บทที่ 2

แนวคิดและทฤษฎีที่เกี่ยวข้อง

ด้วยระบบเครือข่ายในปัจจุบันที่ครอบคลุมถึงกันอย่างกว้างขวาง และมีเครื่องคอมพิวเตอร์อยู่มากมายเชื่อมต่อเข้ากับระบบเครือข่ายนี้ ทำให้สามารถติดต่อถึงกันได้ไม่ว่าจะอยู่ห่างไกลกันเพียงใด ผู้ใช้สามารถเข้าใช้งานบนเครื่องคอมพิวเตอร์เครื่องใดๆ ก็ได้จากทุกจุดที่อยู่ในระบบเครือข่าย โดยใช้โปรแกรมประยุกต์ในการเข้าใช้งานระบบจากระยะทางไกล (remote login) ซึ่งเป็นโปรแกรมที่มีการใช้งานกันอย่างแพร่หลาย

โปรโตคอลเทลเนต (telnet protocol)

การเข้าใช้งานระบบจากระยะทางไกล เป็นโปรแกรมประยุกต์หนึ่งในชุดของโปรแกรมประยุกต์ต่างๆในอินเตอร์เนตโปรโตคอล (Internet Protocol - IP) ที่รู้จักกันในนามทีซีพีไอพี (TCP/IP) แทนที่จะต้องมีการวางสายต่อโดยตรงจากเทอร์มินอลไปยังเครื่องคอมพิวเตอร์ (host) ทุกตัวที่ต้องการใช้งาน ก็ทำได้โดยการเข้าใช้งานบนเครื่องหนึ่งและสามารถผ่านเข้าไปใช้งานยังเครื่องอื่นๆในระยะไกลได้ โดยผ่านทางระบบเครือข่าย

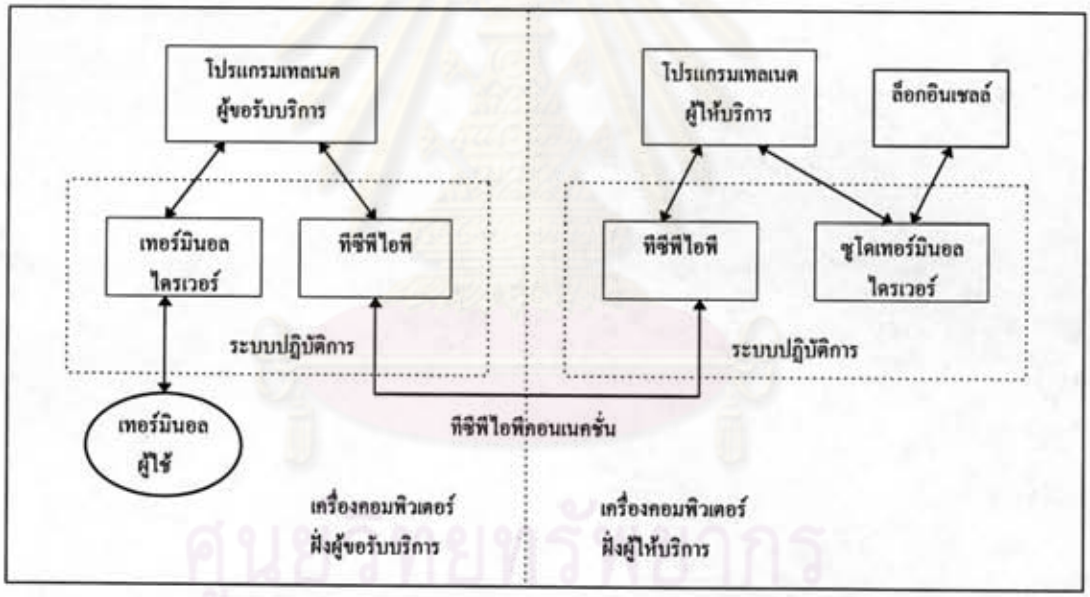
โปรแกรมประยุกต์ที่จัดให้เราสามารถใช้งานบนเครื่องคอมพิวเตอร์จากระยะทางไกล ที่มีการใช้มากที่สุด ในชุดของโปรแกรมประยุกต์ทั่วไปของทีซีพีไอพี คือ

1. เทลเนต (telnet) เป็นโปรแกรมประยุกต์ที่เป็นมาตรฐาน สำหรับทุกระบบที่มีการใช้งานบนโปรโตคอลทีซีพีไอพี เทลเนตสามารถทำงานร่วมกันระหว่างเครื่องคอมพิวเตอร์ที่มีระบบปฏิบัติการที่แตกต่างกันได้ โดยใช้วิธีการเจรจาทางเลือก (obtion negotiation) ระหว่างผู้ขอรับบริการหรือที่เรียกว่า ไคลเอนต์ (client) และผู้ให้บริการคือ เซอร์เวอร์ (server) เพื่อตกลงกันในทางเลือกต่างๆ ให้ทั้งสองฝ่ายจะได้ทำงานได้ถูกต้องตรงกัน

2. อาร์ล็อกอิน (rlogin) เป็นโปรแกรมประยุกต์ที่ถูกพัฒนา เพื่อใช้งานระหว่างเครื่องคอมพิวเตอร์ที่เป็นระบบปฏิบัติการยูนิกซ์ด้วยกัน เริ่มแรกพัฒนาบนระบบปฏิบัติการยูนิกซ์ของเบิร์กเลย์ (Berkeley UNIX)

เทลเนตเป็นโปรแกรมประยุกต์ ที่มีการใช้งานมาอย่างยาวนาน ในบรรดาโปรแกรมประยุกต์ต่างๆที่มีในอินเตอร์เนต โดยในปี ค.ศ.1969 เริ่มมีการใช้งานโปรแกรมเทลเนตบนเครือข่ายอาร์พาเนต (ARPANET) ซึ่งเป็นเครือข่ายของกระทรวงกลาโหมของสหรัฐอเมริกา โดยเทลเนตเป็นชื่อย่อของเทเลคอมมูนิเคชั่นเนตเวิร์คโปรโตคอล (telecommunications network protocol)

รูปที่ 2.1 แสดงลักษณะการทำงานพื้นฐานของ โปรแกรมเทลเนตผู้ขอรับบริการ (telnet client) และโปรแกรมเทลเนตผู้ให้บริการ (telnet server) ที่เกี่ยงข้องกับองค์ประกอบต่างๆ



รูปที่ 2.1 แสดงการทำงานของโปรโตคอลเทลเนต

ตามรูปที่ 2.1 จะเห็นได้ว่า

1. โปรแกรมเทลเนตผู้ขอรับบริการ ติดต่อกับทั้งเทอร์มินอลของผู้ใช้ และโปรโตคอลทีซีพีไอพี โดยปกติแล้วทุกสิ่งที่เราพิมพ์จะถูกส่งผ่านการติดต่อสื่อสารของโปรโตคอลทีซีพี และทุกสิ่งที่รับจากโปรโตคอลทีซีพีจะแสดงออกทางเทอร์มินอล

2. โปรแกรมเทลเนตผู้ให้บริการ จะติดต่อผ่านซูโดเทอร์มินอลดีไวซ์ (pseudo-terminal device) ในการใช้งานบนล็อกอินเชลล์ (login shell) เช่นเดียวกันกับโปรแกรมอื่น ๆ ที่

ทำงานโดยล็อกอินเชลล์ จะเป็นในลักษณะทำงานติดต่อผ่านเทอร์มินอลดีไวซ์นี้ เช่นโปรแกรมประยุกต์ประเภทฟูลสกรีนเอดิเตอร์ (full-screen editors)

3. การติดต่อสื่อสารระหว่างโปรแกรมเทลเนตผู้ขอรับบริการ และโปรแกรมเทลเนตผู้ให้บริการ เกิดขึ้นด้วยการติดต่อของโปรโตคอลทีซีพีเพียงครั้งเดียวเท่านั้น ดังนั้นทั้งสองฝั่งต้องมีวิธีการ ในการจำแนกประเภทระหว่างข้อมูลที่อาจเป็นชุดของคำสั่งของโปรโตคอลเทลเนต เช่นในการเจรจาทางเลือก หรืออาจเป็นข้อมูลของผู้ใช้งาน

4. จากรูปแสดงให้เห็นในกรอบที่เป็นเส้นประ ในส่วนของเทอร์มินอลไดรเวอร์และซูโดว์เทอร์มินอลไดร์เวอร์ พร้อมทั้งทีซีพีไอพี เป็นส่วนหนึ่งของระบบปฏิบัติการ (operation system kernel) สำหรับโปรแกรมเทลเนต ทั้งผู้ขอรับบริการและผู้ให้บริการ เป็นส่วนของโปรแกรมประยุกต์ของผู้ใช้

5. จากรูปแสดงส่วนของล็อกอินเชลล์ เพื่อแสดงว่าการที่จะสามารถใช้งานใดๆ บนเครื่องคอมพิวเตอร์ได้นั้น ผู้ใช้จะต้องมีสิทธิ์โดยมีบัญชีชื่อ (account) บนเครื่องด้วย จึงจะสามารถเข้าไปใช้งานบนระบบได้

โปรโตคอลต่างๆ ที่มีในอินเตอร์เนต จะมีข้อกำหนดอธิบายไว้ในอาร์เอฟซี (Request for Comments : RFCs) ซึ่งเป็นเอกสารที่จัดทำขึ้นไว้ใช้ในการอ้างอิงและเป็นมาตรฐานในการปฏิบัติร่วมกัน สำหรับโปรโตคอลเทลเนตอธิบายไว้ในอาร์เอฟซี 854 (Postel and Reynolds ,1983) และใน TCP/IP Illustrated, Volume 1 (Stevens,1994) มีรายละเอียด ไว้ดังนี้

1. เนตเวิร์คเวอร์ชวลเทอร์มินอล (Network Virtual Terminal - NVT) โปรโตคอลเทลเนตได้ถูกออกแบบและพัฒนา ให้สามารถทำงานร่วมกันได้ระหว่างเครื่องคอมพิวเตอร์ต่างๆ ที่มีระบบปฏิบัติการที่แตกต่างกัน และยังสามารถทำงานกับเทอร์มินอลแบบใดก็ได้ โดยได้กำหนดรูปแบบของเทอร์มินอลที่เรียกว่า เนตเวิร์คเวอร์ชวลเทอร์มินอล หรือ NVT โดยใช้วิธีการสมมาตรข้อมูล (symmetric data representation) แทนรูปแบบข้อมูลของเทอร์มินอลผู้ส่งเปลี่ยนไปเป็นรูปแบบข้อมูลของ NVT ส่งผ่านระบบเครือข่ายไปยังผู้รับ และเปลี่ยนจากรูปแบบข้อมูลของ NVT เมื่อรับข้อมูลจากระบบเครือข่ายมาเป็นรูปแบบข้อมูลของเทอร์มินอลของผู้รับเพื่อใช้งานต่อไป

ในชุดของ NVT จะแทนรหัสแอสกีข้อมูล 1 อักขระด้วย 7 บิต ซึ่งเป็นลักษณะที่ถูกใช้เหมือนกับโปรแกรมประยุกต์โดยทั่วไปในโปรโตคอลอินเตอร์เนต ข้อมูลแต่ละอักขระซึ่งใช้ 7 บิตนี้จะถูกส่งออกไปเป็น 8 บิตต่อ 1 ไบต์ โดยบิตที่ 8 มีค่าเป็น 0

2. คำสั่งของโปรโตคอลเทลเนต (telnet command) โปรโตคอลเทลเนตจะใช้อักขระของ 0xff (เลขฐานสิบคือ 255) ซึ่งเรียกว่า Interpret As Command หรือ IAC เป็นอักขระนำในชุดของคำสั่ง อักขระถัดจาก IAC จะเป็นอักขระของคำสั่งต่างๆ ที่ต้องการ ดังที่ได้กล่าวแล้วว่า NVT ใช้ 7 บิตต่อหนึ่งไบต์ ในการแทนข้อมูลที่ต้องการส่ง สำหรับข้อมูลที่เกินช่วงของรหัสแอสกี 7 บิตนั้นโปรโตคอลเทลเนตจะมีทางเลือกของการเจรจาที่สามารถส่งข้อมูล 8 บิตได้คือไบนารี่ออปชั่น (binary option) เป็นทางเลือกที่สามารถส่งข้อมูลที่แทนด้วยรหัสแอสกีเกิน 7 บิต

ในชุดคำสั่งที่ต้องการส่งอาจเป็นได้ทั้งการเสนอทางเลือก การตอบรับหรือปฏิเสธทางเลือกต่างๆ โปรโตคอลเทลเนตได้กำหนดเป็นมาตรฐานของการเจรจาทางเลือก โดยใช้อักขระสามตัวนี้ในชุดของคำสั่งที่ส่งผ่านระหว่างกัน โดยมีรูปแบบดังนี้

IAC verb option

ตารางที่ 2.1 แสดงรายชื่อของคำสั่งต่างๆ ของโปรโตคอลเทลเนต มีหลายคำสั่งในตารางที่มีการใช้งานไม่มาก จะมีคำสั่งสำคัญๆที่มีการใช้งานอยู่เป็นประจำ ซึ่งจะอธิบายรายละเอียดสำหรับคำสั่งที่สำคัญเหล่านี้ต่อไป

Name	Code (decimal)	Description
EOF	236	end-of-file
SUSP	237	suspend current process (job control)
ABORT	238	abort process
EOR	239	end of record
SE	240	suboption end
NOP	241	no operation
DM	242	data mark
BRK	243	break
IP	244	interrupt process
AO	245	abort output
AYT	246	are you there?
EC	247	escape character
EL	248	erase line
GA	249	go ahead
SB	250	suboption begin
WILL	251	option negotiation
WONT	252	option negotiation
DO	253	option negotiation
DONT	254	option negotiation
IAC	255	data byte 255

ตารางที่ 2.1 คำสั่งต่างๆ ของโปรโตคอลเทลเนต

3. การเจรจาทางเลือก (option negotiation) โปรโตคอลเทลเนตเริ่มต้นการทำงานด้วยการเจรจาทางเลือกต่างๆ ระหว่างโปรแกรมเทลเนตผู้ขอรับบริการและโปรแกรมเทลเนตผู้ให้บริการ สิ่งแรกที่มีการแลกเปลี่ยนผ่านการติดต่อสื่อสารของโปรโตคอลทีซีพี ระหว่างผู้ขอรับบริการและผู้ให้บริการ ก็คือทางเลือกต่างๆ เหล่านี้ การเจรจาทางเลือกเป็นลักษณะสมมาตรกันระหว่างสองฝ่าย โดยที่แต่ละฝ่ายสามารถส่งการร้องขอทางเลือกใดๆ ไปยังอีกฝ่ายหนึ่งได้

ทั้งสองฝ่ายจะส่งรูปแบบของการร้องขอในทางเลือกต่างๆ โดยโปรโตคอลเทลเนตกำหนดกริยาที่ใช้มี 4 คำคือ

1) WILL ผู้ส่งต้องการเป็นผู้กระทำในทางเลือก
2) DO ผู้ส่งต้องการให้ผู้รับกระทำในทางเลือก
3) WONT ผู้ส่งไม่ต้องการกระทำในทางเลือก
4) DONT ผู้ส่งไม่ต้องการให้ผู้รับกระทำในทางเลือก

เพราะว่ากฎเกณฑ์ของโปรโตคอลเทลเนต อนุญาตให้ฝ่ายหนึ่งสามารถยอมรับหรือปฏิเสธการเสนอการกระทำในทางเลือกของอีกฝ่ายได้ แต่กำหนดให้ฝ่ายหนึ่งต้องยอมตามการเสนอไม่กระทำในทางเลือกจากอีกฝ่ายหนึ่ง ด้วยกฎเกณฑ์ดังกล่าว แสดงให้เห็นรูปแบบที่เป็นไปได้ทั้งหมด ในการเสนอการเจรจาทางเลือก ดังแสดงไว้ในตารางที่ 2.2 ข้างล่างนี้

Sender		Receiver	Description
1. WILL	→		ผู้ส่งต้องการกระทำในทางเลือก
	←	DO	ผู้รับตอบตกลง
2. WILL	→		ผู้ส่งต้องการกระทำในทางเลือก
	←	DONT	ผู้รับตอบปฏิเสธ
3. DO	→		ผู้ส่งต้องการให้ผู้รับกระทำในทางเลือก
	←	WILL	ผู้รับตอบตกลง
4. DO	→		ผู้ส่งต้องการให้ผู้รับกระทำในทางเลือก
	←	WONT	ผู้รับตอบปฏิเสธ
5. WONT	→		ผู้ส่งไม่ต้องการกระทำในทางเลือก
	←	DONT	ผู้รับต้องตอบตกลงไม่กระทำในทางเลือก
6. DONT	→		ผู้ส่งไม่ต้องการให้ผู้รับกระทำในทางเลือก
	←	WONT	ผู้รับต้องตอบตกลงไม่กระทำในทางเลือก

ตารางที่ 2.2 ลักษณะของการเจรจาทางเลือก

การเจรจาทางเลือกจะใช้ 3 อักขระประกอบด้วยอักขระ IAC ตามด้วยคำกริยาคำใดคำหนึ่งใน 4 คำ และอักขระ ID ที่ระบุทางเลือกที่ต้องการกระทำหรือไม่ต้องการ ในปัจจุบันมีทางเลือกต่างๆ มากกว่า 40 ทางเลือกที่ใช้กันอยู่ในโปรโตคอลเทลเนต ตารางที่ 2.3 แสดงอาร์เอฟซีต่างๆ ที่อธิบายรายละเอียดขั้นตอนการทำงานในแต่ละทางเลือก

Option ID (decimal)	Name	RFC
1	echo	857
3	suppress go ahead	858
5	status	859
6	timing mark	860
24	terminal type	1091
31	window size	1073
32	terminal speed	1079
33	remote flow control	1372
34	linemode	1184
36	environment variables	1408

ตารางที่ 2.3 อาร์เอฟซีต่างๆ ที่อธิบายรายละเอียดของแต่ละทางเลือก

โปรโตคอลเทลเนต ได้ถูกออกแบบให้การเจรจาทางเลือกระหว่างโปรแกรมเทลเนตผู้ขอรับบริการและโปรแกรมเทลเนตผู้ให้บริการเป็นลักษณะสมมาตรกัน กล่าวคือ เมื่อฝ่ายหนึ่งเริ่มต้นส่งการร้องขอในการกระทำหรือไม่กระทำทางเลือกใดทางเลือกหนึ่ง อีกฝ่ายจะต้องตอบกลับการเจรจาทางเลือกนั้นด้วยการกระทำตามหรือปฏิเสธไม่กระทำตามทางเลือกนั้น ซึ่งจะแตกต่างจากโปรแกรมประยุกต์อาร์ล็อกอิน ที่การร้องขอกระทำทางเลือกเกิดขึ้นจากฝ่ายหนึ่งฝ่ายใดเท่านั้น

4. การเจรจาในส่วนย่อยของทางเลือก (suboption negotiation) มีทางเลือกของการเจรจาที่ต้องการข้อมูลอื่นๆ มากกว่าเพียงแค่การยอมรับที่จะกระทำตามหรือไม่กระทำตามในทางเลือกเท่านั้น ตัวอย่างเช่นในการเจรจาทางเลือกชนิดของเทอร์มินอล (terminal type) ต้องการข้อมูลที่ระบุชนิดของเทอร์มินอลด้วย โดยข้อมูลชนิดของเทอร์มินอลนี้จะถูกส่งจากโปรแกรมเทลเนตผู้ขอรับบริการไปยังโปรแกรมเทลเนตผู้ให้บริการ เพื่อที่สามารถทำงานได้ถูกต้องตามชนิดของเทอร์มินอลที่ใช้ ในขั้นตอนของการเจรจาทางเลือกโปรโตคอลเทลเนตจึงได้กำหนดการเจรจาในส่วนย่อยของทางเลือกนี้ขึ้นมา เพื่อสามารถเจรจาข้อมูลในส่วนที่ต้องการมากขึ้นได้

การเจรจาในส่วนย่อยของทางเลือก จะใช้อักขระ SB เป็นคำสั่งที่บอกจุดเริ่มต้นการเจรจาในส่วนย่อย ตามด้วยอักขระทีแสดงทางเลือกใดๆ และอักขระที่ใช้ในการเจรจาข้อมูลที่

ต้องการ ตัวอย่างของการเจรจาในส่วนย่อยของทางเลือกชนิดของเทอร์มินอล ในการส่งชนิดของเทอร์มินอล VT100 ไปให้อีกฝ่ายหนึ่ง แสดงดังนี้

<IAC, SB, 24, 0, 'V', 'T', '1', '0', '0', IAC, SE>

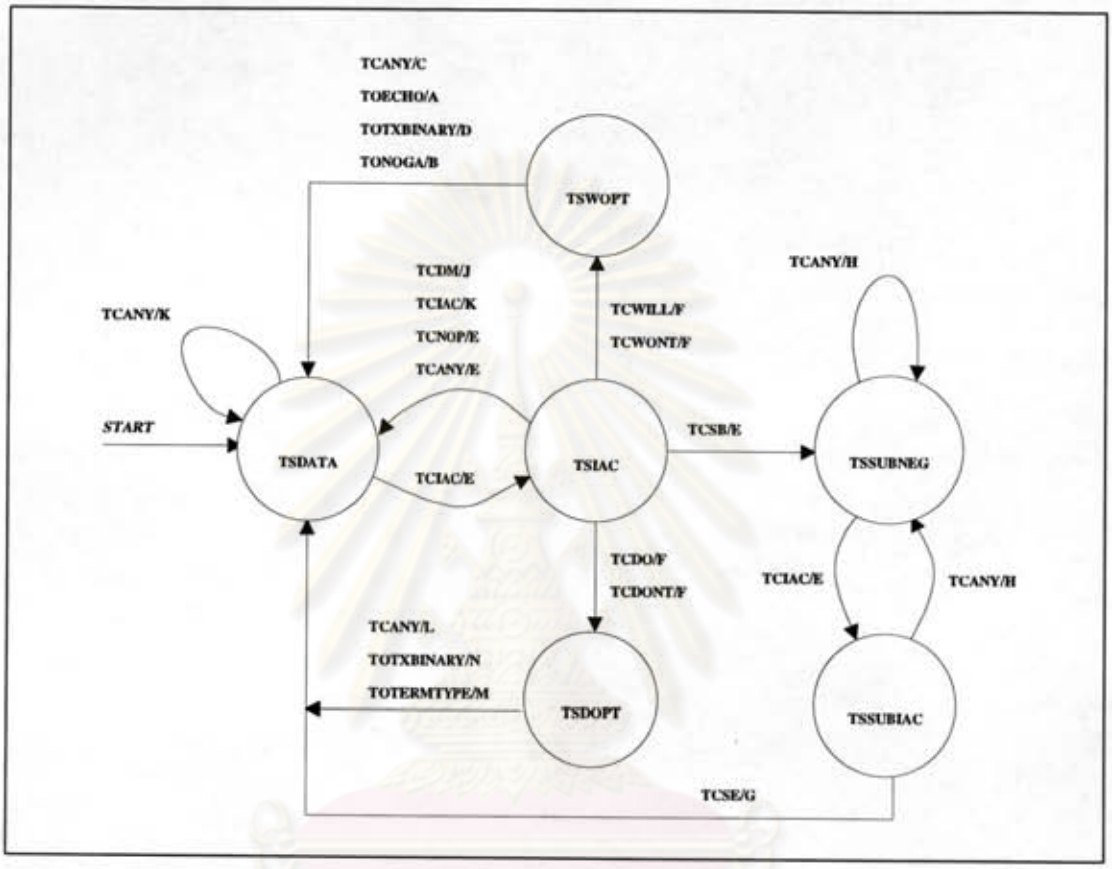
การเจรจาในส่วนย่อยนี้ จะใช้อักขระ SE ในการบอกจุดสิ้นสุดของการเจรจาในส่วนย่อย จะเห็นได้ว่าอักขระ SE จะถูกนำด้วยอักขระ IAC เช่นเดียวกันกับอักขระ SB

สำหรับรายละเอียดของการเจรจาทางเลือกต่างๆของโปรโตคอลเทลเนต มีอยู่ในอาร์เอฟซีมากมาย ดังที่ได้แสดงไว้ส่วนหนึ่งในตารางที่ 2.3 และยังมีอาร์เอฟซีที่ประกาศออกมาใหม่โดยเป็นการแก้ไขในรายละเอียดของทางเลือกเก่า หรือประกาศทางเลือกใหม่ออกมา ซึ่งมีผลทำให้โปรแกรมเทลเนตทำงานได้ดียิ่งขึ้น

ในการพัฒนาโปรแกรมประยุกต์เทลเนตเพื่อนำมาใช้ประโยชน์ ขั้นตอนการออกแบบให้สามารถทำงานได้ถูกต้องตรงกับรายละเอียดต่างๆที่กำหนดไว้ในโปรโตคอลเทลเนตเป็นสิ่งสำคัญ ใน UNIX Network Programming (Stevens,1991) ได้อธิบายรายละเอียดขั้นตอนการทำงานของโปรแกรมเทลเนตที่สร้างขึ้นโดยใช้หลักการของไฟไนต์สเตทเมชีน (finite state machine) ซึ่งจัดว่าเป็นเครื่องมือที่ช่วยในการพัฒนาได้เป็นอย่างดี เพราะมีการกำหนดขั้นตอนวิธีในการทำงานได้ตามรูปแบบของโปรโตคอล และที่สำคัญคือสามารถเปลี่ยนลำดับขั้นตอนที่กำหนดในไฟไนต์สเตทเมชีนนี้ เป็นชุดคำสั่งการทำงานของคอมพิวเตอร์หรือโปรแกรมคอมพิวเตอร์ เพื่อให้ทำงานได้ตามข้อกำหนดของโปรโตคอลเทลเนตได้อย่างถูกต้อง

5. ไฟไนต์สเตทเมชีน (Finite State Machine - FSM) ในขั้นตอนของการพัฒนาโปรแกรมประยุกต์เทลเนตขึ้นมาใช้งาน ได้อธิบายลำดับการทำงานของโปรแกรมเทลเนต โดยนำหลักการของไฟไนต์สเตทเมชีน มาเป็นเครื่องมือที่ช่วยในการพัฒนาโปรแกรม ข้อมูลส่วนใหญ่ที่ส่งผ่านระหว่างโปรแกรมเทลเนตผู้ขอรับบริการและผู้ให้บริการจะเป็นข้อมูลอักขระเดียว (character-oriented) และนอกจากข้อมูลของผู้ใช้แล้ว ยังมีข้อมูลที่เป็นคำสั่งและข้อมูลที่ใช้ในการควบคุมการทำงานของโปรโตคอลเทลเนตส่งผ่านด้วยเช่นกัน โปรโตคอลเทลเนตจะใช้กลไกสำคัญนี้ช่วยในการทำงาน ในการแปลชุดคำสั่งต่างๆ ที่เกิดขึ้น ดังที่ได้กล่าวแล้วว่าไฟไนต์สเตทเมชีน จัดเป็นเครื่องมือที่มีการกำหนดรูปแบบการทำงานตามรายละเอียดที่กำหนดไว้ในโปรโตคอล

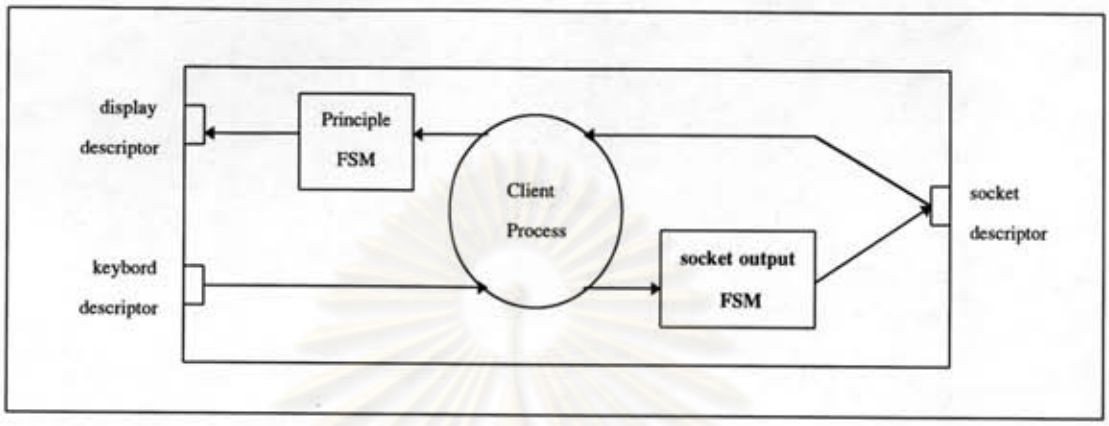
จะแสดงให้เห็นอย่างชัดเจนเมื่อต้องการส่งสายข้อมูลของคำสั่งและการแปลคำสั่ง เมื่อได้รับไปตามลำดับในสายข้อมูล



รูปที่ 2.2 แสดงไฟไนต์สเตทเมชีนของโปรโตคอลเทลเนต

รูปที่ 2.2 เป็นตัวอย่างของไฟไนต์สเตทเมชีน แสดงขั้นตอนการทำงานของโปรโตคอลเทลเนต การแปลคำสั่งในสายของข้อมูลที่รับเข้ามา การเปลี่ยนจากสภาวะหนึ่งไปอีกสภาวะหนึ่งจากรูปสภาวะเริ่มต้นของกลไกกำหนดไว้ที่สภาวะ TSDATA เมื่อเริ่มทำงานสภาวะนี้จะอยู่ในสถานการณ์ที่คาดว่าข้อมูลที่รับส่วนใหญ่เป็นข้อมูลปกติของผู้ใช้ และจะทำการแสดงผลออกทางเทอร์มินอล แต่ในกรณีที่รับข้อมูลที่เป็นคำสั่งเข้ามาคือ TCIAC ซึ่งเป็นอักขระที่แสดงจุดเริ่มต้นสายข้อมูลของคำสั่ง ก็จะเปลี่ยนสภาวะไปเป็นสภาวะ TSIAC และเริ่มทำการแปลคำสั่งต่างๆ ที่ตามมา ถ้าข้อมูลที่ตามมาเป็นคำกริยาต่างๆ เช่น TCDO สภาวะจะเปลี่ยนไปสู่สภาวะของ TSDOPT ต่อไป การเปลี่ยนในแต่ละสภาวะเมื่อได้รับข้อมูลใดๆ จะมีการทำงานบางอย่างเกิดขึ้นพร้อมกัน เช่นเมื่อได้รับอักขระ TCDO มีการทำงานในฟังก์ชั่นเอฟเกิดขึ้น ซึ่งเป็นการบันทึกข้อมูลทางเลือกของการเจรจาที่เกิดขึ้น กำหนดด้วยอักขระในลำดับถัดไป จากรูปแสดงให้เห็นถึงการทำงานรวมทั้งหมด

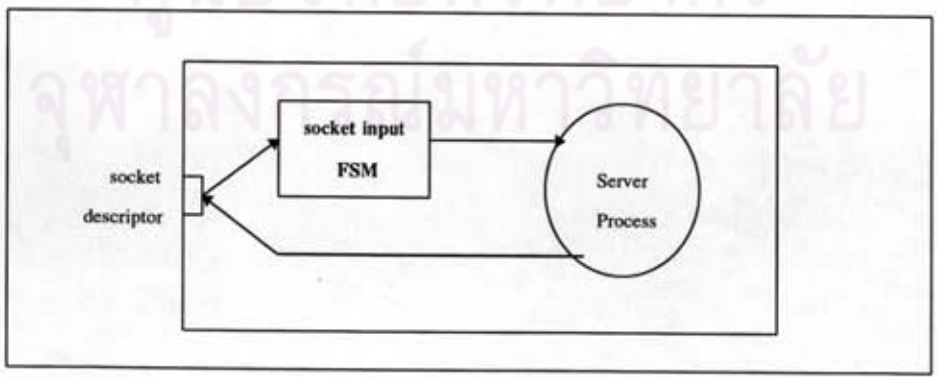
ของไฟไนต์สเตทเมชีน ในส่วนของการรับสายข้อมูลที่เข้ามา การจัดการกับคำสั่งที่ได้รับและการเปลี่ยนจากสภาวะหนึ่งไปสู่อีกสภาวะหนึ่ง เพื่อทำงานต่อไป



รูปที่ 2.3 แสดงไฟไนต์สเตทเมชีนของเทลเนตผู้ขอรับบริการ

รูปที่ 2.3 แสดงให้เห็นภาพรวมของขั้นตอนการทำงานของโปรแกรมเทลเนตผู้ขอรับบริการ โดย principle FSM จัดการข้อมูลที่รับจากระบบเครือข่ายผ่านการติดต่อสื่อสารของโปรโตคอลทีซีพี เพื่อแสดงผลทางเทอร์มินอลของผู้ใช้ และใช้ socket output FSM จัดการข้อมูลที่ผู้ใช้พิมพ์ผ่านคีย์บอร์ดส่งไปยังผู้ให้บริการ ผ่านทางระบบเครือข่ายโปรโตคอลทีซีพีเช่นกัน

ในรูปที่ 2.4 แสดงขั้นตอนการทำงานของโปรแกรมเทลเนตผู้ให้บริการใช้ socket input FSM จัดการข้อมูลที่รับจากระบบเครือข่ายโปรโตคอลทีซีพี ส่งให้เครื่องคอมพิวเตอร์ผู้ให้บริการเพื่อประมวลผล และส่งผลลัพธ์ที่เกิดขึ้นผ่านระบบเครือข่ายกลับไปให้โปรแกรมเทลเนตผู้ขอรับบริการต่อไป



รูปที่ 2.4 แสดงไฟไนต์สเตทเมชีนของเทลเนตผู้ให้บริการ

ดังที่ได้กล่าวไว้แล้วว่า เราสามารถเปลี่ยนลำดับขั้นตอนต่างๆ ในไฟไนต์สเตทเมชีน มาสร้างเป็นชุดคำสั่งในการทำงานของคอมพิวเตอร์ เพราะไฟไนต์สเตทเมชีนจัดเป็นเครื่องมือที่ใช้ในการกำหนดขั้นตอนการทำงานตามรายละเอียดต่างๆ ที่กำหนดขึ้นเป็นโปรโตคอล โปรแกรมประยุกต์โดยส่วนใหญ่ ในชุดของโปรโตคอลทีซีพีไอพี จะมีการใช้ไฟไนต์สเตทเมชีนเป็นเครื่องมือในการสร้างโปรแกรมประยุกต์ขึ้นมาใช้งานด้วยเช่นกัน

โดยสรุปแล้ว การทำงานของโปรโตคอลเทลเนต ในแง่ความปลอดภัยของข้อมูลที่ส่งผ่านระหว่างกันในระบบเครือข่าย ซึ่งอยู่ในรูปแบบปกติ (plain text) ดังที่ได้กล่าวแล้วว่าสามารถถูกลอบดักนำไปใช้ประโยชน์ได้ง่าย โดยใช้ซอฟต์แวร์พิเศษที่ทำหน้าที่คล้ายซอฟต์แวร์ตรวจสอบปริมาณการใช้งานในระบบเครือข่าย (network monitor) เช่น โปรแกรมทีซีพีดัมพ์ (tcpdump) หรือ สนูป (snoop) ที่ทำงานบนเครื่องซัน (SUN workstation) สามารถดักแพกเกต (packet) ที่ส่งผ่านไปมาระหว่างเครื่องต่างๆในระบบเครือข่ายอีเทอร์เนตเดียวกันได้

โปรโตคอลเทลเนตมีทางเลือก authentication ในอาร์เอฟซี 1416 (Borman,1993) เป็นการป้องกันข้อมูลของการแสดงสิทธิ์เข้าใช้ระบบคือ ชื่อรหัสการเข้าใช้ และรหัสลับ แต่หลังจากผ่านเข้าไปใช้งานบนระบบได้แล้วข้อมูลต่างๆ ที่ส่งผ่านระบบเครือข่ายจะอยู่ในรูปแบบปกติ

เป็นสิ่งสำคัญที่จะสร้างช่องทางการสื่อสารที่ปลอดภัย ที่สามารถป้องกันข้อมูลได้ตลอดระยะเวลาของการทำงาน มีหลายวิธีที่จะต้องศึกษาทำความเข้าใจ เพื่อเลือกวิธีที่เหมาะสม มาพัฒนาและติดตั้งใช้งานให้ได้ผลดีที่สุด ซึ่งจะได้กล่าวถึงในหัวข้อต่อไป

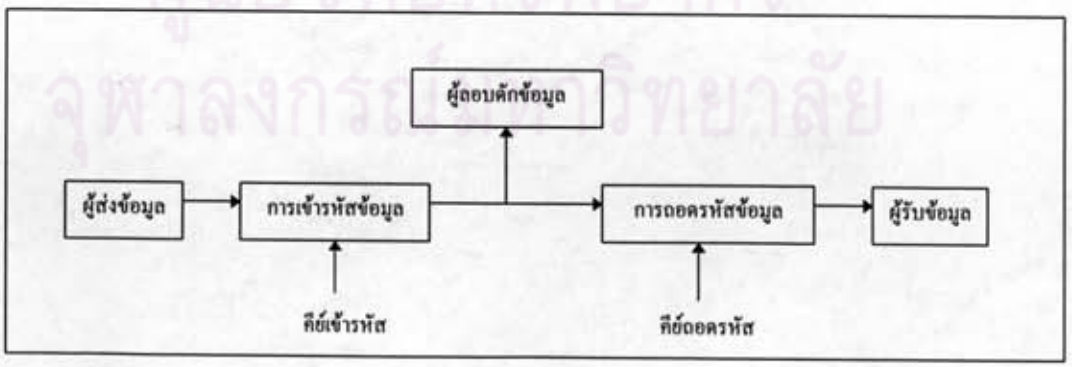
การสื่อสารที่ปลอดภัย (secure communications)

ดังที่ได้กล่าวมาแล้วว่าข้อมูลที่ส่งผ่านระบบเครือข่ายโดยปกติอยู่ในรูปที่เข้าใจได้ ข้อมูลส่วนตัว เช่นรหัสผ่านสำหรับเข้าสู่ระบบ หรือข้อมูลที่เกี่ยวข้องกับการเงิน เช่น หมายเลขบัตรเครดิต เป็นข้อมูลที่มีความสำคัญ หากถูกบุคคลอื่นลอบดักข้อมูลนำไปใช้ อาจทำให้เกิดความเสียหายกับเจ้าของข้อมูลได้ การสร้างช่องทางการสื่อสารที่สร้างความปลอดภัยให้กับข้อมูลของผู้ใช้งาน เป็นสิ่งสำคัญที่สร้างความมั่นใจในการใช้งานให้กับผู้ใช้

1. การสื่อสารข้อมูลที่ปลอดภัย (data communication security) แนวทางสร้างความปลอดภัยให้กับข้อมูลในระบบสื่อสารของข้อมูล แบ่งได้เป็น 2 ลักษณะ คือ

1.1 การป้องกันทางกายภาพ โดยการป้องกันที่ตัวอุปกรณ์ของระบบเครือข่าย (network device) เช่น เราเตอร์ (router) หรือ อีเทอร์เนตสวิทช์ (ethernet switch) เป็นต้น ไม่ให้บุคคลที่ไม่เกี่ยวข้องสามารถเข้าถึงอุปกรณ์เหล่านี้ได้ และป้องกันที่สายสื่อสาร (network cable) โดยวางสายสื่อสารในตำแหน่งที่ยากต่อการลอบดักข้อมูล หรือใช้สายไฟเบอร์ออฟติกเพราะยากต่อการลอบดักข้อมูลเพื่อนำไปใช้

1.2 การป้องกันทางซอฟต์แวร์ โดยการใช้วิธีการเข้ารหัสข้อมูล (data encryption) เปลี่ยนข้อมูลไปอยู่ในรูปที่ไม่สามารถเข้าใจได้ก่อนส่งผ่านระบบเครือข่าย เพื่อป้องกันไม่ให้บุคคลอื่นสามารถนำข้อมูลไปใช้ประโยชน์ได้ ดังรูปที่ 2.5



รูปที่ 2.5 แสดงการเข้ารหัสข้อมูลก่อนส่งผ่านระบบเครือข่าย

การเข้ารหัสข้อมูลสามารถทำได้หลายวิธี อาจใช้โปรแกรมประยุกต์ที่ทำหน้าที่ในการเข้ารหัสข้อมูลในแฟ้มข้อมูล เช่น PGP (Pretty Good Privacy) เป็นต้น โดยผู้ใช้จะต้องศึกษาวิธีการใช้โปรแกรมประยุกต์เหล่านี้ให้เป็น เพื่อสามารถสร้างความปลอดภัยสำหรับข้อมูลของตนได้

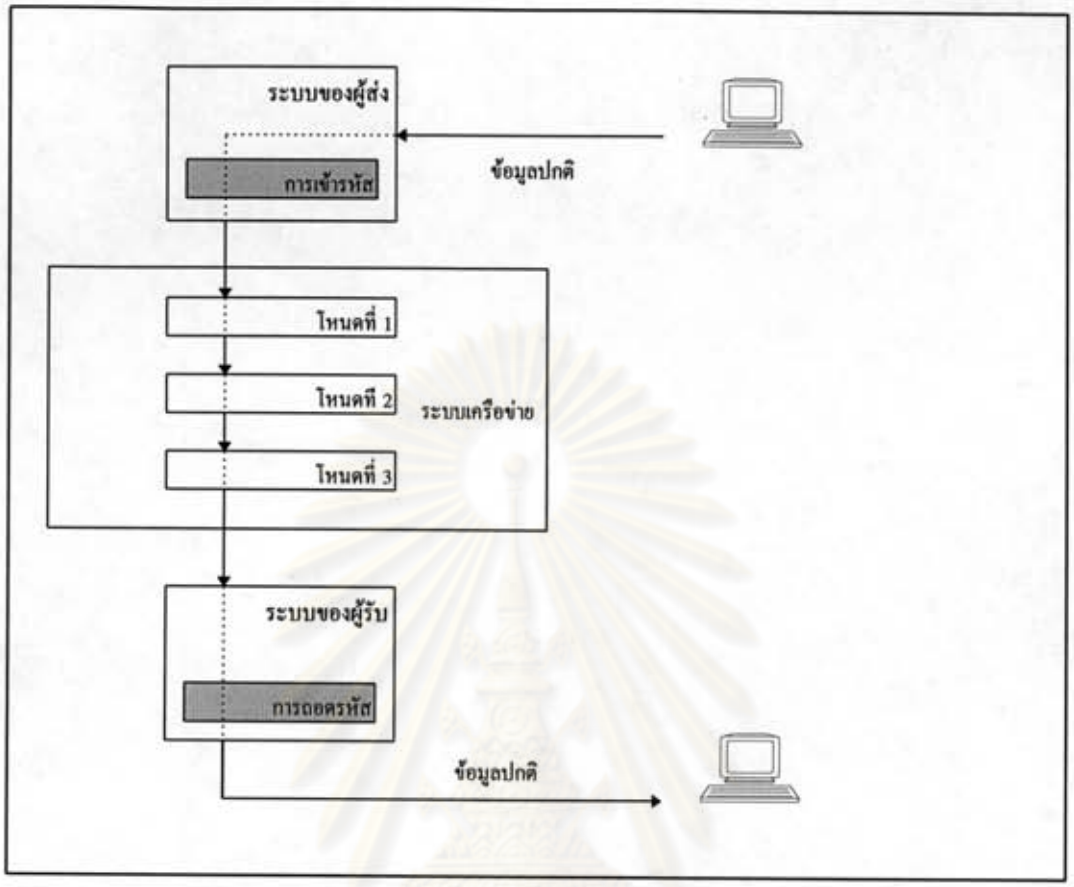
การเข้ารหัสข้อมูลสำหรับโปรโตคอลของการสื่อสารข้อมูล ตามมาตรฐานของโอเอสไอ โมเดล (open systems interconnection model : OSI) เป็นอีกแนวทางหนึ่งที่จัดไว้สำหรับการป้องกันข้อมูลที่สื่อสาร โดยที่ผู้ใช้งานไม่จำเป็นต้องศึกษาวิธีการใช้ หรือรายละเอียดของวิธีการเข้ารหัส การใช้งานระบบจึงเป็นไปในลักษณะเดิม

2. รูปแบบการเข้ารหัสข้อมูลสำหรับโปรโตคอลการสื่อสารข้อมูล (Russell และ Gangemi,1991) เพื่อสร้างช่องทางการสื่อสารที่ปลอดภัยให้แก่ผู้ใช้งานผ่านระบบเครือข่าย การเข้ารหัสข้อมูลสำหรับโปรโตคอลของการสื่อสารข้อมูลสามารถทำได้ 2 ระดับ คือ

2.1 การเข้ารหัสจากต้นทางถึงปลายทาง (end-to-end encryption) บางครั้งเรียกว่า off-line encryption ข้อมูลจะถูกเข้ารหัสเมื่อถูกส่งผ่านระบบเครือข่าย และจะถูกถอดรหัส เมื่อรับข้อมูลจากระบบเครือข่าย

การเข้ารหัสจากต้นทางถึงปลายทางนี้ สามารถวางอยู่ในระหว่างชั้นเครือข่าย (network layer) ขึ้นมาจนถึงชั้นโปรแกรมประยุกต์ (application layer) ถ้าการสร้างโปรแกรมการเข้ารหัสที่ชั้นเครือข่ายหรือชั้นทรานสปอร์ต (transport layer) ในกรณีของการสื่อสารระหว่างระบบที่ต่างกันและมีโปรโตคอลที่ต่างกันด้วยแล้ว การเข้ารหัสจะต้องคำนึงถึงมาตรฐานการสื่อสารของแต่ละโปรโตคอลด้วย เพราะข้อมูลที่ถูกเข้ารหัสประกอบด้วยข้อมูลของผู้ใช้ที่สื่อสารระหว่างกัน และข้อมูลส่วนหัว (header) ของการเข้ารหัส (encapsulation) ของโปรโตคอล ซึ่งแตกต่างกันในแต่ละระบบการสื่อสาร

ถ้าการเข้ารหัสถูกวางอยู่ในชั้นที่สูงกว่า คือชั้นโปรแกรมประยุกต์ หรือชั้นพรีเซนเตชั่น (presentation layer) การเข้ารหัสข้อมูลจะมีความอิสระจากโปรโตคอลของระบบการสื่อสาร มีเฉพาะข้อมูลของผู้ใช้ที่สื่อสารระหว่างกันเท่านั้น ที่เข้าสู่ระบบการเข้ารหัส การเข้ารหัสที่ชั้นโปรแกรมประยุกต์ โดยการนำโปรแกรมประยุกต์ที่ผู้ใช้ในการใช้งานผ่านระบบเครือข่าย เช่น โปรแกรมเทลเนตหรือเอฟทีพี แก้ไขในส่วนของโปรแกรมต้นฉบับ (source program) ทั้งโปรแกรมผู้ขอรับบริการ และโปรแกรมผู้ให้บริการ เพื่อสร้างโปรแกรมใช้งานขึ้นมาใหม่ รูปที่ 2.6 แสดงการเข้ารหัสจากต้นทางถึงปลายทาง



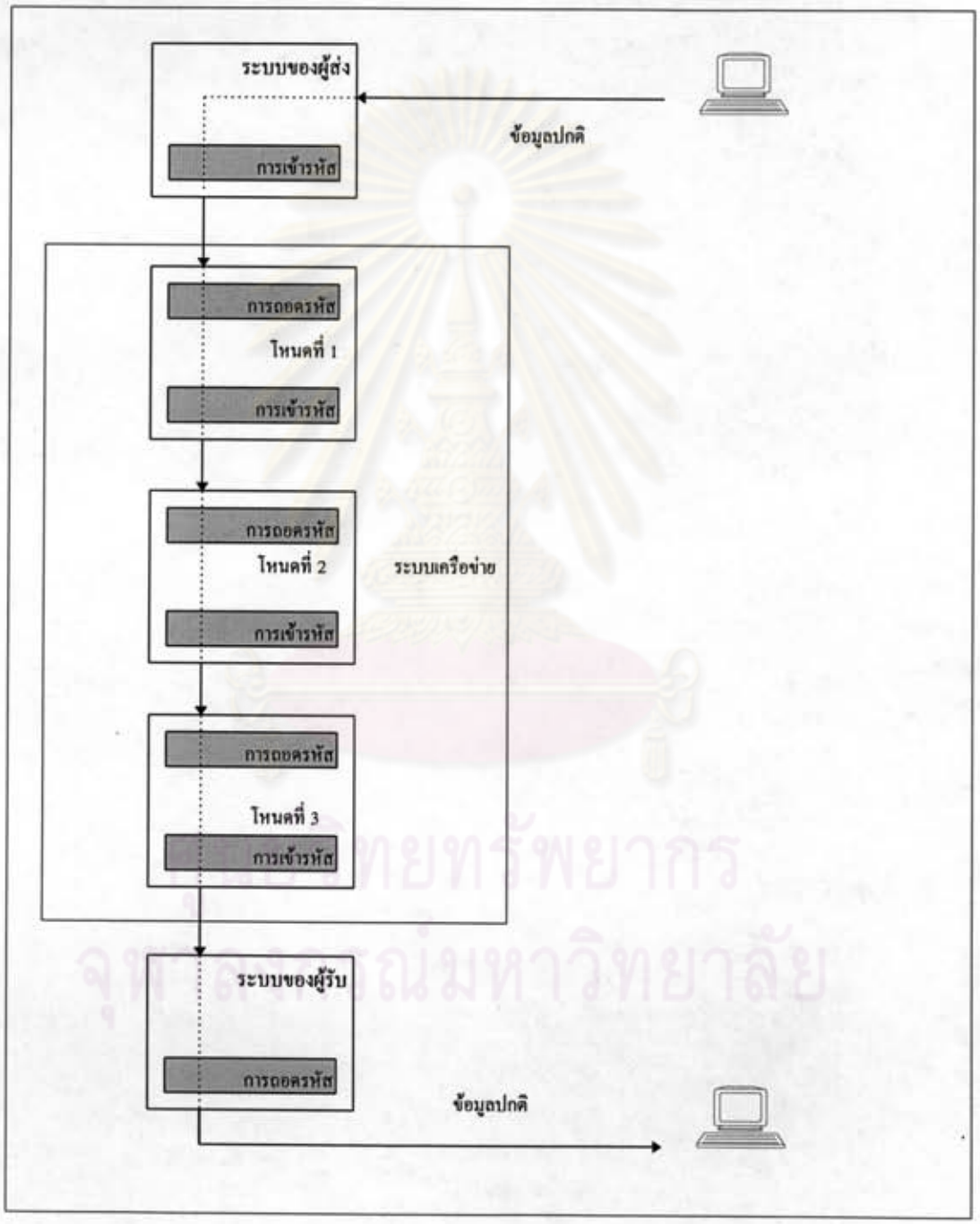
รูปที่ 2.6 แสดงการเข้ารหัสข้อมูลจากต้นทางถึงปลายทาง

สิ่งที่ต้องคำนึงถึงในการเข้ารหัสจากต้นทางถึงปลายทาง คือการจัดการคีย์ที่ใช้ในการเข้ารหัส การแลกเปลี่ยนคีย์ (key exchange) เพื่อที่จุดปลายทางทั้งสองฝ่าย ใช้คีย์ในการเข้ารหัสและถอดรหัสได้ข้อมูลที่ถูกต้องตรงกัน

2.2 การเข้ารหัสระหว่างจุดเชื่อมโยง (link-by-link encryption) หรือเรียกว่า online encryption ข้อมูลถูกเข้ารหัสเมื่อถูกส่งออกไปในเครือข่าย และเมื่อผ่านโหนดแต่ละโหนดจะถูกถอดรหัสแล้วเข้ารหัสใหม่ เพื่อส่งต่อให้โหนดถัดไปในเส้นทาง ดังนั้นข้อมูลจะถูกถอดรหัสและเข้ารหัสซ้ำๆ กันหลายครั้งเมื่อผ่านทุกๆ โหนดของเครือข่ายสื่อสาร ดังรูปที่ 2.7

การเข้ารหัสเกิดขึ้นที่ชั้นกายภาพ (physical layer) โดยการติดตั้งอุปกรณ์การเข้ารหัสที่ชั้นนี้ ข้อมูลทุกชนิดที่ผ่านจะถูกเข้ารหัสทั้งสิ้น รวมทั้งข้อมูลส่วนตัว, ข้อมูลเส้นทางเดิน (routing information) และข้อมูลของโปรโตคอล เป็นต้น การลอบดักข้อมูลไม่สามารถทราบถึงโครงสร้างของข้อมูลว่าเป็นข้อมูลที่สื่อสารระหว่างใคร รวมทั้งขนาดของข้อมูลและวันเวลาที่ส่งข้อมูลด้วย ความปลอดภัยของข้อมูลอย่างสิ้นเชิงเช่นนี้ เรียกว่า traffic-flow security

ข้อมูลที่อยู่ในแต่ละโหนดของระบบเครือข่ายสื่อสารอยู่ในรูปปกติ สิ่งที่ต้องระวังก็คือการป้องกันทางกายภาพ ทุกๆโหนดในระบบเครือข่ายต้องวางอยู่ในตำแหน่งที่ปลอดภัย เพื่อป้องกันไม่ให้บุคคลอื่นที่ไม่ได้เกี่ยวข้องสามารถเข้าถึงโหนดเหล่านี้ได้

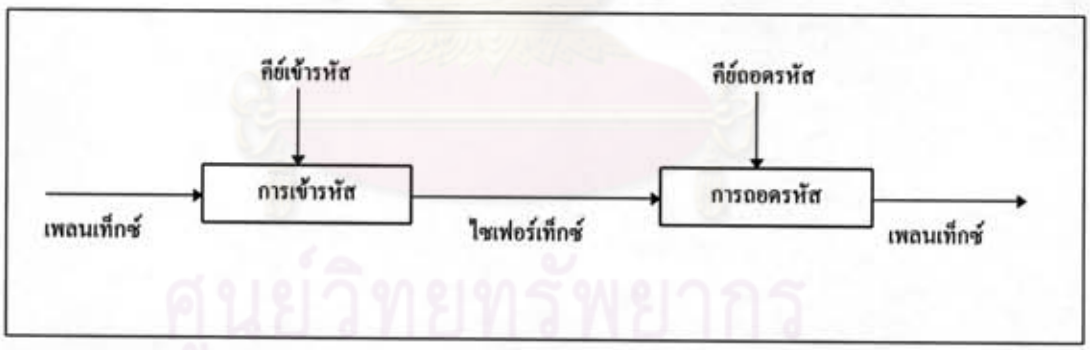


รูปที่ 2.7 แสดงการเข้ารหัสข้อมูลระหว่างจุดเชื่อมโยง

สิ่งที่สำคัญในการใช้วิธีการเข้ารหัสข้อมูล ก็คือการเลือกใช้วิธีการเข้ารหัสที่มีความสามารถป้องกันการลอบดักข้อมูลระหว่างทางและนำไปถอดรหัสได้โดยบุคคลอื่น เป็นความจำเป็นที่จะต้องศึกษาทำความเข้าใจในพื้นฐานการทำงาน รูปแบบและวิธีการในการเข้ารหัส เพื่อหาวิธีที่ดีที่สุดที่มีความแข็งแรงและสามารถป้องกันได้เมื่อนำมาใช้งาน

3. ระบบการเข้ารหัสข้อมูล (cryptography system) การเข้ารหัสข้อมูล (data encryption) เป็นกระบวนการเปลี่ยนรูปของข้อมูลปกติ ที่เรียกว่าเพลนเท็กซ์ (plaintext) ไปเป็นรูปของข้อมูลที่ไม่สามารถเข้าใจได้ ที่เรียกว่าไซเฟอร์เท็กซ์ (ciphertext) หลังจากที่ส่งข้อมูลที่ถูกเข้ารหัสไปยังจุดหมายที่ต้องการแล้ว การถอดรหัส (decryption) เป็นกระบวนการทำงานที่ย้อนกลับโดยเปลี่ยนรูปของข้อมูลที่ถูกเข้ารหัสมาเป็นรูปของข้อมูลปกติที่สามารถเข้าใจได้เหมือนเดิม

กระบวนการหรือกฎเกณฑ์ที่ใช้ในการเข้ารหัส เรียกว่า encryption algorithm โดยส่วนใหญ่กระบวนการเข้ารหัส เป็นการใช้ทฤษฎีทางคณิตศาสตร์ และมีการนำคีย์มาใช้ร่วมกับกระบวนการ ในการเข้ารหัสข้อมูลและถอดรหัสข้อมูล ปกติคีย์เป็นรหัสลับที่ป้องกันข้อมูล ต้องมีการรักษาคีย์อย่างดี เพราะถ้าเสียหายทำให้ไม่สามารถถอดรหัสข้อมูลได้ แสดงได้ดังรูปที่ 2.8



รูปที่ 2.8 แสดงการเข้ารหัส และถอดรหัสข้อมูล

นอกจากคีย์ที่ใช้ในการเข้ารหัสแล้ว ขั้นตอนวิธีหรือรูปแบบที่ใช้ในการเข้ารหัสเป็นสิ่งสำคัญที่ใช้ในการสร้างความแข็งให้กับการเข้ารหัสข้อมูล

3.1 รูปแบบการเข้ารหัส (Russell และ Gangemi,1991) แบ่งได้เป็น 2 ลักษณะดังนี้ คือ

3.1.1 การเข้ารหัสแบบใช้คีย์ส่วนตัว (private-key cryptography) หรือเรียกว่าการเข้ารหัสแบบใช้คีย์เดียว (single key) หรือการเข้ารหัสแบบสมมาตร (symmetric key) การเข้ารหัสวิธีนี้ คีย์ที่ใช้ในการเข้ารหัสและคีย์ที่ใช้ในการถอดรหัสเป็นคีย์เดียวกัน ความปลอดภัยของข้อมูลในการเข้ารหัสด้วยวิธีนี้ ขึ้นอยู่กับคีย์ที่ต้องเก็บรักษาอย่างดี ตัวอย่างของการเข้ารหัสด้วยวิธีนี้ ได้แก่ DES (Data Encryption Standard), IDEA (International Data Encryption Algorithm) เป็นต้น ในวิทยานิพนธ์นี้ใช้อัลกอริทึ่มของ IDEA ในการเข้ารหัสข้อมูล ซึ่งจะกล่าวถึงต่อไป

3.1.2 การเข้ารหัสแบบใช้คีย์สาธารณะ (public-key cryptography) การเข้ารหัสแบบนี้ บางครั้งเรียกว่า การเข้ารหัสแบบอสมมาตร (asymmetric key) โดยจะมีคีย์ 2 ตัว คีย์ที่ใช้ในการเข้ารหัสคือคีย์สาธารณะ (public key) และคีย์ที่ใช้ในการถอดรหัสคือคีย์ส่วนตัว (private key) โดยปกติการเข้ารหัสด้วยวิธีนี้ ผู้ใช้จะเก็บรักษาคีย์ส่วนตัวของตนเป็นความลับ และประกาศคีย์สาธารณะนี้ออกไปให้ผู้อื่นทราบ การเข้ารหัสข้อมูลก็จะใช้คีย์สาธารณะในการเข้ารหัสและส่งไปให้ผู้รับเพื่อใช้คีย์ส่วนตัวในการถอดรหัสข้อมูลนำไปใช้ต่อไป

ทั้งคีย์สาธารณะและคีย์ส่วนตัว เป็นค่าทางคณิตศาสตร์ที่มีความสัมพันธ์กัน การเข้ารหัสด้วยคีย์สาธารณะสามารถถอดรหัสได้ด้วยคีย์ส่วนตัวที่คู่กันเท่านั้น ตัวอย่างของการเข้ารหัสด้วยวิธีนี้ ได้แก่ RSA, Diffie-Hellman Algorithm เป็นต้น

เห็นได้ว่าคีย์ที่ใช้ในการเข้ารหัสมีความสำคัญ ในการสื่อสารข้อมูลระหว่างสองฝ่ายที่ต้องใช้คีย์ร่วมกันในการเข้ารหัสและถอดรหัส คีย์ที่เกิดขึ้นในแต่ละครั้งที่ใช้ในการเข้ารหัสในการสื่อสารข้อมูลระหว่างทั้งสองฝ่าย เรียกว่า เซสชั่นคีย์ (session key)

การแลกเปลี่ยนเซสชั่นคีย์ระหว่างกันเป็นขั้นตอนสำคัญ เพราะถึงแม้ว่าข้อมูลที่ใช้ในการสื่อสารผ่านระบบการเข้ารหัส แต่ถ้าเริ่มต้นส่งผ่านเซสชั่นคีย์ในรูปปกติ และถูกลอบดักนำคีย์ไป ก็สามารถนำข้อมูลที่ผ่านการเข้ารหัสไปถอดรหัสและนำไปใช้ได้ วิธีการในการแลกเปลี่ยนคีย์มีหลายรูปแบบที่ใช้สร้างความปลอดภัยให้กับเซสชั่นคีย์ในการส่งผ่านไปให้อีกฝ่าย

3.2 การแลกเปลี่ยนคีย์ (key exchange) การแลกเปลี่ยนคีย์โดยการเข้ารหัสเซสชั่นคีย์ก่อนส่งผ่านไปยังอีกฝ่าย มีหลายรูปแบบ ในที่นี้กล่าวถึงใน 2 รูปแบบ โดยใช้การเข้ารหัสแบบสมมาตร (symmetric cryptography) และ การเข้ารหัสแบบคีย์สาธารณะ (public cryptography) ดังนี้คือ

3.2.1 การแลกเปลี่ยนคีย์โดยการใช้การเข้ารหัสแบบสมมาตร (key exchange with symmetric cryptography) เป็นการแลกเปลี่ยนคีย์ โดยใช้การเข้ารหัสแบบใช้คีย์เดียวในการเข้ารหัสเซสชั่นคีย์ อธิบายรูปแบบการทำงาน ดังนี้

ยกตัวอย่างให้ A ส่งเซสชั่นคีย์ไปให้ B มีขั้นตอนดังนี้

(1) A ส่งเซสชั่นคีย์ที่ถูกเข้ารหัสด้วยคีย์ของตนไปให้ B

(2) เมื่อ B รับข้อมูลจาก A ทำการเข้ารหัสด้วยคีย์ของตนและส่งกลับคืนไปให้ A

(3) A ได้รับข้อมูลจาก B ถอดรหัสด้วยคีย์ของตน และส่งคืนกลับไปให้ B อีกครั้ง

(4) B รับข้อมูลจาก A อีกครั้ง และทำการถอดรหัสด้วยคีย์ของตน ได้เซสชั่นคีย์ที่ต้องการ

เป็นการสิ้นสุดขั้นตอนการทำงานในการแลกเปลี่ยนคีย์ และมีขั้นตอนการตรวจสอบเซสชั่นคีย์ที่ได้รับว่าถูกต้องตรงกัน ก่อนใช้ในการเข้ารหัสข้อมูลต่อไป

3.2.2 การแลกเปลี่ยนคีย์โดยการใช้การเข้ารหัสแบบคีย์สาธารณะ (key exchange with public-key cryptography) เป็นการแลกเปลี่ยนคีย์ โดยใช้การเข้ารหัสแบบคีย์สาธารณะ ในการเข้ารหัสเซสชั่นคีย์ ก่อนส่งผ่านไปให้อีกฝ่าย อธิบายขั้นตอนการทำงานได้ดังนี้

ตัวอย่างเช่นเดียวกัน A ต้องการส่งเซสชั่นคีย์ไปให้ B เพื่อใช้ในการเข้ารหัสข้อมูลที่ส่งผ่านระหว่างกัน

(1) A ร้องขอให้ B ส่งคีย์สาธารณะมาให้

(2) B ส่งคีย์สาธารณะมาให้ A

(3) A สร้างเซสชั่นคีย์ เข้ารหัสเซสชั่นคีย์ ด้วยคีย์สาธารณะของ B และส่งกลับไปให้ B

(4) B ถอดรหัสข้อมูลที่ได้รับ ด้วยคีย์ส่วนตัว ได้เซสชั่นคีย์ที่ต้องการ

การแลกเปลี่ยนเซสชั่นคีย์ด้วยวิธีนี้ มีความปลอดภัยมากกว่าในวิธีแรก

ในวิทยานิพนธ์นี้ เซสชั่นคีย์ในการเข้ารหัสข้อมูลใช้อัลกอริทึ่มของ IDEA ส่วนการแลกเปลี่ยนเซสชั่นคีย์ใช้อัลกอริทึ่มของ RSA ซึ่งเป็นการเข้ารหัสแบบคีย์สาธารณะ เพื่อส่งผ่านคีย์ที่ถูกเข้ารหัสนี้ไปให้อีกฝ่ายหนึ่ง

3.3 IDEA (Schneier,1994)

3.3.1 ความเป็นมา IDEA ถูกคิดค้นโดย Xuejia Lai และ James Massey ในปีค.ศ. 1990 เริ่มแรกถูกเรียกว่า PES (Proposed Encryption Standard) และในปีถัดมามีการปรับปรุงอัลกอริทึ่มให้ดีขึ้น ถูกเรียกใหม่ว่า IPES (Improved Proposed Encryption Standard) และได้ถูกเปลี่ยนชื่อมาเป็น IDEA (International Data Encryption Algorithm) ในปีค.ศ. 1992

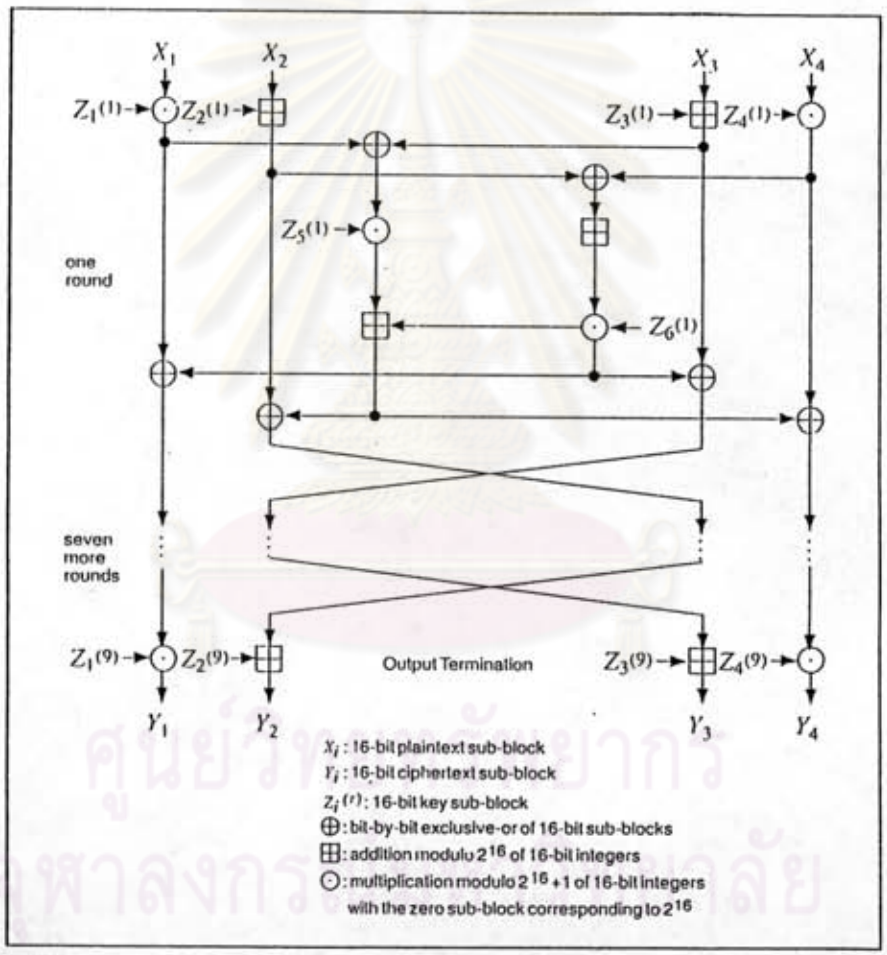
รูปแบบการเข้ารหัสของ IDEA เป็นลักษณะบล็อคไซเฟอร์ (block cipher) โดยมีการทำงานกับข้อมูลเพลนเท็กซ์ 1 บล็อคมีขนาด 64 บิต และคีย์ที่ใช้มีความยาว 128 บิต ใช้อัลกอริทึ่มเดียวกันในการเข้ารหัสและถอดรหัสข้อมูล อัลกอริทึ่มของ IDEA เป็นการผสมการทำงานของกลุ่มพีชคณิตที่แตกต่างกันสามกลุ่ม คือ

1) เอ็กซ์คลูซีฟออร์ (XOR)
2) การบวก (addition modulo 2^{16})
3) การคูณ (multiplication modulo $2^{16} + 1$)

3.3.2 ความเร็วในการทำงานและความปลอดภัยของ IDEA ที่ถูกพัฒนาเป็นซอฟท์แวร์มีความเร็วเท่ากับการทำงานของ DES (Data Encryption Standard) บนเครื่อง 386 ความเร็ว 33 Mhz สามารถเข้ารหัสข้อมูลได้ในอัตรา 880 kbps และถ้าทำงานบนเครื่อง VAX 9000 ความเร็วเพิ่มขึ้นเป็น 4 เท่า

ในแง่ของความปลอดภัย คีย์ที่ใช้ในการเข้ารหัสของ IDEA มีความยาว 128 บิต เป็นสองเท่าของคีย์ที่ใช้ในการเข้ารหัสของ DES การทำ brute-force attack เพื่อถอดคีย์โดยการทดลองเข้ารหัสข้อมูล ต้องทำการเข้ารหัสข้อมูลเป็นจำนวน 2^{128} (10^{38}) ครั้ง ถ้าออกแบบวงจรที่สามารถทำงานได้ในความเร็วของการทดสอบคีย์หนึ่งล้านคีย์ต่อหนึ่งวินาที ต้องใช้เวลาในการถอดคีย์ประมาณ 10 ล้านล้านปี

3.3.3 การเข้ารหัสและการถอดรหัส รูปที่ 2.9 แสดงการทำงานของ IDEA ข้อมูล 64 บิตถูกแบ่งออกเป็น 4 กลุ่มๆละ 16 บิตแทนด้วย X_1, X_2, X_3, และ X_4 โดยข้อมูล 4 กลุ่ม เป็นอินพุทให้กับการทำงานของอัลกอริทึมซึ่งมีทั้งสิ้น 8 รอบ ในแต่ละรอบแต่ละกลุ่มของข้อมูลจะถูกทำ XOR, การบวก และการคูณ ด้วยกันเองพร้อมกับคีย์ซึ่งถูกแบ่งเป็นกลุ่มย่อยด้วยเช่นกันกลุ่มละ 16 บิต 6 กลุ่มแทนด้วย Z_1 ถึง Z_6 และในระหว่างแต่ละรอบของอัลกอริทึม ทำการสลับข้อมูลระหว่าง X_2 และ X_3 เพื่อเป็นอินพุทให้การทำงานในรอบต่อไป



รูปที่ 2.9 แสดงลักษณะการทำงานของ IDEA

ในแต่ละรอบมีลำดับของเหตุการณ์เกิดขึ้นดังนี้

(1) คูณ X_1 ด้วย Z_1

(2) บวก X_2 ด้วย Z_2

(3) บวก X_3 ด้วย Z_3

(4) คูณ X_4 ด้วย Z_4

(5) XOR ผลลัพธ์ของ 1. และ 3.

(6) XOR ผลลัพธ์ของ (2) และ (4)

(7) คูณผลลัพธ์ของ (5) ด้วย Z_5

(8) บวกผลลัพธ์ของ (6) และ (7)

(9) คูณผลลัพธ์ของ (8) ด้วย Z_6

(10) บวกผลลัพธ์ของ (7) และ (9)

(11) XOR ผลลัพธ์ของ (1) และ (9)

(12) XOR ผลลัพธ์ของ (3) และ (9)

(13) XOR ผลลัพธ์ของ (2) และ (10)

(14) XOR ผลลัพธ์ของ (4) และ (10)

ผลลัพธ์ที่ได้ของแต่ละรอบคือ ข้อมูลกลุ่มย่อยที่เป็นผลลัพธ์ที่ได้จากขั้นตอนที่ (11), (12), (13) และ (14) และทำการสลับข้อมูลระหว่าง X_2 และ X_3 (ยกเว้นรอบสุดท้าย) เพื่อเป็นอินพุทให้กับการทำงานรอบต่อไป

หลังจากครบ 8 รอบได้ผลลัพธ์ นำมาผ่านขั้นตอนสุดท้ายดังนี้

(1) คูณ X_1 ด้วย Z_1

(2) บวก X_2 ด้วย Z_2

(3) บวก X_3 ด้วย Z_3

(4) คูณ X_4 ด้วย Z_4

สุดท้ายผลลัพธ์ที่ได้มารวมกันเป็นไซเฟอร์เท็กซ์

3.3.4 การสร้างเซสชั่นคีย์ เริ่มต้น IDEA แบ่งคีย์ที่สร้างขึ้น 128 บิตออกเป็นกลุ่มย่อย 8 กลุ่มๆ ละ 16 บิต เป็น 8 กลุ่มแรกของการทำงานของอัลกอริทึม กล่าวคือ 6 กลุ่มสำหรับการทำงานรอบแรก และ 2 กลุ่มสำหรับการทำงานในรอบที่สอง เมื่อสิ้นสุดรอบแรก คีย์จะถูกเลื่อน (rotate) ไปทางซ้าย 25 บิตและแบ่งออกเป็น 8 กลุ่มย่อยอีกครั้ง โดยคีย์ 4 กลุ่มแรก นำมาใช้ในรอบที่สอง (รวมกับ 2 กลุ่มย่อยที่เหลือจากรอบแรก) ส่วนคีย์ 4 กลุ่มหลัง นำมาใช้ในการทำงานในรอบที่สามต่อไป เมื่อการทำงานในรอบสองเสร็จ คีย์ก็จะถูกเลื่อนไปทางซ้าย 25 บิตอีกครั้งและแบ่งเป็น 8 กลุ่มย่อย ลักษณะการทำงานจะเป็นเช่นนี้ไปจนครบ 8 รอบ เมื่อได้ผลลัพธ์ นำมาผ่านขั้นตอนการเปลี่ยนข้อมูลไปเป็นไซเฟอร์เท็กซ์ โดยใช้คีย์ 4 กลุ่มแรกในการทำงาน

สำหรับการถอดรหัส ใช้อัลกอริทึ่มเดียวกันยกเว้นคีย์ที่ใช้ในการถอดรหัสเป็นกลุ่มย่อยในลักษณะย้อนกลับกับคีย์ที่ใช้ในการเข้ารหัส ตารางที่ 2.4 แสดงการเข้ารหัสด้วยคีย์กลุ่มย่อย ที่เกี่ยวข้องกันกับคีย์กลุ่มย่อยที่ใช้ในการถอดรหัส

Round	Encryption Key Sub-blocks	Decryption Key Sub-blocks
1:	$Z_1^{(1)}\ Z_2^{(1)}\ Z_3^{(1)}\ Z_4^{(1)}\ Z_5^{(1)}\ Z_6^{(1)}$	$Z_1^{(9)}\ -1\ -Z_2^{(9)}\ -Z_3^{(9)}\ Z_4^{(9)}\ -1\ Z_5^{(8)}\ Z_6^{(8)}$
2:	$Z_1^{(2)}\ Z_2^{(2)}\ Z_3^{(2)}\ Z_4^{(2)}\ Z_5^{(2)}\ Z_6^{(2)}$	$Z_1^{(8)}\ -1\ -Z_2^{(8)}\ -Z_3^{(8)}\ Z_4^{(8)}\ -1\ Z_5^{(7)}\ Z_6^{(7)}$
3:	$Z_1^{(3)}\ Z_2^{(3)}\ Z_3^{(3)}\ Z_4^{(3)}\ Z_5^{(3)}\ Z_6^{(3)}$	$Z_1^{(7)}\ -1\ -Z_2^{(7)}\ -Z_3^{(7)}\ Z_4^{(7)}\ -1\ Z_5^{(6)}\ Z_6^{(6)}$
4:	$Z_1^{(4)}\ Z_2^{(4)}\ Z_3^{(4)}\ Z_4^{(4)}\ Z_5^{(4)}\ Z_6^{(4)}$	$Z_1^{(6)}\ -1\ -Z_2^{(6)}\ -Z_3^{(6)}\ Z_4^{(6)}\ -1\ Z_5^{(5)}\ Z_6^{(5)}$
5:	$Z_1^{(5)}\ Z_2^{(5)}\ Z_3^{(5)}\ Z_4^{(5)}\ Z_5^{(5)}\ Z_6^{(5)}$	$Z_1^{(5)}\ -1\ -Z_2^{(5)}\ -Z_3^{(5)}\ Z_4^{(5)}\ -1\ Z_5^{(4)}\ Z_6^{(4)}$
6:	$Z_1^{(6)}\ Z_2^{(6)}\ Z_3^{(6)}\ Z_4^{(6)}\ Z_5^{(6)}\ Z_6^{(6)}$	$Z_1^{(4)}\ -1\ -Z_2^{(4)}\ -Z_3^{(4)}\ Z_4^{(4)}\ -1\ Z_5^{(3)}\ Z_6^{(3)}$
7:	$Z_1^{(7)}\ Z_2^{(7)}\ Z_3^{(7)}\ Z_4^{(7)}\ Z_5^{(7)}\ Z_6^{(7)}$	$Z_1^{(3)}\ -1\ -Z_2^{(3)}\ -Z_3^{(3)}\ Z_4^{(3)}\ -1\ Z_5^{(2)}\ Z_6^{(2)}$
8:	$Z_1^{(8)}\ Z_2^{(8)}\ Z_3^{(8)}\ Z_4^{(8)}\ Z_5^{(8)}\ Z_6^{(8)}$	$Z_1^{(2)}\ -1\ -Z_2^{(2)}\ -Z_3^{(2)}\ Z_4^{(2)}\ -1\ Z_5^{(1)}\ Z_6^{(1)}$
output transformation	$Z_1^{(9)}\ Z_2^{(9)}\ Z_3^{(9)}\ Z_4^{(9)}$	$Z_1^{(1)}\ -1\ -Z_2^{(1)}\ -Z_3^{(1)}\ Z_4^{(1)}\ -1$

ตารางที่ 2.4 การเข้ารหัสด้วยคีย์กลุ่มย่อย และการถอดรหัสด้วยคีย์กลุ่มย่อยของ IDEA

3.4 RSA (Schneier,1994)

3.4.1 ความเป็นมา RSA เป็นการเข้ารหัสแบบการใช้คีย์สาธารณะ ที่คิดค้นโดย Ron Rivest, Adi Shamir และ Leonard Adleman ในปีค.ศ. 1978 ความปลอดภัยของ RSA เกิดจากความยากในการแยกตัวประกอบของเลขที่มีขนาดใหญ่ เพราะคีย์สาธารณะ และคีย์ส่วนตัวเป็นฟังก์ชั่นของเลขจำนวนเฉพาะ (prime number) ที่มีขนาดใหญ่หนึ่งคู่

3.4.2 การเข้ารหัสและถอดรหัส สัญญลักษณ์และสมการต่างๆ ที่ใช้ในการเข้ารหัสมีดังนี้

คีย์สาธารณะ คือ n และ e

n : เป็นผลคูณของเลขจำนวนเฉพาะสองตัวคือ p และ q

e : เป็นเลขจำนวนเฉพาะที่เป็นเลขปฐมสัมพัทธ์

กับ (p - 1) x (q - 1)

คีย์ส่วนตัว คือ n และ d

n : เป็นผลคูณของเลขจำนวนเฉพาะสองตัวคือ p และ q

d : เป็นผลลัพธ์ของสมการ $e^{-1} \pmod{(p - 1) \times (q - 1)}$

การเข้ารหัส ใช้สมการ

$$c = m^{e} \pmod{n}$$

การถอดรหัส ใช้สมการ

$$m = c^{d} \pmod{n}$$

โดยที่ c คือไซเฟอร์เท็กซ์ และ m คือเพลนเท็กซ์

3.4.3 ตัวอย่างการเข้ารหัสของ RSA เลือกเลขจำนวนเฉพาะสองค่า คือ p = 47 และ q = 71 แล้วจะได้

$n = p \times q = 47 \times 71 = 3337$

$(p - 1) \times (q - 1) = 46 \times 70 = 3220$

สมมติว่าเลือกค่า e เป็น 79

$d = 79^{-1} \pmod{3220} = 1019$

เข้ารหัสข้อมูล m = 232

$232^{79} \pmod{3337} = 2756 = c$

ถอดรหัสข้อมูล c = 2756

$2756^{1019} \pmod{3337} = 232 = m$

3.4.4 ความเร็วในการเข้ารหัสและความปลอดภัย การทำงานของ RSA ที่พัฒนาเป็นซอฟท์แวร์ ทำงานช้ากว่า DES ประมาณ 100 เท่า ในเครื่อง 386 ความเร็ว 33 Mhz ใช้เวลาในการเข้ารหัสข้อมูลประมาณ 8 kbps ส่วนความปลอดภัยของ RSA อยู่ที่การแยกตัวประกอบของตัวเลขจำนวนเฉพาะที่มีขนาดใหญ่ ซึ่งเป็นการแก้ปัญหาแบบ NP Complete

โดยสรุป การสร้างช่องทางการสื่อสารข้อมูลที่ปลอดภัย สามารถทำได้หลายรูปแบบหลายวิธี แนวทางหนึ่งในการเพิ่มระบบการเข้ารหัสข้อมูลที่ระดับชั้นของโปรแกรมประยุกต์ โดยการแก้ไขโปรแกรมต้นฉบับของโปรแกรมผู้ขอรับบริการ และโปรแกรมผู้ให้บริการของโปรแกรมประยุกต์เทลเนต แนวทางที่เหมาะสมทางหนึ่ง และนำโปรแกรมที่ได้รับแก้ไขมาใช้ประโยชน์ได้เป็นอย่างดี