โครงงาน ระบบตรวจสอบและเฝ้าดูแพ็กเกจในระบบเครือข่ายท้องถิ่น

Local Computer Network Monitoring and Analysis

โดย

นายพงษ์เกียรติ เพชรกาศ 36014276

นายอนันตศักดิ์ ทิพย์พญาชัย 36014537

อาจารย์ที่ปรึกษา

อาจารย์ ธนา หงษ์สุวรรณ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2539

ปีการศึกษา 2539

โครงงาน ระบบตรวจสอบและเฝ้าดูแพ็กเกจในระบบเครือข่ายท้องถิ่น

Local Computer Network Monitoring and Analysis

โดย

นายพงษ์เกียรติ เพชรกาศ รหัสประจำตัวนักศึกษา 36014276

นายอนันตศักดิ์ ทิพย์พญาชัย รหัสประจำตัวนักศึกษา 36014537

อาจารย์ที่ปรึกษา

อาจารย์ ธนา หงษ์สุวรรณ

ปริญญานิพนธ์ปีการศึกษา 2539

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบตรวจสอบและเฝ้าดูแพ็กเกจในระบบเครือข่ายท้องถิ่น

ผู้จัดทำ

1. นาย พงษ์เกียรติ เพชรกาศ รหัสประจำตัวนักศึกษา 36014276
2. นาย อนันตศักดิ์ ทิพย์พญาชัย รหัสประจำตัวนักศึกษา 36014537

.....อาจารย์ที่ปรึกษา

(อาจารย์ ธนา หงษ์สุวรรณ)

โครงงานระบบตรวจสอบและเฝ้าดูแพ็กเกจในระบบเครือข่ายท้องถิ่น

พงษ์เกียรติ เพชรกาศ
อนันตศักดิ์ ทิพย์พญาชัย
อ. ธนา หงษ์สุวรรณ อาจารย์ที่ปรึกษา
ปีการศึกษา 2539

บทคัดย่อ

ในปัจจุบันการใช้งานคอมพิวเตอร์ (Computer) ในรูปแบบเครือข่าย (Network) ได้เข้ามามีบทบาทเพื่อช่วยเพิ่มประสิทธิภาพในการทำงานมากยิ่งขึ้นโดยเฉพาะองค์กรใหญ่ ๆ จะใช้ประโยชน์จากการใช้ทรัพยากรร่วมกันหรือการทำงานเป็นระบบเครือข่าย จึงได้พัฒนาโปรแกรม (Program) ขึ้นเพื่อใช้เฝ้าดูและวิเคราะห์แพ็กเกจ (Packet) ในระบบเครือข่ายท้องถิ่น (LAN:Local Area Network) โดยตรวจสอบแพ็กเกจที่ได้รับจากสื่อระบบ ผ่านการทำงานของแพ็กเกจไดรเวอร์ (Packet Driver) ซึ่งจะรับแพ็กเกจจากการ์ดเชื่อมต่อเน็ตเวิร์ค (NIC:Network Interface Card) เข้ามาวิเคราะห์ถึงส่วนต่างๆที่จำเป็นต่อการศึกษาการทำงาน และ ตรวจสอบระบบเครือข่าย โดยวิเคราะห์ถึง ปริมาณการใช้งาน (Utilization) คอนเวอร์เซชั่น (Conversation) โปรโตคอลดิสตริบิว-ชั่น (Protocol Distribution) เฟรมไซส์ดิสตริบิวชั่น(Framesize Distribution) และทำการวิเคราะห์โปรโต้คอล (Protocol Analyser)

โครงงานนี้ได้พัฒนามาจากโครงงานของปีที่แล้วโดยเพิ่มส่วนต่างๆจากเดิม และ แก้ไขการทำงานบางส่วนที่ไม่สมบูรณ์ สามารถใช้งานได้ตามวัตถุประสงค์ที่ต้องการ และได้แก้ไขบางส่วนของแพ็กเกจไดรเวอร์เพื่อให้สมบูรณ์ในการใช้งาน

ซึ่งมีรายละการพัฒนาโปรเจ็คดังนี้

- ส่วนติดต่อผู้ใช้
- วิเคราะห์โปรโตคอล
- กรองแพ็กเกจ
- การกระจายของโปรโตคอล และ ขนาดของเฟรม
- การสนทนาระหว่างโฮส
- วิเคราะห์สถิติ

LOCAL COMPUTER NETWORK MONITORING AND ANALYSIS

Pongkeit Phetkat
Anantasak Tippayachai
Mr.Thana Hongsuwan Advisor
1996

Abstract

At present the large organization are using the computers in networking system by sharing the resources and increasing the efficiency. In order to do this, we using the interface card called NIC - (Network Interface Card) connected the Networking system and developed the program for investigation the LAN - (Local Area Network) which call Packet Driver. The Packet Driver software is using for investigation the system. Analysis the problem in the system also investigate the protocol working distribution.

This project is developing from the existing project to make it more completion and modification the packet driver software for efficiency and ability to use according to the required purpose.

The main topic for developing in this project are :-

- User Interface
- Protocol Analysis
- Filtering Packet
- Frame Size, Protocol Distribution
- Host Conversation
- Statistical Analysis Connection

สารบัญ

บทที่ 3 การวางแผนออกแบบและการสร้าง

สารบัญรูปภาพ

สารบัญตาราง

บทนำ

ในการใช้งานในสภาพแวดล้อมที่เป็นแลนนั้นแตกต่างจากการใช้งานในสภาพแวดล้อมที่มีผู้ใช้เพียงคนเดียว ในสายส่งข้อมูลนั้นจะมีแพ็กเกจไหลอยู่และมีโปรโตคอลสื่อสารกันเพื่อใช้ส่งข้อมูลและการบริหารทรัพยากรจากระบบหนึ่งไปยังอื่น เช่น เน็ตแวร์แลน (Netware LAN)มีการใช้งานเพิ่มขึ้น ดังนั้นเราต้องการรายละเอียดข้อมูลเกี่ยวกับประสิทธิการใช้งานของโปรโตคอล (Protocol) ปัญหาการใช้งาน การทดสอบ การทำให้มีผลดีที่สุด และ เพื่อใช้ประกอบในการเรียนรู้ถึงกระบวนการต่างๆที่เกิดขึ้นภายในระบบโดยสามารถเข้าใจถึงการทำงานจากการเฝ้าดูแพ็กเกจที่วิ่งในเครือข่าย สามารถนำไปใช้แก้ปัญหาเกิดขึ้นภายในระบบเครือข่ายซึ่งจะต้องตรวจสอบถึงสาเหตุต่างๆที่เกิดขึ้นโดยดูจากข้อมูลต่างๆที่ได้ในเครือข่าย เช่น ประสิทธิภาพการใช้งานเครือข่าย การสูนทนาระหว่างเครื่อง ปริมาณการใช้งานของแต่ละโปรโตคอล และ การกระจายของขนาดเฟรม ซึ่งข้อมูลต่างๆนี้นำมาใช้ประกอบในการแก้ไขปัญหาของระบบได้เป็นอย่างดี ทั้งนี้โปรแกรมและเครื่องมือต่างๆจำต้องสั่งซื้อมาจากต่างประเทศซึ่งมีราคาแพงและอาจไม่ตรงกับความต้องการมากนัก จึงคิดว่าน่าจะสามารถพัฒนาโปรแกรมนี้เองได้

ส่วนประกอบเนื้อหาทั้งหมดของโครงงาน

ในปริญญานิพนธ์นี้ประกอบด้วย 5 บทด้วยกันซึ่งครอบคุม ความรู้และทฤษฎีพื้นฐาน การวางแผนการออกแบบและการสร้าง การทดลองและผลของการทดลอง และ บทวิจารณ์และสรุป

บทที่ 1 บทนำ จะกล่าวถึงวัตถุประสงค์ ความสำคัญ และ ที่มาของโครงงาน รวมถึงขอบเขตของโครงงาน และ ประโยชน์ที่ได้รับจากโครงงานนี้ รวมทั้งส่วนประกอบเนื้อหาของวิทยานิพนธ์นี้

บทที่ 2 ความรู้และทฤษฎีพื้นฐาน กล่าวถึงความรู้ต่างๆที่ใช้ในโครงงาน เช่น โครงสร้างระบบโอเอสไอ (OSI) โครงสร้างของอีเทอร์เนต (Ethernet) โปรโตคอลในเลเยอร์ (Layer) ต่างๆ หลักการของเน็ตเวิร์คมอนิเตอร์ริง การวัดประสิทธิภาพและปัญหาของเครือข่าย การใช้งานของแพ็กเกจไดร์วเวอร์ ซึ่งในส่วนของโปรโตคอลในเลเยอร์จะกล่าวถึงรูปแบบส่วนหัวของแพ็กเกจ และหลักการของโปรโตคอลในเลเยอร์ทีซีพี/ไอพี จะกล่าวถึงรูปแบบโปรโตคอลไอพี ไอซีเอ็มพี ทีซีพี ยูดีพี

บทที่ 3 การวางแผนการออกแบบ และ การพัฒนา มีเนื้อหาถึงหลักการเบื้องต้นในการออกแบบซอฟต์แวร์ การวางแผนและพัฒนาระบบ วิธีการทำงานของโปรแกรม ซึ่งจะกล่าวถึงรายละเอียดของแต่โมดูล (module)ที่ใช้ในการสร้างโปรแกรม รวมถึงโครงสร้างข้อมูลแต่ละโมดูล ใน

แต่ละโมดูลนั้นประกอบด้วยการเก็บสถิติในรูปแบบระยะสั้น และ ระยะยาว การใช้งานของแต่ละโฮส (host) การเตือนภัยเมื่อมีปัญหาในเน็ตเวิร์ค การกรองแพ็กเกจ การเก็บแพ็กเกจ การส่งแพ็กเกจบนเครือข่าย และสุดท้ายการถอดรหัสโปรโตคอล ตลอดถึงบางส่วนในวิทยานิพนธ์ให้ข้อมูลเกี่ยวกับตัวอย่างปัญหา การแก้ปัญหา และการเพิ่มประสิทธิภาพของเน็ตเวิร์ค

บทที่ 4 การทดลอง และ ผลการทดลอง ผลการทำงานของแต่ละส่วน ปัญหาและอุปสรรคที่พบในการทำงาน

บทที่ 5 บทวิจารณ์ และ บทสรุป

ภาคผนวก ก ประกอบด้วยคำศัพท์ที่ใช้ในวิทยานิพนธ์นี้ คำแปล และความหมาย ภาค ภาคผนวก ข รูปแบบส่วนหัวของแต่ละโปรโตคอล ภาคผนวก ค การกำหนดตัวเลขที่ใช้ในอีเทอร์เนต ภาคผนวก ง ตารางการแปลงเลขฐานสิบหกเป็นฐานสิบ

รูปแบบที่ใช้ในวิทยานิพนธ์

ทุกตัวเลขในเอกสารนี้เขียนในรูปแบบเฉพาะโดยเลขฐานสิบเขียนตรงๆคือ 11 และเลขฐานสิบหก เขียนตัวนำในรูป "0x" เช่น 0x0B ทุกๆค่าที่อ้างอิงกับเน็ตเวิร์คฮาร์ดแวร์แอดเดรส (ต้นทาง ปลายทาง) จะเขียนในรูปแบบเลขฐานสิบหกโดยไม่มีตัวนำ เช่น AA-BB-CC-DD-EE-FF

เมื่อมีคำศัพท์เทคนิคใหม่ที่ไม่ปรากฏมาก่อนจะเขียนในรูปแบบคำอ่านพร้อมกับคำอธิบายในวงเล็บที่เป็นภาษาอังกฤษซึ่งจะพบในภาคผนวก ก โดยเก็บรวบรวมคำศัพท์เทคนิคไว้ คำศัพท์ที่ใช้เฉพาะจะพิมพ์ตัวหนา

บทที่ 1

วัตถุประสงค์ และ ขอบเขตโครงงาน

1.1 วัตถุประสงค์

1.1.1 เพื่อพัฒนาซอฟต์แวร์(Software) ที่สามารถตรวจสอบและเฝ้าดู ระบบเครือข่ายเพื่อการควบคุม และดูแลระบบเครือข่ายโดยเจ้าหน้าที่ดูแลระบบ

1.1.2 เพื่อการศึกษาการใช้งานของโปรโตคอลในแต่ละเลเยอร์เพื่อเข้าใจถึงการทำงานของโปรโตคอล

1.1.3 เพื่อจัดทำการเก็บสถิติเพื่อดูระบบเครือข่ายถึงการใช้งานของระบบ

1.1.4 เพื่อใช้ประกอบในการปรับปรุงระบบเครือข่ายให้ทราบถึงจุดบกพร่องในระบบ

1.1.5 เพื่อช่วยในการพัฒนาระบบให้สามารถสื่อสารข้อมูลได้รวดเร็วเพิ่มขึ้น โดยผ่านหาจุดบกพร่องแล้วนำมาพัฒนาระบบ

1.2 ขอบเขตของโครงงาน

1.2.1 เป็นซอฟต์แวร์คอมพิวเตอร์ทำงานบนระบบเครือข่ายท้องถิ่น (LAN)

1.2.2 สามารถเก็บแพ็กเกจเพื่อนำมาวิเคราะห์ ตรวจสอบ เฝ้าดู โดยวิเคราะห์โปรโตคอลหลักๆในแต่ละเลเยอร์ ดังนี้

1.2.3.1 ดาต้าลิงค์เลเยอร์ (Datalink Layer)

- ไออีอีอี 802.2 (IEEE 802.2)
- ไออีอีอี 802.3 (IEEE 802.3)
- อีเทอร์เนตทู (Ethernet II)
- อีเทอร์เนตสแนบ (Ethernet SNAP)

1.2.3.2 เน็ตเวิร์คเลเยอร์ (Network Layer)

- ไอพี (IP)
- ไอพีเอ็กซ์ (IPX)
- เออาร์พี (ARP)

1.2.3.3 ทรานสปอร์ตเลเยอร์เลเยอร์ (Transport Layer)

- ทีซีพี (TCP)
- ยูดีพี (UDP)
- ไอซีเอ็มพี (ICMP)
- เอสพีเอ็กซ์ (SPX)

1.2.3.4 อัพเปอร์เลเยอร์ (Upper Layer)

- เทลเนต (TELNET)

- เอฟทีพี (FTP) เอฟทีพีดาตา (FTP_DATA)
- อาร์ลอกอิน (RLOGIN)
- อื่นๆ

1.2.4 สามารถกรองแพ็กเกจได้โดยกรองในรูปแบบ ลักษณะของแพ็กเกจ และ ค่าของฟิลด์ (Field) ในแพ็กเกจนั้น รวมถึงการกรองโปรโตคอลก่อนที่จะแสดงผลบนหน้าจอ

1.2.5 แสดงสภาพภายในเน็ตเวิร์คโดยกล่าวถึงระดับการใช้งานของเครือข่าย (Utilize) อัตราการเกิดข้อผิดพลาด (Error Rate) จำนวนแพ็กเกจต่อวินาที (Packet per second) กิโลไบตต่อวินาที (Kilobyte per second) โปรโตคอลที่ใช้อยู่ (Protocol inuse) การกระจายของขนาดแพ็กเกจ (Packet size Distribution) การกระจายการใช้งานของโปรโตคอล (Protocol Distribution)

1.2.6 แสดงการสนทนาระหว่างโฮส (Host Conversation) โดยแสดงการใช้งานของแต่ละโฮส โดยเปรียบเทียบทั้งแบบโฮสเดี่ยว และ ระหว่างโฮสด้วยกัน

1.2.7 สามารถตรวจสอบโฮสว่ามีการตอบสนองการใช้งานหรือไม่ และ ตรวจสอบระยะเวลาที่ตอบรับเพื่อตรวจสอบดูความหนาแน่นของการใช้งานเครือข่ายของระบบแต่ละโฮส

1.2.8 สามารถเก็บข้อมูลเพื่อเป็นสถิติของปริมาณการใช้งานของเครือข่ายของโดยรวม ขนาดของแพ็กเกจ โปรโตคอล และสามารถดูข้อมูลย้อนหลังได้

1.2.9 สามารถจำลองการทำงานของเครือข่ายโดยสร้างแพ็กเกจแล้วส่งไปในเครือข่าย

1.2.10 วิเคราะห์โปรโตคอลโดยถอดรหัสแพ็กเกจ และ แสดงโปรโตคอลที่ใช้ในแพ็กเกจในแต่ละเลเยอร์

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1.3.1 อำนวยความสะดวกรวดเร็วในการติดต่อสื่อสารระหว่างกลุ่มผู้ใช้

1.3.2 ช่วยแบ่งเบาหน้าที่ผู้ดูแลระบบ

1.3.3 ประหยัดทรัพยากรบุคคลในการทำงาน

1.3.4 แก้ปัญหาทางด้านการติดต่อระหว่างบุคคล โดยใช้ คอมพิวเตอร์ช่วยในการสื่อสาร

1.3.5 ศึกษาโครงสร้างของแพ็กเกจ และการสนทนาระหว่างโฮส

บทที่ 2

ความรู้และทฤษฎีพื้นฐานที่ต้องใช้

ความรู้ที่ต้องใช้ในการดำเนินโครงงาน ประกอบด้วย

2.1 สถาปัตยกรรมเครือข่าย ระดับชั้นโปรโตคอล และ โอเอสไอโมเดล (Network Architectures , Layer Protocol and OSI Model)

เป็นการยากที่จะให้อุปกรณ์ในระบบเครือข่ายสามารถใช้งานร่วมกันได้ ถ้าปราศจากมาตรฐานในการเข้ากันได้ (Compatible) ดังนั้นในปี 1984 อินเตอร์เนชั่นแนลสแตนดาร์ดออร์กาไนซ์เซชั่น (ISO:International Standards for Organization) ได้กำหนดสถาปัตยกรรมในการเชื่อมต่อเข้าด้วยกัน ในระบบเครือข่ายคอมพิวเตอร์ เรียกว่า โอเพ่นซิสเต็มอินเตอร์คอนเน็คชั่น (OSI:open system interconnection)

2.1.1 สถาปัตยกรรมเครือข่าย

ก่อนที่จะเริ่มจะกล่าวถึงสถาปัตยกรรมเครือข่าย และ โปรโตคอล คำว่าสถาปัตยกรรมเครือข่ายนั้นมักจะถูกใช้เพื่ออธิบายลักษณะของเครือข่ายโดยพิจารณาจากฮาร์ดแวร์และซอฟต์แวร์ ดาต้าลิงค์คอนโทรล (data link control) มาตรฐาน โทโพโลจี (topology) และ โปรโตคอล โปรโตคอลหมายถึง ระเบียบแบบแผนในการที่ส่วนประกอบของเครือข่ายสร้างการสื่อสาร แลกเปลี่ยนข้อมูล และสิ้นสุดการสื่อสาร คำว่าโทโพโลจี ใช้เพื่อบอกถึงรูปแบบเครือข่ายว่ามีการเชื่อมต่ออย่างไร

2.1.2 ระดับชั้นโปรโตคอล

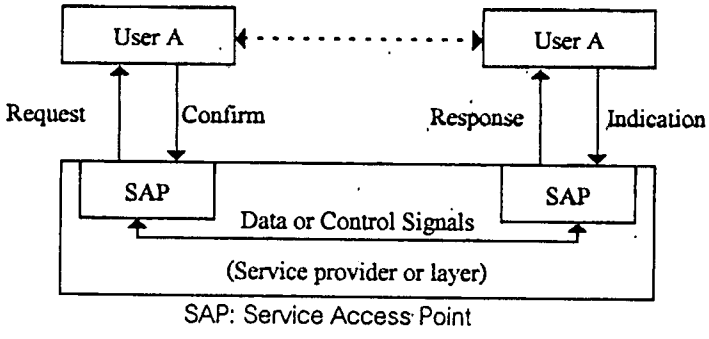
ในการออกแบบโปรโตคอลนั้นจะต้องมี

- แบ่งแยกระบบที่ซับซ้อนเป็นระบบเล็กๆ มีความเข้าในในแต่ละส่วน
- เป็นมาตรฐาน ระหว่างหน้าที่ของเลเยอร์
- แต่ละเลเยอร์ในระดับเดียวกันจะต้องมีหน้าที่เหมือนกัน
- การเปลี่ยนอุปกรณ์ในเลเยอร์หนึ่งจะไม่มีผลในเลเยอร์อื่นๆ

โดยแท้จริงแล้วระดับชั้นโปรโตคอลยึดถือแบบ ที่สามารถให้ระบบต่างกันสามารถสื่อสารกันได้อย่างง่ายและกว้างขวาง โดยปราศจากการเปลี่ยนแปลงในการสื่อสารแต่ละเลเยอร์ หรือเปลี่ยนแปลงให้น้อยที่สุด

การสื่อสารระหว่างเลเยอร์

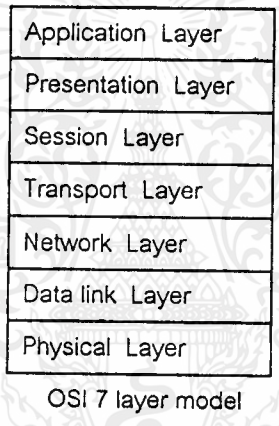
ในระดับชั้นโปรโตคอล และ โอเอสไอโมเดล ระบบหนึ่ง ๆ ประกอบด้วยเลเยอร์ ตามรูป 2.1 แสดงถึงเลเยอร์ชั้นที่หนึ่งจัดการให้บริการระหว่าง User A และ User B การสื่อสารระหว่างผู้ใช้ผ่านการบริการไปถึง แอดเดรส หรือตัวกำหนด มักจะเรียกว่า เซอร์วิสแอคเซสพอยต์ (SAP:Service Access Point) ซึ่งจะจัด การบ่งบอกถึงอุปกรณ์นั้นให้เป็นหนึ่งเดียวไม่ซ้ำ



รูปที่ 2.1 โอเอสไอเลเยอร์ และ การจัดเตรียมบริการ

2.1.3 โอเอสไอโมเดล

เมื่อกำหนดหลักการของโปรโตคอลแล้วจึงได้แบ่งโปรโตคอลเป็นระดับชั้นต่างๆ ได้เป็น 7 ชั้น ดังนี้



รูปที่ 2.2 การแบ่งเลเยอร์ในโอเอสไอโมเดล

1. **ฟิสิคอล เลเยอร์** (Physical layer) ทำหน้าที่กำหนดการสื่อสารทางกายภาพ ระดับสัญญาณทางไฟฟ้า วัสดุตัวนำที่ใช้ในการสื่อสาร การต่อเชื่อมทางกายภาพ ตัวอย่างอุปกรณ์ในเครือข่ายที่ทำงานในชั้นนี้ ได้แก่ **รีพีทเตอร์** (Repeater) หรือ ตัวทวนสัญญาณ

2. **ดาต้าลิงค์ เลเยอร์** (Data link layer) ทำหน้าที่ควบคุมการสื่อสาร แบบจุดต่อจุดที่ติดกัน (Point to Point) ให้สามารถได้รับข้อมูลได้อย่างถูกต้อง ปราศจากข้อมูลที่ผิดพลาด ในโหนดที่ติดกัน มาตรฐานในระดับนี้ ได้แก่ ดีไอเอ็กซ์อีเทอร์เนต ไออีอีอี 802.5 เป็นต้น อุปกรณ์ในเครือข่ายที่ทำงานในชั้นนี้ ได้แก่ บริดจ์

3. **เน็ตเวิร์ค เลเยอร์** (Network layer) ทำหน้าที่ควบคุมการสื่อสารระหว่าง ต้นทาง กับปลายทาง (End to End) ซึ่งในระหว่างทางจะมีเครือข่ายอยู่หรือไม่ก็ได้ รวมทั้งการหาเส้นทางที่เหมาะสมในการเดินทางของข้อมูลจากต้นทางไปยังปลายทาง (ทำโดยเราท์เตอร์) มาตรฐานในระดับนี้ ได้แก่ X.25, ไอพี, เอสพีเอ็กซ์ เป็นต้น อุปกรณ์ในเครือข่ายที่ทำงานในชั้นนี้ เช่น เราท์เตอร์

4.ทรานสปอร์ต เลเยอร์ (Transport layer) ทำหน้าที่ควบคุมการสื่อสารระหว่างต้นทาง และปลายทางให้สามารถได้รับข้อมูลที่ถูกต้องปราศจากข้อผิดพลาด (Error free) มาตรฐานใน ระดับนี้ ได้แก่ TCP, IPX เป็นต้น

5. เซสชั่น เลเยอร์ (Session layer) ทำหน้าที่ควบคุมการจัดการจราจรในการสื่อสาร เช่น การหยุดส่งข้อมูลชั่วคราวเมื่อฝ่ายรับรับข้อมูลไม่ทัน หรือประมวลผลข้อมูลนั้นยังไม่เสร็จ เป็นต้น ซึ่งโดยปรกติจะจัดการโดยโปรแกรมประยุกต์เอง

6. พรีเซนต์เตชั่น เลเยอร์ (Presentation layer) ทำหน้าที่ควบคุมการแสดงผลข้อมูล การแทนค่าข้อมูล การให้ความหมายของข้อมูล เช่น จะให้ข้อมูลชุดนี้แทนรหัส ASCII หรือ EBCDIC, การตีความกลุ่มของข้อมูลว่าเป็น เลขจำนวนเต็ม หรือ ตัวอักษร หรือ จำนวนจริง เป็นต้น ซึ่งโดยปรกติจะจัดการโดยโปรแกรมประยุกต์เอง

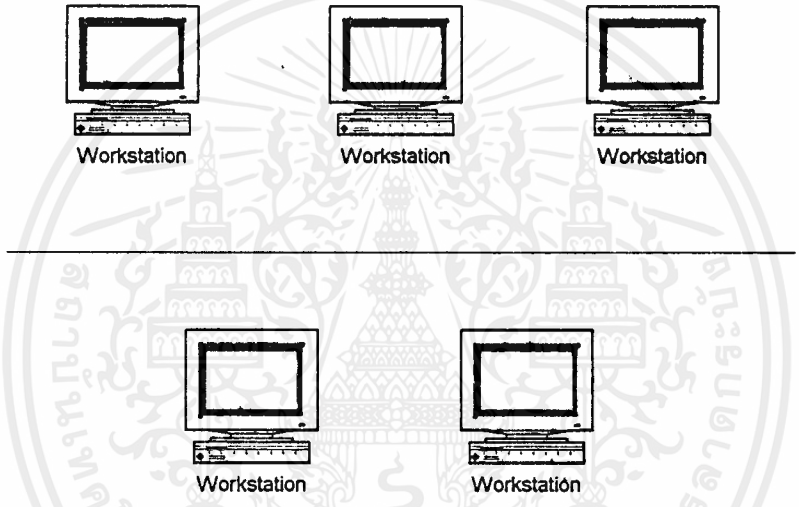
7. แอพพลิเคชั่น เลเยอร์ (Application layer) เป็นระดับโปรแกรมประยุกต์ เช่น การส่งข้อความร้องขอข้อมูล การส่งข้อมูลตอบกลับ เป็นต้น

อีเทอร์เนต

อีเทอร์เนตได้นำมาใช้กับอย่างกว้างขวางในการทำโปรโตคอลระดับชั้นล่าง ซึ่งมีประวัติความเป็นมาและมีผู้ใช้และพัฒนามากมายซึ่งในส่วนนี้จะกล่าวถึงรายละเอียดอีเทอร์เนต

2.2.1 ประวัติการพัฒนา

อีเทอร์เนตเริ่มต้นในปี ค.ศ. 1970 เกิดจากการค้นคว้าและวิจัยของ Palo Alto Research Center ซึ่งเป็นส่วนหนึ่งของบริษัท Xerox ต้องการให้สามารถใช้งานสื่อสารระหว่างคอมพิวเตอร์ได้อย่างกว้างขวาง และรวดเร็วจึงต้องใช้ความเร็วสูงในการส่งข้อมูลซึ่งนี้คือจุดกำเนิดของ อีเทอร์เนต



รูปที่ 2.3 แสดงการเชื่อมต่อเครื่องคอมพิวเตอร์ในระบบเครือข่าย

วิธีการที่ใช้คือฟังสื่อที่ใช้ในการสื่อสารว่าว่างหรือไม่แล้วจึงส่งข้อมูลลงไป ถ้าส่งไม่ได้เนื่องจากมีการใช้งานอยู่ก็จะช่วงเวลาหนึ่งแล้วจึงส่งข้อมูลอีกที แต่อาจจะมีโอกาสที่ข้อมูลชนกันได้เนื่องจากส่งข้อมูลพร้อมกัน จึงต้องมีวิธีการในการป้องกันการส่งพร้อมกันซึ่งเรียกว่า แบ็คออฟอัลกอริทึม (backoff algorithm) ซึ่งจะกล่าวในภายหลัง ในตอนเริ่มแรกมีความเร็วเพียง 2.67 เมกกะบิตต่อวินาที ซึ่งสร้างในปี ค.ศ. 1973 ถึง ค.ศ. 1975 ซึ่งรุ่นที่ออกมานั้นยังเป็นรุ่นทดสอบอยู่ ซึ่งมีข้อกำหนดที่ความเร็วและจำนวนสเตชันที่เชื่อมต่อ (station)

ในปีค.ศ. 1980 เด็ค อินเทลคอร์ปเปอเลชัน และ ซีรอก พร้อมกันที่จะกำหนดมาตรฐานอีเทอร์เนตซึ่งกำหนดเป็นรุ่นที่ 1 โดยมีคุณสมบัติดังนี้

- 10 เมกกะบิตต่อวินาที
- ความยาวสูงสุด 2.8 กิโลเมตร
- จำนวนสเตชันสูงสุด 1024 เครื่อง
- ใช้หลักการส่งแบบเบตแบนด์ (baseband)
- โทโพโลจีแบบบัส (bus)
- ขนาดเฟรมเปลี่ยนแปลงได้

2.2.2 หลักการทำงาน

อีเทอร์เนตใช้หลักการของ CSMA/CD ซึ่งทุกเวิร์คสเตชัน (workstation) ใช้สายสื่อระบบร่วมกันตามรูป เพราะว่าทุกเครื่องสามารถส่งข้อมูลได้ในเวลาเดียวกันบนสาย ซึ่งเป็นสาเหตุให้สัญญาข้อมูลในสายเกิดการชนกันของข้อมูล ดังนั้นจะต้องส่งข้อมูลนั้นใหม่หมด

2.2.2.1 ข้อดีและข้อเสียของอีเทอร์เนต

ข้อดี

- **ง่ายในการติดตั้ง** : คอมพิวเตอร์สามารถเชื่อมต่อกับเซกเมนท์ใดๆได้ง่ายเพียงแต่ใช้ ทีคอนเนคเตอร์ (T-connector) หรือ ตัวรับส่ง (Transceiver) อื่นๆ
- **เป็นวิทยาการที่ใช้อย่างแพร่หลาย** : มีการใช้งานในอีเทอร์เนตมาเป็นระยะเวลานาน มีเครื่องมือใช้ในการเชื่อมต่อมากมายง่ายในการเลือกใช้
- **ราคาอุปกรณ์ที่ใช้ไม่แพงมาก** : อุปกรณ์ที่จำเป็นต่อการเชื่อมต่อเช่น การ์ดอินเทอร์เฟสนั้นมีราคาไม่แพงมากนักทำให้ใช้กันอย่างแพร่หลาย
- **รูปแบบการใช้งานมีอย่างกว้างขวาง** : อีเทอร์เนตสามารถใช้ชนิดของสายได้หลายแบบ เช่น 10 base5, 10base2, 10baseT, Fiber optic

ข้อเสีย

- **เมื่อมีการใช้งานเพื่อขึ้นประสิทธิภาพการใช้งานจะลดลง** : หลักการของ CSMA/CD เมื่อมีการใช้งานของเครือข่ายเพิ่มขึ้น ประสิทธิภาพจะลดลงเพราะเนื่องจากการชนกันของข้อมูลยิ่งมีมากขึ้นถ้ามีการใช้งานเพิ่มขึ้นเพราะโอกาสที่จะส่งข้อมูลพร้อมกันในสายสื่อระบบยิ่งมีมากขึ้น
- **ยากในการตรวจสอบปัญหา** : อีเทอร์เนตยากที่จะตรวจสอบปัญหาที่เกิดขึ้นกับสาย เพราะถ้าสายขาดแล้วจะทำให้ระบบ แลนทั้งเซกเมนท์จะไม่ทำงาน

ก่อนที่จะใช้งานบนเครือข่ายแบบอีเทอร์เนตจะต้องเข้าใจถึงการทำงานของ CSMA/CD ซึ่งจะช่วยในการเข้าใจถึง สถิติ ข้อผิดพลาด และ การใช้งาน ซึ่งจะกล่าวในส่วนต่างๆของปริญญานิพนธ์ฉบับนี้

2.2.2.2 การทำงานของ CSMA/CD ส่วนการส่งข้อมูล

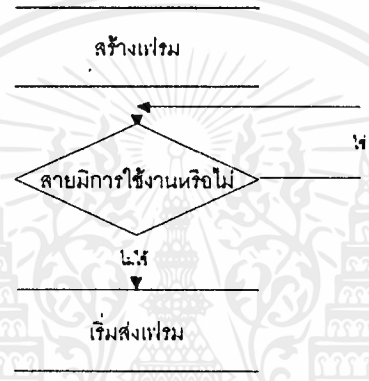
เนื่องจาก CSMA/CD ใช้สายสื่อระบบร่วมกัน ดังนั้นจึงมีข้อกำหนดในการส่งข้อมูล อย่างไรก็ตามก็ยังมีโอกาสที่จะทำให้เกิดการส่งข้อมูลพร้อมกันได้ (CSMA/CD โปรโตคอลคล้ายกับการพูดคุยทางโทรศัพท์ ในขณะที่คนหนึ่งกำลังพูดอยู่ในสาย ถ้าอีกคนหนึ่งพูดขึ้นมาพร้อมกันด้วยจะไม่ทราบถึงข้อมูลที่อีกฝ่ายส่งมาได้ ต้องพูดคุยกันใหม่อีกครั้ง)

2.2.2.2.1 ขั้นตอนการทำงานในการส่งข้อมูลบน CSMA/CD

มีด้วยกันทั้งหมด 4 ขั้นตอนดังนี้

ขั้นที่ 1 ฟังก่อนที่จะส่ง

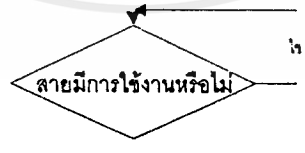
สเตชันจะเฝ้าดูว่าสายที่ส่งนั้นมีสัญญาณแคร์เรีย (carrier) หรือไม่ตามรูปที่ 2.4 สัญญาณนี้วัดโดยค่าระดับความต่างศักย์ซึ่งบ่งบอกถึงการใช้งานของสายนั้น ถ้าสเตชันไม่พบ แคร์เรียออน (carrier on) จะหมายความว่าสายนั้นว่างและพร้อมที่จะส่ง (เหมือนกับการที่เรายกหูโทรศัพท์เพื่อดูว่าสายว่างหรือไม่) ถ้าสายไม่ว่าง (แคร์เรียออน) เมื่อสเตชันกำลังจะส่ง แพ็กเกจที่ส่งนั้นจะชนกัน



รูปที่ 2.4 สเตชันเฝ้าดูการใช้งานสาย

ขั้นที่ 2 รอถ้าสายไม่ว่าง

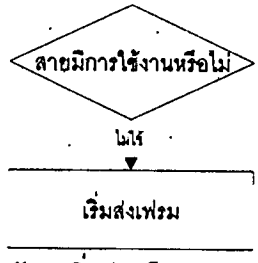
เพื่อป้องกันการชนกัน สเตชันจะรอถ้าสายนั้นถูกใช้งานอยู่ซึ่งแสดงตามรูปที่ 2.5 ซึ่งโดยปกติแล้วการ์ดอินเทอร์เฟสจะยังไม่ส่งถ้าสายไม่ว่าง (เปรียบกับโทรศัพท์แล้วเหมือนกับว่าเมื่ออีกคนกำลังพูดอยู่ต้องรอให้พูดเสร็จก่อนจึงค่อยพูดต่อ) Deferal time คือช่วงเวลาที่สเตชันจะต้องรอก่อนที่จะพยายามส่งใหม่อีกครั้ง



รูปที่ 2.5 สเตชันรอเวลาถ้าสายไม่ว่าง

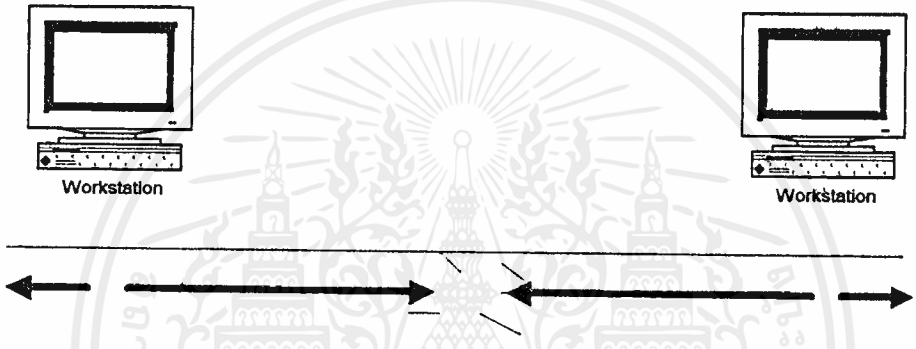
ขั้นที่ 3 ส่ง และ รอว่าแพ็กเกจที่ส่งไปนั้นชนหรือไม่

เมื่อสายว่างอย่างน้อยต้องใช้เวลา 9.6 ไมโครวินาที ถึงจะส่งข้อมูลตามรูป แล้วจึงส่งแพ็กเกจลงไปในสายสื่อระบบ



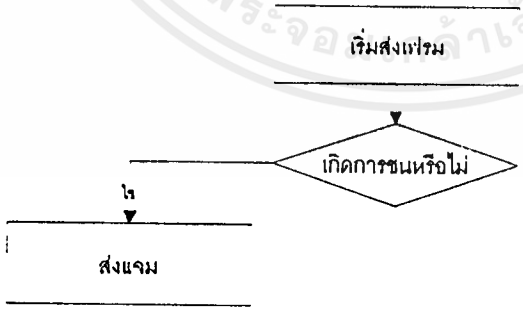
รูปที่ 2.6 ถ้าสายว่างสเตชันจะเริ่มส่งแพ็กเกจ

ถ้าสเตชันอื่นบนเซกเมนท์ส่งแพ็กเกจพร้อมกัน จะชนกันได้ตามรูปที่ 2.7 แสดงถึงการชนกันของแพ็กเกจ (ถ้าคุณพูดพร้อมกันในโทรศัพท์ก็จะคุยกันไม่รู้เรื่อง) หลังจากส่งไปแล้วสเตชันก็จะทดสอบว่าสายมีการชนกันหรือไม่ ซึ่งการชนกันจะตรวจสอบได้โดยสัญญาณที่เกิดบนสายซึ่งจะเท่ากันหรือมากกว่าสัญญาณที่เกิดจากกว่าส่งข้อมูล หรือ มากกว่านั้นพร้อมกัน



รูปที่ 2.7 เมื่อมีการชนกันของแพ็กเกจในสื่อ

ถ้าเกิดการชนขึ้นแต่สเตชันอื่นไม่พบสัญญาณว่าชนกัน อาจจะพยายามส่ง แต่ก็จะเกิดการชนกันอีก เพื่อป้องกันการเกิดเหตุการ์ณแบบนี้ สเตชันจะต้องทำให้ทุกสเตชันจะต้องรู้ถึงว่าสายโดยส่ง แจม (jam) ดังแสดงตามรูปที่ (แจม คือการส่งอย่าง 32 บิตที่ไม่เท่ากับค่า ซีอาร์ซี ของการส่งครั้งก่อนหน้านี้) สเตชันจะเพิ่มค่าพยายามในการส่งอีก 1

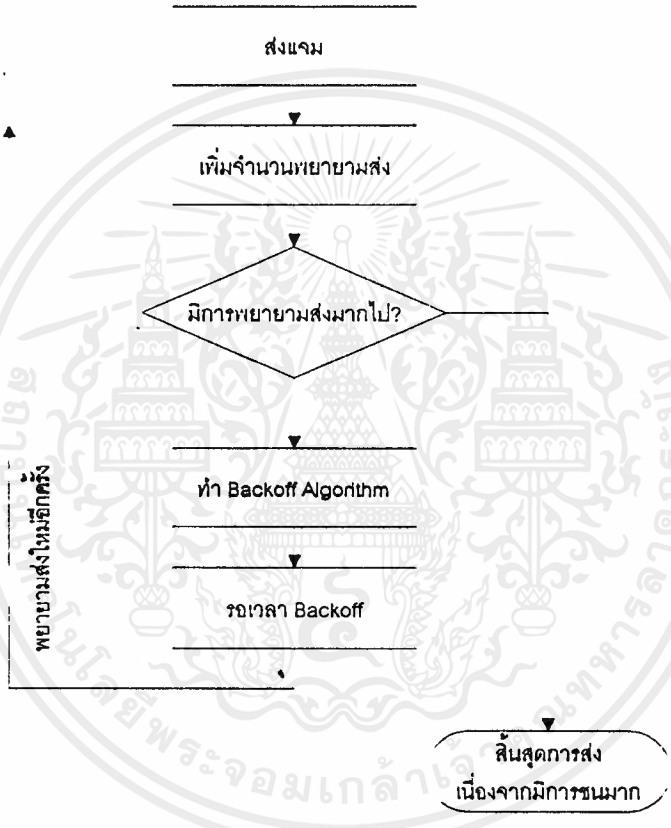


รูปที่ 2.8 ถ้ามีการชนเกิดขึ้นสเตชันจะส่งแจม

ขั้นที่ 4 รอก่อนจะส่งใหม่

ถ้าเราส่งทันทีหลังจากชนกันจะทำให้เกิดการชนครั้งที่สอง จำเป็นอย่างยิ่งที่จะกำหนดเวลาสุ่มเพื่อที่จะรอไม่ให้ชนกันอีก

เพื่ออธิบายถึงการส่งข้อมูลใหม่อีกครั้ง สเตชันจะทำวิธีการที่เรียกว่า แบ็คออฟอัลกอริทึม ซึ่งกำหนดเวลาสุ่มที่จะใช้เพื่อการรอก่อนที่จะส่งแพ็กเกจอีกครั้งหนึ่ง ตามรูปที่ เพื่อลดการชนกันอีกครั้งหนึ่ง (ยกตัวอย่าง การคุยกันพร้อมกันก็จะต้องหยุดทั้งสองฝ่ายแล้วจึงมีฝ่ายหนึ่งค่อยพูดขึ้นมาอีกครั้งหนึ่ง)



รูปที่ 2.9 สเตชันใช้วิธีการแบ็คออฟ เพื่อจะใช้ในการส่งแพ็กเกจอีกครั้ง

ขั้นที่ 5 ส่งอีกครั้งหรือหยุดการส่ง

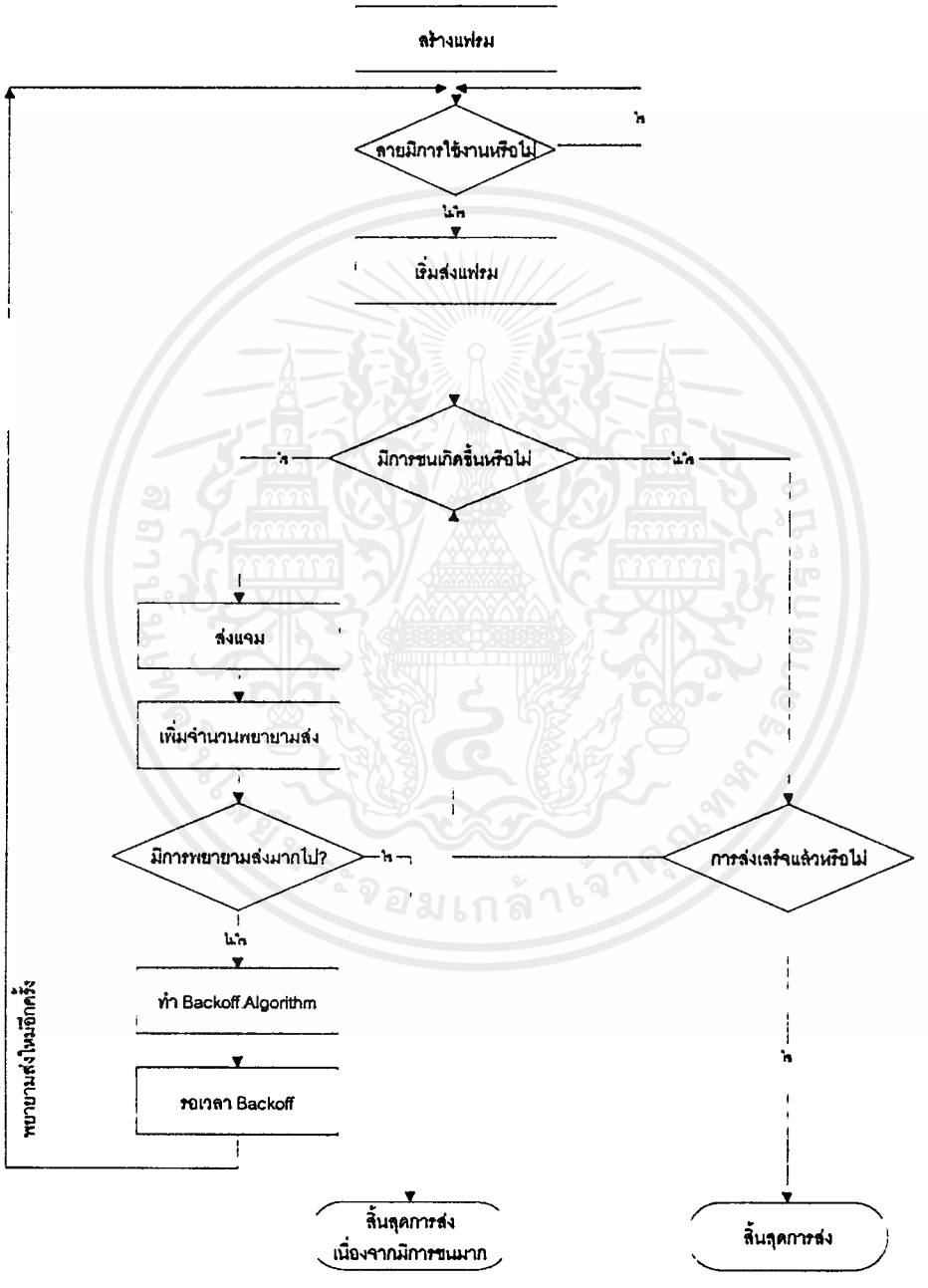
ถ้ามีการส่งแพ็กเกจไปเรื่อยๆแต่ไม่สามารถส่งไปในเครือข่ายได้อาจเป็นไปได้ที่มีปัญหาขึ้นในเครือข่าย จึงต้องกำหนดจำนวนครั้งในการพยายามที่จะส่งแพ็กเกจ โดยทั่วไปจะกำหนดไม่ 16 ครั้ง

2.2.2.2.2 ขั้นตอนการทำงานในการรับข้อมูลบน CSMA/CD

มีด้วยกันทั้งหมด 4 ขั้นตอนดังนี้

ขั้นที่ 1 ตรวจสอบแพ็กเกจที่ได้รับ

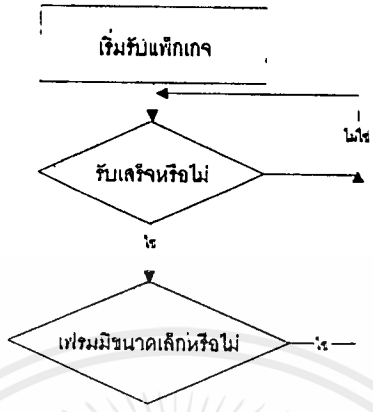
บนอีเทอร์เนตทุกสเตชันจะดูแพ็กเกจที่อยู่บนสาย เพื่อที่จะดูถึงแอดเดรสที่ส่งมายังสเตชันนั้นๆ และต้องตรวจสอบว่ามีขนาดถูกต้องหรือไม่ (อย่างน้อย 64 ไบต์) และ มีครบถ้วนไม่ แฟรกเมนท์เนื่องจากการชน ดูได้จากรูปที่ 2.11



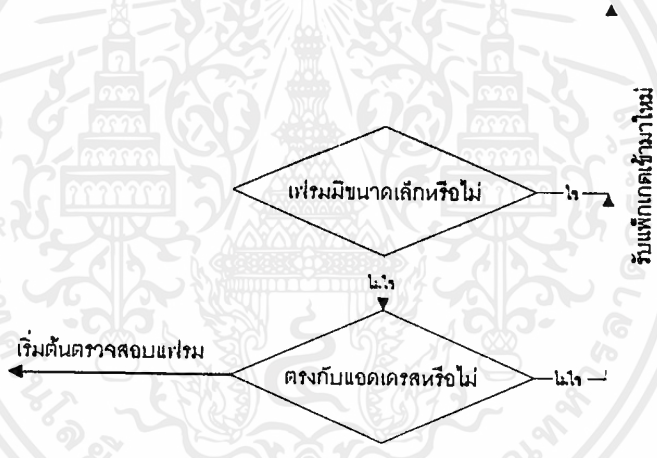
รูปที่ 2.10 ลำดับขั้นการส่งแพ็กเกจ

ขั้นที่ 2 ตรวจสอบว่าแอดเดรสปลายทาง

หลังจากตรวจสอบแพ็กเกจแล้วว่าไม่ แฟรกเมนท์ ก็จะตรวจสอบว่าแอดเดรสปลายทางของแพ็กเกจที่ตรวจสอบนั้นว่าเป็น บรอดคาส หรือ มัลติคาสหรือไม่ ตามรูปที่ 2.12



รูปที่ 2.11 เมื่อตรวจสอบแพ็กเกจสเตชันจะมองหาแฟรกเมนท์



รูปที่ 2.12 สเตชันตรวจสอบที่อยู่ปลายทาง

ขั้นที่ 3 ตรวจสอบถึงความถูกต้องของแพ็กเกจ

ในจุดนี้สเตชันที่รับจะรู้ว่าแพ็กเกจไม่ แฟรกเมนท์ และ เป็นแอดเดรสที่รับได้ แต่จะไม่รู้ว่าแพ็กเกจอาจจะเสียหายเนื่องจากการส่งข้อมูลในสายหรือไม่ เพื่อที่จะป้องกันการรับแพ็กเกจที่เสียหายนี้ สเตชันที่รับจะต้องตรวจสอบถึงคุณสมบัติของแพ็กเกจตามรูปที่ 2.13

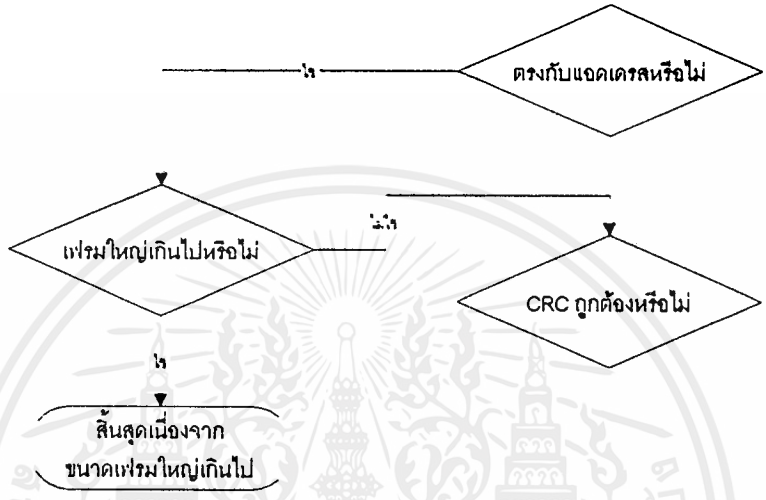
อันดับแรกถ้าจะต้องตรวจสอบขนาดของแพ็กเกจ ถ้าเฟรมขนาดใหญ่เกินกว่า 1518 ไบต์ จะถือว่าเป็น โอเวอร์ไซส์เฟรม (oversized frame) ซึ่งจะตรวจสอบในแพ็กเกจไดร์วเวอร์

บางทีอาจจะมีการสลับบิตได้ จาก 1 อาจเป็น 0 หรือกลับกัน (ซึ่งทำให้แพ็กเกจเสียหาย) เพราะขณะที่ส่งไปบนสายสื่อระบบอาจจะมีการรบกวนของสภาพแวดล้อมซึ่งเป็นสาเหตุให้เกิดเหตุการณ์เช่นนี้ได้ ซึ่งจะต้องตรวจสอบ ซีอาร์ซี (CRC:Cyclical Redundancy Check) ถ้าตรวจสอบซีอาร์ซี แล้วไม่ผ่านจะตรวจสอบเฟรมนั้นต่อว่า อไลน์เมนท์ (alignment) ถูกต้องหรือไม่

แพ็กเกจ มิสอไลน์ (misaligned) คือแพ็กเกจที่ไม่จบลงด้วยจำนวนเท่าของ 8 บิตเช่น แพ็กเกจขนาด 72 ไบต์ 3 บิต อาจเป็นไปได้ที่เป็น 72 ไบต์ หรือ 73 ไบต์

เมื่อตรวจสอบว่าแฟรกเมนท์ ยาวเกินไป มีซีอาร์ซีไม่ถูกต้อง จัดเรียงผิดพลาด (มิสอไลน์) หลังจากนั้นก็ตรวจสอบว่าแพ็กเกจมีขนาดเล็กเกินไปหรือไม่โดยพิจารณาจากขนาดว่าอย่างน้อยต้องมีขนาดมากกว่า 64 ไบต์

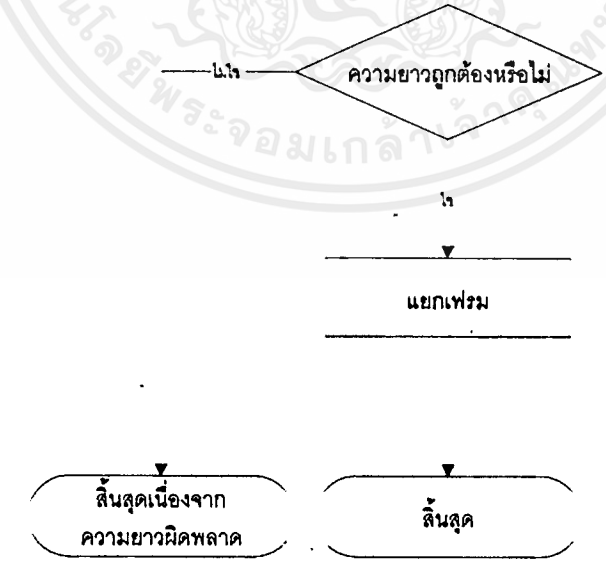
ตามรูปที่ 2.15 อธิบายถึงขั้นตอนการรับแพ็กเกจของสเตชันบนเครือข่ายแบบ CSMA/CD



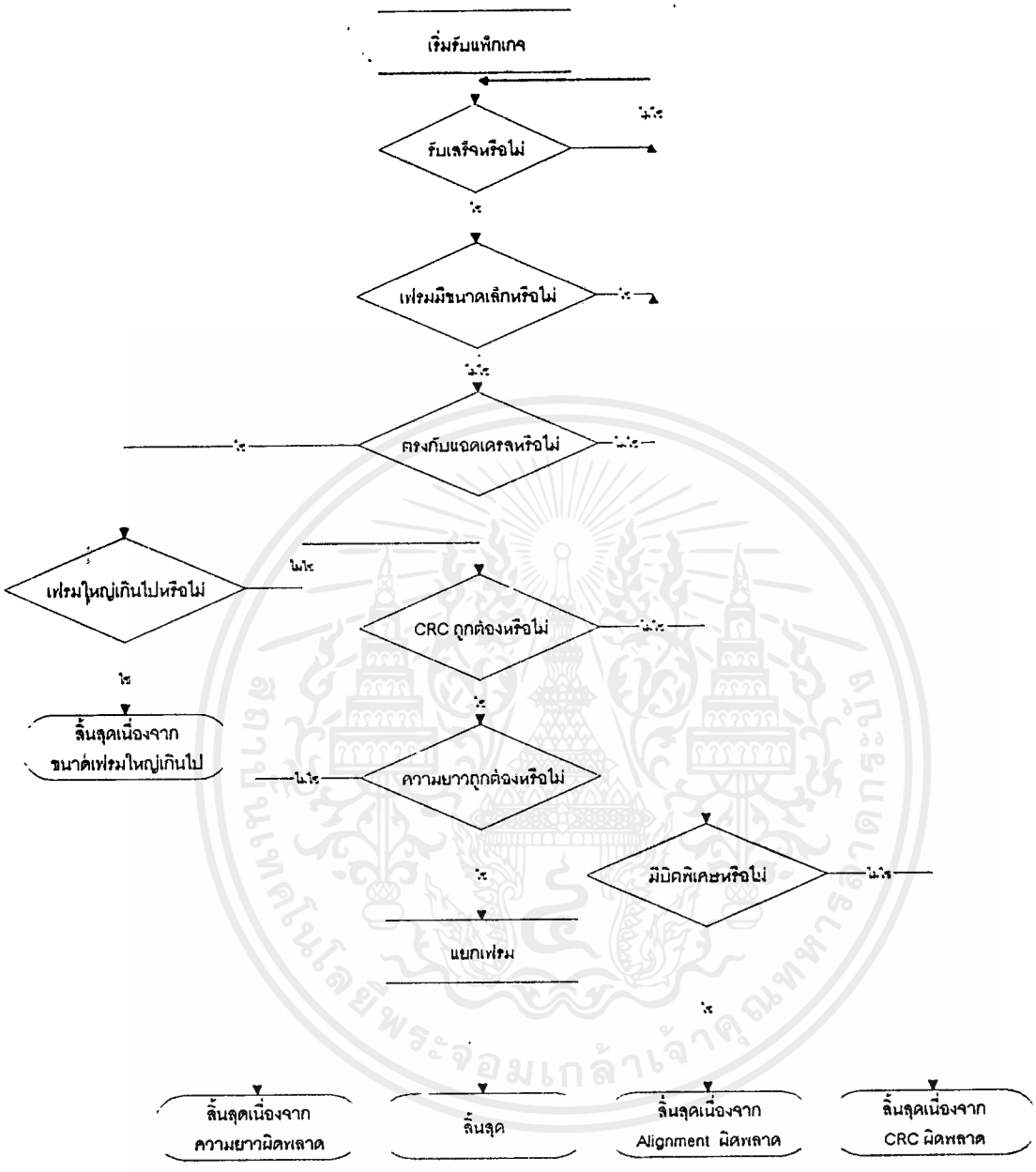
รูปที่ 2.13 แพ็กเกจถูกตรวจสอบสำหรับความถูกต้อง

ขั้นที่ 4 โปรเซสแพ็กเกจ

ถ้าแพ็กเกจผ่านขั้นตอนการตรวจสอบแต่ละขั้นตอนแล้ว ตามรูปที่ จะถือว่าแพ็กเกจนั้นถูกต้อง ในรูปแบบ ขนาด ถ้าสเตชันยังมีปัญหาอีกให้พิจารณาถึงข้างในแพ็กเกจเพื่อที่จะหาปัญหาที่เกิดขึ้นบางทีอาจเกิดขึ้นในโปรโตคอลเลเยอร์อื่นๆก็เป็นไปได้



รูปที่ 2.14 แพ็กเกจที่ถูกต้องจะถูกจัดการต่อ



รูปที่ 2.15 ลำดับขั้นการรับแพ็กเกจ

ส่วนประกอบของอีเทอร์เนตเฟรม

อีเทอร์เนตเฟรมมีขนาดประมาณ 46 และ 1500 ไบต์ เฟรมน้อยกว่า 60 ไบต์ในส่วนข้อมูลจะเรียกว่า รั้น (runt) เฟรม รูปที่ เป็นตัวอย่างของอีเทอร์เนตเฟรม

Preamble	Destination Address	Source Address	Type	Data	FCS

รูปที่ 2.16 โครงสร้างเฟรมอีเทอร์เนต

เฟรมนี้ประกอบด้วย 6 ฟิลด์ด้วยกันทุกฟิลด์มีหน้าที่และขนาดเฉพาะ ซึ่งมีรายละเอียดดังนี้

พลีเอมเบิล (Preamble)

เป็นเลขลำดับ 64 บิตที่ฟิซิคัลเลเยอร์ใช้เพื่อการสร้างสัญญาณพร้อม (synchronization signal) ระหว่างวงจรที่เชื่อมต่อกับสื่อ

ที่อยู่ปลายทาง (Destination Address)

เป็นเลข 48 บิต ฮาร์ดแวร์แอดเดรส ซึ่งเรียกว่า อีเทอร์เนตแอดเดรส หรือ แมคแอดเดรส (ทุกๆ การ์ดอินเทอร์เฟสจะมีเลข 48 บิต ที่บ่งบอกถึงความแตกต่างของการ์ดอินเทอร์เฟสนั้น) ของสเตชันที่ต้องการส่งไป

ที่อยู่ต้นทาง (Source Address)

เป็นเลข 48 บิต ฮาร์ดแวร์แอดเดรส ของผู้ส่ง

ไทป์ (TYPE)

เป็นเลข 2 ไบต์ใช้เพื่อบ่งบอกชนิดโปรโตคอลถ้ามีการใช้งานหลายโปรโตคอลในระดับบนสื่อเดียวกัน จะใช้เลขนี้บ่งบอกถึงโปรโตคอลในระดับบน

ข้อมูล (DATA)

เป็นข้อมูลขนาดระหว่าง 46 ถึง 1500 ไบต์

เอฟซีเอส (FCS:FRAME CHECK SEQUENCE)

เป็นเลขขนาด 32 บิตที่คำนวณจาก ซีอาร์ซีทุกๆฟิลด์ยกเว้นฟิลด์ตัวเอง

ในสภาพแวดล้อมทั่วไปในเครือข่ายจะมีโครงสร้างเฟรมของอีเทอร์เนตที่ใช้พื้นฐานโครงสร้างของโครงสร้างเฟรมอีเทอร์เนต เพียงแต่แตกต่างกันตรงส่วนการใช้งานของแต่แบบไม่เหมือนกันซึ่งมีด้วยกันดังนี้

2.2.3.1 อีเทอร์เนต 802.3

อีเทอร์เนต 802.3 เฟรมนั้นคล้ายๆกับอีเทอร์เน็ต II แต่ไม่เหมือนตรงฟิลด์ต่างๆดูได้ตามรูปที่ 2.17 ซึ่งมีหน้าที่และขนาดเฉพาะซึ่งมีรายละเอียดดังนี้

Preamble	Destination Address	Source Address	Length	Data	FCS

รูปที่ 2.17 โครงสร้างเฟรมอีเทอร์เนต 802.3

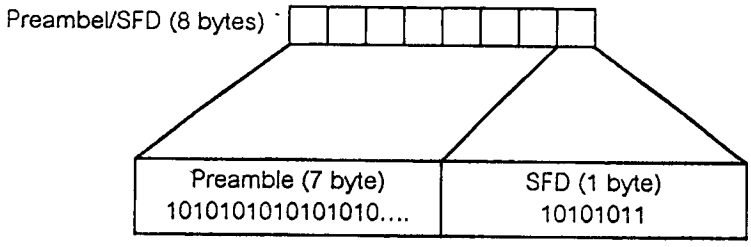
โครงสร้างเฟรม

พรีเอมเบิ้ล

เป็นเลขลำดับ 56 บิตหรือ 7 ไบต์ที่ฟิซิคัลเลเยอร์ใช้เพื่อการสร้างสัญญาณพร้อม (synchronization signal) ระหว่างวงจรที่เชื่อมต่อกับสื่อ

เอสเอฟดี (SFD: Start frame delimiter)

เป็นเลขไบนารี่ (binary) 10101011 ที่ชี้ถึงจุดเริ่มต้นของเฟรมดังตัวอย่างตามรูปที่ 2.18



รูปที่ 2.18 โครงสร้างฟิลด์เอสเอฟดี

หมายเลขปลายทาง

เป็นเลข 48 บิต ฮาร์ดแวร์แอดเดรส ซึ่งเรียกว่า อีเทอร์เนตแอดเดรส หรือ แมคแอดเดรส (ทุกๆ การ์ดอินเทอร์เฟสจะมีเลข 48 บิต ที่บ่งบอกถึงความแตกต่างของการ์ดอินเทอร์เฟสนั้น) ของสเตชันที่ต้องการส่งไป ซึ่งแอดเดรส FFFFFFFFFFFF หมายถึงบรอดคาส

หมายเลขต้นทาง

เป็นเลข 48 บิต ฮาร์ดแวร์แอดเดรส ของผู้ส่งซึ่งจะต้องไม่เป็น บรอดคาส จะต้องเป็นแอดเดรสของ เวิร์คสเตชัน เซิฟเวอร์ หรือ เราท์เตอร์

ความยาว (LENGTH)

เป็นเลข 2 ไบต์ใช้เพื่อบอกขนาดแพ็กเกจซึ่งค่านี้จะต้องน้อยกว่า 1500

ข้อมูล

เป็นข้อมูลขนาดระหว่าง 46 ถึง 1500 ไบต์

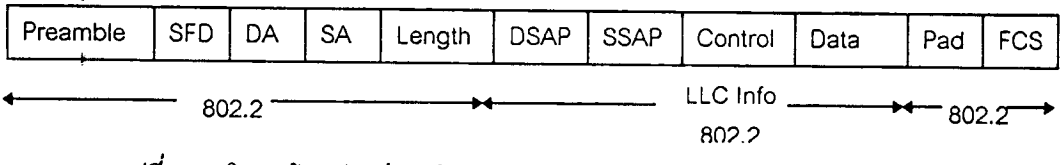
แพดดิ้ง (PADDING)

ฟิลด์นี้เปลี่ยนแปลงได้ ใช้เพื่อตรวจสอบแพ็กเกจให้มีขนาดที่ตรงตามข้อกำหนด เช่นในอีเทอร์เนตต้องมีขนาดอย่างน้อย 64 ไบต์ ซึ่งถ้ามีการส่งข้อมูลไม่ถึงจะต้องเพิ่มแพดดิ้งเข้าไปเพื่อให้ครบ 64 ไบต์

หมายเหตุ ถ้าเฟรมถูกต้องและค่าความยาวมากกว่ากว่า 1500 จะหมายความว่าเป็นอีเทอร์เนตทูเฟรม และ เป็นไทป์ฟิลด์

2.2.3.2 อีเทอร์เนต 802.2

เฟรมอีเทอร์เนต 802.2 กำหนด IEEE-compliant เนื่องจากมีข้อมูลทั้ง 802.3 ฟิลด์ และ 802.2 ฟิลด์ ซึ่ง 802.2 ฟิลด์กล่าวถึง LLC (Logical Link Control) เลเยอร์ภายในเฟรม ดูได้จากรูปที่



รูปที่ 2.19 โครงสร้างเฟรมอีเทอร์เนต 802.2

โครงสร้างเฟรม
พรีเอมเบิ้ล : 8 ไบต์
ที่อยู่ปลายทาง : 6 ไบต์
ที่อยู่ต้นทาง : 6 ไบต์
ความยาว : 2 ไบต์
ข้อมูลและแพดดิ้ง : 46-1500 ไบต์
เอฟซีเอส : 4 ไบต์

ดีเอสเอพี (DSAP : Destination Service Access Point)

เป็นเซอร์วิสแอคเซสพอย์นปลายทางของโฮสปลายทางซึ่งใช้ในเลเยอร์บน หรือ เน็ตเวิร์คเลเยอร์ :

เอสเอสเอพี (SSAP : Source Service Access Point)

เป็นเซอร์วิสแอคเซสพอย์นต้นทางของโฮสต้นทางซึ่งใช้ในเลเยอร์บน หรือ เน็ตเวิร์คเลเยอร์

ควบคุม (CONTROL)

กำหนดถึงการส่งแบบคอนเน็คชันเลสเซอร์วิส (Connectionless Service)

2.2.3.3 อีเทอร์เนต สแนบ

สแนบมาจากแฟรม อีเทอร์เนต 802.2 ดังแสดงตามรูป

Preamble	SFD	DA	SA	Length	DSAP	SSAP	Control	Organization Code	Data	Pad	FCS

รูปที่ 2.20 โครงสร้างเฟรมอีเทอร์เนต สแนบ

โครงสร้างเฟรม
พรีเอมเบิ้ล : 8 ไบต์
ที่อยู่ปลายทาง : 6 ไบต์
ที่อยู่ต้นทาง : 6 ไบต์
ความยาว : 2 ไบต์
ข้อมูลและแพดดิ้ง : 46-1500 ไบต์
ดีเอสเอพี เอสเอสเอพี และ คอนโทรลฟิลด์
เอฟซีเอส : 4 ไบต์

ดีเอสเอพี เอสเอสเอพี และ คอนโทรล

ในอีเทอร์เนตแสนบค่าใน ดีเอสเอพี และ เอสเอสเอพี จะต้องเป็น 0xAA ซึ่งค่า 0xAA เป็นตัวบ่งบอกถึงเฟรมที่เป็นแสนบ

ออร์กาไนซ์เซชันโคด (Orgranization Code)

ฟิลด์นี้กำหนดให้อีเทอร์เนตไทป์ฟิลด์ซึ่งค่านี้เป็น 0x00-00-00 ในออร์กาไนซ์เซชันฟิลด์

อีเทอร์เนตไทป์

ฟิลด์นี้กำหนดถึงโปรโตคอลระดับบน (Upper layer protocol) ซึ่งค่ามีดังนี้

ไอพี	0x0800
เออาร์พี	0x0806
อาร์เออาร์พี	0x8035
แอบเปิ้ลทอร์ค	0x809B
แอบเปิ้ลทอร์ค เออาร์พี	0x80F3
เน็ตแวร์ ไอพีเอ็กซ์/เอสพีเอ็กซ์	0x8137

2.2.3.4 อีเทอร์เนตทู

อีเทอร์เนตทูเฟรมแตกต่างจาก 2 แบบที่กล่าวมาเนื่องจากไทป์ฟิลด์ซึ่งตามหลังที่อยู่ปลายทาง แต่อีเทอร์เนต 802.3 อีเทอร์เนต 802.2 และ อีเทอร์เนตแสนบ จะเป็นฟิลด์ความยาวแทน ซึ่งแสดงตามรูป

Preamble	Destination Address	Source Address	Type	Data	FCS

รูปที่ 2.21 โครงสร้างเฟรมอีเทอร์เนตทู

โครงสร้างเฟรม

พรีเอมเบิ้ล

เป็นเลขลำดับ 64 บิตหรือ 8 ไบต์ที่ฟิซิคัลเลเยอร์ใช้เพื่อการสร้างสัญญาณพร้อม (synchronization signal) ระหว่างวงจรที่เชื่อมต่อกับสื่อ กำหนดด้วยค่าสลับกันระหว่าง 1 และ 0 ซึ่งมีด้วยกัน 7 ไบต์ ส่วนไบต์สุดท้ายเป็นเอสเอฟดี

ไทป์

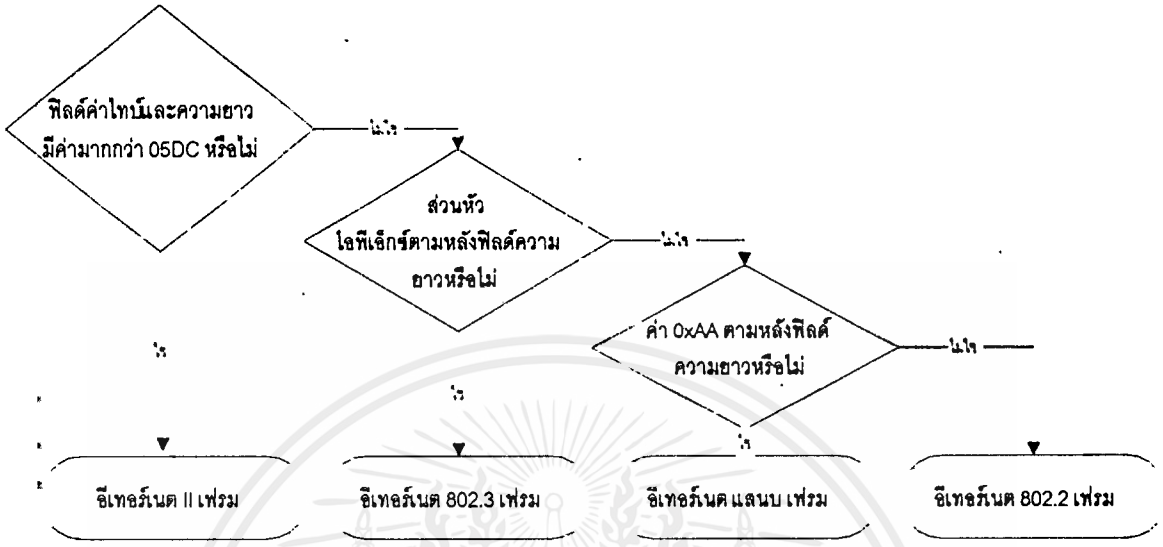
เป็นส่วนที่บอกถึงชนิดโปรโตคอลในระดับบนซึ่งค่าโปรโตคอลที่ใช้มีดังนี้

ไอพี	0x0800
เออาร์พี	0x0806
อาร์เออาร์พี	0x8035
แอบเปิ้ลทอร์ค	0x809B
แอบเปิ้ลทอร์ค เออาร์พี	0x80F3
เน็ตแวร์ ไอพีเอ็กซ์/เอสพีเอ็กซ์	0x8137

สังเกตุว่าค่าเหล่านี้จะคล้ายกับ อีเทอร์เนตแสนบไทป์ฟิลด์

2.2.4 วิธีการแยกชนิดเฟรม

จากรูปที่ 2.22 แสดงถึงวิธีการแยกประเภทเฟรมของอีเทอร์เนต



รูปที่ 2.22 แสดงลำดับขั้นตอนการแยกชนิดอีเทอร์เนตเฟรม

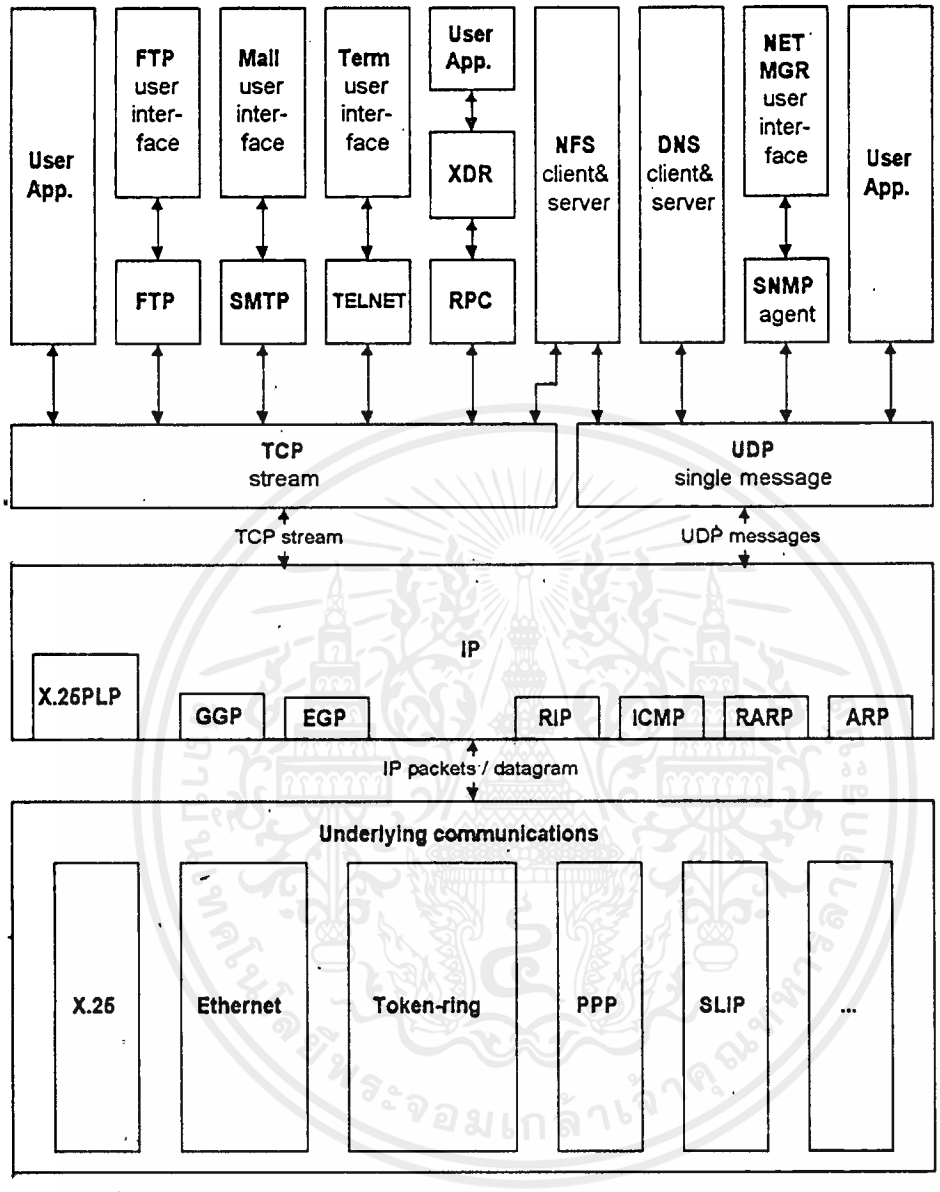
2.3 ทีซีพี/ไอพีโปรโตคอลซูท (TCP/IP Protocol suite)

กลุ่มของโปรโตคอล ซึ่งจะให้บริการ ฟังก์ชั่น พื้นฐานสำหรับการใช้งาน ซึ่งรวม ไอพี ทีซีพี ยูดีพี สามารถนำมาใช้ เป็นการส่ง ไฟล์ (file) เมล์ (mail) ค้นหาว่าใครใช้งานอยู่บ้าง ในเบื้องต้น ทีซีพี/ไอพี ถูกใช้อย่างมากใน มินิคอมพิวเตอร์ (minicomputers) หรือ เมนเฟรม (mainframes) ดังนั้นความสำคัญของ ทีซีพี/ไอพี ในการให้บริการต่างๆได้แก่

- **ไฟล์ทรานสเฟอร์** (files transfer) ไฟล์ทรานสเฟอร์โปรโตคอล (FTP:Files transfer protocol) อนุญาตให้ผู้ใช้เครื่องคอมพิวเตอร์ใดก็ตาม สามารถรับ-ส่งไฟล์จากเครื่อง คอมพิวเตอร์อื่นได้
- **รีโมทล็อกอิน** (remote-login) เทลเน็ตโปรโตคอล (TELNET:Network terminal protocol) อนุญาตให้ผู้ใช้เข้าไปใช้เครื่องคอมพิวเตอร์ที่ใดก็ได้ในเน็ตเวิร์ค สามารถที่จะเริ่ม รีโมทเซสชั่น (remote session) โดยกำหนดคอมพิวเตอร์ที่จะเชื่อมต่อได้
- **อิเลคโทรนิคส์เมล์** (electronic mail) อนุญาตให้ผู้ใช้ส่งข้อความไปยังผู้ใช้คนอื่นใน เน็ตเวิร์คได้

ซึ่งการให้บริการเหล่านี้เป็นการนำไปใช้งานของ ทีซีพี/ไอพี ซึ่งแสดงให้เห็นว่า ทีซีพี/ไอพี มีบทบาทเป็นพื้นฐานของเน็ตเวิร์ค ในปัจจุบัน การใช้งานของคอมพิวเตอร์ก็มีการเปลี่ยนแปลงไป แต่เดิมใช้เป็นลักษณะประมวลผลกลาง หรือ ใช้พวกเมนเฟรม ก็เปลี่ยนมาใช้การประมวลผลแบบกระจาย หรือ ใช้พวก ไคลน์เซิฟเวอร์ (Client Server) แทน ซึ่งการจะทำงานเหล่านี้ก็ต้องใช้พื้นฐานของ ทีซีพี/ไอพี

- **เน็ตเวิร์คไฟล์ซิสเตม** (NFS:network file system) เป็นการอนุญาตให้ระบบเข้าถึงข้อมูล จากคอมพิวเตอร์เครื่องอื่นได้โดยง่ายกว่าการใช้ เอฟทีพี ซึ่งจะมีประสิทธิภาพกว่ามาก
- **การพิมพ์ปลายทาง** (remote printing) อนุญาตให้มีการใช้งานเครื่องพิมพ์ผ่านระบบเน็ตเวิร์ค ได้
- **ทางทำงานปลายทาง** (remote execution) ทำให้มีการทำงานข้ามระบบได้เพื่อประหยัดทรัพยากร โดยสามารถที่จะสั่งงานเครื่องใหญ่ๆ ได้จากเครื่องเล็กๆ
- **เนมเซิฟเวอร์** (name server) ในระบบใหญ่ๆ ที่มีเครื่องหลายเครื่องการใช้ชื่อเป็นสื่อแทนตัวเลขจะเหมาะแก่การบริหารคอมพิวเตอร์มากกว่า
- **เทอร์มินอลเซิฟเวอร์** (terminal servers) ในการทำงานเราสามารถที่จะทำงานจาก เทอร์มินอล เพื่อใช้ทรัพยากรร่วมกันได้



รูปที่ 2.23 การใช้งานของโปรโตคอลต่างๆ

ทีซีพี/ไอพี สร้างด้วยวิทยาการของคอนเน็คชั่นเลส (connectonless) ข้อมูลถูกส่งไปตามลำดับเป็นดาต้าแกรม (Datagram) คือกลุ่มข้อมูลที่ส่งไปหนึ่งแมสเซส และถูกส่งไปยังหลายระบบ เน็ตเวิร์ค หมายความว่าข้อมูลได้ถูกแบ่งเป็นข้อมูลย่อยๆ หลายส่วนเพื่อทยอยส่งไป เนื่องจากว่า บางระบบ เน็ตเวิร์ค ไม่สามารถที่จะส่งข้อมูลขนาดใหญ่ได้ต้องแบ่งข้อมูลออกเป็นส่วนย่อยๆ เพื่อส่งไปได้ และเมื่อส่งไปถึงปลายทางก็จะรวบรวมข้อมูลนั้นเป็นตัวเดิม

2.3.2 เลเยอร์ของทีซีพี/ไอพี

2.3.2.1 อีเทอร์เนตเลเยอร์

ในระบบ เน็ตเวิร์ค ส่วนใหญ่ใช้ อีเทอร์เนต ดังนั้นในส่วนนี้จะอธิบายถึงส่วนหัวของอีเทอร์เนต ในการทำงานของอีเทอร์เนตจะต้องมี แอดเดรส (address) เป็นตัวกำหนดการติดต่อสื่อสาร ซึ่งหมายความว่าจะต้องมีหมายเลขที่ไม่ซ้ำกัน และ จะต้องไม่ให้ผู้ใช้ยุ่งยากในการตั้งค่าเหล่านี้ ดังนั้นหมายเลขเหล่านี้ถูกกำหนดมาจากโรงงานที่ทำ การ์ดอินเทอร์เฟส ซึ่งการกำหนดใช้เลขขนาด 48 บิต เมื่อข้อมูลถูกส่งออกไป จะเป็นการส่งแบบ บรอดคาสท์มีเดียม (broadcast medium) คือเมื่อ โฮสรับแพ็กเกจได้ก็จะตรวจสอบว่าเป็นแพ็กเกจ ที่ส่งมาถึงตนเองหรือไม่ ถ้าใช่ก็รับข้อมูลนั้นซึ่งจะดูที่อีเทอร์เนตส่วนหัวซึ่งทุกๆ อีเทอร์เนตแพ็กเกจ จะมี ส่วนหัวขนาด 14 อ็อกเตต ซึ่งจะอธิบายถึง ต้นทาง (source) และ ปลายทาง (destination) และ ชนิด (type)

Ethernet Destination address (first 32 bits)	
Ethernet dest (last 16 bits)	Ethernet source (first 16 bits)
Ethernet source address (last 32 bits)	
Type code	
Destination Address	
IP header, then TCP header, then your data	
...	
end of your data	
Ethernet Checksum	

รูปที่ 2.24 โครงสร้างของอีเทอร์เนตโปรโตคอล

ถ้าเราใช้ E แทนอีเทอร์เนตส่วนหัว ใช้ C แทน ผลรวมตรวจสอบอีเทอร์เนต (ethernet checksum) ใช้ I แทน ส่วนหัวไอพี T แทน ส่วนหัวทีซีพี เราจะเห็นข้อมูลในรูปแบบดังนี้

EIT...C EIT...C EIT...C EIT...C EIT...C

เมื่อแพ็กเกจได้รับแล้วจะเอาส่วนหัวและผลรวมตรวจสอบออก เนื่องจากไม่ใช้อีกแล้ว และพิจารณาที่ ไทป์โคด (type code) ถ้าเป็นโคด(code) ของ ไอพีก็จะส่ง ดาต้าแกรม ไปยัง ไอพี ในส่วนไอพีจะเอาไอพีส่วนหัวออก และดูตรงส่วน ไอพีโปรโตคอลฟิลด์ และถ้าเป็น ทีซีพี ก็จะส่งไปยังส่วนของ ทีซีพี ในส่วนนี้จะดูที่ หมายเลขลำดับ เพื่อที่จะรวมข้อมูลเป็นข้อมูลเดิม

2.3.2.2 ระดับชั้นไอพี

ทีซีพีส่งดาต้าแกรมมายังชั้นไอพี ซึ่งจะต้องบอกถึงอินเทอร์เนตแอดเดรส ของเครื่องปลายทาง สังเกตว่าในชั้นไอพีนั้นไม่คำนึงถึงข้อมูลที่ส่งมาจากชั้นบน (ดาต้าแกรม) แม้กระทั่ง ส่วนหัวทีซีพี งานหลักของไอพีคือ ค้นหาเส้นทางเพื่อที่จะส่งไปยังปลายทาง ในการที่จะให้ เกตเวย์ (gateway) ส่งดาต้าแกรมจะต้องมีส่วนของส่วนหัวในดาต้าแกรม ซึ่งในส่วนนี้จะบอกถึง อินเทอร์เนตแอดเดรส ต้นทาง และ ปลายทาง(32 บิตแอดเดรส เช่น 161.246.6.71) และ โปรโตคอลนัมเบอร์ (protocol number) และ ผลรวมตรวจสอบ ตำแหน่งต้นทางและปลายทางใช้เพื่อบอกให้ทั้ง 2 ฝ่ายทราบว่าข้อมูลมาจากที่ใด และจะไปที่ใด ส่วนโปรโตคอลนัมเบอร์ใช้บอก ไอพี ว่าจะส่งดาต้าแกรม ไปยังชั้นทีซีพี หรือโปรเซสไอพี โดยการทำงานส่วนใหญ่ของไอพีจะเป็นทีซีพีแต่ก็ยังมีโปรโตคอล อื่นๆที่ใช้ ไอพีซึ่งในส่วนของ โปรโตคอลนัมเบอร์ จะเป็นการบอกว่าใช้ โปรโตคอล อะไร ส่วนสุดท้ายคือผลรวมตรวจสอบส่วนหัว ใช้เพื่อตรวจสอบว่าข้อมูลของส่วนหัวไม่เสียหายระหว่างการส่ง หรือไม่ก็ถูกส่งไปยังผิดที่หมายซึ่งมีรูปร่างดังนี้

Version	IHL	Type of Service	Total Length	
Identification			Flags	Fragment Offset
Time to Live		Protocol	Header Checksum	
Source Address				
Destination Address				
TCP header, then your data				

...

รูปที่ 2.25 โครงสร้างของอินเตอร์เนตโปรโตคอล

ซึ่งข้อมูลที่ส่งมาจะลักษณะดังนี้

IT... IT... IT... IT... IT... IT.... IT....

ส่วนแฟลก (flag) และ แฟรกเมนต์ออฟเซ็ต (fragment offset) ถูกใช้เพื่อแบ่งดาต้าแกรมเป็นขนาดเล็กๆ เนื่องจากข้อมูลที่ส่งไปขนาดใหญ่มากและเพื่อให้ข้อมูลถูกส่งไปยังการเชื่อมต่อแต่ละแบบได้ เวลาคงอยู่คือตัวเลขที่ลดลงเรื่อยๆ เมื่อส่งไปยังแต่ละระบบและเมื่อค่านี้เป็น 0 ดาต้า แกรมนี้ก็จะถูกทิ้งไป เพื่อป้องกันการส่งข้อมูลวนในเน็ตเวิร์ค

แต่ถ้าในการเชื่อมต่อธรรมดาระหว่างแค่เครื่อง 2 เครื่อง ไม่จำเป็นต้องใช้ส่วนหัวที่มีขนาดใหญ่มากเท่านี้ก็ได้เพื่อลดโอเวอร์เฮด

2.3.2.3 ระดับชั้นทีซีพี

ทีซีพี จะแยกข้อมูลเป็นดาต้าแกรมย่อยๆ และถูกรวบรวมกลับคืนเมื่อถึงปลายทาง หรือจะถูกส่งอีกครั้งเมื่อข้อมูลสูญหาย ทีซีพีมีการเรียงข้อมูลตามลำดับ และจะส่งข้อมูลต่างๆ ไปตามเส้นทางของเน็ตเวิร์ค ตัวอย่างเช่น เมื่อใช้ซีเรียลไลน์โมเด็ม (serial line MODEM) จากบ้านมายังโฮสที่ภาควิชา ซึ่ง

การเชื่อมต่อตรงจุดนี้มีแบนด์วิธประมาณ 14.4 กิโลบิตต่อวินาที และเมื่อข้อมูลนี้จะถูกส่งไปยังเราท์เตอร์ (router) ของ ลาดกระบังจุดนี้การส่งเป็น 10 เมกกะบิตต่อวินาที และ เมื่อข้อมูลถูกส่งไปยังเกตเวย์ เพื่อไปยัง เน็คเทค (NECTEC) ตรงจุดนี้จะใช้ประมาณ 2 เมกกะบิตต่อวินาที

ในการทำ มัลติเปิลคอนเนคชั่น (multiple connection) คือการที่สามารถใช้งานได้หลายอย่างในการเชื่อมต่อ ซึ่งทีซีพีจะเป็นตัวกำหนดการทำงานส่วนนี้ งานส่วนนี้เรียกว่า ดีมัลติเพลกซิ่ง (demultiplexing) ในความเป็นจริงแล้วมีหลายชั้นที่ทำการ ดีมัลติเพลกซิ่ง บน ทีซีพี/ไอพี ซึ่งในส่วนหัวจะมีอ็อกเตตที่บอกถึงโปรเซสที่ทำ

ทีซีพี จะมีส่วนหัวที่เพิ่มเข้าไปในส่วนข้อมูล มีอย่างน้อย 20 อ็อกเตต ซึ่งจะมีส่วนที่บอกถึงหมายเลขพอร์ต (port number) และ หมายเลขลำดับ (sequence) หมายเลขพอร์ต นำไปใช้เพื่อบอกถึงโปรเซสที่จะใช้อ้างอิง สมมติว่า มี 3 คนกำลังส่งไฟล์ผ่านทีซีพีอาจกำหนดเป็น 1000,1001 และ 1002 สำหรับการส่งครั้งนี้ เมื่อคนแรกส่งดาต้าแกรม ซึ่งจะมาพร้อมกับหมายเลขพอร์ตต้นทาง ไปยังผู้รับอีกฝ่ายก็จะกำหนดหมายเลขพอร์ต สำหรับการติดต่อนี้ด้วยเช่นกัน และ คอมพิวเตอร์ เครื่องแรก ก็จะทราบด้วยว่าจะส่งไปที่พอร์ตใดในปลายทาง และ ทุกครั้งที่ส่งข้อมูลกัน จะมี หมายเลขลำดับ ที่ใช้เพื่อระบุถึงลำดับของข้อมูลที่จะส่ง

<table>
<tr><td colspan="5">Source Port</td><td colspan="4">Destination Port</td></tr>
<tr><td colspan="9">Sequence Number</td></tr>
<tr><td colspan="9">Acknowledgement Number</td></tr>
<tr><td>Data
Offset</td><td>Reserved</td><td>U
R
G</td><td>A
C
K</td><td>P
S
H</td><td>R
S
T</td><td>S
Y
N</td><td>F
I
N</td><td>Window</td></tr>
<tr><td colspan="6">Checksum</td><td colspan="3">Urgent Pointer</td></tr>
<tr><td colspan="9">your data ... next 500 octets</td></tr>
</table>

...

รูปที่ 2.26 โครงสร้างของทรานสมิดชั่นคอนโทรลโปรโตคอล

ซึ่งลักษณะการส่งข้อมูลเป็นดังนี้

T... T... T... T... T... T...

ยังมีหลายส่วนที่ยังไม่ได้อธิบายคือ ส่วนของการจัดการการเชื่อมต่อ ในการที่จะให้การส่ง ดาต้าแกรมไปยังเป้าหมายได้ ผู้รับจะต้องส่งตอบรับ (acknowledgement) ตอบกลับมา ซึ่งจะมีฟิลด์ ของหมายเลขตอบรับ (Acknowledgement number) ยกตัวอย่างเช่น ส่งแพ็กเกจ โดยมี ตอบรับ ของ 1500 หมายถึง รับข้อมูลอย่างถูกต้องจนถึงอ็อกเตตหมายเลข 1500 ถ้าผู้ส่งยังไม่ได้รับตอบรับ ในเวลาที่เหมาะสม ก็จะส่งข้อมูลเดิมอีกครั้งหนึ่ง วินโดว์ (Window) ถูกนำไปใช้ในการควบคุมจำนวนข้อมูลที่

จะส่งไปในแต่ละครั้ง คือจะไม่รอตอบรับทุกครั้งที่จะส่งดาต้าแกรมถัดไป ซึ่งกำหนดไว้ในส่วนของวินโดว์ เรียกวิธีการนี้ว่า สไลด์ดิ้งวินโดว์ (sliding window)

2.4 ข้อกำหนดและการโปรแกรมแพ็กเกจไดรเวอร์ (Packet Driver Specification & Programing)

2.4.1 ข้อตกลงในเอกสารนี้

ทุกตัวเลขในเอกสารนี้เขียนในรูปแบบภาษาซี โดยเลขฐานสิบเขียนในรูปทั่วไปคือ 11 เลขฐานสิบหกเขียนในรูป 0x0B และเลขฐานแปดเขียนในรูป 013 ซึ่งใช้กับทุกๆค่า ที่อ้างอิงกับเน็ตเวิร์คฮาร์ดแวร์แอดเดรส (ต้นทาง, ปลายทาง และ มัลติคาสท์) และ ข้อมูลดีมัลติเพลกซิ่ง สำหรับแพ็กเกจส่วนหัวสมมุติว่าถูกนำไปแทนในส่วนหัวแพ็กเกจระดับเอ็มเอซี (MAC-level packet header) ก่อนที่จะผ่านไป ฟังก์ชั่น *send_pkt()*

2.4.2 ข้อแนะนำ และ ข้อสำคัญ

ในบทนี้จะกล่าวถึงวิธีการเขียนโปรแกรมเชื่อมต่อกับ แพ็กเกจไดรเวอร์ ซึ่ง แพ็กเกจไดรเวอร์ นั้นต้องง่ายและเป็นพื้นฐานในการเขียนโปรแกรมเชื่อมต่อ ซึ่งสามารถให้หลายโปรแกรมสามารถใช้การ์ดอินเทอร์เฟส ที่ดาต้าลิงค์ลีเวล (data link level) ร่วมกันได้ แพ็กเกจไดรเวอร์จะทำการแยกแพ็กเกจที่เข้ามาไปยังโปรแกรมโดยใช้ชนิดแพ็กเกจมาตรฐานของตัวกลาง (Network media's standard packet type) หรือ เซอร์วิสแอคเซสพอยน์ฟิลด์(Service access point field)

ในรายละเอียดนี้สามารถให้มีการทำโปรโตคอลสแต็ก (Protocol Stack) ซึ่งเป็นอิสระกับยี่ห้อ หรือรุ่นของการ์ดอินเทอร์เฟส ซึ่งแตกต่างไปขึ้นอยู่กับสื่อเน็ตเวิร์ค (network media) เช่น อีเทอร์เนต, ริง 802.5, ซีเรียลไลน์ เนื่องจากความแตกต่าง ในการใช้โปรโตคอลแปลงไปยังฟิซิคอลแอดเดรส (physical address) รูปแบบส่วนหัว และ เอ็มทียู (MTU:Maximum Transmission Units)

แพ็กเกจไดรเวอร์ จะจัดเตรียมส่วนเรียกใช้ที่ กำหนดเริ่มต้นใช้งาน (Initiate access) สิ้นสุดการใช้งาน (End access) ส่งแพ็กเกจ และ ให้ข้อมูลสถิติบนการ์ดอินเทอร์เฟส และให้ข้อมูลเกี่ยวกับอินเทอร์เฟส

ในการใช้โปรโตคอลในแต่ละโปรแกรมร่วมกันนั้น ต้องใช้แพ็กเกจไดรเวอร์ร่วมกัน ผู้ใช้สามารถใช้ ทีซีพี/ไอพี เอกซ์เอ็นเอส (XNS) และ โปรโตคอลที่ใช้เฉพาะ เช่น เดคเน็ต (DECNET) ไบยัน (Banyan's) ไลฟ์เน็ต (LifeNet's) และ โนเวล (Novell's) โดยไม่ต้องไปเชื่อมยุ่งยากกับการ์ดอินเทอร์เฟส เพียงแต่เรียกไปยังแพ็กเกจไดรเวอร์เท่านั้น

โปรแกรมซึ่งใช้แพ็กเกจไดรเวอร์ สามารถนำไปใช้ในเน็ตเวิร์คอื่นๆ ซึ่งอยู่ในคลาสเดียวกันได้โดยไม่ต้องแก้ไขโปรแกรม เพียงแค่หาแพ็กเกจไดรเวอร์ อันใหม่เท่านั้น

ในแพ็กเกจไดรเวอร์ สามารถแบ่งระดับชั้นได้ดังนี้

- ฟังก์ชั่นพื้นฐาน (Basic Function) มีฟังก์ชั่นการทำงานพื้นฐานซึ่งง่ายในการใช้งานเนื่องจากใช้ทรัพยากรน้อย
- ฟังก์ชั่นเพิ่มเติม (Extended Function) มีฟังก์ชั่นมากกว่าแบบพื้นฐานสนับสนุนมัลติคาสท์ และเก็บสถิติในอินเทอร์เฟส
- ฟังก์ชั่นประสิทธิภาพสูง (High-performance Function) สนับสนุน การปรับปรุงประสิทธิภาพ

2.4.3 การระบุการ์ดอินเทอร์เฟส

การ์ดอินเทอร์เฟส กำหนดไว้โดยตัวเลขสามส่วนด้วยกัน คือ

2.4.3.1 **คลาส** (Class) ใช้บอกชนิดของสื่อที่อินเทอร์เฟสนี้สนับสนุน เช่น ดีไอเอ็กซ์ (DIX:DEC /Intel /Xerox) อีเทอร์เนต, ไออีอีอี 802.3, ไออีอีอี 802.5 (IEEE 802.5), โปรเน็ต 10 (ProNET-10), แอ็ปเปิ้ลทอล์ค (Appletalk), ซีเรียลไลน์, อื่นๆ

2.4.3.2 **ชนิด** (Type) ใช้กำหนดรายละเอียดของอินเทอร์เฟส ซึ่งสนับสนุนในคลาสนั้น ๆ เช่น ในอีเทอร์เนตคลาสนั้นประกอบด้วย ชนิด 3COM,3C503,3C505, Interlan NI5210 Univation, BICC Data Networks ISOLAN,Ungermann-Bass NIC ฯลฯ ในคลาส ไออีอีอี 802.5 ประกอบด้วย ชนิด IBM Token Ring adapter, Proteon p1340, อื่นๆ

2.4.3.3 **หมายเลข** (Number) ถ้าในเครื่องประกอบด้วยอินเทอร์เฟส มากกว่าหนึ่งคลาสหรือหนึ่งชนิด จะต้องใช้หมายเลขในการระบุถึงความแตกต่าง

คลาสเป็นตัวเลข 8 บิต และชนิดเป็นตัวเลข 16 บิต ซึ่งกำหนดโดยเอฟทีพีซอฟแวร์ (FTP Software) และ ชนิด 0xFFFF แทนที่ทุกชนิดซึ่งจะตรงกับทุกอินเทอร์เฟสในคลาส และในหมายเลขจะไม่มีไวล์คาร์ด (Wildcard) ซึ่ง 0 จะหมายถึงอินเทอร์เฟสอันแรกของคลาสและชนิด

ในข้อกำหนดไม่ได้สนับสนุนมัลติเปิลการ์ดอินเทอร์เฟส (multiple interfaces card) ในแพ็กเกจไดรเวอร์เดียวกัน ซึ่งหมายความว่าต้องเรียกแพ็กเกจไดรเวอร์หนึ่งตัว ต่อหนึ่งอินเทอร์เฟส ซึ่งต้องเรียกหลาย แพ็กเกจไดรเวอร์ และกำหนดอินเทอร์รัป (Interrupt) ให้แต่ละแพ็กเกจไดรเวอร์ ในกรณีที่ต้องใช้มากกว่าหนึ่งอินเทอร์เฟส ซึ่งโปรแกรมจะต้องตรวจสอบคลาส และ ชนิด ซึ่งได้จากการเรียก *driver_info()* เพื่อให้มั่นใจว่าใช้แพ็กเกจไดรเวอร์ตรงกับสื่อและรูปแบบแพ็กเกจ

2.4.4 การเริ่มทำงานของแพ็กเกจไดรเวอร์

แพ็กเกจไดรเวอร์ ถูกอ้างอิงในโปรแกรมอินเทอร์รัปในช่วง 0x60 ถึง 0x80 จะอธิบายถึงวิธีการหา อินเทอร์รัปที่ไดรเวอร์ใช้งานอยู่ ในการกำหนดอินเทอร์รัป จะต้องสามารถเปลี่ยนแปลงได้เพื่อไม่ไปรบกวนกับการทำงานโปรแกรมอื่นๆ

แฮนด์เดิลเลอร์ (Handler) สำหรับอินเทอร์รัป กำหนดโดยเริ่มต้นด้วย 3 ไบต์ของโค้ดทำงาน (executable code) อาจจะเป็นคำสั่งกระโดด 3 ไบต์ หรือ 2 ไบต์ แล้วตามด้วยคำสั่ง เอ็นโอพี (NOP:No Operation) แล้ว ตามด้วยตัวอักษรที่ไม่มีตัวจบโดยมีข้อความว่า "PKT DRVR" เพื่อจะหาอินเทอร์รัปที่ถูกใช้โดยแพ็กเกจไดร เวอร์โปรแกรมจะต้องค้นหาตั้งแต่ 0x60 ถึง 0x80 จนกระทั่งพบ ข้อความ "PKT DRVR" ใน 12 ไบต์ซึ่งตามหลังจุดเริ่มต้น วิธีการหาแฮดเดอร์ ดูได้จากฟังก์ชั่น *access_type()*

2.4.5 โปรแกรมมิ่งอินเทอร์เฟส (Programing Interface)

ทุกฟังก์ชั่นใช้งานโดยซอฟแวร์อินเทอร์รัป ซึ่งทุกครั้งที่เรียกใช้งานจะผ่านค่าไปให้ AH รีจิสเตอร์ (register) ซึ่งจะเป็นตัวกำหนดฟังก์ชั่นในการใช้งาน

แฮนด์เดิล (Handle) คือ ค่าตัวเลขซึ่งเกี่ยวเนื่องกับแต่ละชนิดของแม็คลีเวลดีมัลติเพล็กซิ่ง (MAC-level demulitplexing) ซึ่งได้มาจากการเรียกประเภทการใช้งาน (access type) ภายในแพ็กเกจไดรเวอร์ จะเป็นพอย์นเตอร์ (Pointer) หรือ เทเบิ้ลออฟเซต (Table offset) ก็ได้

บางฟังก์ชั่นเรียกใช้ได้กรณีที่เป็น **ฟังก์ชั่นเพิ่มเติม** หรือ **ฟังก์ชั่นประสิทธิภาพสูง** เนื่องจากไม่จำเป็นต่อการใช้งานพื้นฐานของเน็ตเวิร์คดังนั้นจึงกำหนดเป็นส่วนเพิ่มเติม ซึ่งจะตรวจสอบว่าสามารถใช้งานได้หรือไม่ต้องตรวจสอบที่ *driver_info()*

เงื่อนไข เริ่มต้น

- สามารถใช้แฮนด์เดิลร่วมกันได้เพียง 16 แฮนด์เดิล
- การเรียงตัวของบิตข้อมูลที่ส่งในเน็ตเวิร์คตรงกันข้ามกับข้อมูลในการเก็บของคอมพิวเตอร์ทั่วไป ยกเว้น 802.5 โทเค็นริง นั่นหมายความว่า ค่าประเภทของอีเทอร์ (Etherrtype values) ซึ่งส่งไปยัง *access_type()* จะต้องกลับกัน

2.4.5.1 Driver_info()

function	driver_info(handle)
Input	handle "Optional"
error return	carry flag set
	error code
possible errors	BAD_HANDLE
non-error return	carry flag clear
	version
	class
	type
	number
	name
	functionality
	1 == basic functions present.
	2 == basic and extended present.
	5 == basic and high-performance.
	6 == basic, high-performance, extended.
	255 == not installed.

ในการเรียกฟังก์ชั่นนี้ใช้เพื่อหาข้อมูลเกี่ยวกับ เวอร์ชั่นอินเทอร์เฟส หมายถึง ตัวระบุฮาร์ดแวร์ไดร เวอร์ (Hardware driver) ในสมัยก่อนนั้น แฮนด์เดิล ที่ส่งเข้าไปนั้นต้องมีเสมอ แต่ในปัจจุบันเป็นตัวเลือกซึ่งจะมีหรือไม่มีก็ได้ ดังนั้นผู้ใช้ต้องระวัง

2.4.5.2 *access_type()*

function int access_type(if_class, if_type, if_number, type, typelen, receiver)

input	if_class
	if_type
	if_number
	type
	typelen
	receiver()
error return	carry flag set
	error code
possible errors	NO_CLASS
	NO_TYPE
	NO_NUMBER
	BAD_TYPE
	NO_SPACE
	TYPE_INUSE
non-error	carry flag clear
	handle
receiver call	(*receiver)(handle, flag, len [, buffer])
input	handle
	flag
	len
if AX == 1	char far *buffer; DS:SI

เป็นตัวเริ่มต้นในการทำงานของ แพ็กเกจไดรเวอร์ ซึ่งมีค่าต่างๆดังนี้

- *type* ค่าที่ส่งผ่านไปฟังก์ชั่นนั้นต้องกำหนดโดย ข้อกำหนดชนิดแพ็กเกจ
- *typelen* คือขนาดเป็นไบต์ของฟิลด์ type ถ้าเป็น 0 หมายความว่า ตัวเรียกต้องการทุกแพ็กเกจ
- *receiver* คือ พอย์นเตอร์ ที่ชี้ไปยังส่วนของโปรแกรมซึ่งถูกเรียกทุกครั้งเมื่อแพ็กเกจได้รับ

เมื่อแพ็กเกจได้รับก็จะเรียก *receiver* สองครั้งด้วยกันครั้งแรกเรียกเพื่อขอบัฟเฟอร์ (buffer) จากโปรแกรมเพื่อคัดลอกแพ็กเกจลงไป ซึ่งค่า AX จะมีค่าเท่ากับ 0 ซึ่งโปรแกรมจะต้องส่งค่าพอย์นเตอร์ ไปยังบัฟเฟอร์ ใน ES:DI และ ถ้าโปรแกรมไม่มีบัฟเฟอร์จะส่งค่า 0:0 ใน ES:DI และไดรเวอร์จะทิ้งแพ็กเกจนั้นเสีย และจะไม่มีการเรียกครั้งที่สอง

ความยาวของแพ็กเกจเป็นส่วนสำคัญมากโดยเก็บในค่า CX ค่านี้จะใช้ได้เมื่อค่า AX มีค่าเท่ากับ 0 ซึ่ง *receiver* สามารถกำหนดบัฟเฟอร์ให้เพียงพอกับขนาดได้ ค่าความยาวนี้รวมถึง ส่วนหัวแม็ค (MAC header) และ ข้อมูลที่รับได้ แต่ไม่รวมส่วน เอฟซีเอส (FCS:Frame Check Sequence)

ในการเรียกครั้งที่สองนั้น ค่า AX มีค่าเท่ากับ 1 ซึ่งหมายถึง การคัดลอกแพ็กเกจนั้นเสร็จสิ้นแล้ว และโปรแกรมสามารถนำบัฟเฟอร์ไปใช้งานได้ ซึ่งบัฟเฟอร์ที่ใช้งานได้จะส่งมาใน DS:SI

2.4.5.3 release_type()

function release_type(handle)

input	handle
error return	carry flag set
	error code
possible errors	BAD_HANDLE
non-error	carry flag clear

ฟังก์ชั่นนี้ใช้เพื่อสิ้นสุดการใช้งานโดยเกี่ยวเนื่องกับแฮนด์เดิลที่ได้มาจาก *access_type()*

2.4.5.4 send_pkt()

function send_pkt(buffer, length)

input	buffer
	length
error return	carry flag set
	error code
possible errors	CANT_SEND
non-error	carry flag clear

เป็นการเรียกเพื่อส่งข้อมูลขนาด *length* ไบต์ใน *buffer* ซึ่งโปรแกรมจะต้องจัดเตรียมทั้งแพ็กเกจรวมทั้งส่วนหัวโลคอลเน็ตเวิร์ค (local network headers) ซึ่งจะไม่ถูกใส่ในไดรเวอร์

2.4.5.5 terminate()

function terminate(handle)

input	handle
error return	carry flag set
	error code
possible errors	BAD_HANDLE
	CANT_TERMINATE
non-error	carry flag clear

เป็นการสิ้นสุดไดรเวอร์ ซึ่งเกี่ยวเนื่องกับแฮนเดิ้ลบางทีไดรเวอร์ จะคืนค่าหน่วยความจำให้กับระบบปฏิบัติการเลย ซึ่งจะแตกต่างกับ *release_type()*

2.4.5.6 get_address()

function get_address(handle, buf, len)

input	handle
	buf
	len
error return	carry flag set
	error code
possible errors	BAD_HANDLE
	NO_SPACE
non-error	carry flag clear
	length

ใช้เพื่อหาค่าฮาร์ดแวร์แอดเดรสของอินเทอร์เฟสการ์ด ใส่ลงไปน *buf* ซึ่งมีความยาว *len* ไบต์ ซึ่งค่าจริงๆ ของตัวเลขจะส่งคืนใน CX ถ้ามี NO_SPACE error หมายความว่า *len* นั้นมีขนาดไม่เพียงพอกับความยาวของฮาร์ดแวร์แอดเดรส ถ้าแอดเดรสถูกเปลี่ยนโดย *set_address()* แอดเดรสใหม่จะถูกส่งคืนแทน

2.4.5.7 reset_interface()

function reset_interface(handle)

input	handle
error return	carry flag set
	error code
possible errors	BAD_HANDLE
	CANT_RESET
non-error	carry flag clear

เริ่ม อินเตอร์เฟส ใหม่อีกครั้ง โดยเกี่ยวเนื่องกับแฮนด์เดิลเพื่อ จะรู้สถานะ (state) จะทำการหยุดการส่งข้อมูล และ กำหนดโหมดการรับ (reciever mode) ใหม่อีกครั้ง ค่าฮาร์ดแวร์แอดเดรสจะเปลี่ยนเป็นค่าเดิม จากหน่วยความจำที่เก็บไว้ ค่ามัลติคาสท์ลิสต์ (multicast list) จะถูกลบ และ โหมดการรับจะเป็น 3 (รับเฉพาะค่าแอดเดรสตัวเองและบอร์ดคาสท์) ถ้าแฮนด์เดิลเปิดอยู่หลายอัน การกระทำนี้จะไปรบกวนการทำงานของโปรแกรมอื่นที่ใช้อินเทอร์เฟสนี้ร่วมกันได้ ดังนั้น CANT_RESET จะถูกคืนมาแทน

*2.4.5.8 get_parameters() *high-performance driver function*

function get_parameters()

error return carry flag set
error code

possible errors BAD_COMMAND

non error carry flag clear
struct param

```
struct param {
        unsigned char   major_rev;      /* Revision of แพ็กเกจไดรเวอร์ spec */
        unsigned char   minor_rev;      /*  this driver conforms to. */
        unsigned char   length;         /* Length of structure in bytes */
        unsigned char   addr_len;       /* Length of a MAC-layer address */
        unsigned short  mtu;            /* MTU, including MAC headers */
        unsigned short  multicast_aval; /* Buffer size for multicast addr */
        unsigned short  rcv_bufs;       /* (# of back-to-back MTU rcvs) - 1 */
        unsigned short  xmt_bufs;       /* (# of successive xmits) - 1 */
        unsigned short  int_num; /* Interrupt # to hook for post-EOI
                                        processing, 0 == none */
};
```

ความสามารถของโปรแกรมจะได้ประโยชน์จากฟังก์ชั่นนี้ เพื่อที่จะได้ตัวเลขของไดรเวอร์พารามิเตอร์ (Driver parameter) ซึ่งฟังก์ชั่นนี้ถูกเพิ่มเติมใน เวอร์ชั่น 1.09 และ อาจไม่ถูกนำไปใช้ในไดร เวอร์อื่นๆ

ค่า *major_rev* และ *minor_rev* fields คือค่าหมายเลขหลัก และ หมายเลขรอง ของรุ่นของข้อกำหนด สำหรับข้อกำหนดนี้ ค่า major_rev เป็น 1 และ minor_rev เป็น 9

ค่า *length* field ใช้เพื่ออธิบายขนาดของ *param* นี้

ค่า *addr_len* field เป็นความยาวของ แม็คแอดเดรสในขนาดไบต

ค่า *mtu* คือ ขนาดใหญ่ที่สุดของ แม็คลีเวลแพ็กเกจที่ไดรเวอร์สามารถควบคุมได้ ใน อีเทอร์เนต ค่านี้จะถูกกำหนดตายตัว แต่ใน 802.5 และ เอฟดีดีไอ (FDDI) ค่านี้อาจจะเปลี่ยนแปลงได้

ค่า *multicast_aval* field คือค่าตัวเลขจำนวนไบต์ซึ่งใช้ในการเก็บ แอดเดรสมัลติคาสท์ ซึ่งเป็น ฮาร์ดแวร์เมคานิซึม (hardware mechanism) ค่า 0 หมายถึงไม่สนับสนุนมัลติคาสท์

ค่า *rcv_bufs* และ *xmt_bufs* ชี้ถึงตัวเลขของแบ็คทูแบ็ครีซีฟ (back-to-back receives) หรือ การส่ง (Transmit) ซึ่งโปรแกรมจะใช้เป็นตัวกำหนด ควบคุมการไหล (flow control) หรือ กลยุทธ์การส่ง (transmit strategies) ค่า 0 ใน *rcv_bufs* ใช้โดย ผู้สร้างไดรเวอร์เพื่อจะระบุว่า ฮาร์ดแวร์นั้นจำกัด เพื่อป้องกันจากการรับ

ข้อมูลที่ระบบอื่นอาจส่งมาเร็วกว่าได้ ซึ่งแนะนำว่า โปรโตคอลระดับสูงควรจะทำการ กำหนดควบคุมการไหลทีละขั้น (lock-step flow control) เพื่อมิให้แพ็กเกจสูญเสีย

ค่า *int_num* ฟิลด์ กำหนดโดย ฮาร์ดแวร์อินเทอร์รัป ซึ่งโปรแกรมสามารถเกาะเพื่อที่จะทำอินเทอร์รัปไทม์โปรโตคอล หลังจาก EOI ซึ่งถูกส่งไปยัง 8259 อินเทอร์รัปคอนโทรลเลอร์ และค่า 0 หมายถึงไม่มี อินเทอร์รัป ถ้าโปรแกรมใดเกาะเข้ากับอินเทอร์รัปนี้ และค่าไม่เป็นศูนย์ในเวคเตอร์ จะต้องส่งอินเทอร์รัปลงไปยังลูกโซ่ (Chain) และรอพรีดีเซสเซอร์ (predecessor) เพื่อคืนค่าก่อนจะทำงานหรือ สแต็คสวิทช์ (stack switches)

*2.4.5.9 as_send_pkt() *high-performance driver function*

function int as_send_pkt(buffer, length, upcall)

input	buffer
	length
	upcall()
error return	carry flag set
	error code
possible errors	CANT_SEND
	BAD_COMMAND
non-error	carry flag clear
	buffer available upcall:
	(*upcall)(buffer, result)
	result
	buffer

แตกต่างจาก *send_pkt()* ตรงที่ว่า upcall() routine ถูกเรียกเมื่อข้อมูลโปรแกรมถูกคัดลอกไปจากบัฟเฟอร์แล้ว และโปรแกรมสามารถแก้ไขและใช้บัฟเฟอร์ได้โดยปลอดภัยคือไม่มีใครมาเขียนอีกไดรเวอร์อาจส่งรหัสผิดพลาดที่ไม่เป็นศูนย์ (non-zero error code) ไปยัง upcall() ถ้าการคัดลอกนั้นผิดพลาด หรือข้อผิดพลาดอื่นๆที่เกิดขึ้น ในกรณีอื่นๆหมายถึงสำเร็จ แม้ว่าแพ็กเกจยังไม่ได้ถูกส่งไปจริงๆ สังเกตุว่าบัฟเฟอร์ที่ส่งไปยัง *send_pkt()* ถูกกำหนดว่าสามารถแก้ไขได้เมื่อเรียกเสร็จ ในขณะที่ *as_send_pkt()* บัฟเฟอร์จะถูกจัดลำดับโดยไดรเวอร์ ถ้ามีข้อผิดพลาดเกิดขึ้น *upcall* จะไม่ทำงาน

*2.4.5.10 set_rcv_mode() *extended driver function*

function set_rcv_mode(handle, mode)

input	handle
	mode
error return	carry flag set
	error code

possible errors BAD_HANDLE

BAD_MODE

non-error carry flag clear

เป็นการกำหนดการทำงานของการรับข้อมูลแพ็กเกจโดยเกี่ยวเนื่องกับแฮนด์เดิลที่ต้องการจะรับ ซึ่งมีโหมดต่างๆดังนี้

1 ไม่รับแพ็กเกจ

2 รับเฉพาะที่ส่งมาอินเทอร์เฟสนี้

3 โหมด 2 รวมกับบอร์ดคาสท์แพ็กเกจ

4 โหมด 3 รวมกับลิมิตเตดมัลติคาสท์แพ็กเกจ

5 โหมด 3 รวมกับมัลติคาสท์แพ็กเกจ

6 ทุกแพ็กเกจ

หมายเหตุ

- อินเทอร์เฟสไม่ทุกอันที่สนับสนุนการรับแบบ ทุกแพ็กเกจ และรีซีฟเวอร์โหมดมีผลกระทบต่ออินเทอร์เฟสโดยตรง ไม่เกี่ยวกับแฮนด์เดิล
- โหมด 3 เป็นค่าเริ่มแรก และถ้า ฟังก์ชั่น *set_rcv_mode()* ไม่ได้กำหนดไว้ จะถือว่าเป็น โหมด 3

2.4.5.11 get_rcv_mode() extended driver function

function get_rcv_mode(handle, mode)

input handle

error return carry flag set

error code

possible errors BAD_HANDLE

non-error carry flag clear

mode

ส่งค่าโหมดการรับปัจจุบัน

2.4.5.12 get_statistics() extended driver function

function get_statistics(handle)

input handle

error return carry flag set

error code

possible errors BAD_HANDLE

non-error carry flag clear

stats

```
struct statistics {
        unsigned long   packets_in;     /* Totals across all handles */
        unsigned long   packets_out;
        unsigned long   bytes_in;       /* Including MAC headers */
        unsigned long   bytes_out;
        unsigned long   errors_in;      /* Totals across all error types */
        unsigned long   errors_out;
        unsigned long   packets_lost;   /* No buffer from receiver(), card */
                                        /*  out of resources, etc. */
};
```

ส่งค่าพอยน์เตอร์ ไปยังโครงสร้างข้อมูลสถิติสำหรับอินเทอร์เฟส ซึ่งค่าเก็บอยู่ในรูป 80xx 32 บิต

*2.4.5.13 set_address() *extended driver function*

```
function set_address(addr, len)
                addr
                len
error return    carry flag set
                error code
possible errors CANT_SET
                BAD_ADDRESS
non-error       carry flag clear
                length
```

การเรียกนี้ใช้เมื่อโปรแกรมหรือโปรโตคอลสแตก ต้องการเจาะจงใช้ฮาร์ดแวร์แอดเดรสหมายเลขนั้นๆ

BAD_ADDRESS หมายความว่าค่า len น้อยไปหรือมากไป หรือ แอดเดรสไม่ถูกต้อง แพ็กเกจไดรเวอร์จะปฏิเสธการเปลี่ยนแอดเดรส ถ้ามีแฮนด์เดิลมากกว่าหนึ่งเปิดใช้งานอยู่

2.4.6 หมายเลขฟังก์ชั่นเรียกและพารามิเตอร์

ในตัวเลขต่อไปนี้ถูกกำหนดเพื่อใช้ในการทำงานของ แพ็กเกจไดรเวอร์ ซึ่งกำหนดในค่า รีจิสเตอร์ AH เพื่อเรียกใช้งาน แพ็กเกจไดรเวอร์

driver_info	1
access_type	2
release_type	3
send_pkt	4
terminate	5
get_address	6

reset_interface	7
+get_parameters	10
+as_send_pkt	11
*set_rcv_mode	20
*get_rcv_mode	21
*set_multicast_list	22
*get_multicast_list	23
*get_statistics	24
*set_address	25

+ หมายถึง ฟังก์ชั่นฟังก์ชั่นประสิทธิภาพสูง

* หมายถึง ฟังก์ชันเพิ่มเติม

ค่า AH ตั้งแต่ 128 ถึง 255 สงวนไว้ใช้สำหรับการพัฒนาอื่นๆ ซึ่งนอกเหนือข้อกำหนด

รหัสผิดพลาด (Error codes)

ในการเรียก แพ็กเกจไดรเวอร์ ถ้ามีข้อผิดพลาดเกิดขึ้นจะมีการเซตค่าแฟลกตัวทด (carry flag) และ ค่ารหัสผิดพลาดจะถูกกำหนดในรีจิสเตอร์ DH (ซึ่งรีจิสเตอร์นี้จะไม่ถูกใช้ในการส่งค่าผ่าน ฟังก์ชั่นแต่ถูกใช้เพื่อส่งรหัสผิดพลาดกลับมา) ซึ่งกำหนดดังนี้

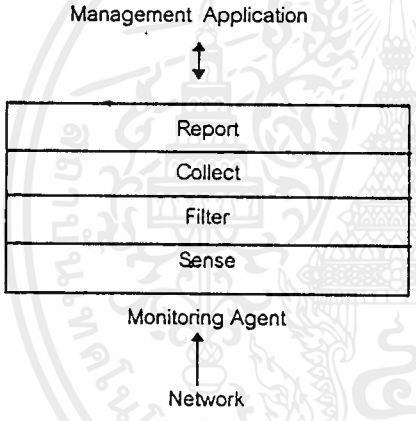
1	BAD_HANDLE	หมายเลขแฮนด์เดิลผิด
2	NO_CLASS	ไม่พบคลาสที่กำหนด
3	NO_TYPE	ไม่พบชนิดที่กำหนด
4	NO_NUMBER	ไม่พบหมายเลขที่กำหนด
5	BAD_TYPE	กำหนดชนิดแพ็กเกจผิด
6	NO_MULTICAST	อินเทอร์เฟสนี้ไม่สนับสนุนมัลติคาสท์
7	CANT_TERMINATE	ไม่สามารถสิ้นสุดการทำงานแพ็กเกจไดรเวอร์
8	BAD_MODE	กำหนดวิธีการแพ็กเกจผิด
9	NO_SPACE	การทำงานผิดพลาดไม่มีที่ว่าง
10	TYPE_INUSE	ชนิด ที่กำหนดไม่สามารถใช้งานได้เนื่องจากถูกใช้งานอยู่
11	BAD_COMMAND	คำสั่งนอกเหนือจากที่กำหนด
12	CANT_SEND	ไม่สามารถสส่งแพ็กเกจได้เนื่องจากฮาร์ดแวร์
13	CANT_SET	ไม่สามารถ์เปลี่ยนฮาร์ดแวร์แอดเดรสได้เนื่องจากมีแฮนด์เดิล มากกว่า 1 เปิดอยู่
14	BAD_ADDRESS	ผิดรูปแบบ หรือ ขนาดฮาร์ดแวร์แอดเดรส
15	CANT_RESET	ไม่สามารถเริ่มใหม่ได้เนื่องจากมีแฮนด์เดิลมากกว่า 1 เปิดอยู่

2.5 หลักการของเน็ตเวิร์คมอนิเตอร์ริง (Design of Network Mornitors)

2.5.1 ฟังก์ชั่นทั่วไปของเน็ตเวิร์คมอนิเตอร์ริงเอเจนท์

มอนิเตอร์ริงเอเจนท์ (monitoring agent) บางตัวไม่ได้มีฟังก์ชั่นครบทั้งหมด เช่น เรียลไทม์ดาต้าอานาไลเซอร์ (real-time data analyzer) จะแสดงผลข้อมูลไปยังหน้าจอแสดงผล โดยตรงไม่ต้องเก็บลงในสื่อบรรจุ (storage media) เช่น ฮาร์ดดิสก์ แต่จะมีฟังก์ชันพื้นฐานเหมือนกันดังนี้

1. ส่วนรับข้อมูล (Sensing)
2. ส่วนกรองข้อมูล (Filter)
3. ส่วนรวบรวมข้อมูล (Collecting)
4. ส่วนแสดงผล (Reporting)



รูปที่ 2.27 หลักการของแต่ละฟังก์ชั่นในเน็ตเวิร์คมอนิเตอร์ริง

2.5.2 ประเภทของการมอนิเตอร์

เน็ตเวิร์คมอนิเตอร์ริงเอเจนท์ สามารถแบ่งออกเป็นประเภทได้ 2 วิธีด้วยกัน คือ

1. แบ่งตามเลเยอร์ที่เอเจนท์ทำการมอนิเตอร์อยู่
2. แบ่งตามลักษณะการมอนิเตอร์คือ เอเจนท์นั้นเป็นแบบ อินทิเกรตเด็ด (Integrated) หรือ เอ็กเทอร์นอล (External) กับ ส่วนประกอบหรือทรัพยากรที่ทำการมอนิเตอร์

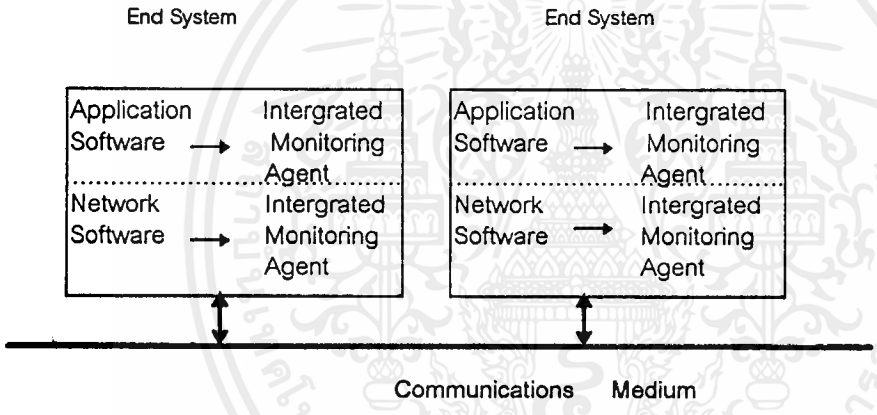
วิธีการแบ่งประเภททั้ง 2 วิธี มีผลต่อการออกแบบและการใช้งานมอนิเตอร์ ดังนั้นเราจะมากล่าวถึงแต่ละประเภทว่ามีลักษณะอย่างไร

2.5.2.1 อินทิเกรตเต็ดมอนิเตอร์เอเจนท์ (Integrated monitoring agents)

อินทิเกรตเต็ดมอนิเตอร์เอเจนท์ สามารถรวมไว้เป็นส่วนหนึ่งของเน็ตเวิร์คแอพพลิเคชั่น หรือในตัวของ เน็ตเวิร์คซอฟแวร์เลยก็ได้ ซึ่งในกรณีหลังนั้น เราต้องทำการรวมเอาส่วนต่าง ๆ ต่อไปนี้เข้ามาไว้ด้วย

- เน็ตเวิร์คเซอร์วิส (network service) ของบางระดับชั้น เช่น รีโมทโพรซีเดอร์คอล (RPC:Remote Procedure Call) คอมมิวนิเคชั่นเซอร์วิส (communication service)
- เน็ตเวิร์คซอฟแวร์ ในพื้นฐานระบบปฏิบัติการเช่น เน็ตเวิร์คดีไวค์ ไดรเวอร์ (network device drivers)
- ส่วนหนึ่งของ ส่วนประกอบเน็ตเวิร์ค (network component) เช่น ซอฟแวร์ หรือ เฟิร์มแวร์ (firmware) ที่ทำงานในฟรอนท์เอ็นคอนเซนเตรเตอร์ (front-end concentrator)

จากตัวอย่างที่กล่าวมานี้ เราจำเป็นต้องมี ตัวนับเหตุการณ์ (event counters) ในทุก ๆ เน็ตเวอร์คเลเยอร์ซอฟแวร์ที่ทำหน้าที่นี้ก็คือมอนิเตอร์เอเจนท์นั่นเอง



รูปที่ 2.28 ไดอะแกรมของเน็ตเวิร์คมอนิเตอร์เอเจนท์

ประโยชน์ของอินทิเกรตเตดมอนิเตอร์

1. สามารถทำการวิเคราะห์ในระยะไกล (remote diagnostic capability) ได้
2. ในกรณีที่มอนิเตอร์ริงเอเจนท์ รวมอยู่กับแอพพลิเคชั่นเฉพาะอย่าง มันสามารถสนองความต้องการของแอพพลิเคชั่นนั้น ๆ มากกว่าจะทำฟังก์ชันทั่วไป
3. ประหยัด เนื่องจากบางระบบมีบริการของมอนิเตอร์อยู่แล้ว ไม่ต้องไปเสียค่าใช้จ่ายในส่วนนี้อีก

ข้อเสียของอินทิเกรตเตดมอนิเตอร์

1. ถ้ามอนิเตอร์ริงเอเจนท์ทำงานบนรูปแบบ (platform) เดียวกันกับตัวบริการด้าน เครือข่าย และแอพพลิเคชั่น ตัวมอนิเตอร์จะทำให้ประสิทธิภาพของ แอพพลิเคชั่นลดลง

2. ยุ่งยากในการเก็บข้อมูลที่ได้จากมอนิเตอร์ริงเอเจนท์ และ ค่าหน่วงเวลาที่เกิดในช่วงเก็บข้อมูล จะทำให้ข้อมูลบางตัวไม่สามารถเก็บได้ทัน

3. มอนิเตอร์ริงเอเจนท์ ต้องปรับเปลี่ยนไปตามซอฟต์แวร์ที่มันไปรวมอยู่ด้วย

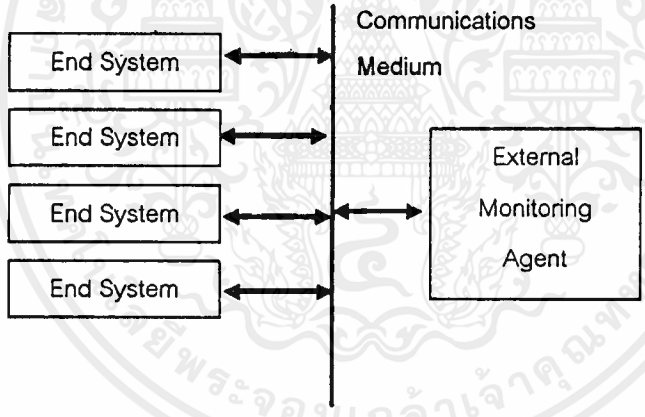
4. ตัวมอนิเตอร์ จะให้รูปแบบของรายงาน ที่ต่างกัน สามารถแก้ได้โดยใช้รูปแบบของรายงานที่เป็นมาตรฐาน

5. การมอนิเตอร์เครือข่าย มีความสำคัญน้อย เมื่อเทียบกับแอพพลิเคชั่น หรือ โอเอส

2.5.2.2 เอ็กเทอร์นอลมอนิเตอร์ (External monitoring agents)

เอ็กเทอร์นอลมอนิเตอร์เอเจนท์ทำงานในคอมโพเน้นท์ที่ไม่เกี่ยวข้องกับซอฟแวร์ที่ทำหน้าที่เน็ตเวิร์คเซอร์วิส เอเจนท์ชนิดนี้สามารถดักจับได้เฉพาะ แชร์เสตท (shared states) ที่ถูกอ้างจากเน็ตเวิร์คโปรโตคอลเท่านั้น และสามารถดัดแปลงได้ง่าย นิยมใช้ในระบบแลน

เอ็กเทอร์นอลมอนิเตอร์เอเจนท์ไม่ต้องติดต่อกับระบบแลนสามารถนำไปติดตั้งไว้ระหว่างพอยน์ทูพอยน์คอมมิวนิเคชั่นลิงค์ (point-to-point communication link) หรือ ติดต่อกับบัสในระบบคอมพิวเตอร์



รูปที่ 2.29 ไดอะแกรมของเอ็กเทอร์นอลมอนิเตอร์

ประโยชน์ของเอ็กเทอร์นอลมอนิเตอร์เอเจนท์

1. สามารถทำเป็นฮาร์ดแวร์ได้ โดยไม่ไปแย่งทรัพยากรกับเน็ตเวิร์คเซอร์วิส และ แอพพลิเคชั่น ช่วยลดสาเหตุที่ทำให้ประสิทธิภาพของ เน็ตเวิร์คลดลงโดยมอนิเตอร์ได้. สามารถใช้ฮาร์ดแวร์และซอฟแวร์พิเศษ เพื่อนำมาใช้สำหรับทำมอนิเตอร์ริงซึ่งจะไม่มีในมอนิเตอร์ริงเอเจนท์แบบ อินทิเกรตเตด

2. ประหยัด

3. เนื่องจากความเป็นอิสระในการใช้งาน ทำให้ปรับปรุงและขยายการใช้งานได้ง่าย

ข้อเสีย

1. ความปลอดภัยในการจัดการน้อย

2. ข้อมูลที่ได้ไม่สามารถรับประกันได้ว่าแสดงสถานะจริง ๆ ของ คอมโพเน้นท์ที่เรามอนิเตอร์จะเป็นเพียงค่าประมาณเท่านั้น

3. เนื่องจากต้องเก็บข้อมูลของเอ็นด์ซิสเต็มจำนวนมาก ตัว เอเจนท์ต้องมีความสามารถสูงมาก ทำให้เพิ่มความยุ่งยากในการผลิตเพื่อทำให้ไม่มีข้อจำกัดในการทำงาน

2.5.3 การประยุกต์ใช้งาน

ปัจจัยที่มีผลต่อการตัดสินใจใช้ อินทิเกรตเตด หรือ เอ็กเทอร์นอลมอนิเตอร์ริงเอเจนท์มีดังนี้

1. แบนด์วิธ (Bandwidth) ของ เน็ตเวิร์คคอมมิวนิเคชั่นมีเดีย (network communication media)
2. สิทธิในการใช้สื่อนั้นๆ
3. ความเร็วของหน่วยประมวลผล
4. ราคาของหน่วยความจำ

เน็ตเวิร์คแบนด์วิธ (Network Bandwidth)

ถ้าแบนด์วิธของเน็ตเวิร์คเพิ่มขึ้น จะมีผลกระทบต่อเอ็กเทอร์นอลมอนิเตอร์ริงเอเจนท์เพราะว่ามีข้อมูลจำนวนมากขึ้นที่ต้องทำการประมวลผล

ใช้สื่อร่วมกัน (Shared medium)

ในเน็ตเวิร์คที่เป็นแบบ พอยน์ทูพอยน์ลิงค์ไม่เหมาะสมที่จะใช้เอ็กเทอร์นอลมอนิเตอร์ริงเอเจนท์เนื่องจาก เอเจนท์หนึ่ง ๆ สามารถมอนิเตอร์การติดต่อระหว่าง 2 ระบบปลายทาง (End Systems) เท่านั้น การจะให้มีการมอนิเตอร์ทุก ๆ ส่วนในเน็ตเวิร์คต้องใช้ค่าใช้จ่ายสูง

จากเหตุผลที่กล่าวมา ทำให้เราใช้เอ็กเทอร์นอลมอนิเตอร์ริงเอเจนท์สำหรับทำทดสอบ (diagnostic) เป็นส่วนใหญ่ โดยใช้ในการวินิจฉัยเฉพาะลงไปในกรณีมีปัญหาเกิดขึ้นในส่วนใดส่วนหนึ่ง

2.5.4 สิ่งที่จะมอนิเตอร์ในแต่ละเลเยอร์

ในแต่ละเน็ตเวิร์คเลเยอร์ต้องการรายละเอียดในการมอนิเตอร์ต่างกัน ดังนี้

1. ดาต้าลิงค์เลเยอร์ ใช้มอนิเตอร์ริงในการตรวจสอบ ซอฟแวร์และฮาร์แวร์ที่ผิดพลาด ซึ่งเป็นผลมาจากการ คอรัปชั่น (corruption) หรือการสูญหายของข้อมูล

2. เน็ตเวิร์คเลเยอร์ ใช้มอนิเตอร์ริงในการรายงานถึง เส้นทางการเชื่อมต่อ(วงจร) ว่า ใช้การได้ หรือ ใช้การไม่ได้

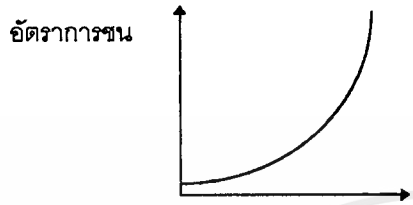
3. ทรานสปอร์ตเลเยอร์ ใน เลเยอร์นี้มอนิเตอร์ริงสามารถช่วยในการจัดการเน็ตเวิร์คและ การทำการวางแผนโดยดูจากระดับที่สัมพันธ์กันของ การใช้งาน ของแต่ละ ทรานสปอร์ตโปรโตคอล

4. เซสชั่นเลเยอร์ ดูเกี่ยวกับปริมาณการใช้งาน (workload) ของเน็ตเวิร์คที่ใช้โดยแอพพลิเคชั่น และ/หรือ ผู้ใช้ซึ่งเป็นประโยชน์ต่อการทำ โหลดบาลานซิ่ง (load balancing) และการทำ แอคเคานท์ติ้งแอพ พลิเคชั่น(accounting applications)

5. แอพพลิเคชั่นเลเยอร์ ใช้ตรวจสอบข้อผิดพลาดของ เซิฟเวอร์โปรเซส (server process) ใน ไคลนท์ /เซิฟเวอร์แอพพลิเคชั่น เป็นต้น

2.6 ประสิทธิภาพและปัญหาของเน็ตแวร์แลน (Netware LANs Performance and Troubleshooting)

2.6.1 บทบาทของการสื่อสารบนเน็ตเวิร์ค



รูปที่ 2.30 ประสิทธิภาพของอีเทอร์เนต

ในการวัดปริมาณการใช้งานนั้นเราวัดขนาดของข้อมูลต่อเวลาเทียบกับแบนด์วิธ (Bandwidth) ซึ่งถ้ามีการใช้งานน้อย นั่นคือน้อยกว่า 5 เปอร์เซนต์ ของแบนด์วิธทั้งหมด (total bandwidth) เมื่อมีการใช้งานเพิ่มขึ้น อัตราการชนกัน (collision rate) ของข้อมูลยิ่งมีมากขึ้นโดยจากสถิติกล่าวว่าที่ 30 เปอร์เซนต์ประสิทธิภาพการใช้งานของเน็ตเวิร์คจะลดลงอย่างรวดเร็ว

เพราะฉะนั้น สิ่งที่เราต้องวัดเพื่อที่จะตรวจสอบการใช้งานเน็ตเวิร์คคือ

- การใช้งานของเน็ตเวิร์ค
- การใช้งานสูงสุด (Peak)
- การใช้งานโดยเฉลี่ย

ซึ่งที่กล่าวมานี้ จะช่วยในการนำไป รีอาร์เรนท์จิ้ง (rearranging) งานที่ใช้ทรัพยากรของเน็ตเวิร์คสูงได้

ข้อแนะนำ ควรจะวัดทั้งสัปดาห์ โดยเว้นระยะห่างในการจับเป็นครึ่งชั่วโมง

2.6.2 สิ่งที่ควรจะวัดในเน็ตเวิร์คทั่วไป

2.6.2.1 เปอร์เซ็นต์การใช้งานของเน็ตเวิร์ค

ทำได้โดยการวัดทราฟฟิค (traffic) บนเน็ตเวิร์คโดยสุ่มในช่วงเวลาสั้น ๆ สำหรับเปอร์เซ็นต์การใช้งานของอีเทอร์เนตที่ดีต้องไม่เกิน 15 เปอร์เซ็นต์ นอกเหนือจากนี้ก็จะมีการวัดการชนกัน (Collision) ซึ่งมีผลต่อประสิทธิภาพ

2.6.2.2 เน็ตเวิร์คทรูพุท (Network throughput)

วัดจำนวนไบต์ทั้งหมดที่ส่งผ่านเน็ตเวิร์คในเวลาใด ๆ ซึ่งจะวัดได้ 2 แบบ คือ

1. ข้อมูลดิบ (raw data) จำนวนไบต์ทั้งหมดที่ได้รับ

2. ข้อมูลที่ใช้จริง (usable data) จำนวนยูเซอร์ดาต้าในแต่ละโปรโตคอล

เนื่องจากโปรโตคอลต่าง ๆ มี โอเว่อร์เฮด (overhead) ซึ่งมีผลต่อ ดาต้าทรูพุท (data throughput) ด้วยเหตุนี้เราจึงนำมาพิจารณาในการออกแบบเน็ตเวิร์คด้วย

ข้อสังเกต อีเทอร์เนต 10 เม็กกะบิตต่อวินาที (Megabit per Second) ไม่ใช่หมายถึงให้ ดาต้าทรูพุทสูงสุด 10 เม็กกะบิตต่อวินาที ในการออกแบบเราคำนึงว่า 2.5 เม็กกะบิตต่อวินาที คือ ทรูพุทสูงสุด (maximum throughput)

2.6.2.3 เวลาตอบสนองของโปรเซสไฟล์เซิฟเวอร์ (Response time of the file server processes)

คือ การวัดเวลาตอบสนอง(response) ของไฟล์เซิฟเวอร์ (File Server) ต่อการร้องขอ (request) เป็นการวัดประสิทธิภาพของระบบปฏิการ และ ไฟล์เซิฟเวอร์ที่มันทำงานอยู่

2.6.2.4 เน็ตเวิร์คคอนเวอร์เซชั่น (Network Conversation)

อินซิเด้นซ์ (incedence) และ ตำแหน่ง (location) ของ คอนเวอร์เซชั่น เป็นสิ่งที่สำคัญในการนำมาพิจารณา ถ้าเกิดมี คอขวด (bottlenecks) ขึ้นที่ เน็ตเวิร์คฮาร์ดแวร์ เช่น บริดจ์ หรือ เราท์เตอร์

2.6.2.5 เก็บบันทึกข้อผิดพลาดในเน็ตเวิร์ค (Recording network errors)

เก็บบันทึกข้อผิดพลาดที่เกิดขึ้นในเน็ตเวิร์ค เพื่อที่สามารถดูข้อผิดพลาดที่เกิดขึ้นนั้นในภายหลังได้

2.6.3 รูปแบบของแพ็กเกจในอีเทอร์เนต และ ประสิทธิภาพ

2.6.3.1 อีเทอร์เนตเฟรม (Ethernet Frame)

Size in octets	7	1	6	6	1	0-1500	0-46	4
	Preamble	SFD	Destination address	Source address	DLF	Data	PAD	Checksum

SFD,Start of frame delimiter

DLF,Length of data field

PAD,(optional) padding field

รูปที่ 2.31 โครงสร้างแพ็กเกจอีเทอร์เนต

2.6.3.2 อีเทอร์เน็ตแอดเดรส (Ethernet address)

2.6.3.2.1 บรอดคาสท์ (Broadcasts)

2.6.3.2.2 มัลติคาสท์ (Multicasts)

2.6.3.3 ส่วนหัวของโปรโตคอลระดับบน (Higher level protocol header)

2.6.3.3.1 เน็ตเวิร์คเลเยอร์

- ส่วนหัวโนเวลไอพีเอ็กซ์ (Novell's IPX header) มีขนาด 30 ไบต์
- ส่วนหัวอินเทอร์เนตโปรโตคอล (IP header) ขนาดเล็กที่สุดมีขนาด 20 ไบต์

2.6.3.3.2 ทรานสปอร์ตเลเยอร์

- ส่วนหัวโนเวลเอสพีเอ็กซ์ (Novel's SPX header) มีขนาด 12 ไบต์
- ทีซีพีเฮดเดอร์ ขนาดเล็กที่สุดมีขนาด 20 ไบต์

เพราะฉะนั้น ถ้าเราใช้ ทีซีพี/ไอพี โปรโตคอลจะมีโอเวอร์เฮดอย่างน้อย 66 ไบต์ และ ส่วนหัวโนเวลไอพีเอ็กซ์/เอสพีเอ็กซ์ จะมีโอเวอร์เฮด 68 ไบต์

2.6.3.4 ผลกระทบของขนาดแพ็กเกจกับประสิทธิภาพ

ขนาดของแพ็กเกจยิ่งมีขนาดเล็กเท่าไรโอเวอร์เฮดที่เกิดจาก ส่วนหัวของโปรโตคอล (protocol header) ก็จะยิ่งมากเท่านั้น เช่น เมื่อมีการส่ง ตัวอักษรเพียงตัวเดียวจาก เทอร์มินอล (terminal) จะพบว่า ข้อมูลที่ใช้จริงคิดเป็น 2 เปอร์เซนต์ ของเฟรมเท่านั้น แต่ถ้าส่ง แพ็กเกจที่มีขนาดข้อมูลสูงสุด พบว่าข้อมูลผู้ใช้คิดเป็น 95 เปอร์เซนต์ของแพ็กเกจ จะเห็นว่าขนาดแพ็กเกจเป็นปัจจัยสำคัญสำหรับประสิทธิภาพของเน็ตเวิร์ค

ในการส่งข้อมูล ถ้าใช้เฟรมขนาดใหญ่จะช่วยลดจำนวนของแพ็กเกจลง ซึ่งเท่ากับช่วยลดจำนวนการเกิด การชนดังนั้นโอเวอร์เฮดที่เกิดจาก การชนและ การส่งใหม่อีกครั้ง จึงลดลงไปด้วย แต่การเพิ่มขนาดของเฟรมให้ใหญ่ขึ้นจะทำให้เกิดผลเสียได้เช่นกัน ดังนี้ ดูเหมือนว่าเฟรมขนาดใหญ่จะทำให้จำนวนของที่ว่างบนเน็ตเวิร์คลดลง ในขณะที่ ทราฟฟิคมีการใช้งานสูง และขนาดบัฟเฟอร์ที่ต้องการโดยซอฟต์แวร์ที่ควบคุม เน็ตเวิร์คอินเทอร์เฟสต้องเพิ่มขึ้น

แต่จากค้นคว้าพบว่าในทางปฏิบัติ ข้อสำคัญที่ใช้พิจารณาคือ ดาต้าทรูพุท (ตรงข้ามกับมินิมั่มดีเลย์ (minimal delay) ที่ต้องการใน ระบบเรียลไทม์ (real-time systems))ดังนั้นขนาดเฟรมควรจะมีขนาดใหญ่

2.6.4 สิ่งที่ต้องทำการวัดประสิทธิภาพ

2.6.4.1 สิ่งที่ต้องทำการวัด สำหรับ วัดประสิทธภาพของเน็ตเวิร์ค

- การใช้งานเน็ตเวิร์ค
- โปรโตคอลที่ใช้งานอยู่ (protocol in use)
- การกระจายของขนาดเฟรม (frame size distribution)
- โหนดที่ใช้งานสูงสุด (busy nodes)

- โหนดที่ไม่ใช้งาน (idle nodes)
- โหนดที่ไม่ตอบสนอง (unresponsive nodes)
- การสนทนาสูงสุด (busiest conversations)
- เวลาและระดับการใช้งานสูงสุด (peak load times and levels)
- เวลาและระดับการใช้งานต่ำสุด (minimum load times and levels)
- แบนด์วิธที่ใช้ในแต่ละโหนด (bandwidth usage by node)
- แบนด์วิธที่ใช้ในแต่ละโปรโตคอล (bandwidth usage by protocol)

ทั้งหมดนี้เป็น ข้อกำหนดขั้นต่ำของเครื่องมือในการวัดที่ควรพิจารณา นอกเหนือจากนี้แล้วยังมีอย่างอื่นที่ควรวัด ซึ่งมีประโยชน์เมื่อ ใช้ในการออกแบบ และ การพัฒนาสำหรับ ประสิทธิภาพของเน็ตเวิร์คคือ

- โปรโตคอลที่ใช้ในการสนทนา (protocol use in conversation)
- กำหนดค่าในการแจ้งเตือนเหตุร้ายที่เกิดขึ้น
- การเตือนเมื่อมีเหตุเกิดขึ้นแก่เนตเวิร์ค

สิ่งสำคัญอย่างอื่นที่ช่วยในการวัดประสิทธิภาพคือ

- สร้างทราฟฟิคในเน็ตเวิร์ค (Generating network traffic) ช่วยให้สามารถจำลองเน็ตเวิร์คได้โดยการสร้างแพ็กเกจและส่งเข้าไปในเน็ตเวิร์ค
- ผลของการแสดงต้องช่วยให้เข้าใจง่าย และสามารถจับเก็บ (capture) ข้อมูลที่กำหนดเพื่อนำมาวิเคราะห์ภายหลังได้

2.6.4.2 การวัดประสิทธิภาพ

2.6.4.2.1 การใช้งานแบนด์วิธ (Bandwidth usage)

หาค่าเฉลี่ยการใช้งานในช่วงเวลาคงที่(ทุก 1 นาที หรือ ทุก 1 วินาที) ทำได้โดยนับจำนวนทราฟฟิคในช่วงเวลาที่กำหนด หาร ด้วยค่า ทรูพุทสูงสุด ค่าที่คำนวณได้จะไม่ต่ำกว่า 5 เปอร์เซนต์และพิจารณาดังนี้

- 10-15 เปอร์เซนต์ แสดงว่า เน็ตเวิร์คมีการใช้งานน้อยถึงปานกลาง
- 25 เปอร์เซนต์ แสดงว่า เน็ตเวิร์คมีการใช้งานสูงเกิด เน็ตเวิร์คแจมมิ่ง (network jamming), การตอบสนองช้า, มีแพ็กเกจเสียมาก หรือ มีการส่งซ้ำปริมาณมาก

2.6.4.2.2 ข้อผิดพลาดในการส่ง (Transmission errors)

แพ็กเกจที่เกิดการชนกันจะต้องทำการส่งใหม่อีกครั้งหนึ่งโดยส่วนใหญ่จะถูกรายงานว่าซีอาร์ซีผิดพลาด หรือ แพ็กเกจขนาดสั้นกว่าปกติ (runts) ฯลฯ ซึ่งสาเหตุมาจากการชนกัน สำหรับเน็ตเวิร์คที่มีการใช้งานสูง (ประมาณ 20 เปอร์เซนต์) อัตราการชนกันที่ยอมรับได้จะอยู่ระหว่าง 1-2 เปอร์เซนต์

ถ้าทั้งการใช้งานแบนด์วิธและ อัตราแพ็กเกจเสีย (failed packet rates) สูงทั้งคู่ หมายความว่าประสิทธิภาพของเน็ตเวิร์คไม่ได้ดีดังที่ควรจะเป็น ซีอาร์ซีผิดพลาด อาจมีสาเหตุมาจาก เน็ตเวิร์คอิน

เทอร์เฟส เช่น เราทราบว่าแต่ละโหนดมีการใช้งานต่ำ ในขณะที่เน็ตเวิร์คมีการใช้งานสูง สาเหตุหลักส่วนใหญ่จะเกิดจากเน็ตเวิร์คอินเทอร์เฟส สาเหตุอย่างอื่น เช่น ตัวเชื่อมต่อ (connecter) หลวม

2.6.4.2.3 **การวัดที่จำเป็นสำหรับการออกแบบเพื่อแบ่งเน็ตเวิร์ค** (Measurements needed for designs to partition the network)

ทำได้หลายวิธี ดังนี้

- แบ่งตามชนิดของโปรโตคอลที่ใช้ออกจากกัน ใช้ในกรณีที่ พีซีแลน(PC LANs) และ ไมโครคอมพิวเตอร์อยู่ในเน็ตเวิร์คเดียวกัน เป็นวิธีที่ง่ายต่อการนำไปปฏิบัติ แต่ในกรณีที่มีการใช้งานที่หลากหลายอยู่ด้วยกัน ไม่เหมาะอย่างยิ่งที่จะแบ่งโดยใช้โปรโตคอล
- แบ่งตามฟิซิคอลเอดเดรส ใช้ในกรณีที่เรามี เวิร์คกรุป (workgroups) ในแต่ละตำแหน่งซึ่งต้องการติดต่อกับเน็ตเวิร์ครวมเป็นบางครั้ง ส่วนใหญ่จะติดต่อกันเองภายในกลุ่มการที่เราจะทราบว่าผู้ใช้ใดจะอยู่ในกลุ่มเดียวกัน เราต้องเก็บข้อมูลของการสนทนาของแต่ละโหนด

2.6.4.2.4 **ขนาดของแพ็กเกจและประสิทธิภาพ (Packet size and performance)**

ทรูพุทของดาต้าที่ใช้งานจริง (usable data throughput) กับ เน็ตเวิร์คทรูพุทแตกต่างกัน แอพพลิเคชั่นจะเป็นตัวกำหนดขนาดแพ็กเกจที่เหมาะสม (optimum packet size) เช่น ถ้าแอพพลิเคชั่นมีการรับส่งข้อมูลที่มีขนาดใหญ่ ซึ่งจำเป็นต้องแบ่งข้อมูลออกเป็นหลายแพ็กเกจเรากำหนดให้ใช้ขนาดแพ็กเกจสูงสุดของ เน็ตเวิร์คโปรโตคอลที่ใช้จะดีที่สุด (1518 ไบต์ในกรณี อีเทอร์เนต) ซึ่งจะช่วยลดจำนวนของแพ็คเกตที่ใช้ในการส่งข้อมูลและช่วยลดผลของโปรโตคอลโอเวอร์เฮดด้วย

ปัจจัยอื่นที่เป็นข้อจำกัดของ การกำหนดขนาดของเฟรมคือ โครงสร้างของเน็ตเวิร์คเช่นการใช้บริดจ์ และ เราท์เตอร์ จะไม่ยอมให้แพ็กเกจใหญ่ผ่าน(บางที) หรือว่าแอพพลิเคชั่นอื่นต้องการพารามิเตอร์ที่ต่างออกไป

ไม่มีกฎตายตัวสำหรับการกำหนดขนาดของเฟรม ทางเดียวก็คือ เมื่อมีการเปลี่ยนแปลงเราต้องคอยดูผลที่เกิดขึ้น ซึ่งได้จากข้อมูลการมอนิเตอร์ริง (monitoring information) โดยไม่เพียงแต่ดูเฉพาะส่วนที่มีการเปลี่ยนแปลง เราต้องดูผลที่เกิดกับเน็ตเวิร์คโดยรวมด้วย

2.6.4.2.5 **โหนดที่ใช้งานและไม่ใช้งาน** (Active and inactive nodes)

โหนดที่ไม่ตอบสนองคือ โหนดที่ได้รับการติดต่อ แต่ไม่ตอบสนองต่อการร้องขอในขณะที่โหนดไม่ใช้งานจะตอบสนองต่อการร้องขอ แต่ไม่ได้ส่งการติดต่อ

2.6.4.2.6 **กำหนดช่วงเวลาใช้งานสูงสุด** (Definition of peak usage times)

เราต้องบันทึกเวลาที่เกิด การใช้งานสูงสุด และ การใช้งานน้อยสุด เพื่อนำไปทำแผน

การทำงาน เช่น ออโตเมติคแบ็คอัพ (automatic backup), การอัพเดตครั้งใหญ่ (large batch updates) หรือ การส่งข้อมูล

มอนิเตอร์ริงแพคเกจ (monitoring package) จะช่วยได้มาก และยังสามารถสร้างทราฟฟิคเพื่อจำลองการทำงานได้

2.7 ภาษาที่ใช้ในการพัฒนาซอฟแวร์

เนื่องจากการใช้งานจะต้องติดต่อกับ การ์ดเชื่อมต่อเน็ตเวิร์ค(Network Interface card) โดยตรง และเพื่อให้มีประสิทธิภาพมากขึ้นโดยไม่ผ่านขั้นตอนการทำงานของสแต็กโปรโตคอล (Stack Protocol) อื่นๆ และ ประกอบกับได้รับ ตัวโปรแกรมมาจากการพัฒนาต่อเนื่องมาจากรุ่นที่แล้ว จึงเลือกภาษาปาสคาล(Pascal) ซึ่ง ง่ายในการพัฒนาทั้งมีเครื่องมือให้ใช้อย่างง่าย

ภาษาปาสคาลเหมาะสำหรับการพัฒนาโปรแกรม บนพื้นฐานระบบดอส(DOS) และเนื่องจากเคยใช้งานมาก่อนแล้วจึงง่ายต่อการเข้าใจ และการพัฒนา และมีเครื่องมือช่วยในการทำ ส่วนติดต่อผู้ใช้

การพัฒนาโปรแกรมนั้นจะต้องเรียกแพ็กเกจไดรว์เวอร์ โดยเรียกผ่าน อินเทอร์เฟสอินเทอร์รัพเวคเตอร์(Interface Interrupt Vecter) ของแพ็กเกจไดรว์เวอร์ และเพื่อให้โปรแกรมนั้นง่ายต่อการพัฒนา จึงได้แบ่งเป็นโมดูลย่อยๆ แล้วจึงเรียกผ่านอินเทอร์เฟสอินเทอร์รัพเวคเตอร์

บทที่ 3

การวางแผนออกแบบและการสร้าง

3.1 หลักการเบื้องต้นในการสร้าง Software

3.1.1. ศึกษาระบบ (System Study)

ทำการศึกษาระบบโดยพิจารณาถึงความต้องการ และศึกษาสิ่งแวดล้อมที่มีความสัมพันธ์กับระบบที่กำลังศึกษาอยู่ เช่น การใช้งานในระบบเครือข่ายจำเป็นต้องใช้อะไรบ้างในการวิเคราะห์

3.1.2. ศึกษาถึงความเป็นไปได้ (Feasibility Study)

ศึกษาความเป็นไปได้ของระบบว่าเราสามารถนำข้อมูลอะไรได้บ้างใช้วิเคราะห์อะไรได้บ้าง

3.1.3. วิเคราะห์ความต้องการ (Requirement Analysis)

ศึกษาและวิเคราะห์ระบบปัจจุบันที่มีอยู่ พิจารณาข้อดีและข้อเสียของระบบ ทำความเข้าใจระบบให้เป็นอย่างดี และหาความต้องการ (Requirement) ของระบบ อาจจะทำการจำลองระบบต่าง ๆ เพื่อช่วยในการทำความเข้าใจ

3.1.4. กำหนดความต้องการ (Requirement Definition)

เป็นกิจกรรมที่ทำการเปลี่ยนข้อมูลต่าง ๆ ที่รวบรวมได้จากการวิเคราะห์ให้เป็นเอกสารที่กำหนดความต้องการต่าง ๆ ที่ตรงตามที่ผู้ใช้ต้องการ เอกสารต้องเขียนด้วยภาษาที่เข้าใจง่าย ซึ่งผู้ใช้ระดับต้นสามารถอ่านเข้าใจได้ง่าย

3.1.5. กำหนดความต้องการ (Requirement Specification)

เป็นรายละเอียด และข้อกำหนดของแต่ละความต้องการ อธิบายถึงหน้าที่ความต้องการ ในรายละเอียดที่ลึกลงไปควรเขียนด้วยภาษาที่ไม่มีความกำกวม เพื่อไม่ให้สับสนระหว่างผู้อ่านและผู้เขียน

3.1.6. ออกแบบซอฟท์แวร์ (Software Design)

ออกแบบซอฟท์แวร์โดยเลือกภาษาที่ใช้ในการพัฒนา ออกแบบวิธีการที่ใช้ในการเขียนซอฟท์แวร์ เลือกใช้กลยุทธ์ในการเขียนซอฟท์แวร์ เช่น ออบเจ็ค-โอเรียนเต็ด (Object-Oriented) เพิ่มรายละเอียดลงไปในข้อกำหนดที่ทำขึ้นในการออกแบบ

โครงสร้างข้อมูลที่จะใช้ การออกแบบแบ่งเป็น

- ออกแบบสถาปัตยกรรม (Architectural Design)
- ข้อกำหนดเบื้องต้น (Abstract Specification)
- ออกแบบส่วนติดต่อ (Interface Design)
- ออกแบบส่วนเชื่อมต่อ (Component Design)
- ออกแบบโครงสร้างข้อมูล (Data Structure Design)
- ออกแบบวิธีการทำงาน (Algorithm Design)

3.1.7. จัดทำและทดสอบการใช้งาน (Implementation and Unit Testing)

ทำการแยกทดสอบโมดูลย่อยที่สร้างขึ้นว่า ทำงานถูกต้องตามต้องการหรือไม่

3.1.8. รวบรวมและทดสอบระบบ (Integration and System Testing)

เมื่อทำการทดสอบโมดูลย่อย ๆ เสร็จแล้ว ก็นำโมดูลต่าง ๆ มารวมกันเป็นโปรแกรมที่ต้องการแล้วทำการทดสอบการทำงานโดยรวมอีกครั้งหนึ่ง

3.1.9. ใช้งานและดูแล (Operation and Maintenance)

นำซอฟท์แวร์นั้นไปใช้งานจริง และบางครั้งอาจมีการเปลี่ยนแปลงเกิดขึ้นจึงต้องมีการดูแล ดังนั้นซอฟท์แวร์ที่ดีควรออกแบบให้สามารถดูแลได้ง่าย

3.2 การวางแผนและพัฒนาระบบ

ขั้นตอนในการวางแผนและพัฒนาระบบมีดังต่อไปนี้

- ศึกษาการทำงานของเนื้อหาข้อมูลในส่วนต่างๆที่ใช้ในการออกแบบโปรแกรมโดยพิจารณาถึงข้อดีข้อเสียของแต่ละส่วนว่าควรจะปรับปรุงอะไร โดยศึกษาถึงการทำงานของโปรแกรมเดิม
- ทำการศึกษาลักษณะการทำงานของโปรแกรมเก่าทั้งหมด และทำการออกแบบโครงสร้างของโปรแกรมใหม่ เพื่อให้เพิ่มประสิทธิในส่วนต่างๆที่จะเป็นในการใช้งาน
- ออกแบบขั้นตอนหลักของโปรแกรมดังนี้
  - โครงสร้างข้อมูลของแต่ละโมดูล
  - อัลกอริทึมโมดูลย่อยต่างๆ
  - ส่วนแสดงผล
- เลือกถึงวิธีการทำของโปรแกรมให้เหมาะสมตามส่วนต่างๆที่กำหนดไว้คือเลือกพัฒนาโปรแกรมบนภาษาปาสคาลและใช้เครื่องมือต่างๆ

3.3 โครงสร้างข้อมูลของแต่ละโมดูล

3.3.1 โมดูลแสดงคอนเวอร์เซชั่น (Display Conversation)

- ที่อยู่ผู้ส่ง
- ที่อยู่ผู้รับ
- จำนวนแพ็กเกจจากผู้ส่งไปยังผู้รับ
- จำนวนแพ็กเกจจากผู้รับไปยังผู้ส่ง
- จำนวนไบต์จากผู้ส่งไปยังผู้รับ
- จำนวนไบต์จากผู้รับไปยังผู้ส่ง
- จำนวนแพ็กเกจจากผู้ส่งไปยังผู้รับต่อวินาที
- จำนวนแพ็กเกจจากผู้รับไปยังผู้ส่งต่อวินาที
- จำนวนไบต์จากผู้ส่งไปยังผู้รับต่อวินาที
- จำนวนไบต์จากผู้รับไปยังผู้ส่งต่อวินาที
- เวลาที่เริ่มใช้งาน

- เวลาสุดท้ายที่ใช้งาน
- โปรโตคอลที่ใช้งาน

การเก็บข้อมูลลงไฟล์ประกอบด้วย 2 ส่วนด้วยกันคือ ส่วนหัว และ ส่วนข้อมูล ในส่วนหัวนั้นจะเก็บข้อมูลที่จำเป็นโดยบอกถึงสถิติข้อมูลโดยรวม ดังต่อไปนี้

- เวลาเริ่มต้นเก็บข้อมูล
- เวลาสิ้นสุดเก็บข้อมูล
- ช่วงเวลาที่เก็บข้อมูล
- หมายเลขที่อยู่ของเครื่องที่เก็บ
- จำนวนแพ็กเกจทั้งหมด
- จำนวนไบต์ทั้งหมด
- จำนวนข้อผิดพลาดทั้งหมด
- จำนวนการสูญเสียข้อมูล

ส่วนของข้อมูลนั้นจะเก็บข้อมูลที่จับได้ ดังนี้

- ส่วนหัวของแพ็กเกจ
- ส่วนของข้อมูล
- ความยาวของข้อมูล
- เวลาที่ได้รับแพ็กเกจ

โมดูลแสดงโปรโตคอล (DisplayProtocol)

ส่วนนับจำนวนโปรโตคอล

- นับจำนวนแพ็กเกจ
- จำนวนไบต์

การเก็บจำนวนของโปรโตคอลในแต่ละเลเยอร์

- เลเยอร์ 2
- เลเยอร์ 3
- เลเยอร์ 4 มีการแยกเก็บตามชนิดของโปรโตคอลในเลเยอร์ 3 และ 2 ดังนี้
  - IPX Ethernet II
  - IPX Ethernet 802.3
  - IPX Ethernet SNAP
  - IPX Ethernet 802.2
  - IP Ethernet II
  - IP Ethernet SNAP

- เลเยอร์สูงขึ้นไป มีการแยกเก็บตามชนิดของโปรโตคอลในเลเยอร์ 4 และ 2 ดังนี้
  - TCP Ethernet II
  - TCP Ethernet SNAP
  - UDP Ethernet II
  - UDP Ethernet SNAP

3.3.3 โมดูลเก็บสถิติ

โครงสร้างของข้อมูล เป็นดังนี้

- วันที่,เวลา (Date,Time)
- แพ็กเกจต่อวินาที (Packet per Second)
- การใช้งาน (Utilization)
- อัตราการเกิดข้อผิดพลาด (Error rates)
- อัตราการสูญเสียข้อมูล (Drop per Second)
- กิโลไบต์ต่อวินาที (KiloBytes per Second)
- บอร์ดคาสท์ต่อวินาที (Broadcast per Second)
- อันเดอร์ไซส์ต่อวินาที (Undersize per Second)
- โอเวอร์ไซส์ต่อวินาที (Oversize per Second)

3.4 ส่วนอัลกอริทึมของโปรแกรม

ช่วงการออกแบบวิธีการทำงานในส่วนต่างๆซึ่งได้แบ่งเป็นโมดูลหลักแยกจากกันเพื่อง่ายในการพัฒนาโปรแกรมโดยมีส่วนต่างๆดังนี้

- การจับข้อมูล
- การกรองแพ็กเกจ
- การแจ้งเตือนภัย
- การวิเคราะห์และแสดงผล

3.4.1 การจับข้อมูล

ส่วนเริ่มต้นของการเก็บแพ็กเกจ โดยจะต้องคำนึงถึงเรื่องการรับแพ็กเกจซึ่งมักจะมีปัญหาตรงที่ไม่สามารถรับแพ็กเกจได้ทัน ทำให้ต้องสูญเสียแพ็กเกจไป ดังนั้นควรจะมีบัฟเฟอร์เพื่อสำรองข้อมูลและทำการจัดลำดับ เพื่อให้สามารถได้ค่าที่ใกล้เคียงความจริงมากที่สุด

การใช้งานจะเรียกผ่านแพ็กเกจไดรเวอร์ ซึ่งต้องติดต่อผ่านทางบริการอินเทอร์รัป (User Interface Interrupt Service) ที่มีให้ ซึ่งจะผ่านข้อมูลมายังโมดูลต่างๆ ที่เรากำหนด แล้วค่อยรวบรวมข้อมูลหรือจัดทำสถิติต่อไป

3.4.2 การกรองแพ็กเกจ

เมื่อผ่านการกรองขั้นแรกมาแล้วก็จะทำการเก็บข้อมูลลงในบัฟเฟอร์ ในที่นี้สามารถเลือกที่จะเก็บข้อมูลในบัฟเฟอร์ที่เป็นหน่วยความจำ (Memory) หรือฮาร์ดดิสก์ (Hardisk) ก็ได้

ถ้าไม่ได้กำหนดการกรองจะถือว่าให้เก็บทุกแพ็กเกจ ซึ่งอาจทำให้บัฟเฟอร์มีขนาดใหญ่และอาจมีข้อมูลที่เราไม่สนใจรวมอยู่ด้วย ในกรณีนี้จะเป็นการยากที่จะดูแพ็กเกจที่สำคัญหรือมีปัญหา เนื่องจากมีข้อมูลมาก ดังนั้นโปรแกรมสามารถจะทำการกรองอีกขั้นหนึ่งเพื่อให้สามารถดูได้เฉพาะแพ็กเกจที่ต้องการ

3.4.2.1 การกรองขั้นแรก

แบ่งเป็นสองชนิดด้วยกันคือ ลักษณะของแพ็กเกจ หรือ ค่าของฟิลด์

3.4.2.1.1 ลักษณะของแพ็กเกจ

ในโปรแกรมสามารถที่จะกรองขั้นแรกได้ซึ่งลักษณะขึ้นอยู่กับรูปแบบแพ็กเกจและการความถูกต้อง ยกตัวอย่างเช่น

- ทุกแพ็กเกจ
- ทุกแพ็กเกจที่ดี
- ทุกแพ็กเกจที่ผิดพลาด
- แพ็กเกจที่มีขนาดผิดพลาด
- แพ็กเกจที่มีขนาดตามต้องการ

ซึ่งเกณฑ์ดังกล่าวนี้เป็นอิสระกับชนิดของโปรโตคอลที่จับได้ เช่น ในเครือข่ายอีเทอร์เนต แพ็กเกจที่ดีและมีขนาดตามที่ต้องการจะเป็นโปรโตคอลทีซีพี/ไอพี หรือเน็ตแวร์ก็ได้ ดังตารางแสดงถึงคุณสมบัติแพ็กเกจและความหมาย

ลักษณะแพ็กเกจ	ความหมาย
ทุกแพ็กเกจ	ทุกแพ็กเกจทั้งดีและผิดพลาด
ทุกแพ็กเกจที่ดี	ทุกแพ็กเกจที่ไม่มีข้อผิดพลาด
แพ็กเกจที่มีขนาดผิดพลาด	แพ็กเกจที่มีขนาดน้อยกว่า 64 หรือมากกว่า 1518 ไบต์
แพ็กเกจที่มีขนาดตามต้องการ	แพ็กเกจที่มีขนาดตรงกับที่กำหนดไว้

ตารางที่ 3.1 ประเภทของการรับแพ็กเกจ

ในการกรองสามารถเลือกขนาดแพ็กเกจที่ต้องการได้โดยกำหนดขนาดแพ็กเกจต่ำสุดและสูงสุด ดังตัวอย่างต่อไปนี้ ต้องการเก็บข้อมูลซึ่งสั้นกว่าปกติ (แพ็กเกจที่มีขนาดน้อยกว่า 64 ไบต์และ CRC ถูกต้อง) อาจจะเลือกโดยกำหนดดังนี้คือ เลือกขนาดแพ็กเกจที่มีขนาดน้อยกว่า 64

3.4.2.1.2 ค่าของฟิลด์

ในการกรองแพ็กเกจโดยใช้การกำหนดประเภทส่วนหัว ประเภทของโปรโตคอล ระดับเน็ตเวิร์ค โปรโตคอลระดับขนส่ง และโปรโตคอลระดับบนที่ต้องการนั้น จะต้องบอกถึง

ตำแหน่งของข้อมูลของโปรโตคอลที่นำมาใช้ในการกรอง เช่น ต้องการเก็บข้อมูลที่เป็น RIP บนแลนเน็ตแวร์ ขั้นแรกก็ต้องระบุว่าในฟิลด์ซ็อกเกตของผู้ส่ง (Source Socket field) มีค่า 0x0453 และ ในฟิลด์ที่อยู่ของผู้รับ (Destination Address field) มีค่า FF-FF-FF-FF-FF-FF

3.4.2.2 การกรองขั้นสุดท้าย

เมื่อคุณใช้การกรองขั้นสุดท้าย (หรือเรียกอีกอย่างว่าการกรองก่อนแสดงผล) ก่อนที่จะใช้ต้องพิจารณาดูว่า ข้อมูลที่อยู่ในบัฟเฟอร์นั้น มีข้อมูลที่เราต้องการหรือไม่ ดังเช่น คุณต้องการเก็บทุกแพ็กเกจของเน็ตแวร์ที่ใช้เฟรมแบบ อีเทอร์เนต 802.2 ในบัฟเฟอร์ ซึ่งผลที่ได้อาจจะมากเกินไปจึงต้องทำการกรองเฉพาะแพ็กเกจที่ต้องการ โดยกรองขั้นสุดท้ายเพื่อเอาเฉพาะแพ็กเกจที่เป็นเน็ตแวร์แซฟ (SAP) เป็นต้น

3.4.3 การแจ้งเตือนภัย

เป็นส่วนที่สำคัญช่วยให้สามารถดูแลระบบได้ง่ายขึ้นซึ่งจะเตือนเราให้ทราบถึงเหตุการณ์ต่างๆที่เกิดขึ้นแทนที่เราจะเฝ้าดูเหตุการณ์ตลอดเวลา ซึ่งจะเตือนเมื่อมีเหตุการณ์ที่ไม่ปกติ ดังเช่น การเพิ่มของข้อผิดพลาดที่เกิดขึ้นในเครือข่าย เพื่อให้สอดคล้องกับการใช้งานในแต่ละเครือข่ายจำต้องมีการกำหนดเทรโชลด์(Threshold) ให้ตรงกับลักษณะของเครือข่ายนั้น ๆ

การเตือนโดยเทรโชลด์นั้นอยู่บนพื้นฐานของสถิติที่เก็บดังเช่น การใช้งาน แพ็กเกจต่อวินาที ข้อผิดพลาดในขณะที่คำนวนสถิตินั้นก็จะเปรียบเทียบกับค่าเทรโชลด์ว่าจะมีการเตือนหรือไม่

<u>ขั้นแรก</u> เป็นขั้นตอนแรกในการวิเคราะห์คือการรวบรวมเก็บสถิติทั้งหมด สถิติทั้งหมดรวมถึงข้อมูลเกี่ยวกับแพ็กเกจที่พบในเครือข่าย ซึ่งแตกต่างจากแพ็กเกจที่ได้การการกรอง

<u>ขั้นสอง</u> คือการกรองชนิดของแพ็กเกจที่จะเก็บในบัฟเฟอร์ (Buffer) เพื่อใช้ในการตรวจสอบหรือคำนวนสถิติ นับว่าเป็นการกรองขั้นแรก

<u>ขั้นสาม</u> คือการเก็บข้อมูลลงในบัฟเฟอร์ ซึ่งอาจทำได้ 2 วิธีด้วยกัน คือให้เขียนทับข้อมูลที่เก่าที่สุดโดยข้อมูลที่ใหม่ที่สุด (FIFO) หรือ หยุดการเก็บข้อมูล

<u>ขั้นสุดท้าย</u> คือการทำการกรองในขั้นแสดงผลซึ่งจะขอกล่าวในช่วงถัดไป

การทำสถิติทั้งหมดนั้น ต้องคำนวนถึง

- เปอร์เซนต์การใช้งาน
- กิโลไบต์ต่อวินาที
- ข้อผิดพลาดต่อวินาที
- แพ็กเกจต่อวินาที
- การกระจายของขนาดแพ็กเกจ

ดังรูปเมื่อมีแพ็กเกจขนาด 64 ไบต์ เป็นโปรโตคอลเน็ตแวร์บรอดคาสท์ผ่านก็จะถูกเก็บในบัฟเฟอร์ที่จะวิเคราะห์ ก่อนที่จะถูกเก็บนั้นก็จะคำนวนสถิติทั้งหมดและการกรองขั้นแรกเสียก่อน ในช่วงของการเก็บสถิติทั้งหมด จะเก็บถึงการใช้งาน กิโลไบต์ต่อวินาที แพ็กเกจต่อวินาที และการกระจายของขนาดแพ็กเกจ

เมื่อได้รับแพ็กเกจที่ผิดพลาด การ์ดจะแสดงถึงข้อผิดพลาด โปรแกรมก็จะเพิ่มตัวนับจำนวนข้อผิดพลาดต่อวินาทีและ ทำการแก้ไขเพิ่มเติมสถิติโดยรวมซึ่งเก็บข้อผิดพลาดต่อไปนี้

- ความยาวผิดพลาด (Length errors)
- ข้อผิดพลาดจากซีอาร์ซีหรือการจัดเรียง (CRC/Alignment errors)

เมื่อได้รับแพ็กเกจขนาดมากกว่า 1518 ไบต์ ค่าโอเวอร์แพ็กเกจก็จะเพิ่มขึ้นรวมถึงค่าผิดพลาดทั้งหมด จำนวนของแพ็กเกจและข้อผิดพลาดเรื่องขนาด

การเตือนโดยเทรโชลด์สามารถตั้งได้โดยตั้งค่าให้เป็น 1 โปรแกรมก็จะเตือนเมื่อมีข้อมูลที่มีความยาวยาวกว่าปกติ(ยาวกว่า 1518 ไบต์)

3.4.4 ส่วนออกแบบการแสดงผล

ในการรายงานผลนั้นสามารถรวมข้อมูลสำคัญต่างๆ เกี่ยวกับเครือข่าย เช่น สถิติของการใช้งานของเราท์เตอร์ สถิติการใช้งานของไดรว์เวอร์ และอื่นๆ อย่างไรก็ตามเราก็ได้ยึดหลักลักษณะสำคัญที่ควรจะมีในการแสดงผลของประสิทธิภาพระบบเครือข่าย ซึ่งส่วนสำคัญนั้นคือ

- อัตราการใช้งาน (Utilization)
- อัตราข้อผิดพลาด (Error rate)
- จำนวนแพ็กเกจต่อวินาที (Packet per second)
- จำนวนกิโลไบต์ต่อวินาที (Kilobytes per second)
- โฮสที่กำลังใช้งานอยู่มากที่สุด (Most active servers)

3.4.4.1 การแสดงผลแนวโน้มของการใช้งาน (Utilization Trends)

การที่ค่าการใช้งานสูงขึ้นเป็นลักษณะโดยปกติของเครือข่ายซึ่งมีการเจริญเติบโตตามกาลเวลา ในขณะที่มีเวิร์คสเตชั่น (Work Station) ใหม่และมีโปรแกรมเพิ่มขึ้นในเครือข่าย มีผู้ใช้และข้อมูลเพิ่มขึ้นตามสภาพการส่งข้อมูลของสายระบบ (Cabling System) ซึ่งแนวโน้มของการใช้งานจะสามารถตระเตรียมและตัดสินใจถึงความต้องการของการขยายตัวในระบบเครือข่ายได้เป็นอย่างดี

การแสดงเป็นแผนภูมิถึงการใช้งานโดยใช้กราฟจะเป็นการใช้งานที่ง่ายขึ้น โดยที่แผนภูมิแนวโน้มการใช้งานแสดงข้อมูลเป็นช่วงเวลา ดังเช่น วัน หรือ สัปดาห์ หรือ เดือน ซึ่งกราฟจะสูงขึ้นหรือลงตามปริมาณการใช้งานของเครือข่ายเช่นในตอนเช้าเมื่อทุกคนเข้ามาใช้ระบบเครือข่ายหรือตอนสำรองการใช้งานระบบช่วงนี้ระบบเครือข่ายจะมีการใช้งานมากซึ่งจะทำให้กราฟสูงขึ้น

ซึ่ง โครงสร้างในการเก็บข้อมูลแบบกราฟนั้นอาศัยข้อมูลพื้นฐานดังต่อไปนี้

- วันที่ (Date)
- เวลา (Time)
- จำนวนแพกเกตต่อวินาที (Packets per Second)
- จำนวนกิโลไบต์ต่อวินาที (Kilobytes per second)
- ค่าเฉลี่ยของผิดพลาด (Average Errors)

- การใช้งาน (Utilization)

3.4.4.2 อัตราการเกิดข้อผิดพลาด (Error Rate)

จะมีการแสดงข้อมูลแนวโน้มของการเกิดข้อผิดพลาดขึ้นในเครือข่าย ซึ่งทำให้แน่ใจว่าเราทราบถึงชนิดของข้อผิดพลาดที่เกิดขึ้นในระบบเครือข่ายทุกช่วงเวลา ในขณะที่เครือข่ายเจริญเติบโตขึ้น อัตราการเกิดข้อผิดพลาดย่อมเพิ่มตามด้วยเช่นกัน เปรียบเสมือนอาการของระบบเครือข่าย ไม่ว่าจะเป็นการเกิดคอขวดที่สายหรือการต่อสายผิดพลาดหรือการที่ส่วนประกอบเกิดเสีย

เมื่อระบบเครือข่ายใช้งานมากการเกิดข้อผิดพลาดยิ่งมากตามไปด้วย แต่อย่างไรก็ตามถ้าพบว่ามีการผิดพลาดเพิ่มขึ้นแต่การใช้งานกลับไม่เพิ่ม อาจเป็นไปได้ที่ส่วนประกอบของเครือข่ายอาจมีปัญหาเช่นอาจเกิดข้อผิดพลาดที่ตัวการ์ดเชื่อมต่อเครือข่ายหรือตัวรับ

3.4.4.3 แพ็กเกจต่อวินาที

ตัวเลขที่แสดงค่าแพ็กเกจต่อวินาทีบนเครือข่าย จะช่วยให้ทราบถึงปริมาณการจราจรบนสาย ซึ่งไม่เหมือนกันกับค่าการใช้งาน เนื่องจากค่าการใช้งานนั้นคำนวนมาจากจำนวนกิโลไบต์บนเครือข่ายต่อวินาที ซึ่งแพ็กเกจมีขนาดไม่แน่นอนและไม่สัมพันธ์กับการใช้งาน การใช้งานอาจจะเพิ่มขึ้นเมื่อมีจำนวนแพ็กเกจเพิ่มขึ้นหรือขนาดของแพ็กเกจเพิ่มขึ้น

ในแลนเน็ตแวร์ (NetWare LAN) ค่าของจำนวนแพ็กเกจต่อวินาทีบ่งบอกถึงการร้องขอตอบกลับ แพ็กเกจและข้อมูลซึ่งมีการใช้บริการอยู่ในเครือข่าย ถ้าค่าจำนวนแพ็กเกจต่อวินาทีเพิ่มแต่ค่าการใช้งานไม่เพิ่ม นั่นอาจหมายความว่ามีการเพิ่มจำนวนของแพ็กเกจขนาดเล็ก ๆ ซึ่งทำให้การใช้งานไม่เพิ่มขึ้น

3.4.4.4 กิโลไบต์ต่อวินาที

โดยแนวทางของกิโลไบต์ต่อวินาที เราสามารถจะคาดคะเนค่าจริงของการไหลของข้อมูลในเครือข่าย ซึ่งการคำนวนสถิติการใช้งานก็ขึ้นอยู่กับกิโลไบต์ต่อวินาที แล้วเปรียบเทียบกับค่าการใช้งานสูงสุดของเครือข่ายเป็นเปอร์เซนต์ ไม่ว่าจะเป็นเครือข่ายแบบอีเทอร์เนต (10 เมกกะบิตต่อวินาที) หรือโทเคนริง (4 หรือ 16 เมกกะบิตต่อวินาที)

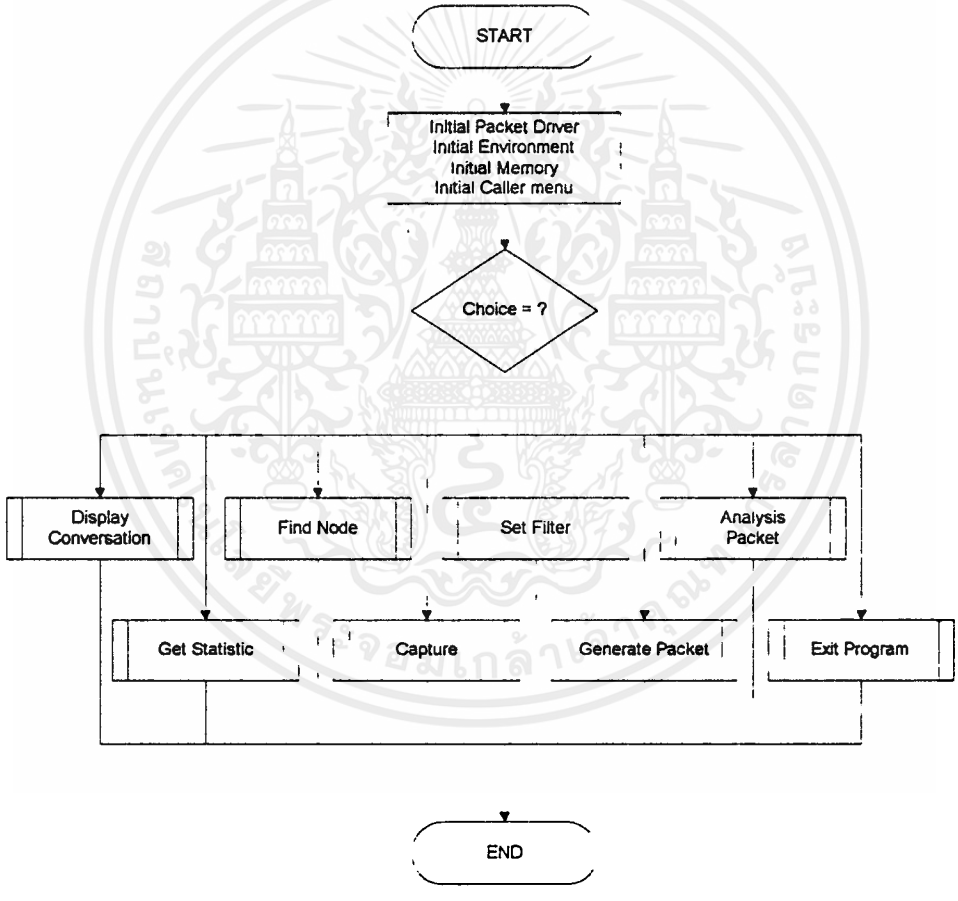
3.4.4.5 โฮสที่กำลังใช้งานอยู่มากที่สุด

ควรทราบถึงการใช้งานโฮสบนเครือข่าย การดูแลการใช้งานโฮสจะช่วยในการแบ่งเบาภาระการทำงานระหว่างโฮส ในการวัดการใช้งานของโฮสควรจะต้องดูจำนวนแพ็กเกจต่อวินาทีจากโฮสเป็นระยะเวลาหลายวันหรือหลายสัปดาห์ ถ้าคุณพบว่ามีจำนวนผู้ใช้มากในหนึ่งเครือข่าย โดยการการเฝ้าดูเซิฟเวอร์(Server) ที่มีการใช้งานมากที่สุด คุณอาจจะหลีกเลี่ยงการใช้งานมากเกินไปได้ ถ้าประสิทธิภาพลดลงแต่กลับมีการโปรเซสแพ็กเกจอยู่ อาจเป็นไปได้ที่เกิดคอขวดที่ การ์ดเชื่อมต่อของเซิฟเวอร์

3.5 การสร้างโปรแกรม

ส่วนเริ่มแรกของโปรแกรมคือส่วนเมนูจะเป็นตัวเลือกเพื่อไปยังส่วนต่างๆของโปรแกรม ซึ่งมีส่วนต่างๆที่ใช้ในโปรแกรมดังนี้

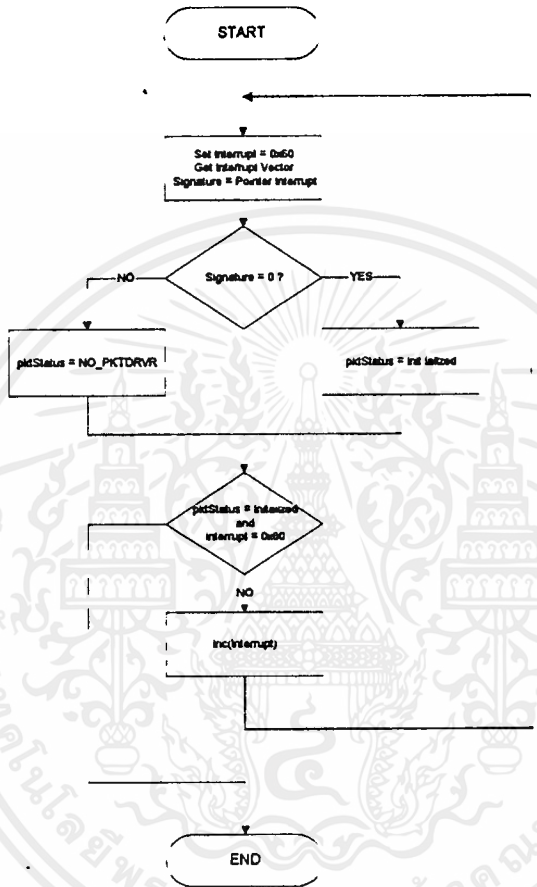
- แสดงการสนทนาระหว่างโฮส (Display Conversation)
- รับและแสดงส่วนของเน็ตเวิร์ค (Get Statistic)
- ตรวจสอบและค้นหาเครื่อง (Find node)
- เก็บแพ็กเกจ (Capture)
- ตั้งค่าการกรองแพ็กเกจ (Set Filter)
- สร้างแพ็กเกจเพื่อทดสอบเน็ตเวิร์ค (Generate Packet)
- วิเคราะห์แพ็กเกจ (Analysis Packet)



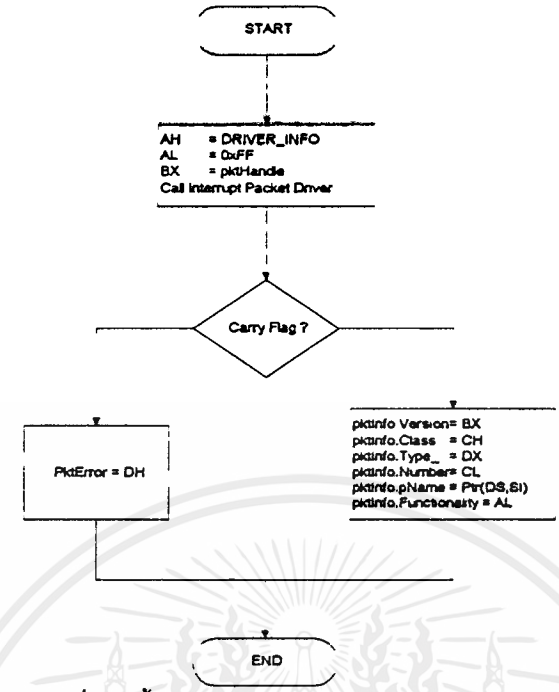
รูปที่ 3.1 ขั้นตอนการทำงานของเมนู

3.5.1 โมดูลเริ่มต้นเมนู

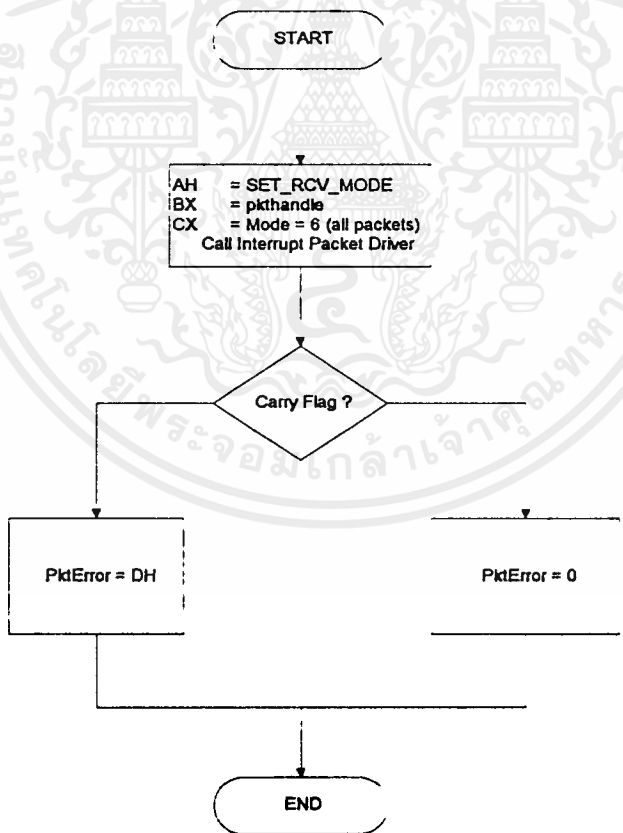
โดยเริ่มแรกของโปรแกรมก็กำหนดการทำงานในส่วนต่างๆ เช่นส่วนของแพ็กเกจไดรเวอร์ ตัวแปร หน่วยความจำ และเมนู แล้วจึงทำโปรเซสต่างๆ ของส่วนโปรแกรม ซึ่งในส่วนการกำหนดค่าของแพ็กเกจ ไดรเวอร์ โดยค้นหาบริการอินเทอร์รัปที่มีให้ (User Interface Interrupt Service) แล้วจึงตั้งค่าการทำงานของแพ็กเกจไดรเวอร์ต่าง ๆ เช่น วิธีการรับแพ็กเกจ การหาค่าแฮนเดิ้ลของบริการ และฟังก์ชั่นที่สนับสนุน



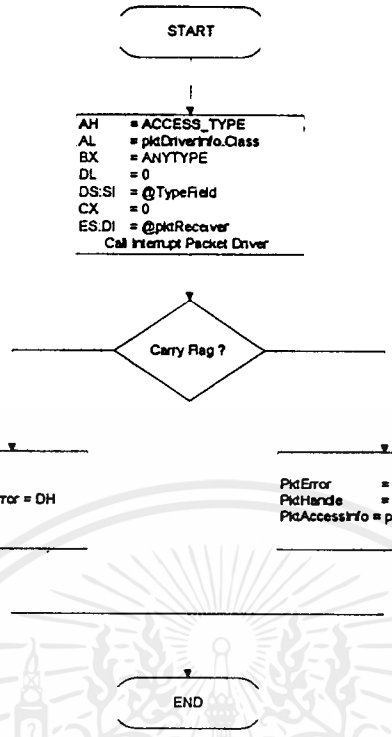
รูปที่ 3.2 ขั้นตอนการหาแพ็กเกจไดรเวอร์



รูปที่ 3.3 ขั้นตอนการรับข้อมูลของแพ็กเกจไดรเวอร์



รูปที่ 3.4 ขั้นตอนการตั้งค่าวิธีการรับแพ็กเกจ

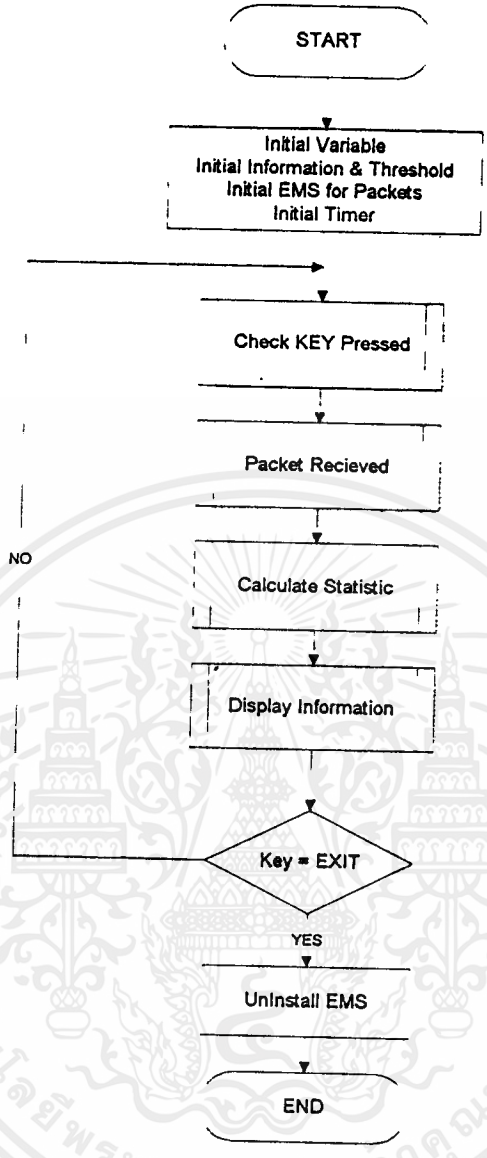


รูปที่ 3.5 ขั้นตอนการตั้งค่าของแพ็กเกจไดรเวอร์

3.5.2 โมดูลแสดงการสนทนา

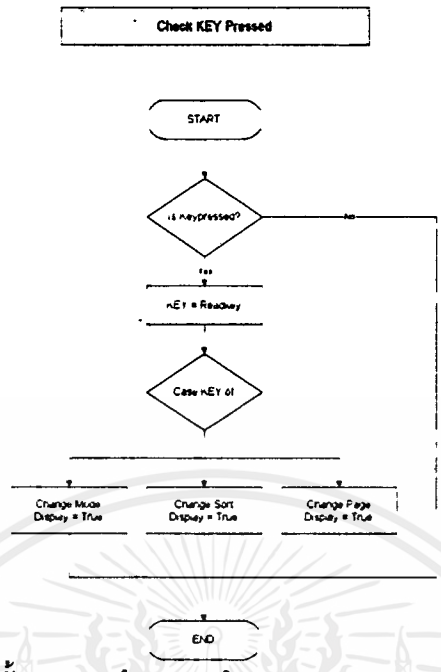
แสดงการสนทนาระหว่างโฮสด้วยกันโดยมีรายละเอียดส่วนหลักๆของโมดูลนี้ดังนี้

- ตรวจสอบคีย์
- รับแพ็กเกจ
- คำนวณสถิติ
- แสดงข้อมูล

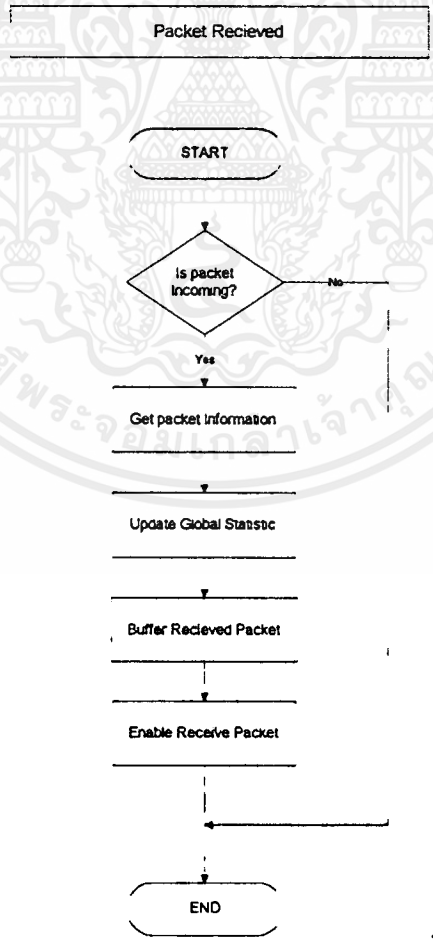


รูปที่ 3.6 ขั้นตอนการทำงานของโมดูลแสดงการสนทนา

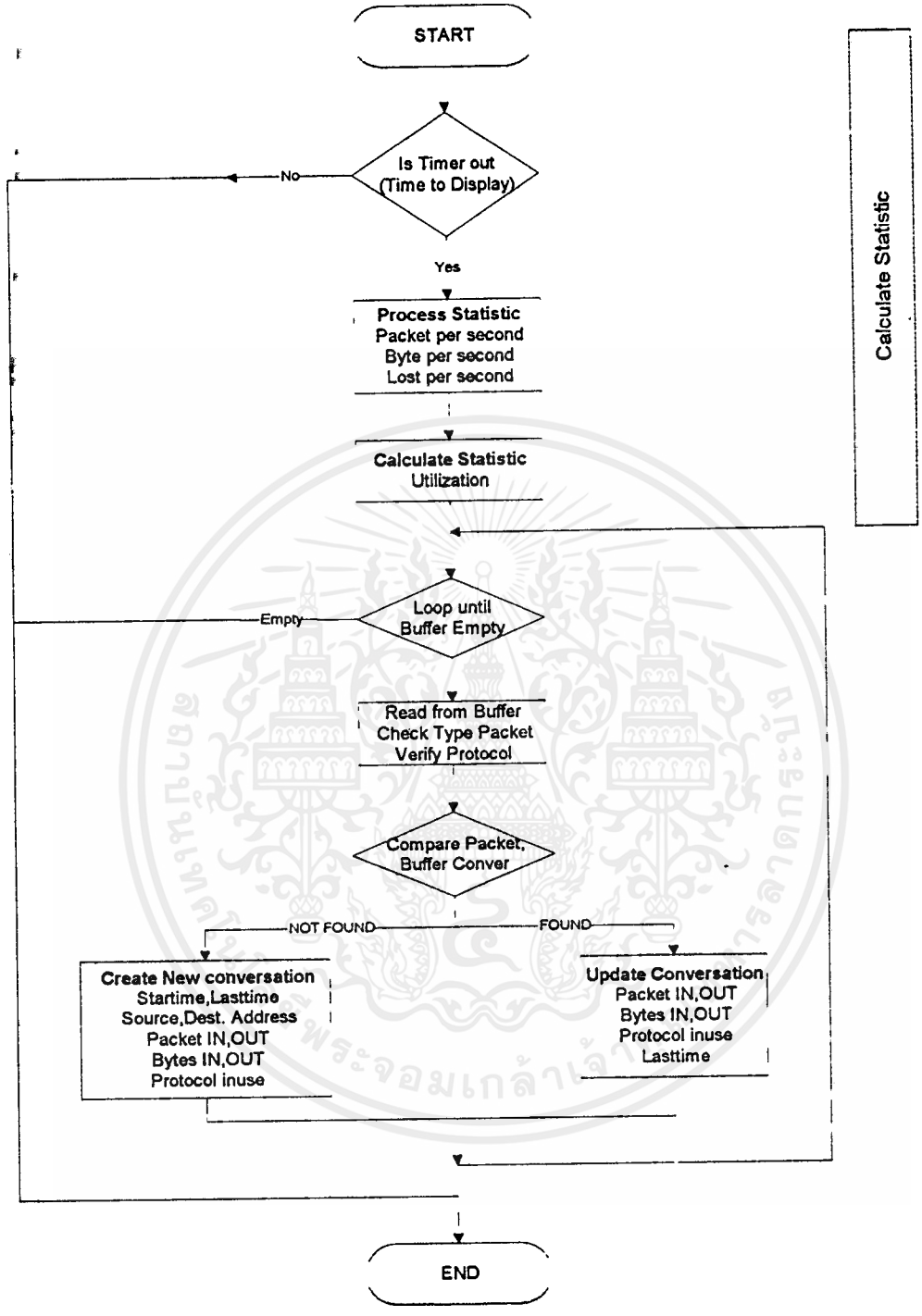
เริ่มต้นก็รับข้อมูลจากแพ็กเกจมาก่อนในช่วงระยะเวลาหนึ่ง โดยเก็บข้อมูลไว้ในบัฟเฟอร์เพื่อให้มีเวลาในการทำงานส่วนของการคำนวนให้มีประสิทธิภาพ และหลังจากนั้นก็แสดงผล



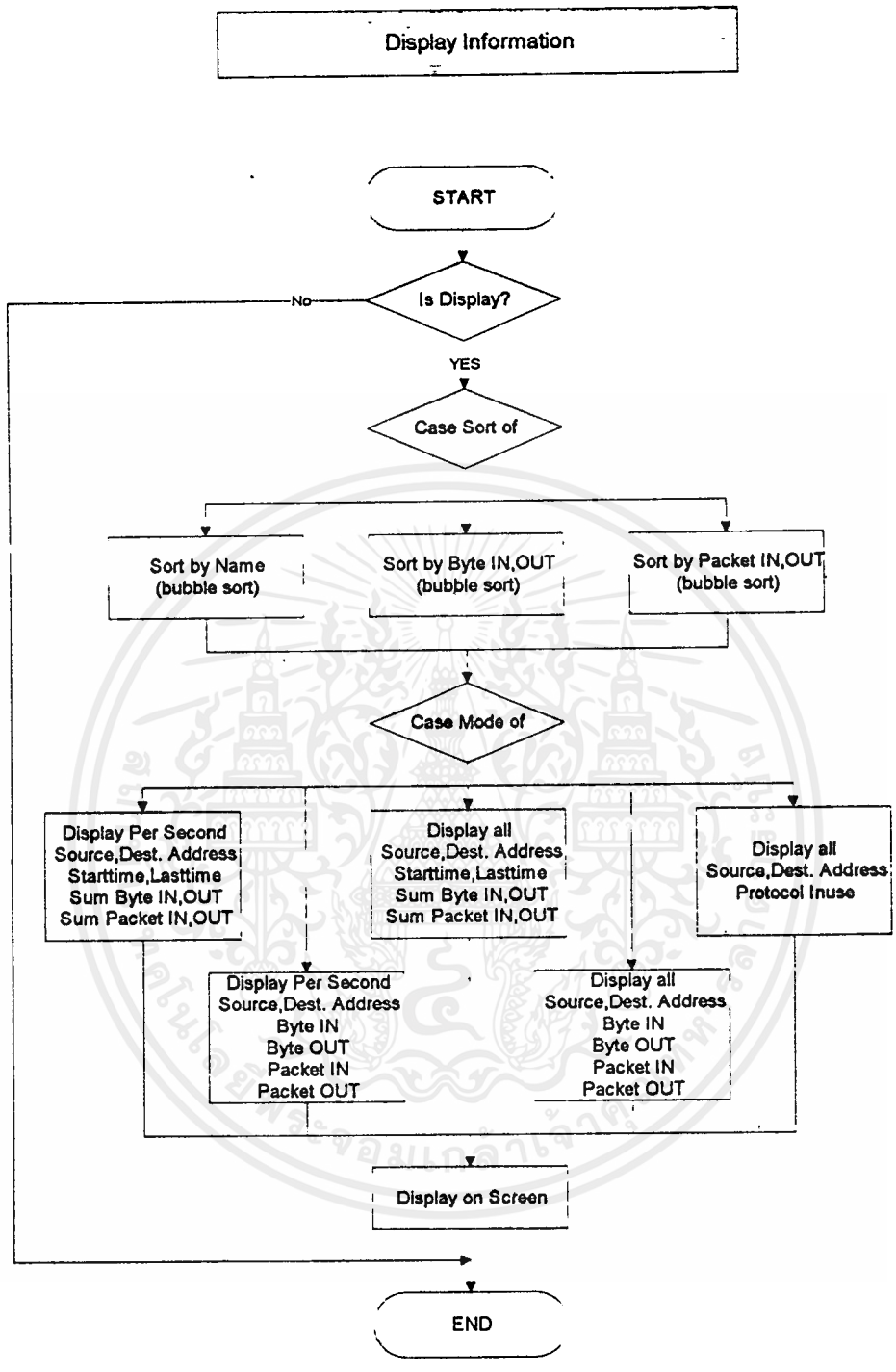
รูปที่ 3.7 ขั้นตอนการทำงานของโมดูลแสดงการสนทนาส่วนรับคีย์



รูปที่ 3.8 ขั้นตอนการทำงานของโมดูลแสดงการสนทนาส่วนรับแพ็กเกจ



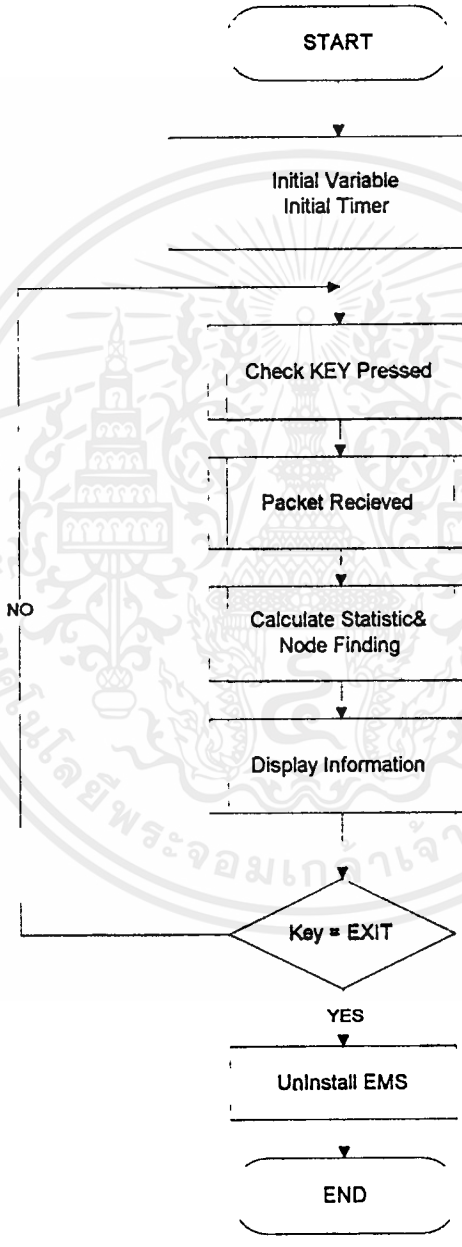
รูปที่ 3.9 ขั้นตอนการทำงานของโมดูลแสดงการลนทนาส่วนคำนวณสถิติ



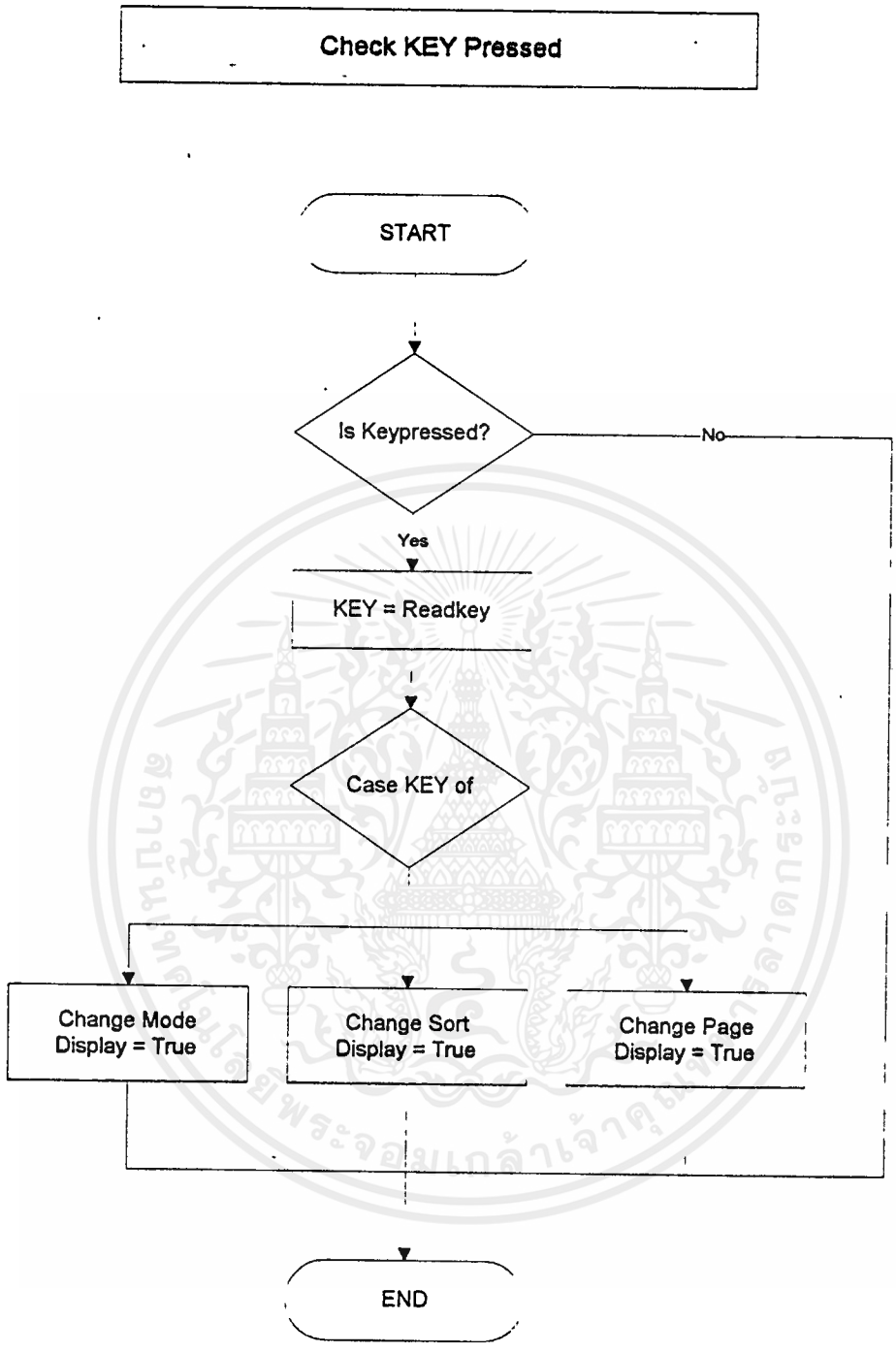
รูปที่ 3.5.10 ขั้นตอนการทำงานของโมดูลแสดงการสนทนาส่วนแสดงผล

3.5.3 โมดูลค้นหาโหนด

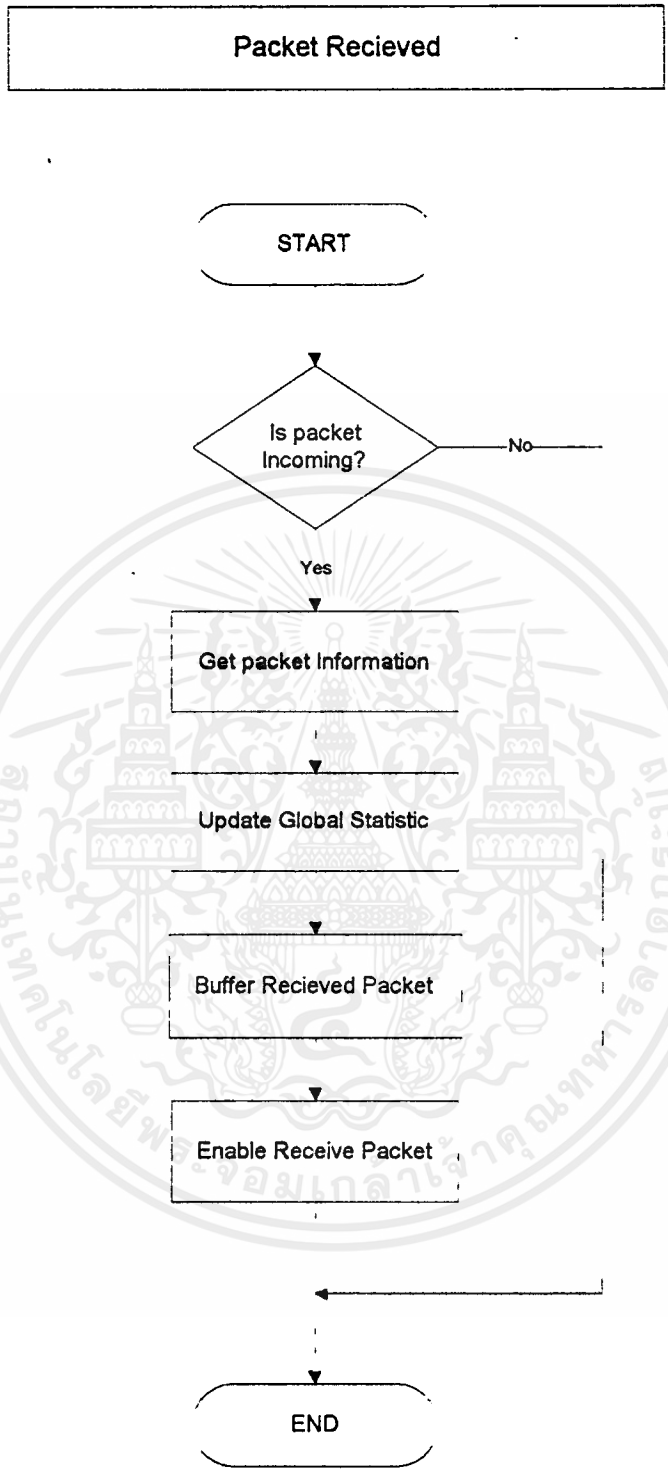
ในส่วนนี้จะแบ่งช่วงการค้นหาโหนดเป็น 2 ช่วงคือช่วงที่ค้นหาจากแพ็กเกจที่วิ่งอยู่ และ จากข้อมูลที่เก็บไว้ ในแบบที่หนึ่งนั้นจะใช้วิธีการรับข้อมูลจากแพ็กเกจแล้วค่อยแสดงข้อมูลต่าง ๆ ในเน็ตเวิร์คจาก แพ็กเกจที่ได้ ในแบบที่สองนั้นจะมีรายชื่อของโฮสไว้แล้วใช้โปรโตคอล ARP และ ICMP ช่วยในการค้นหาโฮส โดยไปถามว่ายังอยู่ดีหรือไม่



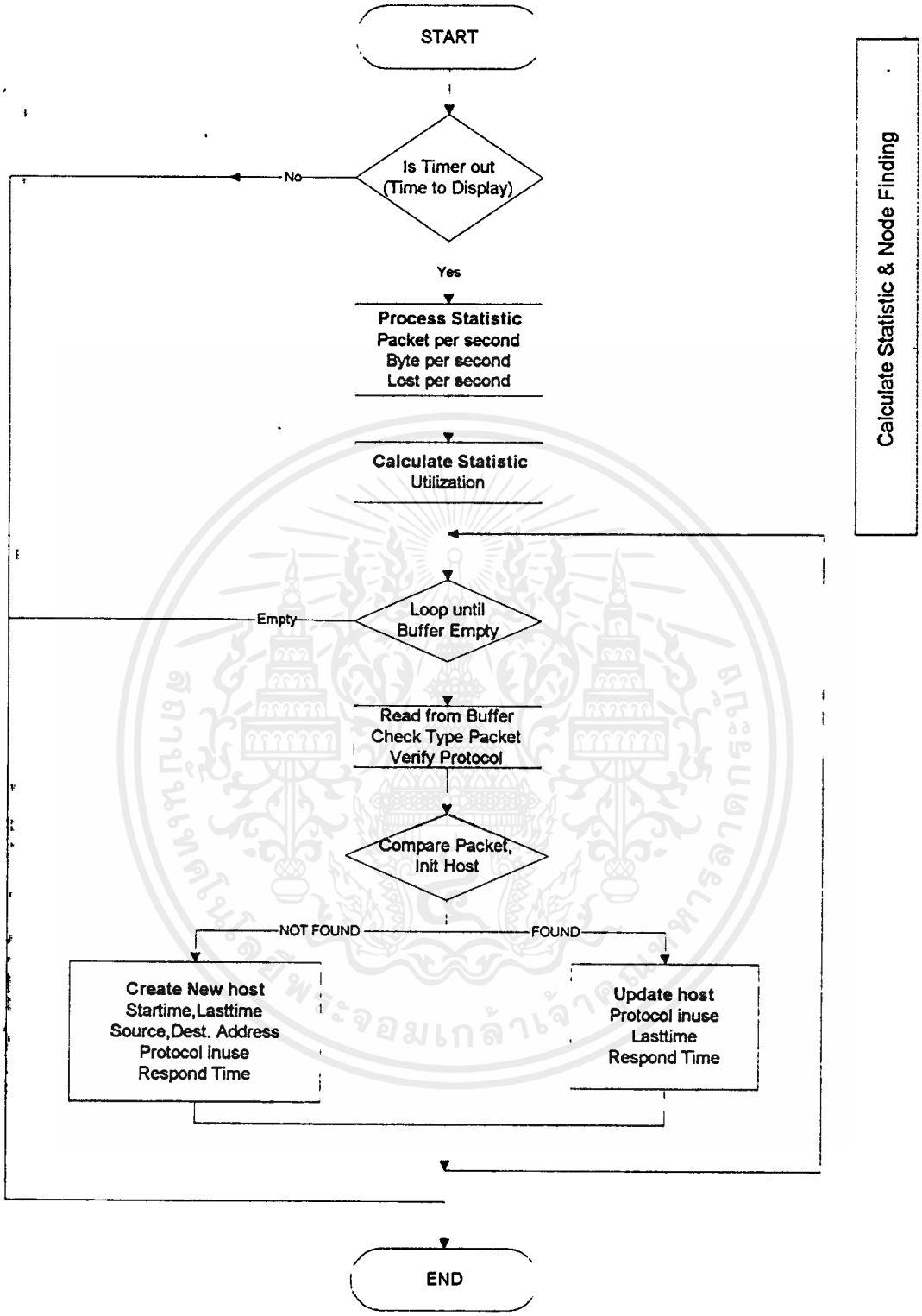
รูปที่ 3.11 ขั้นตอนการทำงานของโมดูลค้นหาโหนด



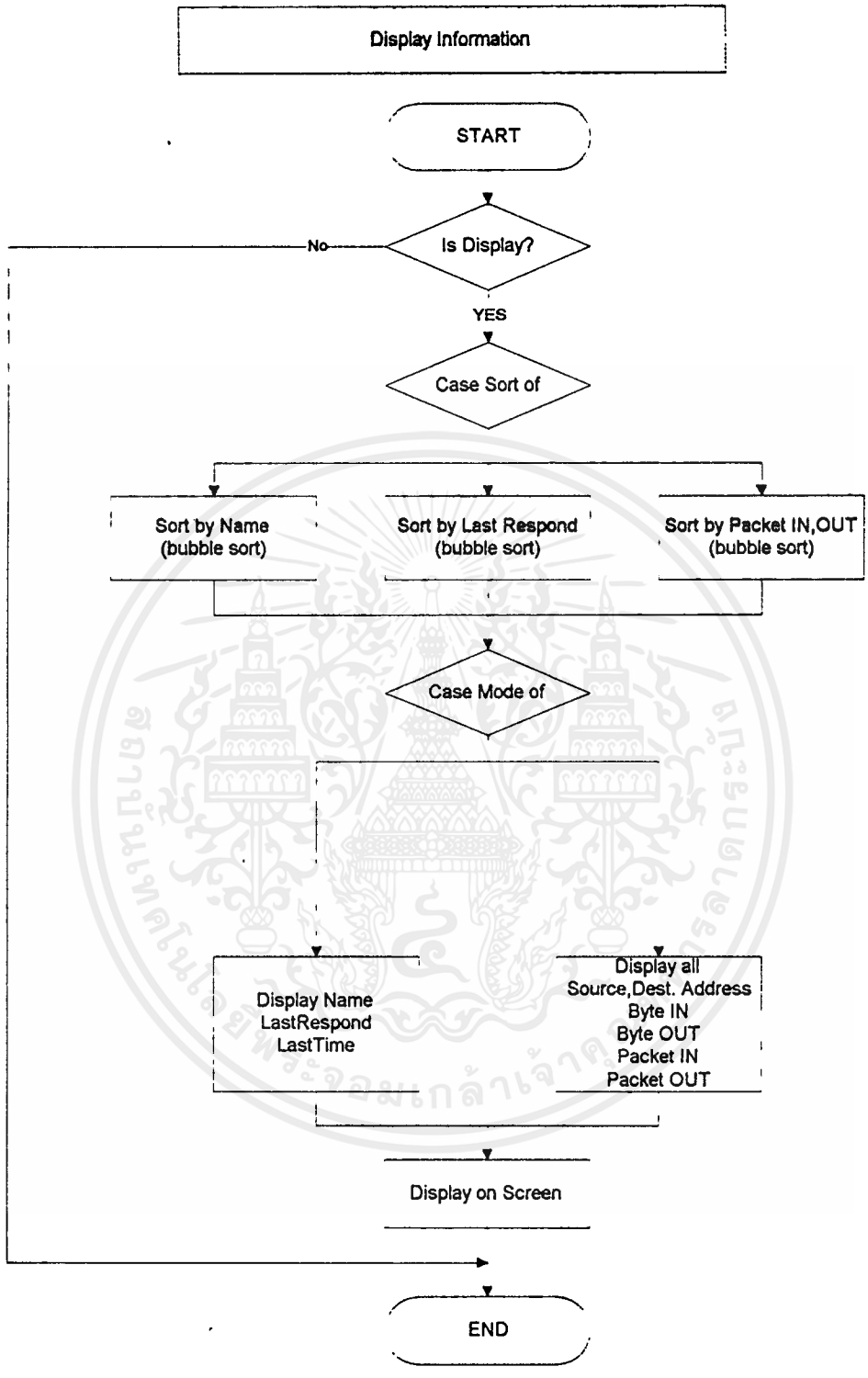
รูปที่ 3.5.12 ขั้นตอนการทำงานของโมดูลค้นหาโหนดส่วนรับคีย์



รูปที่ 3.13 ขั้นตอนการทำงานของโมดูลค้นหาโหนดส่วนรับแพ็กเกจ



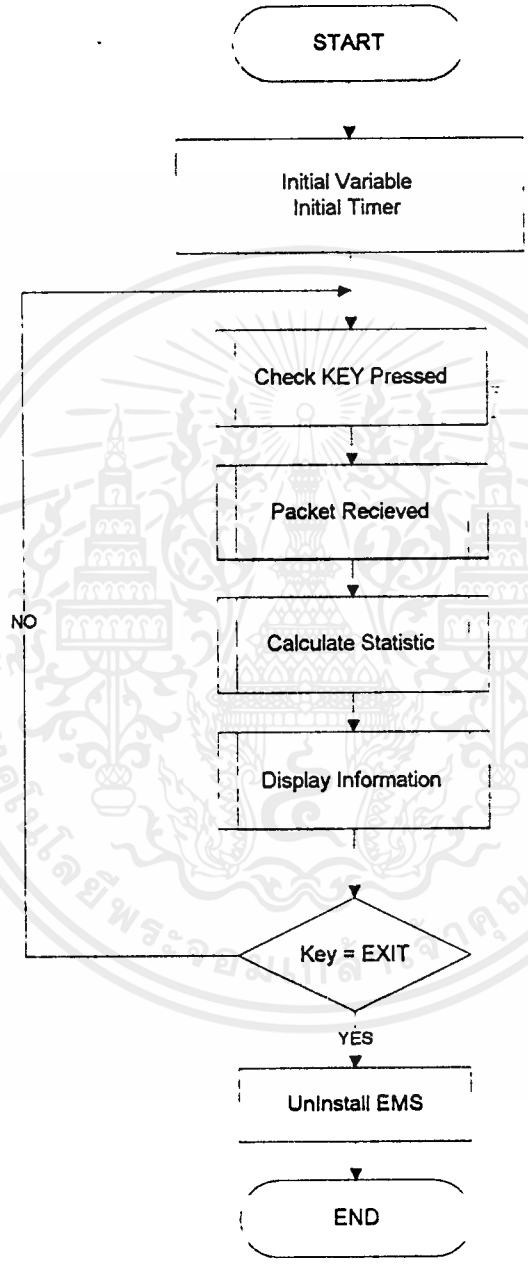
รูปที่ 3.14 ขั้นตอนการทำงานของโมดูลค้นหาโหนดส่วนคำนวณและค้นหาโหนด



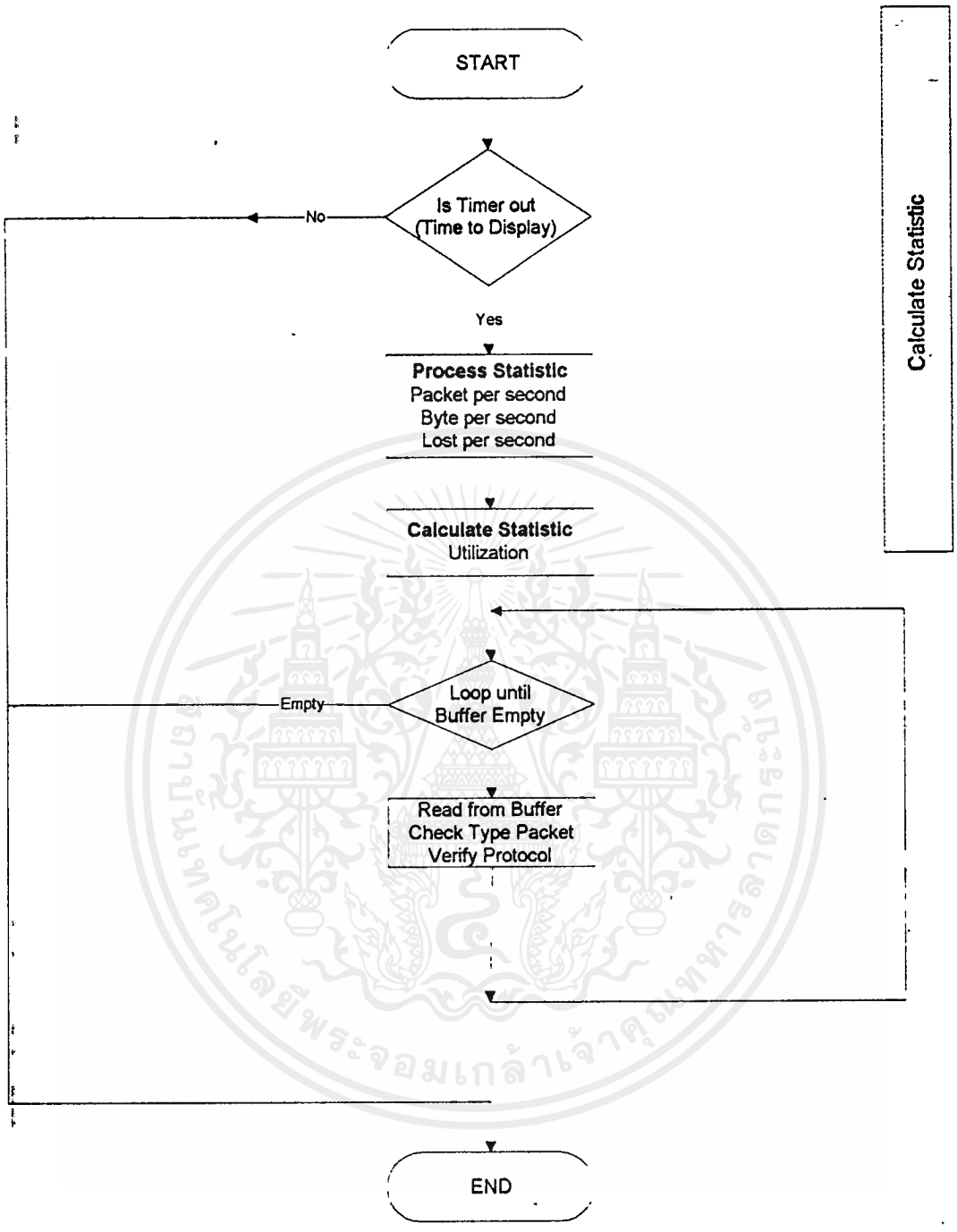
รูปที่ 3.15 ขั้นตอนการทำงานของโมดูลค้นหาโหนดส่วนแสดงผล

3.5.4 โมดูลคำนวณสถิติ

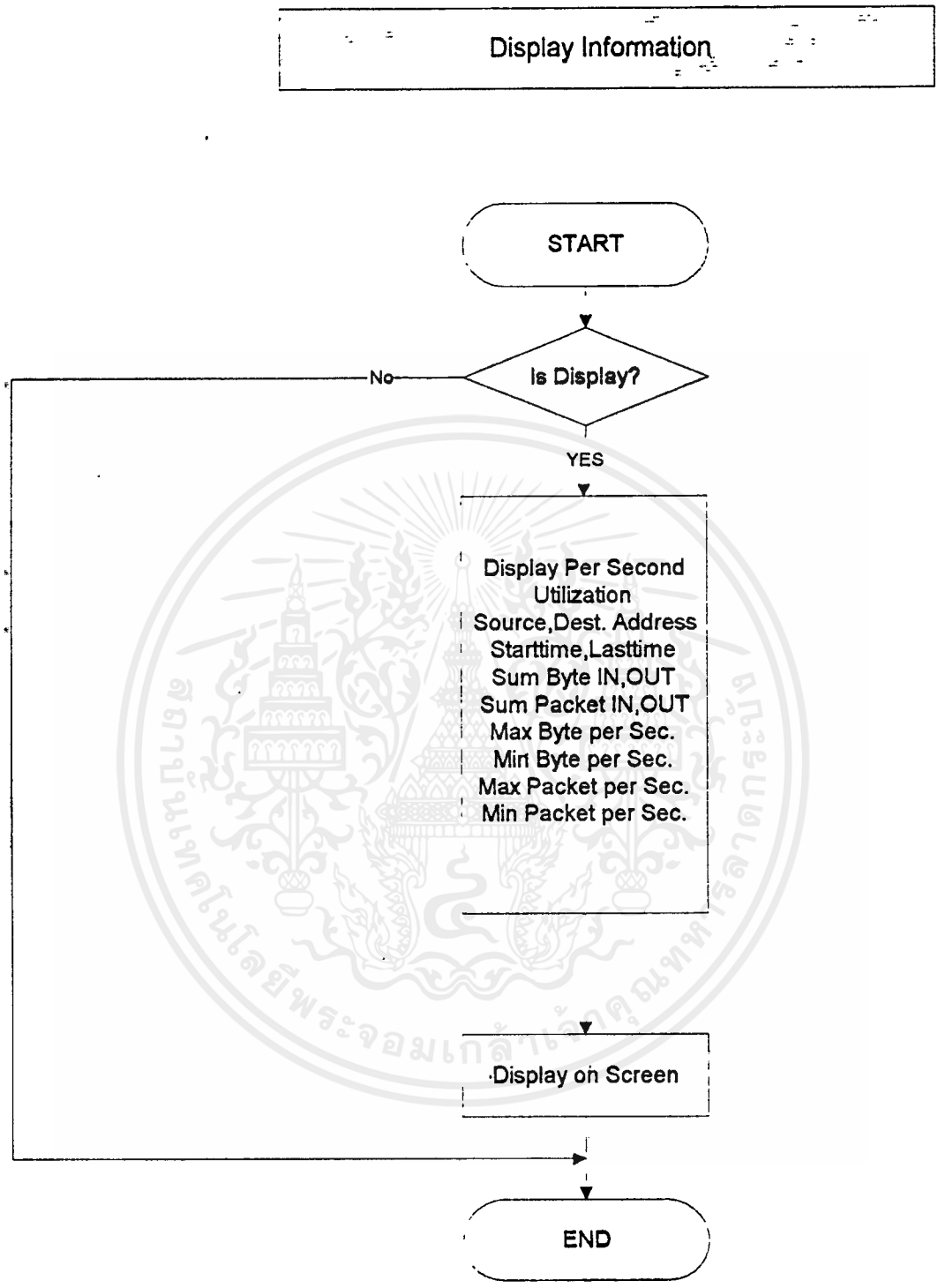
รับข้อมูลจากเน็ตเวิร์คแล้วแสดงผล โดยแสดงถึงการใช้งานของเครือข่าย ปริมาณจำนวนแพ็กเกจที่วิ่งและปริมาณข้อมูลที่ใช้อยู่



รูปที่ 3.16 ขั้นตอนการทำงานของโมดูลคำนวณสถิติ



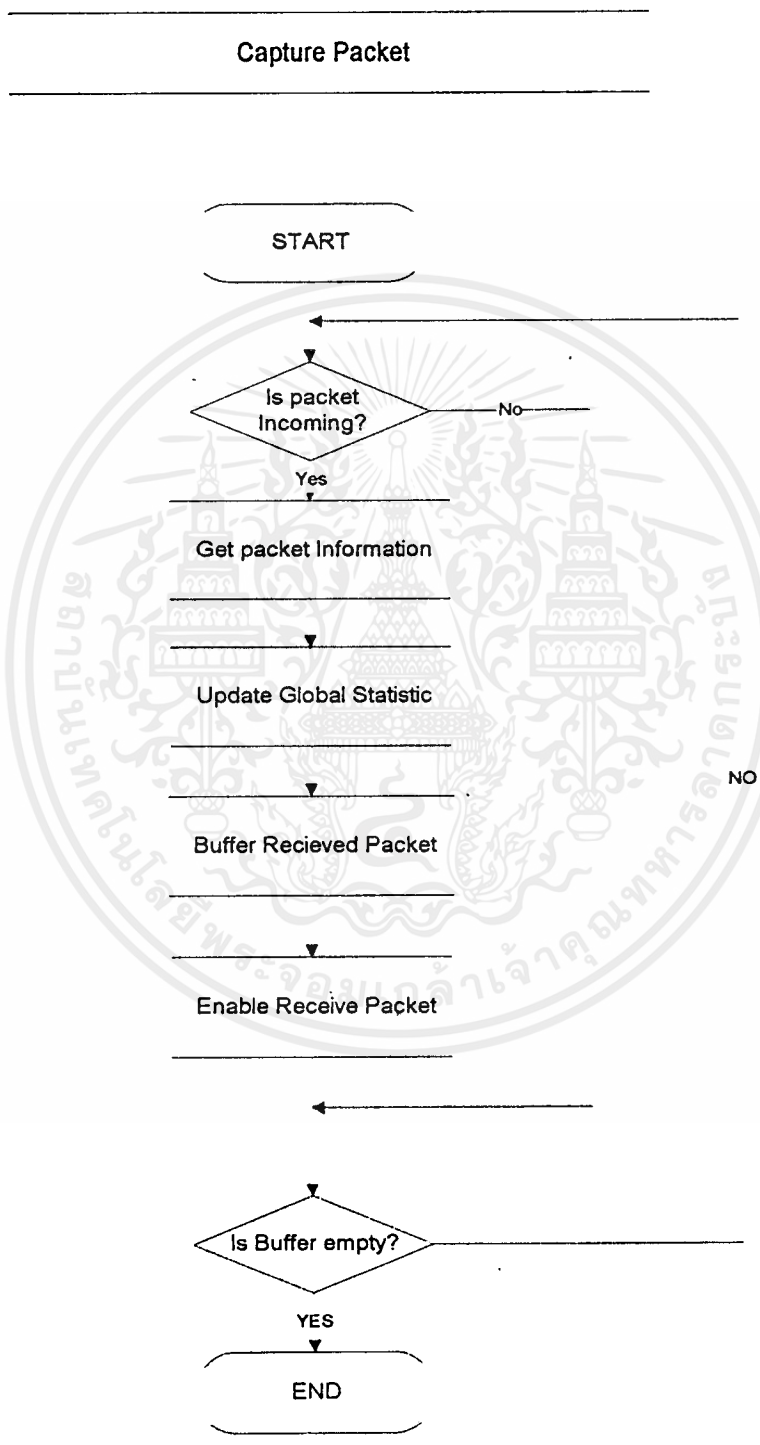
รูปที่ 3.17 ขั้นตอนการทำงานของโมดูลคำนวณสถิติส่วนคำนวณ



รูปที่ 3.18 ขั้นตอนการทำงานของโมดูลคำนวณสถิติส่วนแสดงข้อมูล

3.5.5 โมดูลเก็บแพ็กเกจ

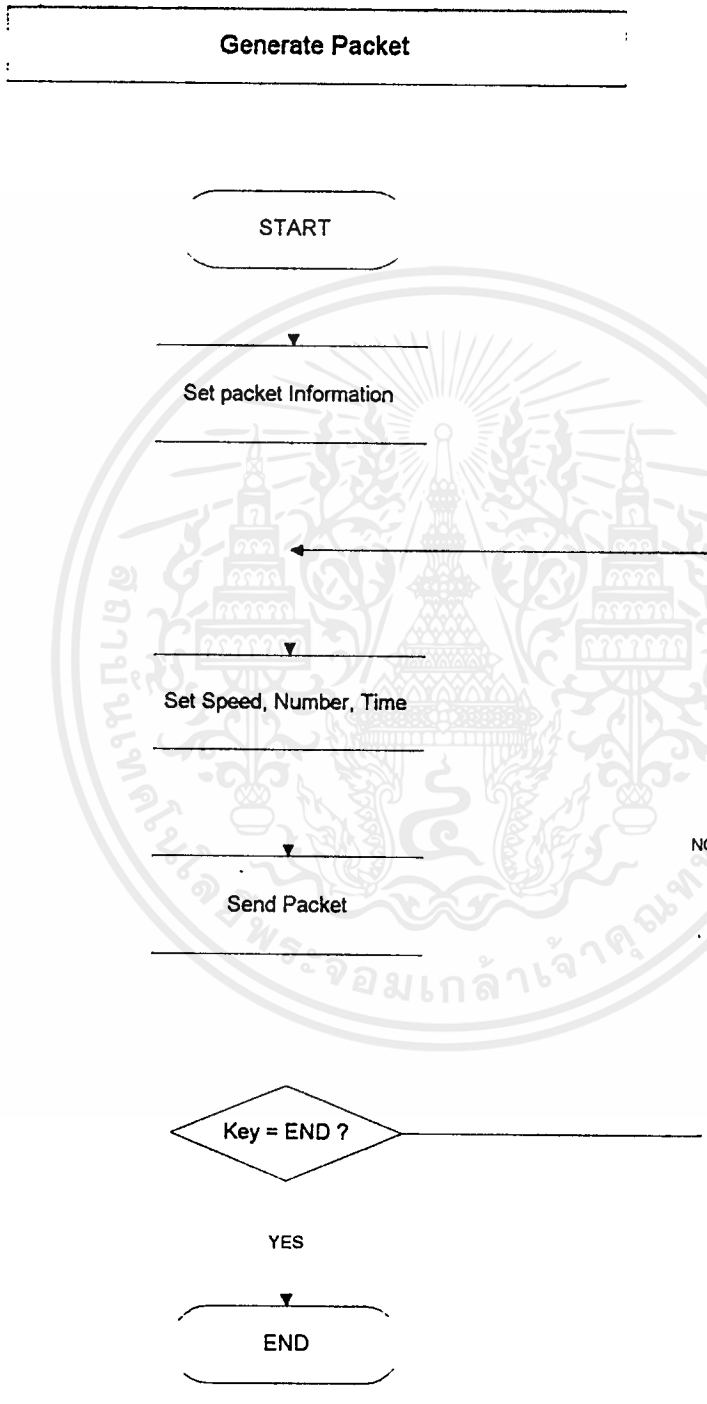
ส่วนที่เก็บข้อมูลของแพ็กเกจที่วิ่งในเน็ตเวิร์คโดยเก็บข้อมูลด้วยกันอยู่ 2 รูปแบบ คือเก็บลงในหน่วยความจำ หรือ ฮาร์ดดิสก์



รูปที่ 3.19 ขั้นตอนการทำงานของโมดูลเก็บแพ็กเกจ

3.5.6 โมดูลสร้างแพ็กเกจ

ในส่วนนี้จะสร้างแพ็กเกจตามรูปแบบและขนาดที่กำหนด โดยส่งข้อมูลไปในเครือข่ายเพื่อทดสอบระบบ สามารถเลือกส่งข้อมูลได้โดยกำหนดจำนวนแพ็กเกจ เวลา และความเร็วที่ใช้ในการส่ง

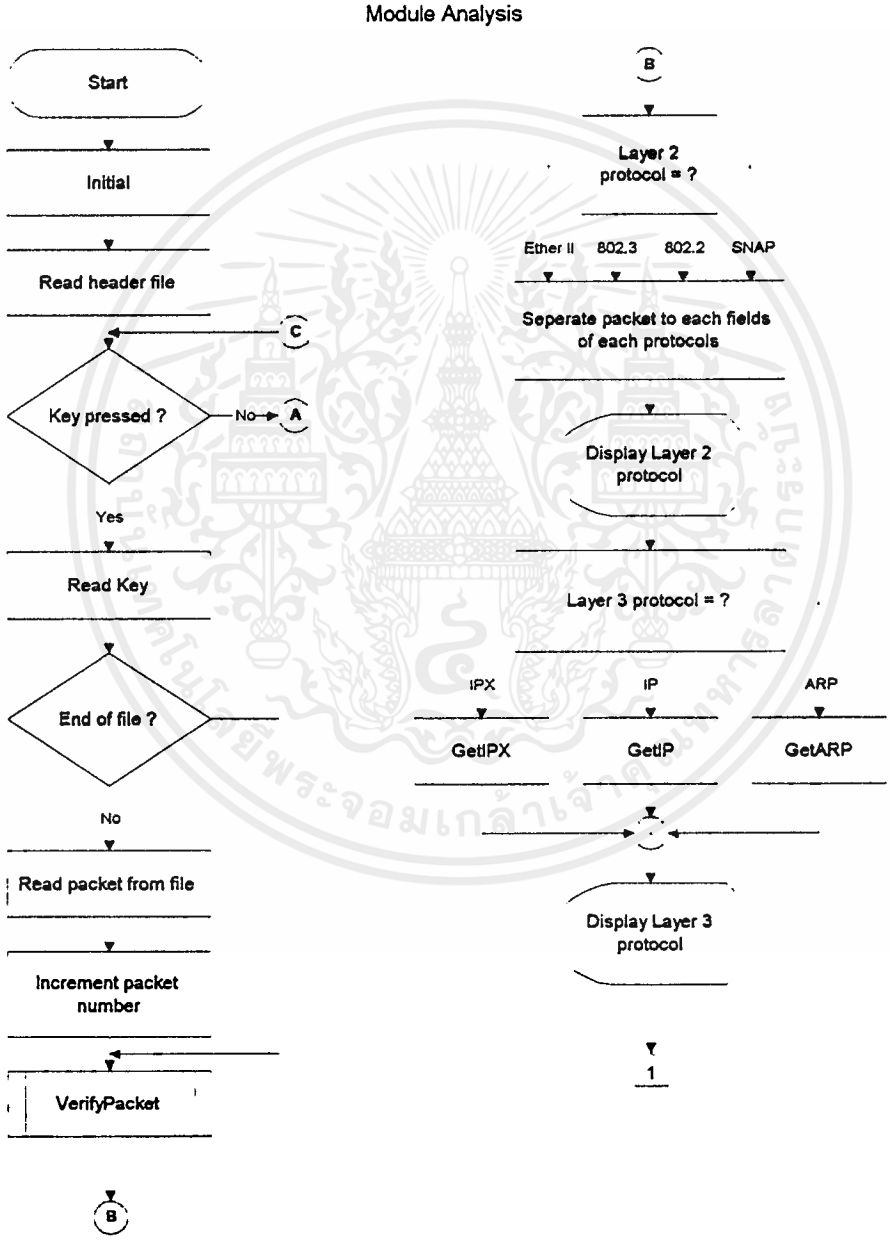


รูปที่ 3.20 ขั้นตอนการทำงานของโมดูลสร้างแพ็กเกจ

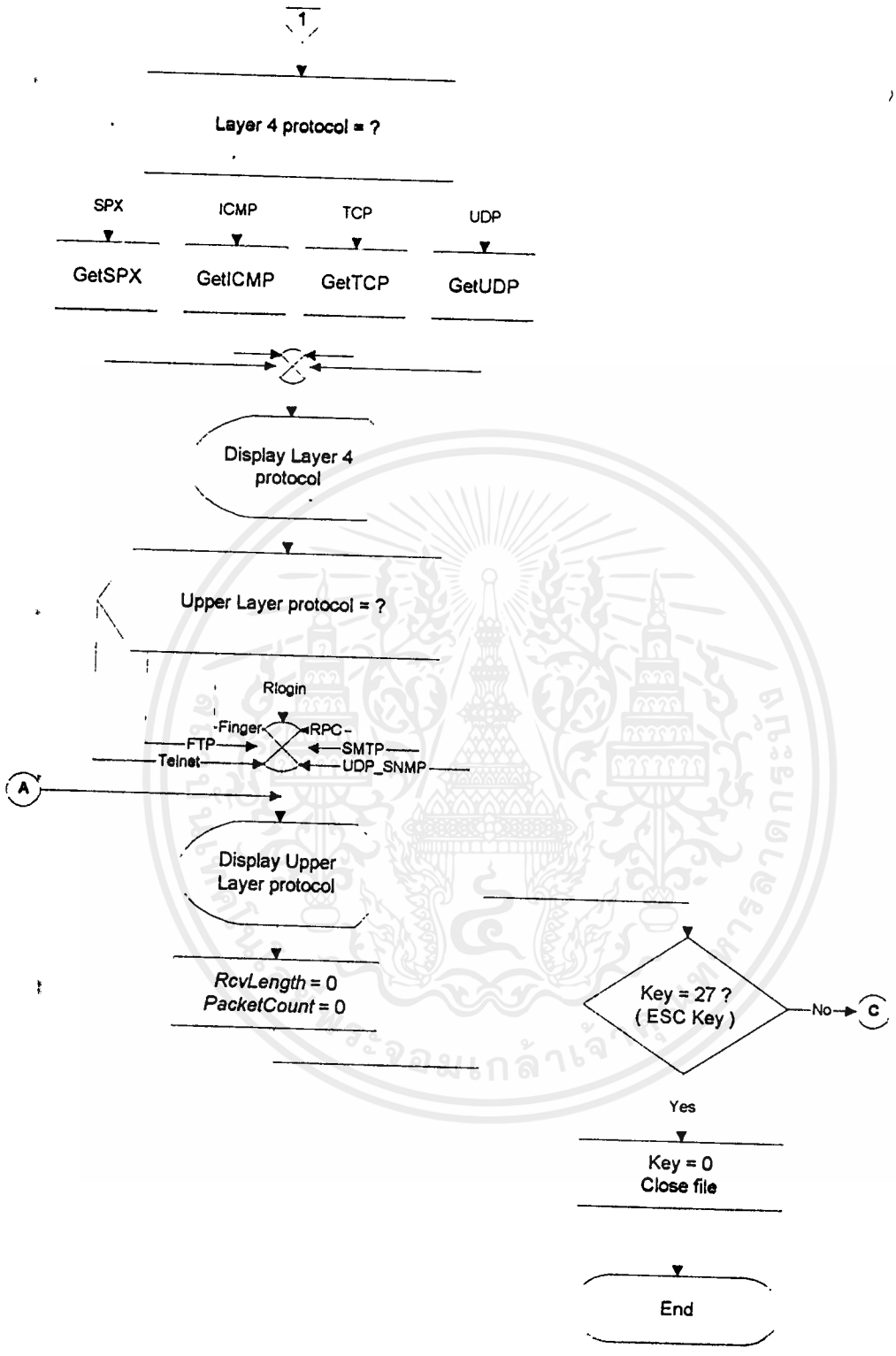
3.5.7 โมดูลวิเคราะห์แพ็กเกจ

ในส่วนของโมดูลนี้มีหน้าที่การทำงานในส่วนต่าง ๆ เกี่ยวกับแพ็กเกจเพื่อใช้ในการวิเคราะห์ถึงข้อมูลต่าง ๆ ได้แบ่งเป็นโมดูลย่อยๆได้ดังนี้

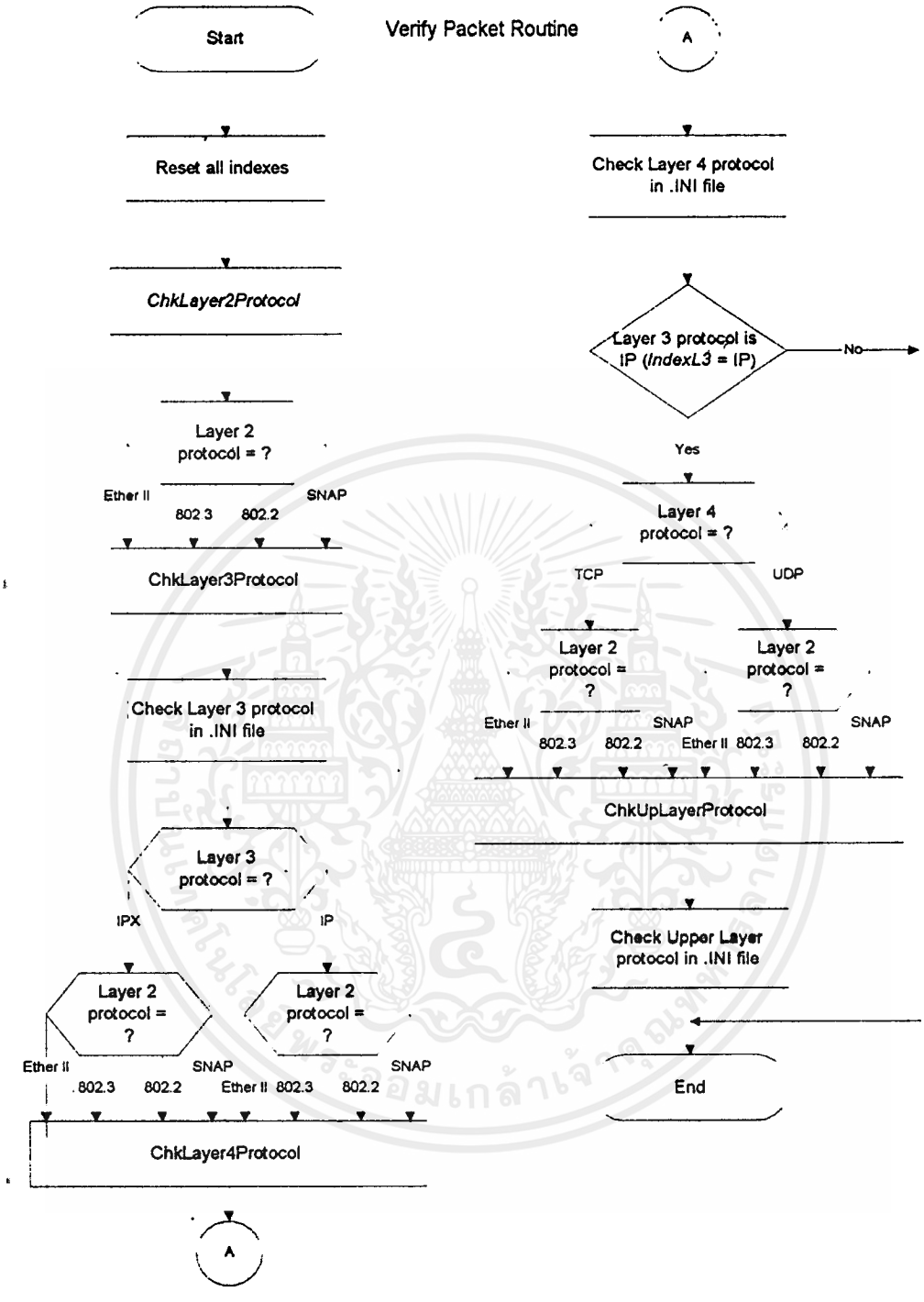
- วิเคราะห์โปรโตคอล
- วิเคราะห์ขนาดเฟรม



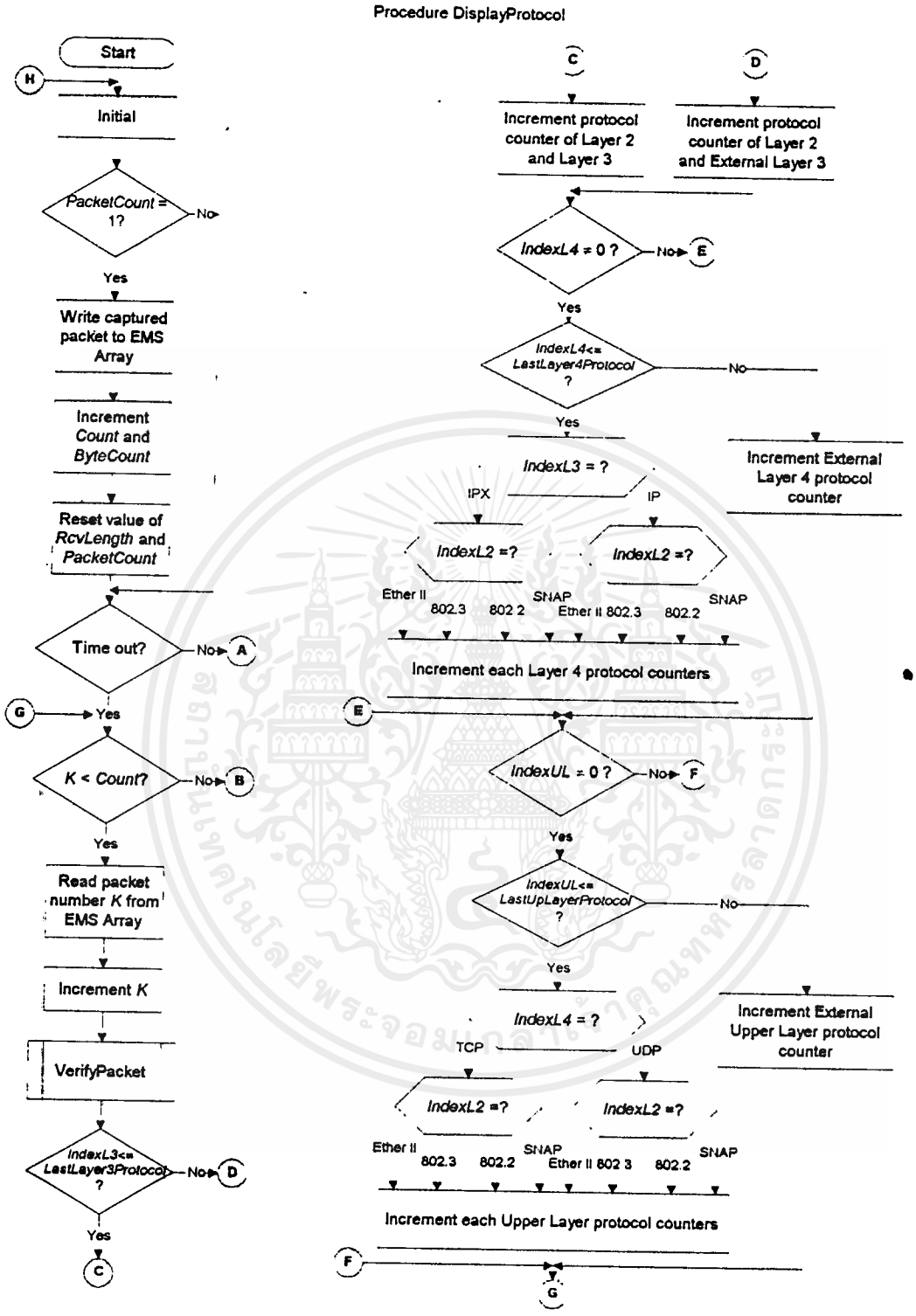
รูปที่ 3.21 ขั้นตอนการทำงานของโมดูลวิเคราะห์แพ็กเกจส่วนที่ 1



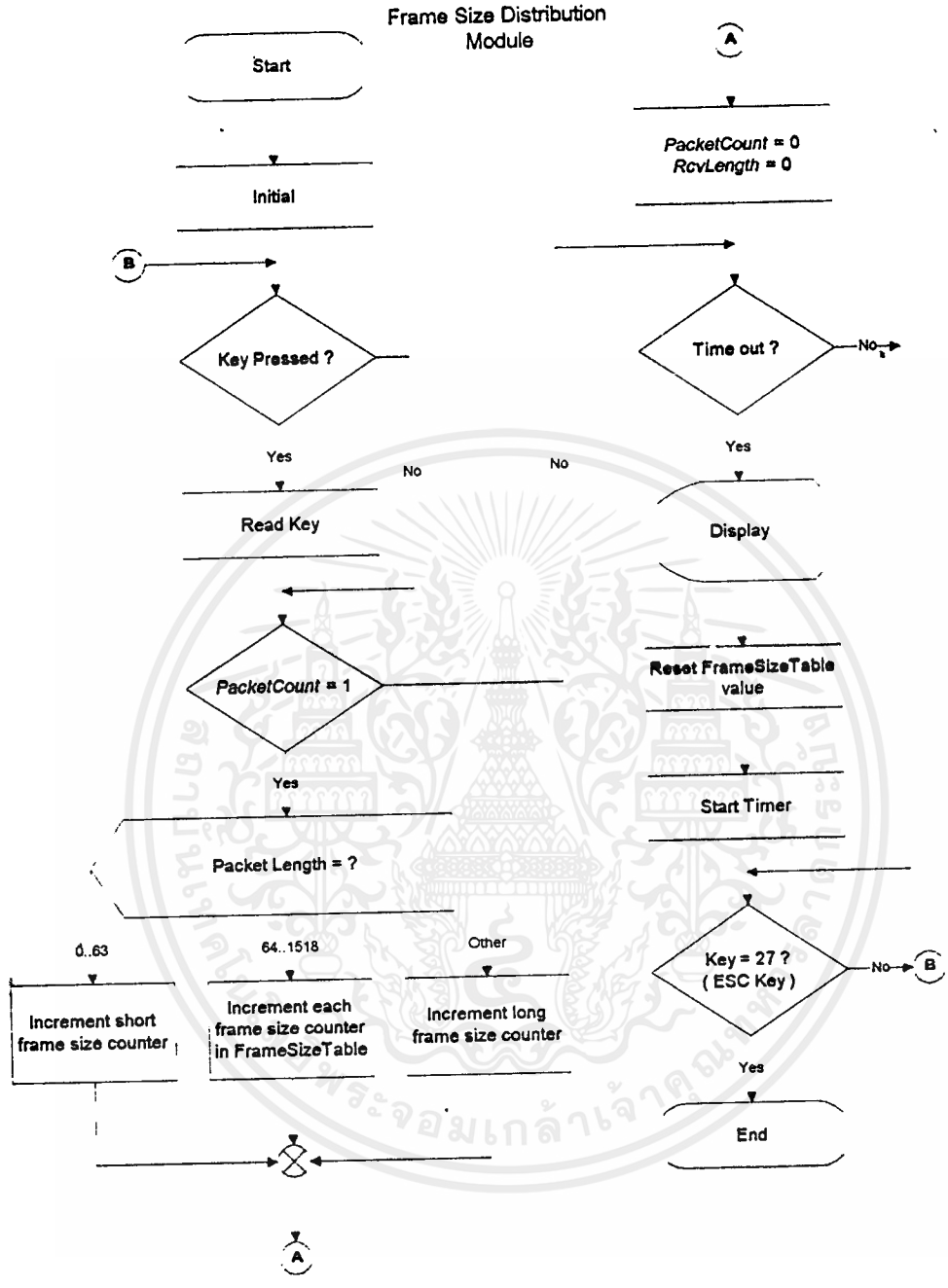
รูปที่ 3.22 ขั้นตอนการทำงานของโมดูลวิเคราะห์แพ็กเกจส่วนที่ 2



รูปที่ 3.23 ขั้นตอนการทำงานของโมดูลตรวจสอบแพ็กเกจ



รูปที่ 3.24 ขั้นตอนการทำงานของโมดูลสร้างแพ็กเกจส่วนแสดงผล



รูปที่ 3.25 ขั้นตอนการทำงานของโมดูลวิเคราะห์แพ็กเกจส่วนแสดงการกระจายขนาดเฟรม

บทที่ 4

การทดลองและผลการทดลอง

4.1 การดำเนินงานในภาคเรียนที่ 1/2539

รายละเอียดและขั้นตอนการดำเนินงานมีดังนี้

- ศึกษาโครงสร้าง เฟรมชนิดอีเทอร์เนต และ โปรโตคอลต่างๆ
- ศึกษาการทำงาน การ์ดเชื่อมต่อเน็ตเวิร์ค และ แพ็กเกจไดร์วเวอร์
- ศึกษาการโครงสร้าง ทีซีพี/ไอพี
- ศึกษาการทำงาน และ วิธีการของโปรแกรมตรวจสอบเน็ตเวิร์คอื่นๆ
- พัฒนาโปรแกรมเบื้องต้น
- ศึกษาวิธีการดักจับข้อมูล,เก็บข้อมูล,วิเคราะห์ข้อมูล
- ออกแบบระบบ
- จัดทำเอกสาร

4.2 ผลการดำเนินงาน

ศึกษาขั้นตอนการทำงานของโปรแกรมต่างๆที่เกี่ยวข้องกับการดักจับข้อมูล ซึ่งได้เลือกวิธีการแบบ เฝ้าดูภายนอก นั้นคือดักจับข้อมูลที่วิ่งผ่าน การ์ดเชื่อมต่อเครือข่าย โดยตรง ซึ่งยังต้องตรวจสอบถึงความสามารถของ ฮาร์ดแวร์ และ ซอฟแวร์ ว่าสามารถดักจับโคดได้ทันหรือไม่ ขั้นต่อมาได้ศึกษาถึงรูปแบบเฟรมต่างๆที่สำคัญได้แก่ ไอพี,เออาร์พี,ไอซีเอ็มพี,ยูดีพี เป็นต้น ได้ลองเขียนโปรแกรมเพื่อติดต่อกับ การ์ดเชื่อมต่อเครือข่ายเพื่อดึงข้อมูลจากเน็ตเวิร์ค

4.3 การดำเนินงานในภาคเรียนที่ 2/2539

รายละเอียดและขั้นตอนการดำเนินงานมีดังนี้

- ออกแบบซอฟต์แวร์
- พัฒนาส่วนการดักจับข้อมูล
- พัฒนาส่วนกรองโคด
- พัฒนาส่วนวิเคราะห์
- พัฒนาส่วนแสดงผล
- รวมระบบพร้อมกับทดสอบการใช้งาน
- จัดทำคู่มือการใช้งาน
- ใช้งานและดูแล

4.3.1 การออกแบบซอฟต์แวร์

แบ่งเป็น

- **ออกแบบโครงสร้าง** จะทำการออกแบบโครงสร้างของซอฟแวร์ ว่าประกอบด้วยส่วนใดบ้าง แต่ละส่วนแบ่งเป็น โมดูลย่อย ๆ ใดบ้าง และสร้างโครงสร้างโมดูลต่าง ๆ ขึ้นเพื่อช่วยให้การทำความเข้าใจซอฟแวร์
- **ออกแบบรายละเอียด** จะแบ่งซอฟแวร์ออกเป็นโมดูลย่อย ๆ และทำการออกแบบแต่ละ โมดูลตามหลัก อ็อบเจคโดยออกแบบ บริการ ต่าง ๆ รวมถึงการอินเทอร์เฟสระหว่างโมดูลด้วย และออกแบบอัลกอริทึม ที่ดีที่สุดแล้ว ออกแบบลงในรายละเอียดของแต่ละโมดูล
- **ออกแบบส่วนติดต่อผู้ใช้** ออกแบบรูปแบบหน้าจอ เพื่ออินเทอร์เฟส กับผู้ใช้โดยพยายามให้ ใช้งานง่ายโดยคำนึงถึงความสะดวกและง่ายต่อการใช้งาน

4.3.2 การพัฒนาและทดสอบส่วนย่อย

เขียนโปรแกรม ที่ได้ทำการออกแบบไว้ และทำการทดสอบว่าโปรแกรมที่ได้สามารถทำงานได้หรือไม่และถูกต้องตามข้อกำหนดที่กำหนดไว้หรือไม่

4.3.3 รวบรวมและทดสอบระบบ

นำโมดูลและส่วนประกอบต่าง ๆ ที่ได้ทำการเขียนและ ทดสอบว่าถูกต้องมารวมกัน เป็นโปรแกรมเดียวและทำการทดสอบโปรแกรมนั้น กับระบบจริง

4.3.4 ทำคู่มือประกอบการใช้งาน

บันทึกรายละเอียดและขั้นตอนการติดตั้งและใช้งานโปรแกรมที่สร้างขึ้นเพื่อเป็น อ้างอิงในการใช้งาน

4.3.5 ใช้งานและดูแล

เมื่อโปรแกรมทำงานได้อย่างถูกต้องแล้ว ก็จะถูกนำไปติดตั้งใช้งานจริง และในอนาคตหากมีการเปลี่ยนแปลงความต้องการบางอย่างก็จะต้องทำการแก้ไขพัฒนาโปรแกรม ให้ได้ตามต้องการ

4.4 ปัญหาและอุปสรรคที่พบในขณะปฏิบัติงาน

1. ขาดแคลนเครื่องมือและอุปกรณ์ที่ใช้ในการทำโปรเจ็คเนื่องจากการพัฒนาในช่วงแรกนั้นทำได้ลำบากต้องทดลองกับเครือข่ายจริงๆซึ่งแตกต่างจากการทดลองเขียนโปรแกรมทั่วไปที่สามารถเขียนได้โดยไม่ต้องใช้ระบบเครือข่าย
2. ขาดประสบการณ์ในการเขียนโปรแกรม ใหญ่ๆ ที่ถูกนำไปใช้งานจริง
3. มีปัญหาเรื่องการประมวลผลข้อมูลที่ไม่ทันกับแพ็กเกจที่เข้ามาได้ทำให้คุณภาพของข้อมูลจริงๆที่ควรจะได้ซึ่งในทางปฏิบัติจริงๆแล้วจะใช้ฮาร์ดแวร์เพื่อวิเคราะห์แทน

4. ขาดประสบการณ์ในการทำส่วนติดต่อแบบกราฟฟิก จึงทำให้เสียเวลามากกว่าการ เขียนโปรแกรมอย่างอื่น

บทที่ 5

บทวิจารณ์และสรุป

5.1 บทสรุปและวิจารณ์

โครงงานที่จัดทำขึ้นนี้เป็นโครงงานสำหรับช่วยในการศึกษาการทำงานภายในเครือข่ายเพื่อให้เข้าใจถึงการทำงานของโปรโตคอลในแต่ละเลเยอร์ และ เฝ้าดูการใช้งานในระบบเครือข่ายเพื่อประกอบกับการปรุงปรุงและพัฒนาระบบเครือข่าย โดยที่สามารถทราบถึงปริมาณการใช้งาน และ ข้อบกพร่องที่เกิดในเครือข่าย เนื่องจากในปัจจุบันการใช้งานคอมพิวเตอร์ (Computer) ในรูปแบบเครือข่าย (Network) ได้เข้ามามีบทบาทเพื่อช่วยเพิ่มประสิทธิภาพในการทำงานมากยิ่งขึ้นโดยเฉพาะองค์กรใหญ่ ๆ จะใช้ประโยชน์จากการใช้ทรัพยากรร่วมกันหรือการทำงานเป็นระบบเครือข่าย จึงได้พัฒนาโปรแกรม (Program) ขึ้นเพื่อใช้เฝ้าดูและวิเคราะห์แพ็กเกจ (Packet) ในระบบเครือข่ายท้องถิ่น (LAN:Local Area Network) โดยตรวจสอบแพ็กเกจที่ได้รับจากสื่อระบบ ผ่านการทำงานของแพ็กเกจไดรเวอร์ (Packet Driver) ซึ่งจะรับแพ็กเกจจากการ์ดเชื่อมต่อเน็ตเวิร์ค (NIC:Network Interface Card) เข้ามาวิเคราะห์ถึงส่วนต่างๆที่จำเป็นต่อการศึกษาการทำงาน และ ตรวจสอบระบบเครือข่าย โดยวิเคราะห์ถึง ปริมาณการใช้งาน (Utilization) คอนเวอร์เซชั่น (Conversation) โปรโตคอลดิสตริบิวชั่น (Protocol Distribution) เฟรมไซส์ดิสตริบิวชั่น(Framesize Distribution) และทำการวิเคราะห์โปรโตคอล (Protocol Analyser) แต่ในทางกลับกันอาจเป็นผลร้ายแก่บุคคลที่ต้องการเข้ามาเจาะทำลายระบบเนื่องจากข้อเสียเปรียบของระบบอีเทอร์เนตที่ข้อมูลไม่มีการเข้ารหัสไว้ดังนั้นบุคคลใดก็ตามก็สามารถดูข้อมูลที่ได้รับเข้ามาซึ่งสามารถมีสิทธิ์ใช้งานและรหัสผ่านได้

โครงงานนี้ได้พัฒนามาจากโครงงานของปีที่แล้วโดยเพิ่มส่วนต่างๆจากเดิม และ แก้ไขการทำงานบางส่วนที่ไม่สมบูรณ์ สามารถใช้งานได้ตามวัตถุประสงค์ที่ต้องการ และได้แก้ไขบางส่วนของแพ็กเกจไดรเวอร์เพื่อให้สมบูรณ์ในการใช้งาน

ซึ่งมีรายละการพัฒนาโปรเจ็คดังนี้

- ส่วนติดต่อผู้ใช้
- วิเคราะห์โปรโตคอล
- กรองแพ็กเกจ
- การกระจายของโปรโตคอล และ ขนาดของเฟรม
- การสนทนาระหว่างโฮส
- วิเคราะห์สถิติ

5.2 แนวทางการพัฒนาต่อ

1. สามารถวิเคราะห์โปรโตคอลในแต่ละเลเยอร์ได้มากขึ้นโดยเฉพาะอย่างยิ่งโปรโตคอลที่ใช้ในเน็ตแวร์แลน ทำให้สามารถ ดูถึงการใช้งานต่างๆ ของเครือข่ายได้มาก เช่น ปริมาณการใช้งานของไฟล์เซิฟเวอร์ , การใช้งานของพริ้นเตอร์ เพื่อนำไปใช้ในการทำบาลานซ์เซิฟเวอร์โหลด (balancing server load)
2. สามารถดักจับแพ็กเกจได้มีความถูกต้องมากยิ่งขึ้น

3. สามารถพัฒนาไปใช้ในระบบวินโดว์ได้โดยผ่านวินซอค (winsock) ของวินโดว์
4. สามารถใช้งานกับระบบเครือข่ายชนิดอื่นได้เช่น โทเค็นริงเป็นต้น
5. สามารถทำการดึงข้อมูลจากเครือข่ายอื่นๆได้โดยเรียกเอเจ้นท์ไว้

ภาคผนวก ก

คำศัพท์ คำแปล และความหมาย

คำศัพท์	คำแปล	ความหมาย
access type		ประเภทการใช้งาน
ACK	เอซีเค	บิตตอบรับใช้ในควบคุมทีซีพี
acknowledgement		การตอบรับ
address	แอดเดรส	ที่อยู่ ตำแหน่ง
ARP	เออาร์พี	โปรโตคอลระดับเน็ตเวิร์คเลเยอร์ใช้ในการตรวจสอบว่าโฮสยังอยู่หรือไม่
back-to-back receives	แบ็คทูแบ็ครีซีฟ	การรับในระดับล่างด้วยกัน
Bandwidth	แบนด์วิท	ความกว้างของช่องสัญญาณที่จะสื่อสารได้
Banyan's	ไบยัน	เครือข่ายชนิดหนึ่ง
Basic Function		ฟังก์ชันพื้นฐาน
bit	บิต	หน่วยข้อมูลที่เล็กที่สุด
bottlenecks	คอขวด	มีการใช้สื่อในการส่งข้อมูลมากไม่เพียงพอกับความต้องการของเครือข่าย
bridges	บริดจ์	อุปกรณ์ที่ใช้เชื่อมเน็ตเวิร์คสองวงเข้าด้วยกันทำงานในระดับดาต้าลิงค์เลเยอร์
broadcast	บอร์ดคาสท์	การกระจายข้อมูลไปทั่ว
broadcast medium	บรอดคาสท์มีเดียม	สื่อที่เป็นลักษณะที่ใช้ในการกระจายข้อมูล
buffer	บัฟเฟอร์	ข้อมูลสำรอง
checksum	เชคซัม	ค่าตรวจสอบ
class	คลาส	กลุ่มของชนิด
Client Server	ไคลน์เซิฟเวอร์	ระบบการให้บริการและรับบริการข้อมูล
Collecting		ส่วนรวบรวมข้อมูล

Collision	คอลลิสชัน	การชนกัน
collision rate		อัตราการชนกัน
communication service	คอมมิวนิเคชั่นเซอร์วิส	การบริการการสื่อสาร
connecter	คอนเน็คเตอร์	ตัวเชื่อมต่อ
connectonless	คอนเน็คชั่นเลส	การเชื่อมต่อแบบไม่ตรวจสอบว่าส่งข้อมูลถึงหรือไม่
control bit	คอนโทรลบิต	บิตควบคุม
corruption	คอรัปชั่น	เสียหาย
data offset	ดาต้าออฟเซต	ตำแหน่งข้อมูล
data throughput	ดาต้าทรูพุต	การไหลของข้อมูล
database	ดาตาเบส	ฐานข้อมูล
datagram	ดาต้าแกรม	หน่วยข้อมูลที่ใช้เรียกในเลเยอร์ดาต้าลิงค์
Datalink Layer	ด้าต้าลิงค์เลเยอร์	ระดับชั้นเชื่อมต่อข้อมูล เป็นเลเยอร์ที่ 2 ของโอเอสไอโมเดล
DECNET	เดคเน็ต	เครือข่ายชนิดหนึ่ง
demultiplexing	ดีมัลติเพลกซิง	แยกส่วน
destination	เดสติเนชัน	ปลายทาง
destination port		พอร์ตปลายทาง
dianostic	ไดอะนอสติค	ทดสอบ
DIX:DEC /Intel /Xerox	ดีไอเอ็กซ์	องค์กรที่คิดค้นอีเทอร์เนต
DOS	ดอส	ระบบปฏิบัติการ
Driver parameter	ไดร์วเวอร์พารามิเตอร์	ค่าที่ส่งแก่ไดร์วเวอร์
electronics mail	อิเลคโทรนิคส์เมล์	จดหมายในระบบเครือข่าย
End access		สิ้นสุดการใช้งาน
end of list		สิ้นสุดรายชื่อ
End Systems		ระบบปลายทาง
Error codes		รหัสผิดพลาด
Error Rate		อัตราการเกิดข้อผิดพลาด
Ethernet address	อีเทอร์เน็ตแอดเดรส	หมายเลขที่ระบุถึงสเตชันในเลเยอร์ดาต้าลิงค์

Ethernet Frame	อีเทอร์เนตเฟรม	โปรโตคอลในระดับดาต้าลิงค์
Ethernet II	อีเทอร์เนต II	โปรโตคอลในระดับดาต้าลิงค์
Ethernet SNAP	อีเทอร์เนตสแนบ	โปรโตคอลในระดับดาต้าลิงค์
ethertype values		ค่าที่บ่งบอกถึงโปรโตคอลในเลเยอร์ถัดไปของอีเทอร์เนตเฟรม
event counters		ตัวนับเหตุการณ์
executable code		โคดทำงาน
Extended Function	ฟังก์ชันเพิ่มเติม	เป็นฟังก์ชันเพิ่มเติมของแพ็กเก็จไดร์วเวอร์
External	เอ็กเทอร์นอล	ภายนอก
failed packet rates		อัตราแพ็กเกจเสีย
FCS:Frame Check Sequence	เอฟซีเอส	ค่าซีอาร์ซีใช้ตรวจสอบแพ็กเกจ
FDDI	เอฟดีดีไอ	ระบบสื่อสารข้อมูลด้วยใยแก้วนำแสง
FIELD	ฟิลด์	บริเวณ
file	ไฟล์	เอกสาร
File Server	ไฟล์เซิฟเวอร์	ผู้ให้บริการเอกสาร
Filter data		ส่วนกรองข้อมูลโปรโตคอล
FIN	เอฟไอเอ็น	บิตควบคุมของ ทีซีพี
finger	ฟิงเกอร์	โปรโตคอลใช้ตรวจสอบข้อมูลผู้ใช้
firmware	เฟิร์มแวร์	ข้อมูลที่เก็บอยู่ในรูปฮาร์ดแวร์
flag	แฟลก	ค่าที่ใช้กำหนดควบคุมการทำงานต่างๆ
flagment offset	แฟรกเมนออฟเซต	ตำแหน่งที่แบ่งแยก
flow control		ควบคุมการไหล
frame size distribution		การกระจายของขนาดเฟรม
FTP	เอฟทีพี	โปรโตคอลที่ใช้รับส่งข้อมูล
FTP Software	เอฟทีพีซอฟแวร์	องค์กรที่สร้างแพ็กเกจไดร์วเวอร์
FTP_DATA	เอฟทีพีดาตา	โปรโตคอลที่ใช้รับส่งข้อมูล
gateway	เกตเวย	อุปกรณ์ใช้เชื่อมระหว่างองค์กร
Generating network traffic		สร้างทราฟฟิคในเน็ตเวิร์ค

Handler	แฮดเดอร์	ตัวถือ
header	เฮดเดอร์	ส่วนหัวของแพ็กเกจ
High-performance Function	ฟังก์ชันประสิทธิภาพสูง	เป็นฟังก์ชันประสิทธิภาพสูงของแพ็กเก็จไดร์วเวอร์
Host Conversation		สนทนาระหว่างโฮส
ICMP	ไอซีเอ็มพี	โปรโตคอลระดับทรานส์ปอร์ตเลเยอร์ใช้เพื่อแจ้งข้อมูลภายในเน็ตเวิร์ค
IEEE 802.2	ไออีอีอี 802.2	มาตรฐานโปรโตคอลดาต้าลิงค์
IEEE 802.3	ไออีอีอี 802.3	มาตรฐานโปรโตคอลดาต้าลิงค์
IEEE 802.5	ไออีอีอี 802.5	มาตรฐานโปรโตคอลดาต้าลิงค์
incedence	อินซิเด้นซ์	มาตรฐานโปรโตคอลดาต้าลิงค์
Initiate access		กำหนดเริ่มต้นใช้งาน
Integrated	อินทิเกรตเด็ด	การรวบรวม
interface card	การ์ดอินเทอร์เฟส	อุปกรณ์เชื่อมต่อเครือข่าย
Interface Interrupt Vecter	อินเทอร์เฟสอินเทอร์รัพเวคเตอร์	เป็นส่วนเชื่อมการใช้งานของแพ็กเกจไดร์เวอร์โดยผ่านการทำงานของอินเทอร์รัพคอมพิวเตอร์
interrupt	อินเทอร์รัพ	ขัดจังหวะ
IP	ไอพี	โปรโตคอลในเน็ตเวิร์คเลเยอร์
IPX	ไอพีเอ็กซ์	โปรโตคอลในเน็ตเวิร์คเลเยอร์
Kilobyte per second	กิโลไบต์ต่อวินาที	อัตราปริมาณข้อมูลที่ใช้
LAN:Local Area Network	แลน	เครือข่ายท้องถิ่น
layer	เลเยอร์	ระดับชั้น
LifeNet's	ไลฟ์เน็ต	เครือข่ายชนิดหนึ่ง
load balancing	โหลดบาลานซิ่ง	การใช้งานให้สมดุล
local network headers		ส่วนหัวของเน็ตเวิร์คเลเยอร์
mainframe	เมนเฟรม	เครื่องคอมพิวเตอร์ขนาดใหญ่
maximum segment size		ขนาดสูงสุดของเซกเม้นท์
maximum throughput		การใช้งานสูงสุด
Megabit per Second	เม็กกะบิตต่อวินาที	อัตราในการส่งข้อมูลในเน็ตเวิร์ค

minicomputer	มินิคอมพิวเตอร์	เครื่องคอมพิวเตอร์ขนาดกลาง
minimal delay	มินิมั่มดีเลย์	ถ่วงเวลาน้อยที่สุด
minimum load times and levels		เวลาและระดับการใช้งานต่ำสุด
module	โมดูล	ส่วนการทำงาน
monitoring	มอนิเตอร์ริง	เฝ้าดู
monitoring agent	มอนิเตอร์ริงเอเจนท์	การทำงานเฝ้าดู
monitoring information		ข้อมูลที่ใช้เฝ้าดู
MTU:Maximum Transmission Units	เอ็มทียู	ขนาดสูงสุดที่สามารถส่งแพ็กเกจในเน็ตเวิร์ค
multicast list	มัลติคาสลิสต์	รายชื่อกลุ่มหมายเลขแอดเดรส
multiple connection	มัลติเปิลคอนเนคชัน	การเชื่อมต่อ
name server	เนมเซิฟเวอร์	ผู้ให้บริการชื่อ
NECTEC	เน็คเทค	องค์กร
netware lan	เน็ตแวร์แลน	เครือข่ายท้องถิ่นที่เป็นเน็ตแวร์
network	เน็ตเวิร์ค	เครือข่าย ประกอบด้วยสเตชันและสื่อที่ใช้ในการเชื่อมต่อ
network communication media	เน็ตเวิร์คคอมมิวนิเคชันมีเดีย	สื่อที่ใช้ในการเชื่อมต่อ
network component		ส่วนประกอบเน็ตเวิร์ค
Network Conversation	เน็ตเวิร์คคอนเวอร์เซชัน	การสนทนาภายในเน็ตเวิร์ค
network file system	เน็ตเวิร์คไฟล์ซิสเตม	ระบบเอกสารในเน็ตเวิร์ค
Network Interface card	เน็ตเวิร์คอินเทอร์เฟสการ์ด	อุปกรณ์ที่ใช้เชื่อมต่อกับเครือข่าย
network jamming	เน็ตเวิร์คแจมมิง	ข้อมูลที่ส่งไปแล้วแต่มีการชนกัน
Network Layer	เน็ตเวิร์คเลเยอร์	ระดับชั้นที่ 3 ของโอเอสไอโมเดล
network media		สื่อเน็ตเวิร์ค
Network media's standard packet type	เน็ตเวิร์คมีเดียสแตนด์ดาร์ดแพ็กเกจไทป์	
network servic	เน็ตเวิร์คเซอร์วิส	การบริการของเน็ตเวิร์ค
Network throughput	เน็ตเวิร์คทรูพุต	การใช้งานของเน็ตเวิร์ค
NIC:Network Interface Card	การ์ดเชื่อมต่อเน็ตเวิร์ค	อุปกรณ์ที่ใช้เชื่อมต่อกับเครือข่าย

NOP:No Operation	เอ็นโอพี	ไม่มีคำสั่ง
Novell's	โนเวล	องค์กร
Novell's IPX header	โนเวลไอพีเอ็กซ์เฮดเดอร์	ส่วนหัวโปรโตคอลไอพีเอ็กซ์
Novel's SPX header	โนเวลเอสพีเอ็กซ์เฮดเดอร์	ส่วนหัวโปรโตคอลเอสพีเอ็กซ์
number	นัมเบอร์	หมายเลข
One's complement	วันคอมพลีเม้นท์	การกลับบิตข้อมูลจาก 0 เป็น 1 และ 1 เป็น 0 แล้ว ลบ 1
optimum packet size		ขนาดแพ็กเกจที่เหมาะสม
option	อ๊อปชั่น	ตัวเลือก
OSI	โอเอสไอ	ระบบเชื่อมต่อแบบเปิด
overhead	โอเวอร์เฮด	ส่วนสิ้นเปลือง
packet	แพ็กเกจ	ข้อมูลที่ใช้ส่งในสื่อเครือข่าย
Packet Driver	แพ็กเกจไดร์วเวอร์	ส่วนติดต่อข้อมูลของการ์ดอินเทอร์เฟสกับโปรแกรม
Packet per second		จำนวนแพ็กเกจต่อวินาที
Packet size Distribution		การกระจายของขนาดแพ็กเกจ
Peak	พีค	สูงสุด
peak load times and levels		เวลาและระดับการใช้งานสูงสุด
physical address	ฟิซิคอลแอดเดรส	ตำแหน่งทางกายภาพ
platform	แพลตฟอร์ม	รูปแบบ
point-to-point communication link	พอย์นทูพอยต์คอมมิวนิเคชั่นลิงค์	การเชื่อมต่อระหว่างจุดหนึ่งไปยังจุดหนึ่ง
pointer	พอย์นเตอร์	ตัวชี้
port number		หมายเลขพอร์ต
predecessor	พรีดีเซสเซอร์	อันก่อน
protocol	โปรโตคอล	กฎเกณฑ์ในการเชื่อมต่อ
Protocol Distribution	โปรโตคอลดิสตริบิวชัน	การกระจายการใช้งานของโปรโตคอล
protocol header		ส่วนหัวของโปรโตคอล
protocol in use		โปรโตคอลที่ใช้งานอยู่
protocol number	โปรโตคอลนัมเบอร์	หมายเลขโปรโตคอล

Protocol Stack	โปรโตคอลสแต็ก	การใช้งานโปรโตคอลที่เรียกผ่านโปรโตคอลอื่นไปเป็นชั้นๆ
protocol use in conversation		โปรโตคอลที่ใช้ในการสนทนา
PSH	พีเอสเอช	บิตควบคุมในโปรโตคอลทีซีพี
Push Function	พุชฟังก์ชั่น	
query	คิวรี	
raw data	รอ ดาต้า	ข้อมูลดิบรวมทั้งส่วนหัวและเอฟซีเอส
real-time data analyzer	เรียลไทม์ดาต้าอานาไลเซอร์	วิเคราะห์ข้อมูลแบบทันที
real-time systems	เรียลไทม์ซิสเต็ม	ระบบตอบสนองทันที
rearranging	รีอาเร้นท์จิ้ง	จัดเตรียมใหม่
reciever mode		โหมดการรับ
Recording network errors		เก็บข้อผิดพลาดในเน็ตเวิร์ค
register	รีจิสเตอร์	หน่วยความจำของคอมพิวเตอร์ในการทำงาน
remote diagnostic capability		วิเคราะห์ในระยะไกล
remote execution		การทำงานปลายทาง
remote login	รีโมทล็อกอิน	
remote printing		การพิมพ์ปลายทาง
remote session	รีโมทเซสชั่น	การประชุมทางไกล
request	รีเควส	การร้องขอ
response	เรสปีอน	ตอบสนอง
Response time of the file server processes		เวลาตอบสนองของโปรเซสไฟล์เซิฟเวอร์
RLOGIN	อาร์ลอกอิน	โปรโตคอลในชุดทีซีพี/ไอพี
router	เราท์เตอร์	อุปกรณ์ใช้เชื่อมต่อเครือข่าย
RPC:Remote Procedure Call	รีโมทโพรซีเดอร์คอล	โปรโตคอลชนิดหนึ่ง
RST	อาร์เอสที	บิตควบคุมของทีซีพี
runts	รั้น	แพ็กเกจขนาดสั้นกว่าปกติ
Sensing	เซนซิ่ง	ส่วนรับข้อมูล
sequence	ซีเคว้น	หมายเลขลำดับ

serial line MODEM	ซีเรียลไลน์โมเด็ม	
server process	เซิฟเวอร์โปรเซส	การทำงานของระบบบริการ
Service access point field	เซอร์วิสแอคเซสพอยน์ฟิลด์	
shared states	แชร์เสตท	สถานะร่วม
sliding window	สไลด์ดิ้งวินโดว์	การใช้งานของทีซีพีโดยแบ่งการตอบรับข้อมูลเป็นช่วงๆ
source	ซอส	ต้นทาง
source port		พอร์ตต้นทาง
Specification	สเปคซิฟิเค่ชัน	ข้อกำหนด
SPX	เอสพีเอ็กซ์	โปรโตคอลระดับทรานส์สปอร์ต
Stack Protocol	สแต็คโปรโตคอล	การใช้งานของโปรโตคอลที่เรียกกันเป็นชั้นๆ
state	สเตต	สถานะ
storage media		สื่อบรรจุ
SYN	เอสวายเอ็น	บิตควบคุมของทีซีพี
Table offset	เทเบิ้ลออฟเซต	ตำแหน่งตาราง
TCP	ทีซีพี	โปรโตคอลระดับ
TCP/IP	ทีซีพี/ไอพี	กลุ่มของโปรโตคอลที่ใช้ทีซีพี และไอพี
TELNET	เทลเนต	โปรโตคอลในระดับทรานส์สปอร์ตเลเยอร์
terminal	เทอร์มินอล	คอมพิวเตอร์ปลายทาง
terminal server	เทอร์มินอลเซิฟเวอร์	บริการคอมพิวเตอร์ปลายทาง
time to live		เวลาที่ยังอยู่
total bandwidth		แบนด์วิทท้งหมด
traffic	ทราฟฟิค	การจราจร
Transmit	ทรานสมิต	การส่ง
transmit strategies		กลยุทธ์การส่ง
Transport Layer	ทรานสปอร์ตเลเยอร์เลเยอร์	ระดับชั้นที่ 4 ในโอเอสไอโมเดล
type	ไทป์	ชนิด
UDP	ยูดีพี	โปรโตคอลในระดับทรานสปอร์ตเล

		เยอร์
UDP protocol number		หมายเลขยูดีพีโปรโตคอล
unresponsive nodes		โหนดที่ไม่ตอบสนอง
Upper Layer	อัพเปอร์เลเยอร์	ระดับชั้นบน
URG	ยูอาร์จี	บิตควบคุมของทีซีพี
Urgent field	ฟิลดเออเจ้นท์	ช่วง
Urgent Pointer	เออเจ้นท์พอยน์เตอร์	ตำแหน่งของเออเจ้นท์
usable data		ข้อมูลที่ใช้จริง
usable data throughput	ยูสเซเบิ้ลดาต้าทรูพุต	การใช้งานข้อมูลที่ใช้งานจริง
Utilize	ยูติไลส์	อัตราการใช้งาน
workgroups	เวิร์คกรุป	การทำงานโดยรวมกลุ่ม
workstation	เวิร์คสเตชัน	เครื่องที่ใช้งาน
XNS	เอกซ์เอ็นเอส	โปรโตคอลชนิดหนึ่ง

ภาคผนวก ข

รูปแบบส่วนหัวของแต่ละโปรโตคอล

Ethernet II Frame	
Destination Address	6 ไบต์
Source Address	6 ไบต์
Type	2 ไบต์
Data	46 -1500 ไบต์

Ethernet 802.3 Frame	
Destination Address	6 ไบต์
Source Address	6 ไบต์
Length	2 ไบต์
Data	46 - 1500 ไบต์ (เริ่มต้นด้วย 0xFFFF)

Ethernet 802.2 Frame	
Destination Address	6 ไบต์
Source Address	6 ไบต์
Type	2 ไบต์
DSAP	1 ไบต์
SSAP	1 ไบต์
Control	1 ไบต์
Data	43 - 1497 ไบต์

Ethernet SNAP Frame	
Destination Address	6 ไบต์
Source Address	6 ไบต์
Type	2 ไบต์
DSAP	1 ไบต์
SSAP	1 ไบต์

Control	1 ไบต์
Organization Code	3 ไบต์
Ethernet Type	2 ไบต์
Data	38 - 1492 ไบต์

IPX Packet	
Check Sum	2 ไบต์
Length	2 ไบต์
Transport	1 ไบต์
Packet Type	1 ไบต์
Destination Network	4 ไบต์
Destination Host	6 ไบต์
Destination Socket	2 ไบต์
Source Network	4 ไบต์
Source Host	6 ไบต์
Source Socket	2 ไบต์
Data	

IP Packet	
Version + Header Length	1 ไบต์ (½ ไบต์ + ½ ไบต์)
Type Of Service	1 ไบต์
Length	2 ไบต์
Identifier	2 ไบต์
Flag	3 บิต
Fragment Offset	13 บิต
Time to Live	1 ไบต์
Protocol	1 ไบต์
Header Check Sum	2 ไบต์
Source Address	4 ไบต์
Destination Address	4 ไบต์
Option	Variable
Data	

ARP Packet	
Hardware Type	2 ไบต์
Protocol Type	2 ไบต์
Hardware Address Length	1 ไบต์
Protocol Address Length	1 ไบต์
Operation Code	2 ไบต์
Send Hardware Address	6 ไบต์
Send Protocol Address	4 ไบต์
Target Hardware Address	6 ไบต์
Target Protocol Address	4 ไบต์

SPX Packet	
Control	1 ไบต์
Data Type	1 ไบต์
Source ID	2 ไบต์
Destination ID	2 ไบต์
Sequence Number	2 ไบต์
Acknowledgment Number	2 ไบต์
Allocation Number	2 ไบต์
Data	

NCP Request Packet	
Request Type	2 ไบต์
Sequence Number	1 ไบต์
Connection Number Low	1 ไบต์
Task Number	1 ไบต์
Connection Number High	1 ไบต์
Data	

NCP Reply Packet	
Request Type	2 ไบต์
Sequence Number	1 ไบต์
Connection Number Low	1 ไบต์
Task Number	1 ไบต์
Connection Number High	1 ไบต์
Completion Code	1 ไบต์
Connection Status	1 ไบต์
Data	

ICMP Packet	
Type	1 ไบต์
Code	1 ไบต์
Check Sum	2 ไบต์
Data	

ICMP Message Types

ชนิด	ความหมาย
0	Echo Reply
3	Destination Unreachable
4	Source Quench
5	Redirect
8	Echo
11	Time Exceeded
12	Parameter Problem
13	Timestamp
14	Timestamp Reply
15	Information Request
16	Information Reply

Echo and Echo Reply Message	
Type	1 ไบต์ (0 or 8)
Code	1 ไบต์
Check Sum	2 ไบต์
Identifier	2 ไบต์
Sequence Number	2 ไบต์
Data	

Information and Information Reply Message	
Type	1 ไบต์ (15 or 16)
Code	1 ไบต์
Check Sum	2 ไบต์
Identifier	2 ไบต์
Sequence Number	2 ไบต์

Destination Unreachable Message + Source Quench Message + Time Exceeded Message	
Type	1 ไบต์ (3, 4 หรือ 11)
Code	1 ไบต์
Check Sum	2 ไบต์
Unused	4 ไบต์
Data	

Redirect Message	
Type	1 ไบต์ (5)
Code	1 ไบต์
Check Sum	2 ไบต์
Gateway Internet Address	4 ไบต์
Data	

Parameter Problem Message	
Type	1 ไบต์ (12)
Code	1 ไบต์
Check Sum	2 ไบต์
Pointer	1 ไบต์
Unused	3 ไบต์
Data	

Timestamp and Timestamp Reply Message	
Type	1 ไบต์ (13 or 14)
Code	1 ไบต์
Check Sum	2 ไบต์
Original Timestamp	4 ไบต์
Receive Timestamp	4 ไบต์
Transmit Timestamp	4 ไบต์

TCP Packet	
Source Port	2 ไบต์
Destination Port	2 ไบต์
Sequence Number	4 ไบต์
Acknowledgment Number	4 ไบต์
Header Length	1 ไบต์
Code Bits	1 ไบต์
Windows	2 ไบต์
Check Sum	2 ไบต์
Urgent Pointer	2 ไบต์
Option	Variable
Data	

UDP Packet	
Source Port	2 ไบต์
Destination Port	2 ไบต์
Length	2 ไบต์

Check Sum	2 ไบต์
Data	

RIP Packet	
Command	1 ไบต์
Version	1 ไบต์
Zero	2 ไบต์
Address Family ID	2 ไบต์
Zero	2 ไบต์
IP Address	4 ไบต์
Zero	8 ไบต์
Distance to network n	4 ไบต์
Can Repeat from first Address Family ID fields	

ภาคผนวก ค

การกำหนดตัวเลขในอีเทอร์เนต

ในการใช้งานของอีเทอร์เนตจะต้องมีกำหนดค่าต่างๆเหล่านี้เพื่อให้เป็นมาตรฐานและสามารถใช้งานร่วมกันได้ ซึ่งต้องมีส่วนต่างๆ ดังนี้ ไทป์โคด, เวนเดอร์โคด(vendorcodes), มัลติคาสท์ (รวมถึงบอร์ดคาสท์) แอดเดรส อีเทอร์เนตไทป์ ตำแหน่งอ๊อกเตต ที่ 13 และ 14 ของอีเทอร์เนตเป็นส่วนของ อีเทอร์ไทป์หรือ IEEE802.3 เล้นท์ ซึ่งอีเทอ์ไทป์นั้น ซีรอก (XEROX) เป็นผู้รับผิดชอบอยู่ บางค่าอาจเป็นมาตรฐานจะใส่เครื่องหมาย "+" นอกนั้นถูกกำหนดเพื่อใช้เป็นการส่วนตัว ซึ่งขอมูลนี้จะมีส่วนที่มาจาก ซีรอกพับลิกอีเทอร์เนตแพ็กเกตไทป์ (Xerox Public Ethernet Packet Type) , IEEE802.3 สแตนด์ดาร์ด , และจาก เน็ตเวิร์คเมเนจเจอร์ และ เวนเดอร์

โปรโตคอลอีเทอร์เนตไทป์ (Protocol Ethernet Type)

@	0000-05DC	IEEE802.3 Length Field (0.:1500.)
+	0101-01FF	Experimental.
	0200	Xerox PUP (conflicts with 802.3 Length Field range)
	0201	Xerox PUP Address Translation (conflicts ...)
	0400	Xixdorf (conflicts with 802.3 Length Field)
+*	0800	DOD Internet Protocol (IP)
+	0805	X.25 Level 3
+*	0806	Address Resolution Protocol (ARP) (for IP)
	8005	HP Probe protocol
+	8019	Apollo DOMAIN
+	8035	Reverse Address Resolution Protocol (RARP)
	8037	IPX (Novell Netware?)
+	809B	EtherTalk (AppleTalk over Ethernet)
	80D5	IBM SNA Services over Ethernet
+	80F3	AppleTalk Address Resolution Protocol (AARP)
+	8137	Novell (old) NetWare IPX (ECONFIG E option)
+	8138	Novell, Inc.
	814C	SNMP over Ethernet (see RFC1089)
	817D	XTP
	86DD	IP version 6
	8888	HP LanProbe test?
+	9000	Loopback (Configuration Test Protocol)
	AAAA	DECNET? Used by VAX 6220 DEBNI
%	FF00	BBN VITAL-LanBridge cache wakeups

หมายเหตุ

- ตัวเลขเป็นเลขฐาน 16
- "*" = โปรโตคอล นี้ถูกใช้ในการทำอีเทอร์เน็ตบอร์ดคาสท์ ซึ่งอาจจะใช้ในมัลติคาส์ท ก็ได้

- "%" = อาจจะนำไปใช้ส่วนตัว คือ ยังไม่ได้ลงทะเบียน

- "+" = โปรโตคอล ซึ่งถูกอ้างอิงโดยซีรอกในหนังสือ "COURIER (page 8-9) October 1988 issue of ในหัวข้อpublicly assigned numbers

- "@" = ตามที่อธิบายใน COURIER (page 8) "ถ้ามีขนาดน้อยกว่า 600H packet จะเป็นของ 802.3 และ ถ้าใหญ่กว่า 600H จะถือว่าเป็น flag และ Ethernet packet

เวนเดอร์โคด

อีเทอร์เนตฮาร์ดแวร์แอดเดรส หรือ แมคประกอบด้วย 48 บิต หรือ 12 เฮกสาเดซิมอลดิจิต (hexadecimal digits) (ตัวเลข0-9 รวมA-F) ประกอบด้วย ช่วงแรก ทางซ้าย 6 ดิจิต ซึ่งจะตรงกับหมายเลขของเวนเดอร์ และ ที่เหลืออีก 6 ดิจิต เป็นซีเรียลนัมเบอร์ (serial number) ของอินเทอร์เฟสนั้นจากเวนเดอร์

อีเทอร์เนตแอดเดรสมักจะเขียนในรูปของการแบ่ง 2 เฮกดิจิต เพื่อในการอ่านและอ้างอิงได้ง่าย เช่น. 12-34-56-78-9A-BC

เลขเหล่านี้เป็นตัวกำหนดถึงฟิซิคอลสเตชั่นแอดเดรส ที่ไม่ใช้มัลติคาส์ทหรือบอร์ดคาส์ท ดังนั้น ดิจิตที่สองจากทางซ้าย จะเป็นเลขคู่ ไม่ใช่เลขคี่

cisco = 00000C

Fujtsu = 00000E

NeXT = 00000F

Novell = 00001B

ATT = 00003D

Nokia = 00004B

NEC = 00004C

ATT = 000055

MIPS = 00006B

Sanyo = 0000A0 /* Sanyo Electronics */

Xerox = 0000AA

Apollo = 0000AC /* Apollo */

HP ON = 0000C6 /* H-P Intlgnt Networks Oper (EON) */

DEC = 0000F8 /* Digital Equipment Corporation */

IEE802 = 000143 /* IEEE 802 */

```
3com   = 0020AF
3com   = 00608C
3Com   = 00608C
CNET   = 0080AD
NET    = 0080B2
IEEE   = 0080C2 /* IEEE     802.1 Committee          */
Intel  = 00AA00
3Com   = 026060
3Com   = 02608C
Bridge = 080002
Apple  = 080007
HP     = 080009
Apollo = 08001E
Sharp  = 08001F /* Sharp                            */
Sun    = 080020
NBI    = 080022
FujiXe = 080037 /* Fuji Xerox */
Motrla = 08003E /* Motorola */
Sony   = 080046
IBM    = 08005A /* (bit-reversed from Token-Ring) */
ATT    = 08006A
Mitsu  = 080070 /* Mitsubishi */
Casio  = 080074 /* Casio                            */
SilicG = 080079 /* Silicon Graphics                 */
Xyplex = 080087
ATT    = 09006A /* AT&T Use in smart hub */
IBM    = 10005A /* (not bit-reversed from Token-Ring) */
DECnet         = 1000D4 /* DEC                      */
AppIUX = 1000E0 /* Apple A/UX      (modified addresses for licensing)  */
```

ATT = 800010 /* AT&T */

DECnet = AA0000

DECnet = AA0003

บอร์ดคาส์ทแอดเดรสและสเตชั่นแอ่ดเดรส (Broadcast Addresses & station Address)

address	*type*	*owner*
FF-FF-FF-FF-FF-FF	0600	XNS packets, Hello or gateway search?
		6 packets every 15 seconds, per XNS station
FF-FF-FF-FF-FF-FF	0800	IP (e.g. RWHOD via UDP) as needed
FF-FF-FF-FF-FF-FF	0806	ARP (for IP and CHAOS) as needed
FF-FF-FF-FF-FF-FF	1600	VALID packets, Hello or gateway search?
		1 packets every 30 seconds, per VALID station
FF-FF-FF-FF-FF-FF	8035	Reverse ARP
FF-FF-FF-FF-FF-FF	809B	EtherTalk

station 09001E000000	"Apollo_DOMAIN"
station 090007FFFFFF	"Atalk_Broadcast"
station 0180C2000000	"Bridge_Group_Addr"
station FFFFFFFFFFFF	"Broadcast"
station 09002B230000	"DEC_Argo_Console"
station 09002B010000	"DEC_Bridges"
station AB0000020000	"DEC_Console"
station 09002B000006	"DEC_Encryption"
station AB0000040000	"DEC_END_nodes"
station 09002B040000	"DEC_LAST"
station AB0003000000	"DEC_LAT"
station 09002B00000F	"DEC_LAT_Units"
station 09002B010001	"DEC_lv1_Bridges"

station AB0000030000 "DEC_lv1_Router"

station 09002B020000 "DEC_lv2_Router"

station 09002B000000 "DEC_Mumps"

station 09002B020100 "DEC_Name_Advert"

station 09002B020101 "DEC_Name_Solicit"

station 09002B000007 "DEC_Netbios"

station AB0000010000 "DEC_Pmp/Load"

station 09002B000003 "DEC_Traffic_Mon"

station 09002B000002 "DEC_VAXELN"

station C00000000100 "Ethernet Broadcast"

station 090009000004 "HP_DLC"

station 090009000001 "HP_Probe"

station 09002B000004 "ISO_END_Stns"

station 09002B000005 "ISO_Int_Stns"

station CF0000000000 "Loopback"

station 090014000101 "NCP_30_Servers"

station 030000000001 "NetBIOS"

station 090002040002 "Vtlink_Bridges"

station 09007C020005 "Vtlink_Diag"

station 09007C010001 "Vtlink_DLS"

station 09007C010004 "Vtlink_DLS/NonDLS"

station 09007C010002 "Vtlink_DLS_Hello"

station 09007C010003 "Vtlink_DLS_Inlink"

station 090002040001 "Vtlink_Printers"

station 09007C050002 "Vtlink_Validation"

station C00000000001 "Active Mon."

station C00000000100 "All Bridges"

station FFFFFFFFFFFF "All Fs Broadcast"

station 800143000000 "Bridge Group"

station C000FFFFFFFF	"Broadcast"
station C00000000010	"Config Srv"
station C00000000008	"Error Mon."
station C00000002000	"LAN Manager"
station C00000000080	"NetBIOS"
station C00000800000	"NetWare"
station C00000000002	"Param Server"

กำหนดอินเตอร์เนตโปรโตคอลนัมเบอร์

Decimal	*Keyword*	*Protocol*	*References*
0	Reserved		[JBP]
1	ICMP	Internet Control Message	[RFC792,JBP]
2	IGMP	Internet Group Management	[RFC1112,JBP]
3	GGP	Gateway-to-Gateway	[RFC823,MB]
4	IP	IP in IP (encasulation)	[JBP]
6	TCP	Transmission Control	[RFC793,JBP]
11	NVP-II	Network Voice Protocol	[RFC741,SC3]
17	UDP	User Datagram	[RFC768,JBP]
29	ISO-TP4	ISO Transport Protocol Class 4	[RFC905,RC77]
41	SIP	Simple Internet Protocol	[SXD]
55-60		Unassigned	[JBP]
61		any host internal protocol	[JBP]
62	CFTP	CFTP	[CFTP,HCF2]
63		any local network	[JBP]
68		any distributed file system	[JBP]
89	OSPFIGP	OSPFIGP	[RFC1583,JTM4]
91	LARP	Locus Address Resolution Protocol	[BXH]
92	MTP	Multicast Transport Protocol	[SXA]

94	IPIP	IP-within-IP Encapsulation Protocol	[JI6]
97	ETHERIP	Ethernet-within-IP Encapsulation	[RXH1]
98	ENCAP	Encapsulation Header	[RFC1241,RXB3]
99		any private encryption scheme	[JBP]
101-254		Unassigned	[JBP]
255		Reserved	[JBP]

คลาส และ ไทป์ ต่างๆในแพ็กเกตไดร์วเวอร์

DEC/Intel/Xerox "Bluebook" Ethernet	**Class 1**
3COM 3C500/3C501	1
3COM 3C505	2
Interlan Ni5010	3
BICC Data Networks 4110	4
BICC Data Networks 4117	5
MICOM-Interlan NP600	6
Ungermann-Bass PC-NIC	8
Univation NC-516	9
TRW PC-2000	10
Interlan Ni5210	11
3COM 3C503	12
3COM 3C523	13
Western Digital WD8003	14
Spider Systems S4	15
Torus Frame Level	16
10NET Communications	17
Gateway PC-bus	18
Gateway AT-bus	19
Gateway MCA-bus	20
IMC PCnic	21
IMC PCnic II	22

IMC PCnic 8bit	23
Tigan Communications	24
Micromatic Research	25
Clarkson "Multiplexor"	26
D-Link 8-bit	27
D-Link 16-bit	28
D-Link PS/2	29
Research Machines 8	30
Research Machines 16	31
Research Machines MCA	32
Radix Microsys. EXM1 16-bit	33
Interlan Ni9210	34
Interlan Ni6510	35
Vestra LANMASTER 16-bit	36
Vestra LANMASTER 8-bit	37
Allied Telesis PC/XT/AT	38
Allied Telesis NEC PC-98	39
Allied Telesis Fujitsu FMR	40
Ungermann-Bass NIC/PS2	41
Tiara LANCard/E AT	42
Tiara LANCard/E MC	43
Tiara LANCard/E TP	44
Spider Comm. SpiderComm8	45
Spider Comm. SpiderComm16	46
AT&T Starlan NAU	47
AT&T Starlan-10 NAU	48
AT&T Ethernet NAU	49
Intel smart card	50

ProNET-10 **Class 2**

Proteon p1300 1

Proteon p1800 2

IEEE 802.5/ProNET-4 **Class 3**

IBM Token ring adapter 1

Proteon p1340 2

Proteon p1344 3

Gateway PC-bus 4

Gateway AT-bus 5

Gateway MCA-bus 6

Omninet **Class 4**

Appletalk **Class 5**

Serial line **Class 6**

Clarkson 8250-SLIP 1

Clarkson "Multiplexor" 2

Starlan **Class 7**

ArcNet **Class 8**

Datapoint RIM 1

AX.25 **Class 9**

KISS **Class 10**

IEEE 802.3 w/802.2 hdrs **Class 11**

FDDI w/802.2 hdrs **Class 12**

Internet X.25 Class 13

Western Digital	1
Frontier Technology	2

N.T. LANSTAR (encapsulating DIX) Class 14

NT LANSTAR/8	1
NT LANSTAR/MC	2

ภาคผนวก ง

ตารางการแปลงเลขฐานสิบหกเป็นฐานสิบ

DEC	HEX	BINARY	DEC	HEX	BINARY	DEC	HEX	BINARY
0	00	0000 0000	55	37	0011 0111	110	6E	0110 1110
1	01	0000 0001	56	38	0011 1000	111	6F	0110 1111
2	02	0000 0010	57	39	0011 1001	112	70	0111 0000
3	03	0000 0011	58	3A	0011 1010	113	71	0111 0001
4	04	0000 0100	59	3B	0011 1011	114	72	0111 0010
5	05	0000 0101	60	3C	0011 1100	115	73	0111 0011
6	06	0000 0110	61	3D	0011 1101	116	74	0111 0100
7	07	0000 0111	62	3E	0011 1110	117	75	0111 0101
8	08	0000 1000	63	3F	0011 1111	118	76	0111 0110
9	09	0000 1001	64	40	0100 0000	119	77	0111 0111
0	0A	0000 1010	65	41	0100 0001	120	78	0111 1000
11	0B	0000 1011	66	42	0100 0010	121	79	0111 1001
12	0C	0000 1100	67	43	0100 0011	123	7A	0111 1010
13	0D	0000 1101	68	44	0100 0100	113	7B	0111 1011
14	0E	0000 1110	69	45	0100 0101	124	7C	0111 1100
15	0F	0000 1111	70	46	0100 0110	125	7D	0111 1101
16	10	0001 0000	71	47	0100 0111	126	7E	0111 1110
17	11	0001 0001	72	48	0100 1000	127	7F	0111 1111
18	12	0001 0010	73	49	0100 1001	128	80	0111 0000
19	13	0001 0011	74	4A	0100 1010	129	81	0111 0001
20	14	0001 0100	75	4B	0100 1011	130	82	0111 0010
21	15	0001 0101	76	4C	0100 1100	131	83	0111 0011
22	16	0001 0110	77	4D	0100 1101	122	84	0111 0100
23	17	0001 0111	78	4E	0100 1110	133	85	0111 0101
24	18	0001 1000	79	4F	0100 1111	134	86	0111 0110
25	19	0001 1001	80	50	0101 0000	135	87	0111 0111
26	1A	0001 1010	81	51	0101 0001	136	88	0111 1000
27	1B	0001 1011	82	52	0101 0010	137	89	0111 1001
28	1C	0001 1100	83	53	0101 0011	138	8A	1000 1010
29	1D	0001 1101	84	54	0101 0100	139	8B	1000 1011
30	1E	0001 1110	85	55	0101 0101	140	8C	1000 1100
31	1F	0001 1111	86	56	0101 0110	141	8D	1000 1101
32	20	0010 0000	87	57	0101 0111	142	8E	1000 1110
33	21	0010 0001	88	58	0101 1000	143	8F	1000 1111
34	22	0010 0010	89	59	0101 1001	144	90	1000 0000
35	23	0010 0011	90	5A	0101 1010	145	91	1000 0001
36	24	0010 0100	91	5B	0101 1011	146	92	1000 0010
37	25	0010 0101	92	5C	0101 1100	147	93	1000 0011
38	26	0010 0110	93	5D	0101 1101	148	94	1000 0100
39	27	0010 0111	94	5E	0101 1110	149	95	1000 0101
40	28	0010 1000	95	5F	0101 1111	150	96	1000 0110
41	29	0010 1001	96	60	0110 0000	151	97	1000 0111
42	2A	0010 1010	97	61	0110 0001	152	98	1000 1000
43	2B	0010 1011	98	62	0110 0010	153	99	1000 1001
44	2C	0010 1100	99	63	0110 0011	154	9A	1001 1010
45	2D	0010 1101	100	64	0110 0100	155	9B	1001 1011
46	2E	0010 1110	101	65	0110 0101	156	9C	1001 1100
47	2F	0010 1111	102	66	0110 0110	157	9D	1001 1101
48	30	0011 0000	103	67	0110 0111	158	9E	1001 1110
49	31	0011 0001	104	68	0110 1000	159	9F	1001 1111
50	32	0011 0010	105	69	0110 1001	160	00	1001 0000
51	33	0011 0011	106	6A	0110 1010	161	A1	1001 0001
52	34	0011 0100	107	6B	0110 1011	162	A2	1001 0010
53	35	0011 0101	108	6C	0110 1100	163	A3	1001 0011
54	36	0011 0110	109	6D	0110 1101	164	A4	1001 0100

DEC	HEX	BINARY
165	A5	1001 0101
166	A6	1001 0110
167	A7	1001 0111
168	A8	1001 1000
169	A9	1001 1001
170	AA	1010 1010
171	AB	1010 1011
172	AC	1010 1100
173	AD	1010 1101
174	AE	1010 1110
175	AF	1010 1111
176	B0	1010 0000
177	B1	1010 0001
178	B2	1010 0010
179	B3	1010 0011
180	B4	1010 0100
181	B5	1010 0101
182	B6	1010 0110
183	B7	1010 0111
184	B8	1010 1000
185	B9	1010 1001
186	BA	1011 1010
187	BB	1011 1011
188	BC	1011 1100
189	BD	1011 1101
190	BE	1011 1110
191	BF	0011 1111
192	C0	1100 0000
193	C1	1100 0001
194	C2	1100 0010
195	C3	1100 0011
196	C4	1100 0100
197	C5	1100 0101
198	C6	1100 0110
199	C7	1100 0111
200	C8	1100 1000
201	C9	1100 1001
202	CA	1100 1010
203	CB	1100 1011
204	CC	1100 1100
205	CD	1100 1101
206	CE	1100 1110
207	CF	1100 1111
208	D0	1101 0000
209	D1	1101 0001
210	D2	1101 0010
211	D3	1101 0011
212	D4	1101 0100
213	D5	1101 0101
214	D6	1101 0110
215	D7	1101 0111
216	D8	1101 1000
217	D9	1101 1001
218	DA	1101 1010
219	DB	1101 1011
220	DC	1101 1100
221	DD	1101 1101
222	DE	1101 1110
223	DF	1101 1111
224	E0	1110 0000
225	E1	1110 0001

DEC	HEX	BINARY
226	E2	1110 0010
227	E3	1110 0011
228	E4	1110 0100
229	5	1110 0101
230	E6	1110 0110
230	E7	1110 0111
232	E8	1110 1000
233	E9	1110 1001
234	EA	1110 1010
235	EB	1110 1011
236	EC	1110 1100
237	ED	1110 1101
238	EE	1110 1110
239	EF	1110 1111
240	F0	1111 0000
241	F1	1111 0001
242	F2	1111 0010
243	F3	1111 0011
244	F4	1111 0100
245	F5	1111 0101
246	F6	1111 0110
247	F7	1111 0111
248	F8	1111 1000
249	F9	1111 1001
250	FA	1111 1010
251	FB	1111 1011
252	FC	1111 1100
253	FD	1111 1101
254	FE	1111 1110
255	FF	1111 1111

DEC	HEX	BINARY

เอกสารอ้างอิง

เอกสารอ้างอิงที่เป็นหนังสือภาษาอังกฤษ

FTP Software, "PC/TCP packet driver specification & programing" , IEEE Trans. On Broadcasting, Vol 1.09 , 66p., 1994

Laura A. Chappell and Dan E. Hakes, "NetWare LAN Analysis", SYBEX Inc., 693 p., 1994.

Novell, "LANalyzer for Windows 2.2 Installation and User's Guide", Novell Inc., 244p., 1996.

Theakston, "NetWare LANs Performance and Troubleshooting", ADDISON-WESLEY, 381p., 1995.

Dah Ming Chiu and Ram Sudama, "NETWORK MONITORING EXPLAINED Design and Application", ELLIS HORWOOD, 207p., 1992.

Ed Taylor, "Internetworking Handbook", McGraw-HILL, Inc, pp. 95-450.

เอกสารอ้างอิงที่เป็นวารสารภาษาอังกฤษ

Jon Postel, "Transmission Control Protocol", RFC 793, USC/Information Sciences Institute, 1981, pp. 1-19.

Jon Postel, "User Datagram Control Protocol", RFC 768, USC/Information Sciences Institute, 1980, pp. 1-3.

Jon Postel, "Internet Protocol", RFC 760, USC/Information Sciences Institute, 1980.

Jon Postel, "Assigned Numbers", RFC 762, USC/Information Sciences Institute, 1980.

Gabriel Ciobotaru, "GRAFICA Ver. 2.0", Grafica ™ ,1996.

กิตติกรรมประกาศ

ต้องขอขอบคุณอย่างมากแก่พ่อและแม่ที่คอยให้กำลังใจ และ กำลังเงินในการทำโครงงานนี้ และท้ายสุดเลี้ยงเคเอฟซีทุกมื้อเลย

ขอขอบคุณอาจารย์ผู้ประสิทประสาทวิชาต่างๆให้เราเพื่อได้ทำวิทยานิพนธ์ฉบับนี้ และ ยังแนะนำข้อมูลต่างๆที่จำเป็น อาจารย์ ธนา หงส์สกุล (ที่ปรึกษาโครงงาน) อาจารย์ บรรจง ปิยะธำรง (สอนวิชาด้านระบบเครือข่าย) และ คนอื่นที่ไม่ได้กล่าวถึงอีกมาก

ขอบคุณรุ่นพี่ที่ได้ทำโครงงานนี้เพื่อเป็นแนวทางต่างๆให้สามารถทำงานได้ง่ายขึ้น

ขอบใจแด่เพื่อนที่คอยให้กำลังใจ และ กำลังแรงในเรื่องต่างๆ ไม่ว่าการกิน การนอน การคุย การฟัง การเล่น

และ ขอบคุณทุกๆท่านที่ได้ช่วยเหลือเพื่อให้โครงงานนี้ประสพความสำเร็จ