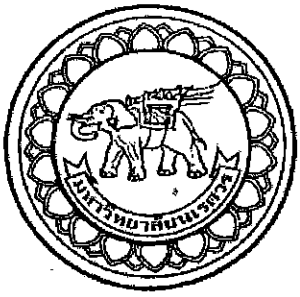


ไฟร์วอลล์และระบบความปลอดภัยบนเครือข่าย
กรณีศึกษาของมหาวิทยาลัยนเรศวร

Firewall and Network Security, Case Study of Naresuan University

นายศรายุทธ หอยสังข์ รหัส 44362762
นางสาวหัสยา โตมณีพิทักษ์ รหัส 44362812

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
ปีการศึกษา 2547

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน ไฟร์วอลล์และระบบความปลอดภัยบนเครือข่าย
กรณีศึกษาของมหาวิทยาลัยนเรศวร

ผู้ดำเนินโครงงาน นายศรายุทธ หอยสังข์ รหัส 44362762
นางสาวหัสยา โตมณีพิทักษ์ รหัส 44362812

อาจารย์ที่ปรึกษา อาจารย์พงศ์พันธ์ กิจสนาโยธิน

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2547

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิจัย

.....ประธานกรรมการ
(อาจารย์พงศ์พันธ์ กิจสนาโยธิน)

.....กรรมการ
(อาจารย์สุรเชษฐ์ กานต์ประชา)

.....กรรมการ
(อาจารย์พนมขวัญ ริยะมงคล)

หัวข้อโครงงาน	ไฟร์วอลล์และระบบความปลอดภัยบนเครือข่าย กรณีศึกษาของมหาวิทยาลัยนเรศวร
ผู้ดำเนินโครงงาน	นายศรายุทธ หอยสังข์ รหัส 44362762 นางสาวหัสยา โตมณีพิทักษ์ รหัส 44362812
อาจารย์ที่ปรึกษา	อาจารย์พงศ์พันธ์ กิจสนาโยธิน
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2547

บทคัดย่อ

โครงงานนี้เป็นการศึกษาและพัฒนาไฟร์วอลล์เพื่อป้องกันและรักษาความปลอดภัยให้กับระบบเครือข่าย เนื่องจากองค์กรที่มีการเชื่อมต่อกับอินเตอร์เน็ตสามารถถูกเข้าถึงโดยเครื่องภายในเครือข่ายใดก็ได้ที่เชื่อมต่ออินเตอร์เน็ตอยู่ ดังนั้นการรักษาความปลอดภัยจึงเป็นสิ่งจำเป็น เพื่อรักษาข้อมูลที่สำคัญและเป็นการป้องกันไม่ให้ผู้ไม่หวังดีนำข้อมูลเหล่านั้นไปใช้ในทางที่ผิด

ไฟร์วอลล์ที่สร้างขึ้นเป็นไฟร์วอลล์ประเภทสเตทฟูลมัลติเลเยอร์อินสเปกชั่นไฟร์วอลล์ (Stateful Multilayer Inspection Firewall) ทำงานบนระบบปฏิบัติการ Linux RedHat 9.0 โดยใช้ความรู้เรื่อง IPtables และ Squid ในการพัฒนาไฟร์วอลล์เพื่อสร้างกฎเกณฑ์ต่างๆ สามารถให้บริการและระงับบริการเซอร์วิสต่างๆ ที่ต้องการและป้องกันการโจมตีประเภท Denial of Services ได้

Project Title	Firewall and Network Security, Case Study of Naresuan University
Name	Mr. Sarayut Hoysang ID. 44362762
	Miss Hatsaya Tomaniphithak ID. 44362812
Project Advisor	Mr. Phongphun Kijsanayothin
Major	Computer Engineering
Department	Electrical and Computer Engineering
Academic Year	2004

Abstract

This project is study and development of firewall to prevent and secure the network security. Due to an organization which connects to internet can be accessed by anyone who connects to internet; therefore, security is necessary to secure the important information, to prevent hacker from stealing the information and to prevent attacking network system.

The firewall studied in this project is Stateful Multilayer Inspection Firewall operating under Linux RedHat 9.0 Operating System. This firewall is developed by using IPtables and Squid to establish rules. It can serve the required services and reject any services in order to prevent Denial of Services attacks.

กิตติกรรมประกาศ

โครงงานนี้สำเร็จไปได้ด้วยความช่วยเหลืออย่างดียิ่งจาก อาจารย์พงศ์พันธ์ กิจสนาโยธิน อาจารย์ที่ปรึกษาโครงงาน ท่านได้สละเวลา ความคิด ประสบการณ์ และคำปรึกษาในการทำโครงงาน จนกระทั่งแล้วเสร็จ ทำให้คณะผู้จัดทำได้รับประสบการณ์ทำงานอันมีค่ายิ่ง

ขอขอบคุณพี่ทุกคนที่ช่วยถ่ายทอดความรู้และประสบการณ์ ขอบคุณเพื่อนหลายคนที่ให้การสนับสนุนอุปกรณ์และหนังสือสำหรับสืบค้นข้อมูลในการทำโครงงาน

นายศรายุทธ หอยสังข์

นางสาวหัสยา โตมณีพิทักษ์

สารบัญ

สารบัญตาราง

สารบัญรูป

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

อินเตอร์เน็ต เป็นเครือข่ายคอมพิวเตอร์ขนาดใหญ่ที่เชื่อมโยงเครือข่ายคอมพิวเตอร์ทั่วโลกเข้าด้วยกัน ทำให้คนที่อยู่คนละซีกโลกสามารถสนทนากันได้ และในขณะเดียวกันเครือข่ายขององค์กรเหล่านี้ที่เชื่อมต่อกับอินเตอร์เน็ตก็สามารถถูกเข้าถึงโดยใครก็ได้ที่ใช้งานอินเตอร์เน็ตอยู่ ซึ่งส่วนใหญ่องค์กรที่เชื่อมต่ออินเตอร์เน็ตอยู่นั้นมักมีข้อมูลที่สำคัญและเป็นความลับ ทำให้ข้อมูลเหล่านี้อยู่ในความเสี่ยงต่อผู้ไม่หวังดีต่อองค์กร ซึ่งอาจจะเป็นคู่แข่งหรือใครก็ตามอย่างหลีกเลี่ยงไม่ได้ ด้วยเหตุนี้ความปลอดภัยในเครือข่ายขององค์กรที่เชื่อมต่อกับอินเตอร์เน็ตจึงเป็นสิ่งที่สำคัญสำหรับองค์กรเหล่านั้นเป็นอย่างมาก

ดังนั้นการรักษาความปลอดภัยบนระบบเครือข่ายที่เชื่อมต่อกับอินเตอร์เน็ตจึงเป็นสิ่งจำเป็นเพื่อรักษาข้อมูลที่สำคัญ และเป็นความลับที่ไม่สามารถเปิดเผยได้ เป็นการป้องกันไม่ให้ผู้ไม่หวังดีนำข้อมูลเหล่านั้นไป หรือมาโจมตีระบบเครือข่ายของเราได้ ทางผู้จัดทำจึงได้ทำการศึกษารายละเอียดเกี่ยวกับการรักษาความปลอดภัยบนระบบเครือข่าย เพื่อเป็นประโยชน์แก่ผู้ที่สนใจหรือต้องการศึกษาเพื่อเป็นแนวทางในการนำไปใช้ประโยชน์ หรือนำไปเป็นพื้นฐานในการศึกษาค้นคว้าต่อไป

1.2 วัตถุประสงค์ของโครงงาน

1. ศึกษาโครงสร้างและลักษณะการทำงานของไฟร์วอลล์ (Firewall)
2. ศึกษาโปรโตคอลประเภทต่างๆที่ใช้ในการรับส่งข้อมูลบนเครือข่าย
3. ศึกษารูปแบบการบุกรุกและการโจมตีเครือข่ายในแบบต่างๆ
4. ศึกษาการใช้งาน iptables และ squid เพื่อนำมาสร้างกฎเกณฑ์ต่างๆ ในการพัฒนาไฟร์วอลล์
5. สร้างไฟร์วอลล์เพื่อป้องกันและรักษาความปลอดภัยให้แก่ระบบเครือข่าย

1.3 ขอบเขตของโครงงาน

1. ไฟร์วอลล์ที่สร้างเป็นไฟร์วอลล์ประเภท Stateful Multilayer Inspection Firewall
2. ไฟร์วอลล์ที่สร้างสามารถให้บริการและระงับบริการเซอร์วิสต่างๆ ที่ต้องการได้
3. ไฟร์วอลล์ที่สร้างสามารถป้องกันการโจมตีเครือข่ายประเภท Denial of Services ได้

1.4 ขั้นตอนการดำเนินงาน

กิจกรรม	เดือน							
	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.
1. ศึกษาและค้นคว้าข้อมูล	←	—	→					
2. ออกแบบไฟร์วอลล์			←	→				
3. พัฒนาไฟร์วอลล์				←	—	→		
4. ทดสอบไฟร์วอลล์					←	—	→	
5. สรุปผล								↔

1.5 ผลที่คาดว่าจะได้รับ

1. มีความรู้ความเข้าใจโครงสร้างและลักษณะการทำงานของไฟร์วอลล์
2. มีความรู้ความเข้าใจโปรโตคอลประเภทต่างๆที่ใช้ในการรับส่งข้อมูลบนเครือข่าย
3. มีความรู้ความเข้าใจรูปแบบการบุกรุกและการโจมตีเครือข่ายในแบบต่างๆ
4. มีความรู้ความเข้าใจการใช้งาน iptables และ squid และนำมาสร้างกฎเกณฑ์ต่างๆ ในการพัฒนาไฟร์วอลล์ได้
5. ไฟร์วอลล์ที่สร้างสามารถป้องกันและรักษาความปลอดภัยให้แก่ระบบเครือข่ายได้

1.6 งบประมาณในการดำเนินงาน นิสิต : คน : 1,000 บาท

1. วัสดุสำนักงาน	190	บาท
2. วัสดุคอมพิวเตอร์	860	บาท
3. ค่าถ่ายเอกสาร	650	บาท
4. ค่าเย็บเล่ม	300	บาท
รวมเป็นเงิน	2,000	บาท

บทที่ 2
หลักการและทฤษฎี

เมื่อพูดถึงความปลอดภัยในเครือข่าย ไฟร์วอลล์ (Firewall) เป็นเครื่องมืออันดับแรกที่ถูกกล่าวถึงเสมอ และปัจจุบันแทบจะกลายเป็นเครื่องมือที่จำเป็นที่ขาดไม่ได้สำหรับเครือข่ายที่ต่อกับอินเตอร์เน็ต ความหมายของชื่อรวมกับความคลุมเครือของหน้าที่การทำงานของไฟร์วอลล์ ทำให้มีผู้เข้าใจคลาดเคลื่อนว่าไฟร์วอลล์เป็นเครื่องมือสำหรับปราบแฮคเกอร์ (Hacker) และการบุกรุกที่มาจากอินเตอร์เน็ตได้อย่างสิ้นเชิง รวมไปถึงการติดตั้งไฟร์วอลล์เสมือนเป็นการป้องกันไม่ให้แฮคเกอร์สามารถเข้ามาได้และทำหน้าที่รักษาเครือข่ายภายในให้เรียบร้อย

ด้วยความเข้าใจในลักษณะนี้และคาดหวังความสามารถของไฟร์วอลล์มากกว่าความเป็นจริง ทำให้ไฟร์วอลล์ถูกนำไปใช้งานโดยไม่เหมาะสม และการดูแลรักษาการปรับปรุงกฎต่างๆ ของไฟร์วอลล์ถูกละเลย ส่งผลให้ความสามารถในการป้องกันของไฟร์วอลล์อ่อนแอลง จนในที่สุดไฟร์วอลล์แทบไม่ได้ช่วยให้เครือข่ายมีความปลอดภัยอีกต่อไป

ถ้าให้เข้าใจได้ง่ายที่สุดไฟร์วอลล์ คือเครื่องมือที่ใช้ป้องกันเครือข่ายจากการสื่อสารทั่วไปที่ไม่ได้รับอนุญาต ปัญหาพื้นฐานในเรื่องความปลอดภัยบนเครือข่าย คือการเข้าถึงระบบหรือข้อมูลภายในผ่านทางเครือข่าย หรือที่เรียกว่า ลอจิคัลแอคเซส (Logical Access) ซึ่งเกิดขึ้นได้ง่ายกว่าฟิสิคัลแอคเซส (Physical Access) คือเข้ามาที่ตัวเครื่องจริงๆ การที่เรานำโฮสต์ (Host)ใดๆ มาต่อเข้ากับเครือข่าย หมายถึงโฮสต์ของเราสามารถถูกแอคเซสได้จากทุกที่ที่เครือข่ายครอบคลุมไปถึง อย่างไรก็ตาม ลอจิคัลแอคเซสจะเกิดขึ้นได้ก็ต่อเมื่อโฮสต์จะต้องสามารถสร้างการเชื่อมต่อ หรือลอจิคัลคอนเนคชั่น (Logical Connection) กับโฮสต์ปลายทางได้ ซึ่งความสามารถในการสร้างลอจิคัลคอนเนคชั่นจะขึ้นอยู่กับโปรโตคอลที่ใช้งานเป็นสำคัญ บางโปรโตคอลสามารถสร้างลอจิคัลแอคเซสระหว่างโฮสต์ที่อยู่บนเซกเมนต์ (Segment) เดียวกัน บางโปรโตคอลสามารถสร้างลอจิคัลคอนเนคชั่นให้ข้ามเซกเมนต์ได้ แต่โปรโตคอลที่สำคัญที่สุดที่ต้องดูแลอย่างระมัดระวัง คือโปรโตคอลทีซีพี /ไอพี (TCP/IP) ซึ่งใช้งานอยู่บนอินเตอร์เน็ต เพราะสามารถสร้างลอจิคัลคอนเนคชั่นได้โดยไม่มีขีดจำกัดในเรื่องระยะทาง กล่าวคือถ้าเราทำการเชื่อมโฮสต์กับเครือข่ายของบริษัท จำนวนโฮสต์อื่นที่สามารถแอคเซสโฮสต์ของเราได้จะมีเฉพาะที่เชื่อมต่ออยู่ในบริษัทเท่านั้น แต่เมื่อเราเชื่อมโฮสต์เข้ากับอินเตอร์เน็ตจะทำให้โฮสต์อีกมากมายบนอินเตอร์เน็ตสามารถแอคเซสโฮสต์ของเราได้ทันที

การแอคเซสเป็นปัจจัยพื้นฐานในการใช้งานอินเตอร์เน็ต เพราะถ้าโฮสต์ไม่สามารถแอคเซสได้ก็ไม่สามารถติดต่อสื่อสารกันในระดับที่สูงขึ้นไปได้ สำหรับผู้ใช้สามารถรับรู้การแอคเซสได้ผ่านแอพพลิเคชั่น (Application) ที่ใช้งาน เช่น สามารถแชร์ไฟล์ระหว่างเครื่อง

คอมพิวเตอร์ของตนกับเพื่อนร่วมงานใช้งานโปรแกรมที่ทำงานอยู่บนแลน (LAN : Local Area Network) ได้ เพราะในปัจจุบันการทำงานร่วมกันด้วยคอมพิวเตอร์ส่วนบุคคลผ่านเครือข่ายเป็นสิ่งปกติทั่วไป และเป็นสิ่งที่ทุกองค์กรจำเป็นต้องมี ประโยชน์จากการที่โฮสต์สามารถแอคเซสระหว่างกันได้เป็นที่ทราบโดยทั่วไปจากการแลกเปลี่ยนข้อมูลกันและจากแอพพลิเคชั่นที่ทำงานผ่านเครือข่าย

อย่างไรก็ตามการแอคเซสกันได้ระหว่างโฮสต์ไม่ใช่มีแต่ประโยชน์เพียงด้านเดียว การแชร์ข้อมูลกันช่วยให้ทำงานร่วมกันได้อย่างสะดวกรวดเร็วถูกต้อง แต่ในทางกลับกันก็ทำให้ข้อมูลสามารถรั่วไหลไปตามเครือข่ายได้อย่างรวดเร็วเช่นกัน ซึ่งเป็นปัญหาพื้นฐานของแอพพลิเคชั่นที่ทำงานผ่านเครือข่ายและรับรู้ได้ไม่ยาก เพราะเป็นปัญหาผ่านแอพพลิเคชั่นที่ผู้ใช้คุ้นเคย แต่การแอคเซสระหว่างโฮสต์จริงๆแล้วค่อนข้างสลับซับซ้อนและผ่านกระบวนการต่างๆมากกว่าที่จะปรากฏเป็นข้อมูลในแอพพลิเคชั่นที่ผู้ใช้เข้าใจ ข้อมูลส่วนที่ปรากฏให้ผู้ใช้รับรู้ได้ผ่านแอพพลิเคชั่นคิดเป็นสัดส่วนน้อยมากเมื่อเทียบกับปริมาณข้อมูลที่สื่อสารกันทั้งหมด ซึ่งแฮคเกอร์หรือผู้ไม่ประสงค์ดีทั้งหลายได้อาศัยเครือข่ายเป็นช่องทางในการกระทำผิดโดยที่ผู้ใช้ไม่มีโอกาสได้รับรู้ ยิ่งเครือข่ายเปิดกว้างต่อการแอคเซสมากเท่าไรก็ย่อมมีความเสี่ยงมากขึ้นเท่านั้น

การมีเครือข่ายที่ครอบคลุมเฉพาะในองค์กรของตนเองนั้น ถึงแม้มีความเสี่ยงจากความไม่ปลอดภัยจากเครือข่ายอยู่บ้าง แต่จะจำกัดขอบเขตเฉพาะจากคอมพิวเตอร์ที่ต่ออยู่ภายใน การแอคเซสของเครือข่ายถูกจำกัดด้วยลักษณะทางกายภาพอยู่แล้ว และถ้ามีสิ่งผิดปกติส่วนใหญ่ก็จะมาจากคนภายในองค์กรเท่านั้น แต่เมื่อบริษัทหรือหน่วยงานต้องการติดต่อสื่อสารกับโลกภายนอกความจำเป็นที่ต้องขยายเครือข่ายให้ครอบคลุมไปยังภายนอกย่อมมีมากขึ้น ความเสี่ยงในการที่เครือข่ายอาจจะถูกแอคเซสได้จากบุคคลภายนอกก็มีมากขึ้น และเมื่อใดก็ตามที่ต้องการติดต่อกับบุคคลภายนอกผ่านทางคอมพิวเตอร์ หมายถึงการที่จำเป็นต้องแอคเซสไปยังอินเตอร์เน็ต และในทางกลับกันก็เป็นการยอมให้โฮสต์อื่นในอินเตอร์เน็ตสามารถแอคเซสเครือข่ายของเราได้ และทำให้ผู้ที่ต่ออยู่กับอินเตอร์เน็ตไม่ว่าจะอยู่ที่ใดก็สามารถแอคเซสเครือข่ายของเราได้เสมือนว่าเข้ามานั่งอยู่ในศูนย์คอมพิวเตอร์ของเราเอง

อินเตอร์เน็ตเป็นเครือข่ายสื่อสารข้อมูลคอมพิวเตอร์สากลที่เชื่อมต่อเครือข่ายต่างๆทั่วโลกเข้าด้วยกันโดยอาศัยโปรโตคอล TCP/IP เป็นกลไกสำคัญในการสื่อสารข้อมูลข้ามเครือข่าย ด้วยเหตุที่โปรโตคอล TCP/IP ขาดกลไกการรักษาความปลอดภัยที่เพียงพอ ปัญหาที่ตามมาจึงมีมากมายและส่งผลกระทบต่อความปลอดภัยของผู้ใช้มากขึ้น และทำให้เกิดความจำเป็นที่ต้องหาอุปกรณ์รักษาความปลอดภัยอื่นๆ เข้ามาช่วย เช่น ไฟร์วอลล์ เป็นต้น

2.1 คุณสมบัติทั่วไปของไฟร์วอลล์

ไฟร์วอลล์เป็นเครื่องมือรักษาความปลอดภัยที่ทำงานในเชิงป้องกัน (Protect) ซึ่งทำหน้าที่ควบคุมการเข้าถึงเครือข่าย (Access Control) โดยอาศัยกฎเป็นพื้นฐาน (Rule Base) สำหรับคุณสมบัติแต่ละอย่างของไฟร์วอลล์มีรายละเอียดดังนี้

2.1.1 Protect ไฟร์วอลล์เป็นเครื่องมือที่ใช้ทำงานในเชิงป้องกัน โดยแพ็คเก็ต (Packet) ที่สามารถผ่านเข้าออกเครือข่ายได้นั้นจะต้องเป็นแพ็คเก็ตที่ไฟร์วอลล์เห็นว่ามีความปลอดภัย แพ็คเก็ตใดที่ไฟร์วอลล์เห็นว่าไม่ปลอดภัย หรืออาจนำมาซึ่งความไม่ปลอดภัยก็จะถูกดร็อป (drop) โดยการที่ไฟร์วอลจะตัดสินใจว่าแพ็คเก็ตใดปลอดภัยและแพ็คเก็ตใดไม่ปลอดภัยนั้นจะอยู่บนพื้นฐานของกฎที่ผู้ดูแลไฟร์วอลล์ (Firewall Administrator) เป็นผู้กำหนดล่วงหน้า ซึ่งเงื่อนไขของกฎเหล่านี้ทำให้ไฟร์วอลล์สามารถป้องกันแพ็คเก็ตที่อาจจะส่งผลร้ายไม่ให้ผ่านเข้าไปถึงเครือข่ายได้

2.1.2 Access Control "แอคเซส" หมายถึง การที่โฮสต์ใดโฮสต์หนึ่งสามารถสื่อสารข้อมูลที่ต้องการไปยังโฮสต์ปลายทางได้สำเร็จ การแอคเซสในแต่ละระดับจะมีวิธีการแตกต่างกันออกไป ทำให้การควบคุมแอคเซสสำหรับแต่ละระดับแตกต่างกันตามไปด้วย ไฟร์วอลล์จึงมีการทำงานหลายลักษณะตามวิธีที่ไฟร์วอลล์ใช้ควบคุมการแอคเซส

2.1.3 Rule Base ไฟร์วอลล์จะควบคุมการแอคเซสโดยอาศัยการเปรียบเทียบคุณสมบัติของแพ็คเก็ตที่ผ่านไฟร์วอลล์กับกฎของการแอคเซสที่ได้กำหนดไว้ ถ้าพบว่าไม่มีกฎที่ห้ามไว้ก็จะอนุญาตให้แพ็คเก็ตนั้นผ่านไปได้ ถ้ามีกฎที่ห้ามไว้แพ็คเก็ตนั้นจะก็จะถูกสกัดกั้นไว้ด้วยวิธีใดวิธีหนึ่ง

ดังนั้นการที่แพ็คเก็ตใดๆ สามารถผ่านเข้าออกไฟร์วอลล์ได้หรือไม่นั้นจึงอยู่ที่กฎเป็นสำคัญ สำหรับไฟร์วอลล์โดยตัวเองแล้วนั้นจะไม่ทราบว่าแพ็คเก็ตใดเป็นแพ็คเก็ตที่ปลอดภัยหรือแพ็คเก็ตใดเป็นแพ็คเก็ตที่ไม่ปลอดภัย (ยกเว้นแพ็คเก็ตที่เป็นอันตรายโดยตัวมันเองอยู่แล้ว เช่น แพ็คเก็ตแปลกประหลาดหรือ Anomalous Packet ที่ใช้สำหรับการโจมตีโดยเฉพาะ) ไฟร์วอลล์จะรู้จักเฉพาะแพ็คเก็ตที่ได้รับอนุญาตและแพ็คเก็ตที่ไม่ได้รับอนุญาต ตามกฎที่ระบุไว้เท่านั้น หมายความว่าแพ็คเก็ตที่ใช้เพื่อจุดประสงค์ร้ายถ้ามีลักษณะไม่เข้าข่ายหรือผิดกฎที่ตั้งไว้ก็อาจได้รับอนุญาตให้ผ่านเข้ามาได้โดยที่ไฟร์วอลล์ไม่สามารถทราบได้ ดังนั้น "ไม่จำเป็นเสมอไปว่าการบุกรุกทั้งหลายสามารถป้องกันได้ด้วยไฟร์วอลล์"

โดยพื้นฐานของเครือข่ายทั่วไปเมื่อต่อเข้ากับอินเตอร์เน็ตจะมีความเสี่ยงเนื่องจากการไม่สามารถป้องกันตนเองได้คือ

- การไม่สามารถจำกัดที่มาของแพ็คเก็ต หมายถึงเมื่อโฮสต์ของเราได้ต่อเข้ากับอินเตอร์เน็ตแล้ว โฮสต์อื่นๆที่อยู่บนอินเตอร์เน็ตไม่ว่าจะอยู่ที่ใดสามารถส่งแพ็คเก็ตมายังโฮสต์ของ

เราได้ทันทีตลอดเวลา โดยที่โฮสต์ของเราจะตอบรับหรือไม่ก็ตาม แต่แพ็คเก็ตเหล่านั้นจะถูกส่งมาจนถึงโฮสต์ของเราในที่สุด

ตัวอย่างเช่น เมื่อทำการต่อกับอินเตอร์เน็ตโดยต้องการเพียงที่จะติดต่อกับโฮสต์ที่อยู่ในประเทศอเมริกา และโฮสต์ที่อยู่ในประเทศไทยเท่านั้น แต่ไม่อาจห้ามไม่ให้โฮสต์ที่อยู่ในประเทศอื่นที่ต่ออยู่บนอินเตอร์เน็ตเช่นกัน ส่งแพ็คเก็ตเข้ามาหายังภายในเครือข่ายของเราได้ ถ้าต้องการป้องกันแพ็คเก็ตในลักษณะนี้เราต้องใช้เครื่องมือในลักษณะของไฟร์วอลล์มาติดตั้งเพื่อป้องกันเพิ่มเติม ซึ่งด้วยคุณสมบัติของไฟร์วอลล์อย่างน้อยที่สุดจะช่วยทำให้สามารถกำหนดได้ว่า ถึงแม้จะต่ออยู่กับอินเตอร์เน็ต แต่เครือข่ายมิได้เปิดกว้างสำหรับทุกๆคนจะเปิดให้เฉพาะต้นทางที่เห็นว่าเหมาะสมเท่านั้น

- การไม่สามารถจำกัดปลายทางของแพ็คเก็ตได้ ภายในเครือข่ายของเราอาจมีโฮสต์อยู่หลายโฮสต์ต่ออยู่ เมื่อเครือข่ายของเราได้เชื่อมเข้ากับอินเตอร์เน็ตก็ทำให้โฮสต์เหล่านั้นต่อกับอินเตอร์เน็ตไปด้วย ทั้งที่ความต้องการอาจมีเพียงโฮสต์ที่ทำหน้าที่เป็นแม่ข่าย (Server) เพียงเครื่องเดียวเท่านั้นที่ต้องการแอคเซสหรือสื่อสารกับอินเตอร์เน็ต แต่เมื่อมีโฮสต์ที่ต่อร่วมเครือข่ายเดียวกันทำให้ต่อกับอินเตอร์เน็ตไปด้วย และทำให้ทุกโอสต์ในเครือข่ายสามารถแอคเซสได้จากอินเตอร์เน็ตไปด้วยเช่นกัน

ด้วยความสามารถพื้นฐานทั่วไป โปรโตคอล TCP/IP และเครือข่าย ถ้าปราศจากไฟร์วอลล์แล้ว ถึงแม้จะรู้ว่าโฮสต์อื่นที่อยู่ในเครือข่ายนั้นมิได้มีความประสงค์ที่จะต่อกับอินเตอร์เน็ต แต่ก็ไม่สามารถป้องกันไม่ให้โฮสต์อื่นบนอินเตอร์เน็ตส่งแพ็คเก็ตเข้ามายังโฮสต์เหล่านั้นได้ ถ้าโฮสต์เหล่านั้นมีความบกพร่องของการรักษาความปลอดภัย อาจถูกเจาะเข้าไปและก่อให้เกิดความเสียหายได้ เหมือนเป็นการผลักภาระให้แก่โฮสต์เหล่านั้นในการป้องกันตนเองจากการรบกวนจากอินเตอร์เน็ต ทั้งๆ ที่โฮสต์เหล่านั้นไม่มีความประสงค์ที่จะติดต่อกับอินเตอร์เน็ตด้วยซ้ำ

ในอีกมุมหนึ่งการที่มีโฮสต์ภายในเครือข่ายเครื่องใดเครื่องหนึ่งต่อกับอินเตอร์เน็ตก็เป็นช่องทางที่โฮสต์ภายในเครื่องอื่นจะติดกับ โลกภายนอกได้โดยตรง ซึ่งเมื่อมีช่องทางอยู่และไม่มีเครื่องมือที่จะควบคุม จึงไม่สามารถรับรองได้ว่าบุคคลภายนอกจะไม่ใช้ช่องทางนี้เพื่อส่งข้อมูลออกไป หรือใช้เป็นเครื่องมือในการกระทำเรื่องส่วนตัวที่อาจไม่เป็นผลดีกับองค์กร หรือแม้กระทั่งใช้ไปเพื่อบุกรุกเครือข่ายของผู้อื่น

- **การไม่สามารถจำกัดลักษณะของแพ็คเก็ตที่ต้องการได้** แอพพลิเคชั่นที่ให้บริการอยู่บนโฮสต์และสามารถใช้บริการผ่านเครือข่ายได้มีอยู่มากมาย เช่น อีเมล์ (e-mail), เว็บไซท์ (website), ไอซีคิว (ICQ), ไฟล์ทรานส์เฟอร์ (file transfer) การใช้งานแอพพลิเคชั่นแต่ละชนิดจะใช้แพ็คเก็ตที่มีลักษณะที่แตกต่างกัน ซึ่งเมื่อเชื่อมต่ออยู่กับอินเตอร์เน็ตแล้วจะไม่สามารถจำกัดได้ว่าแพ็คเก็ตประเภทใดที่ให้ผ่านเข้ามาได้ และประเภทใดผ่านเข้ามาไม่ได้ ทำได้มากที่สุดคือเมื่อไม่มีแอพพลิเคชั่นนั้นให้บริการอยู่ก็ตอบปฏิเสธกลับไป แต่แพ็คเก็ตเหล่านั้นก็เข้ามาถึงโฮสต์อยู่ดี

โดยทั่วไปเมื่อมีการตอบปฏิเสธไปแล้วผู้ที่ส่งแพ็คเก็ตไม่น่าจะได้รับประโยชน์อะไรจากเครือข่ายของเรา แต่ในความเป็นจริงแล้วการที่โฮสต์ได้รับแพ็คเก็ตที่มาคอยรบกวนอยู่ตลอด แล้วต้องคอยตอบปฏิเสธไปตลอดเวลานั้น อาจนำมาซึ่งความสูญเสียที่คาดไม่ถึงก็ได้ การป้องกันแพ็คเก็ตประเภทนี้ไม่ให้เข้าถึงโฮสต์ภายในจึงเป็นการป้องกันที่ช่วยให้โฮสต์มีความปลอดภัยมากขึ้น

จากความเสี่ยงทั้ง 3 ประการล้วนเป็นความเสี่ยงที่เกิดขึ้นจากการแอคเซสระหว่างเครือข่ายภายในกับอินเตอร์เน็ตปราศจากการควบคุม ซึ่งความเสี่ยงเหล่านี้สามารถหลีกเลี่ยง ทำให้ลดลง และสามารถควบคุมได้ด้วยการนำไฟร์วอลล์ไปควบคุมการแอคเซสบนพื้นฐานของกฎที่เหมาะสม

2.2 ขีดจำกัดของไฟร์วอลล์

องค์ประกอบของการบริการต่างๆ ที่มีอยู่บนอินเตอร์เน็ตนั้นประกอบขึ้นจากเทคโนโลยีทางด้านคอมพิวเตอร์ โทรคมนาคมและการสื่อสารข้อมูลที่หลากหลายทำงานร่วมกัน องค์ประกอบแต่ละส่วนต่างมีหน้าที่ความรับผิดชอบแตกต่างกัน และต่างมีข้อบกพร่องที่เป็นความเสี่ยงแตกต่างกันออกไป ข้อบกพร่องขององค์ประกอบแต่ละส่วนมีลักษณะของการป้องกันที่แตกต่างกันไปด้วย

ไฟร์วอลล์จัดเป็นเครื่องมือป้องกันรักษาความปลอดภัยทางเครือข่ายที่สำคัญ แต่ความสามารถในการรักษาความปลอดภัยของไฟร์วอลล์มีขีดจำกัด ผู้ใช้ส่วนมากเข้าใจว่าไฟร์วอลล์จะช่วยป้องกันเครือข่ายจากความไม่ปลอดภัยทั้งหลายที่เกิดขึ้น เพราะผู้ใช้ส่วนใหญ่ไม่เข้าใจว่าภัยต่างๆในเครือข่ายมีอยู่มากมายหลายชนิดและส่งผลกระทบต่อผู้ใช้แตกต่างกันออกไป และเข้าใจว่าความไม่ปลอดภัยมีอยู่เพียงอย่างเดียวเหมือนกันหมด ซึ่งไฟร์วอลล์สามารถป้องกันได้

แท้จริงแล้วยิ่งผู้ใช้งานคอมพิวเตอร์มากขึ้นเท่าไร ความหลากหลายของความไม่ปลอดภัยก็มากขึ้นเท่านั้น เช่น ถ้าใช้คอมพิวเตอร์เพื่อรับ-ส่งอีเมล์ ความไม่ปลอดภัยจะกระทบกระเทือนเฉพาะการที่มีคนอื่นแอบอ่านจดหมายหรือไม่ก็ส่งจดหมายในนามของตนเองเท่านั้น แต่ถ้านำคอมพิวเตอร์ไปใช้โอนเงินกับธนาคาร บริหารกิจการของบริษัท ควบคุมเครื่องจักรในโรงงาน ลักษณะของภัยที่เกิดขึ้นกับระบบคอมพิวเตอร์จะซับซ้อนหลากหลายและมีปริมาณมากขึ้น ผลกระทบและความเสียหายจะแตกต่างกันไปด้วย

การรักษาความปลอดภัยในเครือข่ายเหมือนการรักษาความปลอดภัยทั่วไป เมื่อภัยคุกคามมีหลายรูปแบบ เครื่องมือที่ใช้ต้องมีระดับและขอบเขตความสามารถในการป้องกันภัยที่หลากหลายสอดคล้องกัน และไฟร์วอลล์เป็นหนึ่งในเครื่องมือสำคัญของการรักษาความปลอดภัยในเครือข่าย แต่ไฟร์วอลล์ไม่ใช่ทางออกของทุกปัญหาในเครือข่าย

2.2.1 สิ่งที่ไฟร์วอลล์สามารถป้องกันได้

ด้วยคุณสมบัติที่มีอยู่ของไฟร์วอลล์ทำให้สามารถระบุได้อย่างชัดเจนว่าภัยประเภทใดบ้างที่ไฟร์วอลล์สามารถป้องกันได้โดยตรง แต่ทั้งนี้ต้องขึ้นอยู่กับว่าไฟร์วอลล์ได้มีการกำหนดกฎต่างๆ เอาไว้อย่างถูกต้องจึงจะได้ผล สิ่งที่ไฟร์วอลล์สามารถป้องกันได้มีดังนี้

- Network Scanning

การสแกนเครือข่ายเป็นสัญญาณเริ่มต้นของภัยอื่นๆที่จะติดตามมา ด้วยคุณสมบัติที่สามารถควบคุมการเข้าออกของแพ็คเก็ตได้ ทำให้ผู้ใช้มีโอกาสที่จะจำกัดปลายทางของแพ็คเก็ตที่ให้ผ่านเข้ามาเฉพาะโฮสต์ที่อนุญาตให้ติดต่อได้จากภายนอกเท่านั้น แพ็คเก็ตที่ส่งเข้ามาเพื่อสำรวจเครือข่ายโดยการส่งไปยังโฮสต์อื่นๆในเครือข่ายจะไม่สามารถเล็ดลอดไปถึงเป้าหมายและนำข้อมูลออกไปได้ การที่แฮคเกอร์จะเจาะเข้าไปยังเครือข่ายใดนั้นมักจะเริ่มโดยเจาะเข้าไปยังโฮสต์ที่มีการป้องกันตัวเองน้อยที่สุดก่อน เมื่อแฮคเกอร์สามารถสแกนเครือข่ายได้จะทำให้สามารถค้นพบได้ว่าภายในเครือข่ายนั้นมีโฮสต์อะไรบ้าง และโฮสต์แต่ละเครื่องมีระดับรักษาความปลอดภัยมากน้อยเท่าไร

- Host Scanning

นอกจากการสแกนเครือข่ายแล้ว การสแกนโฮสต์ก็เป็นองค์ประกอบเริ่มต้นที่สำคัญของการเตรียมการของการเจาะระบบ เพราะถึงแม้จะมีไฟร์วอลล์ติดตั้งอยู่ก็ไม่ได้หมายความว่าเครือข่ายที่อยู่หลังไฟร์วอลล์จะถูกตัดขาดจากโลกภายนอก จะต้องมีโฮสต์อย่างน้อยหนึ่งตัวที่จะต้องสามารถติดต่อกับโลกภายนอกได้ และโฮสต์นั้นก็มีโอกาสที่จะถูกสแกนได้ การที่แฮคเกอร์สามารถสแกนโฮสต์ได้นั้นจะทำให้สามารถมีโอกาสค้นหาข้อบกพร่องต่างๆของโฮสต์และนำไปเป็นข้อมูลเพื่อเจาะเข้าไปยังโฮสต์อื่นภายหลังได้ ยิ่งโฮสต์นั้นขาดการป้องกันจากการสแกนมากเท่าไร ความเสี่ยงที่โฮสต์จะถูกเจาะเข้าไปก็มีมากเท่านั้น

- Inbound Access

กฎข้อแรกที่ผู้ใช้ควรทราบคืออินเตอร์เน็ตเป็นเครือข่ายแห่งเสรีภาพ การสื่อสารบนอินเตอร์เน็ตเป็นไปอย่างอิสระ ทุกคนที่ติดต่อกับอินเตอร์เน็ตสามารถส่งข้อมูลออกไป (Outbound) ที่ใดก็ได้โดยไม่มีการควบคุม ไม่ว่าจะต่อแบบตลอดเวลาโดยใช้ลีสไลน์ (leased line) หรือต่อเป็นครั้งคราวโดยใช้โมเด็ม (Modem) ต่อเข้าไปยัง ISP (Internet Service Provider) ทุกคนมีโอกาสเท่าเทียมกันในการส่งข้อมูลไปยังทุกที่ในโลก และไม่ว่าจะยินยอมหรือไม่ก็มีโอกาสที่จะได้รับข้อมูลเข้ามา (Inbound) ได้ทุกรูปแบบและจากทุกที่ในโลกเช่นกัน ไม่ว่าข้อมูลที่รับเข้ามานั้นจะเป็นสิ่งที่ดี เช่น จดหมาย เว็บไซท์ ข้อความ หรือเป็นสิ่งที่ไม่อยากรับ เช่น ไวรัส โปรแกรมโทรจัน หรือเครื่องมืออื่นใดของแฮคเกอร์ ข้อมูลเหล่านั้นก็จะมาถึงโฮสต์ของผู้ใช้ได้ถ้าวิธีการส่งเป็นวิธีที่โปรโตคอล TCP/IP เข้าใจ การนำโฮสต์มาต่อกับอินเตอร์เน็ตจึงอยู่ในสภาพที่ต้องป้องกันตัวเองจากสิ่งที่ไม่ต้องการ ไม่มีกฎเกณฑ์และหน่วยงานใดมาคุ้มครองป้องกัน ถ้าเจ้าของไม่ทำการป้องกันก็แสดงว่าเปิดโอกาสให้ทุกคนสามารถส่งข้อมูลเข้ามาได้อย่างเสรี

ถึงแม้ว่าผู้ใช้ไม่ประสงค์จะรับข้อมูลที่ไม่ต้องการเข้ามา แต่ด้วยกลไกการสื่อสารข้อมูลตามปกติที่มีกำหนดอยู่ในโปรโตคอล TCP/IP ไม่มีกลไกใดในการป้องกันโฮสต์ตนเองจากข้อมูลที่ไม่ต้องการได้ กลไกปกติที่โปรโตคอลระบุไว้คือถ้าเป็นข้อมูลที่โฮสต์ไม่ต้องการ ให้โฮสต์ตอบปฏิเสธการสื่อสารนั้นกลับไปและไม่นำข้อมูลนั้นส่งไปประมวลผลต่อ แต่ก็หมายถึงข้อมูลเหล่านั้นได้ถูกส่งถึงโฮสต์แล้วโฮสต์จึงรู้ว่าเป็นข้อมูลที่ไม่ต้องการ หรืออีกกรณีหนึ่งคือการที่โฮสต์เปิดบริการเอาไว้แต่ต้องการจำกัดว่าให้บริการเฉพาะบางโฮสต์ เช่น ต้องการให้บริการเฉพาะโฮสต์ที่อยู่ในบริษัทเดียวกันเท่านั้น แต่ว่าเมื่อใดที่โฮสต์ต่อกับอินเตอร์เน็ตก็เท่ากับว่าทุกคนบนอินเตอร์เน็ตสามารถเข้าถึงบริการนี้ได้เช่นกันโดยที่โฮสต์ไม่สามารถป้องกันหรือกลั่นกรองได้

ไฟร์วอลล์ได้เข้ามาเพิ่มความปลอดภัยในส่วนนี้ โดยทำหน้าที่ควบคุมและกลั่นกรองข้อมูลทุกชนิดที่เข้ามาให้เหลือเฉพาะข้อมูลที่ต้องการเท่านั้น โดยไฟร์วอลล์จะทำหน้าที่แบ่งแยกเครือข่ายภายในและเครือข่ายภายนอกออกจากกัน ทำให้การบริการต่างๆบนโฮสต์ยังคงสามารถให้บริการได้เช่นเดิม แต่สามารถจำกัดการขอใช้บริการจากบุคคลภายนอกหรือโฮสต์อื่นที่ไม่ได้รับอนุญาตได้ การที่ ไฟร์วอลล์สามารถควบคุมการเข้าถึงจากภายนอกหรืออินบราวนด์แอคเซส (Inbound Access) ได้ทำให้ความปลอดภัยของโฮสต์มีมากขึ้น อย่างน้อยที่สุดก็ทำให้มีช่องทางในการเข้ามาของภัยคุกคามต่างๆ ลดน้อยลง

จริงๆแล้วการควบคุมการเข้ามาของข้อมูลเป็นหน้าที่หลักที่สำคัญที่สุดของไฟร์วอลล์ทุกชนิดจะต้องมีอยู่ เพราะภัยคุกคามทั้งหลายจะสามารถทำอันตรายให้แก่เครือข่ายหรือโฮสต์ได้นั้นจะต้องมีช่องทางในการเข้ามา ถ้าปราศจากช่องทางที่จะเข้ามาแล้วภัยเหล่านั้นก็ไม่สามารถส่งผลกระทบใดๆได้ และไฟร์วอลล์ก็มีหน้าที่ทำให้ช่องทางนั้นเหลือน้อยลงเท่าที่จำเป็นโดยที่ผู้บริหารระบบสามารถตรวจตราดูแลความปลอดภัยได้ง่ายขึ้น

- **Outbound Access**

นอกจากการป้องกันการเข้ามาของข้อมูลจากภายนอกแล้ว ไฟร์วอลล์ยังสามารถป้องกันข้อมูลภายในไม่ให้ออกไปข้างนอกได้ด้วย ภัยคุกคามต่อระบบคอมพิวเตอร์มิได้เพียงแค่เจาะเข้ามาจากแฮคเกอร์ภายนอกเพียงอย่างเดียวเท่านั้น ภัยคุกคามที่เกิดจากภายในองค์กรก็มีอยู่ไม่น้อยและส่งผลเสียหายได้ไม่น้อยกว่าการถูกเจาะระบบเข้ามาจากภายนอกเสียอีก การที่เครือข่ายเชื่อมต่อกับอินเตอร์เน็ตและเปิดให้ใช้งานได้อย่างไม่มีการควบคุมนั้นจะก่อให้เกิดปัญหาได้หลายลักษณะ เช่น การลักลอบส่งข้อมูล

การที่เครือข่ายต่ออยู่กับอินเตอร์เน็ตโดยตรงทำให้เปิดโอกาสในการเคลื่อนย้ายถ่ายเทข้อมูลจากภายในออกไปยังภายนอกกระทำได้ง่ายขึ้น ข้อมูลที่เป็นความลับของบริษัทอาจจะถูกส่งออกไปได้ภายในเวลาเสี้ยววินาที และในบางกรณีอาจไม่ได้เกิดจากการกระทำของพนักงานเอง แต่อาจจะเกิดจากบุคคลภายนอกที่มีโอกาสเข้าถึงข้อมูลได้ ซึ่งถ้าไม่มีช่องทางในการส่งข้อมูลออกไปก็จะไม่สามารถขโมยความลับไปได้ แต่ถ้ามีเครือข่ายเชื่อมต่ออยู่แล้วก็ทำให้สามารถส่ง

ข้อมูลนั้นออกไปได้ทันที ตัวอย่างเช่น บริษัทที่พัฒนาซอฟต์แวร์ระดับที่มีความสำคัญจะห้ามไม่ให้พนักงานนำสื่อบันทึกข้อมูลใดๆผ่านเข้าออกที่ทำงานได้ เพราะเกรงว่าจะลักลอบนำข้อมูลออกไป โดยจะมีเจ้าหน้าที่คอยตรวจค้นทุกครั้งที่พนักงานเข้าออก แต่ถ้าใช้การส่งข้อมูลผ่านทางเครือข่ายแล้ว พนักงานที่จะขโมยข้อมูลออกไปไม่จำเป็นต้องนำแผ่นดิสก์หรือสื่อใดๆ ติดตัวไปเลย เพียงส่งข้อมูลไปเก็บไว้ที่เครื่องแม่ข่ายอื่นในอินเตอร์เน็ตนอกบริษัท หลังจากนั้นก็สามารถเดินตัวเปล่าผ่านเจ้าหน้าที่รักษาความปลอดภัยออกไปได้โดยไม่มีอะไรเกิดขึ้น

การใช้อินเตอร์เน็ตไปโดยไม่เกิดประโยชน์กับการทำงาน นับว่าเป็นปัญหาที่พบได้มากที่สุดสำหรับองค์กรที่ต่ออยู่กับอินเตอร์เน็ต เพราะแทนที่จะใช้อินเตอร์เน็ตไปเพื่อช่วยให้การทำงานมีประสิทธิภาพขึ้นกลับใช้ไปในความเพลิดเพลินส่วนตัวจนทำให้ประสิทธิภาพในการทำงานลดน้อยลง เช่น ใช้เวลาส่วนใหญ่ไปเพื่อการดูเว็บไซท์ ดาวน์โหลดรูป เพลง ภาพยนตร์ คุยกับเพื่อนๆ ผ่านแชตรูมต่างๆ ซึ่งล้วนแล้วแต่ทำให้เสียเวลาในการทำงานอย่างมาก และส่งผลให้คุณภาพของการทำงานต่ำลงด้วย เพราะใจไปจดจ่ออยู่กับเรื่องอื่นที่ไม่ใช่งาน นอกจากจะทำให้งานของตนเองมีประสิทธิภาพต่ำลงแล้ว ถ้าบริษัทมีแบนด์วิดธ์ (Bandwidth) อยู่จำกัด พนักงานอื่นๆที่จะต้องใช้แบนด์วิดธ์เพื่อรับส่งข้อมูลในเรื่องงานจะได้รับผลกระทบไปด้วย เพราะแบนด์วิดธ์คือความสามารถหรือความเร็วในการรับส่งข้อมูลที่จะใช้งานได้เหลือน้อยลง ข้อมูลที่จะได้รับก็จะช้าลงไป

การใช้เครือข่ายของบริษัทเป็นที่ทดลองเจาะระบบผู้อื่น ถ้าไม่มีการป้องกันแล้วปัญหานี้จะส่งผลเสียหายต่อบริษัทอย่างมาก เพราะอาจจะมีพนักงานของบริษัทที่สนใจอยากเป็นแฮคเกอร์ แต่การทดลองจากที่บ้านโดยผ่านโมเด็มนั้นอาจจะไม่เร็วพอและต้องเสียเงินค่าชั่วโมงอินเตอร์เน็ต และเมื่อนึกได้ว่าเครื่องคอมพิวเตอร์ที่บริษัทนั้นต่อกับอินเตอร์เน็ต ซึ่งเป็นการสะดวกเป็นอย่างยิ่ง จึงใช้เครื่องคอมพิวเตอร์และเครือข่ายของบริษัทเป็นฐานทดลองเทคนิคต่างๆ หรือแม้กระทั่งใช้ไปเจาะระบบคนอื่นและไอพีแอดเดรส (IP Address) ที่ปรากฏที่ปลายทางนั้นเป็น IP Address ของบริษัทแล้ว บริษัทก็เลี่ยงความผิดชอบได้ยาก

- **Network Denial Of Services (DoS)**

การโจมตีเพื่อก่อกวนไม่ให้โฮสต์สามารถให้บริการได้โดยใช้เทคนิคในระดับของเครือข่ายด้วยวิธีการต่างๆ ไม่ว่าจะเป็นการส่งอะนอมาลัสแพ็คเก็ต (Anomalous Packet : แพ็คเก็ตที่มีลักษณะผิดกฎของโปรโตคอลเพื่อทำให้โฮสต์ปลายทางทำงานผิดปกติ) การทำให้เครือข่ายเต็มไปด้วยข้อมูล (Network Flooding) การส่งแพ็คเก็ตจำนวนมากไปยังโฮสต์เพื่อขอใช้บริการ (SYN Flooding) โดยส่วนมากจะสามารถป้องกันได้ด้วยไฟร์วอลล์ หรืออย่างน้อยก็สามารถบรรเทาผลกระทบจากการถูกโจมตีจากหนักเป็นเบาได้ นอกจากในกรณีที่การโจมตีนั้นรุนแรงมากเกินกว่าที่จะป้องกันได้ไฟร์วอลล์ก็จะหยุดทำงานไปก่อน ซึ่งจะทำให้ผู้ดูแลระบบรับรู้ปัญหาได้ทันที นอกจากเป็นการโจมตีเพื่อไม่ให้สามารถให้บริการได้ตามปกติที่มีเป้าหมายไปยังโฮสต์ใดโฮสต์

หนึ่งอย่างเฉพาะเจาะจงผ่านช่องทางที่ไฟร์วอลล์เปิดไว้อยู่แล้ว ซึ่งอย่างนี้ต้องใช้วิธีอื่นในการป้องกันแทน

- Trojan Horse, Backdoor, Back Orifice

ภัยประเภทนี้เกิดจากการที่โฮสต์ภายในเครือข่ายรันโปรแกรมที่เข้าใจว่าเป็นโปรแกรมที่ทำงานตามปกติ แต่ที่จริงแล้วโปรแกรมเหล่านั้นได้ซ่อนการทำงานอื่นไว้เบื้องหลัง โดยทั่วไปโปรแกรมประเภทนี้จะทำงานโดยเปิดช่องทางภายในโฮสต์ที่รันโปรแกรมนั้น ทำให้แฮคเกอร์สามารถใช้ช่องทางที่เปิดทิ้งไว้เข้ามาขโมยข้อมูลในโฮสต์ ดักอ่านข้อมูลในเครือข่าย (Sniffing) หรือแม้กระทั่งทำการควบคุมเครื่องนั้นจากระยะไกล (Remote Control) ผ่านทางเครือข่ายได้ โปรแกรมประเภทนี้มีแจกจ่ายแพร่หลายทั่วไปในเว็บไซท์ การกระจายของโปรแกรมประเภทนี้มักอาศัยการแนบไปกับจดหมายและหลอกลวงให้ผู้ใช้เข้าใจว่าเป็นโปรแกรมทั่วไปที่ไม่มีพิษภัย เมื่อผู้ใช้รันโปรแกรมนั้นก็จะทำการติดตั้งตัวเองและแอบเปิดช่องทางไว้

การที่กล่าวว่าไฟร์วอลล์สามารถป้องกันภัยประเภทนี้ได้ มิได้หมายความว่าไฟร์วอลล์จะสามารถตรวจจับโปรแกรมประเภทนี้ได้ หรือไฟร์วอลล์สามารถเตือนผู้ใช้ไม่ให้รันโปรแกรมได้ แต่ไฟร์วอลล์จะสามารถป้องกันไม่ให้โปรแกรมประเภทนี้สามารถทำงานได้อย่างสมบูรณ์ โดยที่ถึงแม้ว่าผู้ใช้จะมีโปรแกรมประเภทนี้ทำงานอยู่ภายในเครื่อง แต่แฮคเกอร์ก็จะไม่สามารถทำการติดต่อเข้ามายังช่องทางที่โปรแกรมเปิดทิ้งไว้ได้

เนื่องจากโปรแกรมประเภทนี้มักจะต้องอาศัยการสื่อสารข้อมูลที่สมบูรณ์ระหว่างโปรแกรมในโฮสต์กับแฮคเกอร์ ดังนั้นจึงจำเป็นต้องเปิดช่องทางการสื่อสารพิเศษไว้เพื่อให้แฮคเกอร์สามารถเข้ามาได้ สำหรับโปรโตคอล TCP/IP การเปิดช่องทางการสื่อสารขึ้น หมายถึงจะต้องให้โปรแกรมประเภทนี้เปิดพอร์ต (Port) ไว้บนโฮสต์ และพอร์ตที่โปรแกรมเปิดไว้นี่เองที่จะเป็นสิ่งที่บ่งบอกให้รู้ว่าโฮสต์นั้นได้ถูกเปิดช่องทางไว้ ซึ่งโดยทั่วไปในการกำหนดกฎของไฟร์วอลล์แล้วพอร์ตหมายเลขที่โปรแกรมประเภทนี้ใช้งานจะเป็นพอร์ตอันตรายที่เป็นพอร์ตต้องห้ามเสมอ ทำให้แพ็คเก็ตที่ใช้พอร์ตหมายเลขเหล่านี้จะถูกดรอปโดยไฟร์วอลล์เสียก่อนที่จะไปถึงที่หมาย

2.2.2 สิ่งที่ไฟร์วอลล์ไม่สามารถป้องกันได้

- Hacker

แฮคเกอร์ (Hacker) หมายถึง คนหรือกลุ่มคนที่พยายามเจาะเข้าไปยังระบบคอมพิวเตอร์โดยอาศัยทักษะทางคอมพิวเตอร์และข้อบกพร่องของระบบที่เป้าหมายทำงานอยู่ โดยที่วัตถุประสงค์หลักของแฮคเกอร์ คือผ่านระบบรักษาความปลอดภัยเข้าไปให้ได้ไม่ว่าด้วยวิธีใดๆ ดังนั้นถ้าพิจารณาถึงวิธีที่แฮคเกอร์ผ่านเข้าไปยังระบบต่างๆ นั้นย่อมมีความหลากหลายพลิกแพลงไปตามระบบ ไม่มีสูตรตายตัวโดยอาจอาศัยเทคนิคทั้งในระดับเครือข่าย ระดับแอพพลิเคชั่นหรือแม้กระทั่งในระดับผู้ใช้ร่วมกันเพื่อให้บรรลุวัตถุประสงค์ที่ต้องการ

การที่จะระบุได้ว่าไฟร์วอลล์สามารถป้องกันแฮคเกอร์ได้อย่างสมบูรณ์นั้นไม่อาจเป็นไปได้ เพราะไฟร์วอลล์ไม่ได้ทำการตัดขาดเครือข่ายภายในออกจากอินเตอร์เน็ตโดยสมบูรณ์ ถึงอย่างไรไฟร์วอลล์ก็จะเปิดช่องไว้ให้การสื่อสารข้อมูลบางประเภทเข้ามายังภายในเครือข่ายได้เสมอ การมีไฟร์วอลล์ทำให้ระดับความยุ่งยากของการเจาะเข้าไปในระบบของแฮคเกอร์มีมากขึ้น และอาจทำให้แฮคเกอร์มือสมัครเล่นท้อถอยและหมดความพยายาม แต่สำหรับแฮคเกอร์ที่มีประสบการณ์ทักษะและเครื่องมือในการเจาะระบบที่เพียงพอ ไฟร์วอลล์ก็เป็นเพียงอุปสรรคในระดับปานกลางเท่านั้น ในไม่ช้าก็จะสามารถค้นพบวิธีที่จะเล็ดลอดผ่านเข้าไปได้ในที่สุด

ถึงแม้ไฟร์วอลล์ไม่สามารถป้องกันแฮคเกอร์ได้ก็จริง แต่นั่นไม่ได้หมายความว่าเครือข่ายที่มีไฟร์วอลล์อยู่นั้นมีสภาพเช่นเดียวกับเครือข่ายที่ไม่มีไฟร์วอลล์ การมีไฟร์วอลล์จะทำให้การสื่อสารข้อมูลนั้นอยู่ภายใต้การควบคุมระดับความปลอดภัยต่างๆเพิ่มมากขึ้น มีความยากลำบากในการเจาะระบบผ่านเข้าไป และลดระดับความเสี่ยงโดยรวมของเครือข่ายลงได้

- **Allowed Service**

ไฟร์วอลล์จะควบคุมการสื่อสารข้อมูลโดยใช้กฎเป็นสำคัญ ไม่มีกฎที่ผิดและกฎที่ถูกแน่นอนตายตัว กฎที่ผิดสำหรับเครือข่ายหนึ่งอาจเป็นกฎที่ถูกสำหรับอีกเครือข่ายหนึ่งได้ ในทางกลับกันกฎที่ถูกต้องของอีกเครือข่ายหนึ่งก็อาจเป็นสิ่งที่ต้องห้ามสำหรับอีกเครือข่ายก็เป็นไปได้ เพราะปัจจัยที่สำคัญในการกำหนดกฎ คือนโยบายความปลอดภัยและลักษณะการบริการที่มีอยู่ภายในเครือข่ายนั้น การพิจารณากฎที่มีอยู่บนไฟร์วอลล์จึงมีเพียงความเหมาะสมของกฎเท่านั้นว่าสอดคล้องกับนโยบายหรือไม่ เหมาะสมหรือไม่ รัดกุมเพียงพอหรือไม่ และมีประโยชน์ในแง่ของการป้องกันหรือไม่

ในการกำหนดกฎนั้นการสื่อสารข้อมูลโดยส่วนใหญ่เกือบทั้งหมดจะถูกบล็อค (Block) ไว้ และจะอนุญาตเฉพาะการสื่อสารข้อมูลสำหรับบริการบางประเภทที่นโยบายความปลอดภัยอนุญาตไว้เท่านั้น การสื่อสารข้อมูลเหล่านั้นก็จะสามารถผ่านเข้าออกไฟร์วอลล์ได้ แต่อย่างไรก็ตามการบริการที่ได้รับอนุญาตให้ผ่านนั้นก็ไม่ได้หมายความว่าจะเป็นการสื่อสารข้อมูลที่มีความปลอดภัยแต่อย่างใด

สำหรับมุมมองในแง่ของการสื่อสารข้อมูลแล้ว การบริการใดที่ได้รับอนุญาตจากไฟร์วอลล์จะทำตัวเสมือนว่าไม่มีการป้องกัน เช่น ถ้าไฟร์วอลล์อนุญาตให้มีการเรียกใช้บริการของเว็บเซิร์ฟเวอร์ (Web Server) ที่อยู่ภายในเครือข่ายได้ (พอร์ต 80) หมายถึงการสื่อสารข้อมูลใดๆที่กระทำอยู่ภายใต้บริการของเว็บเซิร์ฟเวอร์ก็จะได้รับอนุญาตทั้งหมด ถ้ามีการดุกคามใดที่แฝงมากับการเรียกใช้บริการ เว็บเซิร์ฟเวอร์แล้วไฟร์วอลล์ก็ไม่สามารถตรวจสอบและป้องกันได้

ถ้าจะให้เห็นภาพได้ชัดเจนกว่า คือสมมติว่าไฟร์วอลล์เปิดให้มีการสื่อสารข้อมูลของบริการเทลเน็ต (Telnet) ซึ่งเป็นบริการที่สามารถสั่งงานและควบคุมโฮสต์จากระยะไกลผ่านเครือข่ายให้สื่อสารผ่านจากอินเตอร์เน็ตเข้ามายังโฮสต์ที่ตั้งอยู่เครือข่ายภายในได้ ไฟร์วอลล์จะไม่สามารถ

ควบคุมการใช้งานของบริการTelnet นั้นๆได้ ไม่ว่าผู้ที่กำลังใช้ Telnet ในขณะนั้นจะเป็นผู้ใช้งานจริงๆหรือเป็นแฮคเกอร์ก็ตาม เพราะไฟร์วอลล์จะควบคุมการสื่อสารโดยพิจารณาจากบริการเป็นหลักแต่จะไม่พิจารณาเนื้อหา ซึ่งการบริการนั้นก็อาจจะมีเนื้อหาที่ดีและอาจจะมีเนื้อหาที่มีไว้ทำลายล้างก็เป็นไปได้

- Application Vulnerability

Vulnerability คือ ข้อบกพร่องที่เปรียบเสมือนช่องโหว่ของระบบต่างๆ ถ้าอยู่ในระดับแอพพลิเคชั่นจะหมายถึงข้อบกพร่องที่มีอยู่ภายในแอพพลิเคชั่นนั้นที่เป็นช่องทางให้แฮคเกอร์สามารถผ่านเข้ามาได้โดยง่าย ข้อบกพร่องเหล่านี้จะติดตัวอยู่กับแอพพลิเคชั่น เมื่อนำแอพพลิเคชั่นที่มีปัญหาไปบริการจะทำให้ในขณะที่แอพพลิเคชั่นทำงานอยู่นั้นก็จะเป็นการเปิดช่องทางให้แฮคเกอร์ผ่านเข้ามาภายในระบบได้ด้วยข้อบกพร่องของแอพพลิเคชั่นเอง

ตัวอย่างเช่น เราอาจจะตั้งเว็บเซิร์ฟเวอร์เพื่อให้คนเข้ามาเยี่ยมชมดูเว็บไซท์ของเราโดยที่เว็บเซิร์ฟเวอร์นี้อยู่หลังไฟร์วอลล์และอยู่ในบริเวณที่ปลอดภัย โดยที่ทราฟฟิกหรือการสื่อสารข้อมูลจากภายนอกที่เข้ามาถึงเว็บเซิร์ฟเวอร์นั้นอนุญาตเพียงการให้บริการบนพอร์ต 80 ซึ่งเป็นพอร์ตสำหรับเว็บเท่านั้น อย่างไรก็ตามปรากฏว่าเราได้ใช้ Microsoft IIS 4.0 เป็นเว็บเซิร์ฟเวอร์โดยที่ไม่ได้รับการติดตั้งแพตช์ (Patch) หรือโปรแกรมเพิ่มเติมเพื่อแก้ไขปัญหาข้อบกพร่องที่มีอยู่ เมื่อมีผู้ใช้บริการเรียกดูเว็บไซท์โดยผ่านบราวเซอร์มายังเว็บเซิร์ฟเวอร์นี้ก็จะสามารถผ่านไฟร์วอลล์เข้ามาได้ และเมื่อมีการป้อนรหัสพิเศษบางอย่างมายังเว็บเซิร์ฟเวอร์จะกลายเป็นการเปิดช่องโหว่ของ IIS ที่มีอยู่และทำให้ผู้ใช้นั้นสามารถควบคุมเครื่องแม่ข่ายโดยผ่านบราวเซอร์ได้ทันที การเจาะเข้าไปใน IIS โดยใช้บราวเซอร์เป็นเครื่องมือนั้นมีการกระทำกันอย่างแพร่หลาย และมีหลายวิธีไม่ว่าการอาศัย FrontPage Extension, Dot Dot Directory traversal, Unicode, Buffer Overflow โดยที่ทาง Microsoft ต้องออกแพตช์มาทำการแก้ไขช่องโหว่เหล่านี้อยู่อย่างสม่ำเสมอ

ซึ่งในกรณีเช่นนี้สำหรับไฟร์วอลล์ก็ได้ทำหน้าที่อย่างเต็มที่แล้วโดยอนุญาตให้มีทราฟฟิกผ่านเข้ามาเฉพาะพอร์ต 80 เท่านั้น แต่ในเมื่อแอพพลิเคชั่นที่ให้บริการอยู่บนพอร์ตดังกล่าวมีปัญหาเองก็สุดวิสัยของไฟร์วอลล์ที่จะป้องกันข้อบกพร่องที่มีอยู่บนแอพพลิเคชั่นแต่ละตัวได้

ไม่ใช่ว่าจะมีเฉพาะ IIS เท่านั้นที่มีปัญหา ที่จริงแล้วแอพพลิเคชั่นแทบทุกชนิดทุกค่ายต่างก็มีปัญหาทั้งสิ้น ดังนั้นจึงต้องตระหนักว่าไฟร์วอลล์สามารถควบคุมได้เพียงให้ทราฟฟิกที่ผ่านเข้าออกนั้นใช้เฉพาะพอร์ตที่ต้องการเท่านั้น แต่เมื่อทราฟฟิกนั้นได้ใช้บริการตามพอร์ตที่ได้รับอนุญาตแล้ว ก็ถือว่าหมดหน้าที่ของไฟร์วอลล์ ถ้าจะมีปัญหาด้านความปลอดภัยของแอพพลิเคชั่นก็ต้องเป็นหน้าที่ที่จะต้องแก้ไขที่แอพพลิเคชั่นนั้นเอง

- OS Vulnerability

ระบบปฏิบัติการเป็นซอฟต์แวร์ชนิดหนึ่งเช่นเดียวกับแอพพลิเคชั่น แต่ทำหน้าที่ในคนละระดับชั้นหรือเลเยอร์ (Layer) ซึ่งมีข้อบกพร่องเช่นเดียวกับซอฟต์แวร์อื่นๆ ดังนั้นถ้าระบบ

ปฏิบัติการมีปัญหาก็อยู่นอกเหนือความสามารถที่ไฟร์วอลล์จะแก้ไขได้ อย่างไรก็ตาม ถึงแม้ไฟร์วอลล์จะไม่สามารถแก้ไขข้อบกพร่องดังกล่าวได้ แต่ถ้ามีการกำหนดกฎที่ดีไฟร์วอลล์จะช่วยบรรเทาและลดความเสี่ยงจากการเจาะเข้ามาโดยช่องทางนี้ได้ ซึ่งแตกต่างจากข้อบกพร่องของแอพพลิเคชั่น เนื่องจากแอพพลิเคชั่นจำเป็นต้องเปิดพอร์ตให้บริการกับผู้ใช้อยู่แล้ว และแฮคเกอร์ก็จะใช้ช่องทางเดียวกันในการเจาะระบบเข้ามา แต่สำหรับระบบปฏิบัติการแล้วอาจจะไม่มีความจำเป็นต้องเปิดพอร์ตให้บริการกับผู้ใช้แต่อย่างใด ดังนั้นไฟร์วอลล์จึงสามารถป้องกันไม่ให้แฮคเกอร์เข้าถึงพอร์ตของระบบปฏิบัติการได้

อย่างเช่นการใช้ Windows NT หรือ Windows 2000 เป็นเว็บเซิร์ฟเวอร์ซึ่งจะให้บริการเฉพาะพอร์ต 80 สำหรับผู้ใช้ทั่วไปเท่านั้น แต่เครื่องแม่ข่ายจำเป็นต้องเปิดพอร์ตอื่นๆ เพื่อใช้งานด้วยเช่น TCP 135, 136 เพื่อการทำงานของระบบปฏิบัติการ ซึ่งพอร์ตดังกล่าวของ Windows มีปัญหาด้านความปลอดภัย แต่ถ้าเครื่องแม่ข่ายได้ติดตั้งอยู่หลังไฟร์วอลล์แล้ว ผู้ใช้จากภายนอกก็จะสามารถเข้าถึงได้เฉพาะพอร์ต 80 ตามที่ไฟร์วอลล์อนุญาตเท่านั้น ส่วนพอร์ต 135, 136 ที่ถึงแม้ว่าจะมีปัญหาแต่ผู้ใช้จากภายนอกไม่สามารถผ่านเข้ามาถึงได้ ถ้าลักษณะนี้ก็เป็นการป้องกันที่ไฟร์วอลล์สามารถช่วยป้องกันได้

แต่สำหรับในบางกรณีที่การทำงานนั้นมีส่วนคาบเกี่ยวกันระหว่างระบบปฏิบัติการและแอพพลิเคชั่น ไฟร์วอลล์ก็ไม่สามารถช่วยป้องกันได้อยู่ดี เช่น จากตัวอย่างข้างต้น ถึงแม้ว่าไฟร์วอลล์จะปิดไม่ให้แฮคเกอร์สามารถติดต่อผ่านไฟร์วอลล์ได้โดยเปิดเฉพาะพอร์ต 80 ของเว็บเซิร์ฟเวอร์ แต่การที่เว็บเซิร์ฟเวอร์จะสามารถสื่อสารกับผู้ใช้ได้ในการทำงานจริงแล้วระบบปฏิบัติการจะต้องเป็นผู้รับแพ็คเก็ตที่เข้ามาจัดการรูปแบบของข้อมูลก่อนแล้วจึงส่งไปให้แอพพลิเคชั่นจัดการต่อ ดังนั้นระบบปฏิบัติการจึงมีส่วนเกี่ยวข้องกับการให้บริการของเว็บเซิร์ฟเวอร์โดยทางอ้อม ข้อบกพร่องบางประเภทจึงส่งผลกระทบโดยตรงกับวิธีการจัดการสื่อสารข้อมูลของระบบปฏิบัติการ ซึ่งส่วนใหญ่จะเป็นความเปราะบางต่อการโจมตีแบบ DoS หรือ Denail of Service ซึ่งจะพบว่าระบบปฏิบัติการแต่ละค่ายจะมีความสามารถและข้อบกพร่องในการจัดการกับแพ็คเก็ตที่เข้ามาต่างกัน ดังนั้นแฮคเกอร์จึงสามารถส่งแพ็คเก็ตแปลกประหลาดที่ไม่มีกำหนดอยู่ในโปรโตคอลแล้วทำให้กระบวนการสื่อสารของข้อมูลของระบบปฏิบัติการทำงานผิดปกติและหยุดทำงานไปในที่สุด ซึ่งจะส่งผลกระทบให้แอพพลิเคชั่นทั้งหมดที่ทำงานอยู่นบระบบปฏิบัติการนั้นหยุดทำงานไปด้วย การ DoS ในลักษณะนี้ที่พบได้ทั่วไป เช่น Teardrop, UDP Bomb, Winnuke และ Jolt เป็นต้น ซึ่งลักษณะข้อบกพร่องเหล่านี้เป็นคุณสมบัติประจำตัวของระบบปฏิบัติการนั้นอยู่เอง ถ้าได้รับแพ็คเก็ตที่มีปัญหาเมื่อใดระบบปฏิบัติการก็จะหยุดทำงานไปในทันที ดังนั้นไม่ว่าแพ็คเก็ตจะสามารถเข้าถึงระบบปฏิบัติการด้วยพอร์ตใดก็ตามก็มีสิทธิ์ที่จะทำให้ระบบปฏิบัติการหยุดทำงานได้ทั้งสิ้น ซึ่งถึงแม้ว่าไฟร์วอลล์จะป้องกันไว้อย่างสุดความสามารถแล้วก็ตาม ถ้ามีพอร์ต

ใดที่เปิดอนุญาตให้แพ็คเก็ตเข้ามาได้ ก็ย่อมมีความเสี่ยงเกิดขึ้นแน่นอน และสุดวิสัยที่ไฟร์วอลล์ทั่วไปจะป้องกันได้

- Virus

ไวรัสเป็นที่รู้จักกันมานาน โด่งดังมาก่อนแฮคเกอร์ และน่าจะเป็นภัยคุกคามทางคอมพิวเตอร์รุ่นแรกๆที่มีผลกระทบต่อสาธารณชน ไวรัสมีอายุยืนยาวและพัฒนาการแพร่กระจายมาเป็นลำดับอย่างต่อเนื่อง เรียกว่ามีไวรัสใหม่ออกมาแทบทุกเดือน แต่นานกว่าจะมีที่สร้างความเสียหายมาก เช่น I Love You, Anna Kurnilova หรือ Sircam เป็นต้น

เข้าใจว่าโดยทั่วไปนักคอมพิวเตอร์ส่วนใหญ่นั้นพอจะทราบอยู่แล้วว่าไฟรวอลล์ไม่สามารถป้องกันไวรัสได้เพราะเป็นเครื่องมือป้องกันที่ทำงานอยู่คนละส่วน แต่ระยะหลังเนื่องจากช่องทางในการแพร่กระจายของไวรัสอาศัยอีเมล์และอินเตอร์เน็ตเป็นช่องทางหลัก ก็อาจมีผู้สนใจขึ้นมาว่าในเมื่อไวรัสมาทางอินเตอร์เน็ตแล้วไฟร์วอลล์ก็น่าจะป้องกันได้

การที่ระบุถึงไวรัสในหัวข้อนี้เพื่อเป็นการแสดงให้เห็นว่า ไฟร์วอลล์ไม่สามารถป้องกันไวรัสได้ ถึงแม้ว่าไวรัสจะมาทางอินเตอร์เน็ตและเดินทางผ่านไฟรวอลล์มาก็ตาม ไม่ใช่ว่าไฟร์วอลล์ไม่มีความสามารถ แต่เนื่องจากไฟร์วอลล์เป็นเครื่องมือป้องกันในระดับเน็ตเวิร์กเลเยอร์ที่คอยควบคุมทราฟฟิกเป็นหลัก ซึ่งจะทำงานอยู่ในเลเยอร์ที่ต่ำกว่าการทำงานของไวรัส ไฟร์วอลล์จะไม่รู้จักว่าอะไรคือไวรัส อะไรคือจดหมาย ไฟร์วอลล์จะรู้จักเฉพาะ TCP, UDP, ICMP พอร์ตเท่านั้น ดังนั้นการคาดหวังให้ไฟรวอลล์ป้องกันหรือกำจัดไวรัสนั้นจึงไม่สามารถเป็นไปได้ การทำงานของไวรัสจะอยู่ในระดับแอพลิเคชั่น อาศัยการเอ็กซิคิวต์ (Execute) คำสั่งของผู้ใช้ และระบบปฏิบัติการเป็นหลัก สิ่งเดียวที่ไวรัสเกี่ยวข้องกับอินเตอร์เน็ตและไฟร์วอลล์ คืออาศัยเป็นทางผ่านเท่านั้น ถ้าไวรัสแพร่ระบาดโดยอีเมล์ โดยที่เรามีเมล์เซิร์ฟเวอร์อยู่และกำหนดให้ไฟร์วอลล์อนุญาตให้มีการรับทราฟฟิกเพื่อการรับส่งอีเมล์ได้แล้ว ไวรัสก็สามารถผ่านเข้ามาได้เช่นเดียวกับจดหมายอื่นๆ เช่นกัน

- การดักอ่านข้อมูลโดย Sniffer

สนิฟเฟอร์ (Sniffer) เป็นเครื่องมือชนิดหนึ่งที่เมื่อนำไปต่อในเครือข่ายร่วมกับโฮสต์อื่นๆ แล้ว ทำให้สามารถดักอ่านข้อมูลของโฮสต์อื่นที่ใช้งานอยู่บนฮับ (Hub) เดียวกันได้ ไฟร์วอลล์ไม่มีหน้าที่และความสามารถในการป้องกันการดักอ่านข้อมูล ดังนั้นไฟร์วอลล์จึงไม่สามารถป้องกันการแอบอ่านอีเมล์ของผู้อื่นหรือป้องกันข้อมูลรั่วไหลเนื่องจากการมีสนิฟเฟอร์ได้

- Spammed Mail

สแปมเมล์ (Spammed Mail) หรืออีเมล์ขยะที่ได้รับมาโดยที่ผู้ใช้ไม่ต้องการ อาจจะเป็นเมล์โฆษณาชวนเชื่อ จดหมายลูกโซ่ และไม่สามารถติดตามหาผู้ส่งจริงๆได้ อาจมีอันตรายไม่มาก แต่จะส่งผลในการสร้างความรำคาญให้แก่ผู้รับ ทำให้ประสิทธิภาพในการทำงานของระบบเมล์ลดต่ำลง

และสิ้นเปลืองทรัพยากรไปเพื่อจัดเก็บเมล์ขยะพวกนี้ โดยทั่วไปแล้วเราอาจจะได้รับผลกระทบได้ใน 2 ฐานะ คือ

1. เป็นผู้รับเมล์ ผู้ส่งอาจจะได้ชื่อมาจากที่ใดก็ได้ แต่ได้ส่งมายังผู้รับเมล์โดยตรงโดยที่ผู้รับไม่เคยติดต่อและไม่ต้องการได้รับเมล์นั้นแม้แต่น้อย ส่วนใหญ่ก็ต้องคอยตามลบเมล์เหล่านั้นทิ้งออกไป
2. ถูกใช้เป็นตัวกลางสำหรับส่งต่อเมล์ ลักษณะนี้จะเป็นส่งผลกระทบเฉพาะกับเมล์เซิร์ฟเวอร์ (Mail Server) เท่านั้น เนื่องจากเมล์เซิร์ฟเวอร์อาจจะถูกติดตั้งกำหนดค่าไว้ไม่เหมาะสม ทำให้สามารถถูกใช้ไปเพื่อเป็นตัวกลางในการกระจายเมล์ขยะไปหาผู้อื่นได้

ทั้ง 2 กรณีจะส่งผลต่อผู้ใช้ต่างกันแต่ก็ไม่เป็นผลดีทั้งคู่ ถึงแม้ว่าเป็นภัยที่มาจากการส่งเมล์บนอินเตอร์เน็ตโดยที่ทราฟฟิกของเมล์เหล่านี้จะเดินทางผ่านไฟร์วอลล์ก็ตาม แต่ไฟร์วอลล์ก็ไม่สามารถกลั่นกรองเมล์พวกนี้ออกไปได้ สำหรับเมล์ทุกฉบับที่ผ่านเข้าออกนั้นในมุมมองของไฟร์วอลล์แล้วมีค่าเท่าเทียมกันและมีคุณสมบัติเหมือนกัน ดังนั้นจึงทำให้ไม่สามารถทำการตรวจจับสแปมเมล์และทำการดร็อปทราฟฟิกเหล่านั้นออกไปได้ การแก้ไขปัญหาของสแปมเมล์จะต้องแก้ไขในระดับของเมล์เซิร์ฟเวอร์

- **Administration Mistake**

จากที่กล่าวมาข้างต้นจะเป็นปัญหาทางเทคนิคที่ไฟร์วอลล์ไม่สามารถป้องกันได้ แต่ยังมีข้อผิดพลาดที่สำคัญมากที่สุดสิ่งหนึ่ง ซึ่งผู้ที่จะนำไฟร์วอลล์ไปใช้จะต้องตระหนัก คือความหละหลวมส่วนหนึ่งที่เกิดจากผู้บริหารระบบ หรือ System Administrator ได้ปฏิบัติงานผิดพลาดเองและเป็นช่องทางที่ก่อให้เกิดความไม่ปลอดภัยขึ้นในระบบได้ เช่น การกำหนดสิทธิ์ในการใช้งานผิดพลาด โดยอนุญาตให้ผู้ใช้สามารถอัพโหลด (Upload) ไฟล์ขึ้นไปยังเซิร์ฟเวอร์ได้โดยไม่ต้องทำการล็อคอิน (Log in) การไม่มีการป้องกันไฟล์ที่มีความสำคัญให้ดี เปิดโอกาสให้ผู้ใช้สามารถนำไฟล์ Passwd มาทำการ crack หรือแกะดูข้อมูลเพื่อหารหัสผ่านได้ การติดตั้งโปรแกรมประเภท remote control แต่ไม่มีการควบคุมการใช้งานอย่างเพียงพอ ทำให้แฮคเกอร์ก็นำโปรแกรมนั้นไปใช้งานด้วย เป็นต้น เหล่านี้ล้วนแต่จะนำมาซึ่งความไม่ปลอดภัยของระบบโดยรวมได้มาก ดังนั้นการพิจารณาความปลอดภัยของระบบรวมจึงจะต้องตระหนักถึงความเสี่ยงในเรื่องการปฏิบัติงานของบุคคลให้มากและควรมีมาตรการและแนวทางการปฏิบัติที่จะช่วยลดปัญหาเหล่านี้โดยไม่หวังพึ่งแต่การป้องกันจากไฟร์วอลล์ฝ่ายเดียว

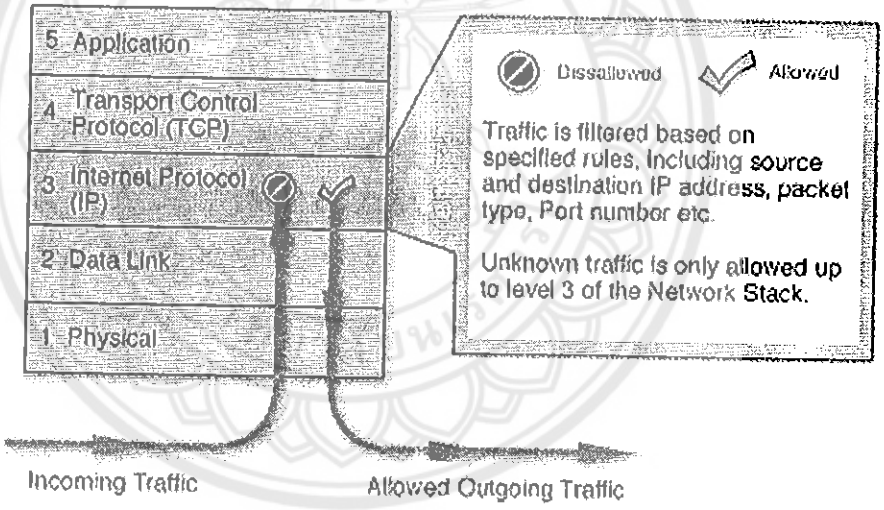
2.3 ประเภทของไฟร์วอลล์

ด้วยคำจำกัดความพื้นฐานที่กำหนดให้ไฟร์วอลล์เป็นเครื่องมือควบคุมทราฟฟิกในเครือข่ายได้นั้น ทำให้มีเครื่องมือที่สามารถทำหน้าที่ดังกล่าวได้หลายลักษณะและต่างเรียกว่า

ไฟร์วอลล์เหมือนกัน ซึ่งโดยแท้จริงแล้วลักษณะการทำงานและขีดความสามารถในรายละเอียดรวมทั้งขีดจำกัดนั้นก็แตกต่างกันมาก การนำไปใช้งานและความเหมาะสมก็แตกต่างกันออกไป ซึ่งถ้าจะทำการจำแนกไฟร์วอลล์โดยการใช้ลักษณะการทำงานเป็นเกณฑ์แล้วจะสามารถจำแนกได้ดังนี้

2.3.1 แพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์ (Packet Filtering Firewall)

เป็นไฟร์วอลล์พื้นฐานที่มีความสามารถในการควบคุมทราฟฟิกโดยอาศัยการตรวจสอบข้อมูลที่ปรากฏอยู่ในแพ็คเก็ต ไฟร์วอลล์ประเภทนี้อาจจะเป็นความสามารถที่เพิ่มมาในเราเตอร์ (Router) โดยการทำงานจะอาศัยโครงสร้างพื้นฐานที่เราเตอร์มีอยู่ให้ทำหน้าที่มากกว่าการจัดเส้นทาง (Route) ให้แพ็คเก็ตไปตามทิศทางที่เหมาะสมเพียงอย่างเดียว แต่จะทำการตรวจสอบเปรียบเทียบกับเงื่อนไขที่กำหนดไว้ก่อนจึงจะทำการจัดส่งแพ็คเก็ตออกไป ไฟร์วอลล์ประเภทนี้ทำงานในระดับเน็ตเวิร์กเลเยอร์ (Network layer) ของ OSI model หรืออินเตอร์เน็ตโปรโตคอลเลเยอร์ (Internet Protocol layer) ของ TCP/IP model ดังรูปที่ 2.1



รูปที่ 2.1 การทำงานของแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์

ก่อนทราบถึงการทำงานของแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์นั้น จะต้องศึกษาและทำความเข้าใจองค์ประกอบของแพ็คเก็ตก่อน ดังนี้

แพ็คเก็ต เป็นหน่วยพื้นฐานของการรับส่งข้อมูลของดาต้าลิงค์เลเยอร์ (Data Link layer) ของ TCP/IP model การรับส่งข้อมูลแต่ละครั้งของดาต้าลิงค์เลเยอร์จะส่งข้อมูลออกไปชุดหนึ่ง โดยที่ความยาวของชุดข้อมูลนี้จะมีเท่าใดนั้นจะเป็นไปตามคุณสมบัติของดาต้าลิงค์เลเยอร์นั้นๆ ข้อมูลแต่ละชุดนั้นเรียกว่าแพ็คเก็ต สำหรับโปรโตคอล TCP/IP นั้นจะใช้ IP เป็นโปรโตคอลหลักในการขนส่งข้อมูลระหว่างโฮสต์ โดย IP ซึ่งอยู่ในอินเตอร์เน็ตโปรโตคอลเลเยอร์จะส่งข้อมูลลงไปยัง

ดาต้าลิงค์เลเยอร์ตามลำดับ โดยที่ถ้าขนาดของดาต้าแกรม (Datagram) ที่จะส่งนั้นสามารถส่งไปได้โดยแพ็คเก็ตเดียว IP ก็จะส่งดาต้าแกรมนั้นไปทันที และแพ็คเก็ตนั้นคือข้อมูล 1 ดาต้าแกรม แต่ถ้าขนาดของดาต้าแกรมใหญ่กว่าขนาดของดาต้าลิงค์เลเยอร์แล้ว IP ก็จะต้องทำการแบ่งส่วนหรือแฟรกเมนเตชั่น (Fragmentation) คือกระจายดาต้าแกรมออกเป็นส่วนย่อยก่อนแล้วจึงค่อยส่งลงไปที่ดาต้าลิงค์เลเยอร์ ซึ่งในกรณีนี้ข้อมูล 1 แพ็คเก็ตจะเป็นเพียงส่วนย่อยหรือแฟรกเมนต์ (Fragment) หนึ่งของดาต้าแกรมเท่านั้น ดังนั้นข้อมูล 1 แพ็คเก็ตจึงไม่จำเป็นต้องเป็นข้อมูล 1 ดาต้าแกรมเสมอไป แต่อย่างไรก็ตามแพ็คเก็ตทุกแพ็คเก็ตที่ส่งมาจาก IP จะมีข้อมูลอย่างน้อยที่สุดคือ IP Address ต้นทางและปลายทางเสมอ ซึ่งจำเป็นสำหรับให้แพ็คเก็ตนั้นเดินทางต่อไปจนถึงที่หมายปลายทางได้

ถ้าข้อมูล 1 แพ็คเก็ตนั้นบรรจุครบถ้วนทั้งดาต้าแกรมแล้วจะทำให้สามารถทราบถึงข้อมูลของโปรโตคอลในเลเยอร์ที่สูงขึ้นไปด้วยว่าเป็น ICMP, TCP, UDP หรือโปรโตคอลอื่นใดที่อาศัยอยู่ใน ดาต้าแกรมนั้น แต่ถ้าแพ็คเก็ตนั้นไม่สามารถบรรจุข้อมูลได้ครอบคลุมทั้งดาต้าแกรมแล้ว ก็จะทำให้เพียงแต่ทราบว่าแพ็คเก็ตนั้นเป็น IP แพ็คเก็ตเท่านั้น ดังนั้นถ้ามีการกล่าวถึงแพ็คเก็ตของ TCP หรือของโปรโตคอลในเลเยอร์สูงขึ้นไปเมื่อใดนั่นหมายถึงการกล่าวถึงแพ็คเก็ตที่มีดาต้าแกรมสมบูรณ์จนทำให้สามารถทราบได้ถึงข้อมูลในเลเยอร์ของ TCP หรือโปรโตคอลในเลเยอร์สูงขึ้นไปนั่นเอง

ข้อมูลที่สำคัญของแพ็คเก็ต

ภายในของแต่ละแพ็คเก็ตนั้นจะประกอบด้วยข้อมูลสำคัญซึ่งสามารถนำมาใช้เพื่อเป็นเงื่อนไขสำหรับการควบคุมทราฟฟิกโดยไฟร์วอลล์ดังนี้

- **Source IP Address** : IP Address ของต้นทางเพื่อใช้ในการพิจารณาต้นทางของข้อมูลว่าอยู่ในเงื่อนไขที่อนุญาตหรือไม่
- **Destination IP Address** : IP Address ของปลายทางเพื่อใช้ในการพิจารณาปลายทางของข้อมูลว่าอยู่ในเงื่อนไขที่อนุญาตหรือไม่
- **Protocol** : ระบุโปรโตคอลที่อาศัยอยู่ในดาต้าแกรมที่กำลังพิจารณานี้
- **Source Port** : ระบุพอร์ตต้นทางสำหรับโปรโตคอลที่ใช้พอร์ตคือ TCP และ UDP ซึ่งข้อมูลพอร์ตต้นทางนี้ส่วนใหญ่มักจะมีความสำคัญในลำดับรองลงไปและไม่ถูกนำมาใช้ควบคุมทราฟฟิกมากนัก
- **Destination Port** : ระบุพอร์ตปลายทางที่แพ็คเก็ตนี้ต้องการติดต่อด้วยสำหรับโปรโตคอลที่ใช้พอร์ตคือ TCP และ UDP
- ข้อมูลสำคัญอื่นๆ ตามลักษณะของโปรโตคอล เช่น TCP Flag, ICMP Massage เป็นต้น

ข้อมูลทั้ง 6 ส่วนนี้จะมีได้อย่างครบถ้วนสมบูรณ์ก็ต่อเมื่อแพ็คเก็ตนั้นมีข้อมูลของดาต้าแกรมครบถ้วนทั้งหมด ถ้าข้อมูลแพ็คเก็ตนั้นเป็นแฟรกเมนต์อาจจะทำให้ข้อมูลในส่วนที่ 3 เป็นต้นไปซึ่งอยู่ในโปรโตคอลที่อยู่เลเยอร์สูงกว่า IP ไม่สมบูรณ์ อย่างไรก็ตามไฟร์วอลล์ส่วนใหญ่

ทำการติดตั้งใช้งานในแลนซึ่งมีขนาดของแพ็คเก็ตที่ใหญ่พอสำหรับรองรับดาต้าแกรมได้ทั้งหมด จึงไม่ค่อยพบปัญหายกเว้นมีการแฟรกเมนต์โดยความประสงค์ของ IP เอง

จากข้อมูลที่สำคัญของแพ็คเก็ต จะสามารถนำมาใช้เป็นเงื่อนไขสำหรับควบคุมการผ่านเข้าออกของข้อมูลได้ โดยการพิจารณาข้อมูลทั้งหมดให้เป็นไปตามกฎที่ระบุไว้ ซึ่งเรียกว่า แอคเซสรูล (Access Rules) หรือกฎของการควบคุมการผ่านเข้าออกของแพ็คเก็ต โดยทั่วไปรูปแบบของแอคเซสรูลเบื้องต้นจะเป็นดังนี้

Source Address	Destination Address	Protocol	Services(Dest. Port)	Action

โดยที่ข้อมูลทั้งหมดจะเป็นเสมือนตัวแปรที่จะนำมาเปรียบเทียบกับค่าที่ระบุไว้ในแอคเซสรูลทีละค่า เงื่อนไขในการเปรียบเทียบของแต่ละตัวแปรจะเป็นตรรกะ "และ" ส่วนข้อมูลฟิลด์ (field) สุดท้าย หมายถึงสิ่งที่ไฟร์วอลล์จะกระทำเมื่อค่าในแพ็คเก็ตนั้นตรงกับเงื่อนไข ตัวอย่างเช่น

Source Address	Destination Address	Protocol	Services(Dest. Port)	Action
192.1.0.5	ANY	TCP	25	Accept

นั่นหมายถึงแอคเซสรูลนี้ได้ระบุไว้ว่าจะอนุญาตให้แพ็คเก็ตที่มีต้นทาง IP Address 192.1.0.5 และปลายทางใดๆ และใช้โปรโตคอล TCP และหมายเลขพอร์ตปลายทางเท่ากับ 25 ผ่านไฟร์วอลล์ไปได้ ถ้าไฟร์วอลล์มีแอคเซสรูลนี้เพียงข้อเดียวก็เท่ากับอนุญาตให้โฮสต์เดียวคือโฮสต์ที่มี IP Address 192.1.0.5 เท่านั้นที่สามารถใช้บริการ SMTP (TCP พอร์ต 25) ไปยังโฮสต์อื่นที่อยู่อีกด้านหนึ่งของไฟร์วอลล์ได้

ไฟร์วอลล์ชนิดนี้โดยทั่วไปจะเรียกว่า สกรีนนิ่งเราเตอร์ (Screening Router) เพราะว่าเป็นการนำเราเตอร์ทั่วไปที่มีความสามารถกำหนดแอคเซสรูล (Access Rule) ได้มาดัดแปลงใช้ในการควบคุมทราฟฟิก ซึ่งการกำหนดแอคเซสรูลของทราฟฟิกทำได้โดยพิจารณาจากข้อมูลแต่ละแพ็คเก็ตแต่เนื่องจากเราเตอร์เป็นอุปกรณ์ที่มีพื้นฐานจากการทำงานในอินเตอร์เน็ตโปรโตคอลเลเยอร์ทำหน้าที่จัดเส้นทางของแพ็คเก็ตเป็นหลักโดยพิจารณาจาก IP Address และจะทำการจัดส่งไปทีละแพ็คเก็ต ดังนั้นจึงสามารถควบคุมทราฟฟิกได้ดีในระดับ IP คือดูจาก IP Address ทั้งต้นทางและปลายทางเท่านั้น สำหรับข้อมูลในส่วนของโปรโตคอลในเลเยอร์สูงขึ้นไป เช่น TCP, UDP และ ICMP นั้น เนื่องจากเราเตอร์มีขีดจำกัดในการรับรู้ข้อมูลในเลเยอร์บนถัดขึ้นไป คือทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์ (Transport Control Protocal(TCP) layer) จึงทำให้สามารถควบคุมทราฟฟิกโดยระบุเงื่อนไขของโปรโตคอลในทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์ได้อย่าง

จำกัด คือจะสามารถควบคุมทราฟฟิกได้เฉพาะเมื่อข้อมูลในทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์นั้นจะสามารถบรรจุได้ในแพ็คเก็ตเดียว ถ้ามีการแฟรกเมนต์และต้องเชื่อมโยงกันระหว่างหลายแพ็คเก็ตแล้ว เราเตอร์จะไม่สามารถรับรู้การเชื่อมโยงนั้นได้

ส่วนใหญ่การควบคุมทราฟฟิกโดยใช้เราเตอร์มักจะไม่เรียกว่า ไฟร์วอลล์ เนื่องจากขีดจำกัดของความสามารถในการควบคุมและความคล่องตัวในการบริหารหรือกำหนดกฎต่างๆ สำหรับแอคเซสรูลนั้นค่อนข้างยาก เพราะการเข้าไปตั้งค่าในเราเตอร์จะต้องอาศัยความชำนาญทางเทคนิคพอสมควรและเนื่องจากเราเตอร์ถูกออกแบบให้ทำหน้าที่หลักเป็นเราเตอร์ ส่วนความสามารถในเรื่องแพ็คเก็ตฟิลเตอร์นั้นเป็นเพียงส่วนประกอบ ดังนั้นฟังก์ชั่นที่เกี่ยวข้องกับแอคเซสรูลไม่ว่าจะเป็นการตรวจสอบ การแก้ไข รวมไปถึงการทดสอบผลในการป้องกันก็ทำได้ยาก นอกจากนี้ถ้าทำการประมวลผลแอคเซสรูลที่ซับซ้อนหรือมากเกินไป ประสิทธิภาพในการทำงานเป็นเราเตอร์จะต่ำลงได้

ข้อดีของสกรีนนิ่งเราเตอร์

- ราคาถูกเพราะเป็นคุณสมบัติที่มักมีในของเราเตอร์อยู่แล้วอาศัยเพียงการกำหนดแอคเซสรูลที่เหมาะสมเท่านั้น ถ้ายังไม่มีไฟร์วอลล์อยู่เลยก็สามารถใช้เพื่อช่วยป้องกันเครือข่ายภายในได้ดีพอสมควรในระดับหนึ่ง
- ถ้าเครือข่ายภายในไม่ใหญ่มากและมีการใช้งานอินเตอร์เน็ตอย่างจำกัดก็สามารถใช้ทดแทนไฟร์วอลล์ได้ทันที
- การใช้สกรีนนิ่งเราเตอร์ควบคู่กับไฟร์วอลล์จะเป็นการแบ่งเบาภาระของไฟร์วอลล์ได้มาก ถ้าทำการกำหนดแอคเซสรูลได้อย่างสอดคล้องกันแล้วจะทำให้มีการป้องกันที่เข้มแข็ง
- การป้องกันบางประเภทไม่สามารถป้องกันได้โดยไฟร์วอลล์ จะต้องทำโดยการกำหนดที่ เราเตอร์เท่านั้น

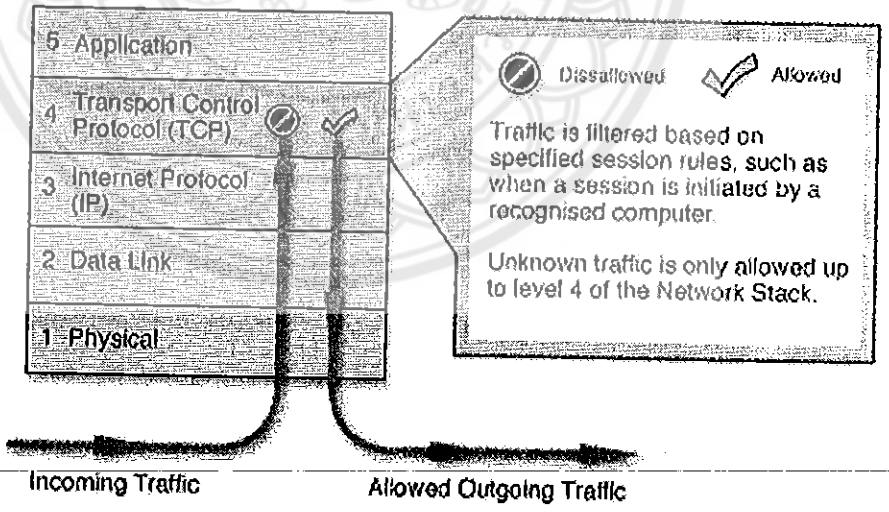
ข้อเสียของสกรีนนิ่งเราเตอร์

- การกำหนดแอคเซสรูลทำได้ยาก ไม่มีระบบยูสเซอร์อินเตอร์เฟส (User Interface) เพื่อช่วยในการทำงาน ส่วนใหญ่จะใช้วิธี Telnet เข้าไปยังเราเตอร์ แล้วป้อนคำสั่งในลักษณะของ Command Line เข้าไปโดยตรงที่เราเตอร์ ทำให้โอกาสที่จะกำหนดผิดพลาดเนื่องจากการป้อนข้อมูลผิดรูปแบบ (Syntax) เป็นไปได้สูง
- คำสั่งในการทำงานจะขึ้นอยู่กับยี่ห้อของเราเตอร์ ไม่มีมาตรฐานของคำสั่ง ถ้าเปลี่ยนยี่ห้อของเราเตอร์ก็จะต้องศึกษารูปแบบของคำสั่งใหม่
- ไม่สามารถกำหนดกฎที่ซับซ้อนได้ เนื่องจากขีดจำกัดของเราเตอร์ที่ทำงานโดยพิจารณาครั้งละแพ็คเก็ตเท่านั้น

- มีความสามารถจำกัด เช่นไม่สามารถบันทึก Log ของแพ็คเก็ตที่ต้องสงสัยไว้ตรวจสอบภายหลังได้
- เราเตอร์มีกำลังในการประมวลผลจำกัด ถ้าเครือข่ายมีขนาดใหญ่และมีการสื่อสารข้อมูลหนาแน่น เราเตอร์จะทำงานหนักอยู่แล้ว เมื่อต้องมาทำการประมวลผลแอคเซสรูลด้วย อาจจะทำให้ประสิทธิภาพในการจัดเส้นทางของแพ็คเก็ตลดต่ำลงไปมาก และการสื่อสารข้อมูลก็จะติดขัดเป็นคอขวดที่เราเตอร์

2.3.2 เซอร์กิตเลเวลไฟร์วอลล์ (Circuit-Level Firewall)

การสื่อสารข้อมูลโดยทั่วไปจะเป็นการสื่อสารแบบต่อเนื่องโต้ตอบกันไปมาระหว่างผู้รับและผู้ส่งอยู่เสมอ โปรโตคอลที่อยู่ในเลเยอร์ที่สูงกว่าอินเตอร์เน็ตโปรโตคอลเลเยอร์ไม่ว่าจะเป็นทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์ อย่างเช่น TCP (Transport Control Protocal), UDP (User Datagram Protocal) หรือในแอพพลิเคชั่นเลเยอร์ เช่น FTP (File Transfer Protocal), HTTP (Hyper Text Transfer Protocal), SMTP (Simple Mail Transfer Protocal) ล้วนแต่จะต้องมีสถานะ (State) ของการสื่อสารเสมอ สถานะนี้จะทำให้ทั้งสองฝั่งสามารถสื่อสารกันได้อย่างต่อเนื่อง คือทราบว่าตอนนี้กำลังอยู่ ณ จุดใดและจะต้องส่งหรือรับข้อมูลใดเป็นลำดับต่อไป ไฟร์วอลล์ประเภทนี้ทำงานในระดับเซสชั่นเลเยอร์ (Session layer) ของ OSI model หรือ ทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์ (Transport Control Protocol (TCP) layer) ของ TCP/IP model ดังรูปที่ 2.2



รูปที่ 2.2 การทำงานของเซอร์กิตเลเวลไฟร์วอลล์

เซอร์กิตเลเวลไฟร์วอลล์เป็นไฟร์วอลล์ที่ทำงานโดยที่สามารถเข้าใจสถานะการสื่อสารทั้งกระบวนการ เพราะว่าการสื่อสารข้อมูลจะสมบูรณ์ได้นั้นจะต้องมีทั้งการส่งและการรับอย่างสอดคล้องสัมพันธ์กัน หมายถึงถ้าไฟร์วอลล์จะสามารถควบคุมการสื่อสารได้จริงจะต้องสามารถเข้าใจกระบวนการของการสื่อสารตั้งแต่ต้นจนจบ โดยทั่วไปจะเรียกไฟร์วอลล์แบบนี้ว่า สเตทฟูล

อินสเปคชั่นไฟร์วอลล์ (Stateful Inspection Firewall) หรือเรียกย่อๆว่าสเตทฟูลไฟร์วอลล์ เป็นไฟร์วอลล์ที่ทำการควบคุมทราฟฟิกโดยใช้หลักการของแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์และการกำหนดแอคเซสรูลเช่นเดียวกับสกรีนนิ่งเราเตอร์ แต่สเตทฟูลไฟร์วอลล์จะมีความสามารถในการวิเคราะห์และรับรู้ความต่อเนื่องของแพ็คเก็ตในโปรโตคอลในระดับสูงขึ้นไปมากกว่า ไม่ว่าจะเป็น TCP, FTP, HTTP หรือแม้กระทั่งโปรโตคอลในระดับแอพพลิเคชั่นเลเยอร์ที่มีลักษณะเฉพาะของแอพพลิเคชั่นนั้นที่จะมีวิธีการกำหนดสถานะของตนเอง

สเตทฟูลไฟร์วอลล์ เป็นเครื่องมือที่ถูกออกแบบมาเพื่อทำหน้าที่ในการควบคุมทราฟฟิกโดยเฉพาะ ไม่ได้เป็นการดัดแปลงการทำงานมาจากเราเตอร์จึงมีความสามารถในการควบคุมทราฟฟิก การกำหนดแอคเซสรูล การบริหาร รวมไปถึงความยืดหยุ่นของการควบคุมทราฟฟิกและประสิทธิภาพสูงกว่าสกรีนนิ่งเราเตอร์เป็นอย่างมาก โดยทั่วไปถ้าพูดถึงไฟร์วอลล์จะหมายถึงสเตทฟูลไฟร์วอลล์นั่นเอง

สเตทฟูลไฟร์วอลล์มีความสามารถในการวิเคราะห์ทราฟฟิกที่ผ่านไปมาในโปรโตคอลที่เลเยอร์สูงขึ้นไปไม่ว่าจะเป็น TCP, UDP, ICMP ได้อย่างสมบูรณ์ต่างจากสกรีนนิ่งเราเตอร์ที่สามารถวิเคราะห์ได้เฉพาะเท่าที่จะมีข้อมูลใน 1 แพ็คเก็ตเท่านั้น เพราะบางครั้งทราฟฟิกที่ผ่านไปมานั้นมีการเชื่อมโยงกันหลายแพ็คเก็ต โดยเฉพาะ TCP ซึ่งจะมีลำดับของการติดต่อสื่อสารที่สัมพันธ์กันในแต่ละแพ็คเก็ต การพิจารณาแพ็คเก็ตใดแพ็คเก็ตหนึ่งโดยไม่พิจารณาแพ็คเก็ตอื่นที่เกี่ยวข้องอาจไม่สามารถควบคุมทราฟฟิกของ TCP ได้ นอกจากนี้ยังรวมไปถึงการที่สเตทฟูลไฟร์วอลล์มีความสามารถในการประกอบรวมแฟรกเมนต์เข้าด้วยกันให้เป็นดาต้าแกรมที่สมบูรณ์ก่อน หลังจากนั้นจึงนำดาต้าแกรมนั้นมาทำการตรวจสอบเปรียบเทียบกับแอคเซสรูล

นอกจากการเชื่อมโยงกันของหลายแพ็คเก็ตสำหรับโปรโตคอล TCP ในทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์แล้ว ในแอพพลิเคชั่นเลเยอร์ก็มีแอพพลิเคชั่นบางชนิดที่จะต้องอาศัยการพิจารณาทราฟฟิกอย่างต่อเนื่องเพื่อที่จะนำมากำหนดเป็นแอคเซสรูลได้ ยกตัวอย่างเช่น การทำงานของ FTP ซึ่งในระหว่างการทำงานของแอพพลิเคชั่นนั้น โฮสต์ที่เป็นเครื่องลูกข่ายจะสามารถกำหนดพอร์ตชั่วคราวขึ้นมาทำหน้าที่เป็นเซิร์ฟเวอร์พอร์ต (Server Port) ใช้สำหรับรับ-ส่งไฟล์ได้ โดยพอร์ตเหล่านี้จะปิดลงเมื่อการรับ-ส่งข้อมูลเสร็จสิ้นสมบูรณ์ ซึ่งในกรณีเช่นนี้ถ้าไม่มีการพิจารณาทราฟฟิกที่มีมาก่อนหน้าแล้ว ไฟร์วอลล์อาจจะถือได้ว่าการเปิดเซิร์ฟเวอร์พอร์ตชั่วคราวของเอฟทีพีไคลเอนท์ (FTP Client) นั้นเป็นการเปิดให้บริการใหม่ขึ้นมาได้ ดังนั้นสเตทฟูลไฟร์วอลล์จึงมีการทำงานที่ใกล้ชิดกับแอพพลิเคชั่นเป็นอย่างมาก จะต้องสามารถเข้าใจคุณสมบัติของการสื่อสารข้อมูลของแต่ละแอพพลิเคชั่นได้ค่อนข้างดี เพราะแอพพลิเคชั่นที่ใช้งานอยู่ในเครือข่ายไม่ได้มีเฉพาะแอพพลิเคชั่นพื้นฐานเท่านั้น มีแอพพลิเคชั่นอื่นๆ อีกมากมาย แต่ถ้าแอพพลิเคชั่นใดมีการใช้งานอย่างแพร่หลายและเป็นที่นิยมของผู้ใช้ โดยส่วนใหญ่ผู้ผลิตจะใส่วิธีการควบคุมทราฟฟิกสำเร็จรูปมาให้อยู่ในไฟร์วอลล์เลย อย่างไรก็ตามถึงแม้ว่าอาจจะมี

แอพพลิเคชั่นใดที่โปรโตคอลไม่ได้เป็นมาตรฐานนั้น สเตทฟูลไฟร์วอลล์ก็เปิดโอกาสให้ผู้ใช้สามารถปรับแต่ง (Customize) ให้สามารถรู้จัก เข้าใจและควบคุมทราฟฟิกของแอพพลิเคชั่นนั้นได้

การกำหนดแอคเซสรูลของสเตทฟูลไฟร์วอลล์นั้น ในเงื่อนไขส่วนที่เป็นเซอร์วิส (Services) ซึ่งโดยทั่วไปจะหมายถึงพอร์ตปลายทางนั้นก็จะหมายถึงการบริการจริง ไม่ใช่หมายถึงเฉพาะพอร์ตปลายทาง ถึงแม้ว่าโดยส่วนใหญ่เกือบจะเป็นสิ่งเดียวกัน เช่น HTTP หมายถึง พอร์ต 80 TCP, POP3 (Post Office Protocol 3) หมายถึง พอร์ต 110 TCP เป็นต้น แต่ในบางกรณีการบริการอาจจะไม่ได้หมายถึงเฉพาะพอร์ตตายตัวเช่น FTP โดยปกติ FTP จะใช้พอร์ตหมายเลข 20 และ 21 เป็นหลัก แต่ก็มีในบางโหมดที่จะใช้พอร์ตหมายเลขอื่นเพิ่มเติมขึ้นมาเป็นเซิร์ฟเวอร์พอร์ตได้ ดังนั้นการกำหนดเซอร์วิสจึงไม่ใช่เป็นเพียงแค่การกำหนดพอร์ตปลายทางเสมอไป การบริการใดๆ อาจจะมีการใช้พอร์ตที่เปลี่ยนแปลงไปได้ ขึ้นอยู่กับแอพพลิเคชั่นที่ทำงาน แต่สเตทฟูลไฟร์วอลล์ก็มีความสามารถมากพอที่จะติดตามไปควบคุมทราฟฟิกและบริการต่างๆ ได้ตามแอคเซสรูล ซึ่งในส่วนนี้จะเห็นความแตกต่างกับสกรีนนิ่งเราเตอร์ได้อย่างชัดเจน เพราะสกรีนนิ่งเราเตอร์นั้นจะควบคุมทราฟฟิกโดยการระบุหมายเลขพอร์ตที่ตายตัวและไม่สามารถยืดหยุ่นไปกับแอพพลิเคชั่นได้ ถ้าจะเปิดพอร์ตก็หมายถึงต้องเปิดอนุญาตให้ทั้งหมด ถ้าจะปิดพอร์ตก็ต้องไม่อนุญาตทั้งหมด โดยไม่คำนึงว่าจะเป็นแอพพลิเคชั่นใด

ข้อดีของสเตทฟูลไฟร์วอลล์

- ใช้งานง่ายเพราะถูกออกแบบมาทำหน้าที่ของไฟร์วอลล์โดยเฉพาะ ตรวจสอบแก้ไขแอคเซสรูลได้ง่าย ทำให้ผู้ใช้ไม่ต้องคอยกังวลถึงคำสั่งและรูปแบบของคำสั่ง ถึงแม้ว่าจะต่างยี่ห้อกันก็สามารถเรียนรู้ใหม่ได้อย่างรวดเร็ว
- ประสิทธิภาพในการทำงานสูง เนื่องจากออกแบบมาทำหน้าที่ไฟร์วอลล์โดยเฉพาะ สามารถรองรับแอคเซสรูลที่ซับซ้อนได้ โดยที่ความสามารถในการทำงานไม่ตกลง
- มีคุณสมบัติเพิ่มเติมให้ใช้ได้มากนอกเหนือจากการควบบคุมทราฟฟิก เช่น สามารถนำไปใช้ร่วมกับระบบตรวจจับการบุกรุกหรือ IDS (Intrusion Detection System) เพื่อป้องกันการโจมตีได้อัตโนมัติ สามารถบันทึกข้อมูลเอาไว้กลับมาดูในภายหลังได้ สามารถใช้งานร่วมกับระบบแอนติไวรัส (Anti-Virus) ได้ เป็นต้น
- การกำหนดแอคเซสรูลทำได้ง่าย เพราะไฟร์วอลล์มีความเข้าใจในโปรโตคอลระดับสูง ดังนั้นผู้ใช้อาจจะไม่จำเป็นต้องมีความเชี่ยวชาญในเรื่องเครือข่ายมาก ก็พอจะใช้งานไฟร์วอลล์ได้ โดยกำหนดกฎบนพื้นฐานของแอพพลิเคชั่นที่ผู้ใช้รู้จักมากกว่าการกำหนดกฎโดยใช้ข้อมูลบนแพ็คเก็ตโดยตรง เช่น แทนที่จะต้องกำหนดแอคเซสรูลให้อนุญาต ICMP Time exceed in Transit ให้ผ่านได้ เพื่อจะใช้คำสั่ง Traceroute (ซึ่งโดยทั่วไปผู้ใช้โดยทั่วไปไม่ทราบว่าโปรแกรมใดใช้โปรโตคอลอะไร แต่จะรู้ว่าตนเองต้องการใช้โปรแกรมหรือแอพพลิเคชั่นอะไรบ้าง) ก็ระบุในไฟร์วอลล์ว่าอนุญาตให้คำสั่ง

Traceroute ทำงานได้ หลังจากนั้นไฟร์วอลล์จึงกำหนดเป็นแอดเซสรูลที่ระบุโปรโตคอลเอง

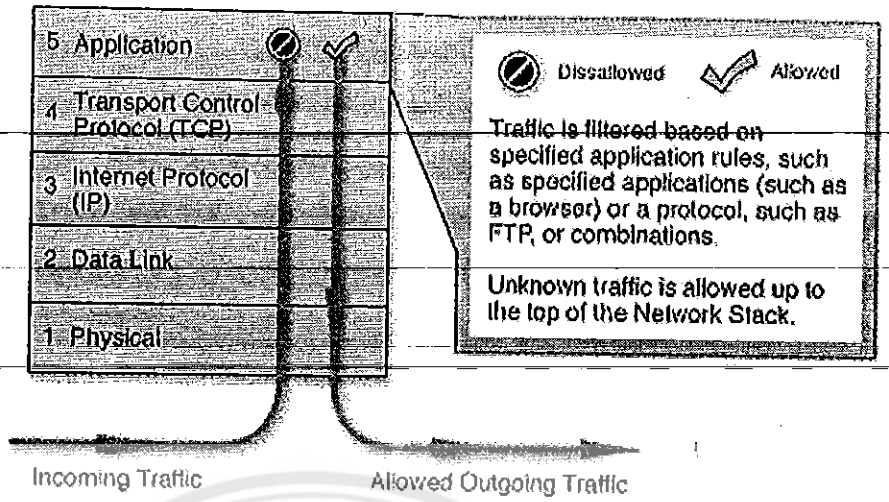
- สามารถเพิ่มเติมบริการอื่นได้ เช่น Virtual Private Network, Tunneling
- สามารถเพิ่มเติมความปลอดภัยโดยระบบการตรวจสอบผู้ใช้ (Authenticate) ได้
- การสื่อสารระหว่างไฟร์วอลล์กับแอดมินคอนโซล (Administration Console : โฮสต์ที่ทำหน้าที่ในการบริการไฟร์วอลล์) จะมีความปลอดภัยสูง มีการตรวจสอบสิทธิ์ของผู้ที่เป็นแอดมินคอนโซล รวมทั้งการสื่อสารระหว่างไฟร์วอลล์กับคอนโซลจะมีการรักษาความปลอดภัยที่เข้มงวด มีการเข้ารหัสเพื่อป้องกันการดักอ่านข้อมูล

ข้อเสียของสเตทฟูลไฟร์วอลล์

- มีราคาแพง ถึงแม้ว่าปัจจุบันจะลดลงไปมากแล้วแต่ก็ยังแพงอยู่
- ในกรณีที่เป็นไฟร์วอลล์แบบซอฟต์แวร์ที่ทำงานอยู่บนระบบปฏิบัติการทั่วไป เช่น Solaris, Window NT, Windows 2000 ต่างก็มีความเสี่ยงที่จะถูกเจาะได้เนื่องจากปัญหาของแต่ละระบบปฏิบัติการ ซึ่งจะสามารถเจาะได้ง่ายกว่าการเจาะเราเตอร์ เพราะรู้รั่วของระบบ ปฏิบัติการมีมากกว่าของเราเตอร์
- ในกรณีที่ไฟร์วอลล์เป็นประเภท Network Appliance คือออกแบบทั้งซอฟแวร์และฮาร์ดแวร์เป็นเครื่องเดียวกันเพื่อทำหน้าที่เป็นไฟร์วอลล์โดยเฉพาะ ผู้ใช้จำเป็นต้องพึ่งพาผู้ผลิตค่อนข้างมาก ถ้ามีปัญหาอาจจะไม่สามารถแก้ไขโดยการใช้อะไหล่ทดแทนจากที่อื่นได้

2.3.3 แอพพลิเคชั่นเลเวลไฟร์วอลล์ (Application Level Firewall (Proxy))

พร็อกซี่เป็นเครื่องมือในการควบคุมทราฟฟิกชนิดหนึ่ง ในลักษณะที่เป็นตัวกลางในการสื่อสารระหว่างเครื่องแม่ข่ายกับเครื่องลูกข่าย โดยทำหน้าที่ป้องกันไม่ให้มีการสื่อสารโดยตรงระหว่างเครื่อง แม่ข่ายกับเครื่องลูกข่าย แต่ยังคงให้เครื่องลูกข่ายสามารถใช้งานแอพพลิเคชั่นบนเครื่องแม่ข่ายได้ตามปกติ และผู้ใช้ซึ่งใช้งานแอพพลิเคชั่นนั้นๆจะไม่ได้รับผลกระทบแต่อย่างใด ไฟร์วอลล์ประเภทนี้ทำงานในระดับแอพพลิเคชั่นเลเยอร์ (Application layer) ของ OSI model หรือแอพพลิเคชั่นเลเยอร์ (Application layer) ของ TCP/IP model ดังรูปที่ 2.3



รูปที่ 2.3 การทำงานของแอพพลิเคชั่นเลเวลไฟร์วอลล์

ไฟร์วอลล์ประเภทนี้ไม่สามารถใช้งานแบบทรานสพาเรนท์พร็อกซี่ (Transparent Proxy) ได้ จึงต้องทำการแก้ไขแอพพลิเคชั่นที่เครื่องลูกข่ายให้ใช้งานพร็อกซี่ ซึ่งการทำงานแบบทรานสพาเรนท์พร็อกซี่เป็นการติดตั้งใช้งานระบบพร็อกซี่ที่ป้องกันไม่ให้มีการสื่อสารโดยตรงระหว่างเครื่องลูกข่ายกับเครื่องแม่ข่าย แต่จะต้องทำการผ่านพร็อกซี่ก่อนทุกครั้งเพื่อทำการตรวจสอบเปรียบเทียบกับเงื่อนไขที่กำหนดไว้แม้ว่าเครื่องลูกข่ายนั้นจะไม่ได้ทำการแก้ไขแอพพลิเคชั่นให้ใช้งานพร็อกซี่ก็ตาม การทำงานแบบทรานสพาเรนท์พร็อกซี่จึงทำให้เกิดความสะดวกในการใช้งานของผู้ใช้ในเครือข่ายที่ไม่ต้องทำการแก้ไขแอพพลิเคชั่นให้ใช้งานพร็อกซี่

โดยปกติทั่วไปแล้วการสื่อสารระหว่างเครื่องแม่ข่ายกับเครื่องลูกข่ายนั้นจะต้องมีการเชื่อมต่อ (Connection) เกิดขึ้นระหว่างเครื่องแม่ข่ายกับเครื่องลูกข่ายอยู่ตลอดเวลาที่สื่อสารกันอยู่ จุดสำคัญอยู่ตรงที่การที่เชื่อมต่อโดยตรงนั้นจะมีความเสี่ยงหลายประการ จึงมีการนำ Proxy เข้ามากั้น

หน้าที่ในการทำงานของพร็อกซี่ คือ เป็นตัวกลางรับข้อมูลจากเครื่องลูกข่ายมาแล้วทำการส่งต่อไปยังเครื่องแม่ข่าย และรับข้อมูลที่ตอบกลับจากเครื่องแม่ข่ายกลับมายังเครื่องลูกข่ายที่ทำการร้องขอ และจะทำหน้าที่นี้อยู่ตลอดเวลาที่เครื่องแม่ข่ายกับเครื่องลูกข่ายติดต่อกัน ซึ่งการที่มีพร็อกซี่มาเป็นตัวกลางระหว่างเครื่องแม่ข่ายกับเครื่องลูกข่ายนั้นทำให้โฮสต์ทั้งคู่ไม่จำเป็นต้องติดต่อกันโดยตรงเพียงแค่ติดต่อกับตัวกลาง คือพร็อกซี่เท่านั้น และการทำงานของแอพพลิเคชั่นทั้งสองฝั่งยังคงทำงานได้เช่นเดิม

ขั้นตอนการนำพร็อกซี่เข้ามาใช้งาน

1. การเริ่มต้นทำงานของแอพพลิเคชั่นโดยทั่วไป เริ่มจากการที่แอพพลิเคชั่นบนเครื่องลูกข่ายขอรับข้อมูลจากเครื่องแม่ข่ายตามโปรโตคอลในแอพพลิเคชั่นเลเยอร์ที่กำหนดไว้ เช่น เว็บบราวเซอร์กับเว็บเซิร์ฟเวอร์จะใช้โปรโตคอล HTTP ในการสื่อสารระหว่างกัน
2. เมื่อเว็บเซิร์ฟเวอร์ได้รับการขอข้อมูลจากบราวเซอร์แล้วก็จะตอบรับกลับไปและเริ่มการติดต่อสื่อสารกัน และทั้งฝั่งเครื่องลูกข่ายและเครื่องแม่ข่ายก็จะทำการติดต่อสื่อสารกันตามที่โปรโตคอล HTTP กำหนดจนจบการสื่อสารเซสชั่น (Session) นั้น อย่างไรก็ตามโปรโตคอล HTTP นั้นจะต้องอาศัย TCP ในการรับส่งข้อมูลระหว่างเครื่องลูกข่ายกับเครื่องแม่ข่าย นั่นหมายถึงเครื่องลูกข่ายจะต้องสามารถติดต่อกับเครื่องแม่ข่ายได้ด้วย TCP เสียก่อน เนื่องจาก TCP เป็นโปรโตคอลในเลเยอร์ที่อยู่ภายใต้ HTTP อีกเลเยอร์หนึ่ง ดังนั้นในสภาวะการทำงานปกติของ HTTP บราวเซอร์จะต้องสามารถติดต่อกับเครื่องแม่ข่ายโดยตรงเสมอ นั่นคือในสภาวะการทำงานปกตินั้นแพ็คเก็ตของ TCP/IP จะต้องสามารถส่งถึงกันระหว่างโฮสต์ทั้งคู่ได้
3. เมื่อนำพร็อกซี่มาใช้งานจะต้องติดตั้งตรงจุดที่คั่นระหว่างเครื่องลูกข่ายกับเครื่องแม่ข่ายเพื่อเป็นตัวกลาง โดยพร็อกซี่จะต้องมี 2 อินเตอร์เฟสโดยอินเตอร์เฟสหนึ่งต่ออยู่กับเครือข่ายของเครื่องลูกข่ายและอีกอินเตอร์เฟสหนึ่งต่ออยู่กับเครื่องแม่ข่าย ซึ่งถ้าพิจารณาที่พร็อกซี่แล้วจะเห็นว่าสามารถติดต่อได้โดยตรงกับทั้งเครื่องลูกข่ายและเครื่องแม่ข่าย แต่สำหรับเครื่องลูกข่ายและเครื่องแม่ข่ายจะติดต่อได้แต่เพียงกับพร็อกซี่เท่านั้น

ในลักษณะที่มีพร็อกซี่มาคั่นกลางระหว่างเครือข่ายทั้งสองนั้น การสื่อสารระหว่างเครื่องลูกข่ายและเครื่องแม่ข่ายด้วยวิธีการเดิมโดยใช้ HTTP เช่นเดิมเหมือนกับมีการสื่อสารกันโดยตรงนั้นย่อมไม่สามารถจะกระทำได้ เพราะการสื่อสารในเลเยอร์ล่างของ TCP/IP นั้นไม่สามารถทำได้สำเร็จ ดังนั้นจึงจำเป็นต้องมีการปรับปรุงแก้ไขโปรโตคอลให้สามารถรองรับการสื่อสารที่มีตัวกลางมาถ่ายทอดข้อมูลได้ โดยให้ในระดับ TCP/IP นั้นกำหนดให้เพียงโฮสต์แต่ละฝั่งสามารถติดต่อกับพร็อกซี่เท่านั้น ส่วนในระดับ HTTP นั้นพร็อกซี่จะทำการส่งต่อระหว่างทั้งสองฝั่งให้ดูประหนึ่งว่าสามารถติดต่อกันได้โดยตรง ซึ่งจุดสำคัญของพร็อกซี่ก็จะอยู่ตรงนี้เอง อาจจะกล่าวโดยสรุปคือพร็อกซี่จะทำให้โฮสต์ไม่สามารถติดต่อกันได้โดยโปรโตคอล TCP/IP แต่จะสามารถติดต่อกันได้ด้วยโปรโตคอลในระดับแอพพลิเคชั่น

แอพพลิเคชั่นที่ใช้งานพร็อกซี่นั้นจะต้องมีการแก้ไขในระดับแอพพลิเคชั่นในบางส่วนเพื่อให้สามารถสื่อสารผ่านพร็อกซี่ได้ เช่น เว็บบราวเซอร์ ถ้าจะทำการสื่อสารกันโดยผ่านพร็อกซี่

นั้นก็จะต้องทำการปรับแต่งเพื่อให้บราวเซอร์ทราบว่าจะให้ติดต่อกับเว็บไซท์โดยผ่านพร็อกซี่หรือจะติดต่อกับเว็บเซิร์ฟเวอร์โดยตรงจะได้ทำการสื่อสารกันได้ถูกต้องและเมื่อบราวเซอร์ต้องการจะติดต่อไปยังเว็บเซิร์ฟเวอร์ใดก็เพียงแต่ส่งคำขอไปยังพร็อกซี่เท่านั้น หลังจากนั้นก็เป็นหน้าที่ของพร็อกซี่ในการไปติดต่อกับเว็บเซิร์ฟเวอร์ตัวจริง แล้วจึงจะนำผลที่ได้จากเว็บเซิร์ฟเวอร์ตอบกลับมายังบราวเซอร์

เมื่อปรับแต่งให้บราวเซอร์ทำการสื่อสารผ่านพร็อกซี่ การทำงานจะมีการเปลี่ยนแปลงไป คือ จากเดิมเมื่อเว็บบราวเซอร์ต้องการติดต่อกับเว็บเซิร์ฟเวอร์ก็จะส่งคำขอในระดับแอพพลิเคชั่น ซึ่งในกรณีนี้คือ HTTP ไปยังเครื่องแม่ข่ายปลายทาง แต่สำหรับในระดับเครือข่ายนั้นแพ็คเก็ตของคำขอดังกล่าวจะมี IP Address ของปลายทางคือพร็อกซี่แค่นั้น ไม่ว่าเว็บบราวเซอร์จะติดต่อไปยังเว็บเซิร์ฟเวอร์ซึ่งตัวอยู่ที่ใด แพ็คเก็ตจริงๆก็จะเดินทางไปแค่พร็อกซี่เท่านั้น ในขณะเดียวกันพร็อกซี่ก็คอยโต้ตอบในระดับของ HTTP กลับไปยังเครื่องลูกข่ายเหมือนกับว่าตนเองเป็นเว็บเซิร์ฟเวอร์ปลายทางจริง

โดยทั่วไปพร็อกซี่ที่พบเห็นว่ามีการนำมาใช้งานมากที่สุดก็คือเว็บพร็อกซี่ แต่จริงๆแล้วยังมีแอพพลิเคชั่นอีกหลายชนิดที่สามารถใช้พร็อกซี่ได้ เช่น เมล์พร็อกซี่, FTP พร็อกซี่ เป็นต้น ซึ่งถ้าแอพพลิเคชั่นที่ใช้งานอยู่ปกติไม่สามารถปรับแต่งให้ใช้พร็อกซี่ได้ เช่น FTP จำเป็นต้องติดตั้งโปรแกรมพร็อกซี่ไคลเอนต์ (Proxy Client) เพื่อใช้งานกับโปรแกรม FTP เพื่อทำหน้าที่ดัดแปลงโปรโตคอลเดิมให้รองรับการสื่อสารผ่านพร็อกซี่ให้ได้

ข้อดีของการใช้งานพร็อกซี่

- สามารถควบคุมการติดต่อสื่อสารระหว่างอินเตอร์เน็ตกับเครือข่ายภายในให้อยู่ในระดับแอพพลิเคชั่นเท่านั้น ตัดขาดการติดต่อโดยตรงในระดับเน็ตเวิร์กเลเยอร์ระหว่างอินเตอร์เน็ตกับเครือข่ายภายในออกจากกันอย่างเด็ดขาด ทำให้ลดความเสี่ยงต่อการถูกคุกคามจากการสแกน การเจาะระบบ การก่อกวนโดยใช้เทคนิคในระดับเน็ตเวิร์กเลเยอร์ที่จะเข้ามายังเครือข่ายภายในได้อย่างเด็ดขาด
- สามารถเพิ่มเติมหน้าที่การทำงานอย่างอื่นเข้าไปในพร็อกซี่ได้ เช่น เว็บพร็อกซี่ นอกจากจะเป็นตัวกลางในการติดต่อแล้ว ยังสามารถควบคุมไม่ให้เว็บบราวเซอร์ติดต่อกับเว็บไซท์ที่ไม่ต้องการได้อีกด้วย โดยการกำหนดรายชื่อเว็บไซท์เหล่านั้นไว้ในพร็อกซี่
- สามารถทำการแคช (Cache) ข้อมูลเก็บไว้ในตัวพร็อกซี่ สำหรับข้อมูลใดที่มีการเรียกใช้บ่อยๆก็ไม่จำเป็นต้องไปอ่านจากเครื่องแม่ข่ายใหม่ทุกครั้ง แต่ส่วนนี้จะใช้งานกับข้อมูลที่เป็นสแตติค (Static) เท่านั้น ข้อมูลที่มีการเปลี่ยนแปลงตลอดเวลาเป็นไดนามิค (Dynamic) อาจจะไม่สามารถแคชไว้ได้
- ทำให้ผู้ใช้มีการใช้แบนด์วิดธ์ร่วมกันอย่างมีประสิทธิภาพ โดยเฉพาะเมื่อใช้ร่วมกับการแคชที่มีอยู่ในพร็อกซี่ ทำให้ช่วยประหยัดการใช้งานแบนด์วิดธ์ไปได้มาก

- สามารถเพิ่มเติมส่วนการตรวจสอบผู้ใช้ เข้าไปเป็นหน้าที่หนึ่งของพร็อกซี่ได้ โดยการอนุญาตให้สามารถใช้งานพร็อกซี่นั้นจะขึ้นอยู่กับสิทธิ์การใช้งานที่ผู้ใช้มีอยู่ ทำให้สามารถควบคุมการใช้งานได้ใกล้ชิดมากขึ้นกว่าการควบคุมโดยพิจารณาจาก IP Address ของโฮสต์เพียงอย่างเดียว
- สามารถทำการกลั่นกรองเนื้อหาของข้อมูลได้ (Content Filtering) ทำให้สามารถนำมาเป็นเงื่อนไขในการอนุญาตให้ข้อมูลเหล่านั้นผ่านเข้าออกได้ เช่น เว็บพร็อกซี่สามารถตรวจสอบเนื้อหาของเว็บไซท์ที่ผู้ใช้เข้าไปดู ถ้าปรากฏว่ามีข้อความที่ไม่เหมาะสม พร็อกซี่ก็จะสามารถดรอ็ปเซสชั่นที่ผู้ใช้ขอมาได้ หรือกรณีที่เป็นอีเมล์พร็อกซี่ก็จะสามารถตรวจสอบเนื้อหาในอีเมล์ได้ว่ามีข้อความที่ไม่เหมาะสมหรือไม่และอาจจะครอบคลุมถึงการตรวจสอบหาไวรัสที่แนบมากับจดหมายได้อีกด้วย

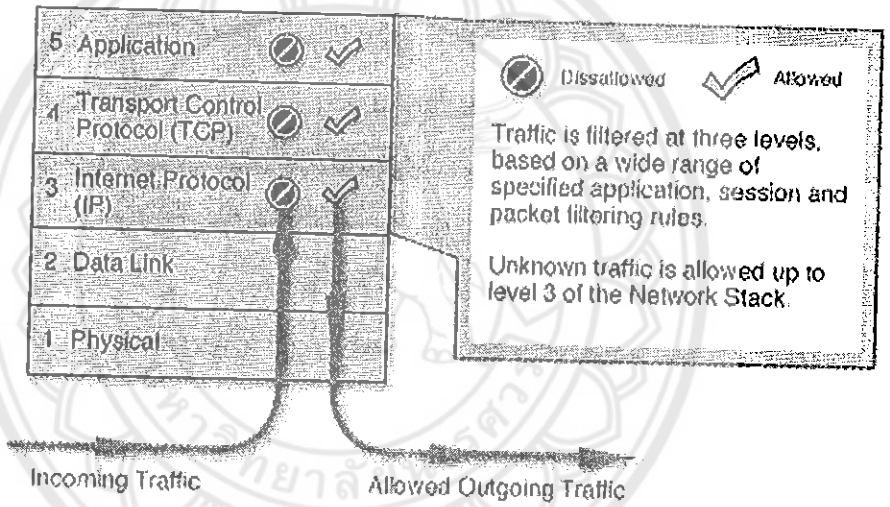
ข้อเสียของการใช้งานพร็อกซี่

- ขึ้นอยู่กับแอพพลิเคชั่น ถ้าแอพพลิเคชั่นไม่รองรับการสื่อสารโดยผ่านพร็อกซี่ก็ไม่สามารถใช้งานได้
- ไม่สามารถใช้งานกับแอพพลิเคชั่นที่ต้องการการสื่อสารโดยตรงแบบ end-to-end ซึ่งแพ็คเก็ตจะต้องมาจากโฮสต์ปลายทางทั้งคู่เท่านั้น ผ่านตัวกลางไม่ได้
- เสี่ยงต่อการละเมิดความเป็นส่วนบุคคล (Privacy) เนื่องจากข้อมูลทั้งหมดที่สื่อสารจะต้อง ผ่านพร็อกซี่ก่อนเสมอ และพร็อกซี่ก็มีความสามารถที่จะเก็บข้อมูลเหล่านั้นไว้ตรวจสอบได้ ถ้ามีผู้นำข้อมูลเหล่านั้นไปวิเคราะห์จะสามารถทราบการใช้งานหรืออาจจะทราบข้อมูลทั้งหมดของผู้ใช้ได้
- เนื่องจากลักษณะของแต่ละแอพพลิเคชั่นนั้นจะแตกต่างกันออกไป ดังนั้นพร็อกซี่ของแต่ละแอพพลิเคชั่นจึงทำหน้าที่เฉพาะแอพพลิเคชั่นนั้นๆ ไม่สามารถใช้ร่วมกันได้ ถ้าโฮสต์ที่อยู่หลังพร็อกซี่มีการใช้งานหลายแอพพลิเคชั่นก็จะต้องมีพร็อกซี่จำนวนมากเปิดให้บริการตามจำนวนแอพพลิเคชั่นนั้นๆ
- ความสามารถในการประมวลผลของโฮสต์ที่ทำหน้าที่พร็อกซี่อาจจะเป็นคอขวดของระบบได้ เพราะการสื่อสารทั้งหมดของเครื่องลูกข่ายและเครื่องแม่ข่ายจะถูกรวมศูนย์ที่พร็อกซี่ก่อนเสมอ แทนที่จะกระจายไปยังเครื่องลูกข่ายและเครื่องแม่ข่าย ปัญหาลักษณะนี้ จะสามารถพบได้ชัดเจนเมื่อมีเครื่องลูกข่ายจำนวนมาก
- เนื่องจากพร็อกซี่เป็นแอพพลิเคชั่นชนิดหนึ่งเช่นกัน การติดต่อกับในเครือข่ายจะอาศัยระบบปฏิบัติการเป็นหลัก จึงมีความสามารถในการป้องกันตัวเองต่ำกว่าไฟร์วอลล์ทั่วไป ตัวพร็อกซี่เองจึงมีความเสี่ยงต่อการถูกโจมตีได้มากและเปราะบางต่อการ DoS ด้วยเทคนิคในระดับเครือข่าย ซึ่งอาจจะส่งผลให้พร็อกซี่อาจจะหยุดทำงานลงได้โดยง่าย โดยเฉพาะเมื่อพร็อกซี่นั้นเป็นโฮสต์ที่ต่อโดยตรงกับอินเตอร์เน็ต จึงเป็นเสมือนด่านหน้า

ของเครือข่ายที่จะต้องถูกสแกน ถูกเจาะอย่างแน่นอน แต่ระดับความต้านทานของพร็อกซี่นั้นต่ำกว่าไฟร์วอลล์โดยทั่วไป จึงมีแนวโน้มว่าถ้าใช้พร็อกซี่โดยปราศจากไฟร์วอลล์ร่วมด้วยโอกาสที่พร็อกซี่จะโดนเจาะได้นั้นมีอยู่สูงมาก

2.3.4 สเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ (Stateful Multilayer Inspection Firewall)

สเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์เป็นการรวมการทำงานของไฟร์วอลล์ทั้ง 3 ประเภท คือ แพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์, เซอร์กิตเลเวลไฟร์วอลล์ และแอพพลิเคชั่นเลเวลไฟร์วอลล์เข้าด้วยกัน ดังรูปที่ 2.4 ทำงานโดยกรองแพ็คเก็ตที่อินเตอร์เน็ตโปรโตคอลเลเยอร์ ตรวจสอบเซสชั่นที่มีการร้องขอว่าเหมาะสมถูกต้องหรือไม่ที่ทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์และตรวจสอบองค์ประกอบของแพ็คเก็ตที่แอพพลิเคชั่นเลเยอร์



รูปที่ 2.4 การทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์

ไฟร์วอลล์ประเภทนี้อนุญาตให้มีการเชื่อมต่อโดยตรงระหว่างเครื่องลูกข่ายกับเครื่องแม่ข่ายโดยการติดต่อกันถ้าใช้โปรโตคอลในระดับอินเตอร์เน็ตโปรโตคอลเลเยอร์ หรือที่ทรานสปอร์ตคอนโทรลโปรโตคอลเลเยอร์ไฟร์วอลล์ประเภทนี้จะมีลักษณะการทำงานเป็นแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์หรือสเตทฟูลอินสเปคชั่นไฟร์วอลล์ แต่ถ้าการติดต่อกันใช้โปรโตคอลในระดับแอพพลิเคชั่นเลเยอร์ไฟร์วอลล์ประเภทนี้จะมีลักษณะการทำงานเป็นพร็อกซี่

ไฟร์วอลล์ประเภทนี้มีความปลอดภัยสูง มีประสิทธิภาพดี และสามารถทำงานแบบทรานสพาเรนท์พร็อกซี่ได้ จึงทำให้เกิดความสะดวกในการใช้งานของผู้ใช้ในเครือข่ายที่ไม่ต้องทำการแก้ไขแอพพลิเคชั่นให้ใช้งานพร็อกซี่ ซึ่งไฟร์วอลล์ประเภท Application level gateway ไม่สามารถทำได้

ข้อดีของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์

- สามารถเชื่อมต่อโดยตรงระหว่างเครื่องลูกข่ายกับเครื่องแม่ข่ายได้
- มีความปลอดภัยสูง เนื่องจากมีการตรวจสอบแพ็คเก็ตทุกเลเยอร์
- สามารถทำงานแบบทรานสพาเรนท์พร็อกซี่ได้

ข้อเสียของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์

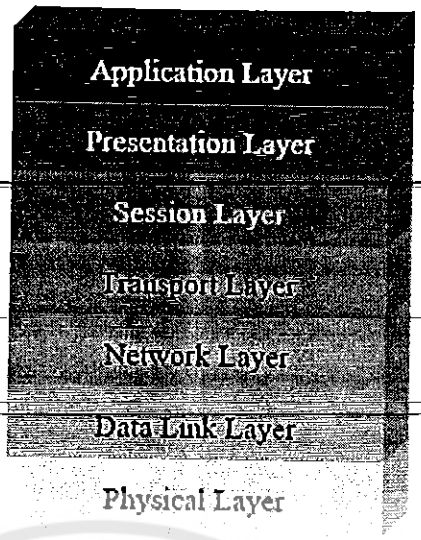
- การเชื่อมต่อโดยตรงระหว่างเครื่องลูกข่ายกับเครื่องแม่ข่าย ทำให้เกิดความเสี่ยง เนื่องจาก แฮคเกอร์สามารถเชื่อมต่อโดยตรงเข้าไปยังเครื่องแม่ข่ายได้
- มีราคาแพง
- เนื่องจากไฟร์วอลล์ประเภทนี้มีความซับซ้อน ถ้าผู้ดูแลไฟร์วอลล์ไม่มีความรู้ความเข้าใจเพียงพอจะทำให้ไฟร์วอลล์ประเภทนี้มีความปลอดภัยน้อยกว่าไฟร์วอลล์ประเภทอื่น

2.4 โอเอสไอโมเดล (OSI Model)

การศึกษาหลักการทำงานของโปรโตคอลบนระบบเครือข่าย เริ่มต้นด้วยการมองการทำงานที่แบ่งออกเป็นชั้นหรือที่เรียกว่าเลเยอร์ โดยที่แต่ละชั้นมีหน้าที่การทำงานที่ชัดเจน และไม่เกี่ยวข้องกัน แต่ละชั้นจะรู้เพียงวิธีการส่งข้อมูลไปยังชั้นอื่น แต่จะไม่รู้การทำงานภายในของชั้นนั้นๆ โปรโตคอลจะมีการแบ่งการทำงานออกเป็นจำนวนชั้นไม่เท่ากัน ทำให้ยากที่จะระบุว่าโปรโตคอลบนระบบเครือข่ายโดยรวมแล้วมีการทำงานกี่ชั้น แต่มีมาตรฐานหนึ่งที่เป็นที่ยอมรับกันโดยทั่วไป คือแบบจำลองสำหรับอ้างอิงแบบโอเอสไอ (OSI Reference Model)

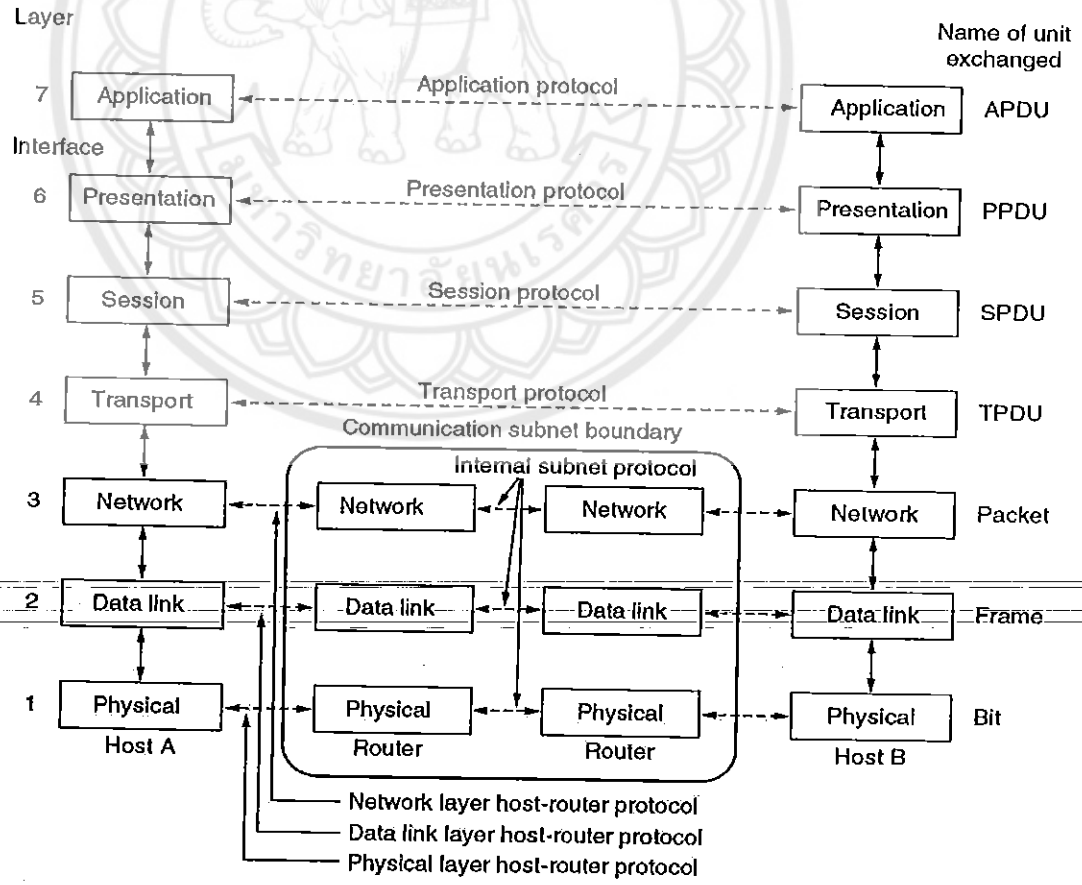
แบบจำลองสำหรับอ้างอิงแบบโอเอสไอ

การเชื่อมต่อคอมพิวเตอร์หลายๆ เครื่องเข้าด้วยกัน จนเกิดเป็นระบบเครือข่ายคอมพิวเตอร์ ซึ่งสามารถรับส่งข้อมูลระหว่างคอมพิวเตอร์เครื่องหนึ่งไปสู่คอมพิวเตอร์อีกเครื่องหนึ่งที่ต่างระบบกันได้ จำเป็นต้องมีมาตรฐานในการสื่อสาร หน่วยงานกำหนดมาตรฐานสากล (ISO : International Standards Organization) จึงกำหนดโครงสร้างทั้งหมดที่จำเป็นต้องใช้ในการรับส่งข้อมูลขึ้นซึ่งเป็นสถาปัตยกรรมแบบระบบเปิด (Open System) เรียกว่า Open Systems Interconnection หรือเรียกสั้นๆว่า OSI Model



รูปที่ 2.5 เลเยอร์ของ OSI Model

OSI Model กำหนดมาตรฐานการสื่อสารข้อมูลจากระบบคอมพิวเตอร์หนึ่งไปยังอีกระบบหนึ่งโดยแบ่งออกเป็น 7 ชั้น ซึ่งคอมพิวเตอร์ทั้ง 2 ฝั่งจะมีชั้นการสื่อสาร 7 ชั้นเหมือนกัน



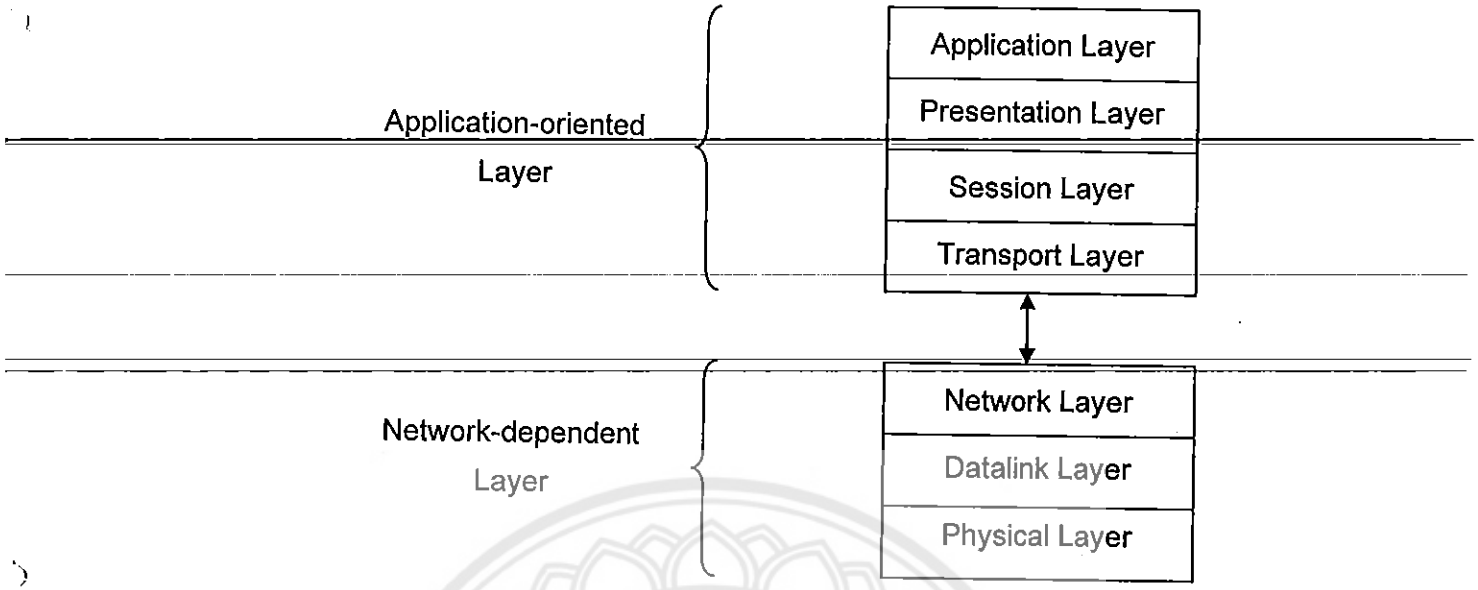
รูปที่ 2.6 การรับส่งข้อมูลของ OSI Model ทั้ง 7 ชั้น

ในแต่ละชั้นจะเหมือนเชื่อมต่ออยู่กับชั้นที่เทียบเท่ากันของคอมพิวเตอร์อีกด้านหนึ่ง เช่น ทรานสปอร์ตเลเยอร์จะติดต่อสื่อสารกับทรานสปอร์ตเลเยอร์ และแอพพลิเคชั่นเลเยอร์จะติดต่อกับแอพพลิเคชั่นของคอมพิวเตอร์อีกฝั่งหนึ่งเท่านั้น แต่จริงๆแล้วมีสายส่งข้อมูลที่เชื่อมต่ออยู่ที่ฟิสิคัลเลเยอร์หรือชั้นล่างสุดเพียงชั้นเดียว

ในการติดต่อรับส่งข้อมูล ผู้ใช้จะทำการรับส่งข้อมูลผ่านทางชั้นที่ 7 คือ แอพพลิเคชั่นเลเยอร์ซึ่งอยู่ด้านบนสุดของ OSI Model เท่านั้น แล้วส่งข้อมูลให้ต่อกับชั้นที่ 6 เรื่อยไปจนถึงชั้นล่างสุดแล้วจึงส่งต่อให้กับชั้นล่างสุดของคอมพิวเตอร์ผู้รับไล่ขึ้นไปจนถึงชั้นที่ 7 ตามลำดับ เนื่องจากการติดต่อรับส่งข้อมูลจะกระโดดข้ามไปส่งให้ชั้นอื่นที่ไม่อยู่ติดกันไม่ได้ แต่ละชั้นที่ทำหน้าที่รับส่งข้อมูลจะติดต่อรับส่งข้อมูลกับชั้นที่ติดอยู่กับตัวเองเท่านั้น แอพพลิเคชั่นเลเยอร์จะส่งให้เซสชั่นเลเยอร์โดยไม่ผ่านพรีเซนเทชั่นเลเยอร์ไม่ได้ ประโยชน์ในการแบ่งเป็นชั้น คือการกำหนดการติดต่อระหว่างชั้นทำได้โดยไม่ต้องคำนึงถึงการเปลี่ยนแปลงในชั้นอื่นๆ

ในทางทฤษฎี แต่ละชั้นของการรับส่งข้อมูลจะมีหน้าที่การทำงานที่แน่นอนและแยกออกจากกัน สามารถที่จะนำแต่ละชั้นของแต่ละบริษัทมาเชื่อมต่อกันได้อย่างไม่มีขีดจำกัด แต่ในทางปฏิบัตินั้น OSI Model จะแบ่งออกเป็น 2 กลุ่มใหญ่ๆ คือกลุ่มแรก ได้แก่ 4 ชั้นด้านบน คือ แอพพลิเคชั่นเลเยอร์, พรีเซนเทชั่นเลเยอร์, เซสชั่นเลเยอร์ และทรานสปอร์ตเลเยอร์ทำหน้าที่เชื่อมต่อต่อกับส่งข้อมูลระหว่างผู้ใช้กับแอพพลิเคชั่นให้รับส่งข้อมูลกับฮาร์ดแวร์ที่อยู่ชั้นล่างได้อย่างถูกต้อง เรียกว่า แอพพลิเคชั่น โอเรียนเต็ดเลเยอร์ (Application-oriented Layer) โดย 4 ชั้นบนนี้ส่วนใหญ่จะเป็นซอฟแวร์ของบริษัทใดบริษัทหนึ่งรวมอยู่เป็นซอฟแวร์เดียว จะแยกออกจากกันเป็นชั้นๆ เพื่อใช้ซอฟแวร์ของบริษัทอื่นทำได้ยากหรือในบางกรณีก็อาจทำไม่ได้เลย

กลุ่มที่สองจะเป็นชั้นล่าง ได้แก่ เน็ตเวิร์กเลเยอร์, ดาต้าลิงค์เลเยอร์ และฟิสิคัลเลเยอร์ ทำหน้าที่เกี่ยวกับการรับส่งข้อมูลผ่านสายส่ง ควบคุมการรับส่งข้อมูล และตรวจสอบข้อผิดพลาด รวมทั้งเลือกเส้นทางที่ใช้ในการรับส่งข้อมูล ซึ่งจะเกี่ยวกับฮาร์ดแวร์เป็นหลัก เรียกว่า เน็ตเวิร์กดีเพนเดนท์เลเยอร์ (Network-dependent Layer) ซึ่งในส่วนนี้เกี่ยวข้องกับฮาร์ดแวร์และโปรแกรมควบคุมฮาร์ดแวร์เป็นหลัก ทำให้สามารถแยกแต่ละชั้นออกจากกันได้ง่ายและใช้ผลิตภัณฑ์ของต่างบริษัทกันในแต่ละชั้นได้



รูปที่ 2.7 การแบ่งเลเยอร์ระดับบนและเลเยอร์ระดับล่าง

หน้าที่การทำงานของแต่ละชั้น

ชั้นที่ 1 ฟิสิคัลเลเยอร์ (Physical Layer)

ทำการรับส่งข้อมูลในรูปแบบบิต (bit) โดยไม่พิจารณาเรื่องความหมายข้อมูล การรับส่งจะส่งข้อมูล "0" หรือ "1" ผ่านสายส่งข้อมูลจริง โดยสามารถกำหนดคุณสมบัติทางกายภายของฮาร์ดแวร์ที่ใช้เชื่อมต่อระหว่างคอมพิวเตอร์ทั้ง 2 ระบบได้ว่าจะใช้สายส่งข้อมูลแบบไหน ข้อต่อหรือปลั๊กที่ใช้ในการรับส่งข้อมูลใช้ความต่างศักย์ไฟฟ้าเท่าใด ความเร็วในการส่งข้อมูลเป็นเท่าใด สัญญาณที่ใช้รับส่งข้อมูลในสายมีรูปร่างอย่างไร ถ้าการรับส่งข้อมูลมีปัญหาเนื่องจากฮาร์ดแวร์ เช่น สายสัญญาณที่ใช้รับส่งข้อมูลขาดหรืออุปกรณ์เสียหาย ก็จะเป็นหน้าที่ของชั้นนี้ที่จะตรวจสอบและแจ้งข้อผิดพลาดนั้นให้ชั้นอื่นๆที่อยู่เหนือขึ้นไปทราบ

ชั้นที่ 2 ดาต้าลิงค์เลเยอร์ (Data Link Layer)

ทำหน้าที่จัดการและรับส่งข้อมูลในระดับฮาร์ดแวร์ โดยแปลคำสั่งที่ได้รับจากเน็ตเวิร์กเลเยอร์ให้เป็นคำสั่งควบคุมฮาร์ดแวร์ และตรวจสอบแก้ไขข้อผิดพลาดในการรับส่งข้อมูล

ชั้นที่ 3 เน็ตเวิร์กเลเยอร์ (Network Layer)

ทำหน้าที่เลือกหรือกำหนดเส้นทางที่จะใช้ในการรับส่งข้อมูลระหว่างเครือข่าย และส่งผ่านข้อมูลที่ได้รับไปสู่ปลายทาง ในชั้นนี้จะมองเห็นข้อมูลทั้งหมดเป็นแพ็คเก็ตหรือเฟรม ซึ่งเป็นการรวมคำสั่งและไดอะล็อก (dialogue) ต่างๆ ไว้ภายใน มีเพียงที่อยู่ของผู้รับ ผู้ส่ง ลำดับการรับส่งและส่วนของข้อมูลเท่านั้นที่ในชั้นนี้จะมองเห็น ดังนั้นตัวเนื้อหาของข้อมูลจะไม่มีผลใดๆในการรับส่งข้อมูลไม่ว่าจะเป็นข้อมูลในระดับสูงอย่างวีดีโอ ภาพ เสียง หรือข้อมูลอื่นใดก็ตาม นอกจากนี้

ยังทำการ Call Setup หรือเรียกติดต่อคอมพิวเตอร์ปลายทางก่อนการรับส่งข้อมูล และทำการ Call Clearing หรือยกเลิกการติดต่อเมื่อการรับส่งข้อมูลจบลง ในกรณีที่การรับส่งข้อมูลนั้นต้องมีการติดต่อกันก่อน (Hand shaking คือ การที่ผู้ส่งร้องขอการติดต่อไปสู่ผู้รับว่าจะขอทำการติดต่อรับส่งข้อมูลได้หรือไม่)

ชั้นที่ 4 ทรานสปอร์ตเลเยอร์ (Transport Layer)

เป็นรอยต่อระหว่างการรับส่งข้อมูลของซอฟต์แวร์กับฮาร์ดแวร์ นำข้อมูลในระดับสูงมาแปลงให้รับส่งได้ในระดับฮาร์ดแวร์ เช่น แปลงค่าหรือชื่อของคอมพิวเตอร์ในเครือข่าย (Mac address) ให้เป็นหมายเลขเครือข่าย (IP Address) พร้อมทั้งเป็นชั้นที่ควบคุมการรับส่งข้อมูลจากปลายด้านส่งถึงปลายด้านรับตลอดเส้นทางตามจังหวะที่กำหนดไว้ในเซสชั่นเลเยอร์ เช่น ไม่ส่งข้อมูลเร็วเกินไปจนฝั่งผู้รับไม่สามารถรับข้อมูลได้ทัน เป็นการควบคุมคุณภาพของการรับส่งข้อมูลให้มีมาตรฐานในระดับที่ตกลงกันของทั้งสองฝ่าย ทำหน้าที่ตัดข้อมูลออกเป็นส่วนย่อยๆ ให้เหมาะกับลักษณะการทำงานของฮาร์ดแวร์ที่ใช้ในเครือข่าย เช่น ถ้าข้อมูลที่ได้รับจากเซสชั่นเลเยอร์มีความยาวเกินกว่าที่เครือข่ายจะส่งได้ ชั้นนี้จะทำการตัดข้อมูลออกเป็นส่วนย่อยๆ แล้วส่งไปให้ผู้รับข้อมูลที่ได้รับปลายทางก็จะถูกนำมาต่อกันที่ชั้นที่ชั้นนี้ของด้านผู้รับ แล้วจึงส่งต่อให้เซสชั่นเลเยอร์นำไปใช้ต่อไป

ชั้นที่ 5 เซสชั่นเลเยอร์ (Session Layer)

ทำหน้าที่ควบคุมจังหวะในการรับส่งข้อมูลของคอมพิวเตอร์ทั้งสองด้านที่รับส่งแลกเปลี่ยนข้อมูลกันให้มีความสอดคล้อง (Synchronization) และกำหนดวิธีที่ใช้รับส่งข้อมูล เช่น อาจจะเป็นในลักษณะสลับกันส่ง คือฝ่ายหนึ่งรับอีกฝ่ายหนึ่งส่ง (Half Duplex) หรือรับส่งข้อมูลพร้อมกันทั้งสองฝ่าย (Full Duplex) โดยจะมองข้อมูลเหมือนเป็นประโยคที่สนทนาโต้ตอบกัน เช่น เมื่อผู้รับได้รับข้อมูลส่วนแรกจากผู้ส่งก็จะโต้ตอบกลับไปให้ผู้ส่งรู้ว่าได้รับข้อมูลส่วนแรกเรียบร้อยแล้วและพร้อมที่จะรับข้อมูลส่วนที่สองต่อไป

ชั้นที่ 6 พรีเซนเทชั่นเลเยอร์ (Presentation Layer)

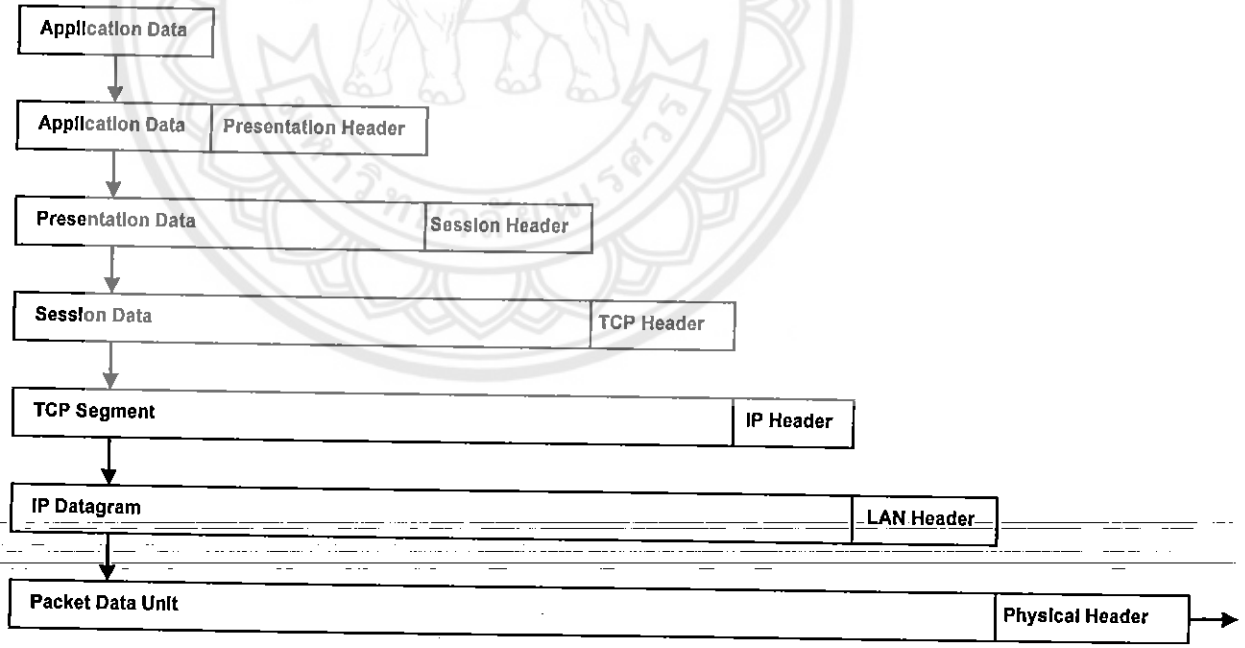
เป็นชั้นที่ทำหน้าที่ตกลงกับคอมพิวเตอร์อีกฝั่งหนึ่งว่ามีขั้นตอนและข้อบังคับอย่างไรในการรับส่งข้อมูล เนื่องจากรูปแบบของข้อมูลจะเป็นคำสั่งที่มีกฎ (Syntax) บังคับอย่างแน่นอน เช่น ในการคัดลอกไฟล์จะต้องสร้างไฟล์ใหม่ขึ้นมาจากนั้นจึงเปิดไฟล์ แล้วจึงรับข้อมูลจากปลายทางมาเก็บลงในไฟล์ที่สร้างใหม่นี้ นอกจากนี้ยังทำหน้าที่แปลความหมายของคำสั่งที่ได้รับจากแอพพลิเคชั่นเลเยอร์ให้เป็นคำสั่งระดับปฏิบัติการแล้วส่งให้เซสชั่นเลเยอร์ต่อไป

ชั้นที่ 7 แอพพลิเคชั่นเลเยอร์ (Application Layer)

เป็นชั้นสูงสุดที่รับคำสั่งต่างๆ จากผู้ใช้นำมาแปลความหมายและทำงานตามคำสั่งที่ได้รับในระดับโปรแกรมประยุกต์ เช่น แปลความหมายของการกดปุ่มเมาส์ให้เป็นคำสั่งในการคัดลอก

ไฟล์หรือดึงข้อมูลมาแสดงผลบนจอภาพ ซึ่งการแปลคำสั่งจะต้องแปลออกมาให้ถูกกฎที่ใช้ในระบบคอมพิวเตอร์นั้นๆ ตัวอย่างเช่น ถ้ามีการคัดลอกไฟล์ ชื่อไฟล์จะต้องยาวไม่เกินจำนวนที่ระบบปฏิบัติการใช้อยู่และต้องประกอบด้วยตัวอักษรตามที่กำหนด รวมไปถึงฟังก์ชันที่ใช้ในการรับส่งข้อมูลระหว่างแอพพลิเคชั่นเลเยอร์กับเซสชั่นเลเยอร์ด้วย

โดยในการรับส่งข้อมูลในโอเอสไอโมเดลนี้ ข้อมูลจากชั้นบนสุดคือแอพพลิเคชั่นเลเยอร์จะถูกส่งลงไปในชั้นถัดไปจนกระทั่งถึงฟิสิคัลเลเยอร์ โดยข้อมูลเดิมจะถูกรวมกับข้อมูลที่ใช้ควบคุมของแต่ละชั้นซ้อนๆ กันเป็นลำดับ เช่น ข้อมูลจากแอพพลิเคชั่นเลเยอร์ คือแอพพลิเคชั่นดาต้า (Application Data) เมื่อถูกส่งลงไปยังชั้นถัดไปก็จะถูกรวมกับแอพพลิเคชั่นเฮดเดอร์ (Application Header) และทั้งแอพพลิเคชั่นเฮดเดอร์และแอพพลิเคชั่นดาต้าจะถูกนำรวมกันเป็นข้อมูลของพรีเซนเทชั่นเลเยอร์ ซึ่งก็จะถูกผนึกด้วยพรีเซนเทชั่นเลเยอร์อีกครั้งก่อนที่จะส่งต่อเป็นข้อมูลให้กับชั้นถัดไป แต่ละชั้นจะผนึกเฮดเดอร์ของตัวเองก่อนส่งต่อชั้นถัดไป จนกระทั่งถึงชั้นล่างสุดซึ่งเป็นฟิสิคัลเลเยอร์ ซึ่งจะทำการส่งข้อมูลต่อไปให้ถึงปลายทางและข้อมูลที่ปลายทางได้รับจะถูกนำมาแยกเฮดเดอร์ที่เพิ่มเข้ามานี้ออกทีละชั้นจนกระทั่งถึงชั้นบนสุดก็จะได้แอพพลิเคชั่นดาต้าที่ผู้ส่งส่งให้แก่ผู้รับ



รูปที่ 2.8 การส่งผ่านข้อมูลใน OSI model

2.5 โปรโตคอลทีซีพี/ไอพี (TCP/IP)

2.5.1 ความเป็นมาของโปรโตคอลทีซีพี/ไอพี

TCP/IP เป็นมาตรฐานของการรับส่งข้อมูลระหว่างคอมพิวเตอร์ 2 ระบบที่มีขึ้นเมื่อกระทรวงกลาโหมสหรัฐฯ หรือ Department of Defense (DOD) ทำโครงการทดลองในปี ค.ศ. 1969 เชื่อมโยงคอมพิวเตอร์ทางทหารของแต่ละหน่วย ซึ่งเป็นคอมพิวเตอร์ต่างชนิดกันให้สามารถติดต่อรับส่งข้อมูลระหว่างกัน (File Transfer) และสามารถใช้บริการอื่นๆ เช่น การใช้งานจากระยะไกล (Remote Login) รวมถึงการรับส่งจดหมายอิเล็กทรอนิกส์ (E-mail) ด้วย จุดประสงค์ของโครงการนี้คือสร้างระบบเครือข่ายคอมพิวเตอร์ให้สามารถรับส่งข้อมูลกันได้ แม้ว่าสายส่งข้อมูลบางส่วนหรือคอมพิวเตอร์บางเครื่องในเครือข่ายจะถูกทำลายเสียหายไปก็ตาม ซึ่งเป็นคุณสมบัติที่สำคัญเมื่อใช้งานในยามเกิดสงคราม

ในขณะนั้นกองทัพบกเลือกใช้คอมพิวเตอร์และระบบเครือข่ายของ Digital Equipment Corporation (DEC) กองทัพเรือเลือกใช้คอมพิวเตอร์ของ Unisys ส่วนกองทัพอากาศเลือกใช้คอมพิวเตอร์ของ IBM เมื่อจะทำการรบกระทรวงกลาโหมสหรัฐฯ พบว่าคอมพิวเตอร์ของทั้ง 3 กองทัพได้รวมเข้าด้วยกันเป็นระบบเครือข่าย โดยมีคุณสมบัติพิเศษแตกต่างจากระบบเครือข่ายที่ใช้งานทั่วๆไปคือ การรับส่งข้อมูลจะแบ่งข้อมูลออกเป็นส่วนย่อยๆ เรียกว่า แพ็คเก็ตข้อมูลแต่ละส่วนนี้จะถูกส่งไปให้คอมพิวเตอร์ผู้รับที่ปลายทางผ่านสายส่งข้อมูล โดยแต่ละส่วนอาจใช้เส้นทางสำหรับส่งข้อมูลคนละทางก็ได้ คอมพิวเตอร์ปลายทางจะนำข้อมูลที่ได้รับมาต่อรวมกันตามลำดับจนครบ ถ้าเส้นทางที่ส่งข้อมูลเสียหายหรือเครื่องคอมพิวเตอร์บางส่วนในเครือข่ายเสียหายข้อมูลก็จะถูกส่งไปใหม่โดยใช้เส้นทางอื่นแทนโดยอัตโนมัติ โครงการนี้มีชื่อว่า Advanced Research Projects Agency Network หรือที่รู้จักกันดีว่า ARPANET ซึ่งประสบความสำเร็จอย่างสูงจนใช้งานกันอย่างจริงจังในปี ค.ศ. 1975

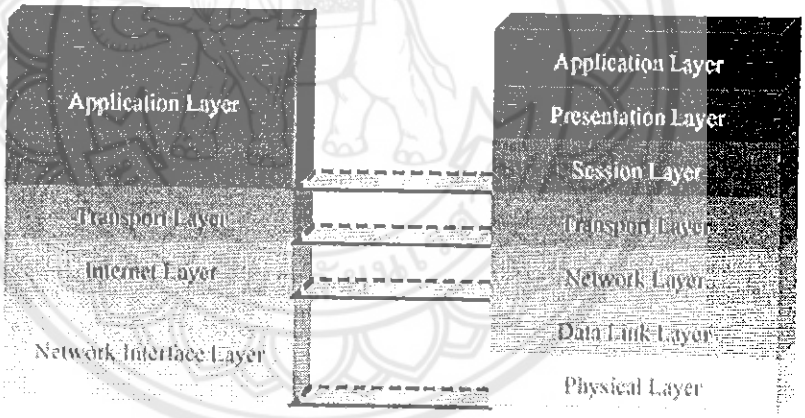
ซอฟต์แวร์ที่ใช้ควบคุมการรับส่งข้อมูลของ ARPANET ประกอบด้วยส่วนหลักๆ 2 ส่วนคือ TCP และ IP ซึ่ง TCP มีหน้าที่ตรวจสอบการรับส่งข้อมูลระหว่างคอมพิวเตอร์ผู้รับและผู้ส่งให้ได้รับข้อมูลถูกต้องครบถ้วน หากข้อมูลสูญหายก็จะแจ้งให้ต้นทางส่งข้อมูลมาใหม่ ส่วน IP จะมีหน้าที่เลือกเส้นทางที่ใช้รับส่งข้อมูลผ่านระบบเครือข่าย และตรวจสอบที่อยู่ของผู้รับโดยใช้ข้อมูลขนาด 4 ไบต์ หรือ 32 บิตเป็นตัวกำหนดที่อยู่ของผู้รับเรียกว่า IP Address ต่อมาในปี ค.ศ. 1983 TCP/IP ถูกกำหนดให้เป็นมาตรฐานการรับส่งข้อมูลของกระทรวงกลาโหมสหรัฐฯ จึงถือว่า TCP/IP มีต้นกำเนิดมาจากโครงการ ARPANET นั่นเอง และได้ถูกรวมเข้าเป็นส่วนหนึ่งของระบบปฏิบัติการ UNIX โดย Bolt. Beranek และ Newman ซึ่งได้รับทุนสนับสนุนจากกระทรวงกลาโหมสหรัฐอเมริกาอีกเช่นเดียวกัน ทำให้ TCP/IP ก้าวเข้าสู่ระบบคอมพิวเตอร์ทางธุรกิจหลังจากใช้งานเฉพาะเครือข่ายของทางทหารมานานถึง 14 ปี

หลังจากที่สถาบันการศึกษาและมหาวิทยาลัยต่างๆในสหรัฐฯ เลือกใช้คอมพิวเตอร์ระบบปฏิบัติการ UNIX กันอย่างแพร่หลาย TCP/IP ก็ยิ่งมีบทบาทในการเชื่อมต่อคอมพิวเตอร์เข้าเป็นเครือข่ายมากขึ้นเรื่อยๆ คนกระทั่งกลายเป็น อินเตอร์เน็ตในปัจจุบันโดยมี ARPANET เป็นแกนกลาง และได้มีการกำหนดมาตรฐานที่ใช้ในการรับส่งข้อมูลเครือข่ายอื่นๆ เพิ่มเติมขึ้นมาในภายหลังรวมทั้ง OSI Model ในเวลาต่อมา

โปรโตคอล TCP/IP สามารถเชื่อมต่อระหว่างเครือข่ายอื่นเข้าด้วยกัน การเชื่อมต่อระหว่างเครือข่ายสามารถทำงานได้กับสารสื่อสารและอุปกรณ์ฮาร์ดแวร์ หรือระบบปฏิบัติการที่แตกต่างกันได้ และยังสามารถสื่อสารถึงกันได้ แม้ว่าอุปกรณ์เครือข่ายบางจุดจะหยุดทำงานหรือเส้นทางถูกตัดขาด โปรโตคอล TCP/IP ประกอบด้วยชุดโปรโตคอลต่างๆ ซึ่งแต่ละโปรโตคอลจะมีความสามารถในการทำงานที่แตกต่างกัน

2.5.2 โครงสร้างโปรโตคอลทีซีพี/ไอพี

โปรโตคอล TCP/IP มีการจัดลักษณะกลไกการการทำงานเป็นชั้นหรือเลเยอร์ (Layer) เรียงต่อกันสามารถเปรียบเทียบกับ OSI Model ตามมาตรฐาน ISO ได้ดังรูปที่ 2.9



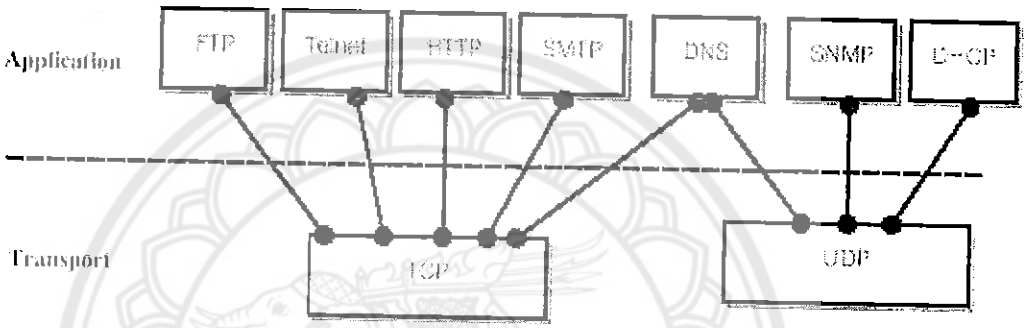
รูปที่ 2.9 การเปรียบเทียบเลเยอร์ของ TCP/IP กับเลเยอร์ของ OSI

ซึ่งเราจะเห็นว่าบางกลไกของโปรโตคอล TCP/IP เทียบได้กับมาตรฐาน OSI สองชั้นหรือบางกลไกก็ทำงานคาบเกี่ยวกันระหว่างบางชั้นของ OSI model ในแต่ละระดับชั้นของ TCP/IP มีการทำงานที่แตกต่างกัน ตั้งแต่การติดต่อกับแอพพลิเคชั่นจนกระทั่งแปลงเป็นสัญญาณส่งไปตามสายสัญญาณ ซึ่งการทำงานในแต่ละระดับชั้นของ TCP/IP มีดังต่อไปนี้

1. แอพพลิเคชั่นเลเยอร์ (Application Layer) แอพพลิเคชั่นเลเยอร์จะรองรับการทำงานของแอพพลิเคชั่นต่างๆ ที่ทำงานเป็นโปรเซสอยู่ในเครื่องที่ให้บริการ (Server) และเครื่องที่ขอใช้บริการ (Client)โดยจัดการเชื่อมต่อระหว่างโปรเซส หรือแอพพลิเคชั่นที่อยู่ต่างเครื่องกัน โดยการ

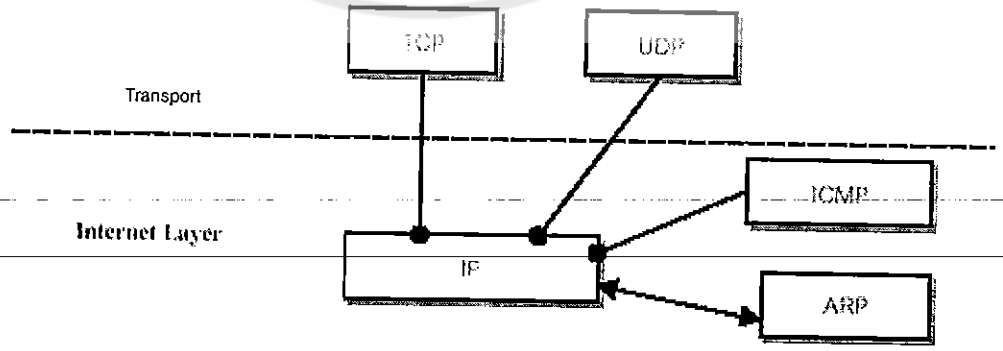
ทำงานของแอพพลิเคชั่นต่างๆ มีการติดต่อกันตามแต่ละโปรโตคอลเฉพาะแล้วแต่แอพพลิเคชั่นที่ใช้งาน ซึ่งจะขอบริการจากทรานสปอร์ตเลเยอร์อีกทีหนึ่ง โปรโตคอลหลักๆ ที่ทำงานในแอพพลิเคชั่นเลเยอร์ ได้แก่ FTP, Telnet, HTTP, SMTP

2. ทรานสปอร์ตเลเยอร์ (Transport Layer) มีหน้าที่จัดการต่อจากแอพพลิเคชั่นเลเยอร์ ทำหน้าที่สร้างการเชื่อมต่อระหว่างแอพพลิเคชั่นแบบ end-to-end โดยจุดที่เชื่อมต่อเพื่อรับส่งข้อมูลนี้เรียกว่า พอร์ต (Port) หรือซ็อกเก็ต (Socket) ในเลเยอร์นี้มีบริการหลักอยู่ 2 แบบ คือ Connection-Oriented โดยเรียกผ่านโปรโตคอล TCP และ Connectionless ซึ่งเรียกผ่านโปรโตคอล UDP



รูปที่ 2.10 โปรโตคอลต่างๆ ที่เรียกใช้ทรานสปอร์ตเลเยอร์

3. อินเทอร์เน็ตเลเยอร์ (Internet Layer) เลเยอร์นี้มีหน้านี้ส่งผ่านข้อมูลระหว่างเครือข่าย โดยมีโปรโตคอลที่ทำงานเป็นกลไกสำคัญในการส่งผ่านข้อมูลไปยังเครือข่ายใดๆ ในอินเทอร์เน็ต คือ IP นอกจากนี้ ในเลเยอร์นี้ยังมีโปรโตคอลที่ทำงานอยู่ด้วยอีก 2 ชนิด คือ ICMP และ ARP



รูปที่ 2.11 โปรโตคอล TCP และ UDP อาศัยโปรโตคอล IP เพื่อส่งข้อมูลระหว่างเครือข่าย

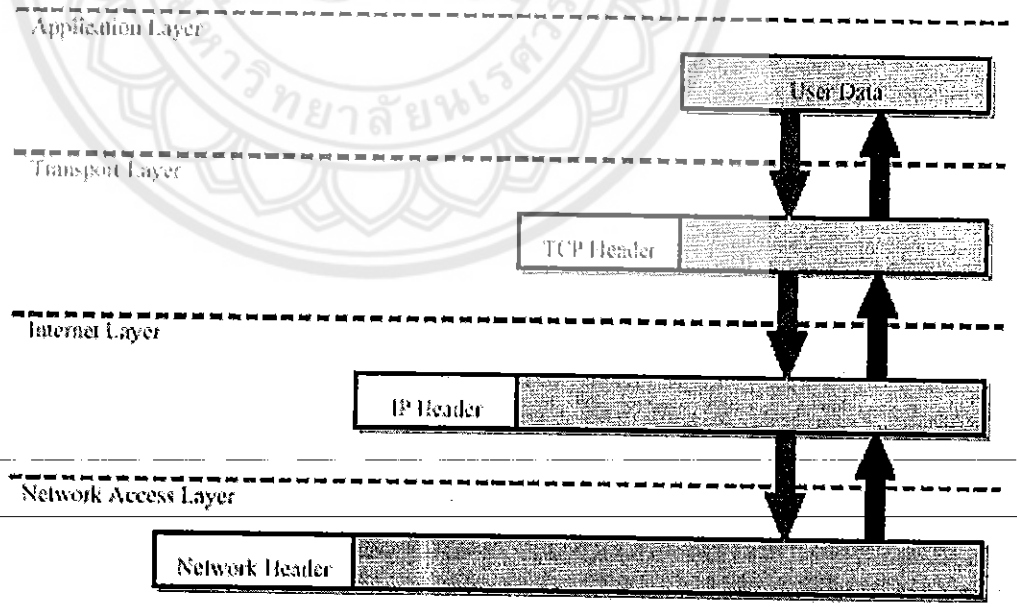
4. เน็ตเวิร์กอินเทอร์เฟสเลเยอร์ (Network Interface Layer) ทำหน้าที่ในการแปลงข้อมูลให้อยู่ในรูปที่เหมาะสมกับเครือข่ายแต่ละแบบ ซึ่งแตกต่างกันออกไปและแปลงเป็นสัญญาณไฟฟ้าส่งไปยังเครือข่าย ในระดับชั้นนี้จะมีการทำงานใน 2 ระดับ คือ ฟิสิคัล (Physical) และดาต้าลิงค์ (Data Link)

- **Physical** เป็นเลเยอร์ที่การกำหนดคุณสมบัติฮาร์ดแวร์ เช่นคุณสมบัติทางกล (หัวต่อและชนิดสายสื่อสาร) และคุณสมบัติทางไฟฟ้า (ลักษณะสัญญาณ และอัตราเร็ว) กล่าวโดยรวมแล้วระดับฟิสิคัลเลเยอร์จะเป็นตัวกำหนดวิธีการถ่ายโอนข้อมูลในระดับบิต ตัวอย่างของการเชื่อมต่อที่ตรงกับระดับฟิสิคัลเลเยอร์ ได้แก่ RS232 และ X.21 เป็นต้น

- **Data Link** เป็นเลเยอร์ของซอฟต์แวร์ (Device Driver) และฮาร์ดแวร์ซึ่งทำงานด้านการเชื่อมโยงเข้ากับสายสื่อสาร ตัวอย่างมาตรฐานในระดับชั้นนี้ ได้แก่ Ethernet, Token-Ring เป็นต้น

2.5.3 การส่งผ่านข้อมูลระหว่างชั้น

การส่งผ่านข้อมูลจากเครื่องที่ให้บริการไปยังเครื่องที่ข้อใช้บริการนั้น ข้อมูลจะส่งผ่านจากโปรโตคอลที่อยู่ระดับบนสุดของเครื่องที่ให้บริการไปยังระดับล่างจนข้อมูลถูกแปลงให้เป็นสัญญาณแล้วเดินทางผ่านเครือข่ายไปยังเครื่องที่ขอใช้บริการ โปรโตคอลระดับล่างสุดที่เครื่องขอใช้บริการจะแปลงสัญญาณที่รับมาแล้วส่งผ่านขึ้นไปยังโปรโตคอลระดับบนต่อไป

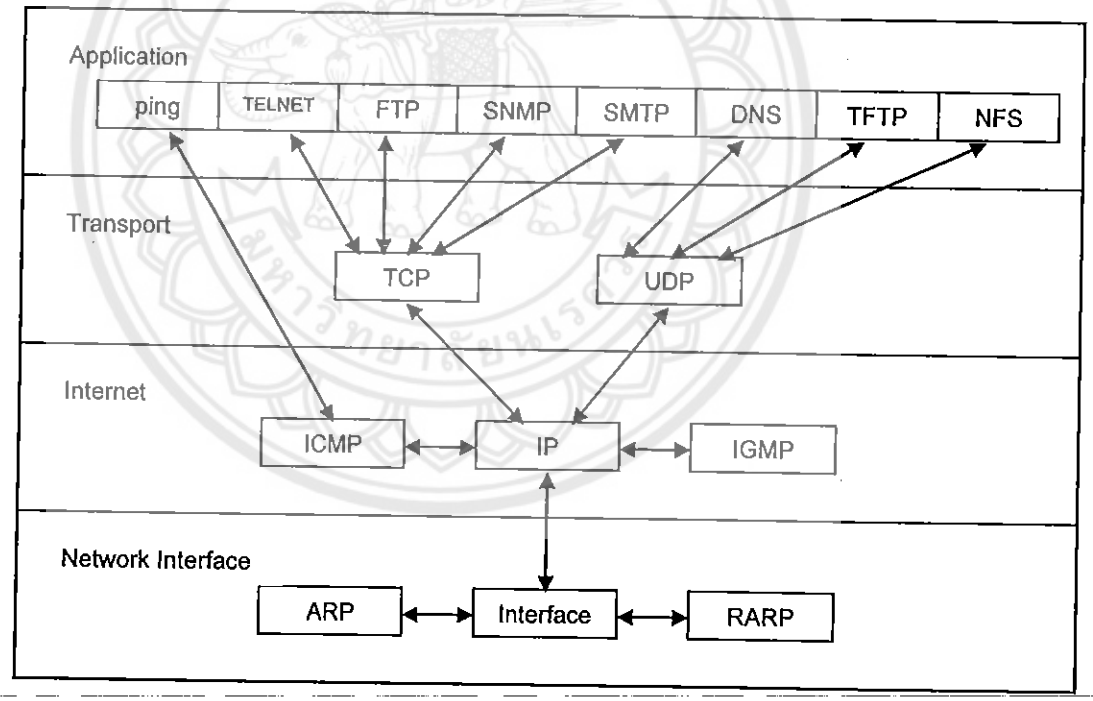


รูปที่ 2.12 การส่งผ่านข้อมูลใน TCP/IP model

เมื่อข้อมูลผ่านแต่ละระดับชั้นโปรโตคอลในแต่ละชั้นจะผนวกข่าวสารและการทำงานของโปรโตคอลภายในชั้นนั้นที่เรียกว่า โปรโตคอลเฮดเดอร์ (Protocol Header) เข้ากับข้อมูลโปรโตคอลระดับล่างจะมองเฮดเดอร์และข้อมูลเหมือนเป็นข้อมูลและเพิ่มข้อมูลของชั้นตัวเองเข้าไป ข้อมูลจะถูกหุ้มเป็นชั้นๆ กระบวนการนี้เรียกว่า การเอ็นแคปซูเลต (Encapsulation) และเมื่อเครื่องที่ขอใช้บริการได้รับแพ็คเก็ตก็จะดำเนินการส่งไปตามลำดับชั้น โปรโตคอลประจำชั้นจะถอดเฮดเดอร์ออกและส่งส่วนที่เหลือไปยังชั้นถัดไป เฮดเดอร์จะถูกถอดออกจนเหลือเฉพาะข้อมูลเมื่อถึงชั้นบนสุด กระบวนการนี้เรียกว่า การดีแคปซูเลต (Decapsulation)

2.5.4 ชุดโปรโตคอลของทีซีพี/ไอพี

การทำงานตามโปรแกรมประยุกต์หนึ่งๆ ไม่ได้ใช้โปรโตคอลพร้อมกันทั้งชุด แต่ใช้เพียงโปรโตคอลที่สัมพันธ์กันไปในแต่ละระดับชั้นของแบบอ้างอิง ตัวอย่างเช่น Telnet จะอาศัย TCP และ IP ตามลำดับ การซ้อนทับของโปรโตคอลจากระดับชั้นบนไปชั้นล่างเรียกว่า โปรโตคอลสแต็ก (Stack Protocol) ดังรูปที่ 2.13



รูปที่ 2.13 โปรโตคอลสแต็กของ TCP/IP

IP ซึ่งอยู่ในเน็ตเวิร์กเลเยอร์ เป็นแกนสำคัญของโปรโตคอลสแต็ก เนื่องจากทั้ง TCP และ UDP ต้องใช้ IP เพื่อเลือกเส้นทางส่งแพ็คเก็ต ในเน็ตเวิร์กเลเยอร์ยังมี ICMP สนับสนุนการทำงานของ IP เพื่อรายงานความผิดปกติที่เกิดขึ้นเนื่องจากการส่งแพ็คเก็ต และมี IGMP ดูแลการจัดกลุ่มโฮสต์ในเครือข่ายมัลติคาสต์ ส่วนทรานสปอร์ตเลเยอร์มี 2 โปรโตคอลที่สำคัญ คือ TCP และ UDP

แอพพลิเคชั่นจะเลือกใช้ TCP หรือUDP ตามความเหมาะสมกับลักษณะงานโปรโตคอลระดับล่างถัดจาก IP ได้แก่ โปรโตคอลระดับเน็ตเวิร์กอินเตอร์เฟสซึ่งกำหนดการทำงานตามเทคโนโลยีเครือข่ายที่ใช้งานในระดับชั้นนี้มีโปรโตคอลในชุดของ TCP/IP ทำหน้าที่สนับสนุนการทำงานอยู่สองโปรโตคอล คือ ARP และ RARP ทั้งสองโปรโตคอลทำหน้าที่แปลงค่าระหว่าง IP Address กับ Hardware Address

2.5.4.1 โปรโตคอลทีซีพี (TCP : Transmission Control Protocol)

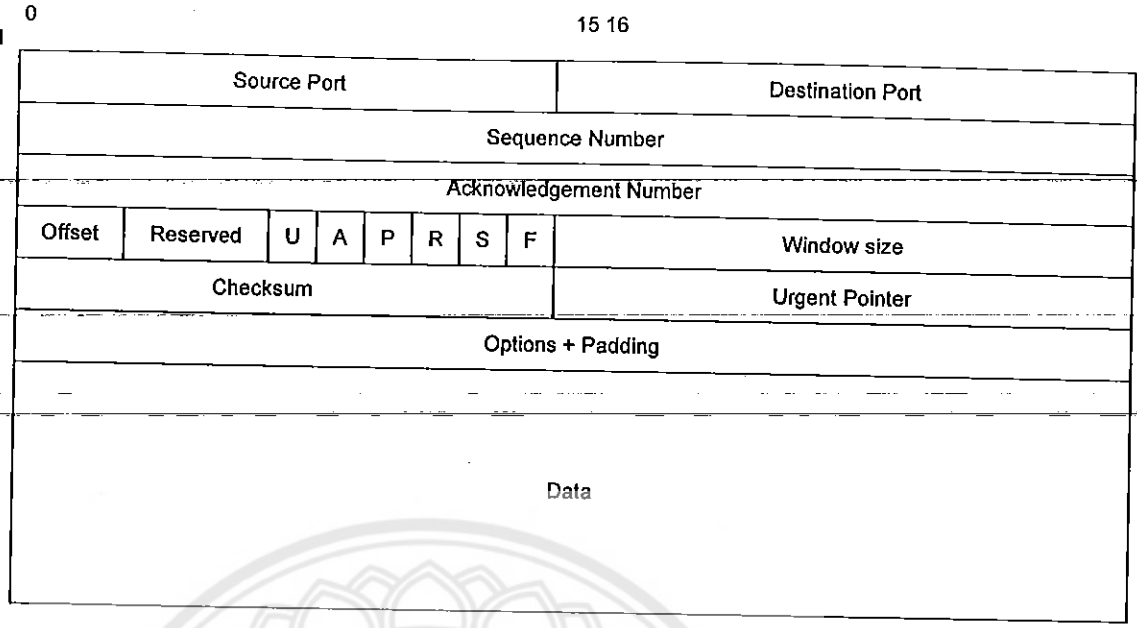
TCP เป็นโปรโตคอลแบบ Connection-Oriented ที่ให้บริการรับส่งข้อมูลโดยรับประกันความเชื่อถือในการส่งข้อมูล โดยทำหน้าที่ตรวจเซกเมนต์ที่ผิดปกติและจัดส่งเซกเมนต์ใหม่ให้ รวมทั้งการจัดลำดับให้ถูกต้องก่อนส่งต่อไปยังโปรแกรมประยุกต์ระดับบน เฮดเดอร์และข้อมูลของ TCP รวมเรียกว่า เซกเมนต์ (segment)

รูปที่ 2.14 TCP Segment

การส่งข้อมูลของ TCP นั้นส่งทีละเซกเมนต์โดยเครื่องปลายทางจะส่งสัญญาณตอบรับมายังเครื่องต้นทางทุกเซกเมนต์ที่ได้รับและตรวจสอบแล้วว่าไม่พบข้อผิดพลาด แต่ถ้าเครื่องต้นทางไม่ได้รับสัญญาณตอบกับมาสำหรับเซกเมนต์ที่ส่งไป สันนิษฐานว่าเซกเมนต์นั้นมีปัญหาเกิดขึ้นและจะส่งเซกเมนต์นั้นไปให้ใหม่อีกครั้ง วิธีการนี้เรียกว่า Positive Acknowledgement with Re-transmission (PAR)

หน้าที่การทำงานของ TCP ในการรับส่งข้อมูลมีหน้าที่หลัก 6 ข้อ คือ

1. ควบคุมการรับส่งข้อมูล (Basic Data Transfer)
2. ความน่าเชื่อถือในการรับส่งข้อมูล (Reliability)
3. ควบคุมการไหลของข้อมูล (Flow Control)
4. การทำมัลติเพล็กซ์ (Multiplexing)
5. ควบคุมการเชื่อมต่อ (Connection)
6. ความปลอดภัยในการรับส่งข้อมูล (Security)



รูปที่ 2.15 แพ็คเก็ตของ TCP

- ส่วนประกอบของทีซีพีเฮดเดอร์

1. Source Port : เป็นหมายเลขพอร์ตของบริการที่เครื่องต้นทาง
2. Destination Port : เป็นหมายเลขพอร์ตของบริการเครื่องปลายทาง
3. Sequence Number : เป็นหมายเลขที่บอกลำดับของการรับส่งข้อมูลของเครื่องที่ต้องการขอส่งข้อมูล
4. Acknowledgement Number : เป็นหมายเลขที่บอกลำดับของการรับส่งข้อมูลที่ฝั่งรับข้อมูล ปกติค่าของ Acknowledgement Number มีค่าเท่ากับ Sequence Number (ของอีกฝั่งหนึ่ง) + 1 เสมอ
5. Data Offset : เป็นตัวบอกค่าออฟเซตของข้อมูล เพราะ TCP นั้นไม่มีการกำหนดความยาวที่แน่นอนของข้อมูล จึงต้องมีออฟเซตเป็นตัวบอก
6. Flag : เป็นบิตที่บอกชนิดของข้อมูล ได้แก่
   - URG : Urgent Pointer Field Significant - แสดง Urgent Pointer
   - ACK : Acknowledgement Field Significant - แสดงการ Acknowledgement
   - PSH : Push Function
   - RST : Reset The Connection - แสดงเมื่อรีเซตการเชื่อมต่อ
   - SYN : Synchronize Sequence Number - หมายเลขแพ็คเก็ตที่ส่งแบบซิงโครนัส

- FIN : No more data from sender - แสดงว่าไม่มีข้อมูลที่ส่งจากผู้ส่งแล้ว

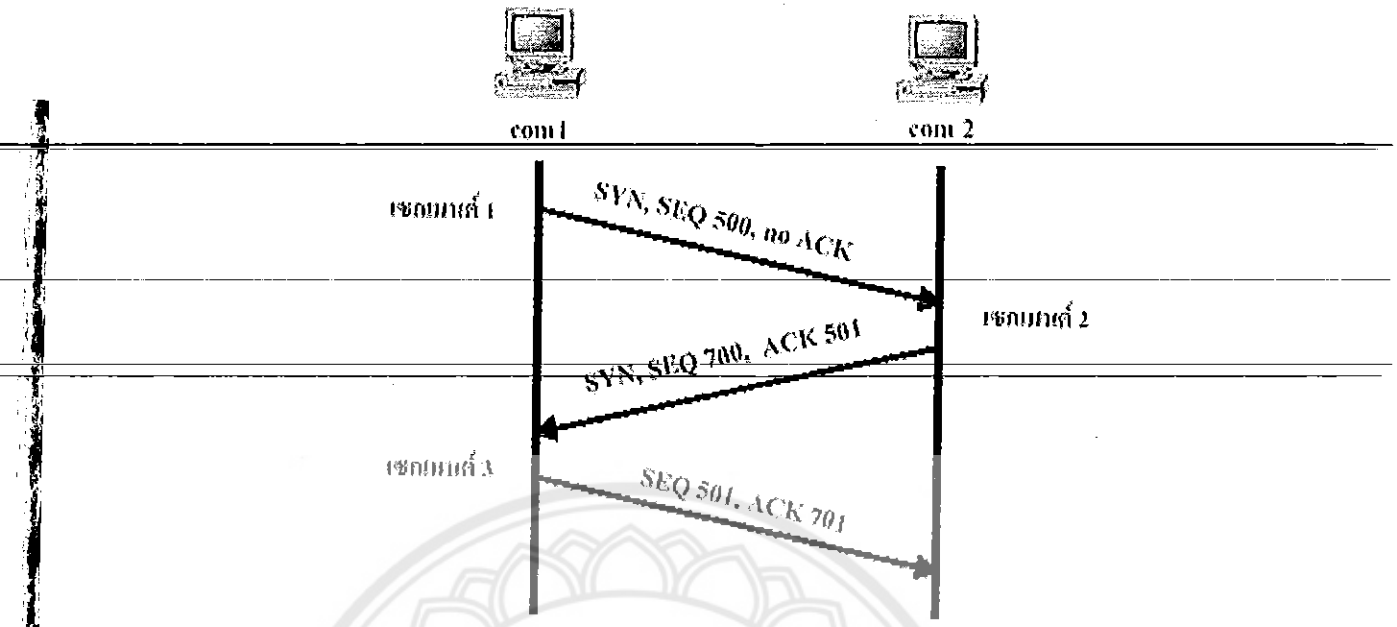
7. Window size : เป็นเลขบอกจำนวนของอ็อกเต็ต (octet) ของข้อมูลจัดการในส่วนของ end-to-end flow control
8. Checksum : เป็นส่วนที่ตรวจสอบความถูกต้องของข้อมูล
9. Urgent Pointer : เป็นตัวชี้ตำแหน่งของ Urgent Data
10. Option and Padding : เป็นตัวบอกออฟชันของโปรเซสที่ใช้ TCP
11. Data : เนื้อข้อมูลที่ต้องการสื่อสาร มีขนาดได้ไม่ต่ำกว่า 532-บิตเวิร์ด (6 บิตแรกสงวนไว้ และกำหนดให้เป็นศูนย์)

- กลไกการทำงานของทีซีพี

ทีซีพีมีกลไกในการลำเลียงข้อมูลระหว่างต้นทางและปลายทางหลายส่วน และมีกลไกที่สำคัญ 3 ประการ ได้แก่ กระบวนการเริ่มต้นในการเชื่อมต่อ กระบวนการถ่ายโอนข้อมูลและกระบวนการยกเลิกการเชื่อมต่อ

การเริ่มต้นการเชื่อมต่อ

การทำ "3-way Handshake" ซึ่งเป็นกระบวนการเริ่มต้นในการสร้างการเชื่อมต่อในทรานสปอร์ตเลเยอร์ คือ ในการติดต่อกันระหว่างระบบในเครือข่ายต้องมีการสร้างการเชื่อมต่อไปยังระบบที่ให้บริการก่อน โดยผู้ขอบริการส่งสัญญาณ SYN เพื่อขอบริการ จากนั้นผู้ให้บริการจะส่งสัญญาณ ACK เพื่อตอบรับการเชื่อมต่อที่ร้องขอมา จึงสามารถรับส่งข้อมูลกันได้ ตัวอย่างเช่น การเริ่มต้นการเชื่อมต่อระหว่างเครื่อง com1 และ com2 โดย com1 เป็นฝ่ายขอบริการจาก com2 โดยโปรโตคอลประยุกต์ของด้าน com1 จะเป็นเครื่องลูกข่ายและโปรโตคอลประยุกต์ทางด้าน com2 เป็นเครื่องแม่ข่าย กระบวนการทำ "3-way Handshake" มีรายละเอียดดังนี้



รูปที่ 2.16 การเริ่มต้นการเชื่อมต่อด้วย TCP

TCP ของ com1 เลือกเลขลำดับเริ่มต้นแล้วส่งเซกเมนต์ที่บรรจุเลขลำดับนี้ไปยัง com2 พร้อมทั้งเซตแฟลกของ SYN ให้เป็น "1" สังเกตว่าเซกเมนต์ 1 แฟลก ACK จะมีค่าเป็น "0"

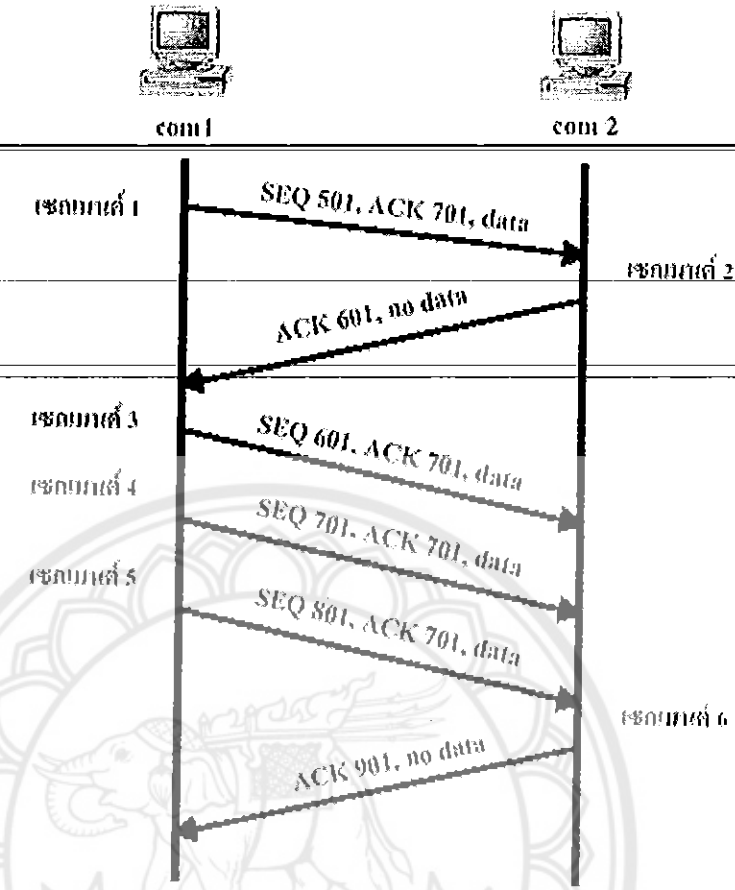
TCP ของเครื่อง com2 ได้รับเซกเมนต์จาก com1 ก็จะเลือกเลขลำดับเริ่มต้นประจำตัวเช่นเดียวกันแล้วตอบกลับด้วยเซกเมนต์ 2 โดยเซตแฟลกของ SYN และ ACK ให้เป็น "1" ทั้งคู่ เพื่อแจ้งว่าได้รับเซกเมนต์1 แล้ว และเลขลำดับที่ได้รับจาก com2 บวกด้วยหนึ่ง

TCP ของ com1 จะส่งเซกเมนต์ตอบรับกลับไปโดยเซตแฟลก ACK และใช้เลขลำดับที่ได้รับจาก com2 บวกด้วยหนึ่ง

ต่อจากนั้น TCP ของ com1 แจ้งไปยังโปรโตคอลระดับบนว่าเชื่อมต่อแล้ว ส่วน com2 เมื่อได้รับเซกเมนต์ 3 ก็แจ้งขึ้นไปยังโปรโตคอลระดับบนว่าเชื่อมต่อแล้ว เมื่อสิ้นสุดนี้การเชื่อมต่อทั้งสองด้านก็เสร็จสมบูรณ์และพร้อมรับส่งข้อมูลได้

การถ่ายโอนข้อมูล

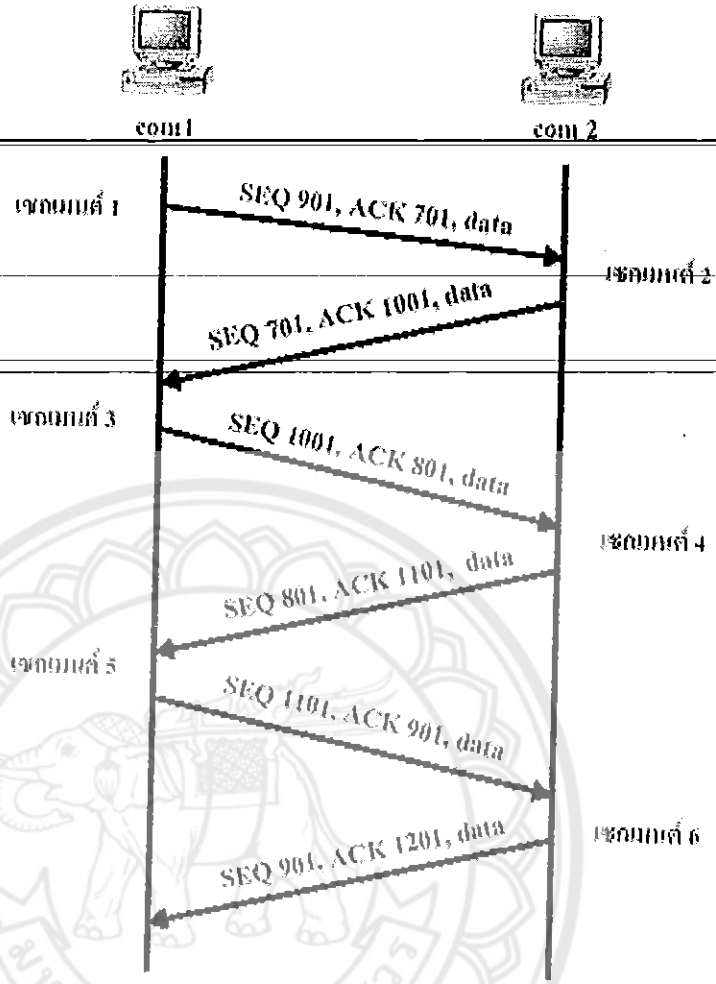
การถ่ายโอนข้อมูลเริ่มต้นหลังจากได้เชื่อมต่อเสร็จสิ้นไปแล้วรูปที่ 2.17 แสดงขั้นตอนการถ่ายโอนข้อมูลจาก com1 ไปยัง com2 ครั้งละ 100 ไบต์ จำนวน 4 ครั้ง โดยเซกเมนต์1 ที่ com1 ส่งไปยัง com2 บรรจุข้อมูลไบต์ 501 ถึง 600 และใช้เลขตอบรับ 701 เมื่อ com2 รับข้อมูลเซกเมนต์1 แล้วก็จะตอบกับด้วยเซกเมนต์ 1 แล้ว ก็ตอบกลับด้วยเซกเมนต์ 2 พร้อมเลขตอบรับกำหนดข้อมูลที่คาดว่าจะได้รับต่อไปคือ 601



รูปที่ 2.17 ขั้นตอนการถ่ายโอนข้อมูล

com1 สามารถส่งเซกเมนต์ไปอย่างต่อเนื่องได้ ดังรูป 2.17 มีการส่งไป 3 เซกเมนต์ คือ 601, 701 และ 801 ทาง com2 สามารถตอบกลับในคราวเดียวกันได้ คือเซกเมต์ 6

ในกรณีที่ส่งข้อมูลทั้งสองทิศทางระหว่าง com1 และ com2 หลักการทั่วไปยังคงเหมือนเดิม เพียงแต่มีการตอบรับพร้อมกับส่งข้อมูลระหว่างกัน ดังรูปที่ 2.18



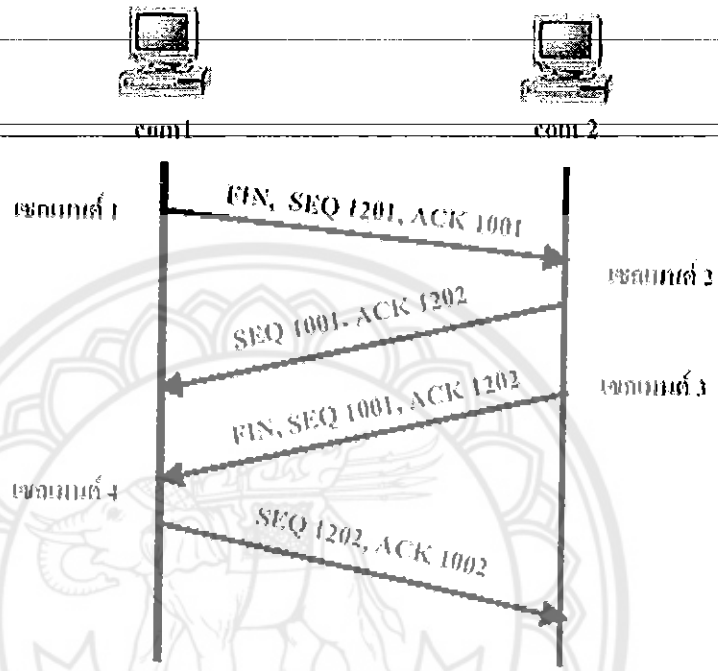
รูปที่ 2.18 การถ่ายโอนเซกเมนต์ทั้งสองทิศทาง

การยกเลิกการเชื่อมต่อ

เนื่องจากการถ่ายโอนใน TCP เป็นแบบฟูลดูเพล็กซ์ การยกเลิกการเชื่อมต่อจึงต้องมีขั้นตอนเกิดขึ้นทั้งสองด้านดังรูปที่ 2.19 ในที่นี้สมมุติว่า com1 ไม่มีข้อมูลส่งอีกต่อไปจึงต้องเป็นฝ่ายขอปิดบริการเชื่อมต่อโดยโปรโตคอลประยุกต์ของ com1 จะแจ้งไปยัง TCP ว่าส่งข้อมูลหมดแล้ว จากนั้นจะมีการแลกเปลี่ยนเซกเมนต์จำนวน 4 เซกเมนต์ดังนี้

1. TCP ของ com1 ส่งเซกเมนต์ที่มีเลขตอบรับและเลขลำดับตามปกติ แต่เซตแฟลก FIN เป็น "1"
2. เมื่อ TCP ของ com2 ได้รับเซกเมนต์ที่มีการเซต FIN จาก com1 จะส่งเซกเมนต์ตอบรับคือเซกเมนต์ 2 และแจ้งไปยังโปรแกรมประยุกต์ com1 ขอปิดการเชื่อมต่อโปรแกรมประยุกต์ com2 แจ้งกลับมายัง TCP ว่าขอปิดการเชื่อมต่อได้

3. TCP com2 ส่งเซกเมนต์ FIN ไปยัง com1 (เซกเมนต์ 3) เมื่อ com1 ได้รับเซกเมนต์ FIN จะตอบรับกลับไป (เซกเมนต์ 4) และแจ้งไปยังโปรแกรมประยุกต์ว่าการเชื่อมต่อปิดลงแล้ว



รูปที่ 2.19 การปิดการเชื่อมต่อ

- การเลื่อนหน้าต่าง (Sliding window)

ในสภาวะการถ่ายโอนข้อมูลระหว่างเครื่องลูกข่าย com1 และเครื่องแม่ข่าย com2 ถ้า com1 ใช้หลักการตอบรับเป็นรายเซกเมนต์ คือทุกครั้งที่ส่งเซกเมนต์ต้องรอการตอบจาก com2 ก่อนส่งเซกเมนต์ถัดไปได้ ช่วงที่ com1 รอการตอบรับนั้นไม่สามารถนำส่งข้อมูลได้และเท่ากับเป็นการใช้สายสื่อสารแบบฮาล์ฟดูเพล็กซ์ซึ่งทำให้เกิดการสูญเสียช่องสัญญาณไปครึ่งหนึ่ง เนื่องจากต้องรอให้อีกด้านส่งข้อมูลเสร็จสิ้นก่อน

TCP แก้ปัญหานี้โดยอาศัยการทำงานแบบฟูลดูเพล็กซ์ ฝ่ายส่งสามารถส่งเซ็กเมนต์ออกได้หลาย เซ็กเมนต์โดยที่ไม่ต้องรอการตอบรับทุกๆ เซกเมนต์ที่ส่งออกไป แต่ต้องมีการควบคุมปริมาณการส่งนี้เพราะบัฟเฟอร์ฝ่ายรับย่อมมีขนาดจำกัด ฝ่ายรับต้องแจ้งให้ทางฝ่ายส่งทราบถึงขนาดข้อมูลที่รับได้โดยใช้เลขตอบรับ (Acknowledgment number) และขนาดหน้าต่าง (windows size) สองค่านี้จึงใช้เป็นตัวกำหนดขนาดการรับส่งข้อมูลแต่ละครั้ง

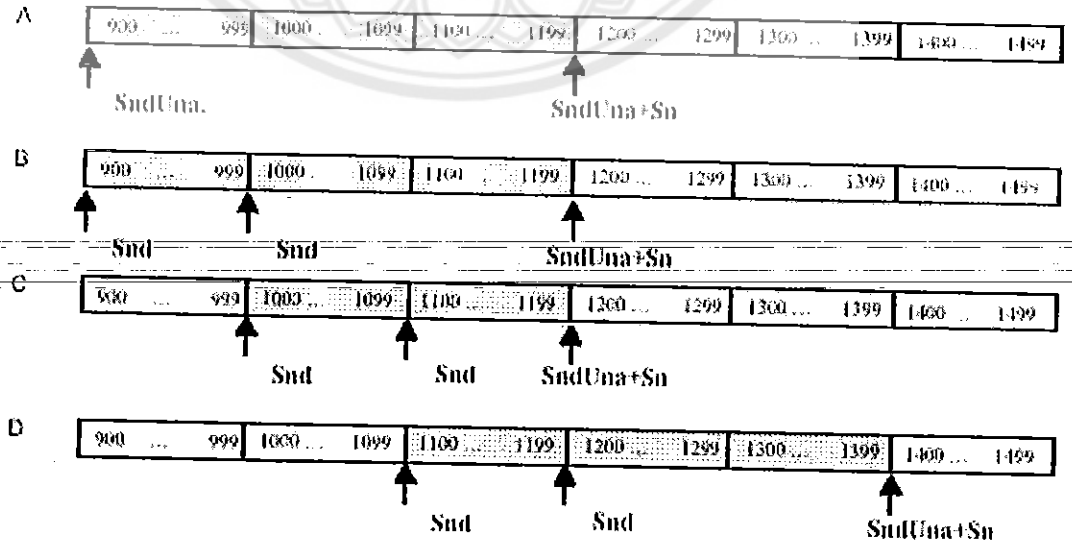
Source Port			Destination Port
Sequence Number			
Acknowledgement Number : 1000			
Offset	Reserved	Flags	Window size : 1024
Checksum			Urgent Pointer
Options + Padding			
Data			

รูปที่ 2.20 ฟิลด์ในทีซีพีเฮดเดอร์ที่ใช้ควบคุมการส่งข้อมูล

จากรูป 2.20 แสดงตัวอย่างทีซีพีเฮดเดอร์ซึ่งประกาศหน้าต่างขนาด 1024 ไบต์ ค่า Acknowledgment จะแจ้งว่าได้รับข้อมูลตั้งแต่ต้นถึงลำดับ 999 ครบถ้วนแล้ว และพร้อมที่จะรับข้อมูลไม่เกินเลขลำดับ 2023 (1000-1+1024)

หน้าต่างฝ่ายส่ง

TCP ควบคุมกระแสข้อมูลที่ฝ่ายส่งโดยใช้ค่า 3 ค่าคือ ตัวแปรชี้ตำแหน่งไบต์ที่ตอบรับ (SndUna), ตัวแปรชี้ตำแหน่งไบร์ที่ต้องส่งครั้งถัดไป (SndNxt) และ ตัวแปรกำหนดหน้าต่างที่ส่งได้ (SndWnd) ค่าทั้ง 3 ค่า จะแบ่งบัฟเฟอร์ของฝ่ายส่งออกเป็น 4 ส่วน เพื่อให้เห็นการทำงานของหน้าต่างฝ่ายรับให้พิจารณารูปที่ 2.21 โดยให้ในสภาวะเริ่มต้นทางฝ่ายส่งมีขนาดหน้าต่างเท่ากับ 300 ไบต์ และหมายเลขลำดับเริ่มต้นที่ 900 ดังรูป 2.21(A)



รูปที่ 2.21 การเลื่อนหน้าต่างฝ่ายส่ง

เมื่อส่งข้อมูลออกไป 100 ไบต์แรก SndNxt จะเลื่อนไปทางขวา 100 ไบต์ ดังรูป 2.21(B) ความจำเป็นของตัวแปรชี้ตำแหน่งไบต์ที่ได้รับการตอบรับล่าสุดเพื่อใช้เก็บตำแหน่งของข้อมูลที่ส่งไปแล้วแต่ยังไม่ได้รับการตอบรับเนื่องจากอาจต้องส่งซ้ำใหม่

ขั้นถัดไปหากมีข้อมูลส่งอีก 100 ไบต์ SndNxt จะขยับไปอยู่ที่ตำแหน่ง 1100 และเมื่อมีการตอบรับข้อมูล 100 ไบต์แรกกลับมา SndUna จะเลื่อนไปที่ตำแหน่ง 1000 ดังรูป 2.21(C)

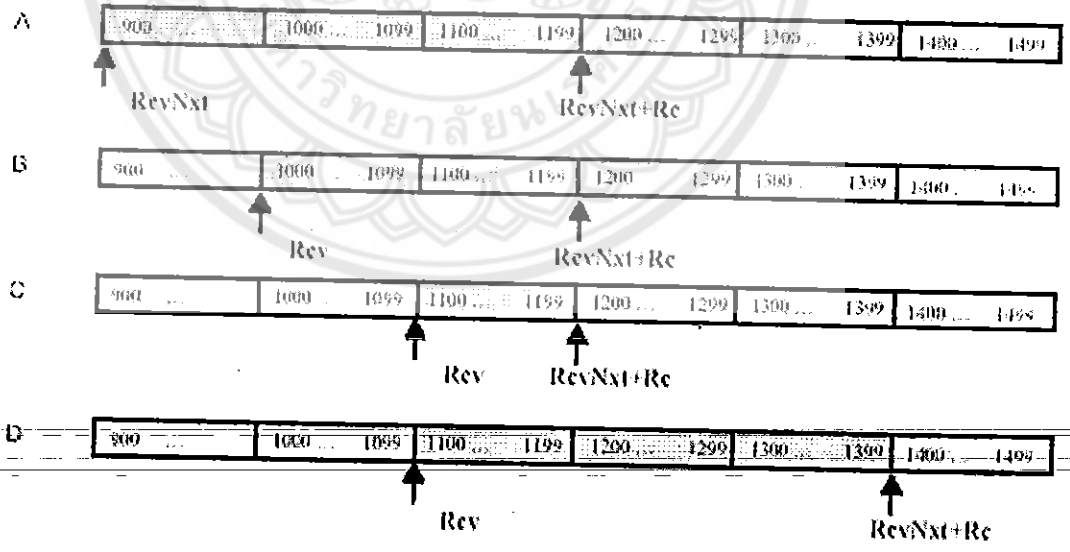
ในขั้นตอนถัดไปหากมีการตอบรับกลับมาพร้อมกับการกำหนดขนาดหน้าต่าง 300 ไบต์ หน้าต่างจะเป็นดังรูปที่ 2.21(D)

หน้าต่างฝ่ายรับ

TCP ฝ่ายรับจัดการกับบัฟเฟอร์โดยให้มีตัวแปรบอกขนาดหน้าต่าง (RcvWnd) และตัวแปรชี้ตำแหน่งข้อมูลถัดไปที่คาดว่าจะได้รับ (RcvNxt) ตัวแปรทั้งสองนี้จะแบ่งบัฟเฟอร์ออกเป็น 3 ส่วน

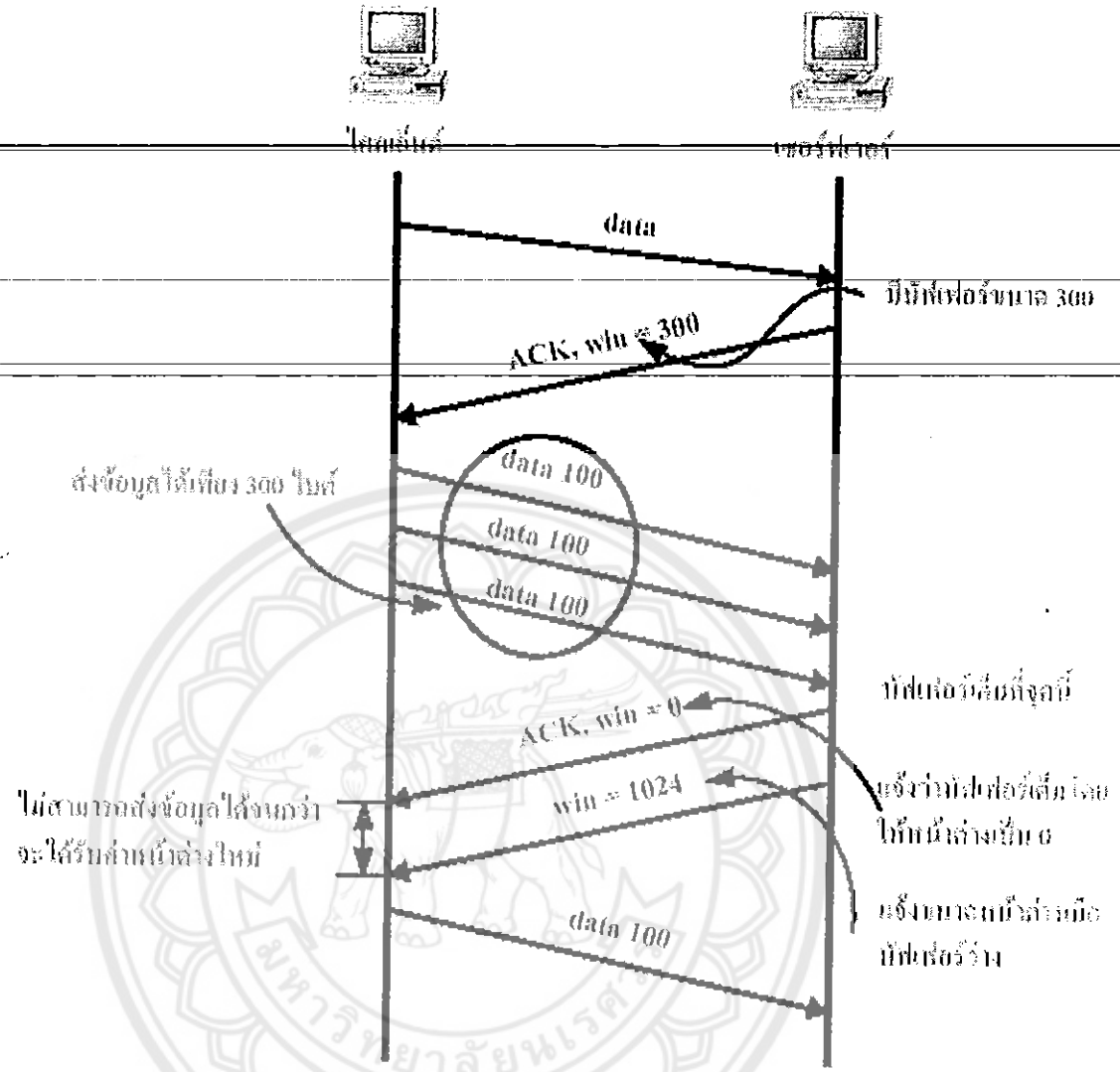
ถ้าพิจารณาการเลื่อนหน้าต่างของฝ่ายรับโดยกำหนดให้มีขนาดหน้าต่างเริ่มต้นที่ 300 ไบต์ ดังรูปที่ 2.22(A) เมื่อข้อมูล 100 ไบต์ แรกเดินทางมาถึงหน้าต่างข้อมูลที่รับได้จะลดลงเหลือ 200 ไบต์ ดังรูปที่ 2.22(B)

ในขั้นต่อมาหากได้รับข้อมูลอีก 100 ไบต์ ขนาดหน้าต่างก็จะลดลงเหลือ 100 ไบต์ ดังรูปที่ 2.22 (C) และเมื่อโปรแกรมประยุกต์นำข้อมูลนั้นไปใช้หน้าต่างก็เลื่อนและขยายขนาดขึ้นดังรูป 2.22 (D)



รูปที่ 2.22 การเลื่อนหน้าต่างฝ่ายรับ

ในกรณีที่บัฟเฟอร์ฝ่ายรับเต็มจนไม่สามารถรับเซกเมนต์เพิ่มได้ TCP จะส่งเซกเมนต์กำหนดขนาดหน้าต่างโดยให้ฟิลด์ window size มีค่าเท่ากับ 0 TCP อีกฝ่ายจะหยุดส่งเซกเมนต์จนกว่าจะได้รับเซกเมนต์ที่กำหนดขนาดหน้าต่างใหม่ ดังรูปที่ 2.23



รูปที่ 2.23 การประกาศขนาดหน้าต่าง

การรอการกำหนดค่าหน้าต่างใหม่เป็นกรณีหนึ่งที่ TCP ต้องดำเนินการเป็นกรณีพิเศษ เพราะถ้าเซกเมนต์ประกาศค่าหน้าต่างใหม่เกิดสูญหาย จะทำให้การสื่อสารไม่สามารถดำเนินต่อไปได้ TCP แก้ปัญหากรณีนี้โดยใช้ตัวจับเวลาเพื่อรับประกันว่าต้องส่งเซกเมนต์กำหนดค่าหน้าต่างใหม่ภายในเวลาที่กำหนด

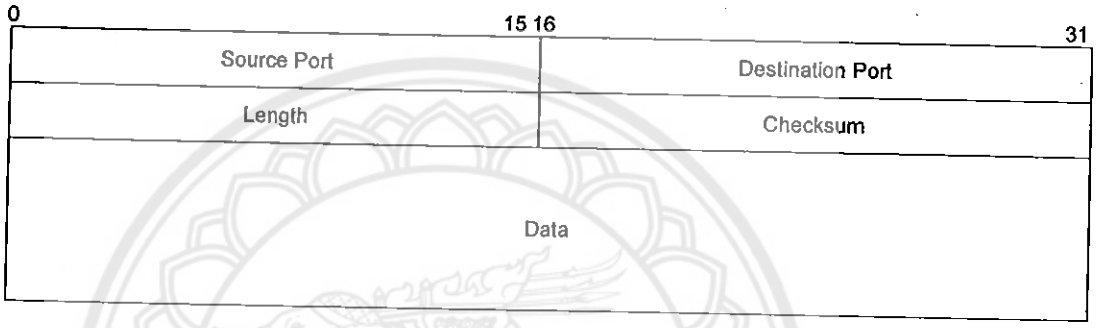
2.5.4.2 โปรโตคอลยูดีพี (UDP: User Datagram Protocol)

โปรโตคอล UDP เป็นโปรโตคอลในการติดต่อสื่อสารในทรานสปอร์ตเลเยอร์ การทำงานคล้ายกับTCP มาก คือจัดการเกี่ยวกับการสื่อสารระหว่างเครื่องแต่เป็นแบบ Connectionless คือทั้งฝ่ายส่งและฝ่ายรับไม่จำเป็นต้องสร้างช่องทางเชื่อมต่อกันและไม่ต้องมีการแจ้งให้ฝ่ายรับข้อมูลเตรียมรับข้อมูลเหมือนโปรโตคอล TCP และไม่มีการส่งสัญญาณตรวจสอบว่าข้อมูลถึงเครื่อง

ปลายทางอย่างถูกต้องครบถ้วน ในการส่งข้อมูลแต่ละครั้งจึงไม่มีการส่งข้อมูลใหม่อีกในกรณีที่เกิดความผิดพลาดของการส่งข้อมูล

- ส่วนประกอบของ UDP Frame

1. Source Port : เป็นค่าตัวเลข 16 บิต บอกพอร์ตของบริการที่เครื่องต้นทาง
2. Destination Port : เป็นค่าตัวเลข 16 บิต บอกพอร์ตของบริการที่เครื่องปลายทาง
3. Length : เป็นค่าตัวเลข 16 บิต บอกความยาวของข้อมูล
4. Checksum : เป็นค่าตัวเลข 16 บิต ตรวจสอบความถูกต้องของข้อมูลที่ส่ง

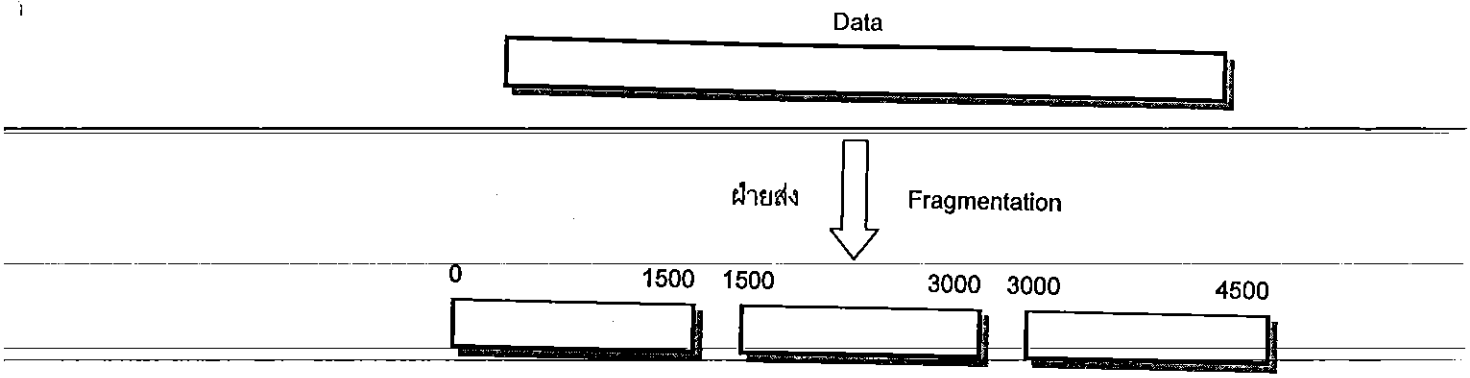


รูปที่ 2.24 แพ็คเก็ต UDP

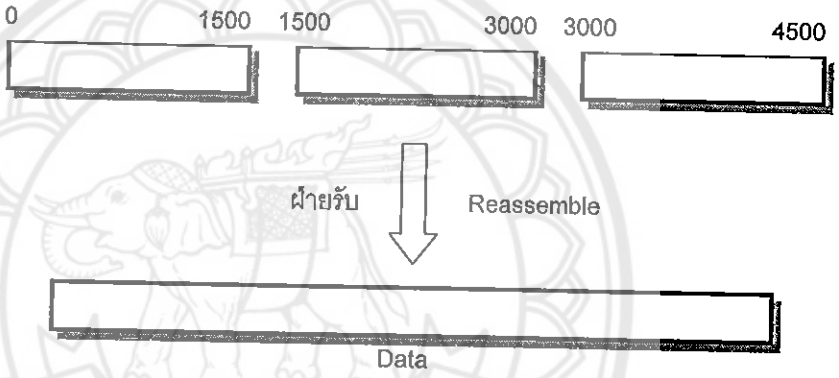
2.5.4.3 โปรโตคอลไอพี (IP: Internet Protocol)

โปรโตคอล IP เป็นโปรโตคอลที่จัดการเกี่ยวกับแอดเดรสของแต่ละแพ็คเก็ต เพื่อให้ส่งแพ็คเก็ตต่างๆ ไปยังเป้าหมายได้ถูกต้อง การทำงานของ IP เป็นเพียงการส่งข้อมูลไปยังเครื่องเป้าหมายเท่านั้น ไม่มีการส่งสัญญาณขอบริการ หรือสัญญาณให้บริการระหว่างกันเหมือน TCP เรียกว่าการเชื่อมต่อแบบ Connectionless ซึ่งระบบทั้งสองตั้งสมมุติฐานว่าการเชื่อมต่อระหว่างกันไม่มีความผิดพลาดเกิดขึ้นแน่นอน

เนื่องจากมาตรฐานในเครือข่ายมีหลากหลาย ขนาดของแพ็คเก็ตในแต่ละมาตรฐานจึงมีความแตกต่างกันออกไป ทำให้การส่งข้อมูลระหว่างอุปกรณ์ในเครือข่ายนั้นอาจมีการแบ่งข้อมูลออกเป็นแพ็คเก็ตย่อยๆ ในระหว่างการส่ง เรียกว่า การทำแฟร็กเมนเตชัน (Fragmentation) เช่น แพ็คเก็ตของ FDDI มีขนาด 4,500 ไบต์ ถ้าเครื่องปลายอยู่ในเครือข่ายอีเธอร์เน็ต (Ethernet) ซึ่งมีขนาดของแพ็คเก็ตสูงสุดเพียง 1,500 ไบต์ ดังนั้นการส่งแพ็คเก็ตไปยังเครื่องปลายทางจึงต้องมีการแบ่งเป็นแพ็คเก็ตย่อย และเมื่อแพ็คเก็ตย่อยมาถึงเครื่องเป้าหมายก็จะมารวมกันเป็นแพ็คเก็ตเดิมที่มีขนาด 4,500 ไบต์อีกครั้ง เรียกการรวมกันนี้ว่า การรีแอสเซมเบิล (Reassemble) ซึ่งทำให้ได้ข้อมูลเหมือนที่ส่งมาจากเครื่องต้นทาง



รูปที่ 2.25 การทำแฟรกเมนเตชั่น



รูปที่ 2.26 การรีแอสเซมเบิล

- ส่วนประกอบของแพ็คเก็ตไอพี

1. version : เป็นค่าตัวเลข 4 บิต บอกเวอร์ชันของมาตรฐาน IP ที่ใช้ โดยปกติมีค่าเป็น 4 ซึ่งหมายถึง IPv4
2. Internet Header Length (IHL) : เป็นตัวบอกความยาวเฮดเดอร์ของ IP
3. Type of Service : เป็นส่วนที่บอกการทำงานของแพ็คเก็ตที่ส่งว่าทำหน้าที่อะไร มีทั้งหมด 8 บิต

Bit 0-2 : บอกรายละเอียดการทำงานของแพ็คเก็ตนั้นๆ

111 - Network Control

110 - Internetwork Control

101 - CRITIC / ECP

100 - Flash Override

011 - Flash

010 - Immediate

001 - Priority

000 - Routine

Bit 3 : บอกถึงลักษณะของดีเลย์ (Delay)

0 = Normal Delay - มีดีเลย์ปกติ

1 = Low Delay - มีดีเลย์ต่ำ

Bit 4 : บอกถึงประเภทของทรูพุต (Throughput)

0 = Normal Throughput – มีทรูพุตปกติ

1= High Throughput – มีทรูพุตสูง

Bit 5 : บอกถึงประเภทของความน่าเชื่อถือ

0 = Normal Reliability - มีความน่าเชื่อถือพอประมาณ

1 = High Reliability - มีความน่าเชื่อถือสูง

Bit 6-7 : กันไว้ใช้ในอนาคต

4. Total Length : มีขนาด 16 บิต บอกถึงความยาวในดาต้าแกรมของ IP
5. Identification field : เป็นตัวเลข 16 บิต เป็นค่าประจำตัวของ IP นั้น โดยโฮสต์ที่ส่งเป็นผู้กำหนด และเพิ่มค่าขึ้นหนึ่งเมื่อมีการส่งดาต้าแกรมของ IP ใหม่ ซึ่งใช้ในการประกอบกลับ
6. Flag : เป็นตัวเลข 3 bit บอกลักษณะของแพ็คเก็ตว่ามีการแฟร็กเมนต์หรือไม่

Bit 0 : สงวนไว้ ปกติเป็น 0

Bit 1 : 0 = บอกว่าแพ็คเก็ตมีการแตกแพ็คเก็ตย่อย

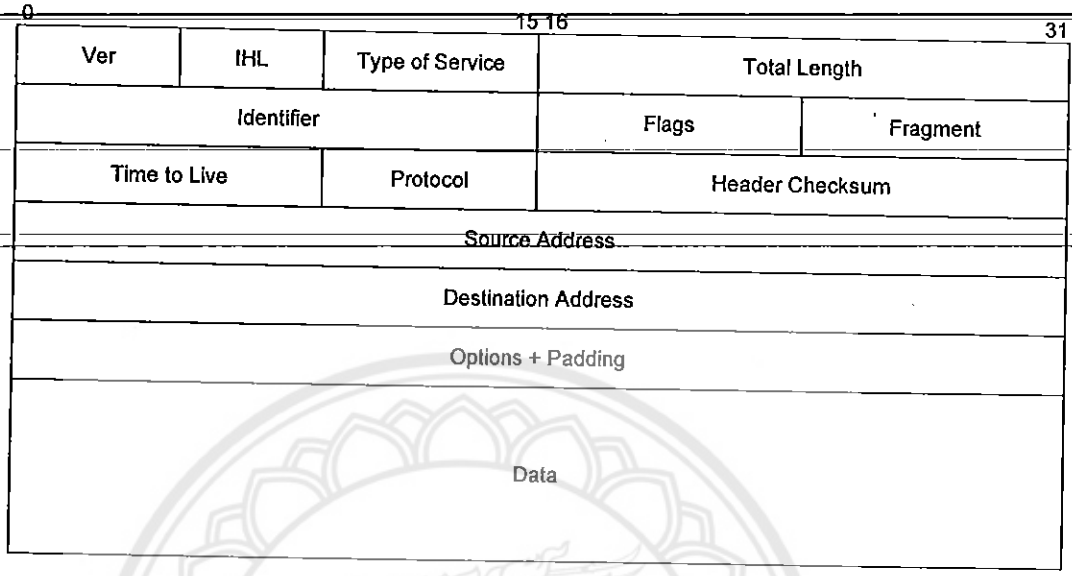
1 = บอกว่าแพ็คเก็ตไม่มีการแตกแพ็คเก็ตย่อย

Bit 2 : 0 = บอกว่าแพ็คเก็ตนั้นเป็นแพ็คเก็ตสุดท้ายที่ได้จากการแตกแพ็คเก็ตย่อย

1 = บอกว่าแพ็คเก็ตนั้นยังไม่ใช่แพ็คเก็ตสุดท้ายที่ได้จากการแตกแพ็คเก็ตย่อย

7. Fragment Offset : เป็นค่าตัวเลข 13 บิต บอกออฟเซต (offset) ของแฟร็กเมนต์เมื่อเทียบในดาต้าแกรม
8. Time To Live (TTL) : เป็นตัวเลข 8 บิต บอกช่วงเวลาของแพ็คเก็ตที่ยังอยู่ในเครือข่ายได้ โดยกำหนดค่าเป็นจำนวนเราเตอร์สูงสุดที่ดาต้าแกรมผ่านได้ ซึ่งโดยทั่วไปมีค่าระหว่าง 32 ถึง 64 และลดค่าลงเรื่อยๆ เมื่อผ่านเราเตอร์ เพื่อเป็นการป้องกันแพ็คเก็ตล้นเครือข่าย
9. Protocol : เป็นตัวเลข 8 bit บอกถึงโปรโตคอลที่อยู่เหนือขึ้นไปว่าเป็นโปรโตคอลระดับสูงกว่าประเภทใด
10. Header Checksum : เป็นค่าตัวเลข 32 บิต ใช้ตรวจสอบความถูกต้องของเฮดเดอร์

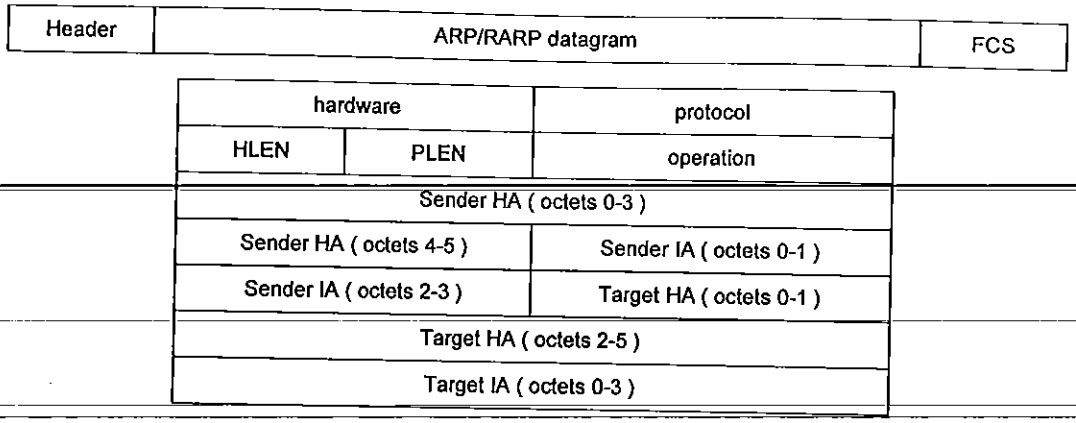
11. Source Address : เป็นค่าตัวเลข 32 บิต บอกถึงไอพีแอดเดรสของเครื่องต้นทาง
12. Destination Address : เป็นค่าตัวเลข 32 บิต บอกถึงไอพีแอดเดรสของเครื่องปลายทาง



รูปที่ 2.27 แพ็คเก็ต IP

2.5.4.4 โปรโตคอลเออาร์พี (ARP:Address Resolution Protocol)

โปรโตคอล ARP เป็นโปรโตคอลที่ออกแบบมาเพื่อใช้ในเครือข่ายที่สนับสนุนการบอร์ดคาสต์ (Broadcast) ถูกเรียกใช้งานโดยโปรโตคอล IP เพื่อช่วยแปลงหมายเลข IP ไปเป็นหมายเลขฮาร์ดแวร์ปลายทาง ตัวอย่างเช่น เว็บเซิร์ฟเวอร์เครื่องหนึ่งเชื่อมต่ออยู่ในเครือข่ายอินเตอร์เน็ตและในการเชื่อมต่อนี้ต้องอาศัยการ์ดแลน (LAN card) ติดตั้งอยู่ ที่แลนการ์ดนี้จะมีหมายเลขเฉพาะประจำฮาร์ดแวร์ที่ไม่ซ้ำกับใคร เพื่อใช้อ้างอิงการส่งข้อมูลในเครือข่าย แต่เมื่อมาใช้งานในโปรโตคอล TCP/IP ก็ต้องมีการกำหนดหมายเลข IP Address ประจำตัวเพื่อใช้อ้างอิงกัน และโปรโตคอล ARP จะทำหน้าที่แปลงค่าหมายเลข IP ให้เป็นหมายเลขฮาร์ดแวร์จริงในระดับการทำงานที่อินเทอร์เน็ตเลเยอร์ ซึ่งกลไกการแปลงนี้เรียกว่า แอดเดรสรีโซลูชั่น (address resolution)



รูปที่ 2.28 ARP Datagram

- ส่วนประกอบของเออาร์พีดาต้าแกรม

1. Hardware 16 บิต : กำหนดชนิดของฮาร์ดแวร์เครือข่ายที่ ARP ทำงานอยู่ ค่าใช้งานมีตัวอย่าง ดังนี้

   1 อีเทอร์เน็ต

   4 โทเค็นริง

   5 เคออส (chaos)

   6 เครือข่าย IEEE 802

   7 อาร์คเน็ต

   12 โลคัลทอล์ค

2. Protocol 16 บิต : ชนิดของโปรโตคอลที่ร้องขอใช้ ARP
3. HLEN 8 บิต : ขนาดของฮาร์ดแวร์แอดเดรสเป็นจำนวนไบต์ ค่าปกติที่ใช้งาน คือ 6 ซึ่งเท่ากับขนาด 6 ไบต์ของอีเทอร์เน็ตฮาร์ดแวร์แอดเดรส
4. PLEN 8 บิต : ขนาดของแอดเดรสระดับเน็ตเวิร์กเป็นจำนวนไบต์ ค่าปกติที่ใช้ คือ 4 ซึ่งเท่ากับขนาด 4 ไบต์ของไอพีแอดเดรส
5. Operation 16 บิต : กำหนดรูปแบบการใช้ดาต้าแกรม ค่าในฟิลด์นี้ใช้กำหนดการทำงานของทั้ง ARP และ RARP ซึ่งมี 4 ค่า คือ

   ARP request (ค่าเท่ากับ 1)

   ARP reply (ค่าเท่ากับ 2)

   RARP request (ค่าเท่ากับ 3)

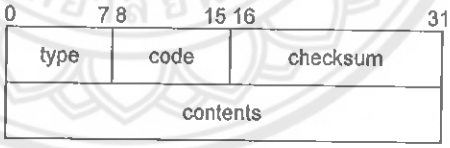
   RARP reply (ค่าเท่ากับ 4)

6. Address : ฟิลด์แอดเดรสเรียงลำดับจากฮาร์ดแวร์และเน็ตเวิร์กแอดเดรสของสถานีที่ร้องขอตามด้วยฮาร์ดแวร์และเน็ตเวิร์กแอดเดรสของสถานีที่ตอบรับ

2.5.4.5 โปรโตคอล ไอซีเอ็มพี (ICMP : Internet Control Message Protocol)

หน้าที่หลักของโปรโตคอล ICMP คือการแจ้งหรือแสดงข้อความจากระบบ เพื่อบอกให้ผู้ใช้ทราบว่าเกิดอะไรขึ้นในการส่งผ่านข้อมูลนั้น ซึ่งปัญหาส่วนมากที่พบ คือ ส่งไปไม่ได้ หรือปลายทางรับข้อมูลไม่ได้ เป็นต้น นอกจากนี้โปรโตคอล ICMP ยังถูกเรียกใช้งานจากเครื่องแม่ข่ายและเราเตอร์อีกด้วย เพื่อแลกเปลี่ยนข้อมูลที่ใช้ควบคุม ส่วนรูปแบบการทำงานของโปรโตคอล ICMP นั้นจะทำงานควบคู่กับโปรโตคอล IP ในระดับเดียวกัน และข้อความต่างๆ ที่แจ้งให้ทราบจะถูกผนึกอยู่ภายในข้อมูลของ IP (IP Datagram) อีกทีหนึ่ง ข้อความที่โปรโตคอล ICMP ส่งนั้นแบ่งออกได้ 2 แบบ คือ ICMP error message หรือข้อความแจ้งข้อผิดพลาด และ ICMP query หรือข้อความเรียกขอข้อมูลเพิ่ม ตัวอย่างกลไกการทำงานของโปรโตคอล ICMP เช่น เมื่อมีการส่งผ่านข้อมูลจากผู้ใช้ไปยังปลายทางที่ไม่ถูกต้อง หรือขณะนั้นเครื่องปลายทางเกิดปัญหาจนไม่สามารถรับข้อมูลได้ที่เราเตอร์จะส่งข้อความแจ้งเป็น ไอซีเอ็มพีเมสเซจ (ICMP message) ที่ชื่อ destination unreachable ให้กับผู้ส่งข้อมูล นอกจากนี้ตัวข้อมูลที่แจ้งข้อความก็จะมีส่วนของข้อมูล IP Datagram ที่เกิดปัญหาด้วย ดังนั้นเมื่อผู้ส่งข้อมูลได้รับข้อความแจ้งแล้วก็จะได้ทราบว่าจุดที่เกิดปัญหานั้นอยู่ที่ใด

ดังนั้นโปรโตคอล ICMP จึงกลายมาเป็นเครื่องมืออย่างหนึ่งในการช่วยทดสอบเครือข่าย เช่น คำสั่ง ping ที่เรามักใช้ทดสอบว่าเครื่องแม่ข่ายที่ให้บริการหรืออุปกรณ์ที่ต่ออยู่ในเครือข่ายอินเตอร์เน็ตนั้นยังทำงานเป็นปกติหรือไม่ แล้วคำสั่ง ping มีการเรียกใช้งานโปรโตคอล ICMP แจ้งเป็นข้อความให้ทราบอีกต่อหนึ่ง



รูปที่ 2.29 รูปแบบของ ICMP

- ส่วนประกอบของไอซีเอ็มพี

1. Type ขนาด 8 บิต : กำหนดค่าความผิดพลาดและการรายงานสถานะการใช้งานในปัจจุบัน มีทั้งหมด 15 ประเภท
2. Code ขนาด 8 บิต : รหัสความผิดพลาดย่อย
3. Checksum ขนาด 16 บิต : ค่าผลรวมตรวจสอบแบบ 1's complement สำหรับใช้ตรวจสอบความผิดพลาด โดยคำนวณผลรวมของ type, code และ contents

4. Contents ขนาดไม่คงที่ : ฟิลด์นี้ใช้บรรจุข้อมูลข่าวสารเพิ่มเติมเพื่อแจ้งกลับซึ่งจะขึ้นอยู่กับค่า type และ code

2.6 การโจมตีเพื่อให้ปิดบริการ (Denial of Services Attack)

Denial of Services เป็นการโจมตีเป้าหมายด้วยวิธีการต่างๆ เพื่อไม่ให้เป้าหมายสามารถให้บริการได้ตามปกติ การ DoS ส่วนใหญ่จะอาศัยข้อบกพร่องของโปรโตคอลประกอบกับข้อบกพร่องของระบบปฏิบัติการที่ทำงานอยู่บนเป้าหมาย ทำให้การทำงานและการตอบสนองต่อการ DoS นั้นส่งผลกระทบอย่างรุนแรงต่อสมรรถนะการทำงาน ซึ่งในสภาวะการทำงานปกติแล้วถึงแม้ว่าระบบปฏิบัติการและโปรโตคอลจะมีข้อบกพร่องดังกล่าว แต่จะไม่มีปัญหาเกิดขึ้นเพราะไม่มีการกระตุ้นลักษณะที่เป็นปัญหานั้น ไม่ว่าจะเป็นข้อมูลที่เข้ามาในภาวะปกติหรือเกิดจากความผิดพลาดในการสื่อสารข้อมูลก็ตาม

ร่องรอยที่เกิดขึ้นจากการโจมตีลักษณะนี้จะสามารถตรวจพบได้จากลักษณะต่างๆของแพ็คเก็ตบนเครือข่าย เช่นเดียวกับที่สามารถตรวจพบการสแกนได้จากรูปแบบของแพ็คเก็ต การ DoS นั้นสามารถทำได้หลายวิธีโดยมีจุดมุ่งหมายไปยังข้อบกพร่องในส่วนต่างๆ ของเครือข่ายที่แตกต่างกันออกไป ซึ่งก่อนที่จะวิเคราะห์การโจมตีได้นั้นจะต้องทราบกลไกการทำงานในระดับเครือข่ายก่อน

เนื่องจากการค้นพบการ DoS แต่ละวิธีนั้นเป็นไปอย่างไม่เป็นทางการและมีที่มาแตกต่างกัน ชื่อของการ DoS ส่วนใหญ่จึงเป็นในลักษณะที่เรียกกันภายในกลุ่มนั้นๆ ส่งผลให้การ DoS วิธีเดียวกันอาจมีชื่อเรียกแตกต่างกันออกไป ดังนั้นเพื่อให้อ้างอิงกับมาตรฐานที่ยอมรับกันทั่วไปจะยกตัวอย่างของ DoS ที่อ้างอิงกับมาตรฐานของ CERT (Computer Emergency Response Team) แห่งประเทศสหรัฐอเมริกา ซึ่งเป็นศูนย์กลางการแก้ปัญหาภัยคุกคามบนอินเตอร์เน็ตของสหรัฐอเมริกา โดย CERT ได้เผยแพร่ คำแนะนำของความไม่ปลอดภัยและข้อบกพร่องในระบบคอมพิวเตอร์อยู่อย่างสม่ำเสมอ และอีกมาตรฐานหนึ่งที่ได้อ้างอิงถึง คือ CVE (Common Vulnerabilities and Exposure) ซึ่งเป็นของ Mitre Corporation ทำหน้าที่รวบรวมข้อบกพร่องจากแหล่งต่างๆและจัดหมวดหมู่ให้อยู่ในฐานการอ้างอิงเดียวกัน เพื่อจะได้ตรงกันและลดปัญหาความซ้ำซ้อนของวิธีการอ้างอิงโดยใช้ชื่อที่แตกต่างกันไปของแต่ละหน่วยงาน

2.6.1 การโจมตีประเภทต่างๆ

2.6.1.1 Anomalous Packet

Anomalous Packet หมายถึง แพ็คเก็ตประหลาดที่ไม่มีโอกาสเกิดขึ้นในสภาวะปกติได้โดยสิ้นเชิง แพ็คเก็ตประเภทนี้เป็นการจงใจเปลี่ยนข้อมูลสำคัญที่ใช้ควบคุมการสื่อสารข้อมูลให้ผิดปกติหรือผิดธรรมชาติของการสื่อสารข้อมูลธรรมดา ไม่ว่า IP, UDP หรือ TCP ต่างก็มีข้อกำหนดอยู่ในโปรโตคอลของตนเอง ผู้ที่จะใช้งานโปรโตคอลเหล่านี้ต้องปฏิบัติตามจึงจะสามารถสื่อสารข้อมูล

กันได้ ระบบปฏิบัติการต่างๆจึงให้ความสนใจไปที่การจัดการสื่อสารภายใต้โปรโตคอลให้ถูกต้องมีเสถียรภาพและมีประสิทธิภาพ โดยไม่ได้ให้ความสนใจต่อกรณีอื่นๆ ที่เกิดขึ้นได้จากการที่ข้อมูลมีรูปแบบผิดปกติไปจากที่ควรจะเป็น

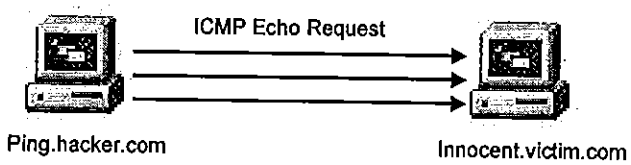
ในแต่ละโปรโตคอลย่อมมีค่าในเฮดเดอร์ที่จะใช้เป็นกลไกการควบคุมการสื่อสารข้อมูล โดยในสภาวะของการสื่อสารข้อมูลตามปกติแล้วค่าในเฮดเดอร์เหล่านี้จะมีค่าที่มีขอบเขตและคาดหมายได้ แต่ Anomalous Packet เหล่านี้จะเป็นแพ็คเก็ตที่ถูกดัดแปลงด้วยเทคนิคของการควบคุมเน็ตเวิร์กเลเยอร์โดยตรงแบบไม่ผ่านโปรโตคอล ซึ่งเป็นช่องทางให้แฮคเกอร์ทั้งหลายได้ใช้โจมตีเป้าหมายให้ทำงานผิดพลาดไปเพราะต้องจัดการกับแพ็คเก็ตที่ไม่ได้ระบุอยู่ในโปรโตคอลหรือไม่เช่นนั้นก็ใช้เพื่ออำพรางตนเองในการสำรวจเป้าหมาย เป็นต้น สำหรับการดัดแปลงของแต่ละโปรโตคอลสามารถแสดงได้พอสังเขปดังนี้

- IP : IP Length, Fragment Offset, Fragment Flag, TCP Option
- UDP : UDP Length, Socket (Address & Port)
- TCP : TCP Flag, Socket (Address & Port)

Anomalous Packet ที่นิยมนำมาใช้งานกันมากที่สุดจะเป็น TCP โดยเฉพาะการปรับเปลี่ยน flag ไปในแบบต่างๆ ที่จะสามารถเปลี่ยนได้

เนื่องจาก Anomalous Packet เหล่านี้เปรียบเสมือนการเรียกใช้งานระบบปฏิบัติการให้ทำงานนอกเหนือจากเงื่อนไขที่ได้กำหนดไว้ตามปกติ ดังนั้นผลที่ได้รับจึงแตกต่างกันออกไปขึ้นอยู่กับระบบปฏิบัติการ ถ้าระบบปฏิบัติการสามารถจัดการกับแพ็คเก็ตประเภทนี้ได้ดีก็จะไม่มีผลกระทบใดๆมากนัก แต่ถ้าระบบปฏิบัติการไม่สามารถจัดการกับแพ็คเก็ตแปลกประหลาดนี้ได้อย่างเหมาะสมก็จะส่งผลกระทบที่รุนแรงได้ Anomalous Packet ที่สามารถทำให้ระบบปฏิบัติการทำงานผิดปกติได้อย่างรุนแรงบางประเภทสามารถทำให้ระบบปฏิบัติการหยุดทำงานได้ในทันทีหลังจากที่ได้รับ Anomalous Packet เพียงแพ็คเก็ตเดียว จึงมีแฮคเกอร์บางกลุ่มที่ใช้คุณสมบัติเหล่านี้ของ Anomalous Packet เพื่อทำการโจมตีเป้าหมายให้หยุดการบริการ ซึ่งเป็นเทคนิคหนึ่งที่ได้ผลและทำให้เกิดความเสียหายต่อระบบเป้าหมายอย่างมากโดยที่เป้าหมายยากที่จะป้องกันตนเองได้

2.6.1.2 Ping Flood Attack



รูปที่ 2.30 การโจมตีแบบ Ping Flood Attack

Ping Flood เป็นการโจมตีที่ใช้กันในยุคแรกๆของ DoS เป็นการโจมตีที่ไม่ได้อาศัยเทคนิคที่ลึกลับซับซ้อนแต่อย่างใด โดยอาศัยปริมาณแพ็คเก็ตมากๆเพียงอย่างเดียว แต่ถึงอย่างนั้นก็ตามจากการโจมตีวิธีนี้ก็สร้างความเสียหายได้ไม่น้อย

หลักการโจมตีของ Ping Flood คือการส่ง ICMP Echo Request (แบบเดียวกับที่ได้จากคำสั่ง Ping) ปริมาณมากไปยังเป้าหมายอย่างรวดเร็ว ทำให้โฮสต์ที่ถูกโจมตีจะต้องคอยตอบ ICMP Echo Reply ตลอดเวลาจนแทบจะไม่มีเวลาทำงานอื่น ความรุนแรงของการโจมตีจะมากหรือน้อยแปรผันตามความเร็วของการส่ง ICMP แพ็คเก็ตเป็นหลัก

นอกจากการสร้างความเสียหายแก่เครื่องคอมพิวเตอร์เป้าหมายแล้ว Ping Flood ยังสร้างความเสียหายให้แก่เครือข่ายที่เครื่องคอมพิวเตอร์เป้าหมายนั้นอยู่ด้วย โดยความเสียหายจะมีขอบเขตมากน้อยเพียงใดขึ้นอยู่กับลักษณะการออกแบบของเครือข่ายนั้นๆ ถึงแม้ว่าแพ็คเก็ตที่ส่งไปยังเครื่องแม่ข่ายของเป้าหมายนั้นจะมีหมายเลข IP เป็นของเป้าหมายเครื่องเดียว แต่ในความเป็นจริงที่เลเยอร์ต่ำลงไป เช่น Ethernet ต่างก็ใช้งานร่วมกันกับเครื่องโฮสต์อื่นๆในเครือข่าย การที่ข้อมูลจะเดินทางจากที่ใดและจะไปที่ใดจะต้องผ่านเครือข่ายที่ใช้งานร่วมกันนี้ ดังนั้นเมื่อแพ็กเก็ตจำนวนมากถูกส่งมายังเครื่องเป้าหมาย นอกจากจะทำให้เครื่องเป้าหมายเสียหาย แล้วเครือข่ายที่เป็นทางผ่านก็จะเต็มไปด้วยแพ็คเก็ตนี้เช่นกัน ลักษณะเช่นนี้จะเกิดมากบนเครือข่ายที่มีการใช้การสื่อสารเลเยอร์ต่ำร่วมกัน เช่น 10Base-5 หรือ 10Base-T ที่ใช้ Hub เป็นตัวกระจายสัญญาณ

ข้อสังเกตของการโจมตีแบบนี้ คือ การปรากฏของ ICMP Echo Request ปริมาณมาก ซึ่งอาจมีที่มาจากที่ใดที่หนึ่งหรือหลายที่แต่มีเป้าหมายเดียวกันในเวลาอันรวดเร็ว และนั่นคือปลายทางของแพ็คเก็ตเหล่านั้นกำลังตกเป็นเป้าหมายของการโจมตีแล้ว และข้อสังเกตอีกประการหนึ่ง คือการโจมตีแบบนี้จะต้องทำควบคู่ไปกับการปลอมแปลงหมายเลข IP ต้นทาง เนื่องจากการส่ง ICMP Echo นั้นจะได้รับ ICMP Echo Reply กลับมาเสมอ ดังนั้นถ้าไม่มีการปลอม IP ต้นทาง ICMP Echo Reply ก็จะตอบกลับมายังแฮคเกอร์ในจำนวนเท่ากับที่ส่งไปโจมตี ซึ่งอาจทำให้แฮคเกอร์เองก็ได้รับผลกระทบด้วย ดังนั้นหากตรวจสอบพบแพ็คเก็ตประเภทนี้แสดงว่าจะต้องมาจากต้นทางปลอมแน่นอน และ IP จากต้นทางปลอมยังเป็นหลักประกันได้ว่าจะไม่สามารถติดตามได้ว่าผู้ใดเป็นผู้โจมตี รูปแบบแพ็คเก็ตที่เกิดขึ้นจากการโจมตีมีลักษณะดังนี้

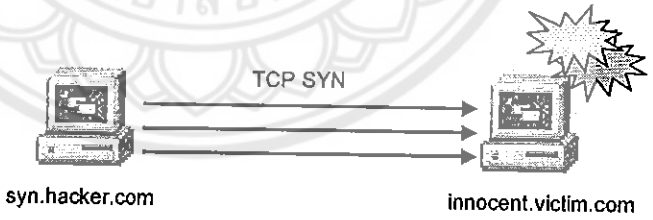
14:49:43.217137	62.51.12.2 > 10.1.1.10	icmp: echo request
14:49:43.217175	10.1.1.10 > 62.51.12.23	icmp: echo reply
14:49:43.217195	62.81.12.3 > 10.1.1.10	icmp: echo request
14:49:43.217219	10.1.1.10 > 62.51.12.23	icmp: echo reply
14:49:43.217245	96.39.61.1 > 10.1.1.10	icmp: echo request
14:49:43.217279	10.1.1.10 > 96.141.10.124	icmp: echo reply
14:49:43.237136	75. 26.62.6> 10.1.1.10	icmp: echo request
14:49:43.237169	10.1.1.10 > 75.126.62.65	icmp: echo reply

รูปที่ 2.31 แพ็คเก็ตของ Ping Flood Attack

การป้องกัน Ping Flood ทำได้โดยการไม่อนุญาตให้แพ็คเก็ตของ ICMP Echo เข้าไปยังเครือข่ายได้โดยการกำหนดที่เราเตอร์หรือไฟร์วอลล์ แต่อย่างไรก็ตาม ICMP Echo ก็ถูกใช้สำหรับโปรแกรม Ping เพื่อตรวจสอบสถานะของเครื่องแม่ข่าย ถ้าปิดไม่ให้ ICMP Echo ผ่านเข้าไปโปรแกรม Ping ก็จะทำงานไม่ได้เช่นกัน

สำหรับการป้องกันการ Ping Flood อีกส่วนหนึ่งสามารถทำได้บนเราเตอร์ โดยควบคุมแบนด์วิดธ์ของแพ็คเก็ต Ping ให้มีอัตราเหมาะสมเท่าที่จำเป็นต่อการ Ping แต่ไม่ควรให้มีแบนด์วิดธ์มากเกินไปจนสามารถนำมาใช้เพื่อโจมตีได้

2.6.1.3 SYN Flood Attack



รูปที่ 2.32 การโจมตีแบบ SYN Flood Attack

ถ้าจะนับว่าการโจมตีของ Ping Flood เป็นการทดลองกำลังกันในระดับ IP (ด้วย ICMP) SYN Flood ก็นับเป็นการลองกำลังในระดับ TCP สิ่งที่เป็นคุณสมบัติสำคัญของ TCP คือการเชื่อมต่อที่มีเสถียรภาพ โดยมีขั้นตอนการเริ่มต้นการเชื่อมต่อและยุติการเชื่อมต่อ แต่ข้อดีเหล่านี้กลับถูกนำไปใช้เป็นอาวุธสำคัญในการโจมตี

ด้วยลักษณะในการเริ่มต้นเชื่อมต่อของ TCP นั้นจะเป็นการตรวจสอบซึ่งกันและกันทั้ง 2 ฝ่ายที่เรียกว่า 3-ways handshake โดยเริ่มต้นจากเครื่องที่ต้องการติดต่อ ส่งสัญญาณ SYN มายังเครื่องแม่ข่าย หลังจากนั้นการเริ่มต้นการเชื่อมต่อตามโปรโตคอลก็จะดำเนินต่อไป

สิ่งที่ยากที่สุดในการทำงานของ TCP คือการพยายามทำให้ทั้งสองฝ่ายสามารถสื่อสารกันได้อย่างถูกต้องและมีเสถียรภาพ สิ่งที่ต้องพิจารณา คือแพ็คเก็ตของการเชื่อมต่อจะเริ่มต้นด้วยการได้รับสัญญาณ SYN และจะต้องตอบ SYN ACK กลับไปให้แก่ผู้ที่ขอ จากนั้นต้องรอการตอบรับอีกครั้งหนึ่งของเครื่องลูกข่ายจึงจะจบกระบวนการ ดังนั้นเพื่อที่จะให้การเชื่อมต่อสามารถดำเนินไปได้อย่างต่อเนื่องโปรแกรมที่จัดการ 3-ways handshake ของเครื่องแม่ข่ายจะต้องจัดสรรหน่วยความจำจำนวนหนึ่งเพื่อรองรับการเชื่อมต่อแต่ละเซสชั่นจนกว่าการทำ 3-ways handshake จะสิ้นสุดลง โดยที่เครื่องแม่ข่ายเองก็ไม่มีทางรู้ได้เลยว่าเครื่องลูกข่ายจะ ACK กลับมาเพื่อจบการเชื่อมต่อนั้นของ 3-ways handshake เมื่อไร ซึ่งแน่นอนว่าในการบริหารหน่วยความจำและการสื่อสารข้อมูลจะต้องสร้างความสมดุลระหว่างเสถียรภาพของเครื่องแม่ข่ายและประสิทธิภาพของการสื่อสาร เครื่องแม่ข่ายเองก็จะมีเวลาค่าหนึ่งที่จะรอให้ได้รับสัญญาณ ACK ตอบกลับมา ถ้าถึงเวลาที่กำหนดแล้วไม่มีแพ็คเก็ต ACK กลับมาเครื่องแม่ข่ายจะต้องยุติการรอนั้นและคืนหน่วยความจำให้แก่ระบบปฏิบัติการ

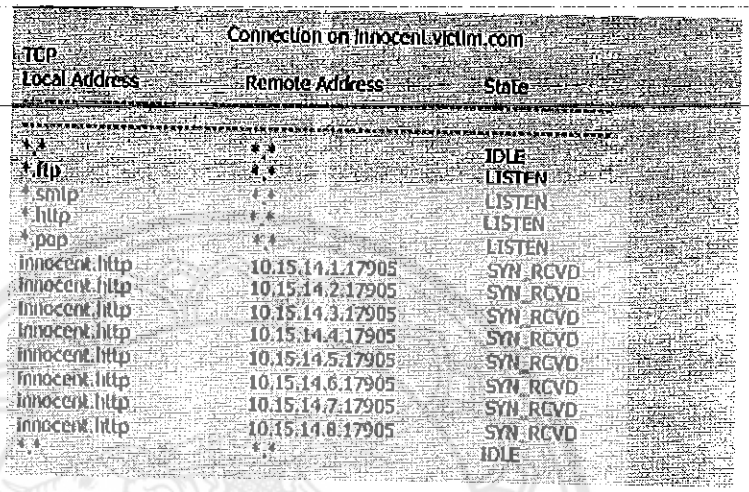
เทคนิคที่อาศัยการ SYN นี้จะส่งผลกระทบโดยตรงกับเครื่องแม่ข่าย ถ้าระบบปฏิบัติการของเครื่องแม่ข่ายเป้าหมายจัดการหน่วยความจำได้ไม่มีประสิทธิภาพพอก็อาจจะหยุดทำงานทันที

จนถึงปัจจุบัน SYN Flood ก็ยังเป็นการโจมตีที่ได้ผลอยู่และหาทางป้องกันได้ยาก เนื่องจากยากที่จะจำแนกลักษณะของแพ็คเก็ตที่ใช้โจมตีกับแพ็คเก็ตที่ขอเริ่มต้นการเชื่อมต่อทั่วไป โดยเฉพาะสำหรับเครื่องแม่ข่ายที่มีการเข้าใช้งานสูงๆ เช่น เว็บเซิร์ฟเวอร์ และถึงแม้ว่าระบบปฏิบัติการรุ่นใหม่จะได้มีการปรับปรุงในด้านการจัดการหน่วยความจำให้ดีขึ้นแต่ก็คงจะช่วยได้เพียงทำให้เครื่องแม่ข่ายไม่ถึงกับหยุดทำงานไปเลยเมื่อโดนโจมตีเท่านั้น แต่ยากที่จะทำให้เครื่องแม่ข่ายยังคงรักษาความสามารถในการให้บริการได้เหมือนในสภาวะปกติ

```
10:09:43.137 10.0.0.1 > innocent.victim.com.80: S 399259715:399259715(0)
10:09:43.139 10.0.0.2 > innocent.victim.com.80: S 399259715:399259715(0)
10:09:43.143 10.0.0.3 > innocent.victim.com.80: S 399259715:399259715(0)
10:09:43.149 10.0.0.4 > innocent.victim.com.80: S 399259715:399259715(0)
10:09:43.158 10.0.0.5 > innocent.victim.com.80: S 399259715:399259715(0)
10:09:43.165 10.0.0.6 > innocent.victim.com.80: S 399259715:399259715(0)
```

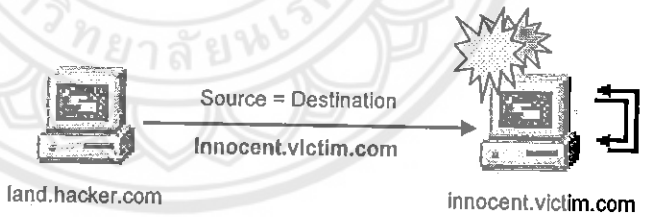
รูปที่ 2.33 แพ็คเก็ตของ SYN Flood Attack

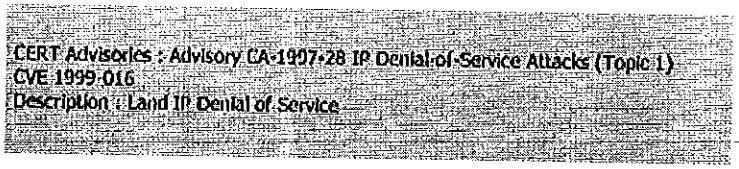
เนื่องจาก SYN Flood เป็นการโจมตีตรงไปยังระบบปฏิบัติการ ดังนั้นวิธีป้องกันที่ดีที่สุด คือ การปรับปรุงระบบปฏิบัติการให้ทันสมัยที่สุด ติดตั้ง Bug Fix และ Service pack ต่างๆ ที่ผู้ผลิตทำออกมาเพื่อป้อกัน TCP Attack จะเป็นหนทางแก้ไขที่ตรงจุดและมีประสิทธิภาพที่สุด สำหรับเราเตอร์จะไม่สามารถตรวจสอบและป้องกันการโจมตีชนิดนี้ได้



รูปที่ 2.34 สถานะการเชื่อมต่อบน innocent.victim.com เมื่อถูกโจมตี

2.6.1.4 Land Attack





รูปที่ 2.35 การโจมตีแบบ Land Attack

ลักษณะการโจมตี

- หมายเลข IP ต้นทางเท่ากับ IP ปลายทาง
- หมายเลขพอร์ตต้นทางเท่ากับหมายเลขพอร์ตปลายทาง

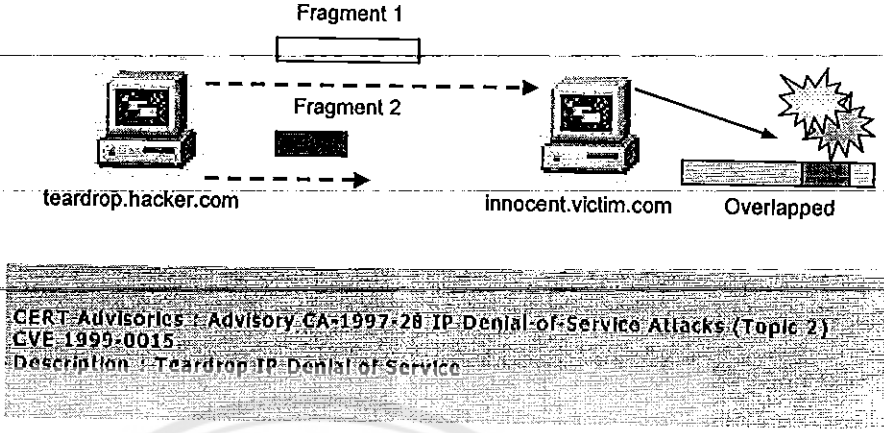
- SYN Flag ถูกเซตเสมือนขอเริ่มต้นการเชื่อมต่อ
- แพ็คเก็ตจะส่งไปยัง TCP พอร์ตที่เปิดอยู่

ปกติในข้อกำหนดของ TCP เมื่อมีการส่งสัญญาณ SYN มาเพื่อการเชื่อมต่อไปยังเป้าหมาย เป้าหมายก็จะตอบรับกลับไปยังผู้ส่งด้วย SYN ACK ตามหมายเลขพอร์ตและหมายเลข IP ต้นทาง แต่สำหรับการโจมตีแบบนี้หมายเลข IP ต้นทางและปลายทางอีกทั้งหมายเลขพอร์ตต้นทางและปลายทางจะถูกตั้งให้เป็นค่าเดียวกัน ดังนั้นการตอบกลับด้วย SYN ACK ก็จะตอบกลับไปที่ปลายทางเหมือนเดิม ซึ่งกรณีนี้ไม่มีกำหนดอยู่ในโปรโตคอลว่าควรจะทำอย่างไร โฮสต์จึงพยายามตอบสนองตามข้อกำหนดเท่าที่มีอยู่โดยการตอบกลับไปที่ IP Address และพอร์ตต้นทางที่ถูกบุกรุกมา นั่นหมายถึงการตอบกลับเข้ามายังตัวเอง ซึ่งจะทำให้มีการตอบกลับไปมาของ TCP วนรอบอยู่ในตัวเองด้วยความเร็วสูง ทำให้คอมพิวเตอร์ต้องใช้ทรัพยากรที่มีอยู่ทั้งหมด ไม่ว่า CPU, หน่วยความจำ และอื่นๆ เพื่อคอยจัดการกับ TCP ที่ตอบกลับไปมาดังกล่าวจนไม่อาจไปทำงานอื่นๆ ได้อีกจนดูเหมือนว่าเครื่องคอมพิวเตอร์หยุดทำงานและไม่ตอบสนองต่อการกระตุ้นใดๆ แม้แต่คีย์บอร์ด ดังนั้นมีวิธีเดียว คือต้องรีเซ็ตเครื่องหรือปิดเครื่องจึงจะสามารถหยุดการวนรอบของ TCP ได้

แพ็คเก็ตประเภทนี้เกิดจากการโจมตีจากภายนอกโดยการปลอม IP Address ให้ดูเหมือนว่าโฮสต์เป้าหมายเป็นผู้ส่งออกมาเอง ซึ่งแม้ว่าจะเป็นแพ็คเก็ตที่ปลอม IP Address มาก็ตาม แต่เมื่อโฮสต์เป้าหมายได้รับแพ็คเก็ตนี้ก็ไม่มีกลไกตรวจสอบใดๆ ที่จะยืนยันได้ว่าเป็นแพ็คเก็ตที่มี IP Address เป็นของจริงหรือถูกปลอมมา โฮสต์เป้าหมายไม่สามารถทำอะไรได้นอกจากต้องตอบกลับไปยังผู้ส่งแพ็คเก็ตเข้ามา และเมื่อนั้นโฮสต์ก็จะหยุดทำงานในที่สุด

Land Attack เป็นการโจมตีที่ได้ผลดีในยุคแรกๆ โดยอาศัยข้อบกพร่องของหมายเลข IP ต้นทาง, หมายเลขพอร์ตต้นทาง, หมายเลข IP ปลายทาง และหมายเลขพอร์ตปลายทางที่เป็นค่าเดียวกัน การโจมตีประเภทนี้ทำงานอยู่บนพื้นฐานของการปลอม IP ดังนั้นถ้าป้องกันการปลอม IP ก็จะป้องกันการโจมตีประเภทนี้ได้ แต่การปลอม IP ใช้ได้ผลกับการปลอมข้ามเครือข่ายเท่านั้น

2.6.1.5 Teardrop Attack



รูปที่ 2.36 การโจมตีแบบ Teardrop Attack

Teardrop เป็นการโจมตีโดยใช้ข้อบกพร่องของแฟรกเมนต์รีแอสเซมเบิล (การรวมแฟรกเมนต์แพ็คเก็ตหลายแพ็คเก็ตกลับมาเป็น IP Datagram เดียว) ของ IP เพื่อทำให้ระบบปฏิบัติการปลายทางของเป้าหมายทำงานผิดพลาด ไม่อยู่ในเงื่อนไขที่กำหนดไว้และหยุดทำงาน

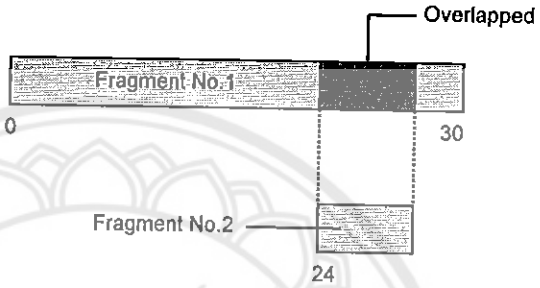
ในการประกอบรวมแฟรกเมนต์แพ็คเก็ตกลับมาอยู่ใน 1 ดาต้าแกรมนั้น IP มีลักษณะการทำงานโดยพิจารณาจากข้อมูลของ 3 ฟิลด์ คือ

1. Data length : ขนาดความยาวของข้อมูลในแพ็คเก็ตนั้น
2. Offset : ตำแหน่งของแพ็คเก็ตที่จะนำไปประกอบรวมกลับใน IP Datagram
3. Flag : แฟลกซึ่งระบุว่าไม่มีแพ็คเก็ตต่อจากแพ็คเก็ตนี้อีก หมายถึงแพ็คเก็ตนี้เป็นส่วนสุดท้ายของดาต้าแกรม

โดยทั่วไปแล้ว IP แพ็กเก็ตที่ถูกแฟรกเมนต์มานั้นไม่จำเป็นจะต้องถึงปลายทางตามลำดับ ดังนั้นปลายทางผู้รับแพ็คเก็ตจึงต้องทำการรีแอสเซมเบิลโดยนำแพ็คเก็ตที่เข้ามาไปวางรอไว้ตรงตำแหน่งของ offset และ รอจนกว่าทุกแพ็คเก็ตจะมาครบ จากนั้นจึงส่ง IP Datagram ที่สมบูรณ์นั้นไปยังเลเยอร์ถัดขึ้นไป ด้วยลักษณะของการทำงานเช่นนี้ทำให้มีข้อบกพร่องมากมายที่สามารถนำไปใช้ในการโจมตี

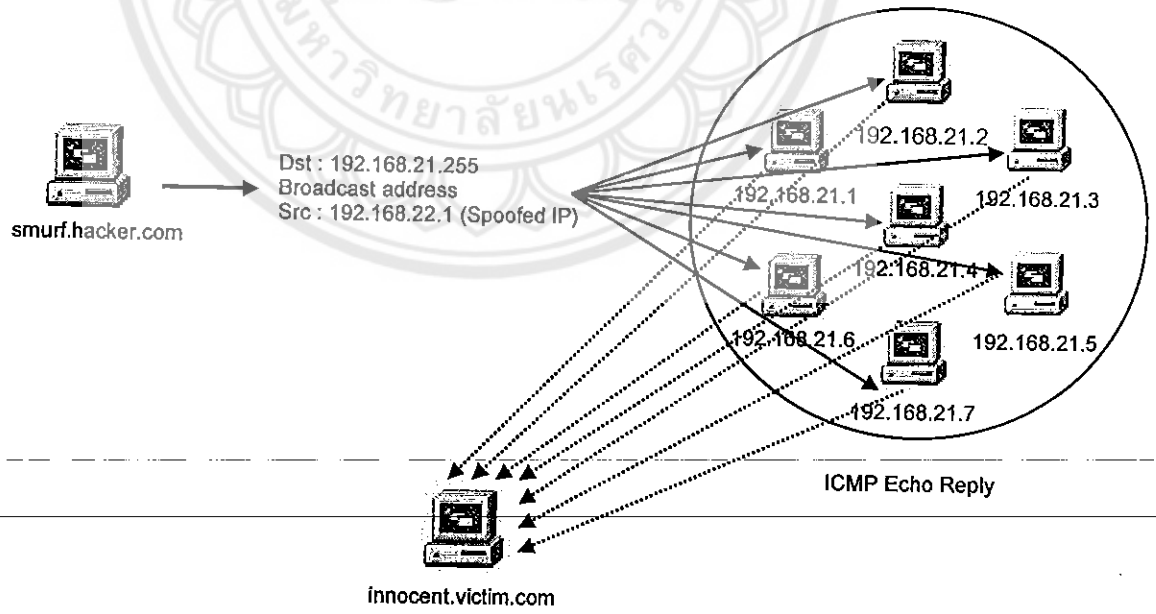
ถ้าเป็นการรับแฟรกเมนต์แพ็คเก็ตตามปกติแล้ว IP จะนำแฟรกเมนต์แต่ละแพ็คเก็ตที่ได้รับมาวางรอไว้ในหน่วยความจำตามตำแหน่งที่ระบุในฟิลด์ offset ตามลำดับการมาถึงของแพ็คเก็ต จนกระทั่งทุกแฟรกเมนต์ได้มาถึงครบหมดจึงรวมกลับมาเป็นดาต้าแกรมที่สมบูรณ์ หรือหากเกินเวลาที่กำหนดแล้วแฟรกเมนต์ยังเดินทางมาไม่ครบ IP ก็จะแจ้งข้อผิดพลาดกลับไปพร้อมทั้งยกเลิกการรับดาต้าแกรมนั้น

การโจมตีของ Teardrop จะใช้การเหลื่อมกันของแพ็คเก็ตในระหว่างที่มีการรวมแฟรกเมนต์แพ็คเก็ตเข้าด้วยกัน เนื่องจากในการแฟรกเมนต์ตามปกติแล้วแพ็คเก็ตจะถูกแบ่งออกเป็นส่วนย่อย แต่สามารถนำมารวมกันใหม่ได้พอดี และตำแหน่งจะถูกต้องสอดคล้องกันเสมอ แพ็คเก็ตที่ใช้ในการโจมตีของ Teardrop จะเป็นแพ็คเก็ตที่ถูกสร้างขึ้นมาโดยเฉพาะไม่ได้ผ่านกลไกการแฟรกเมนต์ตามปกติของ IP โดยมีการระบุ offset ที่เหลื่อมเข้าไปในแพ็คเก็ตอื่นๆ ซึ่งจะไม่มีโอกาสเกิดขึ้นได้เลยในการทำงานปกติ ดังนั้นถ้าระบบปฏิบัติการไม่สามารถจัดการเงื่อนไขไม่ปกติเช่นนี้ได้ดีเพียงพอก็จะทำให้ระบบ ปฏิบัติการหยุดทำงานลงได้



รูปที่ 2.37 แพ็คเก็ตของ Teardrop Attack

2.6.1.6 Smurf Attack



รูปที่ 2.38 การโจมตีแบบ Smurf Attack

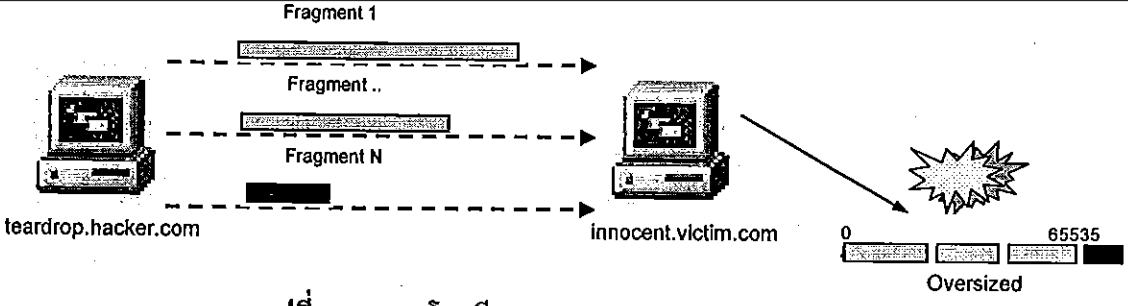
ลักษณะการโจมตี

- ใช้ ICMP Echo Reply เป็นกลไกในการโจมตี
- ใช้ร่วมกับเทคนิคการปลอมหมายเลข IP
- ส่ง ICMP Echo Request ไปยัง Broadcast Address

Smurf Attack เป็นการปรับปรุงเทคนิคของการ Flood เครือข่ายให้ฉลาดกว่าเดิม แทนที่จะอาศัยแบนด์วิดธ์เพื่อโจมตีเป้าหมายอย่างเดียวแบบ Ping Flood, Smurf ได้ค้นพบเทคนิคขยายการโจมตีทำให้แฮคเกอร์มีเครื่องทุ่นแรงและเปิดโอกาสให้ผู้ที่มีแบนด์วิดธ์ต่ำสามารถทำการ Flood ไปยังเป้าหมายได้รุนแรงที่เดียว

ด้วยข้อกำหนดในโปรโตคอล ICMP เมื่อได้รับ ICMP Echo Request แล้วจะต้องส่ง ICMP Echo Reply กลับไปยังหมายเลข IP ของผู้ที่ส่ง ICMP Echo Request มา อย่างไรก็ตามในข้อกำหนดของIP จะมี IP Address ที่ใช้เป็นของบรอดคาสต์สำหรับเครือข่ายนั้น ซึ่งมีไว้เพื่อใช้ในกรณีที่ต้องการส่งข่าวสารถึงทุกๆ โฮสต์ในเครือข่าย คือเมื่อส่งแพ็คเก็ตใดไปยังบรอดคาสต์แอดเดรส จะทำให้ทุกโฮสต์ ได้รับแพ็คเก็ตนั้นอย่างทั่วถึง ดังนั้นเมื่อมีโฮสต์ใดที่ส่ง ICMP Echo Request มายังบรอดคาสต์ก็จะทำให้ทุกโฮสต์ทั้งหมดที่อยู่ในเครือข่ายนั้นได้รับ ICMP Echo พร้อมกัน และจะตอบกลับด้วย ICMP Echo Reply ไปยังผู้ส่งเสมอ และตอบกลับเท่ากับจำนวนเครื่องที่อยู่ในเครือข่ายนั้น ปรากฏการณ์เช่นนี้เรียกว่า การขยายสัญญาณ (Amplification) อัตราการขยายขึ้นอยู่กับปริมาณโฮสต์ที่อยู่ในเครือข่ายขณะนั้น การโจมตีเริ่มด้วยการที่แฮคเกอร์จะส่ง ICMP Echo Request ไปยังบรอดคาสต์แอดเดรสทำให้โฮสต์ทั้งหมดในเครือข่ายขณะนั้นทำการตอบกลับมายัง IP Address ที่ระบุว่าเป็นของผู้ส่ง ดังนั้นวิธีการโจมตีก็ต้องใช้ควบคู่ไปกับการปลอม IP Address โดยให้ IP Address ต้นทางของ ICMP Echo Request ที่ส่งไปนั้นเป็นIP Address ของเป้าหมายที่เราจะโจมตี ทำให้เป้าหมายได้รับ ICMP Echo Reply จำนวนมากจนไม่สามารถสื่อสารกับผู้อื่นบนเครือข่ายได้หรือถึงกับหยุดทำงานไปเลย ทั้งนี้เทคนิคของ Smurf จะอาศัย การบรอดคาสต์ไปยังทุกโฮสต์ ดังนั้นอาจใช้ไม่ได้ผลดีนักกับเครือข่ายที่ใช้เราเตอร์ เพราะสามารถจำกัดขอบเขตการบรอดคาสต์ได้โดยกำหนดที่ Access Control List (ACL) ของเราเตอร์

2.6.1.7 Ping of Death Attack

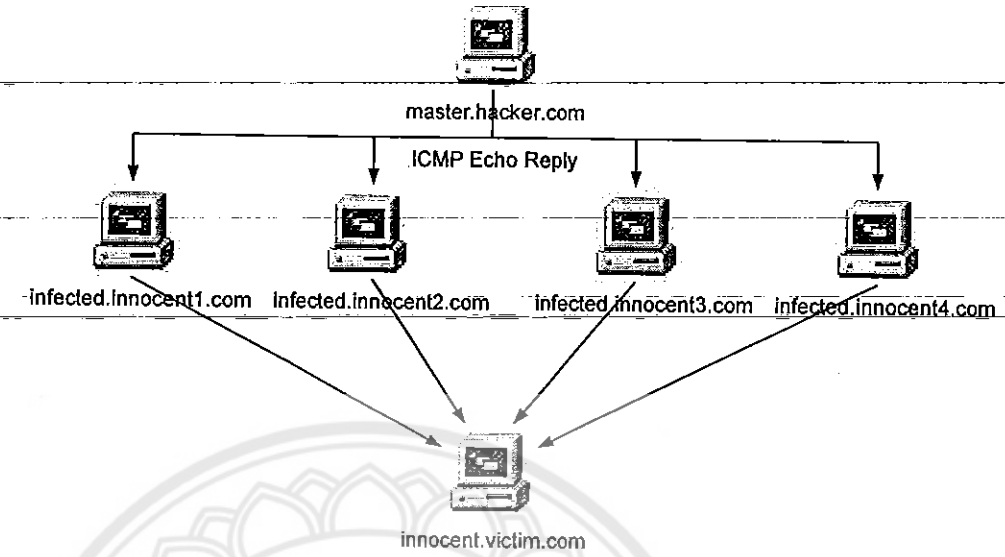


รูปที่ 2.39 การโจมตีแบบ Ping of Death Attack

การโจมตีนี้เป็นวิธีหนึ่งที่ใช้ข้อบกพร่องของการแฟรกเมนต์มาเป็นช่องทางในการโจมตี โดยปกติแล้ว IP Datagram จะมีขนาดสูงสุดได้ไม่เกิน 65535 ไบต์ เพราะถูกจำกัดด้วยขนาดของฟิลด์ที่มีขนาดเพียง 4 ไบต์เท่านั้น แต่ IP ยังมีการแฟรกเมนต์ที่สามารถนำหลายๆแพ็คเก็ตมาต่อรวมเป็นดาต้าแกรมเดียวได้ ซึ่งเป็นช่องทางที่สามารถลบข้อจำกัดของขนาดของดาต้าแกรมอันเนื่องมาจากขนาดของฟิลด์ได้ เพราะผลรวมของขนาดดาต้าแกรมจะต้องเป็นผลรวมของขนาดของแฟรกเมนต์ทั้งหมดรวมกัน และขนาดของแต่ละแฟรกเมนต์ก็สามารถมีได้ถึง 64 K ดังนั้นเมื่อนำแฟรกเมนต์หลายแฟรกเมนต์มารวมเป็นดาต้าแกรมเดียวก็มีโอกาสที่จะทำให้ขนาดของดาต้าแกรมทั้งหมดสูงกว่า 65535 ไบต์ได้

การโจมตีของ Ping of Death จะมุ่งไปที่ระบบปฏิบัติการที่ไม่ได้มีจัดการแฟรกเมนต์อย่างรัดกุม ด้วยพื้นฐานที่ว่าถ้ามีการจัดสรรหน่วยความจำไว้สูงสุด 64 K เพื่อรองรับ 1 ดาต้าแกรม และถ้าระบบปฏิบัติการไม่มีกลไกการตรวจสอบที่รอบคอบแล้ว การรีแอสแซมเบิลของแฟรกเมนต์ก็เป็นเพียงการนำข้อมูลไปใส่ในหน่วยความจำตามตำแหน่งที่ระบุในตัวแฟรกเมนต์ โดยหวังว่าเมื่อทุกแฟรกเมนต์ถูกนำไปใส่ในหน่วยความจำครบถ้วนแล้วจะได้ดาต้าแกรมที่สมบูรณ์ออกมาโดยอัตโนมัติ Ping of Death จึงพยายามหลอกล่อกระบวนการนี้โดยการส่งแฟรกเมนต์ของ ICMP Echo Request ไปยังเป้าหมายเหมือนการใช้คำสั่ง Ping แต่ตั้งใจทำให้ผลรวมของแฟรกเมนต์เกินกว่าขนาด 64 K ถ้าระบบปฏิบัติการไม่มีการตรวจสอบที่ดี แล้วนำข้อมูลในแฟรกเมนต์ไปใส่ในหน่วยความจำตามปกติ ข้อมูลก็จะล้นออกนอกหน่วยความจำที่จัดสรรไว้ (Overflow) ข้อมูลที่ส่วนเกินไปจะตกในตำแหน่งของโปรแกรมอื่นที่ใช้งานอยู่ในโฮสต์นั้น ซึ่งจะทำให้โปรแกรมทำงานผิดเพี้ยนไปหมด และถ้าข้อมูลส่วนเกินไปทับหน่วยความจำของระบบปฏิบัติการจะทำให้โฮสต์นั้นหยุดทำงานไปในทันที (Crash)

2.6.1.8 Tribe Flood Network



รูปที่ 2.40 การโจมตีแบบ Tribe Flood Network

Tribe Flood Network (TFN) เป็นการโจมตีที่รุนแรงและซับซ้อนกว่าเดิม โดยโจมตีเป้าหมายพร้อมกันด้วยหลายโฮสต์ TFN เป็นวิธีการโจมตีที่ใช้ ICMP เป็นคำสั่งสำหรับสั่งงานโฮสต์หลายๆ โฮสต์ให้ทำการโจมตีเป้าหมายพร้อมกัน มีวิวัฒนาการในการใช้โฮสต์จำนวนมากร่วมมือกันและเป็นรุ่นแรกของการโจมตีต่อต้านการทำงานแบบกระจาย DDoS (Distributed Denial of Services) แต่ในระยะหลังการโจมตีดังกล่าวไม่ค่อยสร้างความเสียหายได้มากนัก เนื่องจาก

1. การจัดการ TCP Stack ของระบบปฏิบัติการต่างๆ ได้ถูกปรับปรุงให้มีประสิทธิภาพและเสถียรภาพมากยิ่งขึ้น ดังนั้นการที่ระบบปฏิบัติการจะหยุดทำงานเนื่องจากการรับแพ็คเก็ตมากเกินไปจึงลดลง
2. แบนด์วิดธ์ของเครือข่ายที่ไปยังเครื่องแม่ข่ายเป้าหมายมีขนาดใหญ่ขึ้นกว่าเดิมมาก เนื่องจากมีการปรับปรุงพัฒนาของฮาร์ดแวร์และอุปกรณ์บนเครือข่าย การโจมตีโดยที่จะทำให้เครือข่ายเต็มไปด้วยแพ็คเก็ตนั้นจึงเป็นไปได้ยาก

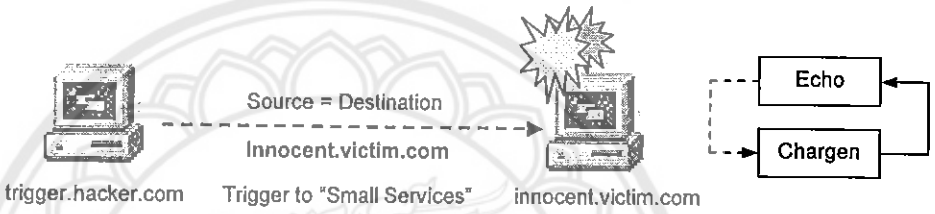
เพื่อแก้ปัญหาดังกล่าวจึงมีผู้พัฒนาเทคนิคของ TFN ขึ้นมาใช้ในการโจมตี โดยมีจุดประสงค์หลัก คือโจมตีโดยใช้เทคนิคของการทำเครือข่ายให้เต็ม (Flood) แต่เปลี่ยนจากการใช้เครื่องของแฮคเกอร์เพียงเครื่องเดียวในการส่งแพ็คเก็ตเป็นการใช้เครื่องจำนวนมากส่งมาโจมตีเป้าหมายเดียวกันพร้อมกัน ซึ่งจะทำให้ความรุนแรงของการโจมตีเพิ่มขึ้นเท่ากับจำนวนเครื่องของผู้ร่วมโจมตี

การที่ TFN สร้างความเสียหายได้มากกว่าการโจมตีแบบ Flood ทั่วไป เพราะว่านอกจากการที่มีเครื่องจำนวนมากโจมตีพร้อมกันแล้ว การมี TFN Daemon เป็นการเปิดช่องทางให้แฮคเกอร์

ที่เป็น TFN Master ซึ่งมีแบนด์วิดธ์ต่ำสามารถสั่งงาน TFN Daemon ที่อยู่ใน LAN Segment เดียวกับเป้าหมายและมีแบนด์วิดธ์สูงมากทำการโจมตีแทนตนเองได้ แบนด์วิดธ์จึงไม่เป็นอุปสรรคในการโจมตีแต่อย่างใด

ดังนั้นการตรวจสอบและการดักจับการโจมตีแบบ DDoS ประเภทเดียวกับ TFN จึงต้องมีการตรวจจับทั้งในส่วนของโฮสต์ที่ตกเป็นเป้าหมายของการโจมตี เพื่อจะได้ติดตามได้ว่าถูกโจมตีจาก Daemon ใด และหลังจากนั้นก็ทำการติดตามไปยัง Daemon อีกครั้งหนึ่งเพื่อหาว่า Master สั่งการมาจากที่ใด ซึ่งการพบแพ็คเก็ต ICMP Echo Reply แปลกๆ โดยที่ไม่มี ICMP Echo Request อาจจะเป็นที่มาของ TFN บนโฮสต์ใดโฮสต์หนึ่งบนเครือข่ายก็เป็นได้

2.6.1.9 Diagnostic Port Attack



รูปที่ 2.41 การโจมตีแบบ Diagnostic Port Attack

ในตอนออกแบบ TCP/IP สมัยแรกๆนั้นได้มีการใส่คุณสมบัติและบริการหลายอย่างใน TCP/IP เพื่อใช้ในการตรวจสอบสถานะการเชื่อมต่อ บริการตรวจสอบเวลา บริการเหล่านี้เรียกว่า "Small Service" ในแต่ละบริการมีรายละเอียดดังนี้

ชื่อบริการ	หมายเลขพอร์ต	รายละเอียดการให้บริการ
Echo	7/UDP 7/TCP	เครื่องแม่ข่ายจะทำการตอบกลับด้วยข้อมูลเดียวกับที่ได้รับมาจากเครื่องลูกข่าย
Discard	9/UDP 9/TCP	เครื่องแม่ข่ายจะไม่ตอบรับข้อมูลใดๆที่ส่งมาจากเครื่องลูกข่าย
Daytime	13/UDP 13/TCP	เครื่องแม่ข่ายจะตอบเวลาและวันที่ กลับไปยังเครื่องลูกข่ายในรูปแบบที่สามารถอ่านได้เข้าใจ
Chargen	19/UDP	เครื่องแม่ข่ายจะตอบกลับไปยังเครื่องลูกข่ายด้วย ASCII Character อย่างต่อเนื่องจนกว่าเครื่องลูกข่ายจะยุติการติดต่อ
Time	34/UDP 34/TCP	เครื่องแม่ข่ายจะตอบค่าเวลาในรูปแบบของ 32 บิต

ตารางที่ 2.1 บริการต่างๆ ของ Small Service

การใช้บริการ	
Echo	ใช้สำหรับตรวจสอบสถานะของเครื่องแม่ข่ายด้วยการส่งแพ็คเก็ตไปที่พอร์ต echo ถ้ามีข้อมูลตอบกลับมาแสดงว่าเครื่องแม่ข่ายยังทำงานอยู่ เป็นเหมือนการ Ping แต่จะใช้ TCP หรือ UDP แทนที่จะใช้ ICMP
Discard	ใช้สำหรับการทดสอบบางชนิด โดยทั่วไป TCP หรือ UDP ถ้าไม่ทำการเปิดพอร์ตให้บริการจะมีการส่ง RST สำหรับ TCP และ ICMP Port Unreachable กลับไปยังเครื่องลูกข่าย แต่สำหรับพอร์ตของ Discard นี้จะไม่มี Error Message ดังกล่าวแต่เครื่องแม่ข่ายก็ไม่ตอบรับใดๆ
Daytime, Time	ใช้สำหรับตรวจสอบเวลาของเครื่องแม่ข่าย เพียงแต่ข้อมูลจะอยู่คนละรูปแบบ
Chargen	ใช้ในการตรวจสอบการเชื่อมต่อ โดยเฉพาะกรณีที่ต้องการ monitor เครื่องแม่ข่ายตลอดเวลา chargen จะมีประโยชน์เพราะตราบใดที่ยังมีข้อมูลมาจากเครื่องแม่ข่ายแสดงว่าเครื่องแม่ข่ายยังคงทำงานตามปกติ

อย่างไรก็ตาม บริการเหล่านี้นอกจากจะนำไปใช้ให้เกิดประโยชน์ดังกล่าวแล้ว แฮคเกอร์ยังสามารถประยุกต์เอาบริการมาเป็นช่องทางของการ DoS ได้อีกด้วย

ลักษณะการโจมตี

หลักสำคัญที่ถูกนำมาใช้โจมตี คือบริการเหล่านี้จะมีการตอบสนองจากเครื่องแม่ข่ายอย่างใดอย่างหนึ่งทุกครั้งที่มีแพ็คเก็ตเข้ามาที่พอร์ตของตนเอง ไม่ว่าจะเป็น chargen, time, daytime โดยไม่ทำการตรวจสอบที่มาของแพ็คเก็ต ซึ่งเปรียบได้กับแหล่งกำเนิดแพ็คเก็ตชั้นดี เมื่อบริการเหล่านี้ถูกนำมาจับคู่กับ Echo ซึ่งจะสะท้อนข้อมูลกลับไปยังผู้ส่งแล้ว Echo ก็จะสะท้อนข้อมูลกลับมายังบริการเหล่านี้ จึงกลายเป็นการกระตุ้นให้บริการเหล่านี้ผลิตแพ็คเก็ตออกมาอีกและส่งเข้ามายัง Echo อีกครั้ง จากนั้นก็จะเกิดการสะท้อนไปมาของข้อมูลไม่รู้จบจนกว่าโฮสต์จะหยุดทำงาน หรือจนกระทั่งเครือข่ายขัดข้องเนื่องจากมีข้อมูลของ Echo อยู่เต็มแบนด์วิดธ์

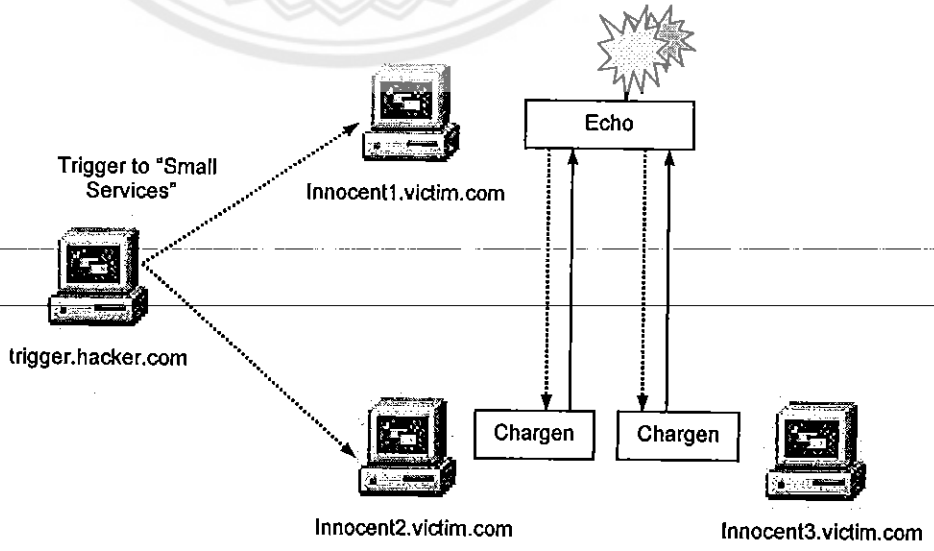
ถึงแม้ว่าจะเป็นการสะท้อนไปมาของข้อมูลระหว่างบริการเล็กๆภายในโฮสต์เอง แต่เนื่องจากการบริการดังกล่าวเกิดขึ้นบนโฮสต์เองด้วยแบนด์วิดธ์สูง Echo ก็จะสะท้อนข้อมูลด้วยความเร็วสูงสุดที่มีด้วยเช่นเดียวกับ Chargen หรือ Daytime ที่จะต้องผลิตแพ็คเก็ตที่ความเร็วสูงสุดเช่นเดียวกัน เมื่อมีการสะท้อนของข้อมูลที่ความเร็วสูงสุดนี้ ระบบปฏิบัติการของโฮสต์ก็จะต้องใช้ทรัพยากรทั้งหมดที่มีมาจัดการกับบริการเหล่านี้ ยิ่งมีการตอบสนองเร็วเท่าไร ระบบปฏิบัติการก็จะหยุดทำงานเร็วเท่านั้น ลักษณะของการโจมตีจะใกล้เคียงกับ Land Attack โดย Land Attack จะอาศัยการสะท้อนกลับไปมาของ TCP แต่การโจมตีแบบนี้จะอาศัยการสะท้อนจากบริการของ UDP แทน

โดยทั่วไปการโจมตีแบบนี้จะใช้ควบคู่ไปกับการปลอมหมายเลข IP ของต้นทางและปลายทาง โดยแฮคเกอร์จะส่งแพ็คเก็ตที่มีต้นทางจากพอร์ต echo ไปยังพอร์ต chargen โดยที่ IP

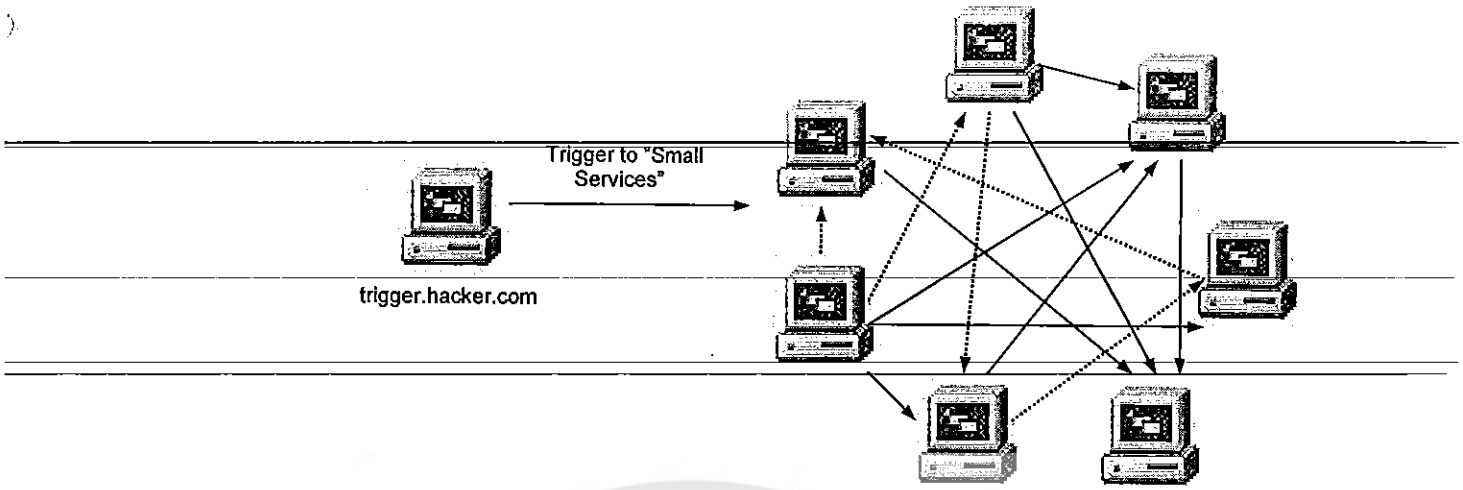
Address ของต้นทางและปลายทางเป็นของโฮสต์เป้าหมายที่จะโจมตี จากนั้นลำดับของการโจมตีจะดำเนินไปโดยอัตโนมัติดังนี้

1. ส่งแพ็คเก็ตจากพอร์ต echo ไปยังพอร์ต chargen โดยที่ต้นทางและปลายทางเป็นโฮสต์เป้าหมาย
2. เมื่อบริการ chargen ได้รับแพ็คเก็ตก็จะเริ่มทำงานโดยการส่ง ASCII character กลับไปยังผู้ที่ส่งแพ็คเก็ตเข้ามาตามหมายเลขพอร์ตที่ระบุไว้ ซึ่งก็คือตอบกลับมาที่เครื่องของตนเองที่พอร์ต echo นั่นเอง
3. เมื่อแพ็คเก็ตจาก chargen มาถึงพอร์ต echo บริการ echo ก็จะทำงานโดยการตอบข้อมูลที่ได้รับกลับไปยังพอร์ต chargen อีก
4. การทำงานของ chargen จะกลับไปยังขั้นตอนที่ 2 และวนเวียนต่อไปไม่รู้จบ ที่โฮสต์เองก็ส่งรับข้อมูลกันระหว่างพอร์ต echo กับ chargen จนในที่สุดก็จะหยุดทำงาน

จะเห็นว่าการปลอมแพ็คเก็ตเพียงแพ็คเก็ตเดียวก็สามารถทำให้โฮสต์หยุดทำงานลงได้ไม่ยากโดยแฮคเกอร์ปลอม IP Address ให้เป็นของเป้าหมายเพื่อเป็นการเริ่มต้นของการทำลายตนเองของเป้าหมาย ซึ่งเพียงแพ็คเก็ตเดียวก็เพียงพอที่จะทำให้กระบวนการทำลายตนเองเกิดขึ้นและทำลายตัวเองได้ อย่างไรก็ตามเพื่อให้การโจมตีมีผลเฉียบพลันและรุนแรงมากขึ้น จึงมีการพัฒนาเทคนิคให้มีการขยายการโจมตี โดยถ้าภายในเครือข่ายนั้นมีโฮสต์อื่นที่เปิดให้บริการเหล่านี้อยู่ด้วยก็จะส่งแพ็คเก็ตปลอมไปหลอกให้โฮสต์ตัวอื่นมาร่วมโจมตีด้วย แต่แทนที่จะส่งเพียงแพ็คเก็ตเดียวก็จะทำการส่งหลายแพ็คเก็ตไปยังพอร์ต chargen ของเครื่องแม่ข่ายตัวอื่นๆที่อยู่ในเครือข่ายด้วย บริการ chargen ของเครื่องแม่ข่ายเหล่านั้นก็จะถูกนำมาใช้ให้ส่งแพ็คเก็ตมายังเป้าหมายด้วย และการตอบไปมาของแพ็คเก็ตก็จะทำให้เครื่องแม่ข่ายเป้าหมายหยุดทำงานได้เร็วขึ้น หรือไม่เครือข่ายก็จะเต็มอย่างรวดเร็ว



รูปที่ 2.42 Diagnostic Port Attack โดยการกระตุ้นให้โฮสต์อื่นมาร่วมโจมตี

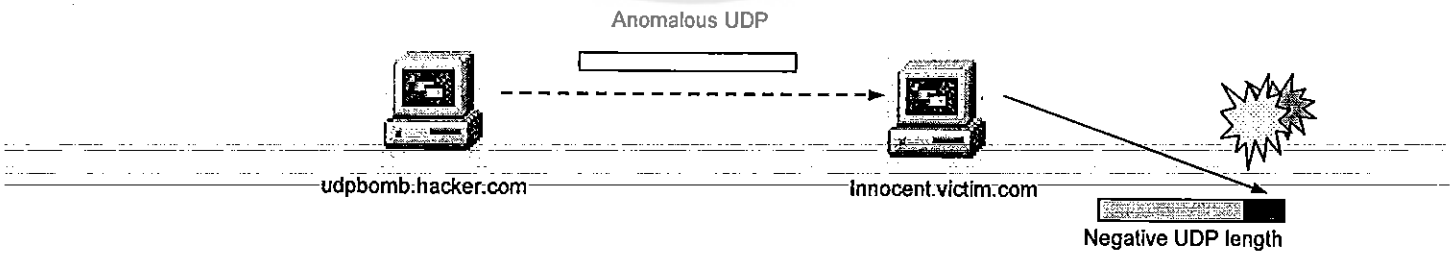


รูปที่ 2.43 Diagnostic Port Attack ทำให้เครือข่ายเต็มไปด้วยแพ็คเก็ต

และนอกจากนี้ถ้าภายในเครือข่ายมีโฮสต์ที่เปิดให้บริการ Small Services อยู่จำนวนมาก การโจมตีของ Diagnostic Port Attack นี้ จะสร้างความวุ่นวายให้เกิดขึ้นภายในเครือข่ายจนถึงขั้นที่เครือข่ายอาจจะไม่สามารถใช้งานได้ เนื่องจากมีแพ็คเก็ตของ Chargen และ Echo ส่งรับกันอยู่เต็มเครือข่าย

ปัจจุบันเครื่องแม่ข่ายส่วนใหญ่มักจะไม่มีบริการเหล่านี้แล้ว เนื่องจากเปราะบางต่อการถูกนำไปใช้ในการโจมตี การโจมตีเครื่องแม่ข่ายด้วยวิธีนี้โดยมากจึงไม่ได้ผล แต่มีอุปกรณ์บางชนิด เช่น เราเตอร์ ยังคงมีบริการเหล่านี้อยู่ในตัว ถ้าผู้ใช้ไม่ปิดบริการนี้บนเราเตอร์ก็อาจถูกโจมตีด้วยวิธีนี้ได้

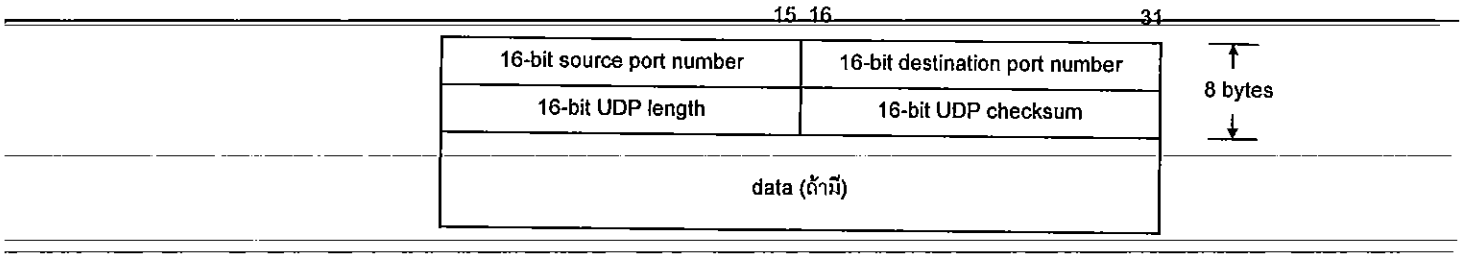
2.6.1.10 UDP Bomb



รูปที่ 2.44 การโจมตีแบบ UDP Bomb Attack

UDP Bomb เป็นการโจมตีโดยอาศัยข้อบกพร่องของ UDP เพื่อทำให้เครื่องแม่ข่ายเป้าหมายรับส่ง UDP Datagram ที่มีขนาดผิดปกติจาก UDP ทั่วๆไป ถ้าระบบปฏิบัติการของเครื่องแม่ข่าย

เป้าหมายไม่สามารถจัดการกับแพ็คเก็ตประเภทนี้ได้ดีก็จะทำให้เครื่องแม่ข่ายเป้าหมายหยุดทำงานได้ทันที

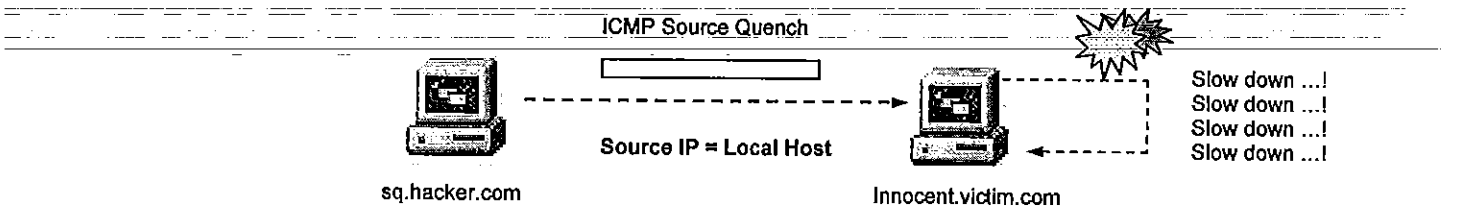


รูปที่ 2.45 UDP Header

ก่อนดูรายละเอียดของ UDP Bomb จะต้องพิจารณารายละเอียดของ UDP Datagram ก่อน ปกติ UDP Datagram จะประกอบด้วย 2 ส่วน คือ UDP Header และ UDP Data ในส่วนของ UDP Header จะประกอบด้วยหมายเลขพอร์ตต้นทาง, หมายเลขพอร์ตปลายทาง, ความยาวของ UDP Datagram และ UDP checksum ไว้คอยตรวจสอบความถูกต้อง จะเห็นว่าถ้า UDP Datagram ไม่มีข้อมูลใดเลย ก็จะต้องมีขนาดอย่างน้อยที่สุด 8 ไบต์ เนื่องจาก UDP Datagram จะต้องมี UDP Header เสมอ

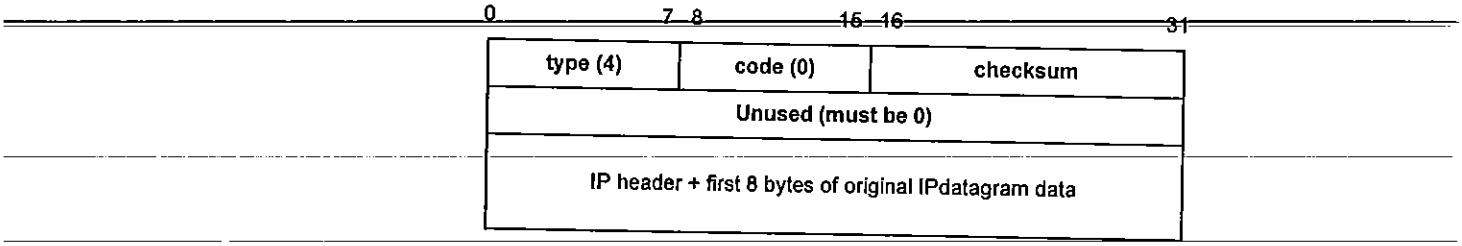
แต่อย่างไรก็ตาม ขนาดความยาวของ UDP Datagram จะต้องถูกระบุอยู่ในเฮดเดอร์ของตัวมันเองเช่นเดียวกับ UDP Bomb ที่ใช้ข้อบกพร่องในการระบุค่าความยาวของ UDP Datagram ลงใน UDP Header ซึ่งเป็นจุดอ่อนในการโจมตี โดยการระบุค่าของ UDP Datalength ที่ต่ำกว่า 8 ไบต์ในแพ็คเก็ตที่ใช้ในการโจมตี ส่วนใหญ่จะระบุเท่ากับ 7 ไบต์ ซึ่งเป็นไปไม่ได้ เงื่อนไขลักษณะนี้จึงเป็นสิ่งที่ถูกมองข้ามไป ทำให้การจัดการกับแพ็คเก็ตประเภทนี้เป็นไปในลักษณะที่ควบคุมไม่ได้ และทำให้เกิดความเสียหายต่อระบบปฏิบัติการ

2.6.1.11 ICMP Source Quench Attack



รูปที่ 2.46 การโจมตีแบบ ICMP Source Quench Attack

ก่อนที่จะรู้จัก ICMP Source Quench Attack นั้นควรทำความรู้จักกับ ICMP Source Quench ก่อน ICMP Source Quench เป็น ICMP Type 4 Code 0 มีโครงสร้างดังรูป

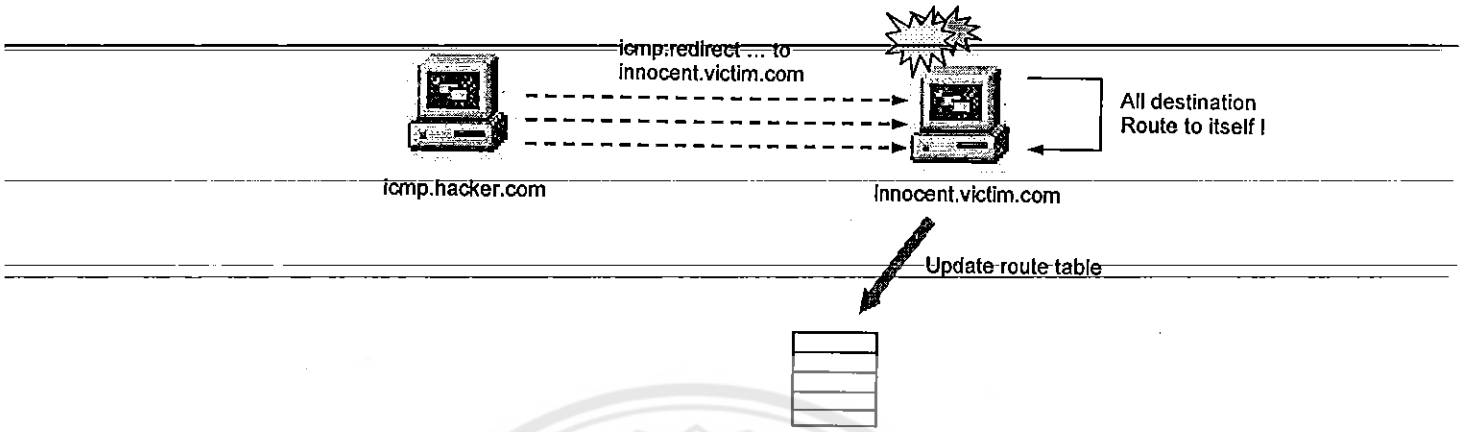


รูปที่ 2.47 ICMP source quench error

ICMP ประเภทนี้ใช้สำหรับควบคุมความเร็วในการรับส่งข้อมูลระหว่างทั้งสองฝั่ง เนื่องจากในบางกรณีแบนด์วิดธ์ของผู้รับและผู้ส่งอาจไม่สมดุลกัน ทำให้การส่งข้อมูลเร็วเกินกว่าที่ผู้รับจะสามารถรับได้ทัน โดยเฉพาะการส่งข้อมูลจากโฮสต์ต้นทางในเครือข่ายปกติที่มีความเร็วสูงไปยังโฮสต์ปลายทางที่เชื่อมต่อกับ WAN โดยผ่านเราเตอร์ซึ่งมีแบนด์วิดธ์จำกัด เราเตอร์จะไม่สามารถส่งแพ็คเก็ตออกไปได้ทัน ดังนั้นเราเตอร์จะต้องทำการพักข้อมูลในบัฟเฟอร์ไว้ก่อน แล้วค่อยทยอยส่งข้อมูลออกไป แต่ขนาดของบัฟเฟอร์ก็มีจำกัด ถ้ามีการส่งแพ็คเก็ตเข้ามาอย่างต่อเนื่องบัฟเฟอร์ที่มีอยู่ก็จะไม่เพียงพอ ซึ่งอาจทำให้แพ็คเก็ตเกิดการสูญหายได้ TCP/IP จึงมีกลไกในการลดปัญหาแพ็คเก็ตที่หายไปเนื่องจากกรณีนี้โดยใช้ ICMP เป็นสัญญาณบอกให้ผู้ส่งลดความเร็วในการส่งข้อมูลลง ICMP ที่ใช้คือ ICMP Source Quench เมื่ออุปกรณ์ TCP/IP ได้รับ ICMP Source Quench ก็จะทำการลดความเร็วในการส่งข้อมูลลง เพื่อให้เราเตอร์ได้ทยอยส่งข้อมูลออกไปจนบัฟเฟอร์ว่างพอที่จะรับข้อมูลใหม่ได้ ปกติ ICMP Source Quench จะเกิดขึ้นกับการสื่อสารของ UDP เท่านั้น เพราะ UDP ไม่มีกระบวนการควบคุมจังหวะในการรับส่งข้อมูลที่แน่นอน ซึ่งกรณีแบบนี้จะไม่เกิดขึ้นกับ TCP เนื่องจาก TCP มีการตรวจสอบทุกขั้นตอนของการรับส่งข้อมูล

นอกจากการแจ้งให้เราเตอร์ลดความเร็วในการส่งข้อมูลแล้ว ICMP Source Quench ได้ถูกนำไปใช้ในหน้าที่อื่นด้วย คือเป็นคำสั่งที่สั่งให้โฮสต์ทำการส่งแพ็คเก็ตทั้งหมดไปยัง IP Address ใหม่ที่กำหนด เพื่อให้มีแบนด์วิดธ์ที่เหมาะสม หมายถึงเมื่อโฮสต์ได้รับข่าวสารนี้แล้วแพ็คเก็ตใดๆที่ส่งไปยังเราเตอร์เดิมก็จะเปลี่ยนเส้นทางไปยังแอดเดรสใหม่ที่ระบุมาใน ICMP Source Quench ดังนั้น แฮคเกอร์จึงใช้ ICMP Source Quench ส่งไปยังเป้าหมาย โดยแจ้งให้โฮสต์เป้าหมายทำการส่งแพ็คเก็ตทั้งหมดไปยังแอดเดรสของลูปแบค (Loopback) คือส่งแพ็คเก็ตทั้งหมดวนเข้าหาตัวเอง พร้อมทั้งแจ้งให้โฮสต์ลดความเร็วของการส่งข้อมูลไปยังลูปแบคด้วย เมื่อเป้าหมายถูกโจมตีก็จะทำให้กระบวนการรับส่งข้อมูลของโฮสต์เป้าหมายทำงานผิดปกติและหยุดทำงานได้

2.6.1.12 Winfreeze



รูปที่ 2.48 การโจมตีแบบ Winfreeze Attack

Winfreeze ใช้ ICMP redirect message เป็นกลไกในการโจมตี โดยปกติแล้ว ICMP redirect message จะใช้ในการแจ้งให้เราเตอร์ตัวอื่นทราบ เมื่อเราเตอร์ตัวนั้นเห็นว่าเราเตอร์ตัวอื่นได้ทำการส่งผ่านแพ็คเก็ตไปในเส้นทางที่ไม่มีประสิทธิภาพ คือแพ็คเก็ตนั้นควรจะถูกส่งไปยังปลายทางโดยผ่านเส้นทางอื่นที่มีระยะทาง (HOP) สั้นกว่า ดังนั้นเพื่อให้เราเตอร์อื่นทราบว่ามีเส้นทางอื่นที่สั้นกว่าในการส่งแพ็คเก็ตไปยังปลายทางโดยการส่ง ICMP redirect message ไปยังเราเตอร์เพื่อทำการปรับปรุงเส้นทางลงในตารางเราต์ติ้ง (routing table)

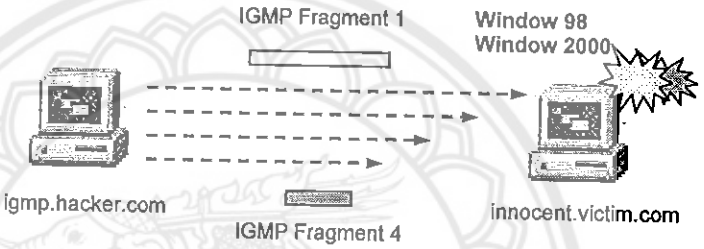
การส่ง ICMP redirect ไปยังเครื่องแม่ข่ายเป้าหมายเป็นจำนวนมากอย่างต่อเนื่อง เมื่อเครื่องแม่ข่ายเป้าหมายได้รับ ICMP ดังกล่าวก็จะพยายามนำข้อมูลที่ส่งมาด้วยเข้าไปปรับปรุงใน routing table ของตนเอง แต่เนื่องจากปริมาณของ ICMP ที่เข้ามามากและเร็วเกินกว่าที่จะนำไปเพิ่มใน routing table ได้ทัน จึงทำให้การปรับปรุง routing table ผิดพลาดได้ ประกอบกับกลไกการจัดสรรหน่วยความจำสำหรับเก็บ routing table ที่ระบบปฏิบัติการเตรียมไว้นั้นน้อยมาก และการปรับปรุง routing table ก็ไม่ได้เกิดขึ้นบ่อย โปรแกรมที่ทำหน้าที่ส่วนนี้จึงไม่ได้รับการออกแบบไว้ให้ทำงานอย่างรอบคอบเท่าที่ควร ดังนั้นเมื่อถูกโจมตีโดยการส่งข้อมูลจำนวนมากเข้ามายัง routing table จนเกินกว่าที่ระบบ ปฏิบัติการจะสามารถจัดการได้ทัน ข้อบกพร่องของส่วนนี้จึงลุกลามไปกระทบการทำงานส่วนอื่นของระบบปฏิบัติการด้วย

นอกจากการนำข้อมูลจำนวนมากใส่เข้าไปใน routing table แล้ว Winfreeze ยังส่งผลให้กลการสื่อสารข้อมูลผิดปกติ เนื่องจาก routing table ของปลายทางในเครือข่ายต่างๆ ระบุ IP Address ที่ให้ส่งแพ็คเก็ตวนกลับเข้ามายังตนเอง ซึ่งเป็นผลให้แพ็คเก็ตที่ต้องส่งไปยังปลายทางไม่ถูกส่งต่อไป

ยังเราเตอร์ที่ถูกต้อง และโฮสต์นั้นก็ไม่มีโอกาสสื่อสารกับโฮสต์อื่นที่อยู่คนละเครือข่ายได้เลย เพราะ IP Address ของเราเตอร์นั้นเกิดความผิดพลาด

เนื่องจากการ DoS ชนิดนี้ได้ผลเป็นอย่างดีในการทำให้ระบบปฏิบัติการตระกูล Windows หยุดทำงานจึงได้ชื่อว่า Winfreeze เมื่อแพ็คเก็ตที่ใช้ในการโจมตีจำนวนมากถูกส่งมายัง Windows NT ระบบปฏิบัติการจะพยายามตอบสนองต่อ ICMP redirect message ตามที่กำหนดไว้โดยพยายามนำข้อมูลใส่ลงใน routing table ทุกครั้งที่ได้รับเข้ามา จนเมื่อปริมาณแพ็คเก็ตมากเกินไป เครื่องก็จะทำงานช้าลงเรื่อยๆ และหยุดทำงานไปในที่สุด

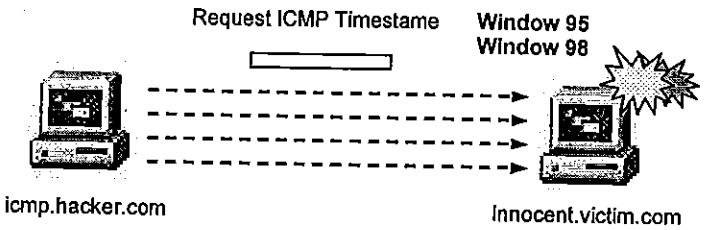
2.6.1.13 Fragmented IGMP Attack



รูปที่ 2.49 การโจมตีแบบ Fragmented IGMP Attack

วิธีการนี้ใช้ได้ผลเนื่องจากข้อผิดพลาดภายในระบบปฏิบัติการของ Microsoft Windows เอง ทั้ง Windows 98 และ Windows 2000 โดยการส่งชุดของแฟรกเมนต์แพ็คเก็ตของ IGMP ตามข้อมูลที่ระบุไว้ จะทำให้ระบบปฏิบัติการหยุดทำงานและปรากฏจอภาพสีฟ้า (Blue Screen) ทำให้ผู้บริหารระบบเข้าใจผิดว่าเกิดความผิดปกติของฮาร์ดแวร์ ถ้าระบบปฏิบัติการที่ใช้งานไม่ได้รับการปรับปรุงด้วยโปรแกรมแก้ไข (Patch) แล้ว เครื่องแม่ข่ายนั้นจะไม่สามารถเปิดให้บริการบนอินเตอร์เน็ตได้อย่างมีเสถียรภาพ และทำให้มีความเสี่ยงที่จะหยุดทำงานได้อีก

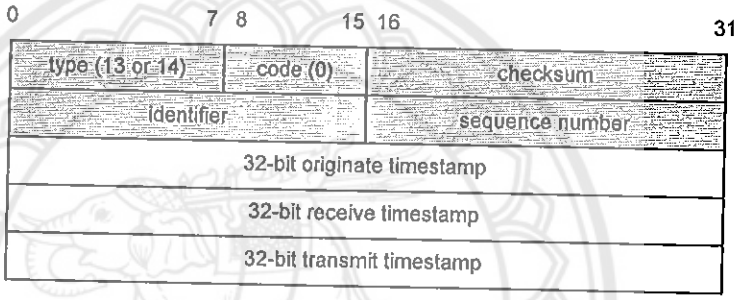
2.6.1.14 ICMP Timestamp Attack



รูปที่ 2.50 การโจมตีแบบ ICMP Timestamp Attack

ICMP Timestamp Attack เป็นการโจมตีที่มีเป้าหมายไปยังระบบปฏิบัติการของ Microsoft Windows คือ Windows 95 และ Window 98 โดยใช้การส่ง ICMP Timestamp Request จำนวนมากและรวดเร็วไปยังเป้าหมาย ก็จะทำให้ทำให้เครื่องเป้าหมายหยุดการทำงานและหน้าจอกลายเป็นสีฟ้าได้

ปกติหน้าที่ของ ICMP Timestamp Request คือการถามเวลาจากโฮสต์ปลายทาง โดยโฮสต์ต้นทางหรือผู้ส่ง ICMP Timestamp Request จะบันทึกเวลาของตนเองลงในฟิลด์เวลาต้นทางแล้วส่งแพ็คเก็ตนี้ไปยังโฮสต์ปลายทาง เมื่อโฮสต์ปลายทางได้รับ ICMP นี้แล้วก็จะทำการตอบกลับมาพร้อมทั้งบันทึกเวลาที่ได้รับแพ็คเก็ตตอบกลับไป มีประโยชน์ในการตรวจสอบเวลาในการเดินทางของแพ็คเก็ตระหว่างจุดต่อจุด โดยเปรียบเทียบเวลาของต้นทางและปลายทางที่บันทึกอยู่ในแพ็คเก็ต



รูปที่ 2.51 ICMP Timestamp Request and Reply

ลำดับการส่ง-รับของ ICMP Timestamp Request and Reply เป็นดังนี้

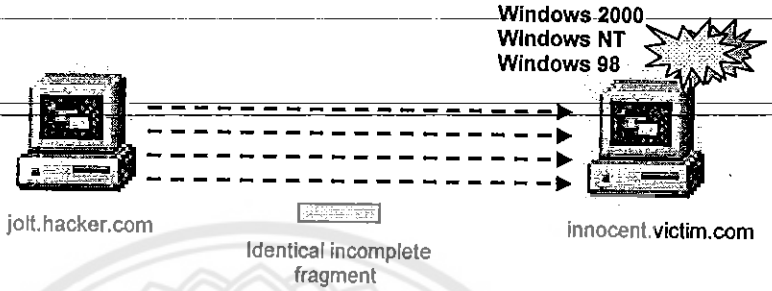
- ผู้ส่งจะบันทึกเวลาที่เริ่มส่งในฟิลด์ของ Originate timestamp ในหน่วยของมิลลิวินาทีของเวลามาตรฐาน (Coordinate University Time) กำหนด type เท่ากับ 13 และระบุหมายเลขการขอลงในฟิลด์ Identifier แล้วจึงส่งออกไปยังปลายทาง
- ผู้รับเมื่อได้รับ ICMP Timestamp Request เข้ามาก็จะตอบกลับด้วย ICMP type = 14 โดยบันทึกเวลาที่ได้รับแพ็คเก็ตลงในฟิลด์ receive timestamp และเวลาที่โฮสต์ทำการส่งแพ็คเก็ตตอบกลับลงในฟิลด์ transmit timestamp แล้วส่ง Reply กลับไป

ด้วยค่าเวลาที่อยู่ในฟิลด์ของ Originate, Receive, Transmit Timestamp จะสามารถนำมาคำนวณหาเวลาในการเดินทางของแพ็คเก็ตระหว่างโฮสต์ต้นทางและโฮสต์ปลายทางได้ และสามารถนำไปวิเคราะห์ประสิทธิภาพของเครือข่ายได้ด้วย

แต่แฮคเกอร์ไม่ได้นำ ICMP นี้มาเพื่อใช้ให้เกิดประโยชน์แต่กลับนำไปเป็นเครื่องมือก่อกวนการทำงานของเป้าหมาย เพราะในการตอบรับของ ICMP Timestamp นี้โฮสต์ปลายทางจะต้องเสียเวลาส่วนหนึ่งในการประมวลผลเพื่อตรวจสอบเวลาในระดับความละเอียดเป็นมิลลิวินาที เพื่อนำข้อมูลไปใส่ลงใน Receive, Transmit Timestamp ทุกครั้งก่อนจะทำการตอบ

กลับไป แฮคเกอร์ก็จะส่ง Request เข้ามาจำนวนมากภายในเวลารวดเร็วอย่างต่อเนื่องโดยไม่สนใจการตอบรับของเป้าหมายแม้แต่น้อย ในส่วนของโฮสต์ที่ถูกโจมตีจะไม่มีทางที่จะตรวจสอบได้ว่าการ Request เข้ามานั้นถูกต้องหรือเหมาะสมหรือไม่ และยังคงพยายามตอบกลับตามโปรโตคอลอย่างเต็มความสามารถ จนกระทั่งเกินกำลังที่จะประมวลผลก็จะหยุดทำงานลง

2.6.1.15 Jolt



CVE -2000-0305
Description : Windows 95, Windows 98, Windows 2000, Windows NT 4.0, and Terminal Server systems allow a remote attacker to cause a denial of service by sending a large number of identical fragmented IP packets, aka jolt2 or the "IP Fragment Reassembly" Vulnerability.

รูปที่ 2.52 การโจมตีแบบ Jolt Attack

เป็นที่สังเกตได้ว่าระบบปฏิบัติการของ Microsoft จะอ่อนไหวต่อแฟรกเมนต์แพ็กเก็ตเป็นพิเศษ การDoS สำหรับ Windows ส่วนใหญ่สามารถทำได้ง่าย โดยอาศัยแฟรกเมนต์ ซึ่งส่งผลเสียต่อเป้าหมายได้ค่อนข้างมาก

ส่วนใหญ่เมื่อเครื่องแม่ข่ายถูกโจมตีด้วยแฟรกเมนต์แล้วจะปรากฏอาการ คือ

- หยุดการทำงานและแสดงหน้าจอสีฟ้า (Blue Screen) ต้องทำการ Cold boot เครื่องใหม่ หรืออาจจะต้องปิดเครื่องแล้วเปิดใหม่
- หยุดการทำงานเครื่องชั่วขณะ เครื่องแม่ข่ายไม่ตอบสนองใดๆ เหมือนกับแฮงค์ ผู้ใช้จะไม่สามารถควบคุมหรือติดต่อกับระบบปฏิบัติการได้ ส่วนใหญ่ผู้ใช้ต้องรีเซตและปิดเครื่องแล้วเปิดใหม่เช่นเดียวกัน
- ทำงานช้าลงไปมาก ตอบสนองต่อผู้ใช้ช้ามาก เนื่องจาก CPU จะถูกนำไปใช้ในการจัดการกับแฟรกเมนต์ที่เข้ามา และถ้าโดนโจมตีมากๆ เครื่องก็จะไม่ตอบสนองอีกเลย

Jolt เป็นการโจมตีโดยอาศัยแฟรกเมนต์ทั่วไป แฮคเกอร์จะอาศัยการส่งแฟรกเมนต์แพ็คเก็ตซ้ำๆจำนวนมากอย่างต่อเนื่องมายังเครื่องแม่ข่ายเป้าหมาย ถ้าข้อมูลมีความเร็วและปริมาณไม่มากพอก็จะเพียงแต่ทำงานช้าลงกว่าปกติ ซึ่งถ้าแฮคเกอร์เพิ่มความเร็วในการส่งข้อมูลเครื่องก็จะหยุดทำงาน

และไม่ตอบสนองต่อผู้ใช้อีกเลย การใช้งานCPU ของเครื่องแม่ข่ายจะขึ้นสูงถึง 100% และถูกใช้ไปโดย Kernel ของระบบ ปฏิบัติการ ซึ่งเป็นเหตุผลที่ทำให้ไม่มีการตอบสนองใดๆ ต่อผู้ใช้ เนื่องจากกำลังของ CPU ถูกใช้ไปในการจัดการกับแฟรกเมนต์ทั้งหมด

2.7 ไอพีเทเบิล (IPtables)

IPtables เป็นเครื่องมือ Built-In ของลีนุกซ์ที่มีเคอร์เนลตั้งแต่ 2.4 เป็นต้นไป ใช้สำหรับสร้าง ควบคุม และตรวจสอบกฎเกณฑ์ในการกรองข้อมูลที่ผ่านไปมาระหว่างเครือข่ายภายนอกและเครือข่ายภายใน และช่วยป้องกันเครือข่ายและข้อมูลสำคัญๆ จากผู้ที่ไม่ได้รับอนุญาตที่แอบเข้ามาในระบบเครือข่าย

หน้าที่หลักที่สำคัญของ IPtables ที่ต้องทำ คือ

- ให้สิทธิข้อมูลที่ ได้รับอนุญาตให้เข้ามาถึงที่หมายได้และปฏิเสธข้อมูลที่ไม่ได้รับอนุญาตไม่ให้เข้ามาในเครือข่ายหรือเครื่องคอมพิวเตอร์ภายในเครือข่าย
- ให้บริการในลักษณะของยามรักษาการอิเล็กทรอนิกส์ที่คอยหยุดความพยายามของแฮคเกอร์หรือบุคคลใดก็ตามที่พยายามเจาะเข้ามายังเครือข่ายของเราจากเครื่องคอมพิวเตอร์ที่ไม่รู้จัก (Untrusted Computer) บนเครือข่ายภายนอก

2.7.1 CHAINS

Chain แบ่งออกเป็น 4 chain ดังนี้

- **INPUT**

  ใช้ตรวจสอบแพ็คเก็ตที่มีแอดเดรสปลายทางอยู่ที่เครื่องไฟร์วอลล์ (แพ็คเก็ตที่ถูกส่งมาให้ไฟร์วอลล์)

- **OUTPUT**

  ใช้ตรวจสอบแพ็คเก็ตที่มีแอดเดรสต้นทางเป็นเครื่องไฟร์วอลล์ (แพ็คเก็ตที่ไฟร์วอลล์ส่งออกมา)

- **FORWARD**

  ใช้ตรวจสอบแพ็คเก็ตที่ไม่มีแอดเดรสต้นทางหรือปลายทางตรงกับเครื่องไฟร์วอลล์ (แพ็คเก็ตที่ส่งผ่านไฟร์วอลล์)

- **User-Defined chaing**

  เป็นกลุ่มของกฎเกณฑ์ที่ผู้ใช้เป็นผู้กำหนดขึ้นเอง

โดยที่แต่ละกลุ่มของกฎเกณฑ์นั้นจะถูกนำมาใช้แยกกัน ถ้าตรวจสอบแล้วไม่ตรงกับกฎแรกก็จะเลื่อนไปตรวจสอบการทำงานกับกฎต่อๆไป และถ้าไม่ตรงกับกฎใดๆ เลย ก็จะไปตรวจสอบที่นโยบาย (Policy) ของ chain นั้นๆ ซึ่งโดยทั่วไปแล้วจะเป็นการดร็อปแพ็คเก็ตนั้นทิ้งไป

หมายเหตุ INPUT, OUTPUT และ FORWARD ถือเป็น Built-In chain

2.7.2 TABLES

-t , –table ใช้สำหรับระบุ packet matching table โดยที่แบ่งออกเป็น 3 table ดังนี้

- **filter**

  เป็น default table ประกอบด้วย Built-In chain คือ INPUT, OUTPUT และ FORWARD

- **nat**

  เป็น table ที่ใช้สำหรับการทำ Network Address Translation (NAT) ประกอบด้วย Built-In chain คือ PREROUTING, POSTROUTING และ OUTPUT

- **mangle**

  เป็น table ที่ใช้สำหรับการเปลี่ยนแปลงแพ็คเก็ตแบบพิเศษ ประกอบด้วย Built-In chain คือ PREROUTING และ OUTPUT

2.7.3 TARGETS (สำหรับ filter table)

การตรวจสอบแพ็คเก็ต iptables จะนำรายละเอียดของแพ็คเก็ตไปตรวจสอบกับกฎต่างๆ ที่กำหนดไว้ตามลำดับ โดยถ้าไม่ตรงกับกฎนั้น ก็จะเลื่อนไปตรวจสอบกับกฎในลำดับต่อไป แต่ถ้ารายละเอียดของแพ็คเก็ตตรงกับกฎนั้น iptables จะปฏิบัติกับแพ็คเก็ตนั้นตามที่ระบุไว้ในส่วนของ Target

Target สามารถระบุได้หลายอย่าง ทั้งแบบที่กำหนดเอง (User-Defined chain) หรือเลือกเอาอย่างใดอย่างหนึ่งใน Built-In chain ต่อไปนี้

- **ACCEPT**

  ส่งผ่านแพ็คเก็ตนั้นต่อไปยังที่ระบุไว้ในแอดเดรสปลายทาง

- **DROP**

  ทิ้ง (ลบ) แพ็คเก็ตนั้นออกจากเครือข่าย

- **REJECT**

  ทิ้งแพ็คเก็ตเช่นเดียวกับ DROP แต่จะมีการส่งข้อความกลับไปที่โฮสต์ต้นทาง เพื่อให้โฮสต์ต้นทางทราบว่าแพ็คเก็ตที่ส่งมานั้นโดนลบทิ้ง

- **QUEUE**

  เป็น Target พิเศษ ใช้สำหรับส่งแพ็คเก็ตไปรอ Userspace Processing นอกจากว่าจะมีบางโปรแกรมกำลังรอแพ็คเก็ตนั้นอยู่ แพ็คเก็ตนั้นก็จะถูกดรอปทิ้งเลย

- **RETURN**

ถ้าเป็น Built-In chain เช่น iptables –A FORWARD –p tcp –j RETURN จะปฏิบัติต่อแพ็คเก็ตเหมือนกับว่าไม่มีกฎใดที่ตรงเลย นั่นคือจะไปใช้ policy ของ chain แทน

ถ้าเป็น User-Defined chain เช่น iptables –A my-new-chain –p tcp –j RETURN จะส่งแพ็คเก็ตกลับไปตรวจสอบกับกฎต่อไปใน chain ก่อนหน้านี้แทน

- **LOG**

บันทึกข้อมูลลง log

2.7.4 OPTIONS

• **COMMANDS (สำหรับจัดการกับ chain)**

-h , --help

ใช้ดูคำอธิบายและโครงสร้างของคำสั่งต่างๆ

Example: iptables -h

-N, --new-chain

สร้าง chain ใหม่ (User-Defined chain) ขึ้นมา

Example: iptables –N my-new-chain

-X, --delete-chain

ลบ chain ที่สร้างขึ้นเอง (User-Defined chain) ซึ่งไม่มีกฎเกณฑ์ใดๆ อยู่เลยและต้องไม่เป็น Target ของกฎเกณฑ์ใดๆ ด้วย (ไม่สามารถลบ Built-In chain ได้)

Example: iptables –X my-new-chain # ลบ my-new-chain
iptables –X # ลบ chain ทั้งหมดที่ถูกสร้างขึ้นและสามารถลบได้

-E, --rename-chain

เปลี่ยนชื่อของ chain

Example: iptables –E oldname newname

-P, --policy

เปลี่ยน policy ของ chain ในกรณีที่รายละเอียดของแพ็คเก็ตไม่ตรงกับกฎใดๆ เลย

Example: iptables –P DROP # ดร็อปทุกแพ็คเก็ต
iptables –P FORWARD DROP # ดร็อปทุกแพ็คเก็ตที่ฟอร์เวิร์ด

-L, --list

แสดงกฎเกณฑ์ทั้งหมดที่อยู่ใน chain

Example: iptables -L

-F, --flush

ลบกฎเกณฑ์ภายใน chain นั้นๆ

Example: iptables –F # ลบกฎเกณฑ์ทั้งหมดที่อยู่ในทุกๆ chain

iptables –F FORWARD # ลบกฎเกณฑ์ทั้งหมดในกลุ่มฟอร์เวิร์ด

-Z, --zero

ตั้งค่าตัวนับจำนวนกฎเกณฑ์ใหม่ให้เป็น 0

Example: iptables –Z FORWARD

- **COMMANDS (สำหรับจัดการกับกฎเกณฑ์ต่างๆ ภายใน chain)**

-A, --append

เพิ่มกฎเข้าไปที่ลำดับท้ายสุดใน chain

Example: iptables –A FORWARD –p tcp –j ACCEPT

-I, --insert

แทรกกฎเข้าไปยังลำดับที่ระบุด้วยหมายเลขที่กำหนด (ถ้าหมายเลขเป็น 1 จะทำให้กฎนั้นเป็นกฎแรกสุดของ chain เลย)

Example: iptables –I FORWARD 1 –p udp –j ACCEPT

-R, --replace

นำกฎใหม่ไปแทนที่กฎเดิมตรงลำดับที่ระบุด้วยหมายเลขที่กำหนด

Example: iptables –R FORWARD 1 –p udp –j DROP

-D, --delete

ลบกฎภายในกลุ่มของกฎเกณฑ์ตรงตำแหน่งที่ระบุด้วยหมายเลขที่กำหนดหรือกฎแรกที่ตรงกับที่กำหนดไว้

Example: iptables –D FORWARD 1 # ลบกฎแรกในกลุ่มฟอร์เวิร์ด

iptables –D FORWARD –p udp –j DROP # ลบกฎแรกที่ตรง

ข้อสังเกต ถ้าเราเพิ่มกฎด้วย option -A แล้วเราสามารถลบกฎข้อนั้นได้ด้วย option -D ที่เป็น mirror กัน เช่น

iptables –A INPUT –p icmp –j DROP # กฎที่เพิ่มด้วย option -A

iptables –D INPUT –p icmp –j DROP # ลบกฎที่เพิ่มเข้าไปได้โดยใช้ mirror

- **PARAMETERS**

Parameter ต่อไปนี้ใช้เพื่อการระบุข้อแม้ต่างๆสำหรับกฎเกณฑ์

-p, --protocol [!] *protocol-name*

สำหรับระบุโปรโตคอลของแพ็คเก็ต โดยสามารถระบุได้ 4 แบบ คือ tcp, udp, icmp หรือ all และสามารถใช้ ! เพิ่มที่ด้านหน้าของ protocol-name เพื่อให้มีความหมายในทางตรงกันข้ามได้

Example: iptables –A FORWARD –p tcp –j ACCEPT

-s, --source, --src [!] *address[/mask]*

สำหรับระบุแอดเดรสต้นทางของแพ็คเก็ต

Example: iptables –A FORWARD –s 161.246.5.119 –j ACCEPT

-d, --destination, dst [!] *address[/mask]*

สำหรับระบุแอดเดรสปลายทางของแพ็คเก็ต

Example: iptables –A FORWARD –d 161.246.5.120 –j DROP

หมายเหตุ แอดเดรสที่ใช้ระบุนั้นสามารถระบุได้ 4 รูปแบบด้วยกัน ดังนี้

1. ระบุเป็นชื่อเต็ม

เช่น localhost

www.ecpe.nu.ac.th

2. ระบุเป็น IP Address

เช่น 127.0.0.1

192.168.67.21

3. ระบุเป็นช่วงของ IP Address โดยใช้การระบุ Subnet Mask แบบย่อ

เช่น 192.168.7.0/24 เป็นการระบุช่วงแอดเดรสตั้งแต่ 192.168.7.0 จนถึง 192.168.7.255 โดยมี Subnet Mask เป็น 1

4. ระบุเป็นช่วงของ IP Address โดยใช้การระบุ Subnet Mask แบบเต็ม

เช่น 192.168.7.0/255.255.255.0 เป็นการระบุช่วงแอดเดรสตั้งแต่ 192.168.7.0 จนถึง 192.168.7.255 โดยมี Subnet Mask คือ 255.255.255.0

-j, --jump *target*

สำหรับระบุ Target ของแพ็คเก็ตว่าควรทำอย่างไรเวลาที่รายละเอียดของแพ็คเก็ตตรงกับกฎที่กำหนดไว้โดยที่ Target สามารถเป็น User-Defined chain ได้ด้วย

Example: iptables –A FORWARD –p tcp –j ACCEPT

-i, --in-interface [!] *[interface-name]*

สำหรับระบุการ์ดอินเตอร์เฟสที่แพ็คเก็ตผ่านเข้ามาได้โดยที่ option นี้สามารถใช้กับ INPUT, FORWARD และ PREROUTING chain ได้เท่านั้น

Example: iptables –A INPUT –i eth0 –j ACCEPT

-o, --out-interface [!] *[interface-name]*

สำหรับระบุการ์ดอินเตอร์เฟสที่แพ็คเก็ตผ่านออกไปโดยที่ option นี้สามารถใช้กับ FORWARD, OUTPUT และ POSTROUTING chain ได้เท่านั้น

Example: iptables –A OUTPUT –o eth1 –j ACCEPT

[i] **–f, --fragment**

ใช้ระบุว่าเป็นกฎที่ใช้สำหรับแฟรกเมนต์ที่สองขึ้นไปของแพ็คเก็ตที่ถูกแฟรกเมนต์เท่านั้น เพราะแพ็คเก็ตที่ถูกแฟรกเมนต์จะไม่สามารถตรวจสอบรายละเอียดต่างๆ ของแพ็คเก็ตได้

Example: iptables –A FORWARD –f –j DROP

• **OTHER OPTIONS**

-v, --verbose

ใช้ควบคู่กับ list command (-L, --list) เพื่อให้แสดงกฎเกณฑ์ทั้งหมดที่อยู่ใน chain อย่างละเอียด

Example: iptables –L –v

-n, --numeric

ใช้ควบคู่กับ list command (-L, --list) เพื่อให้แสดงกฎเกณฑ์ทั้งหมดที่อยู่ใน chain ในรูปแบบของหมายเลขแทนการแสดงเป็นชื่อ

Example: iptables –L –n

-x, --exact

ใช้ควบคู่กับ list command (-L, --list) เพื่อให้แสดงกฎเกณฑ์ทั้งหมดที่อยู่ใน chain โดยใช้จำนวนตัวเลขของตัวนับทั้งหมดตามค่าจริง ไม่ต้องใช้หน่วยแสดงผลต่างๆ

Example: iptables –L –x

--line-numbers

ใช้ควบคู่กับ list command (-L, --list) เพื่อให้แสดงกฎเกณฑ์ทั้งหมดที่อยู่ใน chain โดยที่แสดงตัวเลขลำดับของแต่ละกฎเกณฑ์ไว้ที่ด้านหน้าด้วย

Example: iptables –L –line-numbers

2.7.5 MATCH EXTENSIONS

- **TCP EXTENSIONS**

TCP Extension จะถูกโหลดขึ้นมาถ้ามีการระบุ -p tcp หรือ -protocol tcp และไม่มีกฎเกณฑ์ใดที่ตรงกันอีกแล้ว

--sport, --source-port [!] *[port[:port]]*

สำหรับระบุพอร์ตต้นทางของแพ็คเก็ต สามารถระบุเป็นชื่อบริการ เช่น www หรือหมายเลขบริการ เช่น 80 ได้ และสามารถระบุเป็นช่วงของพอร์ตได้ด้วยโดยใช้รูปแบบ first-port:last-port โดยถ้าไม่มีการระบุ first-port จะหมายถึงเริ่มจากพอร์ตที่ 0 จนถึง last-port และถ้าไม่มีการระบุ last-post จะหมายถึงเริ่มจาก first-port จนถึงพอร์ตที่ 65535

Example : iptables –A OUTPUT –p tcp –sport 1077 –j ACCEPT

--dport, --destination-port [!] *[port[:port]]*

สำหรับระบุพอร์ตปลายทางของแพ็คเก็ต สามารถระบุเป็นชื่อบริการหรือหมายเลขบริการ และสามารถระบุเป็นช่วงของพอร์ตได้เช่นเดียวกับการระบุพอร์ตต้นทางของแพ็คเก็ต

Example: iptables –A INPUT –p tcp –dport 12345 –j DROP

--tcp-flag [!] *mask comp*

สำหรับระบุแฟลกของแพ็คเก็ตที่ต้องการตรวจสอบโดยแบ่งเป็น 2 ชุด โดยที่แฟลกชุดแรกหมายถึงชุดแฟลกที่เราต้องการตรวจสอบ และแฟลกชุดที่สองหมายถึงชุดแฟลกที่ถูกเซต ซึ่งสามารถระบุได้ด้วย SYN, ACK, FIN, RST, URG, PSH, ALL และ NONE

Example: iptables –A INPUT –p tcp –tcp-flag ALL SYN,ACK –j DROP

[!] –syn

สำหรับระบุแพ็คเก็ตที่การร้องขอ TCP connection ซึ่งจะเป็นแพ็คเก็ตที่ SYN flag ถูกเซต และ ACK flag กับ FIN flag ถูกเคลียร์ (มีความหมายเช่นเดียวกับการใช้ --tcp-flag SYN, ACK,FIN SYN)

Example: iptables –A FORWARD –p tcp –syn –j ACCEPT

--tcp-option [!] *number*

สำหรับระบุแพ็คเก็ตที่มีการเซตค่า TCP option (ข้อมูลเพิ่มเติม) เป็นหมายเลขตามที่กำหนด ซึ่งแพ็คเก็ตใดมี TCP header ที่ไม่สมบูรณ์ก็จะถูกดร็อปไปโดยอัตโนมัติ

Example: iptables –A FORWARD –p tcp –tcp-option 536 –j ACCEPT

• **UDP EXTENSIONS**

UDP Extension จะถูกโหลดขึ้นมาถ้ามีการระบุ -p udp หรือ -protocol udp และไม่มีกฎเกณฑ์ใดที่ตรงกันอีกแล้ว

--sport, --source-port [!] *[port[:port]]*

สำหรับระบุพอร์ตต้นทางของแพ็คเก็ต สามารถระบุเป็นชื่อบริการ เช่น www หรือหมายเลขบริการ เช่น 80 ได้ และสามารถระบุเป็นช่วงของพอร์ตได้ด้วยโดยใช้รูปแบบ first-port:last-port โดยถ้าไม่มีการระบุ first-port จะหมายถึงเริ่มจากพอร์ตที่ 0 จนถึง last-port และถ้าไม่มีการระบุ last-post จะหมายถึงเริ่มจาก first-port จนถึงพอร์ตที่ 65535

Example : iptables –A OUTPUT –p udp --sport 1077 –j ACCEPT

--dport, --destination-port [!] *[port[:port]]*

สำหรับระบุพอร์ตปลายทางของแพ็คเก็ต สามารถระบุเป็นชื่อบริการหรือหมายเลขบริการ และสามารถระบุเป็นช่วงของพอร์ตได้เช่นเดียวกับการระบุพอร์ตต้นทางของแพ็คเก็ต

Example: iptables –A INPUT –p udp --dport 12345 –j DROP

• **ICMP EXTENSION**

ICMP Extension จะถูกโหลดขึ้นมาถ้ามีการระบุ -p icmp หรือ -protocol icmp และไม่มีกฎเกณฑ์ใดที่ตรงกันอีกแล้ว

--icmp-type [!] *type*

สำหรับระบุแพ็คเก็ตที่มี ICMP type ตรงตาม type ที่กำหนด ซึ่งสามารถกำหนดเป็นหมายเลขหรือชื่อ type ก็ได้

Example: iptables –A FORWARD –p icmp --icmp-type 3 –j DROP

หมายเหตุ สามารถเรียกดู ICMP type ต่างๆ ได้โดยใช้ iptables –p icmp –h

• **OTHER MATCH EXTENSIONS**

Extension กลุ่มนี้จะต้องขึ้นต้นด้วย –m หรือ --match

mac สำหรับระบุหมายเลขแมคแอดเดรส

–mac-source [!] *address*

สำหรับระบุแมคแอดเดรสของอีเทอร์เน็ตดีไวซ์ต้นทางของแพ็คเก็ตซึ่งจะอยู่ในรูปแบบ XX:XX:XX:XX:XX:XX โดยการระบุในแบบนี้จะใช้ได้กับ INPUT, FORWARD หรือ PREROUTING chain เท่านั้น

Example: iptables -A INPUT -m mac --mac-source 00:60:67:30:AC:E5 -j DROP

limit สำหรับระบุจำนวนที่เข้าคู่กับกฎนั้นๆ โดยมักใช้ควบคู่กับ LOG target เพื่อใช้จำกัดจำนวนการบันทึก log

--limit *rate*

สำหรับค่าเฉลี่ยของจำนวนที่เข้าคู่กับกฎนั้นๆ ต่อช่วงเวลาหนึ่งๆ โดยสามารถเลือกช่วงเวลาได้หลายแบบ ดังนี้ /second, /minute, /hour หรือ /day โดยค่า default คือ 3/hour

หมายเหตุ /s = /second

/m = /minute

/h = /hour

--limit-burst *number*

สำหรับระบุค่าสูงสุดของจำนวนที่เข้าคู่กับกฎนั้นๆ โดยจะถูกเพิ่มทีละหนึ่งทุกๆ ครั้งที่อัตราที่เข้าคู่กันของกฎนั้นๆ น้อยกว่าที่ระบุไว้โดย --limit มีค่า default คือ 5

Example: iptables –A INPUT –m limit --limit 3/m
--limit-burst 3 –j LOG

multiport สำหรับระบุพอร์ตหลายๆ พอร์ตในกฎ ซึ่งสามารถระบุถึง 15 พอร์ตต่อหนึ่งกฎ แต่สามารถใช้งานได้กับโปรโตคอล TCP และ UDP เท่านั้น

--sport, --source-port *[port[,port]]*

สำหรับระบุพอร์ตต้นทางของแพ็คเก็ต

Example: iptables –A FORWARD –p tcp –m multiport –sport 21,23
–j DROP

--dport, --destination-port *[port[,port]]*

สำหรับระบุพอร์ตปลายทางของแพ็คเก็ต

Example: iptables –A FORWARD –p udp –m multiport --dport 21,23
–j ACCEPT

--port *[port[,port]]*

สำหรับระบุพอร์ตต้นทางและปลายทางของแพ็คเก็ต

Example: iptables –A FORWARD –p tcp –m multiport --port 123,1234
–j DROP

state สำหรับระบุสถานะการเชื่อมต่อของแพ็คเก็ตโดยสามารถระบุเป็นชุดได้โดยใช้ , คั่น

--state *state*

สามารถเลือกระบุสถานะการเชื่อมต่อของแพ็คเก็ต ได้ดังนี้

- **INVALID**

สำหรับแพ็คเก็ตที่ไม่ทราบสถานะการเชื่อมต่อ

- **ESTABLISHED**

สำหรับแพ็คเก็ตที่อยู่ในสถานะที่มีการสร้างการเชื่อมต่อระหว่างทั้ง สองฝั่งเรียบร้อยแล้ว

- **NEW**

สำหรับแพ็คเก็ตที่เริ่มต้นเชื่อมต่อใหม่

- **RELATED**

สำหรับแพ็คเก็ตที่เริ่มต้นเชื่อมต่อใหม่ แต่เกี่ยวข้องกับการเชื่อมต่อที่มีอยู่แล้ว เช่น ICMP error

Example: iptables -A FORWARD –m state –state NEW,ESTABLISHED -j DROP

unclean โมดูลนี้ไม่มี option ใดๆ เพียงใช้สำหรับระบุแพ็คเก็ตที่ดูไม่ปกติ โดยพิจารณาจากผลที่ได้จากการทดสอบ

Example: iptables –A FORWARD –m unclean –j DROP

2.7.6 TARGET EXTENSIONS

• **LOG**

--log-level *level*

บันทึก log เฉพาะ level ที่กำหนดเท่านั้น

Example: iptables –A FORWARD –d 127.0.0.1 –j LOG --log-level 7

--log-prefix *prefix*

ใส่ prefix ที่กำหนดไปที่ log message ด้วยเพื่อให้เข้าใจและสังเกตได้ง่าย โดย prefix สามารถยาวได้ถึง 14 ตัวอักษร

Example: iptables –A INPUT –p tcp --dport 12345 –j LOG –log-prefix "Netbus Scan"

–log-tcp-sequence

บันทึก TCP sequence number ด้วยการใช้การบันทึก log แบบนี้เสี่ยงต่อการเกิดอันตรายทางด้านความปลอดภัยได้ เพราะอาจมีผู้ไม่ประสงค์ดีมาอ่านข้อมูลตรงนี้ได้

Example: iptables –A FORWARD –p tcp –j LOG --log-tcp-sequence

--log-tcp-options

บันทึก option ที่อยู่ใน TCP header ของแพ็คเก็ตด้วย

Example: iptables –A FORWARD –p tcp –j LOG --log-tcp-options

--log-ip-options

บันทึก option ที่อยู่ใน IP header ของแพ็คเก็ตด้วย

Example: iptables –A FORWARD –p tcp –j LOG --log-ip-options

- **MARK**

--set-mark *mark*

สำหรับเซตค่า netfilter mark ที่เกี่ยวข้องกับแพ็คเก็ตใช้ได้เฉพาะกับ mangle table

Example: iptables -t mangle -p tcp --dport 80 -j MARK --set-mark 2

- **REJECT**

--reject-with *type*

สำหรับระบุชนิดของข้อความที่ส่งกลับไปตอน REJECT โดยมีชนิดของข้อความที่สามารถระบุได้ ดังนี้

- **ICMP error message** (ค่า default คือ icmp-port-unreachable)
  - icmp-net-unreachable
  - icmp-host-unreachable
  - icmp-port-unreachable
  - icmp-proto-unreachable
  - icmp-net-prohibited
  - icmp-host-prohibited
- **Other**
  - echo-reply
  - tcp-reset

Example: iptables –A INPUT –d 192.168.1.0 –j REJECT
--reject-with icmp-net-unreachable

- **TOS**

--set-tos *tos*

สำหรับเซตค่าให้กับส่วน Type Of Service (8-bit) โดยจะเซตเป็นตัวเลขหรือชื่อของ TOS ก็ได้ และใช้ได้เฉพาะกับ mangle table เท่านั้น

Example: iptables –t mangle –A OUTPUT –p tcp --dport 80 –j TOS --set-tos 8

หมายเหตุ สามารถดูรายชื่อของ TOS ที่ได้โดยใช้ iptables –j TOS –h

• **MIRROR**

ทำการสลับค่าของแอดเดรสต้นทางและแอดเดรสปลายทางใน IP header แล้วทดลองส่งแพ็คเก็ตออกไป ใช้เพื่อพิสูจน์ผล สามารถใช้ได้กับ INPUT, OUTPUT, FORWARD chain หรือ User-Defined chain ที่ถูกเรียกจาก chain เหล่านั้นเท่านั้น

Example: iptables –A FORWARD –p tcp –j MIRROR

• **SNAT**

เป็น target ที่ใช้ได้กับ nat table และ POSTROUTING chain เท่านั้น ใช้สำหรับระบุแอดเดรสต้นทางของแพ็คเก็ตที่ควรจะต้องถูกแก้ไข

--to-source *<ipaddr>[-<ipaddr>][:port-port]*

สำหรับระบุ IP Address ต้นทางใหม่ของแพ็คเก็ต โดยสามารถระบุเป็นช่วงของแอดเดรสได้และสามารถระบุพอร์ตเพิ่มเติมได้อีกด้วย (การระบุพอร์ตใช้ได้กับโปรโตคอล TCP หรือ UDP เท่านั้น)

Example: iptables –t nat –A POSTROUTING –s 192.168.0.7 –j SNAT --to-source 161.246.5.7

• **DNAT**

เป็น target ที่ใช้ได้กับ nat table และ PREROUTING, OUTPUT chain หรือ User-Defined chain ที่ถูกเรียกจาก chain เหล่านั้นเท่านั้น ใช้สำหรับระบุแอดเดรสปลายทางของแพ็คเก็ตที่ควรจะ ต้องถูกแก้ไข

--to-destination *<ipaddr>[-<ipaddr>][:port-port]*

สำหรับระบุ IP Address ปลายทางใหม่ของแพ็คเก็ต โดยสามารถระบุเป็นช่วงของแอดเดรสได้ และสามารถระบุพอร์ตเพิ่มเติมได้อีกด้วย (การระบุพอร์ตใช้ได้กับโปรโตคอล TCP หรือ UDP เท่านั้น)

Example: iptables –t nat --A OUTPUT –d 161.246.5.7 -j DNAT --to-destination 192.168.0.7

• **MASQUERADE**

เป็น target ที่ใช้ได้กับ nat table และ POSTROUTING chain เท่านั้น ใช้สำหรับการกำหนด Dynamic IP Address ในการเชื่อมต่อแบบ Dial-Up โดยถ้าต้องการกำหนด Static IP Address ควรใช้ SNAT target

--to-ports *<port>[-<port>]*

สำหรับระบุพอร์ตต้นทาง โดยสามารถระบุเป็นช่วงของพอร์ตได้ด้วย แต่สามารถใช้ได้กับโปรโตคอล TCP หรือ UDP เท่านั้น

Example: iptables –t nat –A POSTROUTING –p tcp –s 192.168.1.0/24 –j MASQUERADE --to-ports 1200-1455

- **REDIRECT**

เป็น target ที่ใช้ได้กับ nat table และ PREROUTING, OUTPUT chain หรือ User-Defined chain ที่ถูกเรียกจาก chain เหล่านั้นเท่านั้น สำหรับเปลี่ยนแปลงแอดเดรสปลายทางเพื่อให้ส่งแพ็คเก็ตให้ตัวเอง คือแอดเดรสปลายทางถูกเปลี่ยนเป็น 127.0.0.1 นั่นเอง

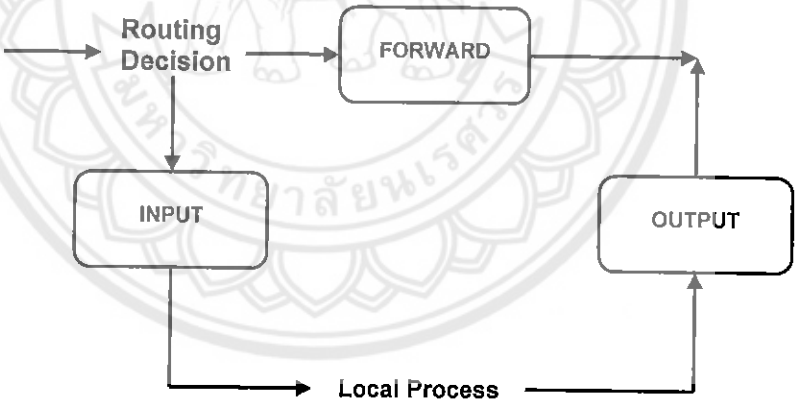
–to-ports *<port>[-<port>]*

สำหรับระบุพอร์ตปลายทาง โดยสามารถระบุเป็นช่วงของพอร์ตได้โดยถ้าไม่ใช้ option นี้ พอร์ตปลายทางก็จะไม่ถูกเปลี่ยนแปลง สามารถใช้ได้กับโปรโตคอล TCP หรือ UDP เท่านั้น

Example: iptables –t nat PREROUTING –p tcp –d ! 192.168.1.0/24 --dport www –j REDIRECT --to-ports 3128

2.7.7 การเดินทางของแพ็คเก็ตผ่าน chain ต่างๆ

2.7.7.1 FILTER TABLE



รูปที่ 2.53 Diagram แสดงการเดินทางผ่าน chain ต่างๆ ใน filter table

1. เมื่อแพ็คเก็ตเข้ามาในเครื่องไฟร์วอลล์ (เข้ามาทางเน็ตเวิร์กอินเตอร์เฟส) ต้องผ่าน Routing Decision ซึ่งเป็นขั้นตอนการตัดสินของ Kernel ว่าแพ็คเก็ตควรเดินทางต่อไปที่ใด โดย Kernel จะมองดูจากแอดเดรสปลายทางของแพ็คเก็ต

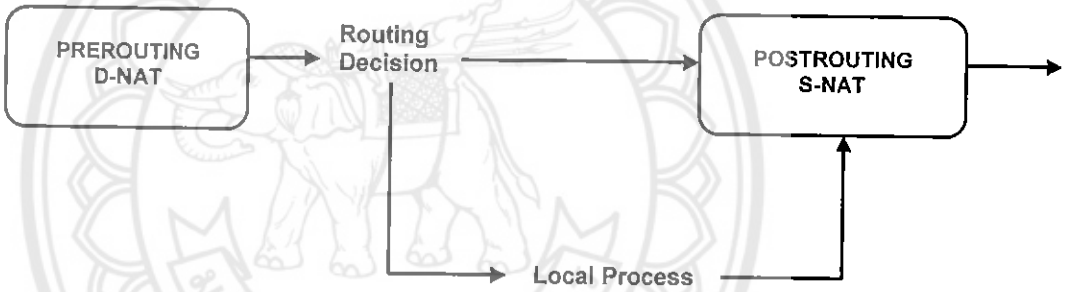
2. ถ้าแอดเดรสปลายทาง คือ แอดเดรสของเครื่องไฟร์วอลล์เอง แพ็คเก็ตจะเดินทางไปตรวจสอบที่ INPUT chain โดยเมื่อตรวจสอบแล้วสามารถยอมให้แพ็คเก็ตผ่านได้ โปรเซสที่กำลัง

รอแพ็คเก็ตนี้อยู่ก็จะได้รับแพ็คเก็ตเพื่อไปประมวลผลต่อไป แต่ถ้าไม่สามารถยอมให้แพ็คเก็ตผ่านได้แพ็คเก็ตก็จะถูกดร็อปทิ้ง

3. แต่ถ้าแอดเดรสปลายทางไม่ใช่แอดเดรสของเครื่องไฟร์วอลล์ แพ็คเก็ตนั้นจะถูกดร็อป ถ้าเครื่องไฟร์วอลล์ไม่มีความสามารถในการทำ forwarding แต่ถ้ามีแพ็คเก็ตก็จะเดินทางไปตรวจสอบที่ FORWARD chain โดยถ้ายอมให้แพ็คเก็ตผ่านได้แพ็คเก็ตก็จะถูกส่งออกไปทางเน็ตเวิร์กอินเตอร์เฟส (คนละอินเตอร์เฟสกับที่เข้ามา) แต่ถ้าไม่ยอมให้ผ่านแพ็คเก็ตก็จะถูกดร็อปทิ้งไป

4. และเมื่อโปรแกรมในเครื่องไฟร์วอลล์ต้องการส่งแพ็คเก็ตออกไปภายนอก แพ็คเก็ตนั้นๆ ก็จะเดินทางไปตรวจสอบที่ OUTPUT chain โดยถ้ายอมให้ผ่านได้ก็จะถูกส่งออกไป แต่ถ้าไม่ยอมให้ผ่าน แพ็คเก็ตก็จะถูกดร็อปทิ้งไป

2.7.7.2 NAT TABLE



รูปที่ 2.54 Diagram แสดงการเดินทางผ่าน chain ต่างๆ ใน nat table

1. เมื่อแพ็คเก็ตเข้ามาในเครื่องไฟร์วอลล์ (เข้ามาทางเน็ตเวิร์กอินเตอร์เฟส) แพ็คเก็ตต้องเดินทางไปตรวจสอบที่ PREROUTING chain ก่อนเพื่อทำการ NAT แอดเดรส

2. จากนั้นแพ็คเก็ตจะต้องผ่าน Routing Decision เพื่อให้ Kernel ตัดสินใจว่าจะให้แพ็คเก็ตเดินทางต่อไปที่ใด

3. โดยถ้าแพ็คเก็ตนี้เป็นการส่งต่อแพ็คเก็ตจะเดินทางต่อไปที่ POSTROUTING chain เพื่อ NAT แอดเดรสเพื่อทำการส่งต่อออกไปอีกที แต่ถ้าแพ็คเก็ตเป็นที่ต้องการของโปรเซสภายในแพ็คเก็ตก็จะถูกส่งไปให้โปรเซสที่รออยู่เพื่อนำไปประมวลต่อ

4. และเมื่อต้องมีการส่งแพ็คเก็ตออกไปภายนอก แพ็คเก็ตนั้นๆ ก็จะต้องเดินทางไปที่ POSTROUTING chain เพื่อทำการ NAT แอดเดรสก่อน

2.8 สควิด (Squid)

Squid เป็นซอฟแวร์สำหรับแคชเว็บเพจที่มีรูปแบบการใช้งานเป็นที่เก็บข้อมูลที่โหลดจากอินเตอร์เน็ตมาเก็บไว้ก่อนที่จะส่งข้อมูลไปให้เครื่องลูกข่าย ซึ่งในกรณีที่มีการโหลดข้อมูลที่ซ้ำกัน เช่น การเปิดเว็บเดียวกันของเครื่องลูกข่าย Squid ก็จะเอาข้อมูลที่เก็บไว้ส่งให้เครื่องลูกข่ายที่ร้องขอเข้ามา โดยมีพร็อกซี่เซิร์ฟเวอร์ หรือ Web Cache เป็นเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นตัวกลางระหว่างเครื่องลูกข่ายและเว็บเซิร์ฟเวอร์ในการร้องขอและจัดส่งหน้าเว็บเพจ ทำให้ไม่ต้องเสียเวลาในการร้องขอและโหลดใหม่

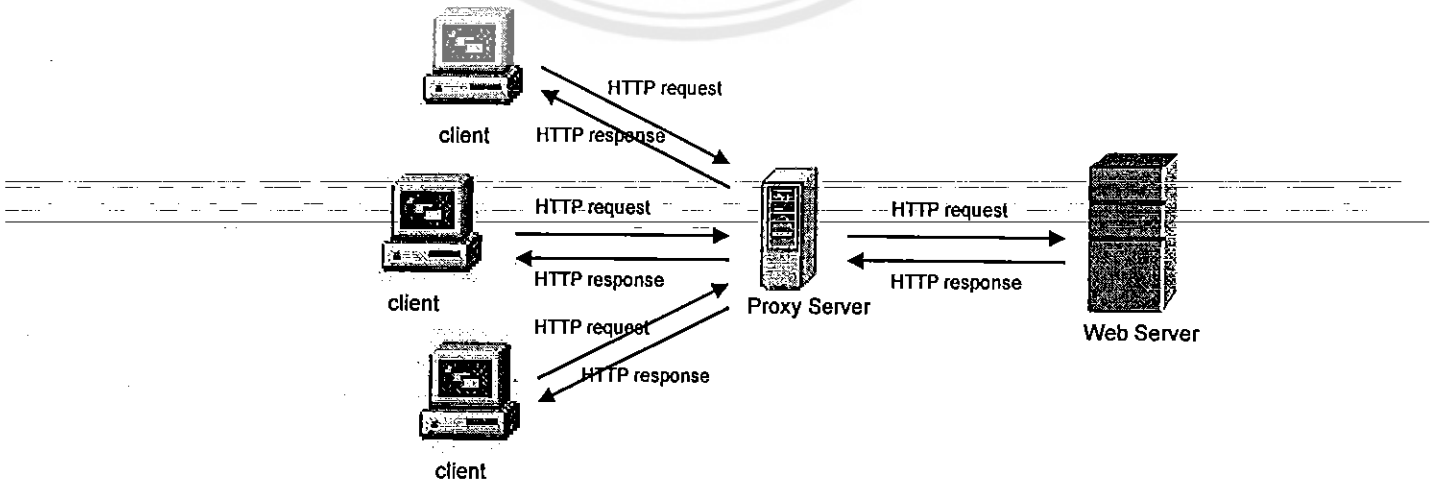
ในปัจจุบันพร็อกซี่เซิร์ฟเวอร์จึงมีการใช้งานที่แพร่หลายมากขึ้น เพราะการติดตั้งพร็อกซี่เซิร์ฟเวอร์ในแต่ละองค์กรจะช่วยในการประหยัดงบประมาณที่ต้องเสียไปกับค่าเช่าวงจรสื่อสาร และทำให้ผู้ใช้เรียกดูข้อมูลได้เร็วขึ้น

2.8.1 ประโยชน์ของพร็อกซี่เซิร์ฟเวอร์

1. เพื่อความรวดเร็วในการดาวน์โหลด เนื่องจากการดาวน์โหลดหน้าเว็บเพจที่ผู้ใช้คนอื่นเคยร้องขอก่อนหน้าแล้ว จะเป็นการดาวน์โหลดมาจากแคชของพร็อกซี่เซิร์ฟเวอร์แทนซึ่งถ้าพร็อกซี่เซิร์ฟเวอร์อยู่ในแลนเดียวกับเครื่องลูกข่าย การดาวน์โหลดก็จะมีความเร็วเท่ากับความเร็วของแลน

2. ประหยัดแบนด์วิดธ์ เพราะถ้าผู้ใช้หลายคนทำการดาวน์โหลดหน้าเว็บเพจเดียวกันโดยไม่มีการใช้พร็อกซี่เซิร์ฟเวอร์ ทำให้ต้องใช้แบนด์วิดธ์ในการดาวน์โหลดเพิ่มขึ้น แต่ถ้าผู้ใช้ทุกคนใช้พร็อกซี่เซิร์ฟเวอร์ก็จะมีการดาวน์โหลดครั้งแรกเพียงครั้งเดียว ส่วนผู้ที่ร้องขอทีหลังก็จะได้รับเว็บเพจหน้านั้นจากแคชของพร็อกซี่เซิร์ฟเวอร์

3. มีประโยชน์ด้าน Security เพราะสามารถทำการกำหนดได้ว่าจะให้ผู้ใช้คนใดสามารถเข้าชมเว็บไซท์ใดได้บ้าง โดยทำการกำหนดค่า ACL (Access Control List) ที่ตัวพร็อกซี่เซิร์ฟเวอร์

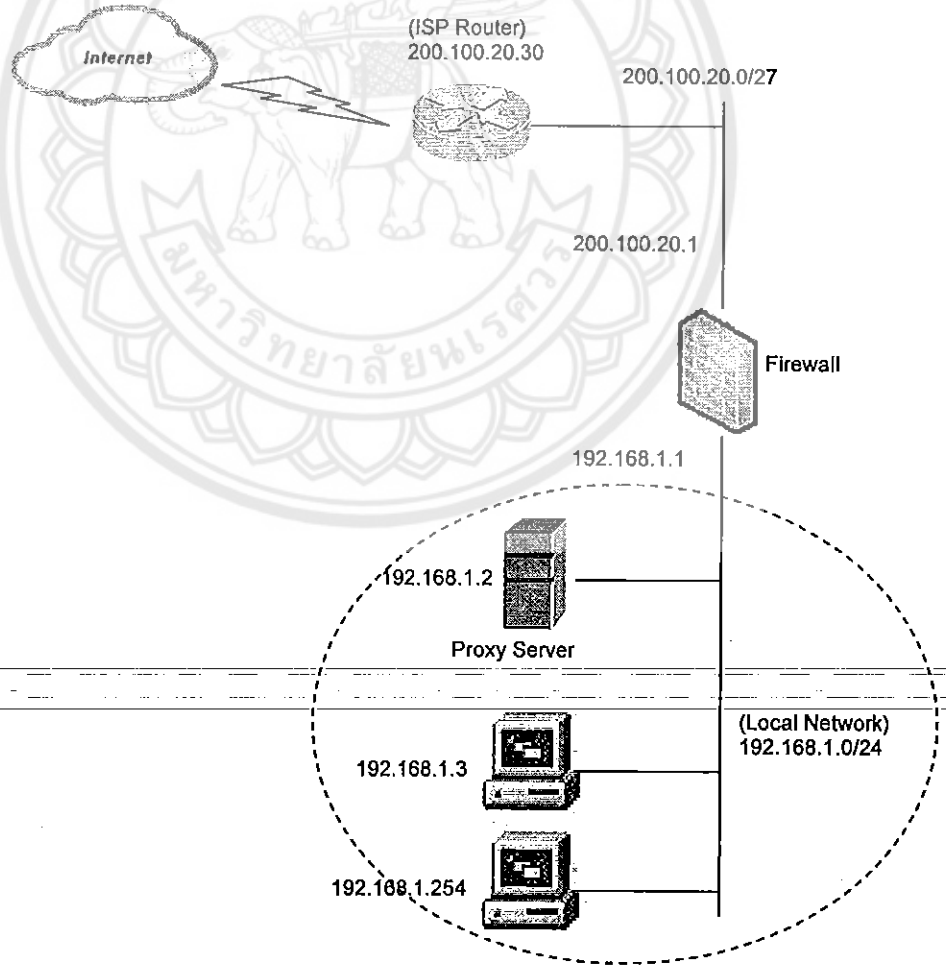


รูปที่ 2.55 เครื่องลูกข่ายทำการร้องขอเว็บเพจผ่านทางพร็อกซี่เซิร์ฟเวอร์

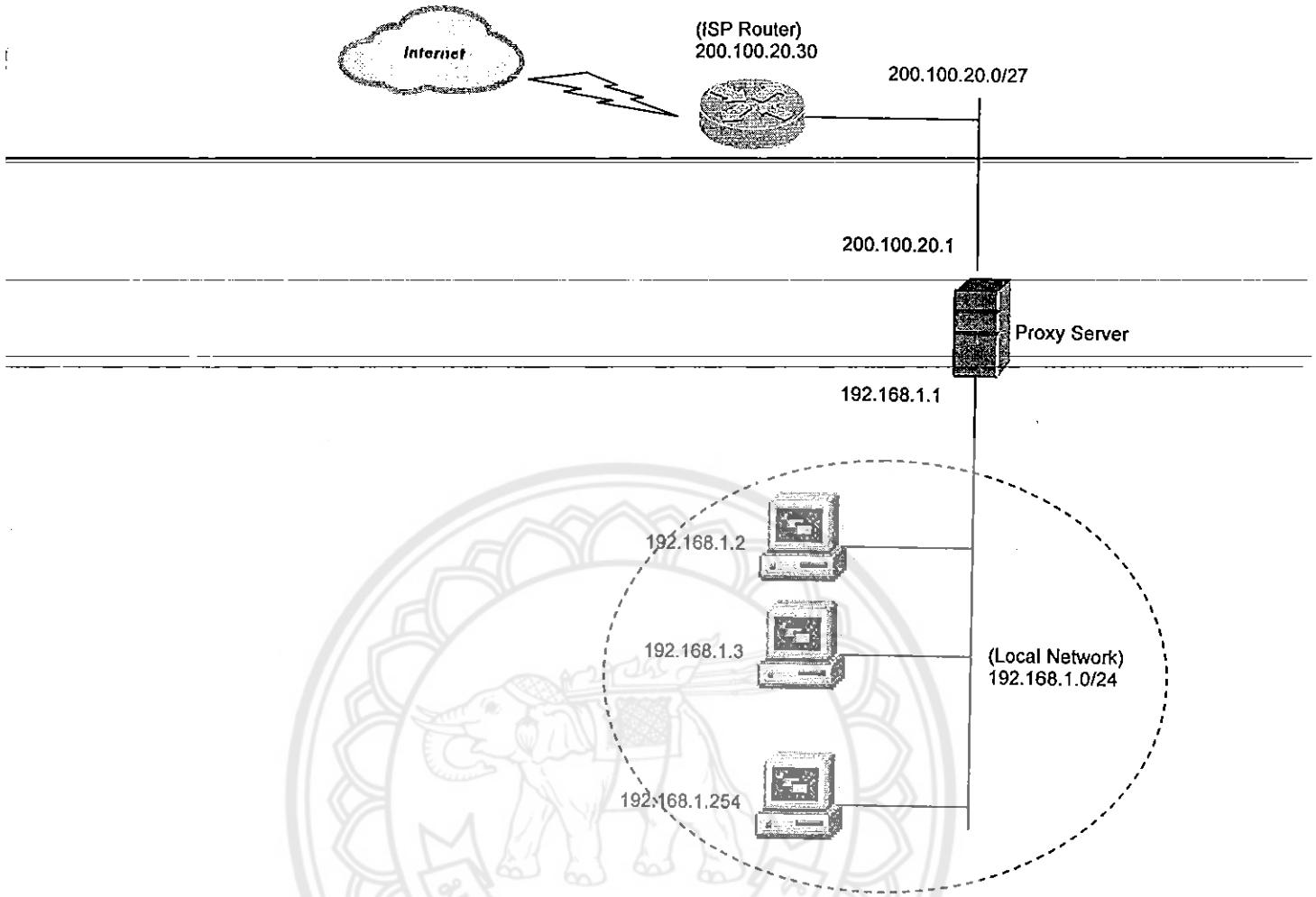
ในการใช้งานพร็อกซี่เซิร์ฟเวอร์ ผู้ใช้เพียงป้อนค่า IP Address ของเครื่องพร็อกซี่เซิร์ฟเวอร์และหมายเลขพอร์ตที่เปิดให้บริการ เมื่อทำการป้อนชื่อของเว็บไซท์ที่เว็บเบราเซอร์แล้วกดปุ่ม Enter เครื่องลูกข่ายก็จะทำการติดต่อไปที่เครื่องพร็อกซี่เซิร์ฟเวอร์ จากนั้นเครื่องลูกข่ายก็จะพูดคุยกับเครื่องพร็อกซี่เซิร์ฟเวอร์โดยใช้โปรโตคอล HTTP ว่าต้องการหน้าเว็บเพจใด จากนั้นพร็อกซี่เซิร์ฟเวอร์ก็จะทำการร้องขอหน้าเว็บเพจที่ผู้ใช้ต้องการโดยทำการติดต่อที่พอร์ต 80 ซึ่งเป็นพอร์ตของเว็บเซิร์ฟเวอร์ปกติ เมื่อได้รับหน้าเว็บเพจก็จะส่งต่อให้กับเครื่องของผู้ใช้ต่อไป

2.8.2 ตำแหน่งที่อยู่ของพร็อกซี่เซิร์ฟเวอร์

พร็อกซี่เซิร์ฟเวอร์สามารถวางไว้ที่ตำแหน่งต่างๆ ภายในเครือข่ายได้ โดยส่วนมากจะวางไว้ใน Local Network หรือไม่ก็วางไว้ที่ตำแหน่งของเกตเวย์ (Gateway) ซึ่งถ้าวางพร็อกซี่เซิร์ฟเวอร์ไว้ในตำแหน่งเกตเวย์จะต้องทำให้เครื่องพร็อกซี่เซิร์ฟเวอร์ทำงานเป็นไฟร์วอลล์หรือเราเตอร์ด้วย เพื่อให้ผู้ใช้สามารถติดต่อทาง TCP/IP โดยตรงกับภายนอกได้ (ในกรณีที่ผู้ใช้ภายในไม่ได้ใช้งานอินเตอร์เน็ตเฉพาะการเข้าเว็บ)



รูปที่ 2.56 การตั้งพร็อกซี่เซิร์ฟเวอร์ไว้ใน Local Network



รูปที่ 2.57 การตั้งพร็อกซี่เซิร์ฟเวอร์ไว้ที่ตำแหน่งของเกตเวย์

2.8.3 วิธีการติดตั้ง

การติดตั้ง Squid

วิธีการติดตั้ง Squid ทำได้โดยใช้คำสั่งในการติดตั้งดังนี้

```
[root@linux root]# mount /mnt/cdrom
[root@linux root]# cp /mnt/cdrom/RedHat/RPMS/squid-2.4.STABLE7-4.i386.rpm
[root@linux root]# rpm -Uv squid-2.4.STABLE7-4.i386.rpm
warning: squid-2.4.STABLE7-4.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
Preparing packages for installation...
squid-2.4.STABLE7-4
[root@linux root]# umount /mnt/cdrom
[root@linux root]#
```

2.8.4 วิธีการใช้งาน

การปรับแต่ง Squid

ก่อนที่จะทำการเรียกใช้งานจะต้องทำการปรับปรุงค่าคอนฟิก (Configure) ที่สำคัญของ Squid ก่อน โดยทำการแก้ไขไฟล์ดังนี้

1. กำหนดหมายเลขพอร์ตโดยเอาเครื่องหมาย # ในบรรทัด # http_port 3128 ออกไป ดังนี้

```
#
# You may specify multiple socket addresses on multiple lines.
# Default :
http_port 3128
```

เพื่อให้ squid รอรับการเชื่อมต่อที่พอร์ต 3128 หรืออาจใช้พอร์ตหมายเลขอื่นก็ได้ เช่น พอร์ต 8080 เป็นต้น หรือใช้ทั้งสองหมายเลขโดยการเพิ่มบรรทัดเข้าไปดังนี้

```
#
# You may specify multiple socket addresses on multiple lines.
# Default :
http_port 3128
http_port 8080
```

2. เพิ่ม ACL โดยการแทรกบรรทัด acl localnetwork src 192.168.1.0/255.255.255.0 เข้าไปเพื่อกำหนดชื่อให้กับเครือข่ายภายใน สำหรับใช้ในการอนุญาตหรือไม่อนุญาตใน ACL

```
# Examples:
# acl myexample dst_as 1241
# acl password proxy_auth REQUIRED
# acl fileupload req_mime_type -I ^multipart/form-data$
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
```

```
acl localnetwork src 192.168.1.0/255.255.255.0
acl SSL_ports port 443 563
acl Safe_ports port 80 # http
acl Safe_ports port 21 # ftp
acl Safe_ports port 443 563 # https, snews
```

3. อนุญาตให้ localnetwork เรียกใช้งาน Squid ได้ โดยการเพิ่มบรรทัด

```
# Default
# http_access deny all
#
# Recommended minimum configuration:
#
# Only allow cachemgr access from localhost
http_access allow manager localhost
http_access allow localnetwork
http_access deny manager
# Deny requests to unknown ports
http_access deny !Safe_ports
# Deny CONNECT to other than SSL ports
http_access deny CONNECT !SSL_ports
#
```

การเชื่อมต่อกับพร็อกซี่เซิร์ฟเวอร์เครื่องอื่น

ตัวแปร cache_peer ใช้ในการกำหนดความสัมพันธ์ในการเชื่อมต่อกับเครื่องพร็อกซี่เซิร์ฟเวอร์เครื่องอื่น ซึ่งสามารถมีได้มากกว่าหนึ่งเครื่อง มีรูปแบบดังนี้

```
cache_peer hostname type http_port icp_port options
```

- **hostname** คือชื่อหรือ IP ของเครื่องพร็อกซี่เซิร์ฟเวอร์ที่จะเชื่อมต่อด้วย ถ้าเป็นชื่อจะต้องมีทั้ง forward และ reverse name หมายความว่า ถ้ารู้ IP สามารถระบุเป็นชื่อได้ และถ้ารู้ชื่อก็สามารถระบุเป็น IP ได้เช่นกัน จากตัวอย่างเป็นเครื่อง proxy.nu.ac.th

- **type** มีค่าได้เป็น
  - parent คือ ถ้าเครื่องลูกข่ายขอข้อมูลมาที่ A แต่ A ไม่มีข้อมูลก็จะไปถาม B ถ้า B มีข้อมูลก็จะส่งข้อมูลนั้นไปให้ A แต่ถ้าไม่มี B จะไปหาข้อมูลนั้นแล้วส่งต่อไปให้ A
  - sibling คือ ถ้าเครื่องลูกข่ายขอข้อมูลมายัง A แต่ A ไม่มีข้อมูลนั้น A ก็จะไปถาม B ถ้า B มีก็จะส่งข้อมูลนั้นให้กับ A แต่ถ้าไม่มี A จะต้องไปหาข้อมูลนั้นเอง B จะไม่ไปหาให้
  - multicast คือ การไปขอข้อมูลจากเครื่องใดก็ได้
- **http_port** คือพอร์ตที่กำหนดการส่งข้อมูล HTTP ของเครื่อง proxy.nu.ac.th
- **icp_port** คือพอร์ตที่ใช้แลกเปลี่ยนข้อมูลระหว่างกันของเครื่อง proxy.nu.ac.th
- **options** จะระบุหรือไม่ก็ได้ ตัวอย่างเช่น weight=n เป็นการกำหนดว่าต้องการไปหา parent ตัวไหนก่อน ค่าของ n แทนตัวเลข ค่าไหนมากกว่าก็ไปหา parent ตัวนั้นก่อน

ตัวอย่างเช่น ถ้าต้องการชี้เครื่อง proxy.nu.ac.th ให้เป็น parent สามารถกำหนดได้ดังนี้

```
cache_peer proxy.nu.ac.th parent 8080 3130
```

การกำหนด Local Domains

เป็นการกำหนดโดเมนหรือ IP ที่ไม่ต้องการให้พร็อกซี่เซิร์ฟเวอร์ไปถามข้อมูลจากเครื่องที่เป็น parent หรือ sibling ส่วนใหญ่ชื่อที่กำหนดจะเป็นโดเมนภายในองค์กร หรือชื่อโดเมนที่พร็อกซี่เซิร์ฟเวอร์สามารถไปหาได้เร็วกว่าที่จะไปถามจากเครื่อง parent หรือ sibling ใดๆ เช่น สำหรับ NECTEC Web Site ที่อยู่ภายใน NECTEC เองก็ไม่ควรให้ออกไปถาม parent หรือ sibling ที่ไหน ควรกำหนด nectec.or.th ให้อยู่เป็น Local Domain ตัวอย่างดังนี้

```
acl local_domain dsdomain nectec.or.th
always_direct allow local_domain
```

การกำหนดขนาดและไดเรกทอรีที่ใช้เก็บข้อมูล

ใช้คำว่า cache_dir แล้วตามด้วยไดเรกทอรี, ขนาด (MB), จำนวนไดเรกทอรีย่อยภายใน

ตัวอย่างเช่น

```
cache_dir /cache/disk1 500 16 256
cache_dir /cache/disk2 500 16 256
```

การกำหนดชื่อของ Log files

Log files ของ Squid มีอยู่ 3 ชนิด แทนด้วยตัวแปร cache_access_log, cache_log และ cache_store _log โดย cache_access_log จะเก็บข้อมูลว่าเครื่องลูกข่ายมีการขอข้อมูลอะไรบ้าง ส่วน cache_log จะเก็บ error message ถ้า พร็อกซี่เซิร์ฟเวอร์ มีข้อผิดพลาดไม่สามารถทำงานได้ ผู้ควบคุมระบบก็สามารถตรวจสอบได้จากไฟล์นี้ว่ามีข้อผิดพลาดอะไร และ cache_store_log จะเก็บข้อมูลการทำงานของ Squid ว่าเก็บข้อมูลอะไรไว้บ้างหรือลบข้อมูลอะไรบ้าง ซึ่งข้อมูลเหล่านี้อาจจะไม่จำเป็นสำหรับผู้ควบคุม ตัวแปรทั้ง 3 ชนิดมีรูปแบบดังนี้

```
cache_access_log /usr/local/etc/squid/logs/access.log
cache_log /usr/local/etc/squid/logs/cache.log
cache_store_log /usr/local/etc/squid/logs/store.log
```

การกำหนดชื่อไฟล์ที่เก็บ Process ID

ทุกครั้งที่มีการ start Squid จะมีการสร้างไฟล์ที่เก็บ Process ID เพื่อให้ง่ายต่อการ restart หรือ shut down Squid โดยมีรูปแบบดังนี้

```
pid_filename /usr/local/etc/squid/logs/squid.log
```

ตัวแปรอื่นๆ

ตัวแปร cache_effective_user และ cache_effective_group จะระบุ user id และ group id ที่ Squid จะทำงาน ตัวอย่างเช่น

```
cache_effective_user squid
cache_effective_group daemon
```

ตัวแปร cache_mgr ใช้ระบุ email address ของผู้ควบคุมระบบ มีรูปแบบดังนี้

```
cache_mgr cachemaster@linux.intranet
```

การอนุญาตให้เครื่องลูกข่ายมาเชื่อมต่อ

การอนุญาตให้เครื่องลูกข่ายมาเชื่อมต่อมีตัวแปรที่ใช้ร่วมกันทั้งหมด 3 ตัวคือ acl, http_access และ icp_access

ตัวแปร acl ใช้ในการกำหนด ชื่อ, IP หรือวงแลนของเครื่องลูกข่ายที่จะมาเชื่อมต่อ, aclname คือ การกำหนดชื่อที่ไว้ใช้อ้างอิงถึงในตัวแปร http_access ซึ่งสามารถใช้ชื่อซ้ำได้ มีรูปแบบดังนี้

```
acl aclname src ip-address/netmask          # ระบุเป็นหมายเลข IP ของเครื่อง
acl aclname src addr1-addr2/netmask         # ระบุเป็นช่วงของ Subnet
acl aclname srcdomain hostname              # ระบุเป็นชื่อ
```

ตัวแปร http_access ใช้กำหนดว่า aclname ใดที่สามารถเชื่อมต่อเพื่อรับข้อมูล HTTP จากเครื่อง พร็อกซี่เซิร์ฟเวอร์ มีการกำหนด 2 แบบ คือ allow (อนุญาต) หรือ deny (ปฏิเสธ) เครื่องหมาย ! แทนการยกเว้น มีรูปแบบดังนี้

```
http_access allow|deny [!]aclname
```

ตัวแปร icp_accessใช้กำหนดว่า aclname ใดที่สามารถเชื่อมต่อเพื่อรับข้อมูล ICP จากเครื่องพร็อกซี่เซิร์ฟเวอร์ มีรูปแบบดังนี้

```
icp_access allow|deny [!]aclname
```

ตัวอย่าง ต้องการอนุญาตให้เครื่องลูกข่ายที่มี IP 204.150.154.20 และ IP วง 202.150.153.0 สามารถเชื่อมต่อกับพร็อกซี่เซิร์ฟเวอร์ได้ และต้องการอนุญาตให้เครื่อง cache.mychild.co.th มาใช้เครื่องเป็น parent หรือ sibling ได้

```
acl client_ip src 204.150.154.20
acl local_ip src 202.150.153.0-202.150.153.255/255.255.255.0
acl child_name srcdomain cache.mychild.co.th
http_access allow client_ip
http_access allow local_ip
http_access allow child_name
icp_access allow child_name
http_access deny all
icp_access deny all
```

การกำหนด icp_access จำเป็นต้องกำหนดสำหรับเครื่องลูกข่ายที่เป็นพร็อกซี่เซิร์ฟเวอร์เท่านั้น เพราะการรับส่งข้อมูลแบบ ICP จะสามารถทำได้เฉพาะเครื่องที่เป็นพร็อกซี่เซิร์ฟเวอร์ ส่วนบรรทัดสองบรรทัดล่าง คือการปฏิเสธไม่ให้เครื่องที่ไม่ได้กำหนดไว้มาเชื่อมต่อ เพื่อเป็นการป้องกันไม่ให้เครื่องอื่นมาแอบใช้ เนื่องจากค่าเริ่มต้นที่ Squid กำหนดจะอนุญาตให้ทุกเครื่องสามารถเข้าถึงได้

ถ้าต้องการให้เครื่องลูกข่ายสามารถเป็นได้เฉพาะ Sibling ไม่สามารถชี้เป็น parent ได้ ใช้ตัวแปร miss_access มีตัวอย่างการใช้ดังนี้

```
acl sibling_only src 203.150.154.20
miss_access allow !sibling_only
```

การกำหนดให้ Cache manager สามารถใช้งานได้

Cache manager คือ โปรแกรม CGI ที่ช่วยในการตรวจสอบการทำงานของ Squid โดยจะให้ข้อมูลที่สำคัญๆ เช่น การใช้ CPU, การใช้ memory เป็นต้นโปรแกรมนี้ชื่อว่า cachemgr.cgi จะมาพร้อมกับ Squid อยู่แล้ว การที่จะทำให้ cachemgr.cgi สามารถทำงานได้ต้องรันผ่านเว็บเซิร์ฟเวอร์ ดังนั้นต้องทำสำเนาไฟล์นี้ไปเก็บไว้ในไดเรกทอรี cgi-bin ของเครื่องเว็บเซิร์ฟเวอร์ และต้องกำหนดให้เครื่องเว็บเซิร์ฟเวอร์เครื่องนั้นสามารถเข้าถึงได้ โดยกำหนดได้ดังนี้

```
acl webmgr src 202.150.154.20          # กำหนด acl ที่ชื่อว่า webmgr แทน IP ของเว็บเซิร์ฟเวอร์
acl manager proto cache_object         # กำหนดโปรโตคอลที่ใช้
```

```
http_access deny manager !webmgr     # กำหนดให้ปฏิเสธไม่ให้ทุกเครื่องเข้าถึงได้ยกเว้น
                                       เครื่องที่มี IP 202.150.154.20 เท่านั้น
```

การเปิดให้บริการ Squid

เมื่อปรับค่าคอนฟิกในไฟล์ /etc/squid/squid.conf เรียบร้อยแล้ว เราสามารถเปิดการให้บริการพร็อกซี่เซิร์ฟเวอร์โดยใช้คำสั่ง service squid start ดังนี้

```
[root@linux root]# service squid start
init_cache_dir /var/spool/squid... Starting squid: . [ OK ]
[root@linux root]#
```

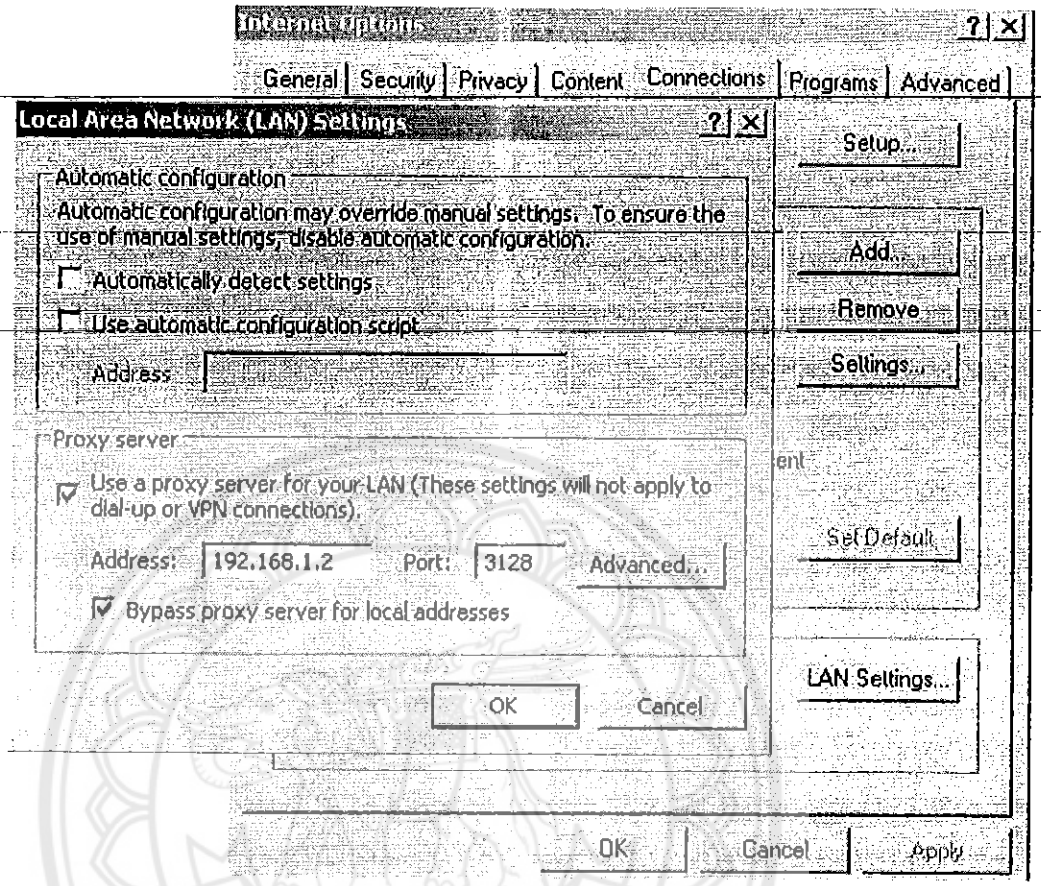
และบางครั้งถ้ามีการปรับปรุงค่าคอนฟิกใหม่ จำเป็นต้อง restart Squid ใหม่ด้วยโดยใช้คำสั่ง service squid restart ดังนี้

```
[root@linux root]# service squid restart
Stopping squid: . [ OK ]
Starting squid: . [ OK ]
[root@linux root]#
```

วิธีการกำหนดให้เครื่องลูกข่ายเรียกใช้งาน Squid

เครื่องลูกข่ายสามารถที่จะเรียกใช้ Squid ได้โดยการตั้งค่าคอนฟิกที่เบราเซอร์ เช่น IE (Internet Explorer) ดังนี้

1. ไปที่เมนู Tool > Internet Options > Connections > LAN Settings
2. ป้อนค่า IP Address ของเครื่องพร็อกซี่เซิร์ฟเวอร์และพอร์ตที่เปิดใช้งาน
3. คลิกที่ปุ่ม OK



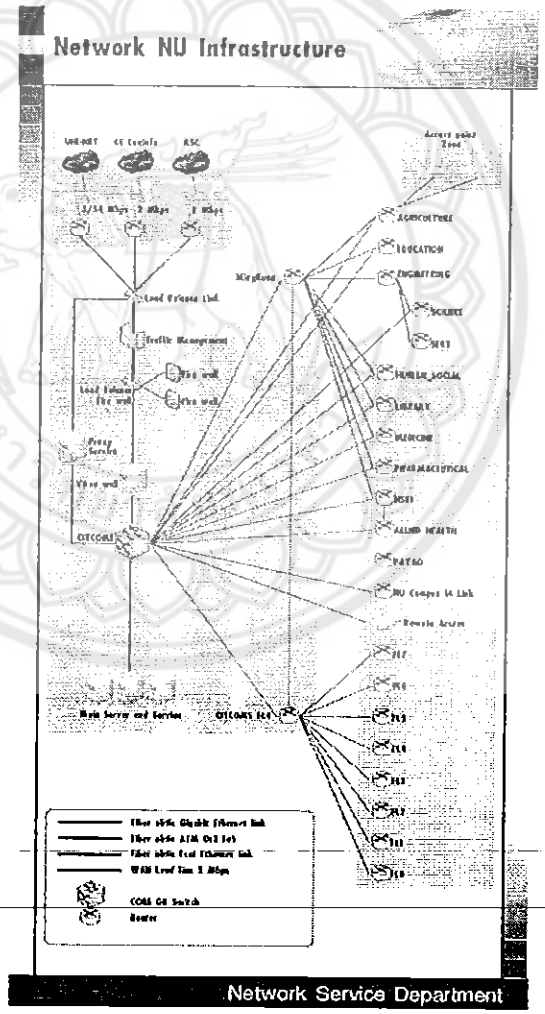
รูปที่ 2.58 การกำหนดให้เครื่องลูกข่ายใช้งานพร็อกซี่เซิร์ฟเวอร์
ที่มี IP Address 192.168.1.2 บนพอร์ต 3128

บทที่ 3

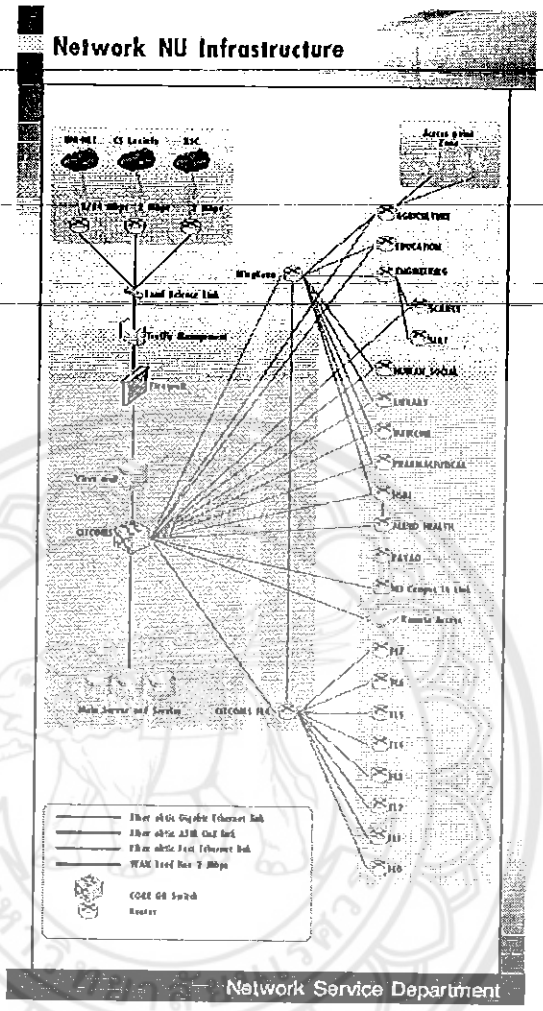
การดำเนินงาน

3.1 โครงสร้างและส่วนประกอบหลักของระบบ

ในการออกแบบไฟร์วอลล์ได้นำโครงสร้างพื้นฐานเครือข่ายของมหาวิทยาลัยนเรศวร ดังรูปที่ 3.1 แต่เปลี่ยนมาใช้ไฟร์วอลล์ประเภทสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์แทนไฟร์วอลล์ที่มีอยู่แบบเดิม ดังรูปที่ 3.2 มาใช้ในการทำเป็นโครงสร้างทางเครือข่ายของระบบทดสอบ

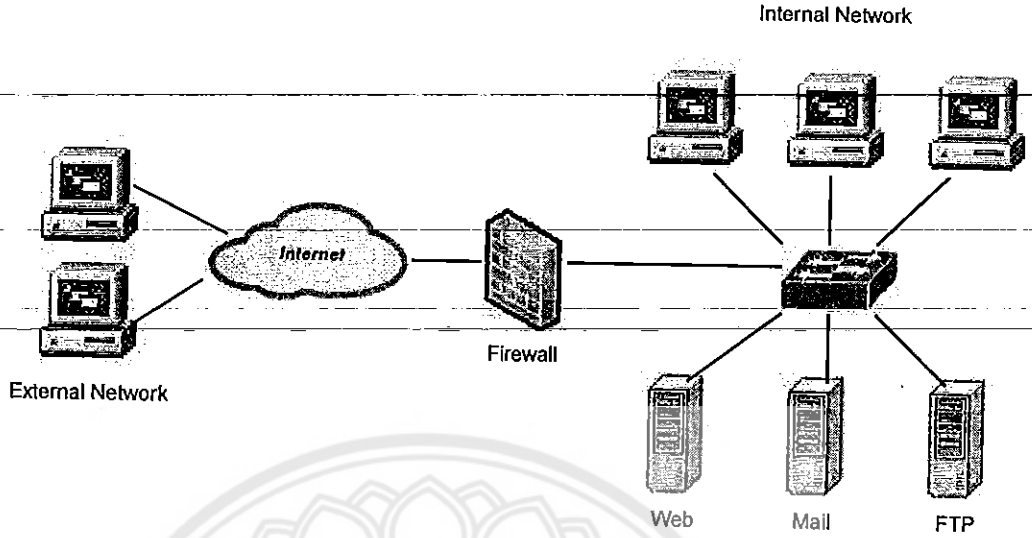


รูปที่ 3.1 โครงสร้างพื้นฐานเครือข่ายของมหาวิทยาลัยนเรศวร



รูปที่ 3.2 โครงสร้างพื้นฐานเครือข่ายของมหาวิทยาลัยนเรศวรที่ใช้
สเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์

จากโครงสร้างพื้นฐานของมหาวิทยาลัยนเรศวรที่เปลี่ยนมาใช้ไฟร์วอลล์ประเภทสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ สามารถออกแบบโครงสร้างทางเครือข่ายของระบบทดสอบได้ดังรูปที่ 3.3



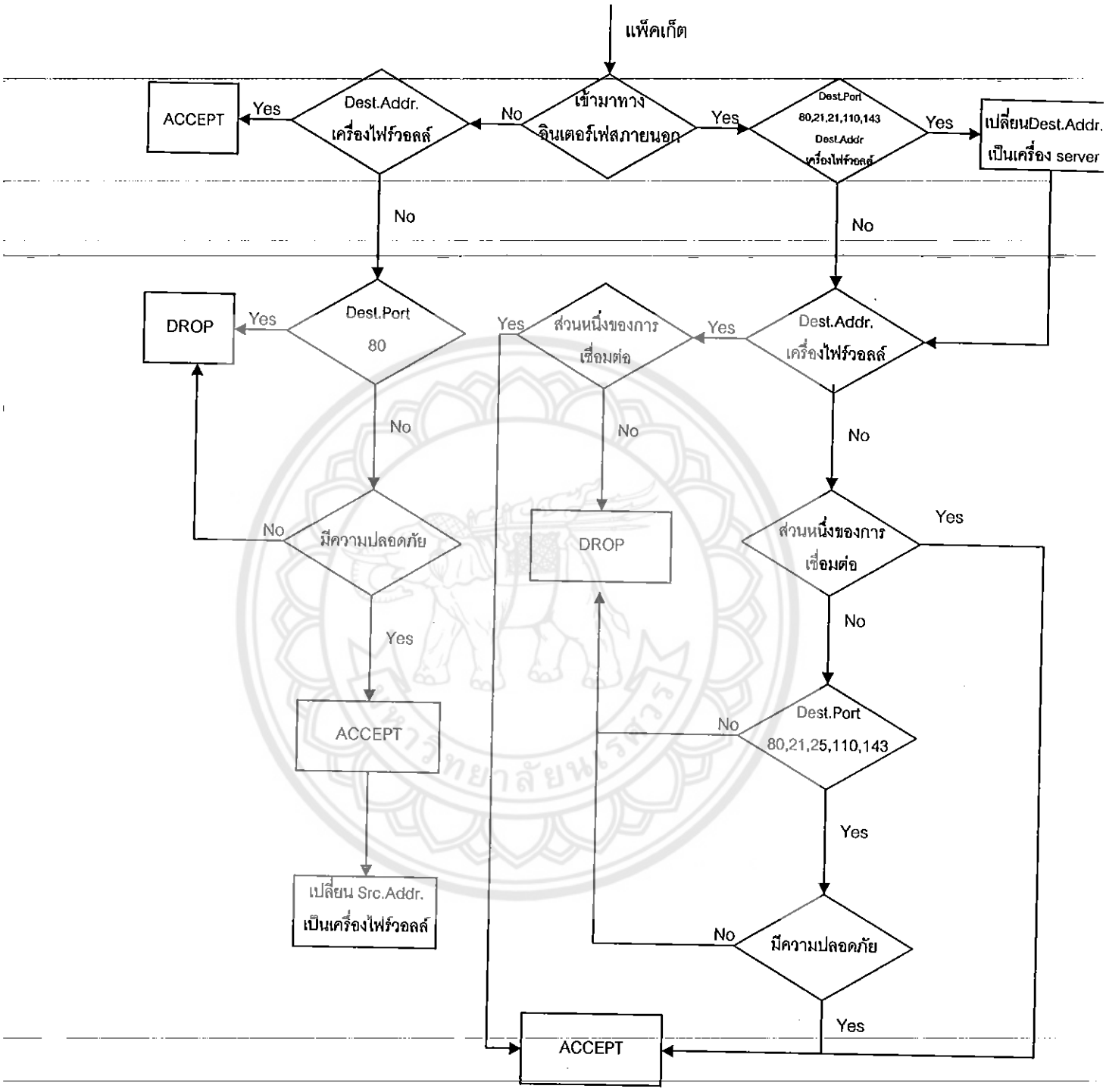
รูปที่ 3.3 โครงสร้างทางเครือข่ายของระบบทดสอบ

3.2 การออกแบบการทำงานของไฟร์วอลล์

การออกแบบการทำงานของไฟร์วอลล์ประเภทสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ได้แบ่งลักษณะการทำงานออกเป็น 2 ส่วน คือส่วนที่มีลักษณะการทำงานเป็นแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์และสเตทฟูลอินสเปคชั่นไฟร์วอลล์ กับอีกส่วนที่มีลักษณะการทำงานเป็นพร็อกซี่

3.2.1 การออกแบบการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ที่มีการทำงานเป็นแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์ และ สเตทฟูลอินสเปคชั่นไฟร์วอลล์

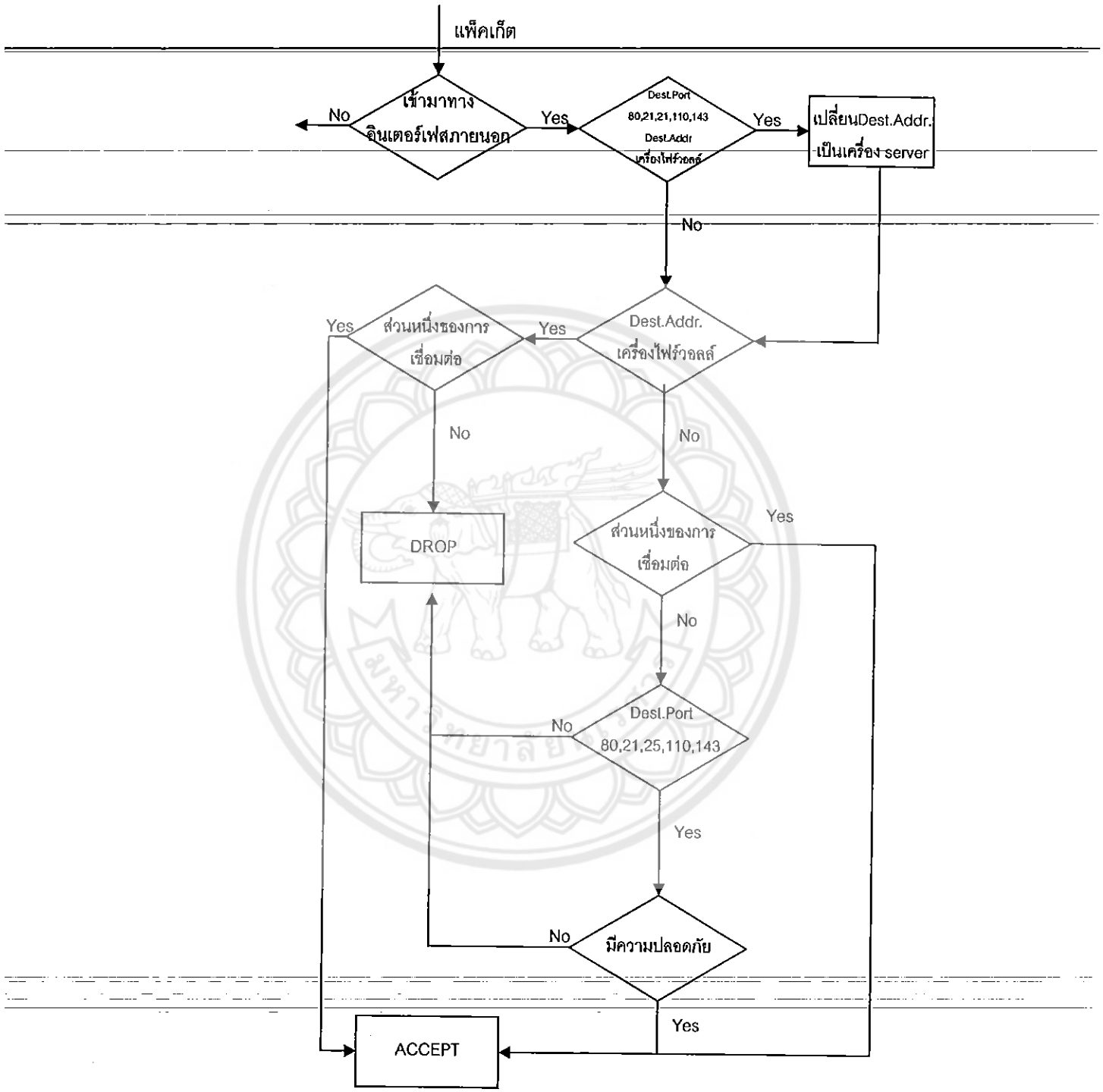
จากการออกแบบการทำงานและกำหนดกฎที่ใช้ในการควบคุมของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ที่มีการทำงานเป็นแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์ และ สเตทฟูลอินสเปคชั่นไฟร์วอลล์สามารถเขียนเป็นแผนผังการทำงานได้ ดังรูปที่ 3.4



รูปที่ 3.4 แผนผังการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ที่มีการทำงานเป็นแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์ และ สเตทฟูลอินสเปคชั่นไฟร์วอลล์

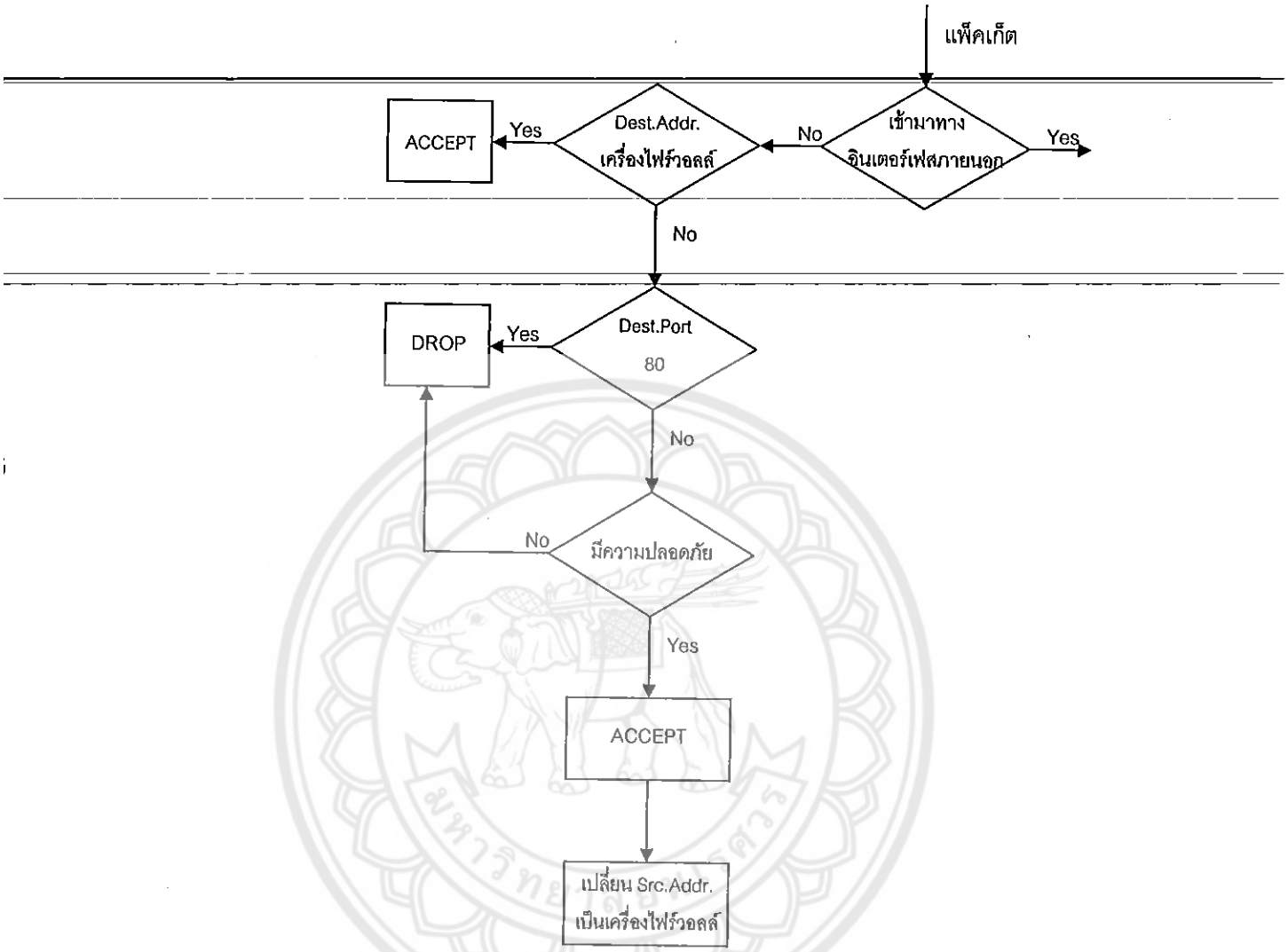
จากรูปที่ 3.4 แผนผังการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ที่มีการทำงานเป็นแพ็คเก็ตฟิลเตอริ่งไฟร์วอลล์และสเตทฟูลอินสเปคชั่นไฟร์วอลล์มีการทำงานดังนี้

1. เมื่อแพ็คเก็ตจากเครือข่ายภายนอกเข้ามาทางอินเตอร์เฟสภายนอก แพ็คเก็ตจะถูกตรวจสอบที่หมายเลข Destination Port ถ้าพบว่ามี Destination Port เป็นพอร์ต 80, 21, 25, 110 หรือ 143 และมี Destination IP Address เป็น IP Address ของเครื่องไฟร์วอลล์ จะเปลี่ยน Destination IP Address ของแพ็คเก็ตเป็น IP Address ของเครื่อง Web, FTP หรือ Mail Server แต่ถ้าแพ็คเก็ตมีหมายเลข Destination Port ไม่ใช่หมายเลข Destination Port ที่กำหนดไว้หรือมี Destination IP Address ไม่ใช่ IP Address ของเครื่องไฟร์วอลล์ จะไม่เปลี่ยนแปลงหรือแก้ไขแพ็คเก็ต หลังจากนั้นแพ็คเก็ตจะถูกตรวจสอบว่ามี Destination IP Address เป็นหมายเลข IP Address ใด ถ้ามี Destination IP Address เป็น IP Address ของเครื่องไฟร์วอลล์จะทำการตรวจสอบสถานะของแพ็คเก็ต ถ้ามีสถานะเป็นส่วนหนึ่งของการเชื่อมต่อที่สร้างไว้แล้วจะยอมรับแพ็คเก็ตนั้น แต่ถ้าไม่ใช่จะถูกดร็อป ในกรณีที่แพ็คเก็ตมี Destination IP Address ไม่ใช่ IP Address ของเครื่องไฟร์วอลล์ จะถูกตรวจสอบก่อนฟอร์เวิร์ดออกไป โดยตรวจสอบสถานะของแพ็คเก็ต ถ้ามีสถานะเป็นส่วนหนึ่งของการเชื่อมต่อที่สร้างไว้แล้วจะยอมรับแพ็คเก็ตนั้นโดยทำการฟอร์เวิร์ดแพ็คเก็ตออกไป แต่ถ้ามีสถานะไม่ใช่สถานะดังกล่าว จะตรวจสอบ Destination Port ของแพ็คเก็ต ถ้าแพ็คเก็ตมี Destination Port ที่ไม่ได้อนุญาตไว้จะถูกดร็อป แต่ถ้าเป็นพอร์ตที่อนุญาตไว้คือพอร์ต 80, 21, 25, 110 หรือ 143 ก็จะถูกตรวจสอบว่าเป็นตามกฎที่กำหนดไว้หรือไม่ ถ้าตรวจสอบแล้วพบว่ามีความปลอดภัยต่อระบบจะยอมรับแพ็คเก็ตนั้นโดยทำการฟอร์เวิร์ดแพ็คเก็ตออกไป แต่ถ้าพบว่าไม่มีความปลอดภัยต่อระบบแพ็คเก็ตนั้นก็จะถูกดร็อป ดังรูปที่ 3.5



รูปที่ 3.5 แผนผังการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์
เมื่อแพ็คเก็ตมาจากเครือข่ายภายนอก

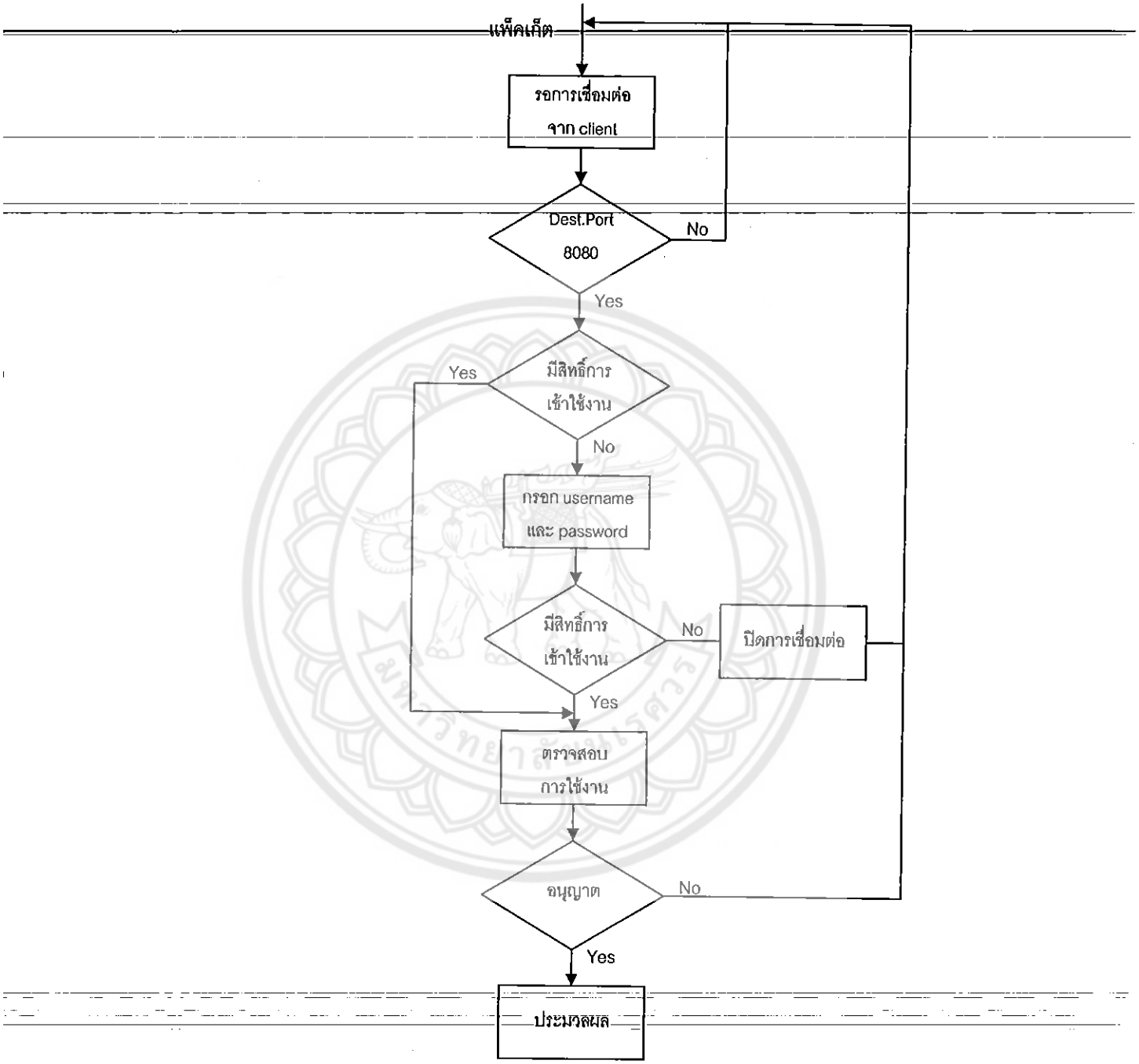
2. เมื่อแพ็คเก็ตจากเครือข่ายภายในเข้ามาทางอินเตอร์เฟสภายใน แพ็คเก็ตจะถูกตรวจสอบ Destination IP Address ว่ามี Destination IP Address เป็นหมายเลข IP Address ใด ถ้ามี Destination IP Address เป็น IP Address ของเครื่องไฟร์วอลล์จะยอมรับแพ็คเก็ตนั้น แต่ถ้าแพ็คเก็ตมี Destination IP Address ไม่ใช่ IP Address ของเครื่องไฟร์วอลล์ จะถูกตรวจสอบ Destination Port ถ้ามี Destination Port เป็นพอร์ต 80 จะถูกดร็อปเพื่อห้ามไม่ให้เครื่องลูกข่ายใช้บริการเว็บโดยไม่ใช้พร็อกซี่ แต่ถ้ามี Destination Port ที่ไม่ใช่พอร์ต 80 จะถูกตรวจสอบว่าเป็นตามกฎที่กำหนดไว้หรือไม่ ถ้าตรวจสอบแล้วไม่มีความปลอดภัยต่อระบบจะถูกดร็อป แต่ถ้าตรวจสอบแล้วมีความปลอดภัยต่อระบบจะยอมรับแพ็คเก็ตนั้น และถ้าแพ็คเก็ตนั้นได้รับการยอมรับจะถูกเปลี่ยน Source IP Address เป็น IP Address ของเครื่องไฟร์วอลล์ก่อนฟอร์เวิร์ดแพ็คเก็ตออกไป ดังรูปที่ 3.6



รูปที่ 3.6 แผนผังการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์เมื่อแพ็คเก็ตมาจากเครือข่ายภายใน

3.2.2 การออกแบบการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ที่มีการทำงานเป็นพร็อกซี่

จากการออกแบบการทำงานและกำหนดกฎที่ใช้ในการควบคุมของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ที่มีการทำงานเป็นพร็อกซี่สามารถเขียนเป็นแผนผังการทำงานได้ ดังรูปที่ 3.7



รูปที่ 3.7 แผนผังการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์
ที่มีการทำงานเป็นพร็อกซี่

โดยการทำงานของสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์ที่มีการทำงานเป็นพร็อกซี่นั้นจะทำการรอรับการร้องขอที่พอร์ต 8080 โดยเครื่องลูกข่ายที่ต้องการเชื่อมต่อจะต้องทำการติดตั้งพร็อกซีแอดเดรสมาที่เครื่องไฟร์วอลล์และพอร์ต 8080 จึงสามารถใช้งานพร็อกซี่ได้ เมื่อเครื่องลูกข่ายทำการเรียกใช้บริการเว็บ จะถูกตรวจสอบว่ามีสิทธิ์การใช้งานหรือไม่ ถ้ายังไม่มีสิทธิ์การใช้งาน ผู้ใช้ต้องแสดงตนและถูกตรวจสอบสิทธิ์การใช้งาน ถ้าไม่มีสิทธิ์จะถูกปิดการเชื่อมต่อทันที แต่ถ้าผู้ใช้มีสิทธิ์การใช้งานจะถูกตรวจสอบการใช้งาน ถ้าเครื่องลูกข่ายใช้บริการเว็บที่ไม่อนุญาตก็จะถูกยกเลิกการประมวลผล แต่ถ้าเป็นเว็บที่ได้รับอนุญาตจะทำการประมวลผลแล้วส่งให้กับเครื่องลูกข่ายตามที่ร้องขอ

บทที่ 4

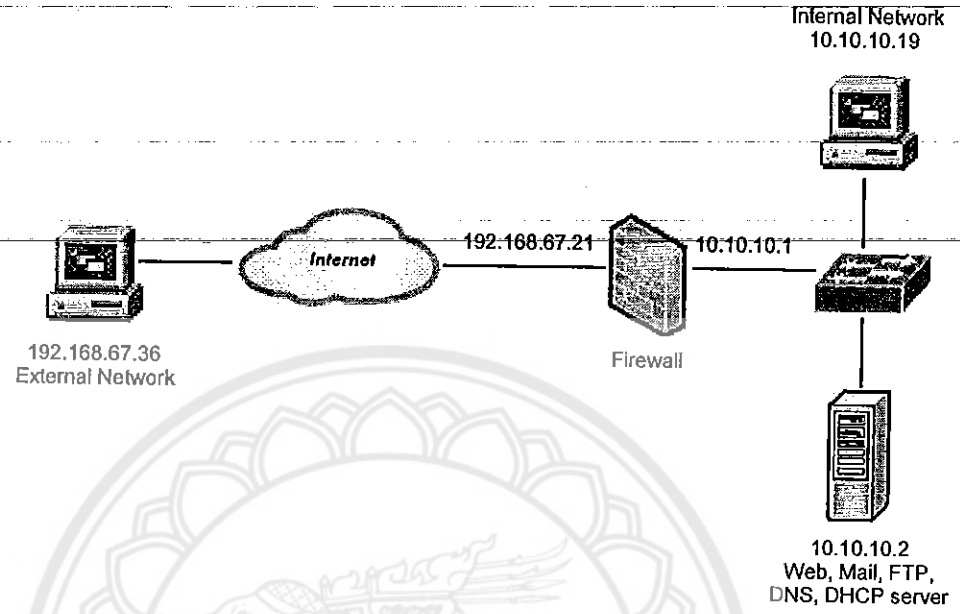
ผลการทดลอง

4.1 เครื่องที่ใช้ในการทดสอบไฟร์วอลล์

มีรายละเอียดของคอมพิวเตอร์ที่ใช้ในการทดสอบดังนี้

1. เครื่องที่ใช้ติดตั้งไฟร์วอลล์ (Firewall)
   - หน่วยประมวลผลความเร็ว 1.2 GHz
   - หน่วยความจำหลัก 128 MB
   - ระบบปฏิบัติการ Linux RedHat 9.0
   - การ์ดแลนความเร็ว 100 Mbps 2 การ์ด
2. เครื่องแม่ข่าย (Server)
   - หน่วยประมวลผลความเร็ว 350 MHz
   - หน่วยความจำหลัก 64 MB
   - ระบบปฏิบัติการ Linux RedHat 9.0
   - การ์ดแลนความเร็ว 100 Mbps
3. เครื่องภายในเครือข่าย (Internal Network)
   - หน่วยประมวลผลความเร็ว 350 MHz
   - หน่วยความจำหลัก 64 MB
   - ระบบปฏิบัติการ Linux RedHat 9.0
   - การ์ดแลนความเร็ว 100 Mbps
4. เครื่องภายนอกเครือข่าย (External Network)
   - หน่วยประมวลผลความเร็ว 350 MHz
   - หน่วยความจำหลัก 192 MB
   - ระบบปฏิบัติการ Windows XP
   - การ์ดแลนความเร็ว 100 Mbps

4.2 การทดสอบการใช้งาน



รูปที่ 4.1 โครงสร้างของระบบทดสอบ

กำหนดให้

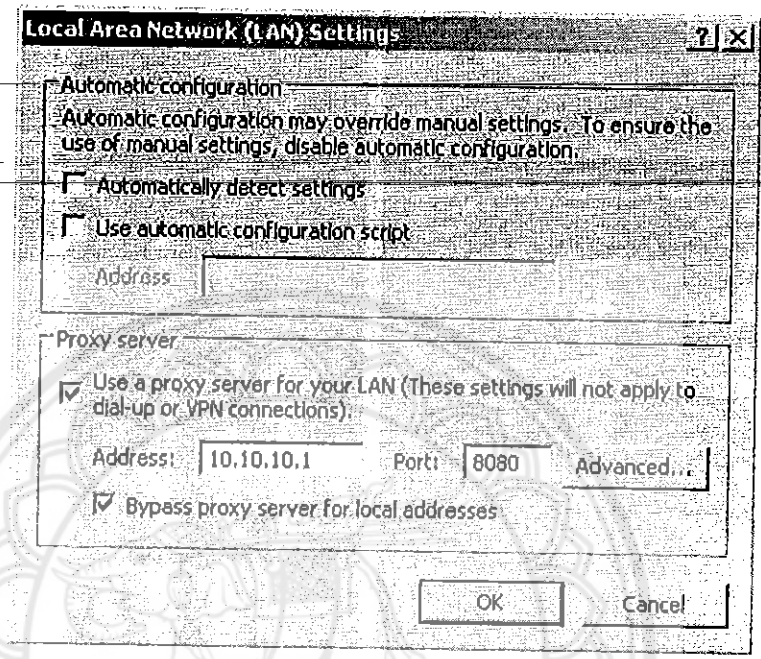
- *เครื่องไฟร์วอลล์* เป็นเครื่องที่กั้นระหว่างเครือข่ายภายใน (10.10.10.0) กับเครือข่ายภายนอก (192.168.67.0) โดยมีแอดเดรสของอินเตอร์เฟสภายในเป็น 10.10.10.1 และแอดเดรสของอินเตอร์เฟสภายนอกเป็น 192.168.67.21
- *เครื่องแม่ข่าย* มี IP Address เป็น 10.10.10.2 โดยให้บริการ Web, Mail, FTP, DNS และ DHCP ตั้งเกตเวย์ไปที่เครื่องไฟร์วอลล์
- *เครื่องภายในเครือข่าย* มี IP Address เป็น 10.10.10.19
- *เครื่องภายนอกเครือข่าย* มี IP Address เป็น 192.168.67.36

4.2.1 การใช้งานของเครื่องภายในเครือข่าย

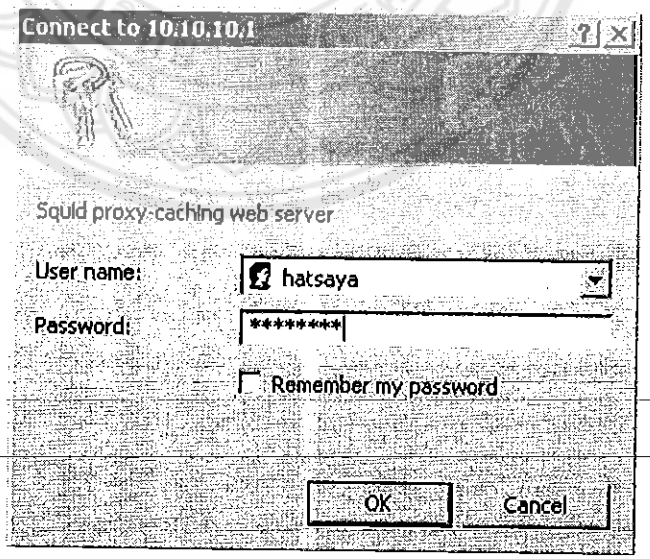
- **การใช้งานเว็บ**

ถ้าเครื่องภายในเครือข่ายจะใช้บริการเว็บจะต้องแก้ไขแอพพลิเคชั่นให้ใช้งานพร็อกซี่โดยการติดตั้งพร็อกซี่แอดเดรสไปที่เครื่องไฟร์วอลล์และใช้พอร์ต 8080 ดังรูปที่ 4.2 ถ้าผู้ใช้งานไม่ทำการแก้ไขแอพพลิเคชั่นให้ใช้งานพร็อกซี่ก็จะไม่สามารถใช้บริการเว็บได้ และเมื่อแก้ไขแอพพลิเคชั่นให้ใช้งานพร็อกซี่แล้ว ผู้ใช้ต้องแสดงสิทธิในการใช้งานโดยการใส่ชื่อผู้ใช้ (Username) และรหัสผ่าน

(Password) ก่อน ดังรูปที่ 4.3 จึงจะใช้งานเว็บได้ แต่จะไม่สามารถใช้งานเว็บที่ถูกบล็อกไว้ได้



รูปที่ 4.2 การติดตั้งพร็อกซี่แอดเดรสของเครื่องภายในเครือข่าย



รูปที่ 4.3 การแสดงสิทธิการใช้งานเว็บของเครื่องภายในเครือข่าย

- **การใช้งานบริการอื่นๆ**

การใช้งานบริการอื่นที่มีความปลอดภัยต่อระบบเครือข่าย เปิดให้ใช้งานได้ทั้งหมด ดังนั้นผู้ใช้งานภายในเครือข่ายก็จะสามารถใช้งานบริการดังกล่าวได้ตามปกติ

4.2.2 การใช้งานของเครื่องภายนอกเครือข่าย

- **การใช้งานเว็บ เมล์ และ เอฟทีพี**

การให้บริการเว็บ เมล์ และ เอฟทีพี เป็นบริการที่ไฟร์วอลล์เปิดให้บริการและอนุญาตให้เครื่องภายนอกเครือข่ายใช้งาน ดังนั้นผู้ใช้งานที่อยู่ภายนอกเครือข่ายจึงสามารถใช้งานบริการเหล่านี้ได้

- **การใช้งานบริการอื่นๆ**

การใช้งานบริการอื่นๆ นั้น ไฟร์วอลล์ไม่ได้เปิดบริการไว้สำหรับเครื่องที่อยู่ภายนอกเครือข่าย ดังนั้นผู้ใช้งานที่อยู่ภายนอกเครือข่ายจะไม่สามารถใช้งานบริการเหล่านี้ได้

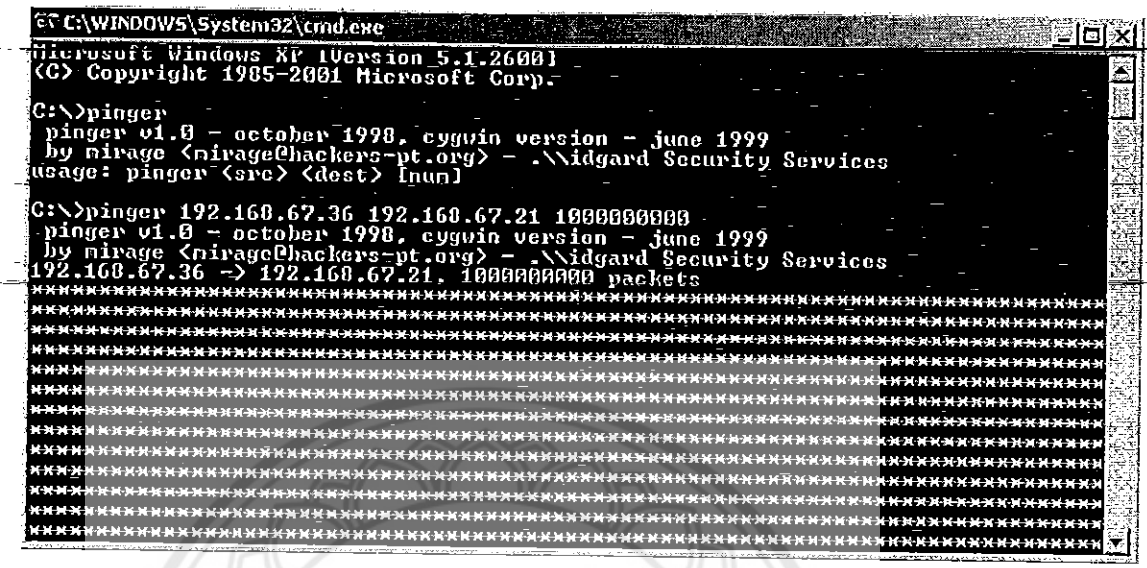
4.3 การทดสอบการโจมตี

เนื่องจากในปัจจุบันได้มีผู้พัฒนาเครื่องมือที่ใช้ในการโจมตีเพื่อให้ปฏิเสธการให้บริการขึ้นมากมาย ส่วนใหญ่สามารถดาวน์โหลดได้ฟรีจากอินเตอร์เน็ต ความแตกต่างกันของเครื่องมือแต่ละตัวคือรูปแบบการโจมตีและระดับความรุนแรงในการโจมตี

4.3.1 โจมตีด้วย Pingflood

การโจมตีประเภทนี้อาศัยการส่งแพ็คเก็ตในปริมาณมีลักษณะการโจมตี คือส่ง ICMP Echo Request ไปยังเป้าหมายมากๆ ในระยะเวลาติดต่อกัน ทำให้เป้าหมายต้องคอยตอบ ICMP Echo Reply ตลอดเวลาจนไม่สามารถให้บริการอย่างอื่นได้

การทดสอบการโจมตีด้วย Pingflood ดังรูปที่ 4.4



รูปที่ 4.4 การทดสอบโจมตีด้วย Pingflood

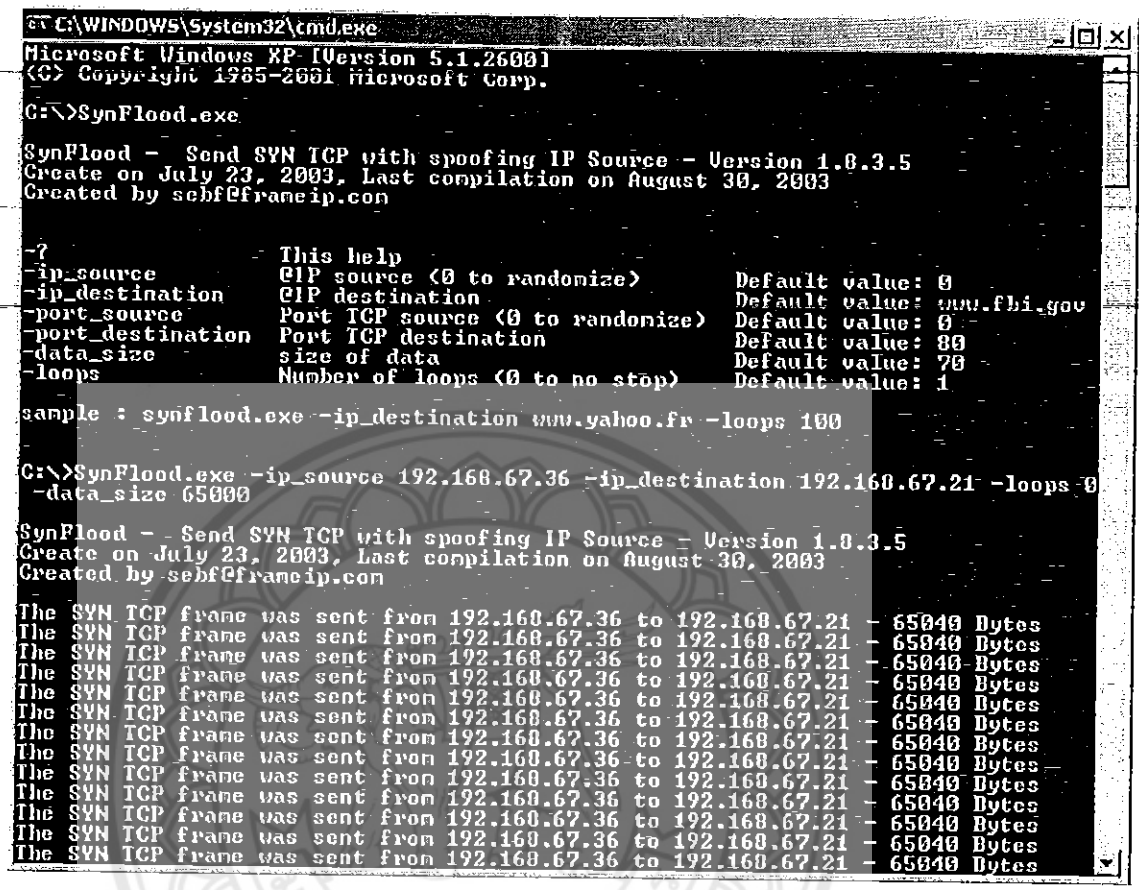
ผลการทดสอบ

- กรณีโจมตีไปที่พอร์ตที่ไม่ได้อนุญาตไว้ เครื่องไฟร์วอลล์และเครื่องแม่ข่ายทำงานได้ตามปกติ
- กรณีโจมตีไปที่พอร์ตที่ได้รับอนุญาต เครื่องไฟร์วอลล์และเครื่องแม่ข่ายทำงานได้ตามปกติ

4.3.2 โจมตีด้วย Synflood

การโจมตีประเภทนี้มีลักษณะการโจมตี คือเครื่องโจมตีส่งสัญญาณ SYN ไปที่เครื่องเป้าหมาย เมื่อได้รับสัญญาณ ACK จากเครื่องเป้าหมายแล้วไม่ส่งสัญญาณ SYN ตอบกลับไป ทำให้เครื่องเป้าหมายต้องเปิดการเชื่อมต่อรอการตอบกลับตลอดเวลา ซึ่งถ้ามีการส่งสัญญาณในลักษณะนี้มากๆจะทำให้ทรัพยากรของระบบมีไม่เพียงพอ อาจทำให้ระบบไม่สามารถให้บริการอย่างอื่นได้

การทดสอบการโจมตีด้วย Synflood ดังรูปที่ 4.5



รูปที่ 4.5 การทดสอบโจมตีด้วย Synflood

ผลการทดสอบ

- กรณีโจมตีไปที่พอร์ต 80 (เปิดพอร์ต 80 สำหรับบริการเว็บ) เครื่องไฟร์วอลล์ทำงานปกติ ส่วนเครื่องแม่ข่ายมีการตอบสนองช้า
- กรณีโจมตีพอร์ตที่ไม่ได้อนุญาตไว้ เครื่องไฟร์วอลล์และเครื่องแม่ข่ายทำงานปกติ

4.3.3 โจมตีด้วย UDP Bomb

UDP Bomb เป็นการโจมตีโดยอาศัยข้อบกพร่องของ UDP เพื่อทำให้เครื่องแม่ข่ายเป้าหมายรับส่ง UDP Datagram ที่มีขนาดผิดปกติจาก UDP ทั่วๆไป ถ้าระบบปฏิบัติการของเครื่องแม่ข่ายเป้าหมายไม่สามารถจัดการกับแพ็คเก็ตประเภทนี้ได้ดีก็จะทำให้เครื่องแม่ข่ายเป้าหมายหยุดทำงานได้ทันที

ผลการทดสอบ

- การโจมตีไปที่พอร์ตที่ได้รับอนุญาตหรือถูกบล็อกเอาไว้ เครื่องไฟร์วอลล์และเครื่องแม่ข่ายสามารถทำงานได้ตามปกติ

4.3.4 โจมตีด้วย Jolt

Jolt เป็นการโจมตีโดยการส่งแฟรกเมนต์แพ็คเก็ตซ้ำๆจำนวนมากอย่างต่อเนื่องมายังเครื่องแม่ข่ายเป้าหมาย ทำให้เครื่องแม่ข่ายเป้าหมายไม่มีการตอบสนองใดๆ เนื่องจากกำลังของ CPU ถูกใช้ไปในการจัดการกับแฟรกเมนต์ทั้งหมด

ผลการทดสอบ

- ไม่ว่าพอร์ตที่ถูกโจมตีนั้นจะเป็นพอร์ตที่ได้รับอนุญาตหรือถูกบล็อกเอาไว้ เครื่องไฟร์วอลล์และเครื่องแม่ข่ายสามารถทำงานได้ตามปกติ

4.3.5 โจมตีด้วย Smurf

Smurf Attack เป็นการโจมตีโดยการส่ง ICMP Echo Request ไปยังบรอดคาสต์แอดเดรสทำให้โฮสต์ทั้งหมดในเครือข่ายขณะนั้นทำการตอบกลับมายัง IP Address ที่ระบุว่าเป็นเป็นของผู้ส่ง การโจมตีต้องปลอม IP Address โดยให้ IP Address ต้นทางของ ICMP Echo Request ที่ส่งไปนั้นเป็นIP Address ของเป้าหมายที่จะโจมตี ทำให้เป้าหมายได้รับ ICMP Echo Reply จำนวนมากจนไม่สามารถสื่อสารกับผู้อื่นบนเครือข่ายได้

ผลการทดสอบ

- ไม่ว่าพอร์ตที่ถูกโจมตีนั้นจะเป็นพอร์ตที่ได้รับอนุญาตหรือถูกบล็อกเอาไว้ เครื่องไฟร์วอลล์และเครื่องแม่ข่ายสามารถทำงานได้ตามปกติ

การใช้งานบริการต่างๆ ที่มีความปลอดภัยต่อระบบ ผู้ใช้งานภายในเครือข่ายสามารถใช้งานได้ตามปกติ สำหรับการใช้งานเว็บของผู้ใช้งานภายในเครือข่าย ผู้ใช้ต้องติดตั้งพร็อกซี่แอดเดรสและต้องแสดงสิทธิในการใช้งานโดยใส่ชื่อผู้ใช้และรหัสผ่านก่อนจึงจะใช้งานได้ แต่จะไม่สามารถใช้บริการเว็บที่ไม่อนุญาตได้ ส่วนการใช้งานของเครื่องภายนอกเครือข่าย สามารถใช้งานได้เฉพาะบริการที่อนุญาตไว้เท่านั้น ส่วนการใช้งานอื่นๆ จะไม่สามารถใช้งานได้

การทดสอบการโจมตีเพื่อให้ปฏิเสธการให้บริการ ไฟร์วอลล์สามารถป้องกันการโจมตีที่โจมตีมาทางพอร์ตที่ไม่เปิดให้บริการได้ โดยที่เครื่องไฟร์วอลล์และเครื่องแม่ข่ายทำงานได้ตามปกติ กรณีที่โจมตีที่พอร์ตที่เปิดให้บริการไฟร์วอลล์ไม่สามารถป้องกันได้ แต่สามารถลดความรุนแรงจากการโจมตีได้ ทำให้เครื่องไฟร์วอลล์ทำงานได้ตามปกติ ส่วนเครื่องแม่ข่ายจะมีการตอบสนองช้า

บทที่ 5

สรุปผลการทดลองและข้อเสนอแนะ

วัตถุประสงค์ของโครงงานนี้ คือการพัฒนาไฟร์วอลล์เพื่อป้องกันและรักษาความปลอดภัยให้กับระบบเครือข่ายขององค์กรที่เชื่อมต่อกับอินเตอร์เน็ต โดยไฟร์วอลล์ที่พัฒนาขึ้นเป็นไฟร์วอลล์ประเภทสเตทฟูลมัลติเลเยอร์อินสเปคชั่นไฟร์วอลล์

ในการพัฒนาไฟร์วอลล์สิ่งที่สำคัญที่สุด คือ ไฟร์วอลล์ที่พัฒนาขึ้นสามารถป้องกันและรักษาความปลอดภัยให้กับระบบเครือข่ายขององค์กรได้ดียิ่งขึ้น ในการพัฒนาไฟร์วอลล์ให้มีประสิทธิภาพดีขึ้นต้องมีความรู้ความเข้าใจโครงสร้างและลักษณะการทำงานของไฟร์วอลล์ โปรโตคอลที่ใช้ในการรับส่งข้อมูล โครงสร้างพื้นฐานเครือข่ายขององค์กร และโดยเฉพาะอย่างยิ่งนโยบายความปลอดภัย (Security Policy) ที่เหมาะสมกับองค์กร

5.1 ปัญหาและอุปสรรค

ในการออกแบบและพัฒนาไฟร์วอลล์มีปัญหาและอุปสรรคในการพัฒนา ได้แก่

- โปรแกรมที่ใช้ในการโจมตีประเภท Denial of Services มีอยู่มากมาย ทำให้การทดสอบการโจมตียังไม่สามารถทดสอบครอบคลุมทุกรูปแบบการโจมตีที่มีอยู่ในปัจจุบันและที่กำลังจะเพิ่มขึ้นในอนาคต
- ไฟร์วอลล์ไม่สามารถป้องกันการโจมตีที่เข้ามาทางพอร์ตที่เปิดให้บริการได้

5.2 ข้อเสนอแนะ

การพัฒนาไฟร์วอลล์ให้สามารถป้องกันและรักษาความปลอดภัยให้กับระบบเครือข่ายได้อย่างมีประสิทธิภาพ ควรใช้เวลาในการทดสอบให้มาก โดยทดสอบทั้งการใช้งานทั่วไปของเครื่องภายนอกเครือข่ายและเครื่องภายในเครือข่าย และทดสอบการโจมตีให้ครอบคลุมทุกรูปแบบมากที่สุด

5.3 แนวทางการพัฒนาต่อในอนาคต

ในการนำไฟร์วอลล์ที่สร้างขึ้นไปพัฒนาต่อนั้น ควรศึกษารายละเอียดการทำงานของไฟร์วอลล์และระบบปฏิบัติการ Linux ให้เข้าใจ เนื่องจากไฟร์วอลล์ที่สร้างขึ้นเป็นการพัฒนาบนระบบปฏิบัติการ Linux และควรใช้เวลาในการทดสอบไฟร์วอลล์ให้มากขึ้น เพื่อให้ทราบถึงปัญหาและข้อบกพร่องต่างๆ ของไฟร์วอลล์ และนำไปปรับปรุงไฟร์วอลล์ให้ดีขึ้น

เอกสารอ้างอิง

[1] เรืองไกร รังสิพล. **เปิดโลก Firewall และ Internet Security**. กรุงเทพฯ : บริษัท โปรวิชั่น จำกัด. 2545.

[2] สุวัฒน์ ปุณณชัยยะ. สุพจน์ ปุณณชัยยะ. ต้น ตัณฑ์สุทธิวงศ์. **เปิดโลก TCP/IP และโปรโตคอลของอินเตอร์เน็ต**. กรุงเทพฯ : บริษัท โปรวิชั่น จำกัด. 2545.

[3] เรืองไกร รังสิพล. เจาะระบบ TCP/IP จุดอ่อนของโปรโตคอลและวิธีป้องกัน. กรุงเทพฯ : บริษัท โปรวิชั่น จำกัด. 2544.

[4] ภัทระ เกียรติเสวี และทีมงานห้องปฏิบัติการเครือข่ายคอมพิวเตอร์(NTL). **สร้างอินเทอร์เน็ตเซิร์ฟเวอร์ด้วย Linux**. กรุงเทพฯ : บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน). 2542.

[5] ธวัชชัย ชมศิริ. ติดตั้ง/ดูแลระบบเครือข่ายคอมพิวเตอร์อย่างมืออาชีพ. กรุงเทพฯ : บริษัท ซีเอ็ดยูเคชั่นจำกัด(มหาชน). 2547.

ประวัติผู้เขียนโครงงาน

ชื่อ นายศรายุทธ หอยสังข์

วันเดือนปีเกิด 25 พฤศจิกายน 2525

ภูมิลำเนา 87/142 ถ.วังจันทน์ ต.ในเมือง อ.เมือง จ.พิษณุโลก

ประวัติการศึกษา

- จบระดับมัธยมศึกษาตอนต้น จากโรงเรียนพิษณุโลกพิทยาคม
- จบระดับมัธยมศึกษาตอนปลาย จากโรงเรียนพิษณุโลกพิทยาคม
- เข้าศึกษาระดับอุดมศึกษา ปีการศึกษา 2544
- ประธานชมรม Computer & IT สโมสรนิสิตคณะวิศวกรรมศาสตร์ ปีการศึกษา 2545
- นิสิตเรียนดี ปีการศึกษา 2545
- ประธานชมรม Computer & IT สโมสรนิสิตคณะวิศวกรรมศาสตร์ ปีการศึกษา 2546
- นิสิตเรียนดี ปีการศึกษา 2546
- นิสิตดีเด่นด้านกิจกรรม ปีการศึกษา 2546
- ทุนช่วยเหลืองาน คณะวิศวกรรมศาสตร์ ปีการศึกษา 2546
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4
  สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
  มหาวิทยาลัยนเรศวร

E-mail : g_team25@hotmail.com

ชื่อ นางสาวหัสยา โตมณีพิทักษ์

วันเดือนปีเกิด 26 มีนาคม 2526

ภูมิลำเนา 132 ถ.รักการดี ต.อุทัยใหม่ อ.เมือง จ.อุทัยธานี

ประวัติการศึกษา

- จบระดับมัธยมศึกษาตอนต้น จากโรงเรียนอุทัยวิทยาคม
- จบระดับมัธยมศึกษาตอนปลาย จากโรงเรียนอุทัยวิทยาคม
- เข้าศึกษาระดับอุดมศึกษา ปีการศึกษา 2544
- เลขานุการชมรม Computer & IT สโมสรนิสิตคณะวิศวกรรมศาสตร์ ปีการศึกษา 2545
- รองประธานชมรม Computer & IT สโมสรนิสิตคณะวิศวกรรมศาสตร์ ปีการศึกษา 2546
- ทุนนิสิตเรียนดีและมีน้ำใจ ปีการศึกษา 2546
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail : nongair99@hotmail.com