวอื่นมาแทนที่ข้อมูลเดิมทำให้ผู้อื่นไม่สามารถเข้าใจความหมายของข้อมูลนั้น

ขั้นตอนในการวิเคราะห์ระบบการเข้ารหัส(Cryptanalytic) ทั่ว ๆ ไป

1. หาความถี่ของตัวอักษรแต่ละตัวอักษรในข้อความที่เข้ารหัสแล้ว(Ciphertext)
2. หาการเกิดร่วมกันของอินเด็ก(Index of Coincidence)
3. พิจารณาลักษณะของตัวอักษร ความเหมือนหรือคำบางคำ
4. วิเคราะห์ข้อความ(คำ)ซ้ำ และใช้กฎแคสสิสกี้(Kasiski)
5. ความขยัน โชค ความฉลาด(Sence) และการเดาที่มีประสิทธิภาพ

2.1.4 การเคลื่อนย้ายตำแหน่งในการเข้ารหัส(Transpositions(Permutations))

เป็นการเข้ารหัสโดยการนำเอาข้อมูลมาทำการเปลี่ยนตำแหน่งโดยที่ข้อมูลที่ส่งมายังคงเป็นข้อมูลจริงเพียงแต่มีการเปลี่ยนตำแหน่งเท่านั้นเพื่อให้ผู้อื่นที่ไม่เกี่ยวข้องเกิดความสับสน

การเคลื่อนย้ายในแนวตั้ง(Columnar Transpositions)

ในการเปลี่ยนตำแหน่งนั้นเราจะใช้การเปลี่ยนตำแหน่งของแนวตั้ง(Column) มาทำการสลับกัน เช่น

C1 C2 C3 C4 C5

C6 C7 C8 C9 C10

C11 C12 etc.

ถ้าข้อมูลอยู่ในลักษณะนี้เราจะได้ข้อมูลที่เข้ารหัสแล้วเป็นดังนี้

C1C6 C11...C2C7C12...C3C8,etc.

การเข้ารหัสจะถูกจัดเรียงข้อมูลใหม่ในแนวตั้ง(Column) เป็นต้น ดังตัวอย่างนี้

นำข้อมูลมาจัดเรียงในรูปแบบของแนวตั้ง(Column) และแนว(Row) ตามที่เรากำหนด เช่น

ข้อความ (Plaintext):

T H I S I

S A M E S

S A G E T

O S H O W

H O W A C

O L U M N

A R T R A

N S P O S

I T I O N

W O R K S

เมื่อเข้ารหัสแล้วจะได้

(Ciphertext) : tssoh oaniw haaso lrsto imghw

utpir seeoa mrook istwc nasns

ข้อมูลเป็นคู่(Digrams), สามตัว(Trigrams)และเป็นข้อความ(Patterns)

เราจะสังเกตความถี่ของตัวอักษรที่เกิดขึ้นในแต่ละแบบ เช่น คู่ตัวอักษรที่อยู่ติดกัน เรียกว่าไดแกรม(Digrams) อย่างเช่น "-re-","-th-","-en-" และ "-ed-" เป็นต้น เป็นคู่ตัวอักษรที่เกิดขึ้นได้บ่อยมากในภาษา อังกฤษ ส่วนไตรแกรม(Trigrams) เป็นตัวอักษรที่เรียงติดกันสามตัว ตามตัวอย่างในตาราง

ตารางที่ 5 แสดงลักษณะข้อมูลแบบไดแกรมและไตรแกรม

Digrams	Trigrams
EN	ENT
RE	ION
ER	AND
NT	ING
TH	IVE
ON	TIO
IN	FOR
TF	OUR
AN	THI
OR	ONE

การวิเคราะห์การเข้ารหัสโดยใช้ไดแกรม(Cryptanalysis by Digram Analysis)

คือการตรวจสอบความถี่ของตัวอักษร ซึ่งอาจจะทำให้ทราบแนวทาง ซึ่ง 2 ข้อความ ที่ต่างกันของข้อมูลที่เข้ารหัส(Ciphertext) อาจจะทำให้ทราบคู่ตัวอักษรที่ติดกันของข้อความ(Pliantext) เทียบดูจากตาราง ซึ่งจะทำการเปรียบเทียบกับข้อมูลที่เข้ารหัส (Ciphertext) โดยจะกำหนดเป็นชุด(Block) (Ciphertext ชุดหนึ่ง) แล้วเปรียบเทียบดังรูป

ฉะนั้นจะหาความถี่ของแต่ละคู่ตัวอักษรและนำมาเปรียบเทียบกันว่าคู่ไหนมีโอกาสที่จะเป็นไปได้มากที่สุด

จากข้อมูลที่เข้ารหัส(Ciphertext) ในตัวอย่างที่แล้ว

เราจะเปรียบเทียบโดยกำหนดข้อมูลที่เข้ารหัสแล้ว(Ciphertext) เป็นชุด(Block) จากตัวอย่างจะใช้ 7 ตัวอักษรใน 1 ชุด(Block) ที่จะใช้เปรียบเทียบ จากนั้นนำไปเปรียบเทียบที่ละส่วนและดูว่า คู่อักษรตัวไหนมีโอกาสเป็นไปได้มากที่สุด โดยอาจจะดูจากตารางเทียบไปด้วย

t	n			n			n			n			n			n
s	i		t	i			i			i			i			i
s	w		s	w		t	w			w			w			w
o	h		s	h		s	h		t	h			h			h
h	a		o	a		s	a		s	a		t	a			a
o	a		h	a		o	a		s	a		s	a		t	a
a	s		o	s		h	s		o	s		s	s		s	s
	o		a	o		o	o		h	o		o	o		s	o
	l			l		a	l		o	l		h	l		o	l
	r			r			r		a	r		o	r		h	r
	s			s			s			s		a	s		o	s
	t			t			t			t			t		s	t
	o			o			o			o			o			o

ตารางที่ 6 การเปรียบเทียบแบบเคลื่อนที่ (Moving Comparsions)

จากตัวอย่างเราสามารถนำมาเขียนเป็นตารางเพื่อหาค่าความน่าจะเป็นเปรียบเทียบกันในแต่ละคู่ตัวอักษรที่เกิดขึ้นจากการเทียบชุด(Block) แต่ละส่วนได้ดังนี้

ตารางที่ 7 แสดงความน่าจะเป็นเทียบกันในแน่ละคู่ตัวอุักษร

First Letter	Second Letter, Digram, Relative Frequency			
c1 t	n tn 17	i ti 1024	w tw 99	h th 529
c2 s	i si 339	w sw 2	h sh 61	a sa 69
c3 s	w sw 2	h sh 61	a sa 69	a sa 69
c4 o	h oh 3	a oa 45	a oa 45	s os 262
c5 h	a ha 555	a ha 555	s hs 41	o ho 216
c6 0	a oa 45	s os 262	o oo 124	l ol 123
c7 a	s as 460	o ao 1	l al 559	r ar 560
Mean	203	279	143	261
Std. Dev.	223	356	172	191

จากตารางจะสรุปได้ว่าคู่ตัวอักษรที่ตารางช่องสุดท้ายมีโอกาสเป็นไปได้มากที่สุด จากนั้นเราจะนำคู่ของไดแกรม(Digrams) ที่มีโอกาสเป็นไปได้มากที่สุดมาทำการเปรียบเทียบในแต่ละส่วนเหมือนลักษณะที่แล้วเพื่อหาไตรแกรม(Trigrams) และข้อความ(Patterns) อื่น ๆ ต่อไป

2.2 การเข้ารหัสถอดรหัสโดยวิธีการของอาร์เอสเอ(RSA)

อาร์เอสเอ(RSA) คือการใช้กุญแจที่เปิดเผย(a public key) ของระบบที่ใช้สำหรับการเข้ารหัสและยืนยันข้อมูล อาร์เอสเอ(RSA)ได้ถูกสร้างขึ้นโดยรอน ไรเวส(Ron Rivest) อดิ ชาไม(Adi Shamir) และลีออน อเดลแมน(Leonard Adleman) ในปี 1977 เราจะศึกษาในเรื่องของตัวเลข(number) และแฟคเตอร์(factor) ซึ่งการเข้ารหัสและถอดรหัสเพื่อไม่ให้มีการก้าวก่ายในส่วนของกุญแจที่ไม่เปิดเผย(private key) ของ ผู้ใช้(user) เอง

เท่านั้น แต่ละคนสามารถจะส่งข้อมูลที่ทำการเข้ารหัสหรือนำข้อมูลออกมาใช้ได้ในส่วนของกุญแจที่เปิดเผย(public keys)

ระบบอาร์เอสเอ(RSA) เป็นระบบที่มีความซับซ้อนของแฟคเตอร์(factoring) แต่ถ้าทราบวิธีของแฟคเตอร์(factoring) แล้วจะสามารถแปรความหมายของข้อมูลที่เข้ารหัสโดยอาร์เอสเอ(RSA) ได้ไม่ยาก

ทำไมจึงนิยมใช้การเข้ารหัสแบบอาร์เอสเอ(RSA) มากกว่าแบบดีอีเอส(DES)

อาร์เอสเอ(RSA)จะสนับสนุนและทำให้ดีอีเอส(DES)มีประสิทธิภาพในการเข้ารหัสซึ่งจะใช้ร่วมกันในการรักษาความปลอดภัยของข้อมูลในระบบสื่อสาร

อาร์เอสเอ(RSA) จะมี 2 ฟังก์ชั่น(functions)-ที่สำคัญซึ่งดีอีเอส(DES) ไม่มี การเข้ารหัสข้อมูลบ่อยครั้งที่อาร์เอสเอ(RSA) และดีอีเอส(DES) จะใช้งานร่วมกัน คือ เมื่อข้อมูลถูกเข้ารหัสโดยการสุ่มกุญแจของดีอีเอส(DES Key) ก่อนที่จะส่งออกไปก็จะมีการนำกุญแจของดีอีเอส(DES Key) มาเข้ารหัสด้วยอาร์เอสเอ(RSA) และจะถูกส่งออกไปพร้อมกับข้อมูล (Ciphertext) ในช่องทางของส่วนรวม(Public Channel) ข้อมูลในส่วนนี้จะไม่เป็นความลับคือจะไม่มีการเข้ารหัส

คุณอาจสงสัยว่าทำไมไม่ใช้อย่างใดอย่างหนึ่งเลยในการเข้ารหัส เนื่องจากอาร์เอสเอ(RSA) จะทำได้ดีกับข้อมูลที่สั้น ๆ แต่ดีอีเอส(DES) หรือการเข้ารหัสแบบอื่น ๆ สามารถที่จะเข้ารหัสข้อมูลยาว ๆ ได้ดีกว่าการเข้ารหัสแบบอาร์เอสเอ(RSA)

บางครั้ง การเข้ารหัสแบบอาร์เอสเอ(RSA) ไม่จำเป็นอาจใช้การเข้ารหัสแบบดีอีเอส(DES) ก็เพียงพอ เช่นในผู้ใช้คนเดียว(Singleuser) แต่ถ้าเป็นผู้ใช้หลายคน (Multiuser) ระบบต้องการการเข้ารหัสที่มีความปลอดภัยในการสูง ข้อมูลซึ่งอาจจะต้องใช้อาร์เอสเอ(RSA) หรือกุญแจที่เปิดเผย(Public Key) อื่น ๆ ของระบบ

อาร์เอสเอ(RSA) กับความปลอดภัยของข้อมูล

อาร์เอสเอ(RSA) จะใช้การคำนวณมาเกี่ยวข้องกับการเข้ารหัสซึ่งจะทำให้ผู้ที่ต้องการจะนำข้อมูล ไปใช้โดยที่ไม่มีสิทธิกับข้อมูลนั้นก็ไม่สามารถที่จะนำข้อมูลนี้ไปใช้ได้หรืออาจต้องใชเวลานานในการแปลความหมายของข้อมูล(ถ้าไม่ทราบ Key หรือวิธีการในการถอดรหัส)

ลักษณะการทำงานของอาร์เอสเอ(RSA)

การใช้อาร์เอสเอ(RSA) จะทำการขยายข้อมูลเล็ก ๆ เท่านั้นสำหรับการเข้ารหัสนั้นข้อมูลจะมี ความยาวซึ่งจะอยู่ในชุด(Block) ที่ซับซ้อน ซึ่งความยาวจะเป็นโมดูล (Modulus) (หลาย ๆ Application มีขนาด 512 bit) สำหรับการแสดงข้อมูลจริง ซึ่งข้อมูลจะไม่มีการเข้ารหัสและไม่มีการขยาย อย่างไรก็ดีกุญแจ(Key) ที่เข้ารหัสจะถูกแสดงต่อท้ายข้อมูลชุดนี้ ซึ่งจะมีความยาว 1 ชุด (Identity และ Public Key.) เพื่อป้องกันบุคคลอื่นใช้กุญแจ(Key) ปลอมในการถอดรหัสข้อมูล

การใช้อาร์เอสเอ(RSA)ในการเข้ารหัส

อาร์เอสเอ(RSA) จะใช้ร่วมกับกุญแจที่เก็บเป็นความลับ(Secrect-Key) ของระบบเช่นดีอีเอส(DES) ขั้นตอนก็คือจะใช้ดีอีเอส(DES) เข้ารหัสข้อมูลก่อน ซึ่งสิ่งนี้จะกลายเป็นข้อมูลทางดิจิตอล(Digital) ของอาร์เอสเอ(RSA) สมมุติว่าคุณต้องการส่งข้อมูลไปให้ Jim คุณจะต้องเข้ารหัสข้อมูลก่อนด้วยวิธีการของดีอีเอส(DES) โดยจะสุ่มกุญแจของดีอีเอส(DES Key) ขึ้นมาแล้วทำการเข้ารหัสจากนั้นใช้กุญแจที่เปิดเผย(Public Key) ของ Jim เข้ารหัสกุญแจของดีอีเอส(DES Key) อีกครั้งหนึ่ง หลังจากเข้ารหัสเสร็จเรียบร้อยแล้ว ก็ทำการส่งข้อมูลทั้งสอง (RSA Digital) ไปให้ Jim และ Jim ก็จะทำการถอดรหัสกุญแจของดีอีเอส(DES Key) ด้วย กุญแจที่ไม่เปิดเผย(Private Key) ของ Jim และใช้กุญแจของดีอีเอส(DES Key) ถอดรหัสข้อมูลอีกครั้งหนึ่ง

การใช้อาร์เอสเอ(RSA) แสดงข้อมูล

สมมุติว่า คุณต้องการส่งข้อมูลไปให้ Jim คุณจะต้องกำหนดขนาดของข้อมูลเรียกว่าแฮชวาร์ลู(Hash Value) เพื่อที่จะสามารถสร้างชุด(Block) ในการส่งข้อมูลได้ถูกต้อง ซึ่งแฮชวาร์ลู(Hash Value) จะเป็นสัญญาณทางดิจิตอล(Digital Signature) ของข้อมูลแล้วก็ทำการเข้ารหัสข้อมูลด้วยกุญแจที่ไม่เปิดเผยของอาร์เอสเอ(RSA Private Key) ของคุณ (Your Digital Signature) จากนั้นส่งไปให้ Jim เมื่อได้รับข้อมูลและสัญลักษณ์(Signature) ก็จะถอดรหัสสัญลักษณ์(Signature) ด้วยกุญแจที่เปิดเผย(Public Key) ของคุณแล้วเขาก็จะหาข้อมูล โดยใช้แฮชฟังก์ชั่น (Hash Function)เหมือนกับที่คุณ

ใช้แล้วก็จะ ทำการเปรียบเทียบข้อมูลกับสัญลักษณ์(Signature) ถ้ามีค่าเท่ากันแสดงว่าข้อมูลนั้นถูกต้อง แต่ถ้าไม่เท่าแสดงว่าข้อมูลอาจจะมาจากที่อื่นหรืออาจถูกดัดแปลงระหว่างการส่งมา Jim ก็จะส่งข้อมูลกลับคืน(Reject) ข้อมูลนั้น

อาร์เอสเอ(RSA) ช่วยในการตรวจสอบความผิดพลาดในการส่งข้อมูล

อาร์เอสเอ(RSA) จะตรวจดูว่ามีการดัดแปลงแก้ไขข้อมูลหรือไม่ถ้ามีอาร์เอสเอ (RSA) จะไม่ส่งข้อมูลเหล่านั้นไปให้กับผู้รับ

อาร์เอสเอ(RSA) ช่วยในการตรวจสอบไวรัสคอมพิวเตอร์(Viruses Computer) การตรวจสอบไวรัสของวิธีการของอาร์เอสเอ(RSA) จะคล้ายกับการตรวจสอบ (Transmission Error) คือสามารถตรวจสอบการเปลี่ยนแปลงข้อมูลของไฟล์ที่เก็บอยู่ในดิสก์(Disk) ซึ่งอาจถูกกระทำโดยไวรัส(Viruses) การตรวจสอบไวรัส(Viruses) ของอาร์เอสเอ(RSA) จะมีการกำหนดสัญลักษณ์ของไฟล์(File) ทุกไฟล์และจะมีการทำเครื่องหมายของการเปลี่ยนแปลงครั้งสุดท้ายที่เกิดขึ้น ถ้าหากสัญญาณผิดพลาด(Signture Fail) (ถูก Verify) มีการเปลี่ยนแปรงเกิดขึ้น อาร์เอสเอ(RSA) ก็จะฟ้องว่าข้อมูลมีการเปลี่ยนแปลง ซึ่งอาจจะถูกระทำโดย ไวรัส(Viruses) แต่ในบางครั้งอาจถูกกระทำโดย สภาพแวดล้อมอื่น ๆ เช่น โค้ดโปรแกรม(Soruce Code) หรือ ปัญหาทางฟิสิกส์คอล(Physical) ของอาร์ดดิสก์(Harddisk) แต่ถ้าสงสัยว่าเกิดจากไวรัส(Viruses) ก็ควรจะสั่งให้โปรแกรมทำงาน(Run Program Detection Viruses) ตรวจสอบดู

2. 2.1 คุณสมบัติทางคณิตศาสตร์(Properties of Arithmetic)

เราจะใช้จำนวนเฉพาะ(Prime numbers),การหาร(divisors) และแฟคเตอร์(factor) เป็นหลักของการรักษาความปลอดภัยอัลกอริทึมการเข้ารหัส(Encrytion Algorithms) เราจะพิจารณาคุณสมบัติข้อจำกัดและการใช้งานของวิธีการทางคณิตศาสตร์ เพื่อใช้ในระบบการเข้ารหัส(Cryptosystems)

อินเวอร์ท(Inverses)

ให้ Θ เป็น การทำงาน(Operation) บนเลขจำนวน

<u>ตัวอย่าง</u> Θ อาจเป็น + หรือ * จำนวน I จะถูกเรียกว่าตัวแปร(identity) สำหรับ Θ ถ้า I Θ x = x และ

x Θ I = x สำหรับ x ทุกตัว เช่น 0 คือ ตัวแปร(identity) สำหรับ + ดังนั้น x + 0 = x และ 0 +x = x ซึ่งคล้ายกับ 1 คือตัวแปร(identity) สำหรับ *

ให้ I เป็นตัวแปร(identity) ของ Θ, จำนวน b ถูกเรียกว่าอินเวอร์ท(invers) ของ a ภายใต้ Θ ถ้า a Θ b = I,ตัวแปร(identity) จะเก็บการจัดการ(entire operation) ดังนั้น I Θ I = I บางครั้งเรียกอินเวอร์ท(invers) ของ a เป็น a^{-1}

ในการบวกอินเวอร์ท(invers) ของ a คือ -a ดังนั้น a+(-a) = 0

ในการคูณอินเวอร์ท(invers) ของ a คือ 1/a ดังนั้น a*(1/a) = 1

จำนวนเฉพาะ(Primes number) คือ จำนวนเต็มบวกที่ไม่มีเลขจำนวนใดหารลงตัวได้ ยกเว้น 1 และตัวมันเอง จำนวนที่ไม่ใช่จำนวนเฉพาะ(Primes number) คือคอมโพสิท (COMPOSITE)

จำนวนเฉพาะ(Primes)

จำนวนเฉพาะ(A prime number) คือ จำนวนเต็มบวกที่ไม่สามารถแยกย่อยออกไปได้ คือ จะสามารถการลงตัวได้ด้วยตัวของมันเอง และ 1 เท่านั้น เช่น 2,3,5,7,11,... เป็นต้น แต่ 4 ซึ่งสามารถแยกย่อยได้เป็น (2*2), 6 จะเป็น (2*3) และ 9 เป็น (3*3) ค่าเหล่านี้จะไม่เป็นจำนวนเฉพาะ(prime number)จะเป็นคอมโพสิท(composite)

ตัวหารร่วมมาก (Grest Common Divisor)

ตัวหาร่วมมาก(Grest Common Divisor)ของ 2 จำนวน a และ b มักจะเขียนแทนด้วย gcd(a,b)คือ จำนวนเต็มที่ค่ามากที่สุดที่หาร a และ b ได้ลงตัว หรือ หารร่วมมาก (ห.ร.ม) นั่นเอง

ตัวหารร่วมมาก(Greateast Common Divisor)

คือการหาตัวหารร่วมมากของจำนวน 2 จำนวน เช่น gcd(15,10) = 5 ; ถ้า p เป็นจำนวนเฉพาะ(prime) และถ้า q<p , gcd(p,q) = 1

<u>หมายเหตุ</u> gcd(a,b) = gcd(b,a)

ยูคิลดีน อัลกอริทึม(Euclidean algorithm)

เป็นโพซีเดอร์(procedure) สำหรับคำนวณหาค่าหารร่วมมาก(gcd) ของ 2 จำนวน ถ้าต้องการหาค่าหารร่วมมาก(gcd) ของ a และ b ซึ่ง a>b จะได้ว่า

a = m*b+r : r มีค่า 0 <= r < b (ซึ่ง m = a/b เหลือเศษ r)

ซึ่งเราสามารถใช้สูตรนี้คำนวณหาค่า gcd ได้ โดยจะนำกลับมาวนใช้เช่นเดิม เราจะทำไปเรื่อย ๆ จนกระทั่ง เศษ r = 0

<u>ตัวอย่าง</u>

การหาค่า gcd(3615807,2763323)

3615807 = (1) * 2763323 + 852484

2763323 = (3) * 852484 + 205871

852484 = (4) * 205871 + 29000

205871 = (7) * 29000 + 2871

29000 = (10) * 2871 + 290

2871 = (9) * 290 + 261

290 = (1) * 261 + 29

261 = (9) * 29 + 0

เราจะได้ค่า gcd(3615807,2763323) = 29

เศษจากการหาร(Modular Arithmetic)

เช่น 11 mod 3 = 2

ถ้า a mod n = b แล้ว a = c*n+b ; c คือ ผลหาร

2 จำนวนที่มีค่า modulus เหมือนกัน เช่น 11 mod 3 = 2 และ 5 mod 3 = 2 จำนวนทั้ง 2 จะ equivalent กันภายใต้ modulus n ถ้า ผลลัพธ์ mod n ของทั้ง 2 จำนวนเท่ากัน

$X \equiv nY$ ก็ต่อเมื่อ (X mod n) = (Y mod n)

Alternately

$X \equiv nY$ ก็ต่อเมื่อ (x-y) = k*n สำหรับ k และตกลงร่วมกันว่า

a+b mod n = ((a+b) mod n)

<u>คุณสมบัติทางคณิตศาสตร์ของการมอด(Properties of Modular Arithmetic)</u>

associativity (การจัดหมู่) a+(b+c) mod n = (a+b)+c mod n

a*(b*c) mod n = (a*b)*c mod n

commutativity (การสลับที่) a+b mod n = b+a mod n

a*b mod n = b*a mod n

distributivity (การกระจาย) a*(b+c) mod n = ((a*b)+(a*c)) mod n

identities (เอกลักษณ์) a+0 mod n = 0+a mod n = a

a*1 mod n = 1*a mod n = a

inverses (การกลับ ม a+(-a) m d n = 0

$a*(a^{-1})$ mod n = 1 ถ้า $a \neq 0$

ลดรูป(reducibility) (a+b) mod n = ((a mod n)+(b mod n)) mod n

(a*b) mod n = ((a mod n)*(b mod n)) mod n

<u>การตรวจสอบอินเวอร์(Computing inverse)</u>

อินเวอร์สการคูณของ a คือ a^{-1} หรือ 1/a เพราะว่า a*1/a = 1

ถ้า a มี อินเวอร์สการคูณเป็น b จะได้ว่า a*b = 1

จากตารางของผลคูณมอด(mod) 5 จะพบว่า

อินเวอร์สของ 1 คือ 1

อินเวอร์สของ 2 คือ 3

อินเวอร์สของ 3 คือ 2

อินเวอร์สของ 4 คือ 1

ตารางที่ 8 แสดงผลการคูณแล้วมอด(mod) 5

*	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

ค่าอินเวอร์สไม่ได้มาจากสมการ แต่มาจากการตรวจเช็คค่าในตารางที่ผลคูณ mod 5 = 1 ในแต่ละแนวตั้ง(column)

<u>ทฤษฎีของเฟอร์แมท(Fermat's Theorem)</u>

ค่าจำนวนเฉพาะ(Prime) เป็นค่าจำนวนเฉพาะ p = Prime number , a = any number ที่ a < p

$$a^{p} \bmod p = a$$

หรือ

$$a^{p-1} \bmod p = 1 \qquad (1)$$

ถ้าอินเวอร์สของ a คือ x จะได้ว่า

$$ax \bmod p = 1 \qquad (2)$$

เมื่อทำการรวมสมการ (1)และ(2) จะได้

$$ax \bmod p = 1 = a^{p-1} \bmod p \qquad (3)$$

ดังนั้น $x = a^{p-2} \bmod p$

สมการ (3) ใช้ได้ในกรณี a < p เท่านั้น

<u>ตัวอย่างที่ 1</u> จงหาอินเวอร์สของ 3 mod 5

$$3^{-1} \bmod 5 = 3^{5-2} \bmod 5$$
$$= 3^{3} \bmod 5$$
$$= 27 \bmod 5$$
$$= 2$$

ค่าที่ได้จะตรงกับค่าในตารางผลคูณ mod 5

<u>ตัวอย่างที่ 2</u> จงหาอินเวอร์สของ 4 mod 5

$4^{-1} \bmod 5 = 4^{5-2} \bmod 5$

$= 4^{3} \bmod 5$

$= 64 \bmod 5$

$= 4$

ค่าที่ได้ตรงกับค่าในตารางผลคูณ mod 5

2.2.2 ระบบการเข้ารหัสแบบกุญแจที่เปิดเผย(Public-Key System)

การเข้ารหัสแบบเปิดเผยกุญแจ(public key),ผู้ใช้(user)แต่ละคนจะมีคีย์ ซึ่งไม่มีการเก็บเป็นความลับ โดยทั่วไปกุญแจที่เปิดเผย(public key)จะไม่การเก็บเป็นความลับ กุญแจที่เปิดเผย(public key) จะถูกเปลี่ยนไปในวิธีการเข้ารหัส แต่คีย์ที่เก็บไว้เป็นความลับ(private key) ที่ใช้ถอดรหัส

<u>การส่งข้อมูลMotivation</u>

ในการส่งข้อมูลระหว่างผู้ใช้(user) จะต้องใช้กุญแจ(Key) ของผู้ส่งและผู้รับเข้ามาร่วมด้วย เพื่อป้องกันผู้ที่ไม่เกี่ยวข้องมาใช้ข้อมูลนี้

ช่องทาง(Channel) เป็นทางที่ข้อมูลใช้ในการสื่อสารระหว่างคู่ ผู้รับและผู้ส่ง ผู้อื่นที่ไม่เกี่ยวข้องจะไม่สามารถใช้เส้นทางนี้ได้ เช่น ถ้ามีผู้ใช้(user) A,B และ C ต้องทำการ ตั้งช่องทางการสื่อสาร(Set communication channels) แต่ละคู่กรณี เช่น A ต้องการแลกเปลี่ยนข้อมูลกับ B โดยปราศจาก C เป็นต้น ในการจะติดต่อในกรณีนี้จะต้องใช้กุญแจ(Key) ที่เกี่ยวข้องคือกุญแจ(Key) AB,กุญแจ(Key) AC และกุญแจ(Key) BC

ถ้ามีการเพิ่มผู้ใช้(user) D จะต้องมีช่องทาง(Channel) ที่จะติดต่อไปยังผู้ใช้(user) A,B และ C และต้องเพิ่ม กุญแจ(Key) DA ,กุญแจ(Key) DB และกุญแจ(Key) DC ดังนั้นเมื่อมีผู้ใช้(user) ใหม่เพิ่มเข้ามาในระบบที่มีอยู่ n ผู้ใช้(user) จะต้องเพิ่มกุญแจ(Key) อีก n กุญแจ(Key)

ดังนั้นระบบที่มี n ผู้ใช้(users) จะมีค่ากุญแจ(Key) = $n*(n-1)/2$Keys ระบบนี้จะมีการเพิ่มขึ้นอย่างมาก ถ้ามีกุญแจ(user) เพิ่มเข้ามาทำให้ระบบมีความยุ่งยากในการบำรุงรักษาหรือบริหาร

<u>คุณสมบัติ(Characteristics)</u>

ในระบบเปิดเผยกุญแจ(public key) ,ผู้ใช้(user) แต่ละคนจะมี 2 กุญแจ(key) คือ กุญแจที่เปิดเผย (public key) และกุญแจที่ไม่เปิดเผย(private key),

กำหนดให้ k_{PRIV} = private key

k_{PUB} = public key

$$P = D(k_{PRIV}, E(k_{PUB}, P)) \quad (1)$$

สมการ(1) แสดงให้เห็นว่า ข้อความ P ถูกเข้ารหัสโดยใช้กุญแจที่เปิดเผย (Public key) ก่อน แล้วหลังจากนั้นถูกถอดรหัสโดยใช้กุญแจที่ไม่เปิดเผย(Private key) ก็จะได้ข้อความ P ตามเดิม

$$P = D(k_{PUB}, E(k_{PRIV}, P)) \quad (2)$$

สมการ(2) แสดงให้เห็นว่า ข้อความ P ถูกเข้ารหัสโดยใช้กุญแจที่ไม่เปิดเผย(Private key) ก่อน แล้วหลังจากนั้นถูกถอดรหัสโดยใช้กุญแจที่เปิดเผย(Public key) ทีหลังก็จะได้ข้อความ P ตามเดิมเช่นกัน

แนวคิดคือว่าฟังก์ชัน D จะมีอกรูเมน(argument) ใดๆก็ได้ นั่นคือเป็นไปได้ว่าถอดรหัสก่อนแล้วเข้ารหัสที่หลัง

<u>ตัวอย่าง</u> คน 3 คน ชื่อ B,C,D ทั้งหมดทุกคนสามารถส่งข้อความ(message) ไปให้ A ได้ โดยเข้ารหัสข้อความ(message) โดยใช้กุญแจที่เปิดเผย(public key) ของ A และถ้า B เข้ารหัสโดยใช้กุญแจที่เปิดเผย (public key) ของ A ,C จะถอดรหัสข้อความที่ B ส่งไปไม่ได้ เพราะว่า C ไม่รู้ว่ากุญแจที่ไม่เปิดเผย(private key) ของ A คืออะไร

2.2.3 มาร์เคล-เฮลแมน คแนพแซค(Merkel - Hellman Knapsacks)

แนวคิดคือทำการเข้ารหัสข้อความแบบไบนารี(Binary message) และลดจำนวนข้อมูลที่เข้ารหัส(ciphertext) ลงให้เป็นผลรวมโดยการรวมกันเฉพาะข้อความ(Plaintext) ที่เป็น 1 เท่านั้น นั่นคือชุด(Block) ของข้อความ(Plaintext) จะถูกเปลี่ยนเป็นผลรวมของคแนพแซค(Knapsack) โดยการรวมเทอม(Term) ที่ สมดุลกัน(match) กับบิท(bit) 1 เท่านั้น

Knapsack = เวกเตอร์ ของ Integer Term

$$\sum_i a_i * v_i = T$$

ซุปเปอร์รีนีสซิ่ง คแนพแซค(Superincreasing Knapsacks)

จำนวนเต็ม(Integer) ของซุปเปอร์รีนีสซิ่ง S ต้องอยู่ในรูปแบบซุปเปอร์รีนีสซิ่งซีเควน(Superincpeasing Sequenec)

จำนวนเต็ม(Integer) แต่ละตัวต้องมากกว่าผลบวกของตัวก่อนหน้าทุกตัว เช่น S = [1, 4, 11, 17, 38, 73,]

$$a_k > \sum_{j=1}^{k-1} a_j$$

ซุปเปอร์รีนีสซิ่ง คแนพแซค(Superincreasing Knapsack (simple knapsack))

เริ่มต้นที่ T เปรียบเทียบจำนวนเต็ม(integer) ที่มากที่สุดใน S ถ้าจำนวนเต็ม (integer) มากกว่า T มันจะไม่อยู่ในรูปผลรวม(sum) ดังนั้น ตำแหน่งใน C คือ 0 ถ้าจำนวนเต็ม(integer) ที่มากที่สุด น้อยกว่าหรือเท่ากับ T แล้วจำนวนเต็ม(integer) อยู่ในผลรวม(SUM) ให้ตำแหน่งใน C เป็น 1 และลดค่า T โดยจำนวนเต็ม(integer) ทำซ้ำกับจำนวนเต็ม(integer) ที่เหลือทั้งหมด ใน S ดังรูป

96:	73? Yes	95:	73? Yes <--------
96-73=23:	38? No	95-73=22:	38? No	
23:	17? Yes	22:	17? Yes <--------
23-17=6:	11? No	22-17=5:	11? No	
6:	4? Yes	5:	4? Yes <--------
6-4=2:	1? Yes	5-4=1:	1? Yes <--------
1:	-	1-1=0	
	No Solution		Solution

รูปที่ 6 แสดงการแก้ปัญหาซิมเบิ้ล คแนพแซค(Simple Knapsack)

เทคนิคการเข้ารหัส(THE ENCRYTION TECHNIQUE)

public key = เซตของ integer ของ knapsack (ไม่ใช่ Supserincreasing Knapsack)

private key = superincreasing knapsack

หลักการมอดทางคณิตศาสตร์(PRINCIPLES OF MODULAR ARITHMETIC)

การมอดทางคณิตศาสตร์(Modular Arithmetic) ที่มี แฟคเตอร์ร่วมกัน(Common factors) ถ้า ผลงาน(product) ทุกตัวของจำนวนเต็ม(integer) ทุกตัวถูกแมบ(mapped) ไปที่จำนวนเต็มมอดด้วยเอ็น(integer mod n.) ชัดเจนเลยว่าจะมีผลของงาน(Product) บางตัวเหมือนกัน นั้นคือ 2 ผลงาน(product) ที่ต่างกัน สามารถที่จะสร้างงาน(produce) แล้วทำให้เกิดผลที่เหมือนกันได้

ถ้า $w*x \bmod n = r$ แล้ว $w*x + n \bmod n = r$

$w*x + 2n \bmod n = r$

ถ้า w และ n มีแฟคเตอร์(factor) ร่วมกัน แล้ว ผลลัพธ์จะไม่ใช่จำนวนเต็ม(integer) ทุกตัวระหว่าง 0 และ n-1 จะเป็นผลลัพธ์ของ

$w*r \bmod n$ สำหรับ x บางตัว

<u>พิจารณา</u> จำนวนเต็ม(integer) mod 5

ถ้า w = 3 และ x= 1,2,3,..... ผลคูณของ $x*w \bmod 5$ ผลลัพธ์ทั้งหมดจะเป็น 0-4 ดังตาราง 9 สังเกตุว่า หลังจาก x=5 modular คำตอบจะซ้ำ

ตารางที่ 9 แสดงการ $3 * x \bmod 5$

x	3*x	3*x mod 5
1	3	3
2	6	1
3	9	4
4	12	2
5	15	0
6	18	3
7	21	1

อย่างไรก็ตาม ถ้าเราเลือก w=3 และ n=6 จำนวนเต็ม(integer) ไม่ทุกตัวที่อยู่ระหว่าง0-5ที่จะถูกใช้ เพราะว่า w และ n ใช้แฟคเตอร์ร่วมกัน(share common factor) 3 ร่วมกับ ในตาราง 10 แสดงผลลัพธ์ของ 3*x mod 6

ตารางที่ 10 แสดงผลลัพธ์ของ 3*x mod 6

x	3*x	3*x mod 6
1	3	3
2	6	0
3	9	3
4	12	0
5	15	3
6	18	0
7	21	3

จะมีบางค่าที่ไม่ได้เขียนว่าเป็นผลคูณของทั้ง 2 จำนวนเต็ม(integer) mod n สำหรับค่าของ n ที่แน่นอน ค่าของผลลัพธ์ ทั้งหมดอยู่ระหว่าง 0 และ n-1 , n ต้องเป็นค่าจำนวนเฉพาะ(Prime) ที่สัมพันธ์กับ w

ถ้า w และ n สัมพันธ์กับค่าจำนวนเฉพาะ(Prime) w มีผลคูณย้อนกลับ mod n อันนี้หมายความว่าจำนวนเต็ม(integer) w ทุกตัว มี จำนวนเต็ม(integer) อีกตัวหนึ่ง w^{-1} เพราะฉะนั้น $w * w^{-1} = 1 \bmod n$

อินเวอร์สผลคูณ คือ วิธีการเปลี่ยนกลับของผลคูณ

$$(w*q) * w^{-1} = q$$

(การคูณมีคุณสมบัติ การสลับที่และการจัดหมู่ ในกลุ่ม mod n ดังนั้น

$$w*q* w^{-1} = (w* w^{-1})*q = q \bmod n)$$

ผลลัพธ์(Result) เหล่านี้จากการมอดทางคณิตศาสตร์(modular arithmatic) ,ดิฟฟี่และเฮแมน(Diffie and Hellman) พบวิธีการหยุด(Break) ของการซุปเปอร์รีครีสซิ่ง (Superincreasing) ลำดับของจำนวนเต็ม(integer),ข้อความ(Pattern) สามารถถูกทำให้

แตกสลายได้โดย การคูณจำนวนเต็ม(integer) ทุกตัวโดยค่าคงที่ w ,และหาผลลัพธ์ mod n ขณะที่ w และ n เป็นค่าจำนวนเฉพาะ(Prime)

<u>การเข้ารหัสโดยใช้ ซุปเปอร์รีนครีสซิ่ง คแนพแซค(Transforming a Superincreasing Knapsack)</u>

เพื่อจะทำการเข้ารหัสโดยใช้อัลกอริทึม มาร์เคิล เฮลแมน(Merkle-Hellman algorithm) เราต้องการซุปเปอร์รีนครีสซิ่ง คแนพแซค(Superincreasing Knapsack) ซึ่งสามารถถูกเปลี่ยนเป็นฮาร์ด คแนพแซค(Hard Knapsack)

เริ่มต้นที่การหยิบซุปเปอร์รีนครีสซิ่ง ซีเควน(Superincreasing sequence) S ของ m จำนวนเต็ม(integer)

เริ่มแรกเลือกตัวเลขจำนวนเต็มที่มีค่าน้อยที่สุด

เลือกจำนวนเต็มตัวต่อไปที่มากกว่าตัวแรก แล้วดำเนิน

เลือกจำนวนเต็มที่มากกว่า ผลรวมของ 2 ตัวแรก ยังคงต่อไปเรื่อยๆในงาน (Process)นี้โดยการเลือกจำนวน ตัวใหม่ที่มากกว่า ผลรวมของจำนวนเต็มทั้งหมดที่ถูกเลือกแล้ว

<u>ตัวอย่าง</u>

[1,

[1, ? > 1 : 2

[1,2, ? > 1+2 : 4

[1,2,4 ? > 1+2+4: 9

[1,2,4,9 ? > 1+2+4+9: 19

ซุปเปอร์รีนครีสซิ่ง ซีเควน(Superincreasing Sequence)ที่เลือกแล้ว เรียกว่า ซิมเบิ้ล คแนพแซค(Simple knapsack) ในตัวอย่างของคแนพแซค(knapsack problem , knapsack) มีคำตอบที่จะแก้ปัญหาได้หลังจากเลือกซิมเบิ้ล คแนพแซค(Simple knapsack) S=[s1,s2,s3,...,sn],เราเลือก ผลคูณของ w และ a มอด(modulas) n

ค่าการมอด(modulas) ควรจะเป็นจำนวนที่มากกว่าจำนวนเต็ม(integer) ที่ค่ามากที่สุด (Sm) การคูณควรจะไม่มีปัจจัยร่วมกับการมอด(Modulus) วิธีการหนึ่งที่ประกันได้

ว่าอันนี้คือการเลือกการมอด(modulus) ซึ่งเป็นค่าจำนวนเฉพาะ(Prime) ดังนั้น จึงไม่มีจำนวนที่น้อยกว่าการมอด(modulus) สุดท้าย เราแทนจำนสนเต็ม(integer) s_i ทุกตัว ในซิมเบิ้ล คแนพแซค(simple knapsack)

$$h_i = w^* s_i \bmod n$$

$H = [h_1, h_2, \ldots, h_m]$ คือ ฮาร์ด คแนพแซค(hard knapsack)

เราจะใช้ฮาร์ด(hard) และซิมเบิ้ล คแนพแซค(simple knapsack) ในการเข้ารหัส

<u>ตัวอย่าง ซุปเปอร์รีนครีสซิ่ง คแนพแซค</u>(Superincreasing knapsack) S=[1,2,3,4,9] และการเปลี่ยน S โดยการคูณด้วย w และ การลด รูปโดยมอด(mod) n ซึ่ง w=15,n=17

$1*15 = 15 \bmod 17 = 15$

$2*15 = 30 \bmod 17 = 13$

$4*15 = 60 \bmod 17 = 9$

$9*15 = 135 \bmod 17 = 16$

ฮาร์ด คแนพแซค(hard knapsack) H=[15,13,9,16]

<u>ตัวอย่าง การใช้ มาร์เคิล,เฮลแมน คแนพแซค(Merkle-Hellman Knapsack)</u>

มาร์เคิล,เฮลแมน(Merkle - Hellman) เริ่มต้นที่ข้อมูลของเลขฐานสอง(binary message) ,ข้อความ(message) คือเลขฐานสองเรียงกัน(binary sequence)จริงๆ P= $[p_1, p_2, \ldots, p_k]$ แบ่ง ข้อความ(Massage) ออกเป็นชุด(block) ๆ ละ m บิท(bit)

$P_0 = [p_1, p_2, \ldots, p_m]$ $P_1 = [p_{m+1}, p_{m+2}, \ldots, p_{2m}]$ และจนถึงอันดับที่ 4

ค่าของ m คือ จำนวนเทอม (term) ในซิมเบิ้ล คแนพแซค(Simple Knapsack) หรือฮาร์ด คแนพแซค(Hard knapsack) การเข้ารหัสของข้อความ(message)

P คือ ลำดับเป้าหมาย(target) ในแต่ละ เป้าหมาย(target) คือ ผลรวมของเทอม(term) ของฮาร์ด คแนพแซค(hard knapsack) H เทอม(term) ที่ถูกเลือกจะเป็น 1 ใน Pi Pi จะให้เวคเตอร์(vecter) เลือกสมาชิก(element)ของ H แต่ละเทอม(term) ของข้อมูลที่เข้ารหัส (ciphertext) คือ Pi*H เป็นหมาย(target) มาจากชุด(block) Pi เป็นการเลือกเวคเตอร์ (Selection vecter)

สมมติให้ S = [1,2,4,9] และ H = [15,13,9.16]w = 15, n = 17 และ m = 4

ข้อความ(Plain text) = 0 1 0 0 1 0 1 1 1 0 1 0 0 1 0 1

เข้ารหัสด้วยคแนพแซค(Knapsack) H = [15,13,9,16]

แบ่งข้อความ(Plaintext) เป็นชุด(block) เท่ากับ จำนวน m

P = 0 1 0 0 1 0 1 1 1 0 1 0 0 1 0 1

[0,1,0,0] * [15,13,9,16] = 13

[1,0,1,1] * [15,13,9,16] = 40

[1,0,1,0] * [15,13,9,16] = 24

[0,1,0,1] * [15,13,9,16] = 29

ดังนั้นข้อความ(message) ที่เข้ารหัสแล้ว = 13,40,24,29 โดยใช้ฮาร์ด คแนพแซค(Hard Knapsack)

อัลกอริทึมการถอดรหัส(Decryption Algorithm)

ผู้รับจะต้องทราบซิมเบิ้ล คแนพแซค(Simple Knapsack) และค่า w และ n เพื่อจะหาฮาร์ด คแนพแซค(Hard Knapsack) ซึ่ง

$$w * 1/w = 1 \bmod n$$

<u>ตัวอย่าง</u> $15^{(-1)} \bmod 17 = 8$

พิสูจน์ $15 * 8 \bmod 17 = 120 \bmod 17$

$= (17*17)+1$

$= 1$

จากที่แล้วมา

$$C = H * P = w * S * P \bmod n$$

ในการถอดรหัสข้อมูล Ciphertext จำเป็นต้องใช้ $w^{(-1)}$

$w^{(-1)} * C = w^{(-1)} * H * P = w^{(-1)} * w * S * P = S * P \bmod n$

<u>ตัวอย่าง</u> จากตัวอย่างการเข้ารหัสที่ผ่านมา S = [1,2,4,9] w = 5 และ n = 17 หลังจากเข้ารหัสจะได้ 13,40,24,28

การถอดรหัส จะใช้อินเวอร์ส(inverse) คือ $15^{-1} \bmod 17 = 8$ มา คูณกับ ข้อความที่เข้ารัหส(Ciphertext) จะได้

$13 * 8 = 104 \bmod 17 = 2 = [0100]$

$40 * 8 = 320 \bmod 17 = 14 = [1011]$

$24 * 8 = 192 \bmod 17 = 5 = [1010]$

$29 * 8 = 232 \bmod 17 = 11 = [0101]$

ก็จะได้ข้อความ(Plaintext) คืนมา

P = 0100 1011 1010 0101

การเข้ารหัสแบบอาร์เอสเอ(RSA Encryption)

ทำงานกับการมอด(Arithmetic mod n) วิธีการนี้ ข้อความ(Plaintext Block) จะถูกเปลี่ยนมาเป็นข้อมูลที่เข้ารหัส(Unsigned Integer) มี 2 กุญแจ(Keys) คือ d และ e ซึ่งใช้ในการถอดรหัสและเข้ารหัส

ข้อความ(Plaintext Block) P จะถูกเข้ารหัสเป็น $P^e \bmod n$ ดังนั้นการยกกำลังแล้วมอด(Exponentiation จะถูกกระทำโดย mod n) มันเป็นการยากมากที่หาแฟคเตอร์(Factor) P^e มาเข้ารหัสข้อความ(Encrypted Plaintext) เพื่อไม่ให้ผู้อื่นถอดรหัสได้

กุญแจ(Key) ถอดรหัส d จะถูกเลือกอย่างระมัดระวังคือ $(P^e)^d \bmod n = P$ ผู้รับที่แท้จริงจะทราบค่า d และจะสามารถถอดรหัสได้

รายละเอียดของอัลกอริทึมการเข้ารหัส

(Detailed Description of the Encrytion Algorithm)

อาร์เอสเอ(RSA) มี 2 กุญแจ(Keys) คือ d และ e ซึ่งทำงานเป็นคู่ของการเข้ารหัสและถอดรหัส ข้อความ(Plaintext Massage) P จะถูกเข้ารหัสเป็น(Ciphertext) C โดย

$$C = P^e \bmod n$$

ข้อความ(Plaintext) ถูกเรียกกลับคืนมาโดย

$$\begin{aligned} P &= C^d \bmod n \\ &= (P^e)^d \bmod n \\ &= (P^d)^e \bmod n \end{aligned}$$

<u>การเลือกกุญแจ(Choosing Keys)</u>

กุญแจการเข้ารหัส(Encryption Key) ประกอบด้วยคู่ลำดับของจำนวนเต็ม (e,n) และกุญแจการถอดรหัส(Decryption Key) ประกอบด้วยคู่ลำดับของจำนวนเต็ม (d,n) การเลือกค่าของ n ควรเป็นค่าที่มากที่สุด ซึ่งเป็นผลงาน(Product) ของ 2 จำนวนเฉพาะ (Prime) p และ q

<u>หมายเหตุ</u> ; จำนวนเฉพาะ(Primes number) = จำนวนเต็มบวกที่ถูกหารด้วยตัวมันเองและ 1 แล้วทำให้ได้เศษเท่ากับ 0

โดยทั่วไป p และ q มีค่าประมาณ 100 หลัก ดังนั้น n จะมีค่าประมาณ 200 หลัก การเลือกค่า e นั้น ค่า e จะต้องมีความสัมพันธ์จำนวนเฉพาะ(relatively prime) กับ (p-1) *(q-1) คือ จะต้องไม่มีแฟคเตอร์ร่วมกัน(common factor) วิธีง่ายในการยืนยัน

(Guarantee) ว่า e คือมีความสัมพันธ์จำนวนเฉพาะ(Relatively Prime)กับ (p-1)*(q-1) คือให้เลือก e ที่เป็นจำนวนเฉพาะ(Prime) ซึ่งต้องมากกว่า (p-1) และ (q-1) ในที่สุด ก็จะเลือก d ได้ว่า

$$e*d \equiv 1 \mod (p-1) * (q-1)$$

การคำนวณทางคณิตศาสตร์ของอัลกอริทึมอาร์เอสเอ

(Mathematical Foundations of the RSA Algorithm)

$\varphi(n)$ = จำนวนเต็มบวกที่น้อยกว่า n และมีความสัมพันธ์จำนวนเฉพาะ (Relatively Prime) กับ n

ถ้า p คือจำนวนเฉพาะ(Prime) แล้ว

$$\varphi(p) = (p-1)$$

มากไปกว่านั้น ถ้า $n = p*q$ ขณะที่ p และ q เป็น จำนวนเฉพาะ(Prime) ทั้งคู่

$$\varphi(n) = \varphi(p)*\varphi(q) = (p-1) * (q-1)$$

แสดงในรูปแบบของยูเลอร์(Euler) และเฟอร์แมท(Fermat)

$$x^{\varphi(n)} \equiv 1 \mod n$$

โดยที่ x = จำนวนเต็มใด ๆ และ n กับ x มีความสัมพันธ์จำนวนเฉพาะ (Relatively Prime)

สมมุติว่าเราต้องการเข้ารหัส ข้อความ(Pliantext) ข้อมูล P โดยวิธีของอาร์เอสเอ (RSA Algorithm) ดังนั้น $E(P) = P^e$ เราจะต้องแน่ใจว่าเราสามารถที่จะควบคุมข้อมูลได้

ค่า e,d เป็นอินเวอร์ท(inverses) mod $\varphi(n)$

$$e * d \equiv 1 \bmod \varphi(n)$$

หรือ

$$e * d \equiv k * \varphi(n) + 1 \qquad ; k = \text{จำนวนเต็ม} \qquad \text{---------- (0)}$$

จาก ยูเลอร์(Euler) และเฟอร์แมท(Fermat) ผลลัพธ์จะได้

$$P^{p-1} \equiv 1 \bmod p$$

และ (p-1) เป็นแฟคเตอร์(Factor) ของ $\varphi(n)$

$$P^{k*\varphi(n)} \equiv 1 \bmod p$$

คูณด้วย P

$$P^{k*\varphi(n)+1} \equiv P \bmod p$$

ถ้าอยู่ในรูปของ q จะได้

$$P^{k*\varphi(n)+1} \equiv P \bmod q$$

นำผลลัพธ์ทั้งสองเข้ากับสมการ (0) จะได้

$$(P^e)^d = P^{e*d}$$
$$= P^{k*(\varphi(n)+1}$$
$$\equiv P \bmod p$$
$$\equiv P \bmod q$$

ดังนั้นจะได้

$$(P^e)^d \equiv P \bmod n$$

และ $a * b \bmod n = (a \bmod n) * (b \bmod n) \bmod n$

<u>ตัวอย่าง</u>

$p = 11, q = 13$

$n = p * q = 143$

$\varphi(n) = (p-1) * (q-1) = 10 * 12 = 120$

เลือก e ที่ต้องการ และ e ต้องเป็นความสัมพันธ์จำนวนเฉพาะ(Relatively Prime) กับ (p-1) * (q-1) เลือก

$e = 11$

อินเวอร์ส(Inverse) ของ 11 mod 120 คือ 11

$11 * 11 = 121 = 1 \bmod 120$ ซึ่งทั้งกุญแจ(key) เข้ารหัสและถอดรหัสจะเหมือนกัน คือ

$$e = d = 11$$

ให้ P = ข้อความ(massage) ที่ถูกเข้ารหัส

ถ้า $P = 7$

$E(7) = 7^{11} \bmod 143 = 106$

$D(106) = 106^{11} \bmod 143 = 7$

<u>การนำอัลกอริทึมมาใช้งาน</u>(Practical Implementation of the Algorithm)

e เป็น ค่าที่ถูกเลือก ซึ่งเป็นความสัมพันธ์จำนวนเฉพาะ(Relatively Prime to) (p-1)*(q-1)

e มักจะเป็น จำนวนเฉพาะ(Prime) ที่มากกว่า (P-1) หรือ (q-1)

d คือ อินเวอร์ส(inverses) ของ e mod $\varphi(n)$

ผู้ใช้(User) ได้แบ่ง e และ n และเก็บ d เป็นความลับ p,q และ $\varphi(n)$ จะถูกตัดทิ้งไปได้ในส่วนนี้ สังเกตว่าถ้ารู้ว่า n เป็นผลงาน(Product)ของ 2 จำนวนเฉพาะ(Primes) และถ้า n มีค่ามาก เช่น 100 หลัก(digit) มันจะเป็นไปไม่ได้ที่จะหาจำนวนเฉพาะ(Prime) p และ q หรือกุญแจที่เป็นความลับ(Private Key) d จาก e ดังนั้นรูปแบบนี้จึงมีความปลอดภัยให้สำหรับ d

รูปแบบนี้ยังไม่เคยทำการตรวจสอบความถูกต้องของ p และ q เว้นแต่ว่าพวกมันเป็นจำนวนเฉพาะ(Prime) ตั้งแต่นั้นจะกำหนดการพิจารณาบนลำดับของ 10 ยกกำลัง 50 แฟคเตอร์(Factor) ที่เป็นไปได้

ทุก ๆ จำนวนเฉพาะ(Prime Number) จะผ่าน 2 การทดสอบ ถ้า p เป็นจำนวนเฉพาะ (Prime) และ r คือจำนวนที่น้อยกว่า P

$$gcd(p,r) = 1$$

และ

$$J(r,p) \equiv r^{(p-1)/2} \mod p$$

ซึ่ง J(r,p) เป็นแจคโคบิฟังก์(Jacobi Function)กำหนดดังนี้

$$J(r,p) = \begin{cases} 1 & ; \text{ถ้า } r = 1 \\ J(r/2.p) * (-1)^{(p2-1)/8} & ; \text{ถ้า } r \text{ เป็นจำนวนคู่} \\ J(P \bmod r.r) * (-1)^{(r-1)*(p-1)/4} & ; \text{ถ้า } r \text{ เป็นจำนวนคี่,} \neq 1 \end{cases}$$

ถ้าจำนวนที่ถูกสงสัยว่าเป็นจำนวนเฉพาะ(Prime) หรือไม่ให้ใช้วิธีการนี้ตรวจสอบได้ ถ้าจำนวนที่ผ่านการตรวจสอบจะมีความเป็นไปได้ว่าจำนวนเป็นจำนวนเฉพาะ(Prime) อย่างน้อยที่สุด 1/2 ปัญหาที่เกี่ยวกับอาร์เอสเอ(RSA Algorithm) คือการหา 2 จำนวนเฉพาะ(Prime) ที่มีค่ามากที่สุดของ p และ q พยายามเดาค่าจำนวนเฉพาะ(Prime) ที่มาก ๆ ของ p แล้วสุ่มจำนวน r และคำนวณ gcd(p,r) และ J(r,p) ถ้าผลการตรวจสอบผิดพลาด (Fail)จะสรุปว่า p ไม่ใช่ จำนวนเฉพาะ(Prime) และโพซีเดอร์(Procedure) จะหยุดการทำงาน แต่ถ้าทั้งคู่ผ่านเป็นไปได้ว่า p ไม่ใช่จำนวนเฉพาะ(Prime) คือมากที่สุด 1/2 โพซีเดอร์(Procedure) จะทำซ้ำด้วยค่าใหม่ จากนั้น r จะถูกสุ่ม(Random) ขึ้นมา ถ้า r ผ่าน เป็นไปได้ว่าไม่ใช่จำนวนเฉพาะ(non-Prime) p สามารถผ่านการตรวจสอบ(Test) ที่ 1/4โดยทั่วไป หลังจาก ทำ งาน(Process) k ครั้ง โดยไม่ผิดพลาด(Fail) เป็นไปได้ว่า p ไม่เป็นจำนวนเฉพาะ(Prime) ที่ $(1/2)^k$

2.2.4 ระบบที่ใช้กุญแจเดียว(Single Key (Conventional) Systems)

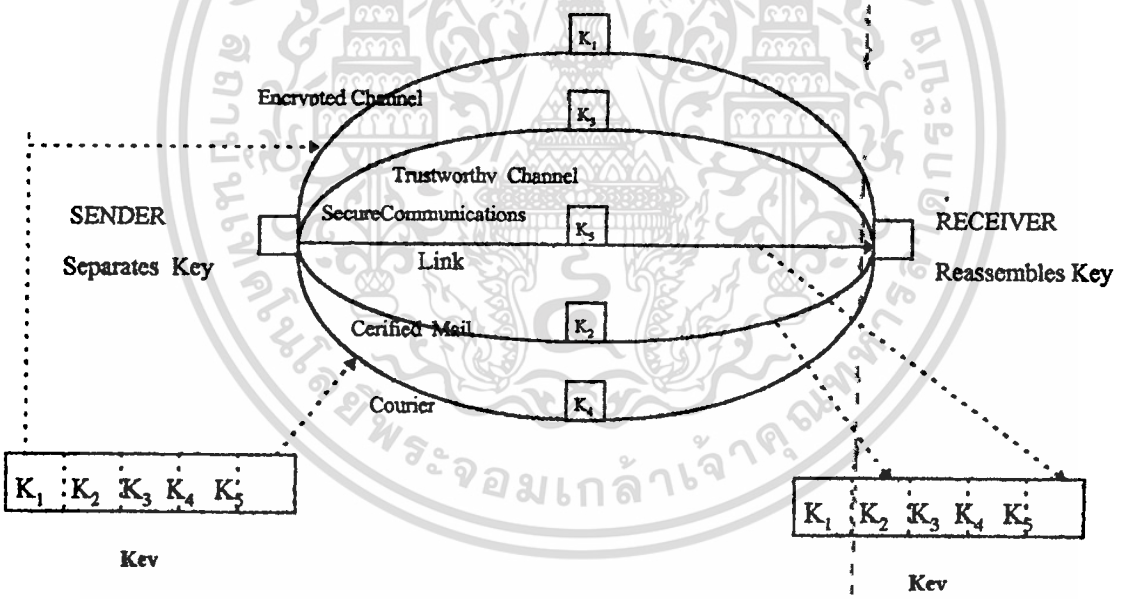
เมอร์เคิล,เฮลแมน(Merkle-Hellman) และอาร์เอสเอ(RSA Algorithm) ทั้งคู่เป็นระบบที่เปิดเผยกุญแจ(Public Key Algorithm) ซึ่งได้มีการแบ่งผู้ใช้(User) ออกโดยใช้กุญแจ(Key) 2 ตัวคือการเข้ารหัส(Encryption) ตัวหนึ่งและอีกตัวหนึ่งเป็นการถอดรหัส (Decryption) ดังนั้น 1 กุญแจ(Key) สามารถที่จะทำประโยชน์ต่อผู้ที่ต้องการใช้การส่งการข้อมูลการเข้ารหัส(Encryption Information) และอีกตัวหนึ่งจะถูกเก็บเป็นความลับคือกุญแจ(Key) เดียวเท่านั้นที่สามารถถอดรหัสข้อมูล(Information) นั้นได้

ระบบที่ใช้กุญแจตัวเดียว(Single Key หรือ Conventional Encryption Algorithm บางครั้งเรียกว่า Private Key System)ซึ่งการเข้ารหัสและถอดรหัสใช้จำนวนเดียว(Key) เดียวกัน ซึ่งต้องเก็บเป็นความลับ

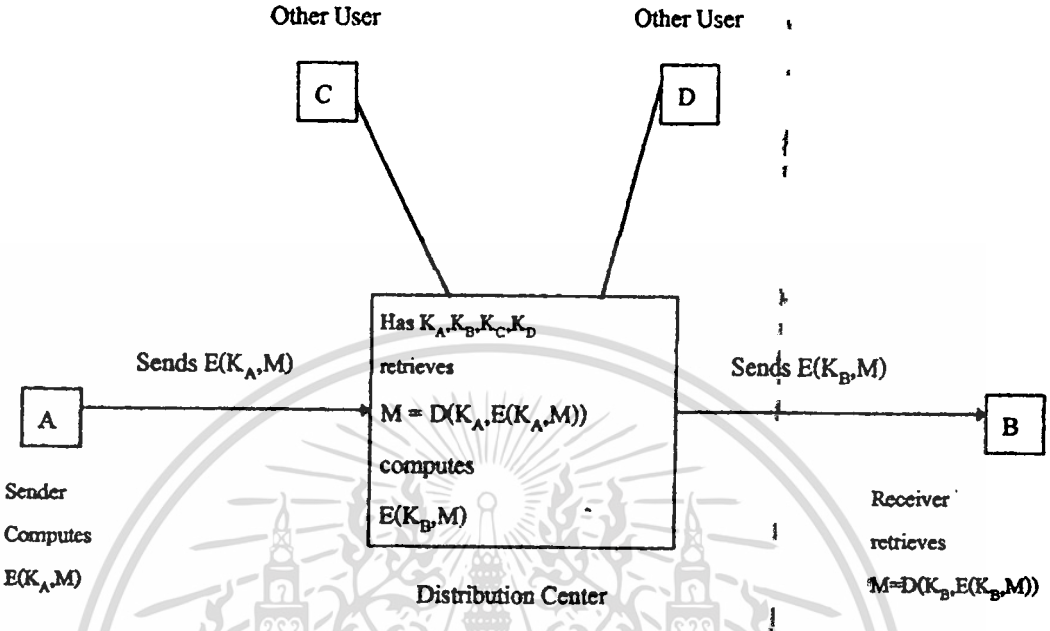
<u>ข้อดีและข้อเสีย</u>

ระบบใช้กุญแจตัวเดียว(Single Key System) แบ่งออกเป็น 2 ช่องทาง(Channel) ไปที่ใช้(User); A และ Bใช้กุญแจตัวเดียวกัน(Share Secret Key) และ A,B จะเข้ารหัสข้อมูลเพื่อที่จะส่งไปที่อื่น ก็เหมือนกับการถอดรหัสข้อมูลจากที่อื่น

กุญแจ(Key) ที่เป็นความลับ ระบบจะจัดการจัดการ(Authentication) เพื่อพิสูจน์ว่าข้อความ(massages) ที่รับเข้ามาเป็นข้อมูลที่ถูกต้องหรือไม่ มีการปลอมแปลงหรือแก้ไขหรือเปล่า มาจากผู้ที่ส่งมาจริงหรือไม่ เป็นต้น



รูปที่ 7 แสดงการกระจายของกุญแจ



รูปที่ 8 แสดงศูนย์กลางการกระจายของระบบ(Distribution Center)

<u>ปัญหาของระบบที่ใช้กุญแจตัวเดียว(Single Key Systems)</u>

1.กุญแจของระบบ(Key System) ทั้งหมด ถ้ากุญแจ(Key) ถูกเปิดเผย(ขโมยมา,เดา, หรือด้วยวิธีใด ๆ ก็ตาม)จากผู้ไม่กีดกัน(Interceptors) จะถอดรหัสข้อมูลได้ มากไปกว่านั้นไม่มีการประกาศการใช้ กุญแจที่ถูกขโมย(Intercepted Key) ที่ทำข้อความ(Message) ปลอม ภายใต้วิธีการของผู้ส่งจริง สำหรับเหตุผลชั้นนี้ในความมั่นใจการระบบการเข้ารหัส (Encryption System) กุญแจ(Key) จะถูกเปลี่ยนบ่อย ๆ ดังนั้นจะต้องจำกัดปริมาณข้อมูล(Compromisted Key)

2. การกระจายของกุญแจ(Key) เปลี่ยนเป็นปัญหา กุญแจ(Key) ต้องถูกส่งด้วยความปลอดภัยมากที่สุด และอนุญาติให้เข้าถึง(Access) ข้อมูลที่เข้ารหัสได้เฉพาะกุญแจ(Key) เท่านั้น

3. จำนวนของกุญแจ(Key) ได้เพิ่มขึ้นเป็น กำลัง 2 ของจำนวนผู้ใช้ที่แลกเปลี่ยนข้อมูลลับปัญหานี้จะถูกบรรจุโดยผู้ใช้ 2-3 คน ที่แลกเปลี่ยนความลับกันตรง ๆ บน การ

สื่อสารข้อมูล(Network of Interchange) ถ้า ผู้ใช้(People) แบ่งเน็ตเวิร์ค(Network) ที่ต้องการแลกเปลี่ยนความลับ(Secret) พวกเขามีศูนย์(Clearing House หรือ Forwarding Office) ซึ่งยอมรับความลับ(Secret) จากบุคคลและ ถอดรหัส เข้ารหัสไปใหม่ โดยใช้กุญแจ(Key) ของอีกคนหนึ่งแล้วจึงจะส่งไป

4. ระบบการเข้ารหัสที่ใช้กุญแจตัวเดียว(Single Key Encryption) มีจุดอ่อนที่ถูกโจมตีอย่างมากในการเข้ารหัส ทั้ง มาร์เคิล,เฮลแมน(Merkle-Hellman) และระบบอาร์เอสเอ(RSA Algorithm) จะมีปัญหามาก ปัญหาที่ได้รับความสนใจโดยนักคณิตศาสตร์ คือ ความมั่นคงในระบบ กุญแจตัวเดียว(Single Key System) ต้องเป็นวิธีที่มั่นคงแน่นอน (Solid)

2.3 หลักการของไคล์เอนท์เซิร์ฟเวอร์

ไคล์เอนท์เซิร์ฟเวอร์เป็นโครงสร้างของระบบคอมพิวเตอร์รูปแบบหนึ่งที่ แบ่งแยกการประมวลผลข้อมูลออกเป็น 2 ระบบ โดยฝั่งไคล์เอนท์(ผู้ขอใช้บริการ) จะถูกเรียกว่าระบบฟรอนท์เอ็น(Front-end system)หรือส่วนคาต้าเบสแอพพลิเคชั่นทำงานอยู่ และฝั่งค้าต้าเบสเซิร์ฟเวอร์(ผู้ให้บริการ) จะถูกเรียกว่าระบบเบ็คเอ็น(Back-end system) หรือส่วนที่เป็นระบบการจัดการฐานข้อมูลจริง ๆ ทำงานอยู่ ซึ่งระบบฟรอนท์เอ็นนี้จะจัดการการประมวลเกี่ยวกับหน้าจอและอินพุทเอาท์พุทของผู้ใช้และระบบแบคเอ็นจะจัดการการประมวลผลข้อมูลและการทำเข้าถึงดิสค์ เช่นเมื่อผู้ใช้บนระบบฟรอนท์เอ็นสร้างคิวรี่ (query) เพื่อสอบถามข้อมูลจากค้าต้าเบสเซิร์ฟเวอร์ ส่วนฟรอนท์เอ็น (Front-end) แอฟพลิเคชั่นจะส่งการร้องขอให้เซิร์ฟเวอร์โดยผ่านระบบเครือข่าย ส่วนเซิร์ฟเวอร์ก็จะทำการค้นหาข้อมูลที่ผู้ใช้ต้องการแล้วส่งข้อมูลกลับไปให้

วัตถุประสงค์หลักของระบบไคล์เอนท์เซิร์ฟเวอร์คือการอนุญาติให้แอพพลิเคชั่นของผู้ใช้ปริการเข้ามาเรียกใช้ข้อมูลที่ถูกจัดการโดยผู้ให้บริการได้ โดยผู้ให้บริการสามาถรันอยู่ในเครื่องที่ตั้งในที่ห่างไกลกับเครื่องที่ผู้ขอให้บริการรันอยู่

โดยทั่วไปแล้วระบบไคลเอนท์จะถูกใช้ทำงานกับพีซีและส่วนคาต้าเบสเซิร์ฟเวอร์สามารถทำงานบนเครื่องใดก็ได้ตั้งแต่พีซีไปจนถึงเมนเฟรม

2.3.1 ชนิดของการประมวลผลไคลเอนท์เซิร์ฟเวอร์

รูปแบบของแอพพลิเคชั่นของระบบไคลเอนท์เซิร์ฟเวอร์แบ่งได้เป็น 6 ประเภท โดยมีรายละเอียดดังนี้

1. ไคลเอนท์เซิร์ฟเวอร์ที่ทำงานบนเครื่องเดียวกัน (Stand-alone Client/Server) แอพพลิเคชั่นประเภทนี้จะมีผู้ขอใช้บริการประมวลผลอยู่บนเครื่องเดียวกับที่ให้บริการ ลักษณะการทำงานเช่นนี้จะเป็นการบั่นทอนประสิทธิภาพการประมวลผลสำหรับระบบจัดการฐานข้อมูลลงบ้าง แต่ความเร็วในการสื่อสารระหว่างผู้ขอใช้บริการกับผู้ให้บริการจะสูงมาก ผู้ให้บริการจะยังสามารถที่จะทำงานได้โดยการประมวลผลร่วมกับแอพพลิเคชั่นอื่น ๆ ของผู้ขอใช้บริการ ในกรณีที่มีผู้ขอใช้บริการและผู้ให้บริการหลาย ๆ ตัวรันอยู่บนฮาร์ดแวร์แพลตฟอร์มเดียวกันการใช้มัลติโปรเซสเซอร์อาจจะช่วยเพิ่มประสิทธิภาพการทำงานขึ้นได้ แต่ว่าจะไม่สามารถนำเอาเทคโนโลยีด้านการประมวลผลแบบกระจายหรือการประมวลผลแบบกระจายมาใช้ในกรณีนี้ได้เลย

2. แสตนอโลนไคลเอนท์เซิร์ฟเวอร์ (Stand-alone LAN Client/Server)

ระบบไคลเอนท์เซิร์ฟเวอร์แบบนี้จะเป็นรูปแบบของไคลเอนท์เซิร์ฟเวอร์ในวงแลนวงหนึ่ง มีการทำงานของผู้ขอใช้บริการแต่ละตัวอาจจะรับผิดชอบงานด้านการนำเสนอข้อมูลประมวลผลธุรกิจและลอจิกทางด้านฐานข้อมูลในขณะที่ผู้ให้บริการจะรับผิดชอบในเรื่องของการเรียกใช้ข้อมูลสำหรับผู้ขอใช้บริการภายในวงแลน ข้อเสียของระบบนี้เมื่อเทียบกับระบบในแบบก่อนนี้ คือการสื่อสารระหว่างผู้ขอใช้บริการกับผู้ให้บริการทำโดยผ่านการเชื่อมต่อของแลนจะช้ากว่าการใช้หน่วยความจำร่วมกันของระบบที่แล้วมาก

3. แมนนวลเอ็กแทรกต์ไคลเอนท์เซิร์ฟเวอร์ (Mannual extract Client/Sever)

เป็นไคลเอนท์เซิร์ฟเวอร์ที่การประมวลผลกระทำโดยเรียกใช้ข้อมูลบางส่วนของทั้งหมดที่ได้ทำการย้ายไปเก็นไวในเครื่องของผู้ใช้บริการ ข้อมูลส่วนนี้ถูกสร้างขึ้นด้วยวิธีการกระจายข้อมูลแบบแมนนวลเอ็กแทรกต์ ลักษณะการทำงานของแอพพลิเคชั่นสามารถเกิดขึ้นโดยผู้ใช้ส่งคำสั่งไปยังผู้ให้บริการเพื่อเรียกใช้ข้อมูล ซึ่งในกรณีนี้มักจะถูกกำหนดให้ทำการอ่านอย่างเดียว การคัดข้อมูลและทำการย้ายนั้นเป็นสิ่งที่สำคัญและจำเป็นต้องทำเพราะว่าโดยปกติแล้วข้อมูลทั้งหมดมักจะไม่ได้อยู่ในรูปแบบที่ผู้ใช้

ต้องการ ตัวอย่างเช่น ผู้ใช้อาจจะต้องการดูข้อมูลสรุป หรือข้อมูลที่ได้ทำการรวบรวมแล้วมากกว่าที่จะดูข้อมูลโดยละเอียด การรวบรวมข้อมูลหรือทำสรุปจะการทำที่เครื่องของผู้ขอใช้บริการ ข้อเสียอย่างหนึ่งที่อาจจะเกิดขึ้นคือข้อมูลในส่วนที่เก็บอยู่ที่เครื่องของผู้ขอใช้บริการอายยะไม่ถูกต้องกับความเป็นจริง ถ้าข้อมูลส่วนดังกล่าวกำลังถูกเรียกใช้โดยผู้ขอใช้บริการและในขณะเดียวกันก็กำลังถูกเปลี่ยนแปลงที่ผู้ให้บริการ

4. ซิงเกิลไซต์อัพเดทไคลเอนท์เซิร์ฟเวอร์ (Single-site updata Client/Server)

เป็นระบบที่มีความสามารถสูง โดยมันจะสามารถส่งคำสั่งที่ประกอบด้วยคำสั่งหลายคำสั่งส่งไปยังผู้ให้บริการหลาย ๆ ตัวที่อยู่ห่างไกลได้ แต่ข้อมูลที่ทำการเรียกใช้จากผู้ให้บริการแต่ละตัวมักจะไม่มีความสัมพันธ์กัน ทั้งนี้เนื่องจากว่าผู้ให้บริการแต่ละตัวไม่ได้ต่อเชื่อมกันเป็นเครือข่ายเดียวกันและไม่มีผู้ให้บริการตัวใดทำหน้าที่เป็นตัวกลางในการสื่อสารระหว่างผู้ให้บริการ โดยการใช้เส้นทางเครือข่ายผ่านทางผู้ขอใช้บริการอีกทอดหนึ่ง (Two phase commit protocol) จากสาเหตุนี้ทำให้การประมวลผลแบบนี้อนุญาตให้ผู้ขอใช้บริการสามารถที่จะทำการแก้ไขข้อมูลที่ผู้ให้บริการได้เพียงตัวเดียวเท่านั้น ถ้าหากข้อมูลที่เก็บอยู่ ณ หน่วยเก็บข้อมูลของผู้ให้บริการตัวอื่นด้วย ถ้าแอพพลิเคชั่นของผู้ใช้บริการสามารถสนับสนุนให้ตัวผู้ขอใช้บริการทำหน้าที่เป็นตัวกลางระหว่างผู้ให้บริการทั้งหลายแล้ว ข้อจำกัดข้างต้นก็สามารถที่จะแก้ไขได้

ถึงแม้ว่าประมวลผลแบบนี้จะสามารถแก้ไข้ขอมูลที่ผู้ให้บริการได้เพียงหนึ่งตัว แต่ก็ยังมีความเป็นไปได้ที่อาจจะเกิดเดดล็อคขึ้นในเวลาที่มีผู้ใช้หลาย ๆ คนเรียกใช้ข้อมูลพร้อม ๆ กันดังนั้นจำเป็นต้องมีระบบมาทำการควบคุมการเรียกใช้ข้อมูลด้วย การกระจายข้อมูลของไคลเอนท์เซิร์ฟเวอร์ประเภทนี้อาจทำได้โดยวิธีแมนนวลเอ็กแทรกต์

5. มัลติไซต์อัพเดทไคลเอนท์เซิร์ฟเวอร์ (Multi-site update Client/Server)

ลักษณะแอพพลิเคชั่นประเภทนี้จะสนับสนุนการติดต่อระหว่างผู้ให้บริการแต่ละตัว ดังนั้นผู้ใช้จึงสามารถที่ออกคำสั่งประเภทที่จะแก้ไขข้อมูลที่เก็บอยู่หลายที่ได้ ถ้ามองในอีกแง่หนึ่งก็คือข้อมูลที่เก็บอยู่ ณ ที่ต่าง ๆ กัน สามารถที่จะมีความสัมพันธ์กันได้ ลักษณะของไคลเอนท์เซิร์ฟเวอร์ประเภทนี้จะเป็นประเภทแรกที่มีความสามารถในเรื่องการกระจายฐานข้อมูลและเมื่อมีความสามารถในเรื่องนี้แล้ว การกระจายข้อมูลจะถูกการ

ทำด้วยวิธีสแนพชอต (Snapshots) จากผู้ใช้บริการฐานข้อมูลที่จะเป็นวิธีแมนนวลเอ็กแทรกต์

6. ไคลเอนท์เซิร์ฟเวอร์แบบระบบฐานข้อมูลแบบกระจาย (Distribute database Client/Server)

เป็นระบบไคลเอนท์เซิร์ฟเวอร์ที่ใช้แอพพลิเคชั่นฐานข้อมูลแบบกระจายและใช้การประมวลผลแบบดิสทริบิวเต็ดรีเควสต์ (Distribute request) มีลักษณะคือ ผู้ให้บริการฐานข้อมูลจะสนับสนุนทั้งการตัดแบ่งข้อมูล หรือการทำก็อปปี้ข้อมูลทั้งหมดไปเก็บไว้ตามหน่วยเก็บข้อมูลของผู้ให้บริการต่าง ๆ ซึ่งทำให้การอ่านข้อมูลสามาถทำได้ด้วยความรวดเร็วแต่การแก้ไขข้อมูลอาจจะต้องใช้เวลามากกว่าเพราะว่าจะต้องมีการติดต่อกันระหว่างผู้ให้บริการซึ่งอาจจะไม่ใช่เพียงแค่ 2 ตัว ดังนั้นเทคโนโลยีทางด้านการสื่อสารจึงมีบทบาทสำคัญในการที่จะขจัดปัญหาในเรื่องของความเร็ว

ความสามารถที่จำเป็นสำหรับแอพพลิเคชั่นประเภทนี้คือ การที่แอพพลิเคชั่นจะไม่จำเป็นต้องรู้ตำแหน่งของผู้ให้บริการ หรือตำแหน่งที่เกิดการประมวลผลฐานข้อมูล การควบคุมประสิทธิภาพโดยรวมของระบบ การควบคุมความถูกต้องของข้อมูลที่กระจายเห็นอยู่ตามที่ต่าง ๆ และการควบคุมการทำการกระจายข้อมูลซึ่งเป็นส่วนสำคัญของระบบไคลเอนท์เซิร์ฟเวอร์ประเภทนี้

ข้อดีข้อเสียของระบบไคลเอนท์เซิร์ฟเวอร์

ข้อดี

- เนื่องจากระบบไคลเอนท์เซิร์ฟเวอร์มีการแบ่งแยกการประมวลออกระหว่างส่วนไคลเอนท์และส่วนเซิร์ฟเวอร์ ซึ่งทำให้การประมวลผลข้อมูลจะทำที่เซิร์ฟเวอร์ ทำให้ความเร็วของระบบจัดการฐานข้อมูลไม่ขึ้นอยู่กับความเร็วของเวิร์คสเตชั่น (WorkStation) ดังนั้นเวิร์คสเตชั่นที่ใช้ไม่จำเป็นต้องเป็นเครื่องที่มีประสิทธิภาพสูง เพียงแต่สามารถให้ระบบฟรอนท์เอ็นทำงานได้ก็เพียงพอ

- เนื่องจากมีการแบ่งแยกการประมวลผลออกเป็น 2 ฝั่ง ทำให้ลดโหลดในการติดต่อระบบเครือข่ายระหว่างเซิร์ฟเวอร์และไคล์เอนท์ ถ้าเปรียบเทียบกับระบบโครงสร้างแบบรวมศูนย์ จะต้องมีการส่งไฟล์ฐานข้อมูลทั้งไฟล์ไปกลับระหว่างผู้ให้บริการ

และผู้ใช้บริการอยู่ตลอดเวลาที่มีการเรียกใช้ข้อมูล แต่สำหรับระบบไคลเอนท์เซิร์ฟเวอร์จะเป็นการส่งคิวรี่และผลลัพธ์ที่ได้จากดาต้าเบสเซิร์ฟเวอร์ เพราะการใช้ประโยคคำสั่งเอสคิวแอลในแอพพลิเคชั่นของระบบไคลเอนท์เซิร์ฟเวอร์สามารถที่จะสร้างตารางข้อมูลบรรจุผลลัพธ์ที่ได้จากการรวม ตัดทอน และเปลี่ยนแปลงข้อมูลจากตารางข้อมูลจากผู้ให้บริการแล้วค่อยส่งคำสั่งเอสคิวแอล (SQL) ไว้ในตัวมันเองโดยทำการเก็บได้วในลักษณะของสตอร์โพรซีเดอร์ (Store procedure) ซึ่งผู้ใช้จะทำการเรียกใช้โดยออกคำสั่งสั้น ๆ ให้ผู้บริการเรียกประโยคคำสั่งนั้น ๆ ออกมาทำงานจะเป็นการช่วยลดปริมาณข้อมูลที่ส่งผ่านเข้าไปในเครือข่ายได้ทางหนึ่งแต่ก็มีปัญหาอยู่ว่ายังไม่มีการกำหนดมาตรฐานในเรื่องของสตอร์โพรซิเดอร์ขึ้นมาและระบบจัดการฐานข้อมูลหลายตัวยังไม่สนับสนุนความสามารถในเรื่องนี้จากการแบ่งแยกออกเป็นไคลเอนท์เซิร์ฟเวอร์ทำให้ส่วนที่ไคลเอนท์ทำงานอยู่และแพลตฟอร์มสามารถเป็นอะไรก็ได้ซึ่งแพลตฟร์อมที่ใช้อาจจะเป็นพีซีที่เข้ากันได้กับพีซีของไอบีเอ็ม (IBM) ,แมคอินทอช (MACINTOSH), ยูนิกซ์เวิร์คสเตชั่น (UNIX Workstation) นอกจากนั้นยังสามารถใช้กับระบบปฏิบัติการได้หลายตัว เช่น ดอส(DOS),พีซีดอส(PC DOS),เอ็มเอสวินโดว์(Ms Windows),ไอบีเอ็มโอเอสทู(IBM OS/2),ระบบแอพเปิล(Apples System) ทำให้เวิร์คสเตชั่นสามารถใช้แอพพลิเคชั่นสามารถใช้แอพพลิเคชั่นตัวใดก็ได้ใส่การเข้าถึงฐานข้อมูล

- ระบบไคลเอนท์เซิร์ฟเวอร์สามารถรักษาความคงสภาพของข้อมูล (Data integrity) ได้โดยระบบจัดการฐานข้อมูลจะไม่อนุญาตให้ผู้ใช้เข้าถึงฐานข้อมูลจากภายนอก เช่นอาจจะทำการเข้ารหัสไฟล์เพื่อป้องกันผู้ใช้ดูข้อมูลจากภายนอก นอกจากนั้นระบบจัดการฐานข้อมูลยังสามารถทำการแบคอัพไปยังเทปแบบเรียลไทล์ (real time) ได้คือขณะที่ฐานข้อมูลกำลังถูกใช้ก็มีการแบคอัพไปยังเทป การทำดิสค์มิเรอร์ (DisK Mirroring) ซึ่งการทำสิ่งเหล่านี้เพื่อรักษาความถูกต้องของข้อมูลจากการเกิดการเสียหายของระบบหรือไฟดับ

- สำหรับกำหนดขนาดของผู้ใช้บริการและผู้ให้บริการได้อย่างอิสระและสามารถลดและขยายในภายหลัง

- แอพพลิเคชั่นต่าง ๆ สามารถใช้ข้อมูลบนผู้ให้บรีการร่วมกันได้

ข้อเสีย

- เสียค่าใช้จ่ายในการดูและบำรุงรักษาสูง
- ฮาร์ดแวร์ที่ทำหน้าที่เป็นดาต้าเบสเซิร์ฟเวอร์จะต้องเป็นเครื่องที่มีประสิทธิภาพสูงซึ่งจะทำให้เสียค่าใช้จ่ายสูง
- ซอฟท์แวร์ที่เป็นระบบจัดการฐานข้อมูลจะมีราคาสูง
- มีผลกระทบจากการกระจายข้อมูลต่อประสิทธิภาพของระบบ
- การบริการระบบข้อมูลทำได้ลำบาก ในระบบที่ข้อมูลทุกอย่างเก็บรวบรวมอยู่ที่ส่วนกลายการควบคุมจะกรทำได้สะดวก แต่เมื่อเรากระจายการพัฒนาระบบงานการประมวลผลแอพพลิเคชั่นและการจัดเก็บข้อมูลออกไปแล้ว ความง่ายและความสะดวกในการควบคุมจะสูญเสียไปซึ่งจะมีปัญหาในเรื่องต่าง ๆ ดังนี้
  - การจัดการไลบารีของโปรแกรมต่าง ๆ ที่กระจายกันอยู่
  - การจัดการข้อมูลที่เก็บอยู่ตามที่ต่าง ๆ
  - การตรวจสอบและเพิ่มประสิทธิภาพในการทำงานของระบบ
  - การสำรองข้อมูลที่เก็บอย่างกระจายในระบบ
  - การจัดการระบบเครือข่าย

ดาต้าเบส(DataBase)

คือแหล่งที่ใช้เก็บข้อมูลซึ่งข้อมูลเหล่านั้นมีความสัมพันธ์กันในทางโลจิคอลลี่ (Logically) ซึ่งอาจเก็บไว้ในรูปแบบความสัมพันธ์ในลักษณะต่าง ๆ ซึ่งผู้ใช้อาจเรียกดูข้อมูลในตาราง(Table) โดยผ่านวิว(View) ได้ ตามขอบเขตที่กำหนด

ในทางฟิสิกส์คอล(Physical) อาจจะเก็บตาราง(Table) ไว้ในไดร์ฟ(Drive) เดียวกัน หรือต่างกันและสามารถกระจายข้อมูลไปอย่างสม่ำเสมอ เมื่ออ่านข้อมูลจะอ่านมาแบบขนาน ซึ่งทำให้สามารถอ่านข้อมูลได้รวดเร็ว แต่เมื่อเวลาเรียกมาใช้ก็จะเห็นเป็นตาราง(Table) เดียวกัน

ดาต้าเบส(Database) ส่วนใหญ่จะใช้การมองข้อมูลแบบโลจิคอล(Logical) หากมองในทางฟิสิกส์คอล(Physical) จะเป็นระดับที่ซับซ้อนมากขึ้น

2.3.2 โอเพนดาต้าเบสคอนเน็กติวิตี้(โดดีบีซี)((Open DataBase Connectivity:ODBC)

คือวิธีการติดต่อและเข้าถึงแอพพลิเคชั่น(Application) สู่ระบบการจัดการฐานข้อมูลโดยใช้ภาษาเอสคิวแอล(SQL) เป็นมาตราฐานการเข้าถึงข้อมูล ความสามารถในการเชื่อมต่อแบบนี้ทำให้แอพพลิเคชั่น(Application) สามารถเข้าถึงฐานข้อมูลได้หลายรูปแบบซึ่งทำให้ผู้พัฒนาโปรแกรมสามารถพัฒนาโปรแกรมไปได้โดยไม่ต้องทำการระบุชนิดของระบบการจัดการฐานข้อมูล

แต่เดิมนั้นการพัฒนาโปรแกรมประยุกต์ที่ใช้งานเกี่ยวกับฐานข้อมูล การเข้าใช้ฐานข้อมูลโปรแกรมเหล่านี้จะทำผ่านการเรียกใช้เอ็มเบ็ดเต็ด เอสคิวแอล (Embedded SQL) ซึ่งในขณะนั้นวิถีทางแบบนี้ก็ดูจะเป็นไปได้ที่เดียว เพราะว่าตัวโปรแกรมสามารถทำการเปลี่ยนรูปแบบของระบบไม่ว่าจะเป็นทางด้านฮาร์ดแวร์ หรือซอฟท์แวร์ได้หลายรูปแบบ รวมทั้งระบบปฏิบัติการด้วย (โดยการคอมไฟล์ใหม่ทุกครั้งที่มีการย้ายระบบ)

อย่างไรก็ตามในการพัฒนาโปรแกรมในระบบที่มีความแตกต่างกัน เช่น การเรียกใช้ข้อมูลของออร์เดิล(Oracle) จากไมโครซอฟท์ เอ็กเซล(Microsoft Excel) วิธีการเข้าถึงข้อมูลแบบเดิมนั้นจะต้องทำการพรีคอมไพล์โค้ดของเอ็กเซล(Excel) และ ออร์ราเดิล(Oracle) โดยการใช้ไอบีเอ็ม พรีคอมพลาย(IBM Precompiler) และออร์เดิล พรีคอมพลาย(Oracle Precompiler) ตามลำดับ ซึ่งจะเห็นว่าเป็นการยุ่งยากมากทีเดียว

สำหรับวิธีการต่อเชื่อมแบบโอดีบีซี(ODBC) จะให้ความสะดวกในการติดต่อข้อมูลมากกว่าวิธีการดั้งเดิม โดยการกำหนดมาตรฐานการต่อเชื่อมของข้อมูล (Data Protocol,DBMS Capability) และแนวทางนี้ได้ทำให้เกิดความคิดที่จะสร้างไดร์เวอร์(Driver) การติดต่อกับการของานข้อมูลขึ้นมา (DLL)

2.3.2.1 ข้อดีของการติดต่อโดยใช้โอดีบีซี

- ฟังก์ชั่นของโอดีบีซีอนุญาตให้แอพพลิเคชั่น(Application) ติดต่อกับระบบจัดการฐานข้อมูล ได้โดยสะดวก (การทำคำสั่ง SQL และการรับผลลัพธ์)
- ใช้ภาษาเอสคิวแอล(SQL) ตามมาตรฐาน SQL CAE,X/OPEN และ SQL ACCESS GROUP (SAG)
- มีการกำหนดการส่งกลับรหัสความผิดพลาด (Error Code) เป็นมาตรฐานเดียวกัน
- เป็นวิธีการมาตรฐานในการติดต่อกับระบบจัดการฐานข้อมูล
- มีการกำหนดชนิดของข้อมูล (Data Type) เป็นมาตรฐาน
- ชุดคำสั่งเอสคิวแอลสามารถกำหนดได้แม้ในขณะรัน
- สามารถเขียนโปรแกรมชุดเดียวแต่สามารถเข้าใช้ระบบจัดการฐานข้อมูลได้หลายตัว
- ตัวโปรแกรมไม่ต้องรับผิดชอบในการดูแลการติดต่อข้อมูลกับระบบจัดการฐานข้อมูล
- ค่าข้อมูลสามารถถูกส่งหรือรับได้ในรูปแบบที่สะดวกขึ้น

2.3.2.2 องค์ประกอบของโอดีบีซี(ODBC)

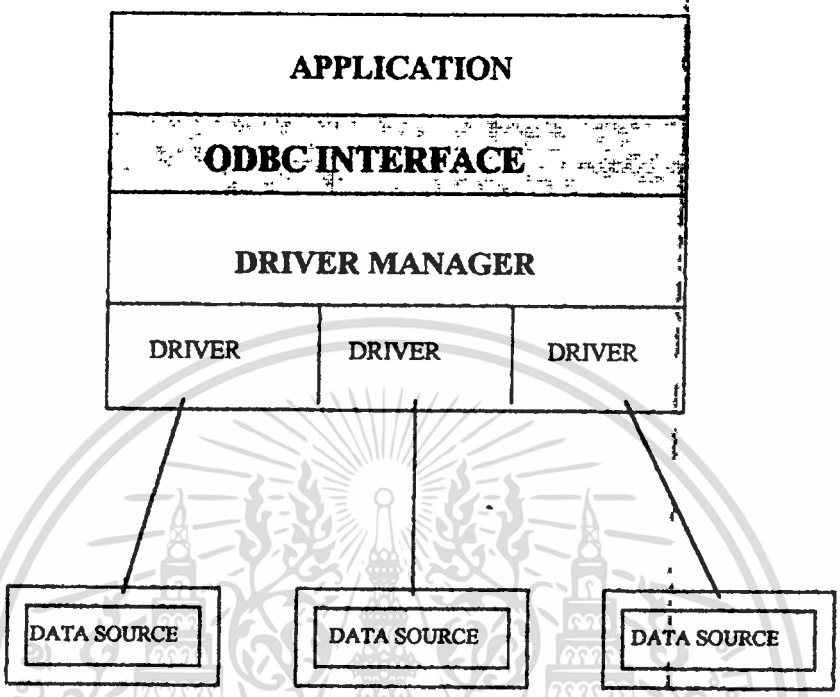
สถาปัตยกรรมของโอดีบีซี(ODBC) ประกอบด้วย 4 ส่วนสำคัญ

1. แอพพลิเคชั่น(Application) ทำหน้าที่ประมวลผลและเรียกใช้ฟังก์ชั่นของโอดีซี(ODBC) ตามคำสั่งภาษาเอสคิวแอล(SQL) พร้อมทั้งทำการรับผลลัพธ์ด้วย

2. ตัวจัดการไดเวอร์(Driver Manager) ทำหน้าที่โหลดไดร์เวอร์(Driver) และเชื่อมต่อกับแหล่งข้อมูล

3. ไดร์เวอร์(Driver) ทำหน้าที่ประมวลผลการเรียกใช้ผังก์ชั่นของ โอดีบีซี(ODBC) ส่งคำสั่งเอสคิวแอล(SQL) ไปสู่แหล่งข้อมูลที่ถูกต้องและทำการส่งผลลัพธ์กลับให้แอพพลิเคชั่น(Application) และในบางครั้งไดร์เวอร์(Driver) จะทำหน้าที่แปลงคำสั่งที่ส่งมาให้อยู่ในรูปที่สนับสนุนโดยระบบจัดการฐานข้อมูลแต่ละชนิดอีกด้วย

4. ดาต้า ซอร์ส(Data Source) เป็นแหล่งข้อมูที่ผู้ใช้ต้องการเข้าถึง



รูปที่ 9 แสดงองค์ประกอบของ โอดีบีซี(ODBC)

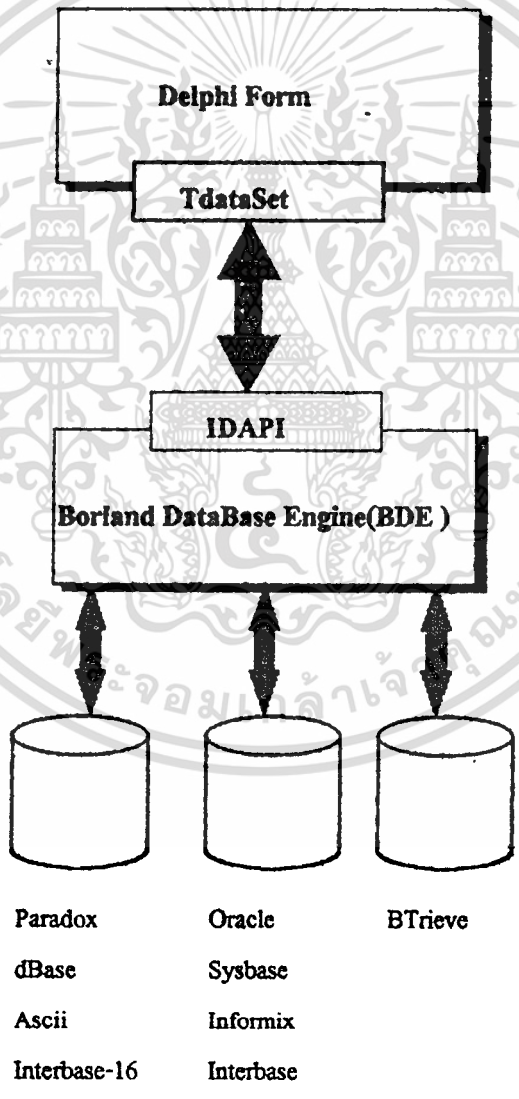
โปรแกรมจะเรียกใช้การเชื่อมโอดีบีซี(ODBC) ในการทำงานต่อไปนี้

1. ร้องขอการต่อเชื่อมกับแหล่งข้อมูล
2. ส่งคำสั่งเอสคิวแอล(SQL) สู่แหล่งข้อมูล
3. กำหนดพื้นที่การจัดเก็บและรูปแบบของข้อมูล ที่เป็นผลลัพธ์จาก เอสคิวแอล รีเควสซ์(SQL Request)
4. ร้องขอผลลัพธ์
5. ประมวลผลและจัดการกับข้อผิดพลาด
6. รายงานผลให้กับผู้ใช้ (ถ้าจำเป็น)
7. ร้องขอการคอมมิท(Commit) หรือ โรลแบ็ค(Roll back) สำหรับควบคุมการประมวลผล ทรานแซคชั่น(Transaction)
8. ยกเลิกการติดต่อกับแหล่งข้อมูล

2.3.3 เดลไฟล์(Delphi) กับงานด้าต้าเบส(Data base)

สถาปัตยกรรมคาต้าเบสของเดลไฟล์(Architecture of Delphi Database Applications)

ซึ่งประกอบด้วย 3 Layers ดังรูปที่แสดง



รูปที่ 10 แสดงสถาปัตยกรรมทางคาต้าเบสของเดลไฟล์

เลเยอร์แรก (layer 1) เป็น การเข้าถึง (Accessing) และจัดการกับดาต้าเบส (database) สามารถทำได้โดยใช้สแตนดาดดาต้าเบสคอมโพเนนท์(Standard Database Companents)ซึ่ง เดลไฟล์(Delphi) ได้จัดเตรียมไว้ให้

เลเยอร์ที่สอง(layer 2) เป็นตัวกลางในการจัดการกับดาต้าเบส(database) จะใช้บอร์แลนด์ดาต้าเบสเอ็นจิน(BDE:Borland Database Engine) ทำการติดต่ออินเตอร์เฟส (Interface) กับดาต้าเบส(Database) ซึ่งมีสับรูทีน(Subroutines) และการบริการ(Services) สำหรับใช้งาน โดยบอร์แลนด์ดาต้าเบสเอ็นจินแอพพลิเคชั่นโปรแกรมมิ่งอินเตอร์เฟส (BDE Application Programming Interface)

เลเยอร์ที่สาม(layer 3) เป็นการเข้าถึงข้อมูลทางฟิสิกก์คอล(Physical) ซึ่งเก็บอยู่ในตารางดาต้าเบส(Database Tables) ซึ่งจะมีดีบีเอ็มเอส(DBMS) เป็นตัวจัดการกับ ดาต้าเบส(Database)เช่น อาจมีรูปแบบเป็นพาราดอก,ดีเบสและดีบีเอ็มเอส(Paradox หรือ Dbase Tables และ DBMS) คือ ออราเคิล(Oracle),ซีสเบส(Sybase),อินฟอมิค(Informix), หรืออินเตอร์เบส เซิร์ฟเวอร์(Interbase Server) ซึ่งมีวิธีจัดการแตกต่างกัน

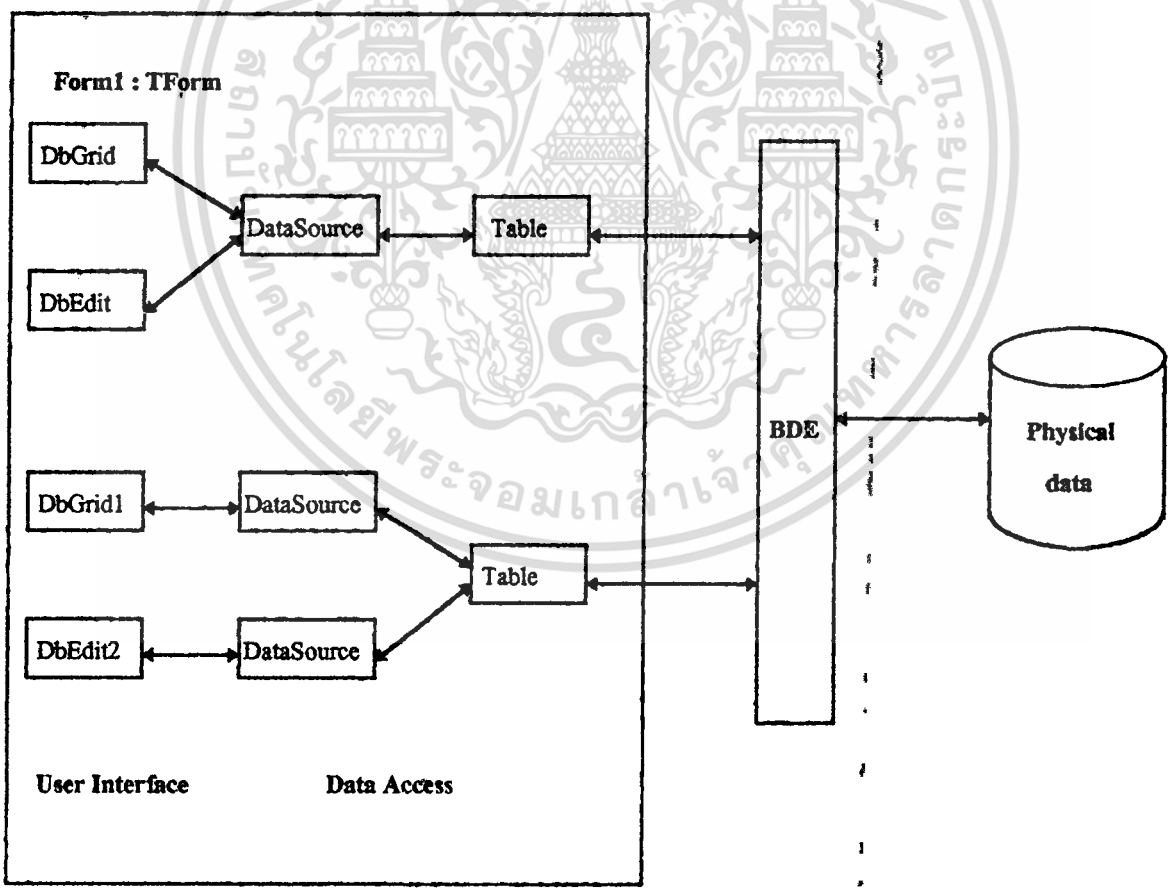
2.3.3.1 ความสามารถและคุณลักษณะพิเศษทางด้านงานฐานข้อมูลของ Delphi

การสร้างแอพพลิเคชั่น(Application) งาน ดาต้าเบส(DataBase) โดยเดลไฟล์ Delphi) จะอาศัย

- เครื่องมือพัฒนางานด้านฐานข้อมูลของ Delphi (delphi database development tool)
- ดาต้า-แอคเซส คอมโพเนนท์(Delphi data access component) คุณสามารถใช้มันเข้าไปกำหนดค่าของฟิลด์(Fields) ในเรคคอร์ด(records) และสามารถจัดการข้อมูลได้ เช่น กำหนด ตาราง(table,list table) ภายในดาตาเบส(Database) ซึ่งเครื่องมือเหล่านี้จะอยู่ในหน้าดาต้า-เอคเซส(Data Access page ของ Delphi's Component Palette) ซึ่งประกอบด้วย Ttable,TDataSource,TQuery และอื่น ๆ ซึ่งในการผ่านเข้าไปในโลเลเวล(Low Level) จะผ่านบอร์แลนด์ดาต้าเบสเอ็นจิน(BDE)
- ดาต้า-อแวร์ จียูไอ คอมโพเนนท์ของ(Delphi data aware GUI component) เป็นวิชวล (Visual Component) ที่ใช้แสดงข้อมูลของดาต้าฟิลด์(Data Field) ของตาราง(Table) ได้

เช่น DbGrid,Dblabel,DblistBox และอื่น ๆ ซึ่งจะเกี่ยวข้องกับดาต้าเบส(Database) แต่เครื่องมือ(Tool) เหล่านี้จะต้องใช้ร่วมกับดาต้าซอร์ท,ดาต้าแอกเซท(DataSource,Data Access Component)

โดยใช้คอมโพเนนท์(component) ในการติดต่อกับบอร์เครือข่ายท้องถิ่น ดาต้าเบส เอนจิ้น หรือบีดีอี(Borland Database Engine,BDE) ซึ่งจะทำหน้าที่ติดต่อกับฐานข้อมูลอีกที ภาพต่อไปนี้จะแสดงความสัมพันธ์ของเครื่องมือต่าง ๆ และ แอพพลิเคชั่น(Application) งานดาต้าเบส(Database) กับบอร์แลนด์ดาต้าเบสเอ็นจิน(BDE) และแหล่งข้อมูล (DataSource)



รูปที่ 11 แสดงสถาปัตยกรรมการติดต่อระหว่างเดลไฟล์กับดาต้าเบส

เดลไฟล์ (Delphi) ใช้ออปเจคโอเรียนเต็ด คอมโพเนนท์ (Object-Oriented Component) ในการสร้างแอพพลิเคชั่นงานฐานข้อมูลเช่นเดียวกับแอพพลิเคชั่นอื่นๆที่ไม่ใช่แอพพลิเคชั่นงานฐานข้อมูล ดาต้าเบสคอมโพเนนท์ (Database Component) จะเหมือนกับคอมโพเนนท์มาตรฐานทั่วไป ที่มีแอททริบิวท์ หรือเรียกว่า พรอพเพอร์ตี้ (Property) ซึ่งจะถูกตั้งค่าโดยโปรแกรมเมอร์ในขณะที่ทำการออกแบบ และอาจจะตั้งค่าในขณะที่กำลังรันโปรแกรมก็ได้ ดาต้าเบสคอมโพเนนท์จะมีค่าที่ถูกตั้งไว้ให้แล้ว (Default) เพื่อให้มันสามารถทำงานได้โดยไม่ต้องเขียนโปรแกรมเพิ่มหรือเพิ่มเพียงเล็กน้อย

เดลไฟล์Delphi ประกอบด้วย 2 คอมโพเนนท์พาเลท (Component Palette) ที่มีดาต้าเบส
คอมโพเนนท์ คือ

- หน้าสำหรับคอมโพเนนท์ที่เข้าถึงข้อมูล (Data-Access Page) จะมีออปเจคที่ทำให้การเข้าถึงข้อมูลง่ายขึ้น โดยการเอนแคปซูเลท (Encapsulate) แหล่งข้อมูล เช่น ฐานข้อมูลที่จะติดต่อตารางในฐานข้อมูล หรือฟิลด์ (Field) ของข้อมูลที่ต้องการตัวอย่างของออปเจคเหล่านี้ได้แก่ Ttable Tquery ,Tdata- source และ Treport
- หน้าสำหรับคอมดพเนนท์ที่ควบคุมข้อมูล (Data-Control Page) จะมีคอมโพเนนท์ไว้ติดต่อกับผู้ใช้เพื่อแสดงข้อมูลในรูปแบบต่างๆซึ่งจะเหมือนกับคอมโพเนนท์มาตรฐานที่ใช้ติดต่อกับผู้ใช้ เพียงแต่ข้อมูลเหล่านั้นมาจากตารางในฐานข้อมูล ตัวอย่างของคอมโพเนนท์ควบคุมข้อมูลที่ใช้บ่อยๆได้แก่ TDBEdit,TDBNavigator และ TDBGrid

ดาต้าเซทคอมโพเนนท์(Dataset Component) เช่น TTable,TQuery และ TStoreProc จะมองไม่เห็นในขณะรันโปรแกรม แต่มีไว้ให้แอพพลิเคชั่นติดต่อกับข้อมูลของมันผ่านบีดีอีคอมโพเนนท์ ควบคุมข้อมูลจะติดต่อกับดาต้าเซทคอมโพเนนท์โดยผ่านทาง Tdatasource คอมโพเนนท์ เพื่อให้การติดต่อกับข้อมูลแบบมุมองเห็นได้

จากภาพแสดงให้เห็นว่า คอมโพเนนท์เข้าถึงข้อมูลกับคอมโพเนนท์ควบคุมข้อมูลสัมพันธ์กับข้อมูล สัมพันธ์ซึ่งกันและกัน และสัมพันธ์กับส่วนติดต่อกับผู้ใช้แอพพลิเคชั่นอย่างไร

2.3.3.2 หลักการพัฒนาแอพพลิเคชั่นงานฐานข้อมูล

1. รูปแบบของการพัฒนา

เนื่องจากการออกแบบแอพพลิเคชั่น จะต้องขึ้นกับโครงสร้างฐานข้อมูลที่จะเข้าถึงโดยฐานข้อมูลจะต้องถูกกำหนดล่วงหน้าก่อนที่จะทำการพัฒนาแอพพลิเคชั่น

รูปแบบในการพัฒนาแอพพลิเคชั่นงานฐานข้อมูลของเดลไฟล์(Delphi) เป็นไปได้ 4 รูปแบบคือ

- ไม่มีฐานข้อมูลอยู่ก่อน หรือจำเป็นต้องทำการกำหนดใหม่หมด สามารถทำได้โดย

- ใช้ ดาต้าเบส เดสค์ทอป ยูทิลิตี้ (Database Desktop Utility) เพื่อกำหนดตารางแบบพาราดอกซ์และดีเบส

- สำหรับ เอสคิวแอลเซิร์ฟเวอร์ สามารถใช้เครื่องมือที่เป็นของเซิร์ฟเวอร์หรือของดาต้าเบส เดสค์ทอปก็ได้ ตัวอย่างเช่น สำหรับโลคอลอินเตอร์เบสเซิร์ฟเวอร์สามารถใช้ วินโดว์ ไอเอสคิวแอล (Windows ISQL) ในการกำหนดฐานข้อมูล

- มีฐานข้อมูลปรากฏอยู่บนแหล่งข้อมูล ที่เป็นเดสค์ทอป หรือ เป็นเครือข่ายท้องถิ่น (LAN) และฐานข้อมูลเข้าถึงข้อมูล ณ ที่แห่งนั้น ถ้าบีดีอีและแหล่งข้อมูลอยู่บนเครื่องเดียวกันกับแอพพลิเคชั่นแล้ว แอพพลิเคชั่นนั้นจะเป็นแบบแสตนด์อโลน (Stand-Alone)

- ฐานข้อมูลปรากฏบนแหล่งข้อมูลแบบเดสค์ทอป และจะถูกอัพไซซ์ (Upsize) เป็น เอสคิวแอลเซิร์ฟเวอร์ แบบนี้จะเรียกว่าเป็น โลคอลเอสคิวแอลเซิร์ฟเวอร์

- ฐานข้อมูลอยู่บนเอสคิวแอลเซิร์ฟเวอร์ และแอพพลิเคชั่นจะเข้าถึงข้อมูลที่อยู่ที่เซิร์ฟเวอร์ แบบนี้จะเรียกว่าเป็นไคลเอ็นท์เซิร์ฟเวอร์แบบมาตรฐาน

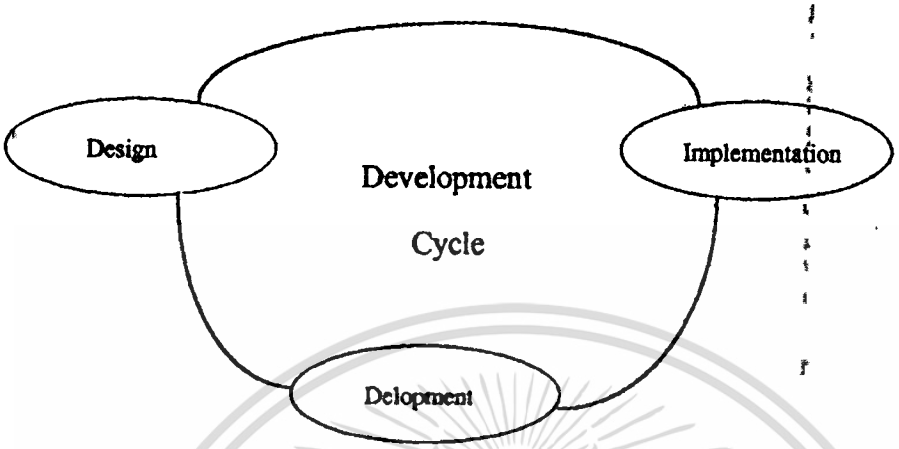
2. วงจรในการพัฒนาแอพพลิเคลชั่นงานฐานข้อมูล

การพัฒนาแอพพลิเคชั่นงานฐานข้อมูลมี 3 ขั้นตอนหลักคือ

- การออกแบบ และ ทำต้นแบบ (Prototyping)

- การสร้าง (Implement)

- การส่งมอบระบบใช้งานจริง (Deployment) และการบำรุงรักษา (maintenance)



รูปที่ 12 แสดงวงจรในการพัฒนาโปรแกรม

สำหรับแอพพลิเคชั่นแบบไคลเอนท์เซิร์ฟเวอร์ งานของฐานข้อมูลและงานของแอพพลิเคชั่นจะแยกกันเด่นชัดเนื่องจากจะรันบนแพลตฟอร์มที่ต่างกัน และระบบปฏิบัติการมักจะต่างกันด้วย เช่น ยูนิกซ์เซิร์ฟเวอร์ และวินโดว์ 3.1 ไคลเอนท์

ดังนั้นในการพัฒนา ความรับผิดชอบในการแบ่งว่างานใดควรทำโดยดาต้าเบสเซิร์ฟเวอร์ งานใดควรทำโดยไคลเอนท์แอพพลิเคชั่นจะอยู่ในขั้นตอนการออกแบบ โดยทั่วไปแล้วจะมีการแบ่งงานกันอย่างชัดเจนระหว่าง 2 ฝั่ง แต่กรรมวิธีทางฐานข้อมูลเช่น สตอร์โพรซิเดอร์บางอย่างก็สามารถทำได้โดยไคลเอ็นท์แอพพลิเคชั่น ดังนั้นจึงขึ้นกับลักษณะของการนำระบบไปใช้จริงที่คาดหวังไว้,ความต้องการของแอพพลิเคชั่น และข้อควรพิจานณาอื่นๆ ดังนั้นในการออกแบบสามารถจะกำหนดไว้ก่ารทำงานจะอยู่ฝั่ง ไคลเอนท์ หรือฝั่งเซิร์ฟเวอร์

2.3.3.3 การส่งมอบแอพพลิเคชั่นเพื่อนำไปใช้จริง (Deploying an application)

หมายถึงการนำแอพพลิเคชั่นไปให้ผู้ใช้จริงใช้ และจัดหาซอฟแวร์ที่จำเป็นที่ผู้ใช้จะต้องมีไว้ใช้ในการรันแอพพลิเคชั่น ในสภาวะแวดล้อมของผู้ใช้ สำหรับแอพพลิเคชั่นที่ไม่ใช่งานฐานข้อมูลของ Delphi จะใช้เพียงไฟล์ .EXE เท่านั้น เนื่องแอพพลิเคชั่นของ

เดลไฟล์ (Delphi) ไม่ต้องการตัวแปลภาษาขณะทำงาน (run-time interpreter) หรือ ไดนามิก ลิงค์ ไลบรารี (DLL)

2.3.3.4 ไฟล์ที่ต้องการในการส่งมอบแอพพลิเคชั่นเพื่อไปใช้จริง (แบบสแตนด์อโลน)

โดยทั่วไปในการส่งมอบแอพพลิเคชั่นงานฐานข้อมูลเพื่อนำไปใช้จริง จะต้องสร้างแพคเกจ (Package) ที่ประกอบด้วยไฟล์ทั้งหมดที่ผู้ใช้ต้องการในการรันแอพพลิเคชั่นและเข้าถึงแหล่งข้อมูล

ไฟล์เหล่านั้นได้แก่

- ไฟล์ .EXE และ ไฟล์ .DLL (ถ้ามี) ของแอพพลิเคชั่น
- ไฟล์ที่เป็นส่วนประกอบต่าง ๆ เช่น ไฟล์ .README . ไฟล์ .HLP หรือไฟล์สำหรับ ออนไลน์เฮลพ์ (OnLine Help)
- บีดีอี ซัพพอร์ท สำหรับการเข้าถึงฐานข้อมูล (เดสก์ทอป หรือเซิร์ฟเวอร์)
- รีพอร์ทสมิธ รันไทม์ (ReportSmith Runtime) สำหรับการพิมพ์รายงาน
- ถ้าแอพพลิเคชั่นใช้ วีบีเอ็กซ์คอนโทรล (VBX Control) จะต้องใช้ ไฟล์ BIVBX11.DLL

บทที่ 3
การวางแผนและขั้นตอนการทำงาน

หลังจากการศึกษาและวิเคราะห์ออกแบบระบบแล้วก็จะได้ผลลัพธ์ออกมา 3 ส่วน คือ ส่วนที่ทำการเข้ารหัสข้อมูลแบบอาร์เอสเอ(RSA) , ส่วนที่ทำงานกับฐานข้อมูลและส่วนของการออกแบบหน้าจอ(Front-end) ซึ่งจะแสดงรายละเอียดในการทำงานและวางแผนดังนี้

3.1 ส่วนของการเข้ารหัสข้อมูลแบบอาร์เอสเอ(RSA)

เป็นการเข้ารหัสโดยใช้ระบบระบบที่เปิดเผยคีย์(Public key Algorithm) ซึ่งความยาวของกุญแจ(key) สามารถเปลี่ยนแปลงได้ บางคนจะเลือกกุญแจ(key) ที่มีความยาวมากเพื่อเพิ่มความปลอดภัยในการป้องกันข้อมูล แต่บางคนจะเลือกกุญแจ(key) ที่มีความยาวสั้นเพื่อประสิทธิภาพมากขึ้นในการเข้ารหัสถอดรหัส เพราะกุญแจ(key) ยาวอาจมีความผิดพลาดขึ้นได้ แต่ส่วนมากจะใช้ที่มีความยาว 512 bitsขนาด

ขนาดชุด(block) ของข้อมูลที่ใช้ในการเข้ารหัสของอาร์เอสเอ(RSA) สามารถเปลี่ยนแปลงได้ แต่ชุดข้อความ(plaintext block) จะต้องมีขนาดเล็กกว่าความยาวของกุญแจ(Key) n และเมื่อเข้ารหัสจะได้ข้อความที่เข้ารหัส(ciphertext block) ที่มีความยาวเท่ากับขนาดความยาวของกุญแจ(Key) n การเข้ารหัสของอาร์เอสเอ(RSA) จะมีการทำงานที่ช้ากว่าการเข้ารหัสแบบระบบกุญแจที่เป็นความลับ(Secret Key Algorithm) เช่นดีอีเอส(DES) และไดเดีย(IDEA) ดังนั้นอาร์เอสเอ(RSA) มักจะไม่ใช้ในการเข้ารหัสข้อความที่มีขนาดยาว ๆ ส่วนมากจะใช้สำหรับเข้ารหัสแบบกุญแจที่เป็นความลับ(Secret Key) อีกครั้งหนึ่งและกุญแจที่เป็นความลับ(Secret Key) จะเป็นตัวเข้ารหัสข้อความ

ขั้นตอนการต่าง ๆ ในการเข้ารหัสและถอดรหัสแบบอาร์เอสเอ(RSA)

ตรวจสอบอัลกอริทึม ต่าง ๆ ที่ใช้ในการเข้ารหัสแบบอาร์เอสเอ(RSA)

- อัลกอริทึม (Algorithm) การหาค่าหารร่วมมาก(gcd)
- อัลกอริทึม (Algorithm) การหาค่าการหาค่ายกกำลัง แล้วมอด(Fastexp)
- อัลกอริทึม (Algorithm) การหาค่าอินเวอร์ส(Inverse)
- อัลกอริทึม (Algorithm) การหาค่าจำนวนเฉพาะ(Prime)
- อัลกอริทึม (Algorithm)การเข้ารหัสแล้วแปลงเป็นตัวอักษร(String)
- อัลกอริทึม (Algorithm) การถอดรหัสแล้วแปลงเป็นอักษร(String)

อาร์เอสเอ อัลกอริทึม (RSA Algorithm)

ขั้นแรกทำการสร้างกุญแจที่เปิดเผย(Generate A Public Key) และกุญแจที่ไม่เปิดเผย(Private Key) ที่มีความสัมพันธ์กันโดยการเลือกค่าจำนวนเฉพาะ(Prime) 2 ค่า p และ q (ขนาดประมาณ 256 bits) ซึ่งเราก็จะนำค่าทั้งสองนี้มาทำการหาค่า n โดย

$n = p*q$;

และค่า p และ q จะถูกเก็บเป็นความลับ

- การสร้างกุญแจที่เปิดเผย(Generate Public Key)

เลือกค่า e ที่เป็นความสัมพันธ์จำนวนเฉพาะ(Relatively Prime) กับ $\varphi(n)$ ซึ่ง

$\varphi(n) = (p-1)*(q-1)$;

และจะได้

Public Key = (e,n)

- การสร้างกุญแจที่ไม่เปิดเผย(Generate Private Key)

หาค่า d ซึ่งเป็น Multiplicative Inverse ของ e mod $\varphi(n)$ จะได้

$de = 1 \bmod \varphi(n)$

$de = 1 \bmod (p-1)*(q-1)$

การเข้ารหัส message m (<n) จะใช้ Public Key = (e,n) มาทำการเข้ารหัส โดยใช้

$$C = P^e \bmod n$$

และถูกถอดรหัสโดยใช้ Private key = (d,n) ดังนี้

$$P = C^d \bmod n$$

โดย ค่า P = Plaintext (ข้อมูล)

C = Ciphertext (ข้อมูลที่เข้ารหัสแล้ว)

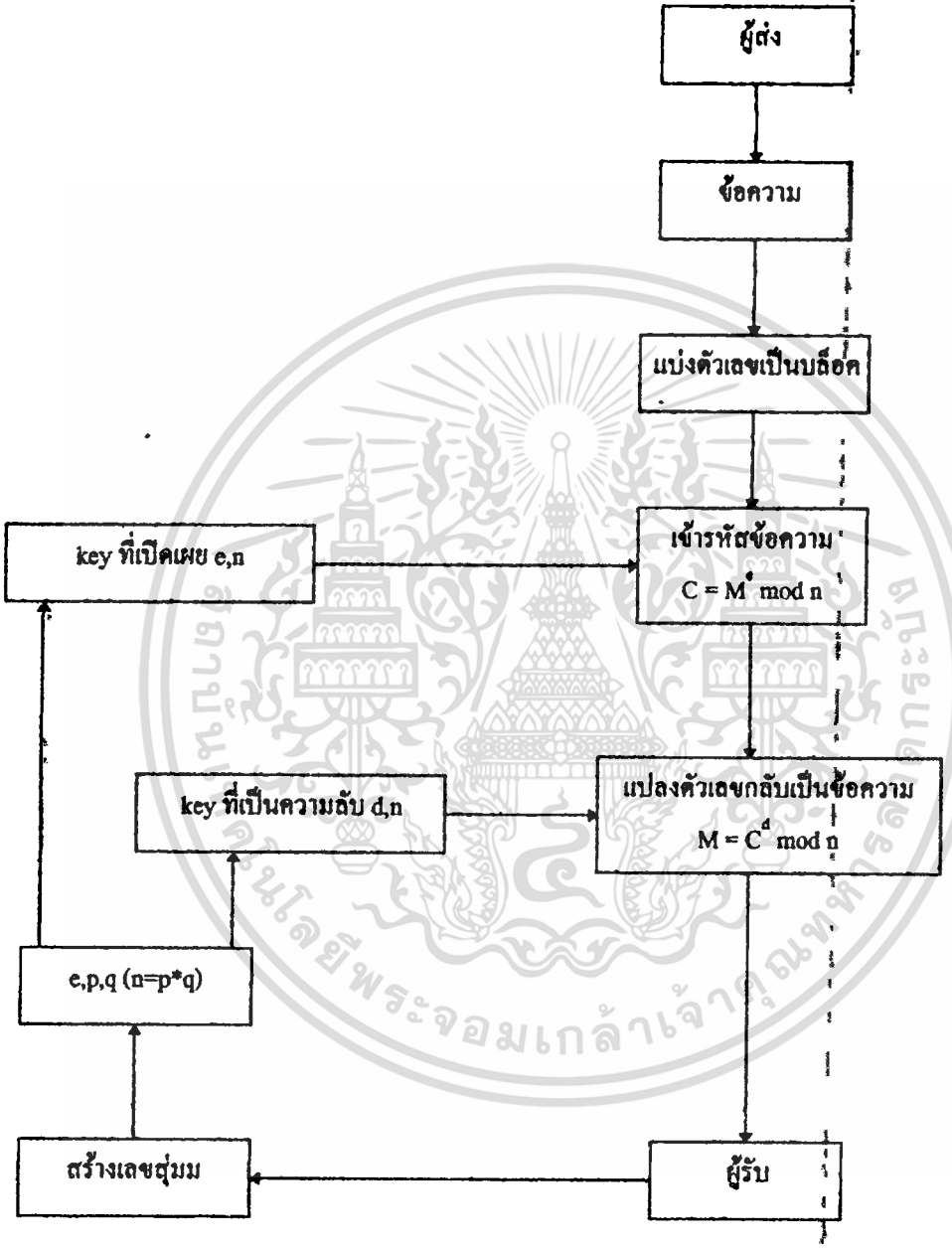
n = p*q ; p และ q เป็นค่า Prime

Public key = (e,n) โดยค่า e ต้องมี relatively prime to φ (n)

โดย $\varphi(n) = (p-1)(q-1)$

Private key = (d,n) โดยค่า d เป็น inverse ของ e mod $\varphi(n)$

รูปที่ 13 แสดงบล็อคไดอะแกรมการเข้ารหัสถอดรหัสของอาร์เอสเอ(RSA)



<u>อัลกอริทึมการหาค่าหารร่วมมาก</u>(Find gcd Algorithem)

เป็นโพซีเดอร์(Procedure) สำหรับตรวจสอบ ค่า gcd ของจำนวน 2 จำนวน :gcd (a,n);

1. ให้ a,b เป็นค่าที่ต้องการหาค่า gcd

2. หา x โดยให้ x = a div b (เป็นตัวคูณ)

3. หา b' จาก b'= a div b (เศษ)

4. จะได้ สมการ

a = (x*b)+b'

5. ตรวจสอบว่า เศษ (b') = 0 หรือไม่

- ถ้า เศษ(b') = 0 ให้ gcd(a,n) = b
- ถ้า เศษ(b') <> 0 ให้

a = b

b = b'

6. กลับไปทำข้อ 2.

<u>ตัวอย่าง</u>

การหาค่า gcd(3,615,807 , 2,763,323)

3,615,807 = (1) * 2,763,323 + 852,484

2,763,323 = (3) * 852,484 + 205,871

852,484 = (4) * 205,871 + 29,000

205,871 = (7) * 29,000 + 2,871

29,000 = (10) * 2,871 + 290

2,871 = (9) * 290 + 261

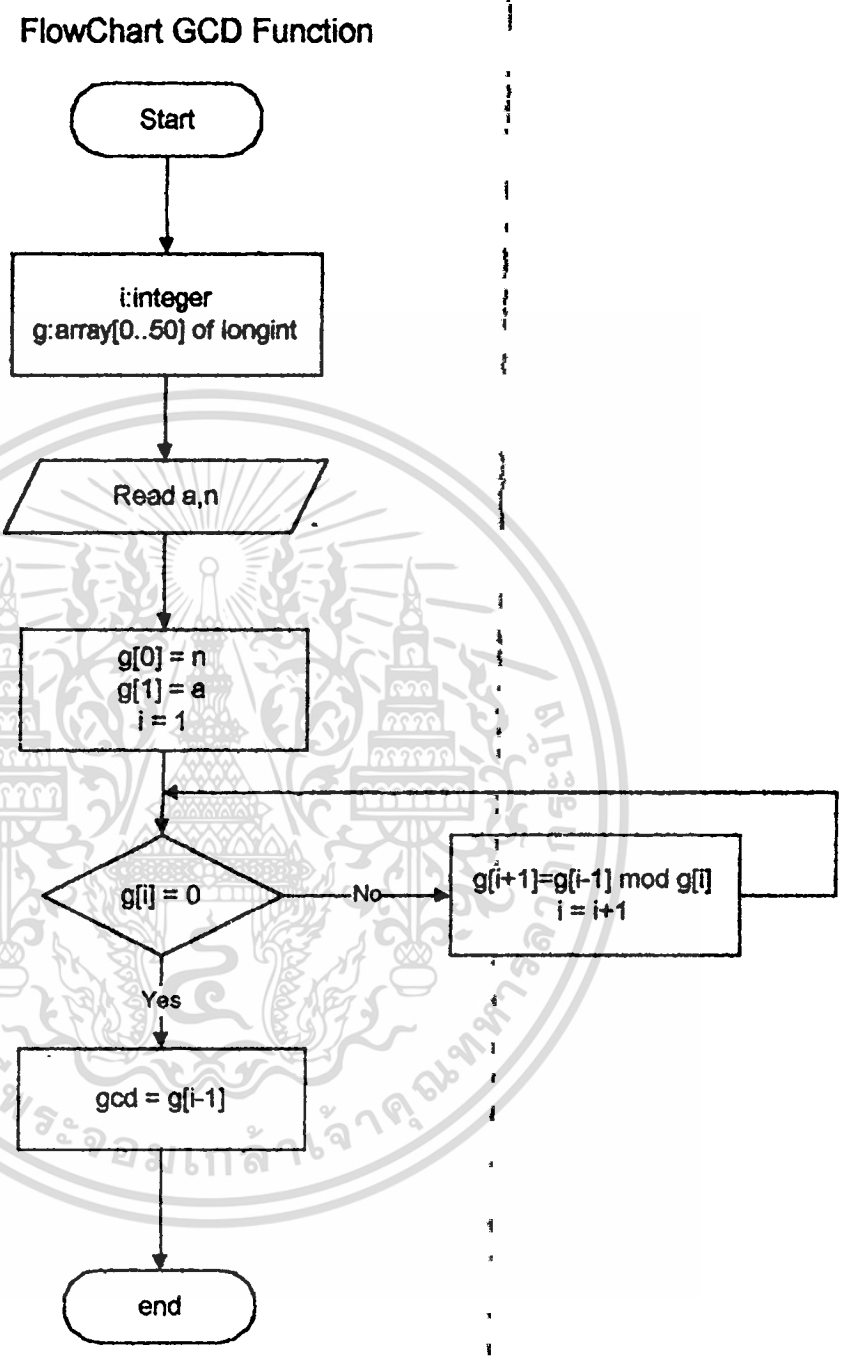
290 = (1) * 261 + 29

261 = (9) * 29 + 0

เราจะได้ค่า gcd(3,615,807 , 2,763,323) = 29

<u>หมายเหตุ</u> ค่า gcd(a,b) = gcd(b,a)

ค่าที่ใช้ต้องไม่เกินขอบเขตของ longint;



รูปที่ 14 แสดงโพล์ชาร์ทการหาค่าตัวหารร่วมมาก

<u>อัลกอริทึม(Algorithm) การหาค่ามอดยกกำลัง Fast Exponentiation</u> ($x=a^z \bmod n$)

1. กำหนดค่าเริ่มต้นให้ x = 1;
2. ตรวจสอบค่า z มีค่าเท่ากับ 0 หรือไม่
   - ถ้า z = 0 ให้ fastexp = x แล้วจบการทำงาน
   - ถ้า z <> 0 ให้ทำข้อถัดไป
3. ตรวจสอบค่า z เป็นจำนวนคู่หรือคี่
   3.1 ถ้าเป็นจำนวนคู่
      - ให้ z = z div 2
      - ให้ a = (a*a) mod n ----------(1)
   3.2 ถ้าเป็นจำนวนคี่
      - ให้ z = z-1
      - ให้ x = (x*a) mod n
4. ย้อนกลับไปทำที่ข้อ 2. จนกระทั่ง z=0 ก็จะได้คำตอบ fastexp

<u>ตัวอย่าง</u> การหาค่า $3^5 \bmod 7$

	3^2	จะได้ 3*3 mod 7 = 2
the result ของ	$(3^2)^2=3^4$	จะได้ 2*2 mod 7 = 4
คูณด้วย 3	$(3^4)*3=3^5$	จะได้ 4*3 mod 7 = 5

<u>หมายเหตุ</u> ปัญหาและข้อจำกัดของการหาค่ามอดยกกำลัง(algorithm fastexp(a,z,n))

เรากำหนดให้ทุกค่าที่ใช้ใน ฟังก์ชัน(function) fastexp(a,z,n) เป็นลองอิน (longint) ซึ่งลองอิน(longint) จะใช้หน่วยความจำขนาด 4 byte = 32 bit เป็นเลขจำนวนเต็มที่มีค่าตั้งแต่

-2,147,483,648 ถึง +2,147,483,647

ในกรณีที่เราให้ a = (a*a) mod n จาก สมการที่ (1)

การหาค่า fastexp(a,z,n)

ค่า a,n จะมีค่าสูงสุดได้ไม่เกิน 46,340 เนื่องจากว่า เมื่อ ค่า a = 46,340

จะทำให้

$a^2 = (46,340)^2 = 2,147,395,600$ จะไม่เกินขอบเขตของลองอิน (longint) แต่ ถ้า a = 46,341 จะทำให้

$a^2 = (46,341)^2 = 2,147,488,281$ จะทำให้เกินขอบเขตของลองอิน (longint) และในกรณีเดียวกัน ถ้า n เป็น 46,340 และจากสมการ (1) a = (a*a) mod n

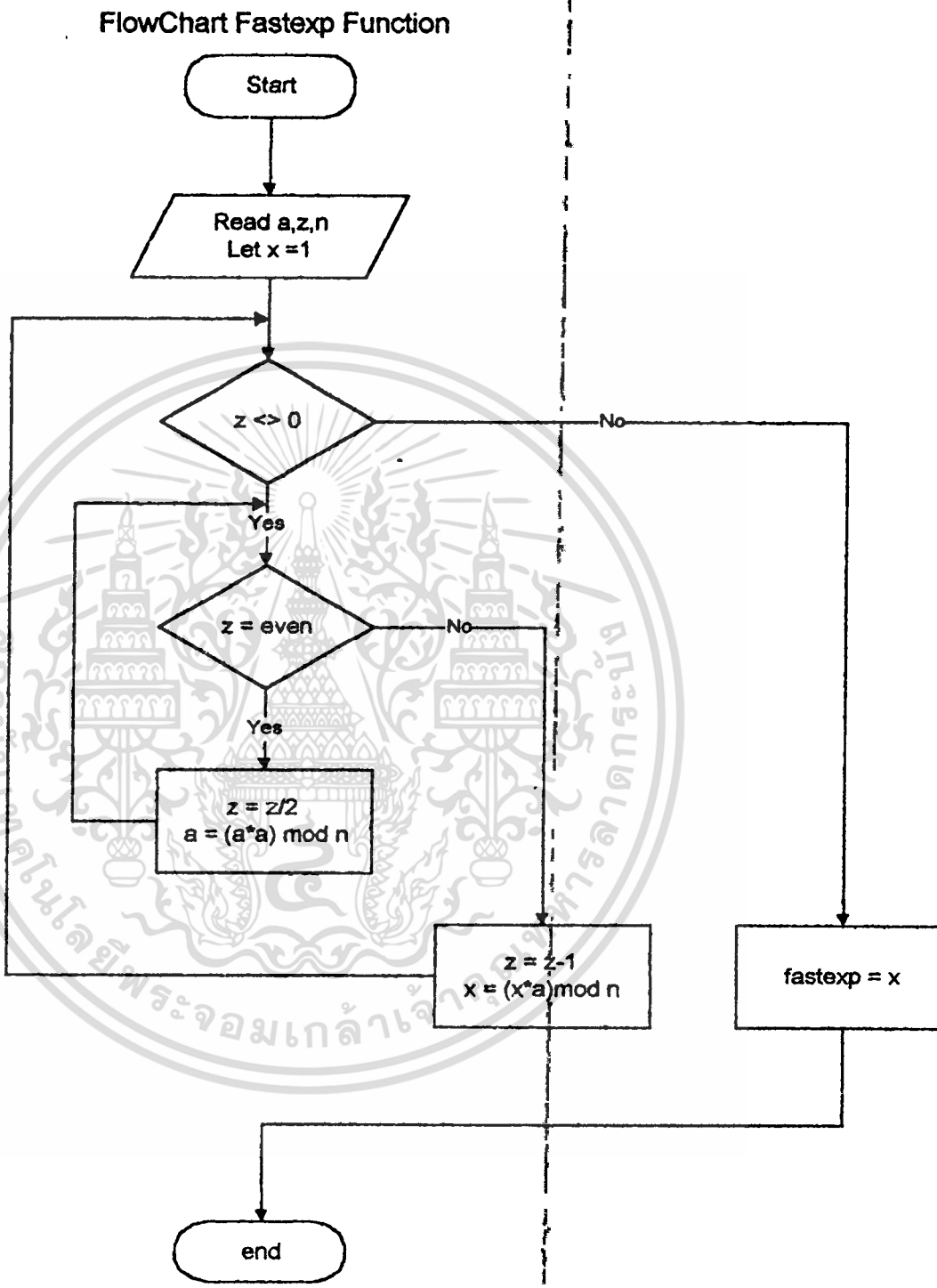
เมื่อจำนวนใด ๆ mod 46,340 จะทำให้ได้ค่า กระจายอยู่ ระหว่าง 0 ถึง 46,340 จากสมการ (1) จะทำให้ ค่า a อาจเท่ากับ 46,340 ซึ่งเมื่อยกกำลังสองจะทำให้ค่าไม่เกินขอบ เขตของลองอิน(longint) แต่เมื่อค่า n มากกว่า 46,340 และเมื่อ ทำการ mod n ก็อาจจะทำให้ค่า a เกินค่า 46,340 ดังนั้น เมื่อเข้าสมการแล้ว และ a ยกกำลังสอง ก็จะทำให้ ค่าที่ได้เกินขอบเขตของลองอิน(longint)

ทั้งสองกรณีนี้ ก็จะทำให้เกิดข้อผิดพลาดในการเข้ารหัสถอดรหัสซึ่งในโปรแกรม(Program) ที่เขียนขึ้น ค่า a จะไม่เกินค่า 46,340 เนื่องจากว่าเราแปลงตัวอักษรให้เป็นรหัสแอสกี้(ASCII) ดังนั้นค่า a จะอยู่ระหว่าง 0 - 255

ดังนั้นเราจึงจะต้องควบคุมค่า n ไม่ให้เกิน 46,340 ซึ่ง

$p*q = n <= 46340$

เพราะฉนั้น การสุ่ม(random) ค่าจำนวนเฉพาะ(prime) p,q จะต้องตรวจสอบทุกครั้ง



รูปที่ 15 แสดงโฟล์ชาร์ทการหาค่าการยกกำลังแล้วมอด

<u>อัลกอริทึม(Algorithm) การหาค่าอินเวอร์ส(Inverse ;inv(a,n)</u>

การหาค่าอินเวอร์ส(inverse) ของ a mod n ; โดยที่ 0<a<n

จะต้องตรวจสอบว่า gcd(a,n) = 1 หรือไม่ ถ้าใช่จะสามารถหาค่า inverse ได้ แต่ถ้าไม่ใช่ค่า a mod n จะไม่มี inverse

เริ่มต้นให้ $g_0 = n$,$g_1 = a$

$u_0 = 1$,$u_1 = 0$

$v_0 = 0$,$v_1 = 1$

ที่ step i ให้ $y = g_i\text{-}1 \text{ div } g_i$

$g_i+1 = g_i\text{-}1\text{-}(y*g_i)$

$u_i+1 = u_i\text{-}1\text{-}(y*u_i)$

$v_i+1 = vi\text{-}1\text{-}(y*v_i)$

$i = i+1$

- เช็คว่า $(u_i*n)+(v_i*a) = 1$ หรือไม่
  - ถ้าเท่ากับ 1 แสดงว่า v_i เป็นอินเวอร์ส(inverse)
    - ถ้า v_i เป็น ค่าบวกก็จะได้ค่า v_i เป็นอินเวอร์ส(inverse)
    - ถ้า v_i เป็น ค่าลบก็จะได้ inverse = n+v_i
  - ถ้าไม่เท่ากับ 1 ให้กลับไปทำที่ step i ถัดไป

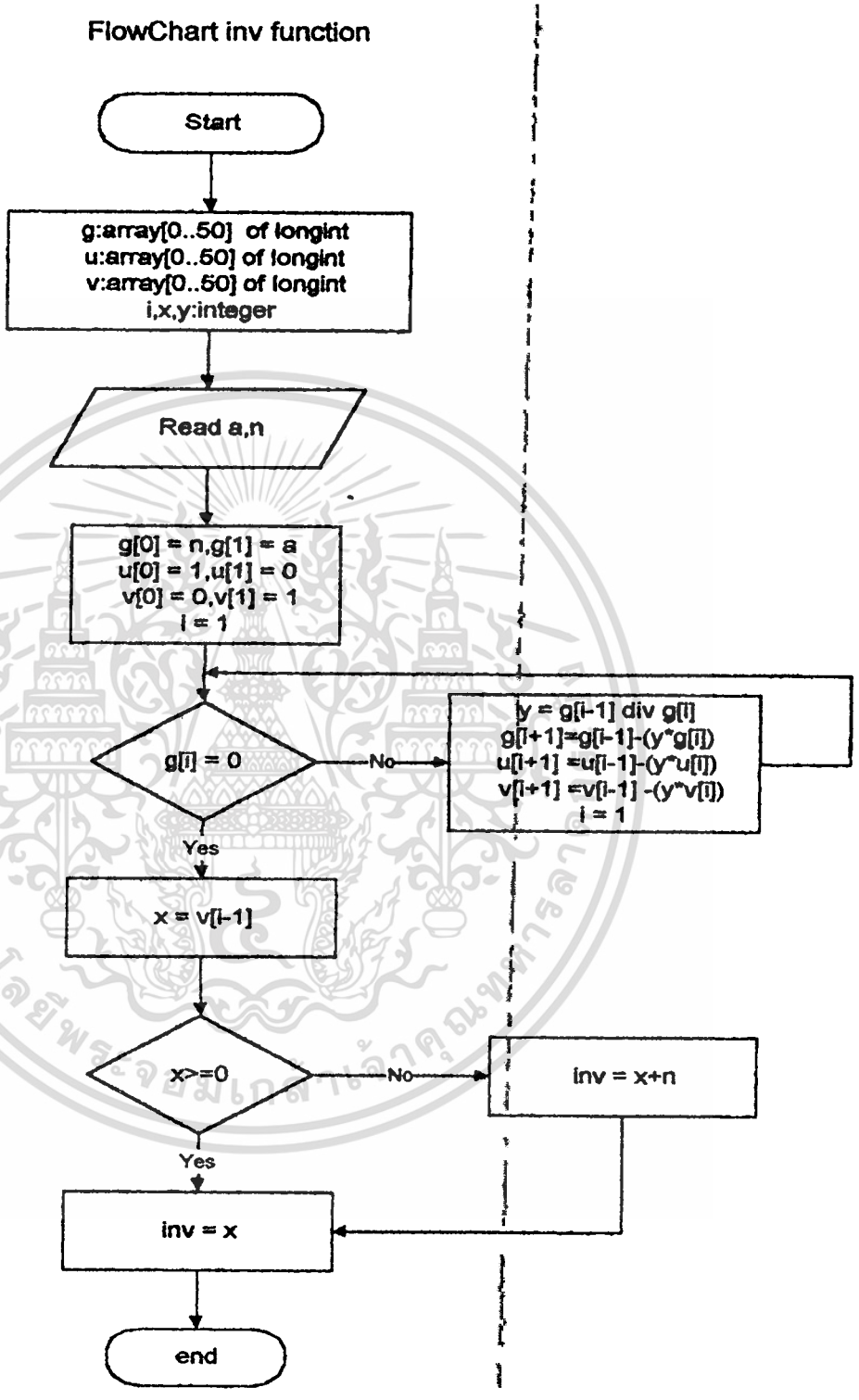
<u>ตัวอย่าง</u> ให้หา inv ของ 3 mod 7 ; a = 3,u = 7

i	g_i	u_i	v_i	y
0	7	1	0	
1	3	0	1	2
2	1	1	-2	3
3	0			

ที่ i = 2; $(u_2*n)+(v_2*a) = (1*7)+((-2)+3) = 7+(-6) = 1$

ดังนั้น v2 =(-2);

inverse = 7+(-2) = 5



รูปที่ 16 แสดงโฟล์ชาร์ทการหาค่าอินเวอร์สของการมอด

อัลกอริทึม(Algorithm)การหาค่าจำนวนเฉพาะ(Prime)

1. สุ่ม(random) ตัวเลขขึ้นมาค่าหนึ่งให้เท่ากับ p

2. หาค่ารากที่สองของ p

3. นำค่า p หารด้วย ทุกจำนวนที่มีค่าน้อยกว่าหรือเท่ากับ $\lfloor\sqrt{p}\rfloor$ (ใช้จำนวนเต็มหาร)

4. ถ้าผลจากการหารทุกค่าไม่ลงตัวแสดงว่า ค่า p นี้มีโอกาสเป็นไปได้ที่จะเป็นค่าจำนวนเฉพาะ(prime) สูง แต่ในกรณีที่มีตัวใดตัวหนึ่งหารลงตัวก็แสดงว่า p ไม่ใช่ค่าจำนวนเฉพาะ(prime)

5. จากข้อ 4 ถ้าไม่ใช่ค่าจำนวนเฉพาะ(prime) ให้ เริ่มทำข้อ 1 ใหม่แต่ถ้ามีโอกาสเป็นจำนวนเฉพาะ(prime) สูงก็ให้แสดงค่า p นั้น เท่ากับจำนวนเฉพาะ (prime)

ตัวอย่างที่ 1

สุ่ม(random);

p = 16 , $\sqrt{p}$ = 4

16 mod (all number <=4) ; ไม่คิดค่า 1

16 mod 2 = 0

16 mod 3 = 1

16 mod 4 = 0 ดังนั้น 16 ไม่ใช่ค่าจำนวนเฉพาะ(prime)

ตัวอย่างที่ 2

สุ่ม(random);

p = 19 , $\sqrt{p}$ = 4.358898

19 mod (all number <=4) ; ไม่คิดค่า 1

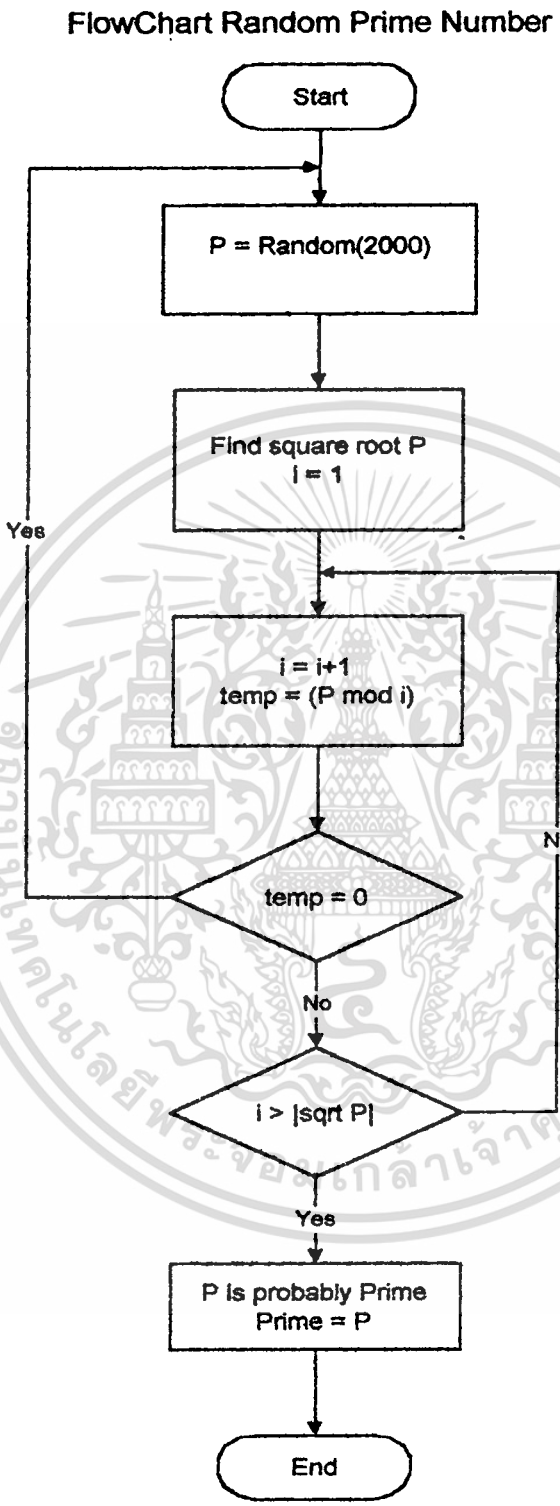
19 mod 2 = 1

19 mod 3 = 1

19 mod 4 = 3 ดังนั้น 19 จะเป็นค่าจำนวนเฉพาะ(prime)

หมายเหตุ

- ค่าที่ได้มีโอกาสเป็นค่า prime สูง แต่ก็แต่ก็ยังไม่แน่นอนที่สุด
- ขอบเขตของ function random( );เราสามารถ random ได้ค่าสูงที่สุดคือ 65,535



รูปที่ 17 แสดงโพล์ชาร์ทการหาค่าจำนวนเฉพาะ

<u>อัลกอริทึม(Algorithm) การเข้ารหัส</u>

1. ทำการแปลงตัวอักษร(charactor) ให้เป็นแอสกี้(Ascii) ซึ่งจะได้ค่าตัวเลขประมาณช่วง 33 ถึง 255 (ช่วงของตัวอักษร) โดยกำหนดให้เท่ากับ M

2. เข้ารหัสโดย

$$M^e \bmod n = C$$

3. ทำการแปลงกลับให้เป็นค่าแอสกี้(Ascii) ที่อยู่ในช่วงตัวอักษร เพื่อแสดงผล

ขั้นตอนการแปลง

- ถ้า (C mod 256)>220 ;ตรวจสอบถ้าบวกด้วย 33 จะเกิน 256 ทำเช่นนี้ เพราะว่าไม่ให้ ค่า ascii ที่ได้ อยู่ในช่วง 0 ถึง 32 ซึ่งค่าช่วงนี้ จะใช้เป็นส่วนจัดการไฟล์หรือใช้ในส่วนอื่น เช่น จบแฟ้มข้อมูล(eof), ขึ้นบรรทัดใหม่ เป็นต้น

จากนั้นให้

temp1 = C div 256 ;เก็บตัวคูณกับ 256

temp1 = temp1+33 ;div แล้วจะให้ค่าน้อย ∴ บวก 33 เพื่อให้อยู่ในช่วง 33 - 256 ,temp1 จะเป็นอักษรตัวที่ 2

temp2 = C mod 256 ;เศษของการ mod อยู่ในช่วง 220 ขึ้นแต่ไม่เกิน 256, temp2 จะเป็นอักษรตัวที่ 3

ใส่ค่ารหัส chr(33) เป็นอักษรตัวแรก เพื่อบอกว่า (C mod 256)>220

ดังนั้น 1 ตัวอักษร(charactor) แปลงเป็นแอสกี้(ascii) = M เข้ารหัสแล้วจะได้ C แปลงเป็นตัวอักษร(charactor) ได้ 3 ตัวอักษรคือ $c_1c_2c_3$

c_1 = ตัวบอกว่าค่า c_2c_3 เมื่อแปลงเป็น C แล้ว mod 256 >220

c_2 = ตัวคูณกับ 256

c_3 = เศษสำหรับบวก

∴ จะหา C จาก $C = ((c_2-33)*256)+c_3)$

- ถ้า (C mod 256)<220 ;หมายถึง +33 จะไม่เกิน 256

temp1 = C div 256

temp1 = temp1+34 ;กรณีนี้ +34 ให้ต่างจากกรณีที่แล้ว

temp2 = C mod 256

temp2 = temp2+34

กรณีนี้ 1 อักษร(charactor) แปลงเป็นแอสกี้(ascii) = M เข้ารหัสแล้วจะได้ C แปลงเป็น อักษร(charactor) จะได้

2 ตัวอักษร(charactor) คือ c_1c_2

c_1 = ตัวคูณกับ 256

c_2 = เศษสำหรับบวก

∴ จะหา C จาก $C = ((c_1-34)*256)+(c_2-34))$

4. จากนั้นแปลงค่า c ต่าง ๆให้เป็น ตัวอักษร(charactor)

5. กลับไปทำข้อ 1 ทำการแปลงตัวอักษร(charactor) ตัวถัดไปจนกระทั่งหมดข้อความที่จะเข้ารหัส

<u>อักอริทึม(Algorithm) การถอดรหัส</u>

1. แปลงตัวอักษร(charactor) ให้เป็นแอสกี้(ascii) เพื่อหาค่า C

2. ตรวจสอบครั้งละ 2 ค่าดูว่าค่าแรกว่า เป็น 33 หรือไม่

- ถ้าใช่แสดงว่าแอสกี้(ascii) อีก 2 ตัวถัดไปจะเป็นข้อมูลในการหา C โดยให้ค่าแอสกี้(ascii) อีก 2 ตัวที่ต่อจากค่าแอสกี้(ascii) 33 เป็น c_1 และ c_2

หาค่า C โดย

$C = ((c_1-33)*256)+c_2)$

- ถ้าไม่ใช่แสดงว่าแอสกี้(ascii) ตัวนี้และตัวถัดไปเป็นข้อมูลในการหา C โดยให้ค่าแอสกี้(ascii) ทั้งสองตัวเป็น c_1 และ c_2 ตามลำดับ

หาค่า C โดย

$C = ((c_1-34)*256)+(c_2-34)$

3. หลังจากได้ค่า C แล้วก็จะทำการถอดรหัสเพื่อหา M โดย

$$M = C^{d} \bmod n$$

4. เมื่อได้ค่า M แล้วก็ทำการแปลงค่า M กลับเป็นตัวอักษร ก็จะได้ค่าตัวอักษร ตัวเดิม

3.2 ส่วนที่ทำงานกับฐานข้อมูล

ในส่วนนี้จะทำการกำหนดข้อมูลสำหรับแอพพลิเคชั่น(Application) โดย จะทำการสร้าง Table สำหรับเก็บประวัติบุคคลทั่วไป โดยอาจจะนำไปประยุกต์ใช้ในการเก็บประวัติพนักงานของบริษัท แล้วอาจจะมีการเข้ารหัสประวัติของบุคคล นั้นไว้เพื่อไม่ให้ผู้ที่ไม่เกี่ยวข้องสามารถทราบข้อมูลในส่วนนี้ได้ เช่นอาจจะมีบริษัทคู่แข่งที่ต้องการทราบข้อมูลในส่วนนี้ ซึ่งถ้ามีการเข้ารหัสก็จะไม่สามารถนำไปใช้ได้

กำหนดข้อมูลประวัติที่ใช้เก็บข้อมูล

- First Name
- Last Name
- Company
- Address1
- Address2
- City
- State
- Zip
- Home Phone
- Work Phone
- Fax
- Email1
- Email2
- Comment
- Category

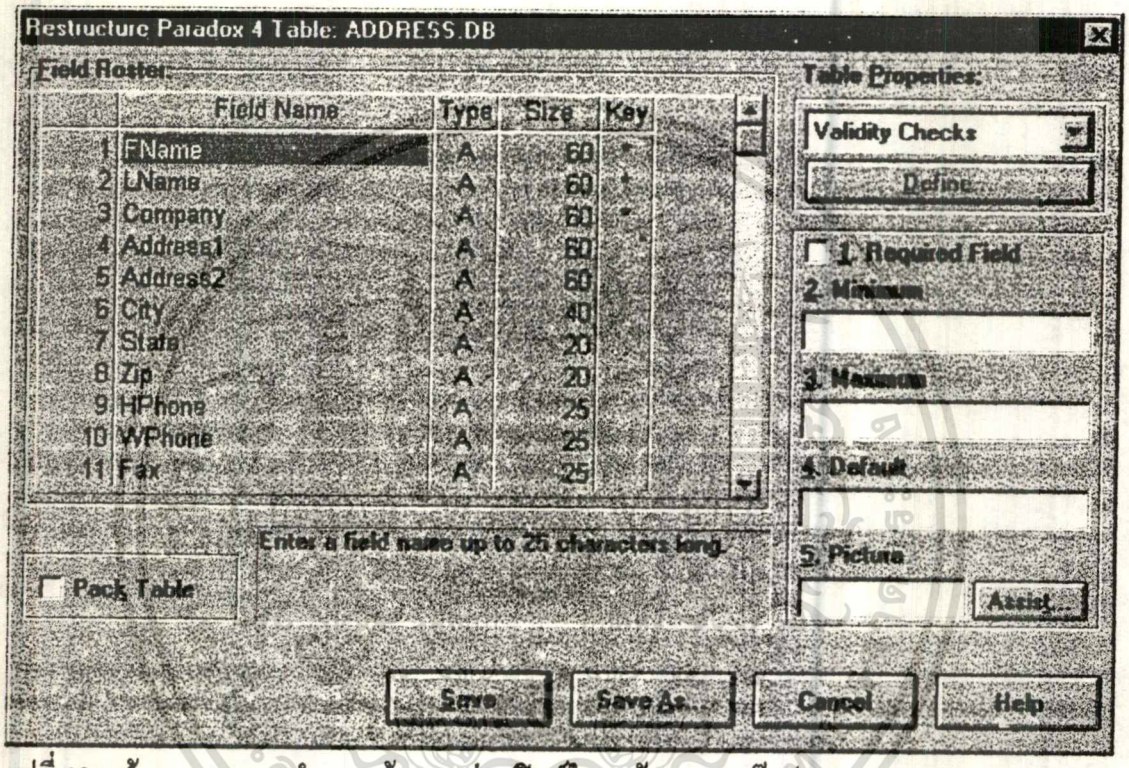
กำหนดฟิลด์(Fields) ที่ใช้ในโปรแกรม

Name	Type	Size	PIdx	Index
FName	Character	60	*	FNameIndex
LName	Character	60	*	LNameIndex
Company	Character	60	*	CompanyIndex
Address1	Character	60		
Address2	Character	60		
City	Character	40		
State	Character	20		
Zip	Character	20		
HPhone	Character	25		
WPhone	Character	25		
Fax	Character	25		
EMail1	Character	60		
EMail2	Character	60		
Comment	Character	254		
Category	Character	25		

ตารางที่ 11 แสดงการกำหนดค่าฟิลด์ต่าง ๆ ภายในตาราง

สร้างตารางตามลักษณะข้อมูลที่กำหนด

ใช้ คาต้าเบสเดสท็อป(Databast Desktop) สร้างตาราง(Table) โดยกำหนดฟิลด์ (Field) ต่าง ๆ ตามที่กำหนดไว้ โดยให้ชื่อตาราง(Table) นี้ว่า Address.DB โดยจะมีลักษณะดังนี้



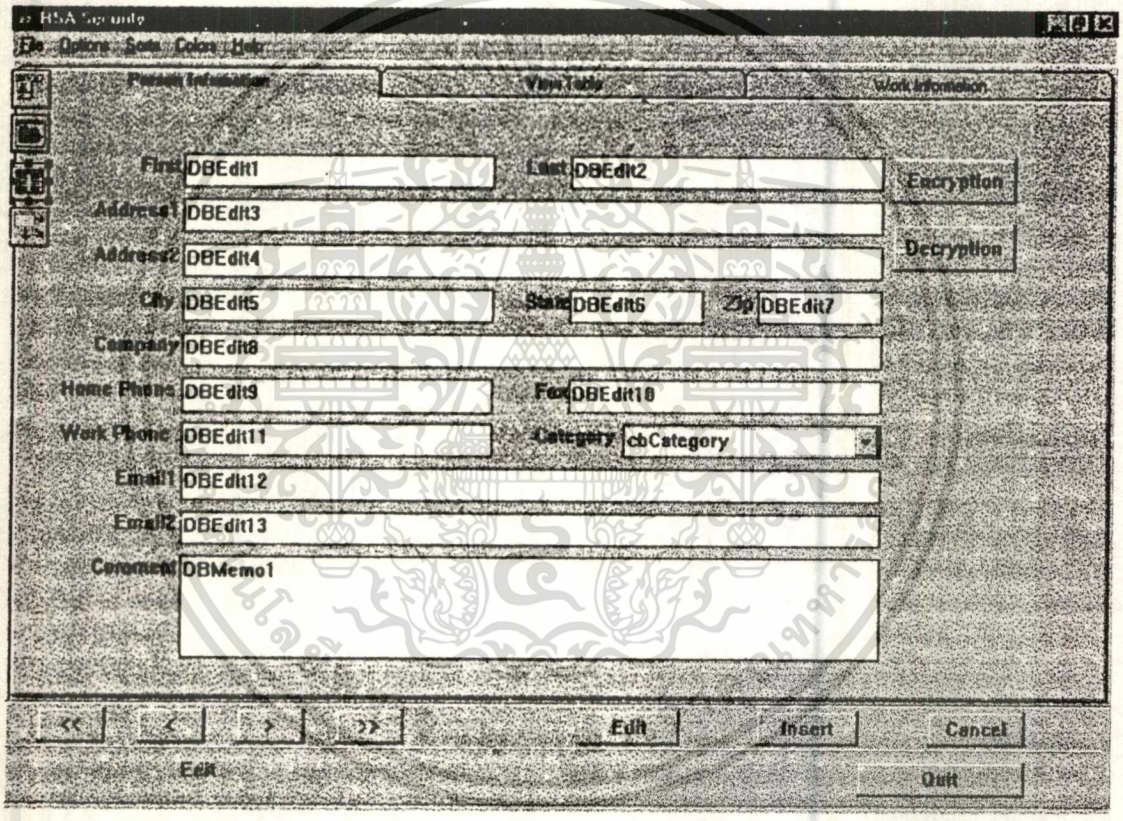
รูปที่ 18 หน้าจอแสดงการกำหนดข้อมูลแต่ละฟิลด์ในคาต้าเบสเดสท็อป

หลังจากนั้นทำการกำหนดเซคคั่นดารี่ อินเด็กซ์(Secondary index) คือ FNameIndex,LNameIndex และCompanyIndex เพื่อใช้ในการ ค้นหา(Search),เรียงลำดับ (Sort)

3.3 การออกแบบหน้าจอ (Front - end)

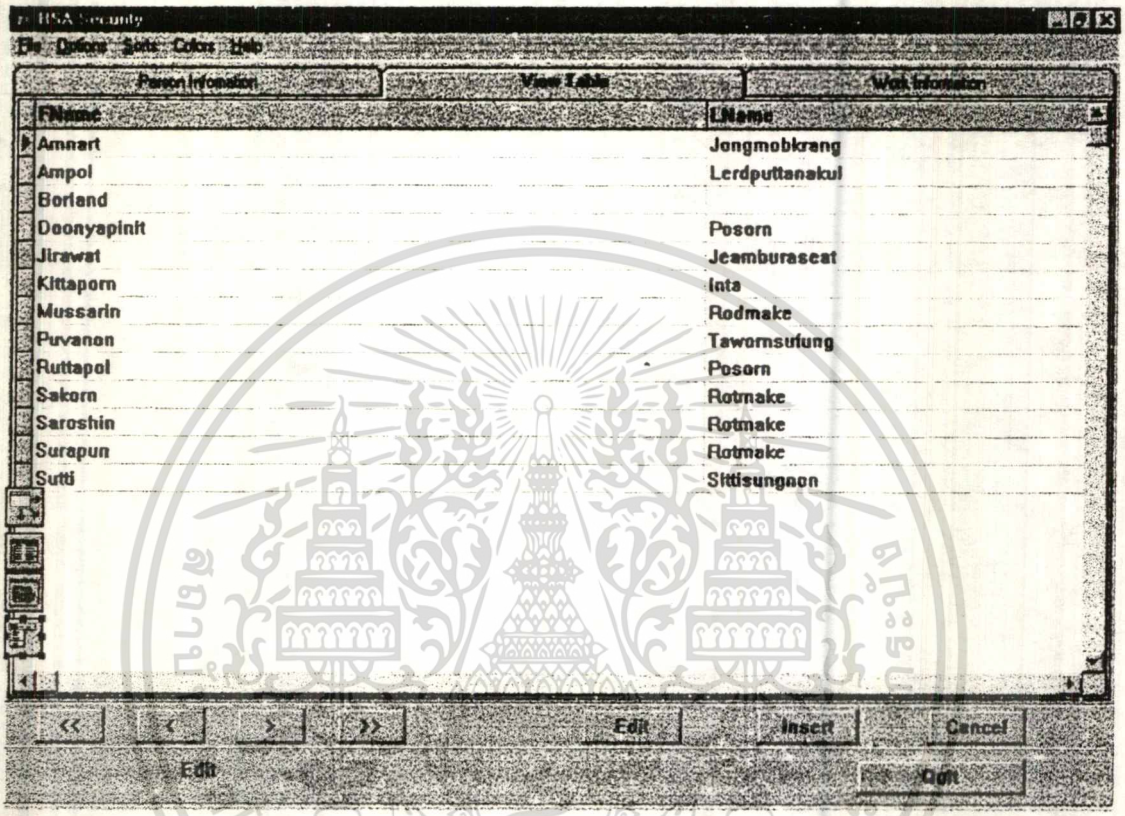
การออกแบบหน้าจอที่ดีจะต้องมีลักษณะที่ไม่ซับซ้อนเกินไปเพราะจะทำให้ผู้ใช้สับสนและอาจใช้งานไม่ได้ตามวัตถุประสงค์

การออกแบบหน้าจอของของโปรแกรมอาร์เอสเอเซอร์คูริตี้(Program RSA Security) สำหรับการแสดงข้อมูล,การป้อนข้อมูลสำหรับการเพิ่ม(Insert),ลบ(Delete), แก้ไข(Updata) และแสดงข้อมูลที่เข้ารหัสถอดรหัส



รูปที่ 19 หน้าจอที่ออกแบบสำหรับติดต่อกับผู้ใช้

หน้าจอแสดงตาราง(Table) Address.DB จะแสดงข้อมูลที่มีอยู่ในแฟ้มข้อมูลทั้งหมดทั้งที่เข้ารหัสและไม่ได้เข้ารหัส ซึ่งผู้ใช้(User) สามารถเรียกดูข้อมูลได้



รูปที่ 20 หน้าจอที่ออกแบบสำหรับแสดงส่วนที่ใช้ในการดูข้อมูลทั้งหมด

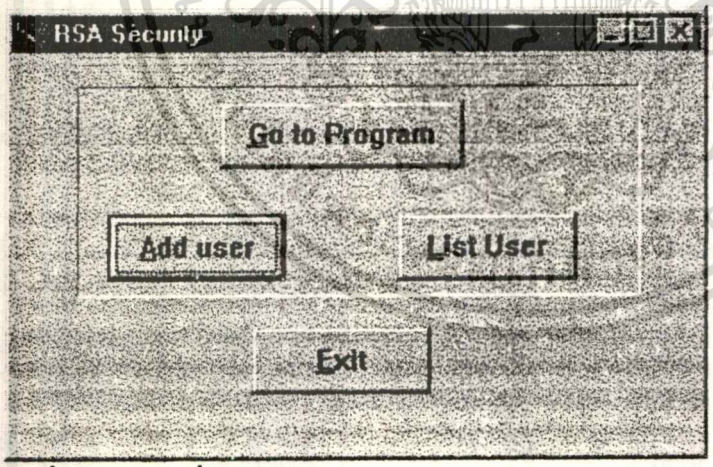
บทที่ 4

การใช้งานและผลการทดลอง

ในบทนี้จะเป็นการกล่าวถึงการทำงานของระบบ ซึ่งเป็นการนำเอาผลที่ได้จากการออกแบบและวิเคราะห์มาทำการเขียนโปรแกรม โดยจะได้ทำการอธิบายไปพร้อมกับแสดงหน้าจอหลักบางส่วนเพื่อให้ได้เห็นภาพรวมของระบบมากขึ้น

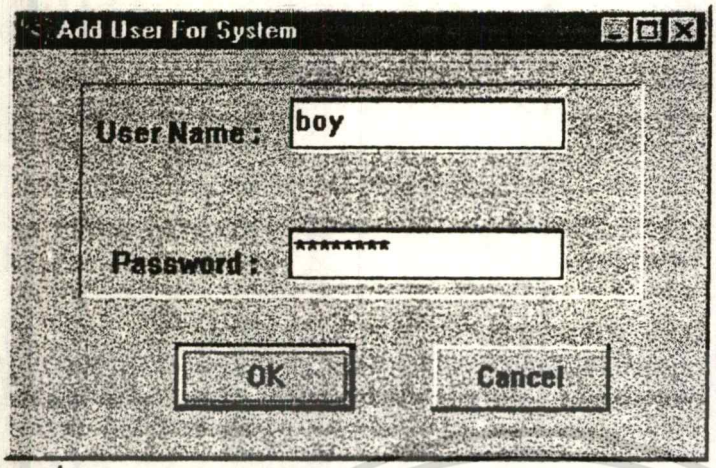
การทำงานของระบบการเข้ารหัสอาร์เอสเอ(RSA) ได้แบ่งออกเป็นสองส่วนคือ การเข้ารหัสถอดรหัสเท็กซ์ไฟล์(Text File) และ การเข้ารหัสถอดรหัสประวัติของบุคคล(AddressBook) แต่ก่อนที่จะทำงานในส่วนอื่น ๆ นั้นจะต้องมีการสร้างผู้ที่มีสิทธิในการเข้าไปใช้งานโปรแกรม

4.1 การสร้างรหัสกุญแจ(Private Key และ Public Key) โดยเราจะมีการสร้างกุญแจที่ไม่เปิดเผย(Private Key) และกุญแจที่ไม่เปิดเผย(Public Key) ให้กับผู้ที่จะใช้งานโปรแกรมนี้ ดังรูป



รูปที่ 21 หน้าจอเริ่มต้นสำหรับโปรแกรม

ทำการคลิกที่ปุ่มแอดยูเซอร์(Add user) หลังจากนั้นโปรแกรมจะแสดงหน้าจอให้เราทำการป้อนยูเซอร์เนม(User name) และรหัสผ่าน(Password) ซึ่งควรจะมากกว่า 6 ตัวอักษรแล้ว คลิก(Click) ที่ปุ่มโอเค(OK)



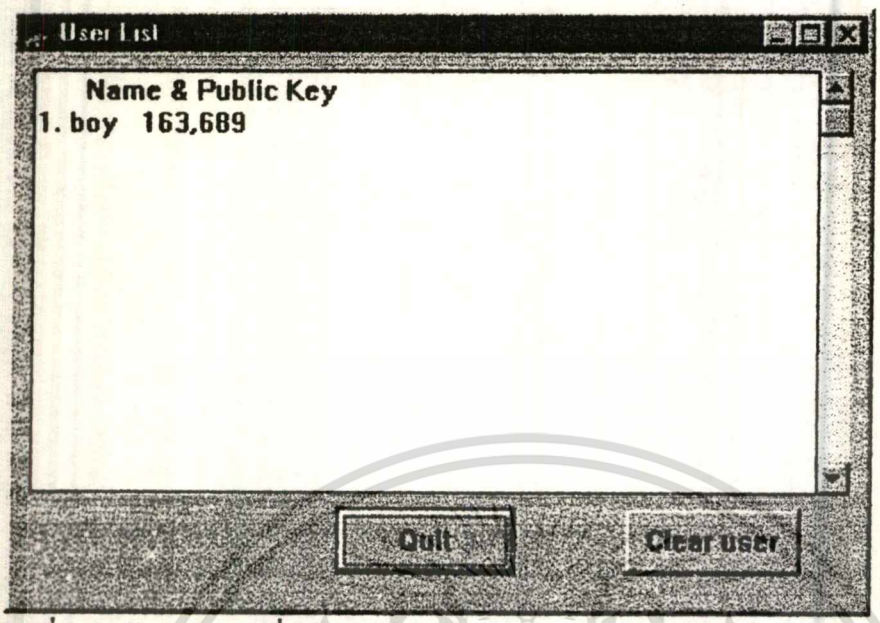
รูปที่ 22 หน้าจอสำหรับการสร้างรหัสผ่านและยูเซอร์เนมของผู้ใช้

จากนั้นโปรแกรมจะทำการหาค่า จำนวนเฉพาะ(Prime) p,q,e ซึ่งจะนำไปใช้คำนวณหาค่า n โดยใช้ p*q จะได้ค่า n ออกมาต่อจากนั้น โปรแกรมจะทำการหาค่า $\phi(n)$ จาก (p-1)*(q-1) เมื่อได้ค่า $\phi(n)$ แล้วก็จะทำการหาค่า d ซึ่งเป็นอินเวอร์ส(inverse) ของ e mod $\phi(n)$ จากนั้นเราก็จะได้

e,n = Public Key

และ d,n = Private Key

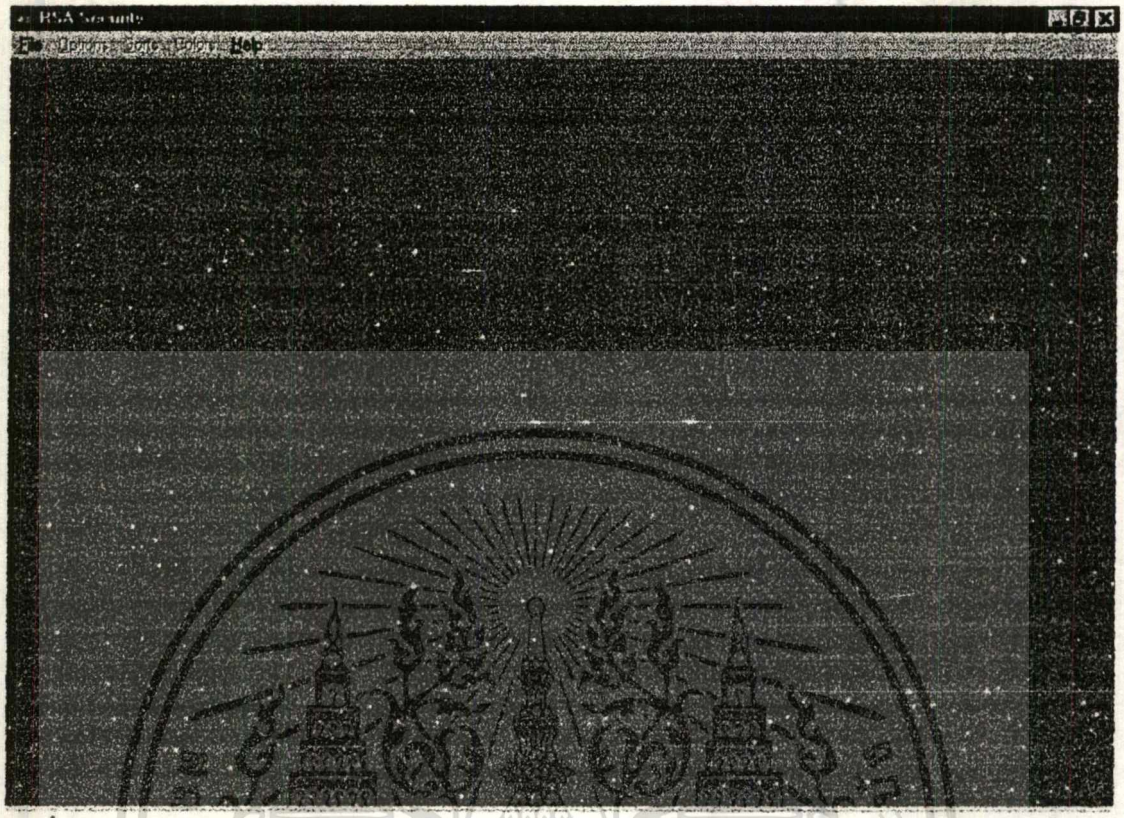
เมื่อได้กุญแจที่เปิดเผย(Public Key) และกุญแจที่ไม่เปิดเผย(Private Key) แล้ว โปรแกรมก็จะกลับมาที่เมนูหลัก(Main) และถ้าคลิก(Click) ที่ลิสยูสเซอร์(List User) ของเมน ฟอร์ม(Main Form) ก็จะมีชื่อที่เราป้อนเข้าไปซึ่งเราจะใช้ชื่อนี้เป็นยูสเซอร์ (User Name) ส่วนตัวเลขข้างหลังจะเป็น ค่า กุญแจที่เปิดเผย(Public Key) (e,n) แสดงอยู่ ดังรูปที่ 23



รูปที่ 23 หน้าจอแสดงผู้ที่มีสิทธิใช้โปรแกรม

หลังจากมียูสเซอร์เนม(User Name) และรหัสผ่าน(Password) แล้วให้คลิก(Click) ที่ปุ่มโกทูโปรแกรม(Go to Program) เพื่อเข้าสู่โปรแกรมโดยป้อนยูสเซอร์เนม(User Name) และรหัสผ่าน(Password) ของเราที่สร้างไว้เพื่อเข้าสู่โปรแกรม

หลังจากเข้าสู่โปรแกรมแล้วจะมีหน้าจอแสดงดังรูปที่ 24



รูปที่ 24 หน้าจอหลังจากผ่านเข้าสู่โปรแกรมแล้ว

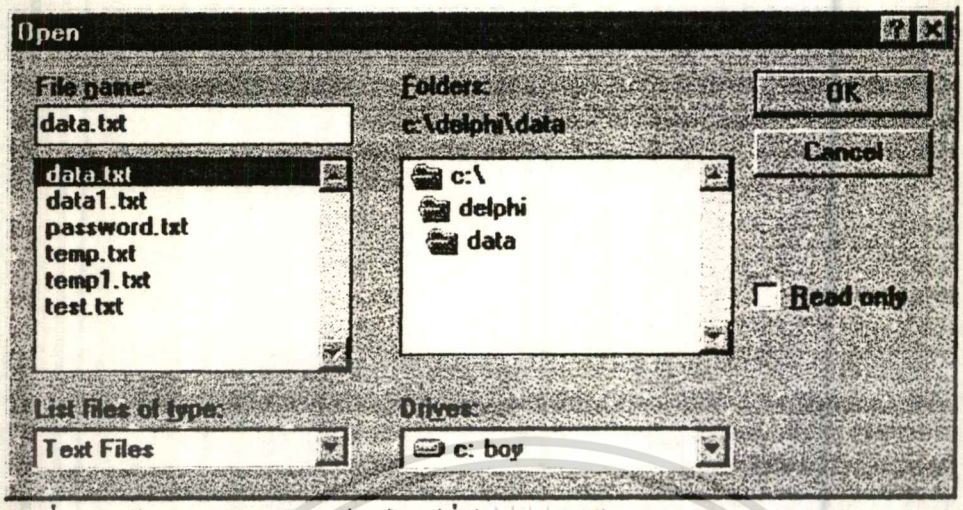
4.2 การทำงานของโปรแกรม

ซึ่งการทำงานจะแบ่งออกเป็น 2 ส่วน คือ

4.2.1 การเข้ารหัส Text File ซึ่งในบางครั้งเรามีความจำเป็นเก็บข้อมูลบางอย่างไว้เป็นความลับโดยที่ไม่ยอมให้ผู้อื่นเข้ามานำข้อมูลในส่วนนี้ไปใช้งานได้เราจึงต้องมีการเข้ารหัสข้อมูลในส่วนนี้ไว้เพื่อเป็นการป้องกัน

ขั้นตอนในการใช้งาน

คลิก(Click) ที่ File | Open ที่ เมนู(Menu) เพื่อทำการเลือกไฟล์(File) ที่ต้องการเข้ารหัสข้อมูลโปรแกรมจะแสดงไดอะล็อก(Dialogs) ดังรูปที่ 25



รูปที่ 25 หน้าจอสำหรับเลือกเท็กซ์ไฟล์ที่ต้องการจะเข้ารหัส,ถอดรหัส

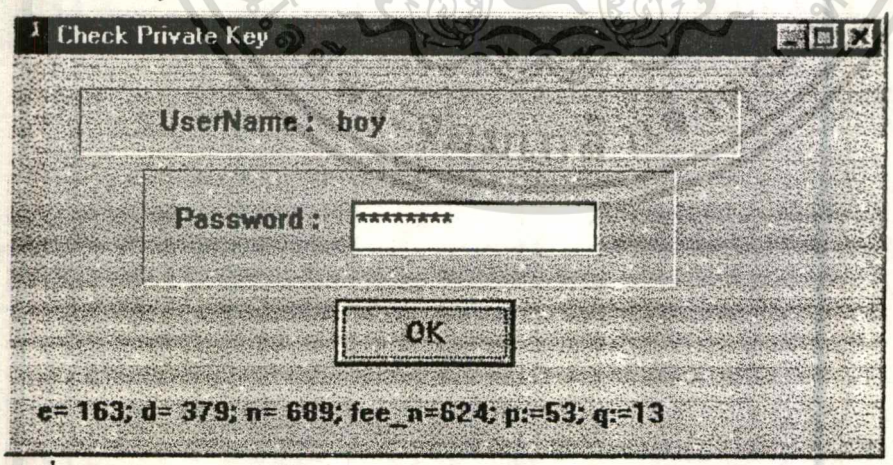
หลังจากทำการเลือกไฟล์(File) ที่จะเข้ารหัสแล้ว(ในที่นี้คือ Data.txt) โปรแกรมก็จะทำการเปิดไฟล์(File) Data.txt และให้เลือก Security | Encryption ที่ เมนู(Menu) เพื่อทำการเข้ารหัสไฟล์(File) ที่เปิดขึ้น โปรแกรมจะให้ใส่รหัสผ่าน(Password ของ User) ก่อนการเข้ารหัส ซึ่งผู้ใช้(User) ต้องป้อนให้ถูกต้องไม่เช่นนั้นโปรแกรมจะไม่เข้ารหัสข้อมูลให้ จะแสดงดังรูป

ก่อน คลิก(Click) Security | Encryption ที่เมนู(Menu) แสดงดังรูปที่ 26

```
unit Prime;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    quit: TButton;
    Buttom2: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Bevel3: TBevel;
    procedure quitClick(Sender: TObject);
    procedure Buttom2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}
```

รูปที่ 26 หน้าจอแสดงไฟล์ที่เปิดขึ้นมาเพื่อจะทำการเข้ารหัสหรือถอดรหัส

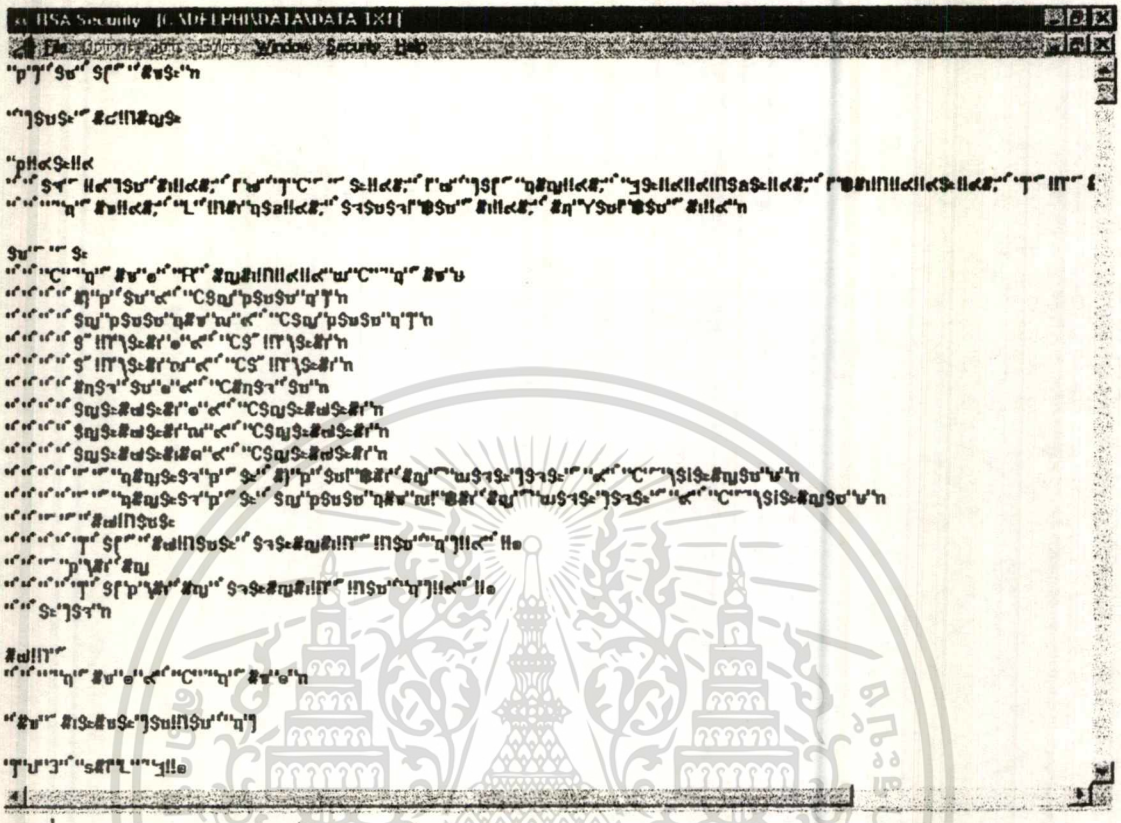
หลังจาก คลิก(Click) Security | Encryption ที่เมนู(Menu) จะแสดงหน้าจอให้ป้อนรหัสดังรูป



รูปที่ 27 หน้าจอแสดงให้ป้อนรหัสก่อนการเข้ารหัส,ถอดรหัส

เมื่อป้อนรหัสผ่าน(Password) ถูกต้องโปรแกรมจะทำการเข้ารหัสไฟล์ให้ แต่ถ้าไม่ถูกต้องจะกลับไปที่หน้าจอที่แล้วเพื่อให้ทำการคลิกใหม่

หลังจากที่ทำการเข้ารหัสแล้วจะได้ File data.txt ดังรูป

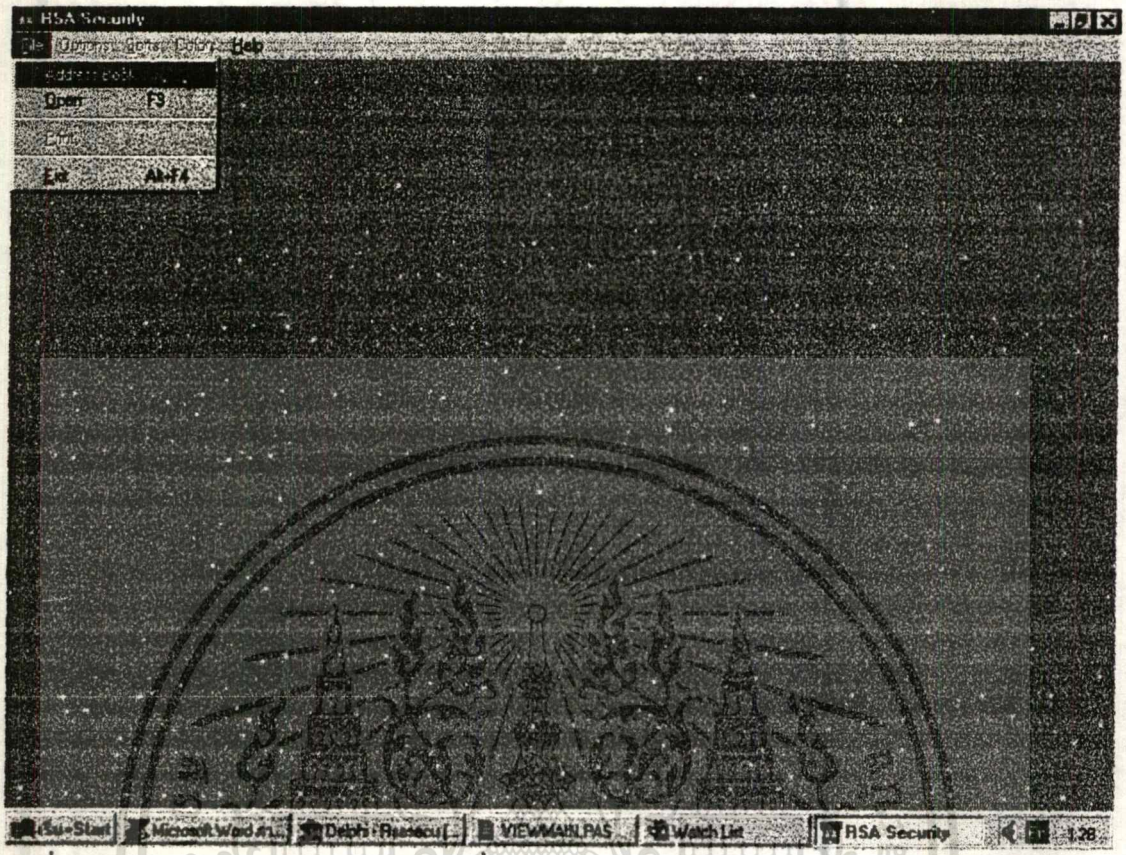


รูปที่ 28 หน้าจอแสดงไฟล์หลังการเข้ารหัสโปรแกรม

เมื่อต้องการถอดรหัสไฟล์ที่เข้ารหัสแล้วให้คลิกที่ Security | Decryption แล้วโปรแกรมจะให้ทำการใส่ รหัส(Password ของ User) เอง ถ้าป้อนถูกต้องก็จะทำการถอดรหัสให้แต่ถ้าไม่ถูกต้องก็จะไม่สามารถถอดรหัสได้

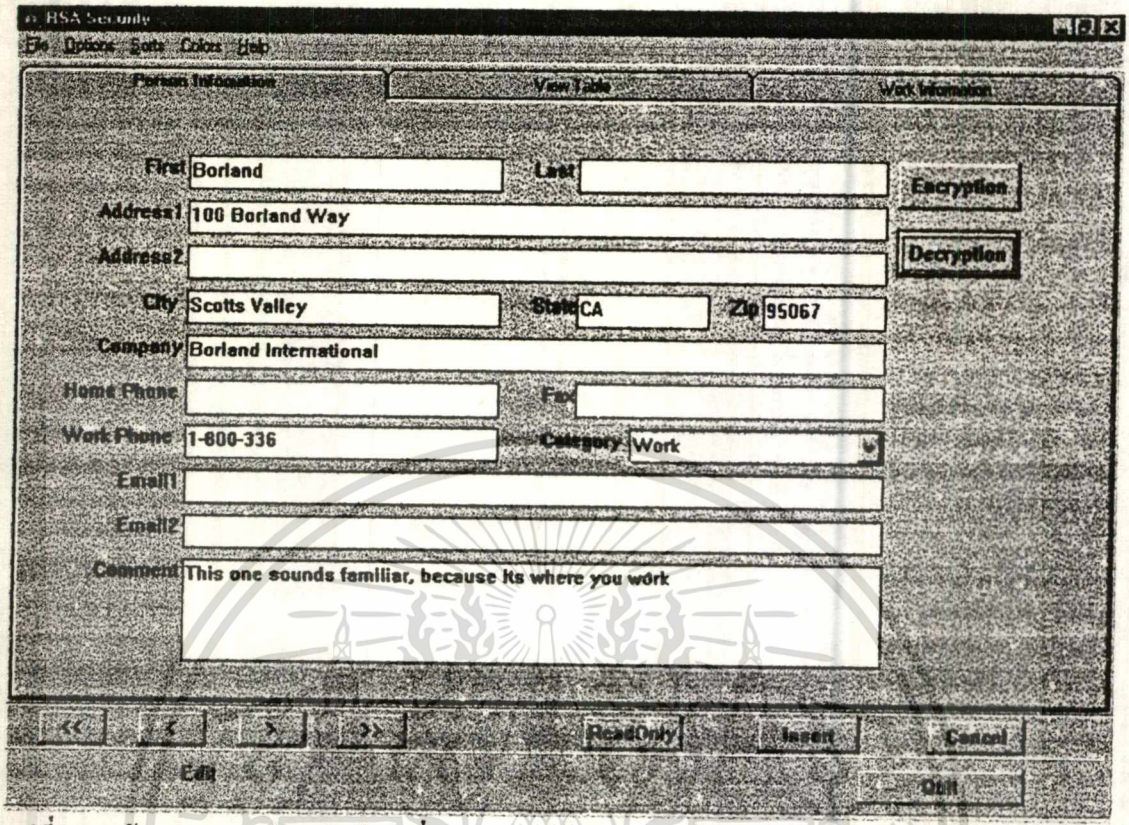
4.2.2 การเข้ารหัส ประวัติของบุคคลในแฟ้มประวัติที่จัดเก็บไว้ภายใน Table Adress.DB ซึ่ง Table นี้จะใช้รวบรวมข้อมูลต่าง ๆ ของแต่ละบุคคลที่เราเก็บไว้ซึ่งเราสามารถที่จะ เพิ่ม(Insert) , ลบ(Delete),เปลี่ยนแปลงแก้ไข(Update) ข้อมูลได้

คลิกที่ File | Address Book ที่เมนู(Menu) เพื่อเข้าสู่การเข้ารหัสในส่วนของข้อมูลในตาราง(Table) ดังรูปที่ 29



รูปที่ 29 หน้าจอแสดงการเลือกการใช้งานที่เมนู(File | Address Book)

หลังจากคลิกที่ File | Address Book แล้วจะแสดงหน้าจอดังรูป จะแสดง ข้อมูลของบุคคลออกมา(Person Information)



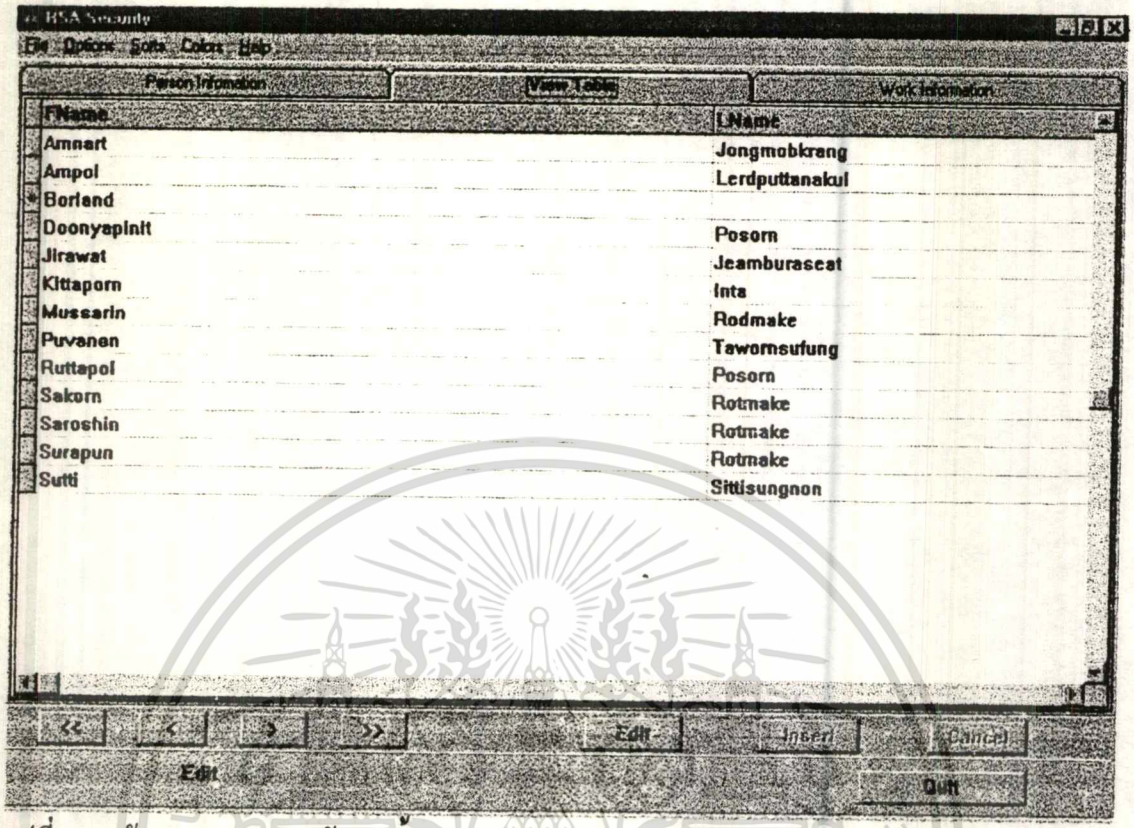
รูปที่ 30 หน้าจอแสดงหลังจากเข้ามาที่ระบบแอดเดรสบุค

โดยในรูปจะแสดงข้อมูลที่อินเด็กซ์ (Index) ของตาราง(Table) ชี้อยู่ซึ่งการใช้งานของโปรแกรมในขณะนี้จะมีอยู่ 2 โหมด(Mode) คือ

Mode Readonly ซึ่งจะไม่สามารถทำการเปลี่ยนแปลงแก้ไขข้อมูล(update),เพิ่ม(Insert),ลบ(Delete) จะสามารถเรียกดูได้อย่างเดียว(แต่เราสามารถเข้ารหัสถอดรหัสได้)

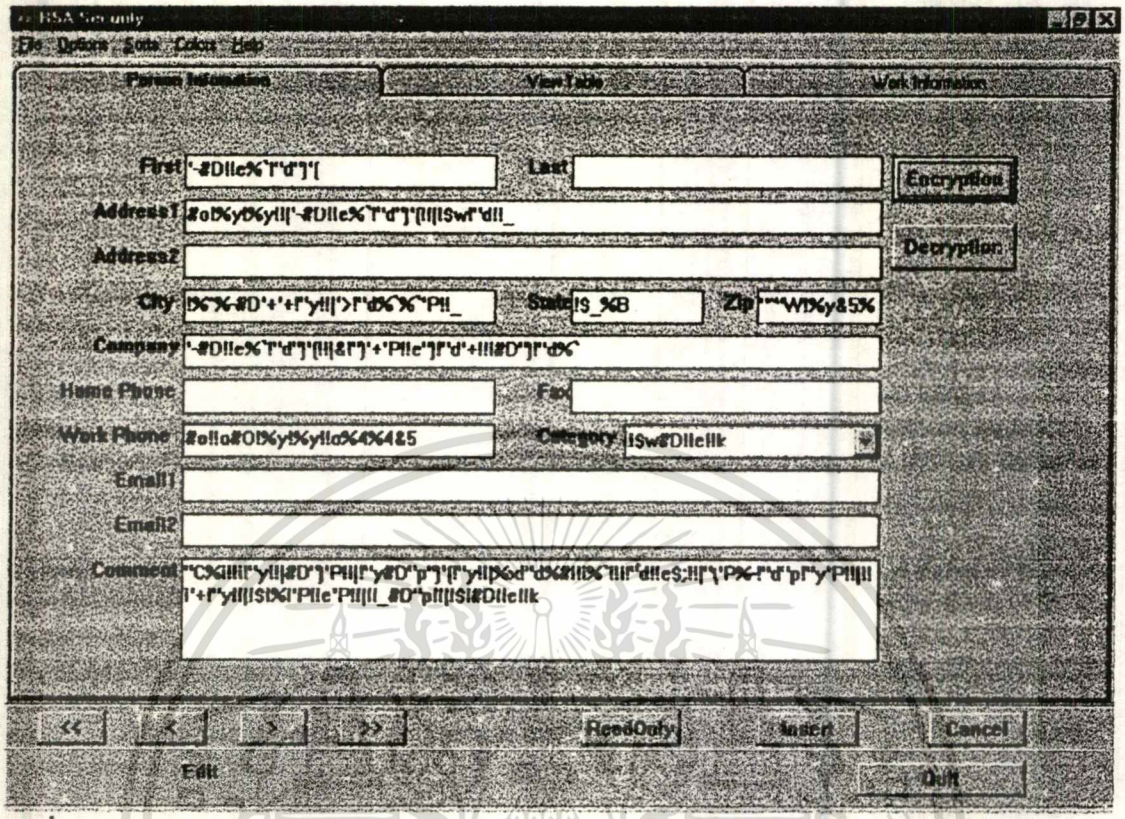
Mode Edit จะสามารถเปลี่ยนแปลงแก้ไขข้อมูล(update),ลบ(delete),เพิ่ม(Insert) เข้ารหัส(encryption),ถอดรหัส(encryption)ได้

ในกรณีที่ต้องการดูข้อมูลทั้หมดให้ คลิก ที่วิวเทเบิ้ล(View Tabel) ที่ฟอร์ม (Form) ของโปรแกรมจะแสดงข้อมูลดังรูป ใน วิวเทเบิล(View Table) นี้จะไม่สามารถเปลี่ยนแปลงแก้ไขข้อมูลได้จะดูได้เพียงอย่างเดียว

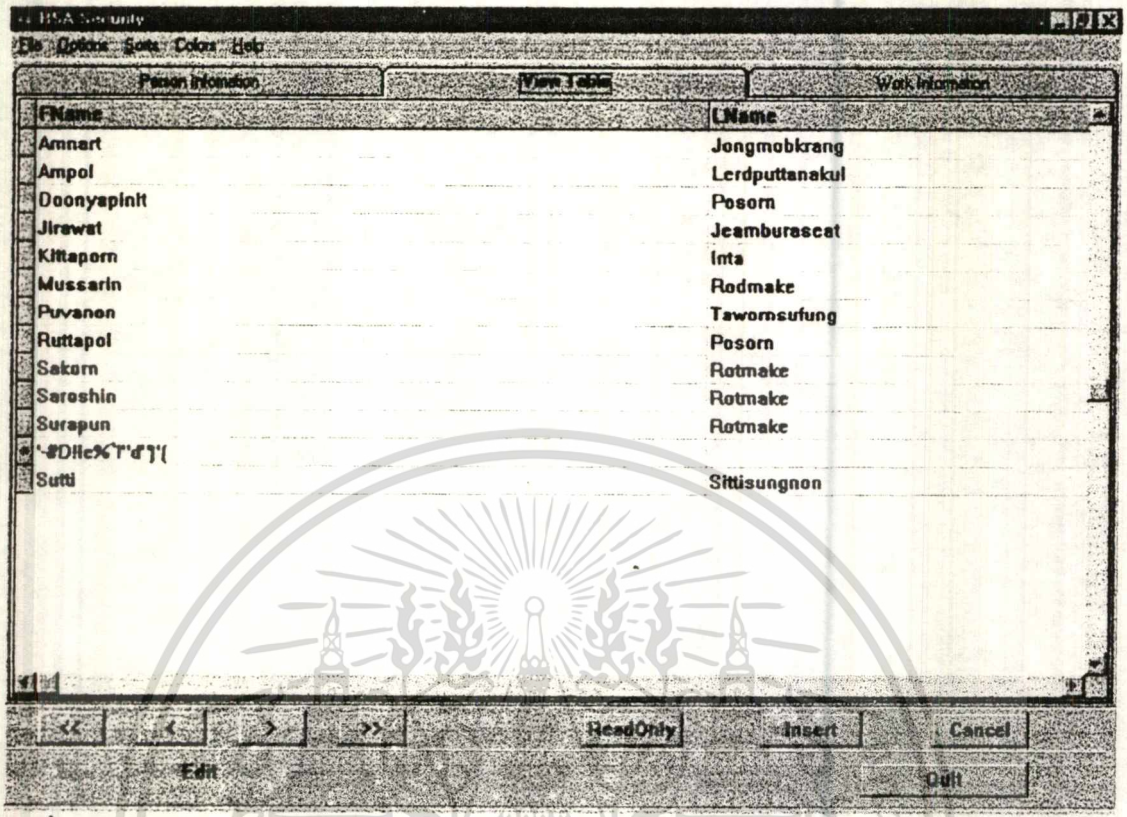


รูปที่ 31 หน้าจอแสดงตารางข้อมูลทั้งหมด

ในการเข้ารหัสนั้นให้เราเลื่อนข้อมูลไปที่ข้อมูลที่เราต้องการ โดยใช้ปุ่มเลื่อนซ้าย-ขวาที่หน้าจอด้านล่างซ้าย หรือคลิกที่ Option | Search ของเมนูค้นหา ถ้าพบยินเด็กซ์(Index) ของตาราง(Table) จะชี้ไปที่ข้อมูลนั้นหรือถ้าไม่พบจะชี้ในตำแหน่งข้อมูล (String) ที่ใกล้เคียง จากนั้นถ้าพบข้อมูลที่ต้องการเข้ารหัสแล้วให้คลิกที่ปุ่มเอ็นครับชั่น (Encryption) ก็จะทำการเข้ารหัสข้อมูล แสดงดังรูปที่ 32

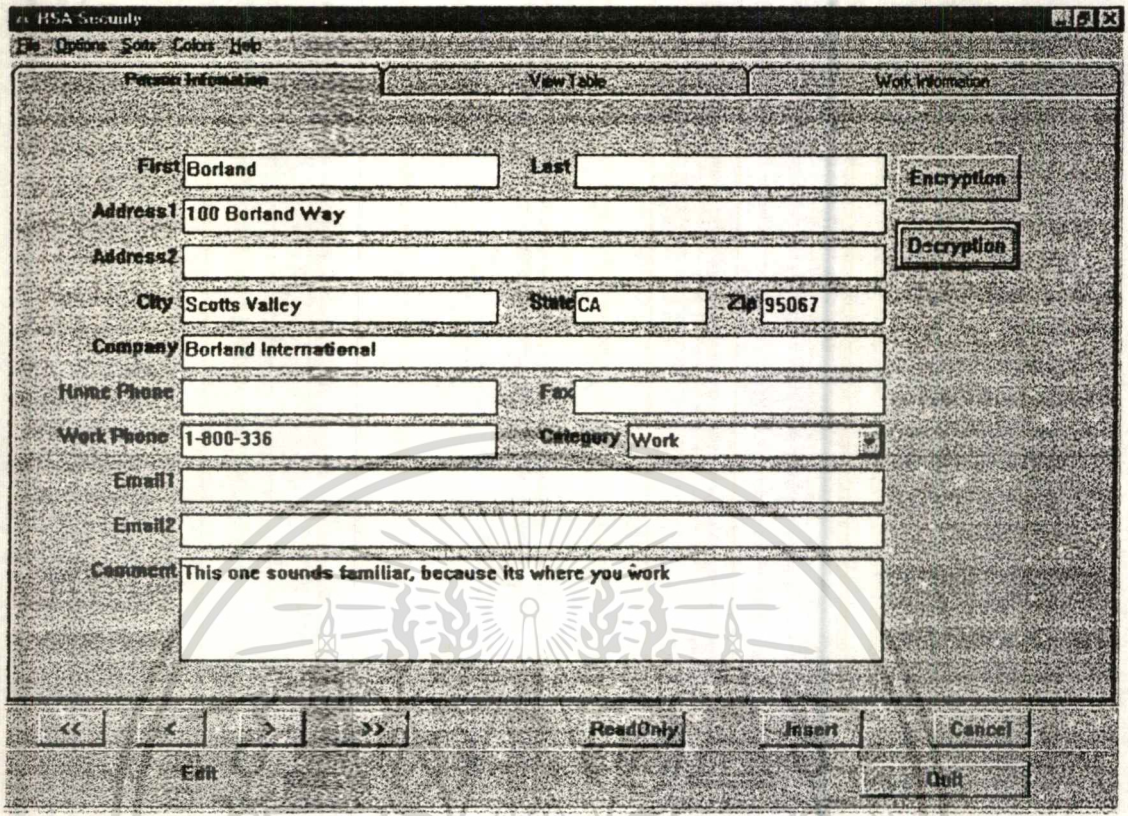


รูปที่ 32 หน้าจอหลักหลังการเข้ารหัสของข้อมูล



รูปที่ 33 หน้าจอตารางข้อมูลหลังมีการเข้ารหัส

และเมื่อต้องการถอดรหัสก็ คลิกที่ปุ่มดีครับชั่น(Decryption) ก็จะทำการถอดรหัสให้ข้อมูลก็จะคืนมาดังรูปที่ 34

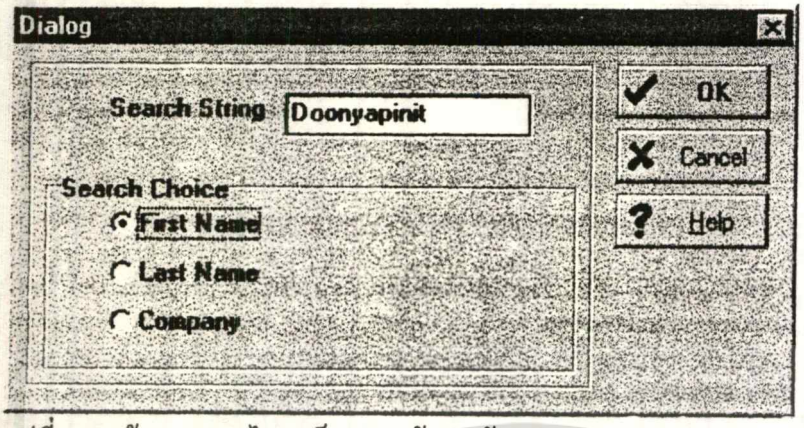


รูปที่ 34 หน้าจอหลักหลังการถอดรหัสของข้อมูล

ในโปรแกรมเราสามารถที่จะทำการเรียงลำดับข้อมูลได้โดย คลิกที่เมนู(Menu) ซอร์ท(Sort) ซึ่งจะมีให้เราเลือกว่าจะเรียงลำดับข้อมูล ในลักษณะใด เช่น อาจจะเรียงจาก First Name,Last Name หรือ Company หลังจากเรียงแล้วเราสามารถเรียกดูว่ามีการเรียงจริงหรือไม่จากตาราง(View Table)

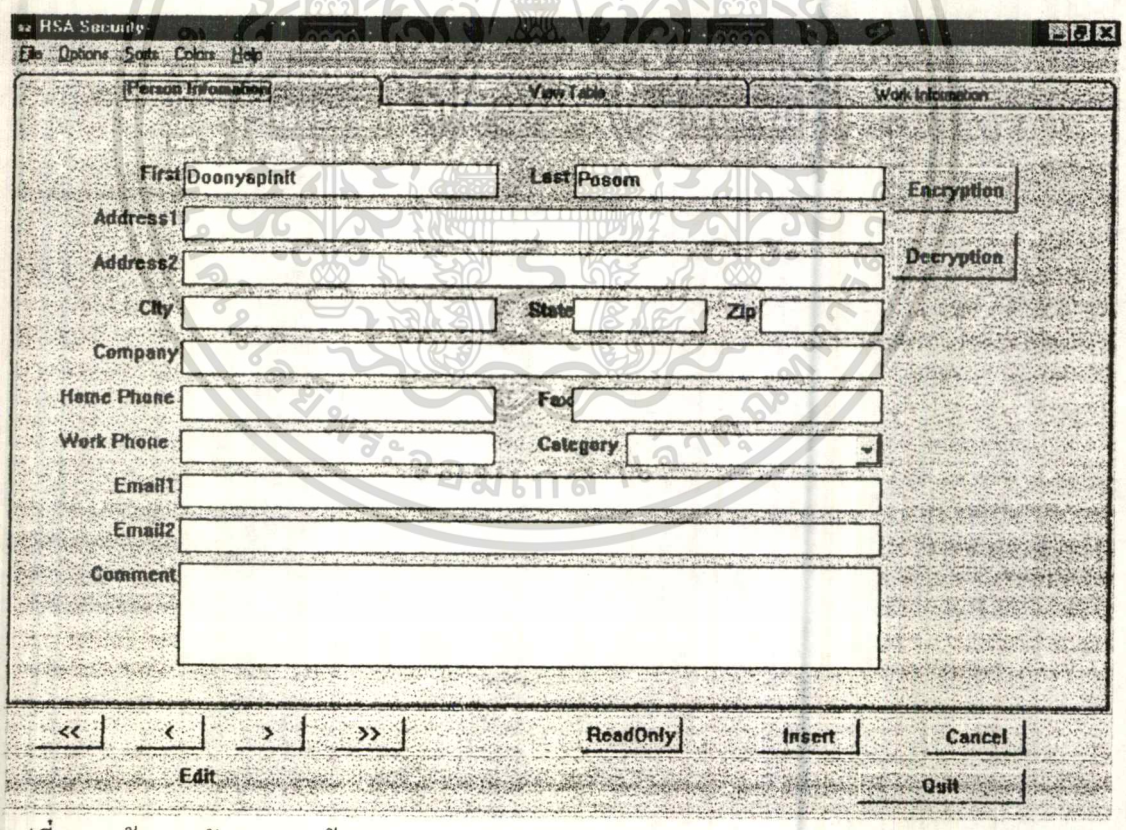
ในการลบ(Delete) นั้นจะต้องอยู่ใน Edit Mode จึงจะสามารถทำการลบข้อมูลได้ โดยเลื่อนไปที่ข้อมูลที่ต้องการลบแล้วคลิกที่เมนู(Menu) Option | Delete

ในกรณีที่ต้องการค้นหาข้อมูลที่ต้องการให้ คลิก(Click) ที่ เมนู(Menu) Option | Search จะแสดงหน้าจอให้พิมพ์ข้อมูลที่ต้องการค้นหาและเลือกด้วยว่าจะค้นหาด้วย First Name ,Last Name หรือ Company จากนั้นคลิกโอเค(Click OK) ก็จะทำการค้นหาให้ ดังรูปในหน้าถัดไป



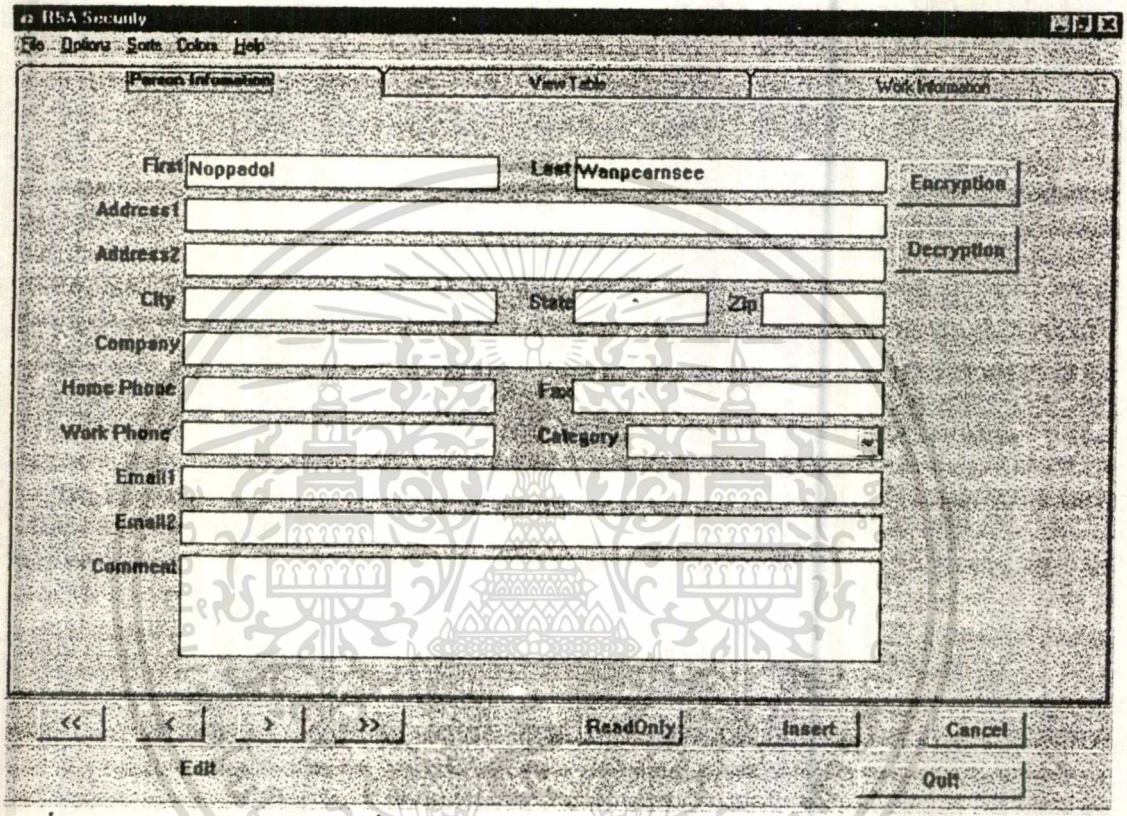
รูปที่ 35 หน้าจอแสดงไดอะล็อกการค้นหาข้อมูล

หลังจาก คลิกโอเค(Click OK) ถ้าพบข้อมูล ก็จะแสดงออกมาดังรูปแต่ถ้าไม่พบจะแสดงข้อความที่ใกล้เคียง



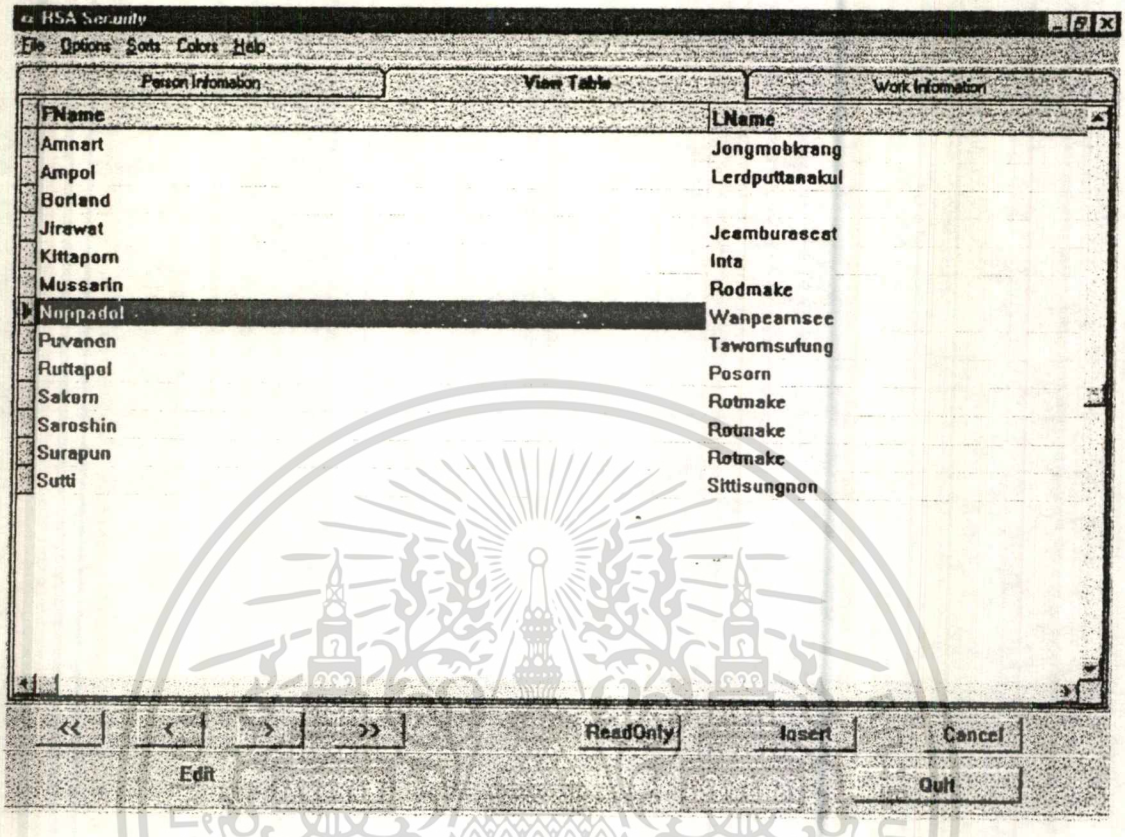
รูปที่ 36 หน้าจอหลังจากการค้นหา

ส่วนในการเพิ่ม (Insert) จะต้องทำใน Edit Mode จึงจะสามารถกระทำได้เช่นเราต้องการเพิ่มชื่อ Noppadol นามสกุล Wanpearnsee เข้าไปให้คลิกอินเสิรค(Click Insert) แล้วพิมพ์ข้อมูลที่ต้องการเพิ่ม ดังรูป



รูปที่ 37 หน้าจอสำหรับต้องการเพิ่มบุคคลใหม่

รูปที่ 38 หน้าจอแสดงตารางหลังการอินเสริด



บทที่ 5
สรุป วิจารณ์ และแนวทางในการพัฒนา

5.1 ความสามารถของระบบในโครงงานเมื่อเปรียบเทียบกับระบบที่ใช้งานอยู่จริง

เมื่อทำการเปรียบเทียบระหว่างระบบที่ทำการพัฒนาในโครงงานนี้กับระบบที่ใช้งานอยู่ หรือระบบที่มีขายอยู่ในท้องตลาดพบว่า ความสามารถพื้นฐานในการเข้ารหัสนั้นทำได้ดีในระดับหนึ่งเช่น

- สามารถเข้ารหัสเท็กซ์ไฟล์(Text File) ได้
- สามารถถอดรหัสเท็กซ์ไฟล์(Text File) ที่ถูกเข้ารหัสโดยโปรแกรมนี้ได้
- เก็บประวัติของบุคคลได้ในส่วนที่จำเป็นแล้วสามารถเข้ารหัสได้
- สามารถถอดรหัสประวัติของบุคคลที่ถูกเข้ารหัสโดยโปรแกรมนี้ได้

5.2 ข้อจำกัดของระบบ

- ในการหาค่าจำนวนเฉพาะ(prime number) เพื่อใช้ในการสร้างกุญแจที่เปิดเผย (Public Key) และกุญแจที่ไม่เปิดเผย (Private Key) ค่าที่ได้มีโอกาสเป็นค่าจำนวนเฉพาะ (prime number) สูง แต่ก็แต่ก็ยังไม่แน่นอนที่สุด ซึ่งในกรณีที่ได้ค่าที่ไม่ใช่จำนวนเฉพาะ (prime number) ก็จะทำให้ผลการเข้ารหัสและถอดรหัสมีความผิดพลาด

- มีข้อจำกัดในหาค่า a^z mod n การทำงานของฟังก์ชั่น(function) ที่ใช้ในการคำนวณหาค่าเรากำหนดให้ทุกค่าที่ใช้ในฟังก์ชัน(function) fastexp(a,z,n) เป็น longint ซึ่งlongint จะใช้หน่วยความจำขนาด 4 byte = 32 bit เป็นเลขจำนวนเต็มที่มีค่าตั้งแต่ -2,147,483,648 ถึง +2,147,483,647

ในกรณีที่เราให้ a = (a*a) mod n ---------- (1)

การหาค่า fastexp(a,z,n) ค่า a,n จะมีค่าสูงสุดได้ไม่เกิน 46,340 เนื่องจากว่าเมื่อ ค่า a = 46,340

จะทำให้

$a^2 = (46,340)^2 = 2,147,395,600$ จะไม่เกินขอบเขตของ longint

แต่ ถ้า a = 46,341 จะทำให้

$a^2 = (46,341)^2 = 2,147,488,281$ จะทำให้เกินขอบเขตของ longint

และในกรณีเดียวกัน ถ้า n เป็น 46,340 และจากสมการ (1) a = (a*a) mod n

เมื่อจำนวนใด ๆ mod 46,340 จะทำให้ได้ค่า กระจายอยู่ ระหว่าง 0 ถึง 46,340 จากสมการ (1) จะทำให้ ค่า a อาจเท่ากับ 46,340 ซึ่งเมื่อยกกำลังสองจะทำให้ค่าไม่เกินขอบเขตของ longint

แต่เมื่อค่า n มากกว่า 46,340 และเมื่อ ทำการ mod n ก็อาจจะทำให้ค่า a เกินค่า 46,340 ดังนั้น เมื่อเข้าสมการแล้ว และ a ยกกำลังสอง ก็จะทำให้ ค่าที่ได้เกินขอบเขตของ longint

ทั้งสองกรณีนี้ ก็จะทำให้เกิดข้อผิดพลาดในการเข้ารหัสถอดรหัสซึ่งในโปรแกรม(Program) ที่เขียนขึ้น ค่า a จะไม่เกินค่า 46,340 เนื่องจากว่าเราแปลงตัวอักษรให้เป็นรหัสแอสกี้(ASCII) ดังนั้นค่า a จะอยู่ระหว่าง 0 - 255

ดังนั้นเราจึงจะต้องควบคุมค่า n ไม่ให้เกิน 46,340 ซึ่ง

p*q = n <= 46340

เพราะฉนั้น การ สุ่มค่าจำนวนเฉพาะ(prime) p,q จะต้องตรวจสอบทุกครั้ง

- ในการหาค่าจำนวนเฉพาะ(prime number) ของโปรแกรมนั้นได้ค่าที่มีค่าไม่มากทำให้ความปลอดภัยในการเข้ารหัสข้อมูลนั้นมีไม่มาก

- ในการเข้ารหัสตัวอักษร 1 ตัวอักษรนั้นจะทำให้ได้ตัวอักษรเพิ่มขึ้นมาจากเดิมประมาณ 1-2 ตัว ในการเข้ารหัสแฟ้มข้อมูล(Text File) นั้นจะมีตั่วอักษรเพิ่มขึ้นจากเดิม ส่วนในการเข้ารหัสบุคคลในตาราง นั้นจะมีข้อจำกัดเพิ่มมากขึ้น คือถ้าข้อมูลในช่อง DBedit มีจำนวนมากๆ จะทำให้การเข้ารหัสอาจจะได้จำนวนตัวอักษรเกินขอบเขตที่จะสามารถเก็บได้ของฟิลด์(Field) นั้น จะทำให้การถอดรหัสจะถอดได้ไม่ครบ เนื่องจาก

รหัสของข้อมูลที่เข้ารหัสไม่สามารถเก็บในฟิลด์(Field) ของมันในตาราง(Tabel)ได้จึงทำให้เกิดข้อผิดพลาดขึ้นได้

- ในการเข้ารหัสในแต่ละครั้งนั้นตัวอักษรเดียวกันจะได้โค้ดในการเข้ารหัส (Cipher Text) ที่เหมือนกันซึ่งอาจจะทำให้ผู้ไม่หวังดีสามารถตีความหมายข้อมูลนี้ได้

5.3 แนวทางในการพัฒนาระบบต่อไปในอนาคต

ควรจะมีการปรับปรุงแก้ไขโปรแกรมในส่วนที่มีข้อจำกัดให้มีประสิทธิภาพมากขึ้น

- ในข้อจำกัดในการหาค่าจำนวนเฉพาะนั้นควรจะหาให้ได้ค่าที่มีโอกาสเป็นค่าจำนวนเฉพาะมากที่สุดเพื่อที่จะทำให้ข้อมูลที่เข้ารหัสมีความปลอดภัยของข้อมูลสูง

- การเข้ารหัสข้อมูลนั้นควรจะเข้ารหัสเป็นชุด(Block) ครั้งละมากกว่า 1 ตัวอักษร เพราะจะทำให้ข้อมูลที่เข้ารหัสแล้วได้โค้ด(Cipher Text) ที่ไม่เหมือนกันทำให้ผู้ที่ไม่ประสงค์ดีถอดรหัสข้อมูลได้ยากมากขึ้น และการเข้ารหัสเป็นชุด(Block) นี้จะทำให้ได้โค้ดที่เข้ารหัส(Cipher) ที่มีขนาดไม่มากจนเกินไปทำให้สามารถจัดเก็บโค้ดที่เข้ารหัสแล้วได้หมดทำให้การถอดรหัสได้ออกมาอย่างถูกต้อง

- ในการหาค่ายกกำลังนั้นค่าที่ได้อาจได้ค่าที่เกินขอบเขตของลองอินท์ (longint) นั้น เราอาจจะใช้ภาษาแอสแซมบลี้เข้ามาช่วยในการเขียนโปรแกรม เพราะเดลไฟล์ (Delphi)สามารถที่จะเชื่อมต่อกับแอสแซมบลี้ได้ซึ่งจะทำให้เราสามารถใช้โค้ดของแอสแซมบลี้มาช่วยในการเข้าถึงในระดับต่ำได้ ซึ่งจะทำให้เราสามารถหาค่า จำนวนเฉพาะที่มีค่ามาก ๆ ได้ และยังสามารถรองรับผลของค่าที่ยกกำลังที่มีค่ามากได้ ทำให้ความสามารถในการเข้ารหัสมีความซับซ้อนเพิ่มมากขึ้น

ภาคผนวก

```
program Rsasecu;

uses
  Forms,
  Viewmain in 'VIEWMAIN.PAS' {MDIFileViewer},
  Textview in 'TEXTVIEW.PAS' {TextViewer},
  Che_pk in 'CHE_PK.PAS' {Passwd2},
  Cre_pass in 'CRE_PASS.PAS' {Passwd1},
  Stepgen in 'STEPGEN.PAS' {step_gen},
  Search in 'SEARCH.PAS' {SearchDlg},
  Filter in 'FILTER.PAS' {FilterDlg};

{$R *.RES}

begin
  Application.CreateForm(TMDIFileViewer, MDIFileViewer);
  Application.CreateForm(TPasswd1, Passwd1);
  Application.CreateForm(TSearchDlg, SearchDlg);
  Application.CreateForm(TFilterDlg, FilterDlg);
  Application.Run;
end.
```

```
unit Viewmain;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Menus, StdCtrls, Grids, DBGrids, DBTables, DB, Buttons,
  ExtCtrls, DBCtrls, Mask, TabNotBk, Report, IniFiles;

type
  TColorType = (ccForm,ccEdit,ccEditText,ccLabel,ccPanel);
  TChangeType = (tcColor,tcFontcolor);
  TMDIFileViewer = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    N1: TMenuItem;
    Help1: TMenuItem;
    About1: TMenuItem;
    FileOpenDialog: TOpenDialog;
    Exit1: TMenuItem;
    Open1: TMenuItem;
    Panel1: TPanel;
    addr: TMenuItem;
    Panel2: TPanel;
    label16: TLabel;
    Bevel1: TBevel;
    Print1: TMenuItem;
    N2: TMenuItem;
    Option1: TMenuItem;
    Delete1: TMenuItem;
    Search1: TMenuItem;
    Filter1: TMenuItem;
    Sort1: TMenuItem;
    stFirst1: TMenuItem;
```

```
stLast1: TMenuItem;
stCompany1: TMenuItem;
Color1: TMenuItem;
ccEdits: TMenuItem;
ccEditText: TMenuItem;
ccLabels: TMenuItem;
ccPanels: TMenuItem;
Table1: TTable;
DataSource1: TDataSource;
btFirst: TButton;
btPrior: TButton;
btNext: TButton;
btLast: TButton;
btEdit: TButton;
btInsert: TButton;
btCancel: TButton;
TabbedNotebook1: TTabbedNotebook;
DBEdit1: TDBEdit;
DBEdit2: TDBEdit;
DBEdit3: TDBEdit;
DBEdit4: TDBEdit;
DBEdit5: TDBEdit;
DBEdit6: TDBEdit;
DBEdit7: TDBEdit;
DBEdit9: TDBEdit;
DBEdit10: TDBEdit;
DBEdit11: TDBEdit;
cbCategory: TDBComboBox;
DBMemo1: TDBMemo;
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
```

Label6: TLabel;

Label7: TLabel;

Label9: TLabel;

Label10: TLabel;

Label11: TLabel;

Label13: TLabel;

Label12: TLabel;

Label15: TLabel;

DBGrid1: TDBGrid;

Button1: TButton;

btEncryp: TButton;

btDecryp: TButton;

Label17: TLabel;

Label18: TLabel;

Label19: TLabel;

Label20: TLabel;

Label21: TLabel;

Label22: TLabel;

Label23: TLabel;

Label24: TLabel;

Label25: TLabel;

Label26: TLabel;

Label27: TLabel;

Label28: TLabel;

GroupBox1: TGroupBox;

Label29: TLabel;

Label30: TLabel;

GroupBox2: TGroupBox;

Label31: TLabel;

Label32: TLabel;

GroupBox3: TGroupBox;

Label33: TLabel;

Label34: TLabel;

DBEdit14: TDBEdit;

DBEdit15: TDBEdit;

DBEdit16: TDBEdit;

DBEdit17: TDBEdit;

DBEdit18: TDBEdit;

DBEdit19: TDBEdit;

DBEdit20: TDBEdit;

DBEdit21: TDBEdit;

DBEdit22: TDBEdit;

DBEdit23: TDBEdit;

DBEdit24: TDBEdit;

DBEdit25: TDBEdit;

DBEdit26: TDBEdit;

DBEdit27: TDBEdit;

DBEdit28: TDBEdit;

DBEdit29: TDBEdit;

DBEdit30: TDBEdit;

DBEdit31: TDBEdit;

Button2: TButton;

Button3: TButton;

Button4: TButton;

Button5: TButton;

Label35: TLabel;

Label36: TLabel;

DBEdit32: TDBEdit;

Label37: TLabel;

DBEdit33: TDBEdit;

DBEdit8: TDBEdit;

Label8: TLabel;

Label14: TLabel;

DBEdit12: TDBEdit;

Label38: TLabel;

Button6: TButton;

Button7: TButton;

ColorDialog1: TColorDialog;

```
Label39: TLabel;
Label40: TLabel;
Label41: TLabel;
Label42: TLabel;
Label43: TLabel;
Label44: TLabel;
Label45: TLabel;
Label46: TLabel;
Label47: TLabel;
Exemple1: TMenuItem;
Panel3: TPanel;
Edit1: TEdit;
Label48: TLabel;
Label50: TLabel;
Label51: TLabel;
Button8: TButton;
Button9: TButton;
Label53: TLabel;
Label54: TLabel;
Button10: TButton;
Label49: TLabel;
Label52: TLabel;
Bevel2: TBevel;
Bevel3: TBevel;
Label55: TLabel;
Edit2: TEdit;
Report1: TReport;
procedure Open1Click(Sender: TObject);
procedure CloseAllChildren;
procedure Exit1Click(Sender: TObject);
procedure addrClick(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure btFirstClick(Sender: TObject);
procedure btPriorClick(Sender: TObject);
```

```
    procedure btNextClick(Sender: TObject);
    procedure btLastClick(Sender: TObject);
    procedure btEditClick(Sender: TObject);
    procedure btInsertClick(Sender: TObject);
    procedure btCancelClick(Sender: TObject);
    procedure Print1Click(Sender: TObject);
    procedure HandleEditMode;
    Procedure SetReadOnly(NewState:Boolean);
    procedure Delete1Click(Sender: TObject);
    procedure stFirst1Click(Sender: TObject);
    procedure stLast1Click(Sender: TObject);
    procedure stCompany1Click(Sender: TObject);
    procedure Search1Click(Sender: TObject);
    procedure Dosort;
    procedure DoSearch(S:String);
    procedure Button1Click(Sender: TObject);
    procedure btEncrypClick(Sender: TObject);
    procedure btDecrypClick(Sender: TObject);
    procedure Filter1Click(Sender: TObject);
    procedure ccEditsClick(Sender: TObject);
    procedure ccEditTextClick(Sender: TObject);
    procedure ccLabelsClick(Sender: TObject);
    procedure ccPanelsClick(Sender: TObject);
    procedure About1Click(Sender: TObject);
    procedure Exemple1Click(Sender: TObject);
    procedure Button8Click(Sender: TObject);
    procedure Button9Click(Sender: TObject);
    procedure Button10Click(Sender: TObject);
    procedure Help1Click(Sender: TObject);
    procedure File1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
```

```
    procedure encrypstr(var tempstr:string);
    procedure decrypstr(var tempstr:string);
    function no_encryp(sttemp:string;min,max:integer):Boolean;
    procedure printdata;
    function CloseReportSmith:Boolean;
    procedure SetEdits(TypeChange:TchangeType;NewValue:TColor);
    procedure SetLabels(C:TColor);
    procedure SetPanels(C:TColor);

  end;

var

  MDIFileViewer: TMDIFileViewer;
  data:integer;

implementation
uses TextView,Che_pk,Cre_pass,Search,Filter;

{$R *.DFM}


procedure TMDIFileViewer.Open1Click(Sender: TObject);
begin
addr.tag:=0;
panel1.Hide;
panel2.Hide;
option1.enabled:=false;
sort1.enabled:=false;
color1.enabled:=false;
print1.enabled:=false;
 If FileOpenDialog.Execute Then
   Begin
    TextViewer:=TTextViewer.Create(Self);
```

```
    TextViewer.Open(FileOpenDialog.Filename);
    TextViewer.Visible:=True;
    TextViewer.SetFocus;
   end;
end;

Procedure TMDIFileViewer.CloseAllChildren;
var
 i:Integer;
 begin
  for i:=0 To MDIChildCount-1 do
   MDIChildren[i].Close
  end;


procedure TMDIFileViewer.Exit1Click(Sender: TObject);
begin
  addr.tag:=1;
  print1.tag:=0;
  addrclick(sender);
  passwd1.Buttoncancelclick(sender);
  IF TV_chk=1 then begin
  TextViewer.Closeall1Click(Sender);
  tv_chk:=0;
            end;
  MDIFileViewer.Enabled:=False;
  passwd1.show;

end;



procedure TMDIFileViewer.addrClick(Sender: TObject);
begin
```

```
TabbedNotebook1.PageIndex:=0;
if addr.Tag=0 then
  begin
      tabbednotebook1.show;
      Panel1.show;
      panel2.show;
      option1.enabled:=true;
      sort1.enabled:=true;
      color1.enabled:=true;
      addr.Tag:= 1;
      print1.Enabled:=true;
      panel3.hide;
  end
else if addr.Tag=1 then
      begin
          Panel1.Hide;
          panel2.hide;
          option1.enabled:=false;
          sort1.enabled:=false;
          color1.enabled:=false;
          addr.Tag:=2;
          print1.Enabled:=false;
          panel3.hide;
      end
     else begin
            tabbednotebook1.show;
            Panel1.show;
            Panel2.show;
            option1.enabled:=true;
            sort1.enabled:=true;
            color1.enabled:=true;
            print1.Enabled:=true;
            addr.Tag:=1;
            panel3.hide;
```

```
        end;

end;


procedure TMDIFileViewer.FormCreate(Sender: TObject);
begin
option1.enabled:=false;
sort1.enabled:=false;
color1.enabled:=false;
tv_chk:=0;

end;

procedure TMDIFileViewer.btFirstClick(Sender: TObject);
begin
   Table1.First;
end;

procedure TMDIFileViewer.btPriorClick(Sender: TObject);
begin
   Table1.Prior;
end;

procedure TMDIFileViewer.btNextClick(Sender: TObject);
begin
   Table1.Next;
end;

procedure TMDIFileViewer.btLastClick(Sender: TObject);
begin
   Table1.Last;
end;
```

```
procedure TMDIFileViewer.btEditClick(Sender: TObject);
begin
    HandleEditMode;
end;

procedure TMDIFileViewer.btInsertClick(Sender: TObject);
begin
    TabbedNotebook1.PageIndex:=0;
    if DbEdit1.Text='' then begin messagebeep(0);exit;end;
    Table1.Insert;
end;

procedure TMDIFileViewer.btCancelClick(Sender: TObject);
begin
    Table1.Cancel;
end;

procedure TMDIFileViewer.Print1Click(Sender: TObject);
begin
  PrintData;

end;

procedure TMDIFileViewer.HandleEditMode;
begin
    btInsert.Enabled:=not DataSource1.AutoEdit;
    btCancel.Enabled:=not DataSource1.AutoEdit;
    Delete1.Enabled:=not DataSource1.AutoEdit;
    if not DataSource1.AutoEdit then begin
      SetreadOnly(True);
      btEdit.Caption := 'ReadOnly';
      label16.Caption:='Edit Mode'
      end else begin
           if Table1.State<>dsBrowse Then Table1.Post;
```

```
            SetReadOnly(False);
            btEdit.Caption:='Edit';
            label16.caption:='ReadOnly Mode'
             end;
end;


Procedure TMDIFileviewer.SetReadOnly(NewState:Boolean);
begin
   DataSource1.AutoEdit:=NewState;
end;

procedure TMDIFileViewer.Delete1Click(Sender: TObject);
begin
   if DbEdit1.Text='' then begin messagebeep(0);exit;end;
   Table1.Delete;
end;

procedure TMDIFileViewer.stFirst1Click(Sender: TObject);
begin
   Table1.IndexName:='FNameIndex';
   FSortType:=TSortType((StFirst1).tag);
   DoSearch('A');
end;

procedure TMDIFileViewer.stLast1Click(Sender: TObject);
begin
   Table1.IndexName:='LNameIndex';
   FSortType:=TSortType((StLast1).tag);
   DoSearch('A');
end;

procedure TMDIFileViewer.stCompany1Click(Sender: TObject);
begin
```

```
    Table1.IndexName:='CompanyIndex';
    FSortType:=TSortType((StCompany1).tag);
    DoSearch('A');
end;

procedure TMDIFileViewer.Search1Click(Sender: TObject);
var s:String;
begin
 if not SearchDlg.GetSearchStr(FSortType,S) then exit;
 DoSort;
 DoSearch(S);
end;

procedure TMDIFileviewer.Dosort;
begin
 if FSortType=TSortType((stFirst1).tag) then
  Table1.IndexName:='FnameIndex'
  else if FSortType=TSortType((StLast1).tag) then
      Table1.IndexName:='LnameIndex'
      else Table1.IndexName:='CompanyIndex';
end;

procedure TMDIFileviewer.DoSearch(S:String);
begin
  Table1.SetKey;
 if FSortType=TSortType((StFirst1).tag) then
    Table1.FieldByName('FName').Asstring:=S
 else if FSortType=TSortType((StLast1).tag)then
       Table1.FieldByName('LName').Asstring:=S
      else Table1.FieldByName('Company').Asstring:=S;
 Table1.GotoNearest;
end;

procedure TMDIFileViewer.Button1Click(Sender: TObject);
```

```
begin
    option1.enabled:=false;
    sort1.enabled:=false;
    color1.enabled:=false;
    addr.tag:=0;
    print1.tag:=0;
    panel1.hide;
    panel2.hide;

end;

procedure TMDIFileViewer.btEncrypClick(Sender: TObject);
var temp:array[1..34] of string;
begin
temp[1]:=Table1.FieldByName('Fname').AsString;
if no_encryp(temp[1],64,126) then begin messagebeep(0);exit end;
Encrypstr(temp[1]);
passwd1.w_data_file_en(us_na,Table1.FieldByName('Fname').AsString);
temp[2]:=Dbedit2.text;Encrypstr(temp[2]);
temp[3]:=Dbedit3.text;Encrypstr(temp[3]);
temp[4]:=Dbedit4.text;Encrypstr(temp[4]);
temp[5]:=Dbedit5.text;Encrypstr(temp[5]);
temp[6]:=Dbedit6.text;Encrypstr(temp[6]);
temp[7]:=Dbedit7.text;Encrypstr(temp[7]);
temp[8]:=Dbedit8.text;Encrypstr(temp[8]);
temp[9]:=Dbedit9.text;Encrypstr(temp[9]);
temp[10]:=Dbedit10.text;Encrypstr(temp[10]);
temp[11]:=Dbedit11.text;Encrypstr(temp[11]);
temp[12]:=Dbedit12.text;Encrypstr(temp[12]);
temp[14]:=Dbedit14.text;Encrypstr(temp[14]);
temp[15]:=Dbedit15.text;Encrypstr(temp[15]);
temp[16]:=Dbedit16.text;Encrypstr(temp[16]);
temp[17]:=Dbedit17.text;Encrypstr(temp[17]);
temp[18]:=Dbedit18.text;Encrypstr(temp[18]);
```

```
temp[19]:=Dbedit19.text;Encrypstr(temp[19]);
temp[20]:=Dbedit20.text;Encrypstr(temp[20]);
temp[21]:=Dbedit21.text;Encrypstr(temp[21]);
temp[22]:=Dbedit22.text;Encrypstr(temp[22]);
temp[23]:=Dbedit23.text;Encrypstr(temp[23]);
temp[24]:=Dbedit24.text;Encrypstr(temp[24]);
temp[25]:=Dbedit25.text;Encrypstr(temp[25]);
temp[26]:=Dbedit26.text;Encrypstr(temp[26]);
temp[27]:=Dbedit27.text;Encrypstr(temp[27]);
temp[28]:=Dbedit28.text;Encrypstr(temp[28]);
temp[29]:=Dbedit29.text;Encrypstr(temp[29]);
temp[30]:=Dbedit30.text;Encrypstr(temp[30]);
temp[31]:=Dbedit31.text;Encrypstr(temp[31]);
temp[32]:=Dbedit32.text;Encrypstr(temp[32]);
temp[33]:=Dbedit33.text;Encrypstr(temp[33]);

temp[13]:=Table1.FieldByname('Comment').AsString;Encrypstr(temp[13]);
temp[34]:=cbcategory.text;Encrypstr(temp[34]);
Table1.Delete;
Table1.Insert;
Table1.FieldByName('Fname').AsString:=temp[1];
Table1.FieldByName('Lname').AsString:=temp[2];
Table1.FieldByName('Dbirth').AsString:=temp[3];
Table1.FieldByName('Nationality').AsString:=temp[4];
Table1.FieldByName('Race').AsString:=temp[5];
Table1.FieldByName('Age').AsString:=temp[6];
Table1.FieldByName('Riligion').AsString:=temp[7];
Table1.FieldByName('Sex').AsString:=temp[8];
Table1.FieldByName('Address').AsString:=temp[9];
Table1.FieldByName('City').AsString:=temp[10];
Table1.FieldByName('HPhone').AsString:=temp[11];
Table1.FieldByName('Email').AsString:=temp[12];
Table1.FieldByName('Comment').AsString:=temp[13];
Table1.FieldByName('Degree').AsString:=temp[14];
```

```
Table1.FieldByName('Major').AsString:=temp[15];
Table1.FieldByName('University').AsString:=temp[16];
Table1.FieldByName('UAddress').AsString:=temp[17];
Table1.FieldByName('Ucity').AsString:=temp[18];
Table1.FieldByName('Gpa').AsString:=temp[19];
Table1.FieldByName('Dgraduate').AsString:=temp[20];
Table1.FieldByName('Lang1').AsString:=temp[21];
Table1.FieldByName('Lang2').AsString:=temp[22];
Table1.FieldByName('Sport1').AsString:=temp[23];
Table1.FieldByName('Sport2').AsString:=temp[24];
Table1.FieldByName('Prog1').AsString:=temp[25];
Table1.FieldByName('Prog2').AsString:=temp[26];
Table1.FieldByName('Company').AsString:=temp[27];
Table1.FieldByName('Waddress').AsString:=temp[28];
Table1.FieldByName('Wcity').AsString:=temp[29];
Table1.FieldByName('Position').AsString:=temp[30];
Table1.FieldByName('Salary').AsString:=temp[31];
Table1.FieldByName('Wage').AsString:=temp[32];
Table1.FieldByName('WPhone').AsString:=temp[33];
Table1.FieldByName('Category').AsString:=temp[34];
Table1.Post;
end;

procedure TMDIFileViewer.btDecrypClick(Sender: TObject);
var temp:array[1..34] of string;
begin
temp[1]:=Table1.FieldByName('Fname').AsString;
if not(no_encryp(temp[1],64,126)) then begin messagebeep(0);exit end;
decrypstr(temp[1]);
temp[2]:=Dbedit2.text;Decrypstr(temp[2]);
temp[3]:=Dbedit3.text;Decrypstr(temp[3]);
temp[4]:=Dbedit4.text;Decrypstr(temp[4]);
temp[5]:=Dbedit5.text;Decrypstr(temp[5]);
temp[6]:=Dbedit6.text;Decrypstr(temp[6]);
```

```
temp[7]:=Dbedit7.text;Decrypstr(temp[7]);
temp[8]:=Dbedit8.text;Decrypstr(temp[8]);
temp[9]:=Dbedit9.text;Decrypstr(temp[9]);
temp[10]:=Dbedit10.text;Decrypstr(temp[10]);
temp[11]:=Dbedit11.text;Decrypstr(temp[11]);
temp[12]:=Dbedit12.text;Decrypstr(temp[12]);
temp[14]:=Dbedit14.text;Decrypstr(temp[14]);
temp[15]:=Dbedit15.text;Decrypstr(temp[15]);
temp[16]:=Dbedit16.text;Decrypstr(temp[16]);
temp[17]:=Dbedit17.text;Decrypstr(temp[17]);
temp[18]:=Dbedit18.text;Decrypstr(temp[18]);
temp[19]:=Dbedit19.text;Decrypstr(temp[19]);
temp[20]:=Dbedit20.text;Decrypstr(temp[20]);
temp[21]:=Dbedit21.text;Decrypstr(temp[21]);
temp[22]:=Dbedit22.text;Decrypstr(temp[22]);
temp[23]:=Dbedit23.text;Decrypstr(temp[23]);
temp[24]:=Dbedit24.text;Decrypstr(temp[24]);
temp[25]:=Dbedit25.text;Decrypstr(temp[25]);
temp[26]:=Dbedit26.text;Decrypstr(temp[26]);
temp[27]:=Dbedit27.text;Decrypstr(temp[27]);
temp[28]:=Dbedit28.text;Decrypstr(temp[28]);
temp[29]:=Dbedit29.text;Decrypstr(temp[29]);
temp[30]:=Dbedit30.text;Decrypstr(temp[30]);
temp[31]:=Dbedit31.text;Decrypstr(temp[31]);
temp[32]:=Dbedit32.text;Decrypstr(temp[32]);
temp[33]:=Dbedit33.text;Decrypstr(temp[33]);
temp[13]:=Table1.FieldByname('Comment').AsString;Decrypstr(temp[13]);
temp[34]:=cbcategory.text;Decrypstr(temp[34]);

if not(passwd1.check_data(us_na,temp[1])) then
begin messagebeep(0);showmessage('Data has owner');exit; end;
passwd1.del_data(us_na,temp[1]);

Table1.Delete;
```

```
Table1.Insert;
Table1.FieldByName('Fname').AsString:=temp[1];
Table1.FieldByName('Lname').AsString:=temp[2];
Table1.FieldByName('Dbirth').AsString:=temp[3];
Table1.FieldByName('Nationality').AsString:=temp[4];
Table1.FieldByName('Race').AsString:=temp[5];
Table1.FieldByName('Age').AsString:=temp[6];
Table1.FieldByName('Riligion').AsString:=temp[7];
Table1.FieldByName('Sex').AsString:=temp[8];
Table1.FieldByName('Address').AsString:=temp[9];
Table1.FieldByName('City').AsString:=temp[10];
Table1.FieldByName('HPhone').AsString:=temp[11];
Table1.FieldByName('Email').AsString:=temp[12];
Table1.FieldByName('Comment').AsString:=temp[13];
Table1.FieldByName('Degree').AsString:=temp[14];
Table1.FieldByName('Major').AsString:=temp[15];
Table1.FieldByName('University').AsString:=temp[16];
Table1.FieldByName('UAddress').AsString:=temp[17];
Table1.FieldByName('Ucity').AsString:=temp[18];
Table1.FieldByName('Gpa').AsString:=temp[19];
Table1.FieldByName('Dgraduate').AsString:=temp[20];
Table1.FieldByName('Lang1').AsString:=temp[21];
Table1.FieldByName('Lang2').AsString:=temp[22];
Table1.FieldByName('Sport1').AsString:=temp[23];
Table1.FieldByName('Sport2').AsString:=temp[24];
Table1.FieldByName('Prog1').AsString:=temp[25];
Table1.FieldByName('Prog2').AsString:=temp[26];
Table1.FieldByName('Company').AsString:=temp[27];
Table1.FieldByName('Waddress').AsString:=temp[28];
Table1.FieldByName('Wcity').AsString:=temp[29];
Table1.FieldByName('Position').AsString:=temp[30];
Table1.FieldByName('Salary').AsString:=temp[31];
Table1.FieldByName('Wage').AsString:=temp[32];
Table1.FieldByName('WPhone').AsString:=temp[33];
```

```
Table1.FieldByName('Category').AsString:=temp[34];


Table1.Post;


end;

procedure TMDIFileviewer.encrypstr(var tempstr:string);
var col,count_c,temp1,temp2:integer;int:longint;
   st1:string;onech,twoch:char;
begin
st1:='';
     count_c:=ord(tempstr[0]);
     for col:=1 to count_c do
      begin
       int:=ord(tempstr[col]);    {change char to ascii code}
       int:=TextViewer.fastexp(int,e,n);    {encryption}
     if((int mod 128)>93) then
     begin
          temp1:=int div 128;
          temp1:=temp1+33;
          temp2:=int mod 128;
          st1:=st1+chr(33);
     end
      else begin
      temp1:=int div 128;
      temp1:=temp1+34;
      temp2:=int mod 128;
      temp2:=temp2+34;
         end;
     onech:=chr(temp1);
     twoch:=chr(temp2);
     st1:=st1+onech+twoch;
```

```
      end;
 tempstr:=st1;
end;

procedure TMDIFileviewer.decrypstr(var tempStr:string);
var col,count_c,cou:integer;temp1,temp2:longint;
   sttemp,st:string;one:char;
begin
     sttemp:='';
     st:='';
     cou:=1;
     count_c:=ord(tempstr[0]);      {find How much of string}
     col:=2;                {initial of one sting}
     while col<=count_c do
    begin
      temp1:=ord(tempstr[col-1]);
       if temp1=33 then  begin
         temp1:=ord(tempstr[col])-33;
         temp2:=ord(tempstr[col+1]);
         col:=col+1;
                  end
       else begin
      temp1:=temp1-34;
      temp2:=ord(tempstr[col]);
      temp2:=temp2-34;
           end;
      temp1:=(temp1*128)+temp2;
      temp1:=Textviewer.fastexp(temp1,d,n); {decryption}
      st[cou]:=chr(temp1);
      sttemp:=sttemp+st[cou];
      cou:=cou+1;
      col:=col+2;
     {end;}
    end;
```

```
tempstr:=sttemp;
end;

function TMDIFileViewer.no_Encryp(sttemp:string;min,max:integer):Boolean;
var col,ck,ck1,i:integer;
begin
    col:=ord(sttemp[0]);ck1:=0;
    for i:=1 to col do
    begin
        ck:=ord(sttemp[i]);
        if (ck<=min)or(ck>=max) then  ck1:=ck1+1;
    end;
    no_encryp:=ck1>1; {have charactor encryp ago }
end;

procedure TMDIFileViewer.Filter1Click(Sender: TObject);
var S:String;
begin
 S:=FilterDlg.GetFilter(cbCategory.items);
 if s='' then Exit;
 Table1.IndexName:='categoryIndex';
 Table1.SetRangeStart;
 Table1.FieldByName('Category').AsString:=S;
 Table1.SetRangeEnd;
 Table1.FieldByName('Category').AsString:=S;
 Table1.ApplyRange;
end;

procedure TMDIFileViewer.PrintData;
begin
 Cursor:=crHourGlass;
 Report1.Run;
 Cursor:=crDefault;
end;
```

```
function TMDIFileviewer.CloseReportSmith:Boolean;
begin
  Result:=Report1.CloseApplication(False);
end;

procedure TMDIFileViewer.SetEdits(TypeChange:TchangeTYpe;
                        NewValue:TColor);
var i:integer;
begin
   for i:=0 to ComponentCount-1 do
    if (Components[i] is TdbEdit) or
      (Components[i] is TdbComboBox) or
      (Components[i] is TdbMemo) then
     case TypeChange of
      tcColor:TdbEdit(Components[i]).Color:=NewValue;
      tcFontColor:TdbEdit(Components[i]).Font.Color:=NewValue;
     end;
end;



procedure TMDIFileViewer.Setlabels(C:TColor);
var i:integer;
begin
  for i:=0 to ComponentCount-1 do
    if (Components[i] is Tlabel) then
      TLabel(Components[i]).Font.color:=C;
end;

procedure TMDIFileViewer.SetPanels(C:Tcolor);
var i:integer;
begin
  for i:=0 to ComponentCount-1 do
```

```
    if (Components[i] is Tpanel) then
     Tpanel(Components[i]).color:=C;
end;


procedure TMDIFileViewer.ccEditsClick(Sender: TObject);
begin
 if not ColorDialog1.Execute then Exit;
   SetEdits(tcColor,ColorDialog1.Color);
end;

procedure TMDIFileViewer.ccEditTextClick(Sender: TObject);
begin
 if not ColorDialog1.Execute then Exit;
   SetEdits(tcFontColor,ColorDialog1.Color);
end;

procedure TMDIFileViewer.ccLabelsClick(Sender: TObject);
begin
 if not ColorDialog1.Execute then Exit;
   SetLabels(ColorDialog1.color);
end;

procedure TMDIFileViewer.ccPanelsClick(Sender: TObject);
begin
 if not ColorDialog1.Execute then Exit;
   SetPanels(ColorDialog1.Color);
end;

procedure TMDIFileViewer.About1Click(Sender: TObject);
begin
 TextViewer.CloseAll1Click(Sender);
 Sort1.enabled:=false;
 Color1.enabled:=false;
```

```
 Option1.enabled:=false;
 panel1.show;
 TabbedNotebook1.hide;
 panel2.hide;
 panel3.hide;
end;

procedure TMDIFileViewer.Exemple1Click(Sender: TObject);
var ste,std,stn:string;
begin
 TextViewer.Closeall1Click(Sender);
 Sort1.enabled:=false;
 Color1.enabled:=false;
 Option1.enabled:=false;
 panel1.show;
 TabbedNotebook1.hide;
 panel2.hide;
 panel3.show;
 str(e,ste);str(d,std);str(n,stn);
 label49.caption:='Public Key <e,n> = <'+ste+','+stn+'> ;'+
 ' Private Key <d,n> = <'+std+','+stn+'> ;';

end;

procedure TMDIFileViewer.Button8Click(Sender: TObject);
var ste,stn,stans:string;code:integer;
begin
val(edit1.text,data,code);
if code<>0 then begin edit1.text:='';messagebeep(0);exit;end;
str(e,ste);str(n,stn);
data:=textviewer.fastexp(data,e,n);
str(data,stans);
label52.caption:='('+edit1.text+' power '+ste+') mod '+stn+
' = '+stans;
```

```
button9.enabled:=true;
button8.enabled:=false;
edit2.visible:=true;
label55.visible:=true;

end;

procedure TMDIFileViewer.Button9Click(Sender: TObject);
var std,stn,stans,stcipher:string;code,data1:integer;
begin
val(edit2.text,data1,code);
if code<>0 then begin edit2.text:='';messagebeep(0);exit;end;

str(data,stcipher);
data:=textviewer.fastexp(data,data1,n);
str(data,stans);str(data1,std);str(n,stn);
label54.caption:='('+stcipher+' power '+std+') mod '+stn+
' = '+stans ;
button9.enabled:=false;

end;

procedure TMDIFileViewer.Button10Click(Sender: TObject);
begin
button8.enabled:=true;
edit1.text:='';
label54.caption:='';
label52.caption:='';
edit2.text:='';
edit2.visible:=false;
label55.visible:=false;
button9.enabled:=false;
end;
```

```
procedure TMDIFileViewer.Help1Click(Sender: TObject);
begin
   Button10Click(Sender);
end;

procedure TMDIFileViewer.File1Click(Sender: TObject);
begin
   Button10Click(Sender);
end;

end.
```

```
unit Textview;

interface

uses
 SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
 Forms, Dialogs, StdCtrls, Menus, Printers;

type
 TTextViewer = class(TForm)
  Memo1: TMemo;
  MainMenu1: TMainMenu;
  Window1: TMenuItem;
  Tile1: TMenuItem;
  Cascade1: TMenuItem;
  Closeall1: TMenuItem;
  File1: TMenuItem;
  Open1: TMenuItem;
  Close1: TMenuItem;
  N1: TMenuItem;
  Exit1: TMenuItem;
  Security1: TMenuItem;
  Encryption1: TMenuItem;
  Decryption1: TMenuItem;
  Help1: TMenuItem;
  About1: TMenuItem;
  N2: TMenuItem;
  Exemple1: TMenuItem;
  Print1: TMenuItem;
  PrintSetup1: TMenuItem;
  N3: TMenuItem;
  PrintDialog1: TPrintDialog;
  PrinterSetupDialog1: TPrinterSetupDialog;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
```

```
    procedure Open1Click(Sender: TObject);
    procedure Close1Click(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure Tile1Click(Sender: TObject);
    procedure Cascade1Click(Sender: TObject);
    procedure Closeall1Click(Sender: TObject);
    procedure Encryption1Click(Sender: TObject);
    procedure Decryption1Click(Sender: TObject);
    procedure About1Click(Sender: TObject);
    procedure Exemple1Click(Sender: TObject);
    procedure PrintSetup1Click(Sender: TObject);
    procedure Print1Click(Sender: TObject);

  private
    Filename:String;
    { Private declarations }
  public
    Procedure Open(Const AFilename:String);
    function fastexp(a,z,num:longint):longint;
    function inv(a,num:longint):longint;
    function gcd(a,num:longint):longint;
    procedure ran_pri(var prime:longint);
    { Public declarations }
  end;

var
  TextViewer: TTextViewer;
  TV_chk:integer;

implementation
uses ViewMain,Che_pk,stepgen,Cre_pass;

{$R *.DFM}
function TTextViewer.fastexp(a,z,num:longint):longint;
```

```
  var a1,z1:longint;
     x:longint;
  begin {return x=a^z mod n}
     a1:=a; z1:=z;
     x:=1;
   while z1<>0 do {x(a1^z1 mod n)=a^z mod n}
    begin
      while z1 mod 2=0 do
       begin {square a1 while z1 is even}
        z1:=z1 div 2;
        a1:=(a1*a1)mod num
       end;
      z1:=z1-1;
      x:=(x*a1)mod num {multiply}
    end;
   fastexp:=x
 end;

 function TTextViewer.gcd(a,num:longint):longint;
 var  i:integer;
     g:array[0..100] of longint;
 begin
  g[0]:=num;
  g[1]:=a;
  i:=1;
  while g[i]<>0 do
   begin
    g[i+1]:=g[i-1]mod g[i];
    i:=i+1;
   end;
  gcd:=g[i-1];
 end;

 function TTextViewer.inv(a,num:longint):longint;
```

```
var g:array[0..50] of integer;
  u:array[0..50] of integer;
   v:array[0..50] of integer;
   sti,stg,stu,stv,sty:string;
   i,y,x:integer;
begin   {Return x such that ax mod n=1,where 0<a<n}
   g[0]:=num; g[1]:=a;
   u[0]:=1; u[1]:=0;
   v[0]:=0; v[1]:=1;
  i:=1;str(i:8,sti);
   str(g[0]:17,stg);str(u[0]:15,stu);str(v[0]:17,stv);
   step_gen.memo1.lines[0]:='     i          g[i]          u[i]'+
   '          v[i]          y';
   step_gen.memo1.lines[1]:='     0'+stg+stu+stv;
  str(g[1]:17,stg);str(u[1]:17,stu);str(v[1]:17,stv);
   step_gen.memo1.lines[2]:=+sti+stg+stu+stv;
showmessage('Next Step');
  while g[i]<>0 do {g[i]=u[i]n + v[i]a}
   begin
     y    := g[i-1] div g[i];
     g[i+1]:= g[i-1]-(y*g[i]);
     u[i+1]:= u[i-1]-(y*u[i]);
     v[i+1]:= v[i-1]-(y*v[i]);

 str(i:8,sti);str(g[i]:17,stg);str(u[i]:17,stu);
 str(v[i]:17,stv);str(y:13,sty);
 step_gen.memo1.lines[i+1]:=+sti+stg+stu+stv+sty;
 showmessage('Next Step');
    i    := i+1
   end;
str(i:8,sti);str(g[i]:17,stg);str(u[i]:17,stu);
str(v[i]:17,stv);str(y:13,sty);
step_gen.memo1.lines[i+1]:=+sti+stg+stu+stv;
```

```
    showmessage('Next Step');

    x:=v[i-1];
    if x>=0 then begin inv:=x;
    str(x:5,sti);
    step_gen.memo1.lines[i+2]:='inverse = '+sti;
            end
    else begin
     inv:=x+num;str(x:5,sti);str(num:5,stg);str(x+num:5,stu);
    step_gen.memo1.lines[i+2]:='inverse ='+sti+'+ ('+stg+') ='+stu;
       end;
 end;


procedure TTextViewer.ran_pri(var prime:longint);
var p:longint;ans,i,int,check:integer;
   rel:real;
begin
 repeat
  randomize;
  p:=random(215);
  rel:=sqrt(p);
  int:=round(rel);
for i:=2 to int do
begin
 ans:=p mod i;
 if ans=0 then begin
  check:=0;
  i:=int;   end
 else check:=1;
end;
 until   check=1;
 prime:=p;
```

```
end;


procedure TTextViewer.Open(Const AFilename:String);
begin
 Filename:=AFilename;
 Memo1.Lines.LoadFromFile(Filename);
 Caption:=Filename;
 TV_chk:=1;
end;


procedure TTextViewer.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 Action:=caFree;
end;

procedure TTextViewer.Open1Click(Sender: TObject);
begin
 MDIFileViewer.Open1Click(Sender);
end;

procedure TTextViewer.Close1Click(Sender: TObject);
begin
 Close;
end;

procedure TTextViewer.Exit1Click(Sender: TObject);
begin
 MDIFileViewer.Exit1Click(sender);
end;

procedure TTextViewer.Tile1Click(Sender: TObject);
begin
```

```
  Mdifileviewer.Tile;
 end;

 procedure TTextViewer.Cascade1Click(Sender: TObject);
 begin
  Mdifileviewer.cascade;
 end;

 procedure TTextViewer.Closeall1Click(Sender: TObject);
 begin
  mdifileviewer.closeallchildren;
 end;

 procedure TTextViewer.Decryption1Click(Sender: TObject);
 var F:textFile;i:integer;
    st:array[1..10]of string;
 begin
 If MDIFileviewer.FileOpenDialog.Execute Then
  begin
  if not(passwd1.check_data('total_en',MDIfileViewer.FileOpenDialog.FileName)) then
   begin messagebeep(0);showmessage('file not encryption');exit; end;
 end else exit;

  if not(passwd1.check_data(us_na,MDIfileViewer.FileOpenDialog.FileName)) then
  begin messagebeep(0);showmessage('file has owner');exit; end;

  check:=1;   {check=1=decryption}
  Passwd2:=TPasswd2.Create(Self);     {Create new form to check}
  Passwd2.Visible:=true;              {Private Key}
  Passwd2.Setfocus;
  Textviewer.enabled:=false;
  Mdifileviewer.enabled:=false;
  passwd1.del_data('total_en',MDIfileViewer.Fileopendialog.FileName);
  passwd1.del_data(us_na,MDIfileViewer.Fileopendialog.FileName);
```

```
  end;

procedure TTextViewer.Encryption1Click(Sender: TObject);
var F:textfile;
   st:string;
begin
 If MDIFileViewer.FileOpenDialog.Execute Then
  begin
  if (passwd1.check_data('total_en',MDIfileViewer.FileOpenDialog.FileName)) then
   begin messagebeep(0);
   showmessage('file encryption ago');exit; end;

  end else exit;

'check:=0;
 Passwd2:=TPasswd2.Create(Self);     {Create new form to check}
 Passwd2.Visible:=true;              {Private Key}
 Passwd2.Setfocus;
 Textviewer.enabled:=false;
 Mdifileviewer.enabled:=false;
 passwd1.w_data_file_en('total_en',MDIFileViewer.FileOpenDialog.FileName);
 passwd1.w_data_file_en(us_na,MDIFileViewer.FileOpenDialog.FileName);
 end;


procedure TTextViewer.About1Click(Sender: TObject);
begin
  MDIFileViewer.About1Click(Sender);
end;

procedure TTextViewer.Exemple1Click(Sender: TObject);
begin
  MDIFileViewer.Exemple1Click(Sender);
end;
```

```
procedure TTextViewer.PrintSetup1Click(Sender: TObject);
begin
  printerSetupDialog1.Execute;
end;

procedure TTextViewer.Print1Click(Sender: TObject);
var Line:integer;
PrintText:System.Text;
begin
  if Printdialog1.Execute then begin
    AssignPrn(PrintText);
    Rewrite(PrintText);
    Printer.Canvas.Font:=Memo1.Font;
    For Line:=0 to Memo1.Lines.Count-1 do
      writeln(PrintText,Memo1.Lines[Line]);
      System.Close(PrintText);
      end;
end;



end.
```

```
unit Che_pk;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, ExtCtrls, StdCtrls;

type
  TPasswd2 = class(TForm)
    Edit1: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Label4: TLabel;
    ch_de: TButton;
    procedure ch_deClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    procedure writefile(var fsource,fdest:textfile);
  end;

var
  Passwd2: TPasswd2;
  stint1,stint2:longint;
  check:integer; {check for decryp=1 or encryp=0}
implementation
uses textview,viewmain,Cre_pass;
```

```
{$R *.DFM}

procedure TPasswd2.ch_deClick(Sender: TObject);
var i,out,cou,col,count_c:integer;
   F,F1:textfile;
   st,st1:string;st_pass:string[8];
   temp1,temp2:longint;
   onech,twoch:string[1];
   int:longint;

begin
  st_pass:=edit1.text;
if passw = st_pass then  {input d,n is correct}

if check = 1 then    {check = decryption}
begin
  i:=0;
  AssignFile(F,MDIfileViewer.FileOpenDialog.FileName); { File selected in dialog }
  Reset(F);
  AssignFile(F1,'c:\delphi\data\temp.dat');
  Rewrite(F1);
   while not eof(F) do
    begin
     cou:=1;
     Readln(F, St);  { Read the first line out of the file }
     count_c:=ord(st[0]);       {find How much of string}
     col:=2;                    {initial of one sting}
     while col<=count_c do
    begin
      temp1:=ord(st[col-1]);
       if temp1=33 then  begin
         temp1:=ord(st[col])-33;
         temp2:=ord(st[col+1]);
```

```
            col:=col+1;
                    end
          else begin
          temp1:=temp1-34;
          temp2:=ord(st[col]);
          temp2:=temp2-34;
              end;
          temp1:=(temp1*256)+temp2;
          temp1:=Textviewer.fastexp(temp1,d,n); {decryption}
          st[cou]:=chr(temp1);
          cou:=cou+1;
          col:=col+2;
       {end;}
      end;
         while cou<=count_c do
         begin  st[cou]:=' ';          {clear next st[cou]}
              cou:=cou+1;
         end;
       writeln(F1,st);
    end;

    writefile(F1,F);
    CloseFile(F1);   {close file}
    CloseFile(F);
    Textviewer.enabled:=True;
    Mdifileviewer.enabled:=True;
    Passwd2.close;
    ShowMessage('Decryption is complete');
  end

  else begin   {check = encryption}
    AssignFile(F,MDIfileViewer.FileOpenDialog.FileName);
    { File selected in dialog }
    Reset(F);
```

```
AssignFile(F1,'c:\delphi\data\temp.dat');
rewrite(F1);
 while not eof(F) do
 begin
  st1:='';
  Readln(F,St);  { Read the first line out of the file }
  count_c:=ord(st[0]);
  for col:=1 to count_c do
  begin
   int:=ord(st[col]);    {change char to ascii code}
   int:=TextViewer.fastexp(int,e,n);    {encryption}
  if((int mod 256)>220) then
  begin
       temp1:=int div 256;
       temp1:=temp1+33;
       temp2:=int mod 256;
       st1:=st1+chr(33);
  end
  else begin
  temp1:=int div 256;
  temp1:=temp1+34;
  temp2:=int mod 256;
  temp2:=temp2+34;
      end;
  onech:=chr(temp1);
  twoch:=chr(temp2);
  st1:=st1+onech+twoch;
 end;
writeln(F1,st1);

 end;
writefile(F1,F);

CloseFile(F1);
```

```
    CloseFile(F);   {close file}

  Textviewer.enabled:=True;
  Mdifileviewer.enabled:=True;
  Passwd2.close;
  ShowMessage('Encryption is complete');
  {end;}
  end
    else begin ShowMessage('Private Key Incorrect');  {input d,n is incorrect}
          Textviewer.enabled:=True;
          Mdifileviewer.enabled:=True;
          Passwd2.close;
       end ;
end;

procedure TPasswd2.FormCreate(Sender: TObject);
var stint1,stint2,stint3,stint4,stint5,stint6:string;
begin
   str(e,stint1);
   str(d,stint2);
   str(n,stint3);
   str(fee_n,stint4);
   str(p,stint5);
   str(q,stint6);
   label2.caption:=+Us_na;
   label4.caption:='e= '+stint1+'; d= '
   +stint2+'; n= '+stint3+'; fec_n='+stint4+
   '; p:='+stint5+'; q:='+stint6;     {show Private Key}
end;


procedure TPasswd2.writefile(var fsource,fdest:textfile);
var st:string;
begin
```

```
    Rewrite(fdest);
    reset(fsource);
    while not eof(fsource) do
    begin
      readln(fsource,st);
      writeln(fdest,st);
    end;
end;

end.
```

```
unit Cre_pass;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, ExtCtrls;

type
  TPasswd1 = class(TForm)
    Edit1: TEdit;
    Edit2: TEdit;
    Ok_pass: TButton;
    Label1: TLabel;
    Label2: TLabel;
    ButtonAdd: TButton;
    ButtonSys: TButton;
    ButtonCancel: TButton;
    ButtonExit: TButton;
    ButtonList: TButton;
    Bevel2: TBevel;
    procedure Ok_passClick(Sender: TObject);
    procedure ButtonSysClick(Sender: TObject);
    procedure ButtonAddClick(Sender: TObject);
    procedure ButtonCancelClick(Sender: TObject);
    procedure ButtonExitClick(Sender: TObject);
    procedure ButtonListClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
    procedure creatfile_chk(name:String);
```

```
    procedure w_data_file_en(name,info:string);
    function check_data(name,info:string):boolean;
    procedure del_data(name,info:string);
    procedure swap(name:string);
  end;
const Filename = 'c:\delphi\data\password.txt';
type
  secure = RECORD
        username :string;
        pass : string[8];
        e : longint;
        d : longint;
        n : longint;
        p : longint;
        q : longint;
        fee_n : longint;
  end;
  struc = record
        f_name:string;
  end;

var
  Passwd1: TPasswd1;
  Us_na:string;
  passw:string[8];checkA_S:integer; {add or system}
  p,q,e,d,n,fee_n:longint;
  passwfile : file of secure;
  sec_rec : secure;
  file_ck : struc;
implementation
uses Viewmain,TextView,Stepgen,che_pk;

{$R *.DFM}
```

```
procedure TPasswd1.Ok_passClick(Sender: TObject);
const Filename = 'c:\delphi\data\password.txt';
var   size:longint;
    check:integer;
begin
check:=0;
Us_na:=edit1.text;      {read username}
passw:=edit2.text;        {read password}
if ord(Us_na[0])>0 then  {username more than 0 charactor}
if ord(passw[0])<4 then   {password > 6 charactor}
  begin
    edit2.text:='';
    showmessage('Please type Password over 4 charactor');
  end
  else begin


 if checkA_S=0 then {buttonSysClick goto system}begin
   assignfile(passwfile,filename);
   reset(passwfile);          {read password file}


   while not eof(passwfile) do
    begin
      read(passwfile,sec_rec); {check password with file}
      if (sec_rec.username=us_na)and(sec_rec.pass=passw) then
            begin
       showmessage('Password OK');  {old user}
       creatfile_chk(sec_rec.username);
       e:=sec_rec.e;
       d:=sec_rec.d;
       n:=sec_rec.n;
       p:=sec_rec.p;
       q:=sec_rec.q;
```

```
        fee_n:=sec_rec.fee_n;
        check:=1;  end; {check=1=old user}
      end;
    closefile(passwfile);
  if check<>1 then  begin
  showmessage('Password incorrect'); {new user}
  edit1.text:='';
  edit2.text:='';end    {user not member}
 else   begin    {check=1=old user}
    MDIFileViewer.Enabled:=True; {goto RSA Program}
    Passwd1.Close;
     end;                end {end checkA_S}
 else {add user checkA_S=1}

 begin{add user}      {generate Key}
   assignfile(passwfile,filename);
   reset(passwfile);          {read password file}

   while not eof(passwfile) do
    begin
     read(passwfile,sec_rec); {check password with file}
     if (sec_rec.username=us_na) then
        check:=1;
    end;

if check<>1 then begin {if}  {username is new :ok}
Step_gen:=TStep_gen.create(Self);
Step_gen.Visible:=True;
Step_gen.SetFocus;     {create step form}
passwd1.visible:=false;
          end{if check<>1}
else {if check =1}  {old username}
begin {else}
   showmessage('Select new user name');
```

```
    edit1.text:='';
 end; {else}
end; {add user}
  end   {end if ..<6}
 else showmessage('Please type UserName and Password');

 end;

procedure TPasswd1.ButtonSysClick(Sender: TObject);
begin
buttonAdd.visible:=false;
buttonSys.visible:=false;
buttonExit.visible:=false;
buttonList.visible:=false;
ok_pass.visible:=true;
edit1.visible:=true;
edit2.visible:=true;
label1.visible:=true;
label2.visible:=true;
buttoncancel.visible:=true;
passwd1.caption:='Login To System';
checkA_S:=0;
end;

procedure TPasswd1.ButtonAddClick(Sender: TObject);
begin
buttonAdd.visible:=false;
buttonSys.visible:=false;
buttonExit.visible:=false;
buttonList.visible:=false;
ok_pass.visible:=true;
edit1.visible:=true;
edit2.visible:=true;
label1.visible:=true;
```

```
  label2.visible:=true;
  buttoncancel.visible:=true;
  passwd1.caption:='Add User For System';
  checkA_S:=1;
  end;

  procedure TPasswd1.ButtonCancelClick(Sender: TObject);
  begin
  buttonAdd.visible:=true;
  buttonSys.visible:=true;
  buttonExit.visible:=true;
  buttonList.visible:=true;
  ok_pass.visible:=false;
  edit1.visible:=false;
  edit2.visible:=false;
  label1.visible:=false;
  label2.visible:=false;
  buttoncancel.visible:=false;
  passwd1.caption:='RSA Security';
  edit1.text:='';
  edit2.text:='';
  end;

 procedure TPasswd1.ButtonExitClick(Sender: TObject);
 begin
 passwd1.close;
 MDIFileviewer.close;
end;

procedure TPasswd1.ButtonListClick(Sender: TObject);
var  sec_rec : secure;sti,ste,stn:string;
    passwfile : file of secure; i:integer ;
begin
  checkA_S:=2;
```

```
    Step_gen:=TStep_gen.create(Self);
    Step_gen.Visible:=True;
    Step_gen.SetFocus;     {create step form use to user list}
    step_gen.caption:='User List';
    step_gen.memo1.lines[0]:='     Name & Public Key';
    passwd1.visible:=false;
    step_gen.buttonrandom.visible:=false;
    step_gen.memo1.visible:=true;
    step_gen.buttonclear.visible:=true;
    step_gen.buttonout_list.visible:=true;

   i:=1;
   Assignfile(passwfile,filename);
   reset(passwfile);
   while not eof(passwfile) do
    begin {while}
     read(passwfile,sec_rec); {read file for list user}
     str(i,sti);str(sec_rec.e,ste);str(sec_rec.n,stn);
     step_gen.memo1.lines[i]:=+sti+'. '+sec_rec.username+
     '  '+ste+','+stn;
     i:=i+1;
    end; {while}
   closefile(passwfile);
end;

procedure TPasswd1.FormCreate(Sender: TObject);
begin
  assignfile(passwfile,filename);
   {$I-} Reset(Passwfile); {$I+}
   if IOResult<>0 then
     begin Rewrite(passwfile);
         closefile(passwfile);
     end;
  creatfile_chk('total_en');
```

```
end;


procedure Tpasswd1.creatfile_chk(name:String);
var F:file of struc;
begin
  assignfile(F,'c:\delphi\data\'+name);
   {$I-} Reset(F); {$I+}
   if IOResult<>0 then
     begin rewrite(F);
        closeFile(F);
     end;
end;

procedure Tpasswd1.w_data_file_en(name,info:string);
var F:file of Struc;
begin
 assignfile(f,'c:\delphi\data\'+name);
 reset(f);          {read file}
 seek(f,filesize(f)); {find size file for write}
 file_ck.f_name:=info;
 write(f,file_ck);  {write new user and key to file}
 closeFile(f);
end;

function Tpasswd1.check_data(name,info:string):boolean;
var F:file of struc;check:integer;
begin
 check:=0;
 assignfile(f,'c:\delphi\data\'+name);
 reset(f);
 while not eof(f) do
 begin
  read(f,file_ck);
```

```
      if file_ck.f_name = info then
        check:=1;
    end;
    check_data:=check = 1;
    closefile(f);
  end;

  procedure Tpasswd1.del_data(name,info:string);
  var F:file of struc;i:longint;
  begin
   assignFile(f,'c:\delphi\data\'+name);
   reset(f);
   while not eof(f) do
   begin
      read(f,file_ck);
      if file_ck.f_name = info then
       begin file_ck.f_name:='';
          i:=filepos(f);
          seek(f,i-1);
          write(f,file_ck);
      end;
   end;
  closefile(f);
  swap(name);
 end;

 procedure Tpasswd1.swap(name:string);
 var F,F1:file of struc;
 begin
   assignfile(F1,'c:\delphi\data\'+name);
   creatfile_chk('swap');
   assignfile(F,'c:\delphi\data\swap');
   reset(F1);
   rewrite(F);
```

```
    while not eof(F1) do
    begin  read(F1,file_ck);
        if file_ck.f_name<>'' then
         write(F,file_ck);
    end;
    reset(F);
    rewrite(F1);
    while not eof(F) do
    begin read(F,file_ck);
        write(F1,file_ck);
    end;
   closefile(F1);
   closefile(F);
end;
end.
```

```
unit Stepgen;

interface

uses
 SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
 Forms, Dialogs, StdCtrls, ExtCtrls;

type
 Tstep_gen = class(TForm)
  Label1: TLabel;
  ButtonN_fee: TButton;
  Label2: TLabel;
  Label3: TLabel;
  Bevel1: TBevel;
  ButtonF_e: TButton;
  Label4: TLabel;
  ButtonInv: TButton;
  Label5: TLabel;
  Buttonclose: TButton;
  Bevel2: TBevel;
  Memo1: TMemo;
  Buttonclear: TButton;
  ButtonRandom: TButton;
  Buttonout_list: TButton;
  Panel1: TPanel;
  Label6: TLabel;
  Label7: TLabel;
  Label8: TLabel;
  Label9: TLabel;
  procedure ButtonN_feeClick(Sender: TObject);
  procedure ButtonF_eClick(Sender: TObject);
  procedure ButtonInvClick(Sender: TObject);
  procedure ButtoncloseClick(Sender: TObject);
```

```
    procedure ButtonclearClick(Sender: TObject);
    procedure ButtonRandomClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure test(p:integer);
    procedure Buttonout_listClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
const Filename = 'c:\delphi\data\password.txt';
type
  secure = RECORD
        username :string;
        pass : string[8];
        e : longint;
        d : longint;
        n : longint;
        p : longint;
        q : longint;
        fee_n : longint;
  end;

var
  step_gen: Tstep_gen;

implementation
uses Textview,cre_pass,Viewmain,che_pk;
{$R *.DFM}

procedure Tstep_gen.ButtonN_feeClick(Sender: TObject);
var  stn,st,st1,st2:string;
begin
str(n,stn);
```

```
label3.caption:='n = p*q = '+stn;

fee_n:=(p-1)*(q-1);
str(fee_n,st);
str(p-1,st1);
str(q-1,st2);
label4.caption:='0(n) = (p-1)*(q-1) = '+st1+'*'+st2+' = '+st;

buttonN_fee.visible:=false;
buttonF_e.visible:=true;
end;

procedure Tstep_gen.ButtonF_eClick(Sender: TObject);
var ste,st,st1:string;
begin
memo1.visible:=true;
str(e,ste);
memo1.lines[0]:='e = '+ste;
test(e);
str(p,st);
str(q,st1);
label2.Caption:='p = '+st+' ; q = '+st1+' ; e = '+ste;

buttonF_e.visible:=false;
buttonInv.visible:=true;
end;

procedure Tstep_gen.ButtonInvClick(Sender: TObject);
var st,st1,st2:string; i:integer;
begin
for i:=0 to 50 do
memo1.lines[i]:='';
if buttonInv.tag=0 then
begin
```

```
  d:=Textviewer.inv(e,fee_n);       {find d}
  str(d,st);
  str(e,st1);
  str(fee_n,st2);
  label5.caption:='d = inverse(e mod 0(n)) = inverse('+st1+' mod '+st2+') = '+st;
 buttonInv.tag:=1;
  buttonInv.caption:='Inverse Ok';
  end
  else if buttonInv.tag=1 then
  begin
  {showmessage('inverse OK');}
  memo1.visible:=false;
  buttoninv.visible:=false;
  buttonclose.visible:=true;
  end;
  end;

 procedure Tstep_gen.ButtoncloseClick(Sender: TObject);
 var  sec_rec : secure;ste,std,stfee_n,stans:string;
     passwfile : file of secure;ans:integer;
 begin
 if buttonclose.tag=0 then
 begin
   buttonclose.tag:=1;
   buttonclose.Caption:='d,e Correct';
   panel1.show;
   str(e,ste);str(d,std);str(fee_n,stfee_n);
   ans:=(e*d)mod fee_n;
   str(ans,stans);
   label8.caption:='('+ste+' * '+std+') mod '+stfee_n+
   ' = '+stans;
 end
 else if buttonclose.tag=1 then
  begin
```

```
  buttonclose.tag:=2;
  buttonclose.caption:='Close';
  panel1.hide;
  end
 else if buttonclose.tag=2 then
  begin
assignfile(passwfile,filename);
reset(passwfile);              {read password file}
seek(passwfile,filesize(passwfile)); {find size file for write}
 sec_rec.username:=us_na;
 sec_rec.pass:=passw;
 sec_rec.e:=e;
 sec_rec.d:=d;
 sec_rec.n:=n;
 sec_rec.p:=p;
 sec_rec.q:=q;
 sec_rec.fee_n:=fee_n;
write(passwfile,sec_rec);  {write new user and key to file}
closeFile(passwfile);

 passwd1.buttonAdd.visible:=true;
 passwd1.buttonSys.visible:=true;
 passwd1.buttonExit.visible:=true;
 passwd1.buttonList.visible:=true;
 passwd1.ok_pass.visible:=false;
 passwd1.edit1.visible:=false;
 passwd1.edit2.visible:=false;
 passwd1.label1.visible:=false;
 passwd1.label2.visible:=false;
 passwd1.buttoncancel.visible:=false;
 passwd1.caption:='RSA Security';
 passwd1.edit1.text:='';
 passwd1.edit2.text:='';
 passwd1.visible:=true;
```

```
  close;
 end;
 end;

procedure Tstep_gen.ButtonclearClick(Sender: TObject);
var  sec_rec : secure;
    passwfile : file of secure;
    F,F1:file of struc;
    i:integer;
begin
 assignfile(passwfile,filename);
 reset(passwfile);
 assignfile(F1,'c:\delphi\data\total_en');
 rewrite(F1);
 closefile(F1);
 while not eof(passwfile) do
 begin
    read(passwfile,sec_rec);
    assignfile(f,'c:\delphi\data\'+sec_rec.username);
  {$I-} Reset(f); {$I+}
   if IOResult=0 then  erase(f);
 end;
 rewrite(passwfile);
 closefile(passwfile);
for i:=1 to 50 do
 memo1.lines[i]:='';
end;

procedure Tstep_gen.ButtonRandomClick(Sender: TObject);
var  st,st1,st2,stp,stq,std,ste:string;
begin
if buttonrandom.tag=0 then
begin
 Textviewer.ran_pri(p); {find p}
```

```
Textviewer.ran_pri(q); {find q}
n:=p*q;
while(n<=256)or(q=p) do
 begin {while}
 Textviewer.ran_pri(p);
 Textviewer.ran_pri(q);
 n:=p*q;
 end; {while}

 str(p,stp);
 memo1.lines[0]:='p = '+stp;
 test(p);
 buttonrandom.tag:=1 ;
 buttonrandom.caption:='Find q';
end
else if buttonrandom.tag=1 then
 {showmessage('find q');}
begin
 str(q,stq);
 memo1.lines[0]:='q = '+stq;
 test(q);

fee_n:=(p-1)*(q-1);
Textviewer.ran_pri(e);          {find e}
while(Textviewer.gcd(e,fee_n)<>1)or(e=q) do
    Textviewer.ran_pri(e);

label1.Caption:='Random Prime Number';
str(p,st);
str(q,st1);
label2.Caption:='p = '+st+' ; q = '+st1;
buttonrandom.tag:=2;
buttonrandom.caption:='Next Step';
end
```

```
  else if buttonrandom.tag=2 then
  begin
  {showmessage('find n and 0(n)');}
  memo1.visible:=false;
  buttonrandom.visible:=false;
  buttonN_fee.visible:=true;
  end;
  end;

  procedure Tstep_gen.FormCreate(Sender: TObject);
  begin
  memo1.visible:=true;
  end;

  procedure Tstep_gen.test(p:integer);
  var  rel:real;
      i,ans,int:integer;strel,stint,stans,sti,stp:string;
  begin
   for i:=1 to 50 do
    memo1.lines[i]:='';

   rel:=sqrt(p);str(rel,strel);str(p,stp);
   memo1.lines[1]:='  root 2 of '+stp+' = '+strel;
   int:=round(rel);str(int,stint);
   memo1.lines[2]:=+strel+' turn up = '+stint;
   for i:=2 to int do
   begin ans:= p mod i;str(i,sti);str(ans,stans);
       memo1.lines[i+1]:=+stp+' mod '+sti+' = '+
       stans;
   end;
  end;
 procedure Tstep_gen.Buttonout_listClick(Sender: TObject);
 begin
  passwd1.buttonAdd.visible:=true;
```

```
    passwd1.buttonSys.visible:=true;
    passwd1.buttonExit.visible:=true;
    passwd1.buttonList.visible:=true;
    passwd1.ok_pass.visible:=false;
    passwd1.edit1.visible:=false;
    passwd1.edit2.visible:=false;
    passwd1.label1.visible:=false;
    passwd1.label2.visible:=false;
    passwd1.buttoncancel.visible:=false;
    passwd1.caption:='RSA Security';
    passwd1.edit1.text:='';
    passwd1.edit2.text:='';
    passwd1.visible:=true;
    close;

,end;

end.
```

```
unit Search;

interface

uses WinTypes, WinProcs, Classes, Graphics, Forms, Controls, Buttons,
 StdCtrls, ExtCtrls;

type
 TSortType =(stFirst1,StLast1,stCompany1);
 TSearchDlg = class(TForm)
  OKBtn: TBitBtn;
  CancelBtn: TBitBtn;
  HelpBtn: TBitBtn;
  Bevel1: TBevel;
  GroupBox1: TGroupBox;
  FirstName: TRadioButton;
  LastName: TRadioButton;
  Company: TRadioButton;
  Edit1: TEdit;
  Label1: TLabel;
  function GetSearchStr(var ST:TSortType;var S:String):Boolean;
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 SearchDlg: TSearchDlg;
 FSortType :TSortType;
implementation

{$R *.DFM}
uses viewmain;
```

```
'function TSearchDlg.GetSearchStr(var ST:TSortType;var S:String):Boolean;
 begin
    Result:=False;
    if ShowModal = mrCancel then Exit;
    s:=Edit1.Text;
    if FirstName.checked then ST:=STFirst1
    else if LastName.Checked then ST:= stLast1
    else ST:=stCompany1;
    Result:=True;
end;

end.
```

```
unit Filter;

interface

uses WinTypes, WinProcs, Classes, Graphics, Forms, Controls, Buttons,
 StdCtrls, ExtCtrls;

type
 TFilterDlg = class(TForm)
  OKBtn: TBitBtn;
  CancelBtn: TBitBtn;
  HelpBtn: TBitBtn;
  Bevel1: TBevel;
  ListBox1: TListBox;
 private
  { Private declarations }
 public
  { Public declarations }
  Function GetFilter(Slist:Tstrings):String;
 end;

var
 FilterDlg: TFilterDlg;
implementation
{$R *.DFM}
Function TFilterDlg.GetFilter(Slist:Tstrings):string;
begin
 ListBox1.Items:=Slist;
 if ShowModal = mrCancel then
  Result:=''
  else Result:=ListBox1.Items.Strings[ListBox1.ItemIndex];
end;
end.
```

หนังสืออ้างอิง

- Jeff Duntemann,Jim Mischel and Don Taylor," DELPHI PROGRAMMING EXPLORER", The Coriolis Group, 1995
- Dorothy Elizabeth Robling Denning," Cryptography and Data Security ", PURDUE UNIVERSITY
- Jennifer Seberry,Josef Pieprzyk," Cryptography An Introduction To Computer Security ",University College The University Of University Of New South Wales Australian Defence Force Academy
- บุญเลิศ เอี่ยมทัศนา," โปรแกรม Delphi : เริ่มต้น ",ไมโครคอมพิวเตอร์,ฉบับที่ 121,2538,หน้า 233-240
- ยุทธนา สวนสุข," การเข้ารหัสข้อมูลในระบบอินเทอร์เน็ต ",อินเทอร์เน็ต-อินทราเน็ต,ฉบับที่ 1,2539,หน้า 65-73

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้จะไม่สามารถทำให้เสร็จสิ้นสุมบูรณ์ หากไม่ได้รับความช่วยเหลือจาก **ดร. วรวัฒน์ ลิ้มโภคา** (อาจารย์ที่ปรึกษา) ผู้ซึ่งคอยให้ข้อคิดเห็น,ข้อเสนอแนะ,แนวทางในการพัฒนาและการวางแผนงานในหลาย ๆ ด้าน ทางผู้จัดทำต้องขอขอบพระคุณอย่างสูงไว้ ณ โอกาสนี้ด้วยครับ