# บทที่ 5
# โปรโตคอล (Protocol)

การที่จะทราบว่าโปรโตคอล (Protocol) คืออะไรต้องกล่าวถึงยุคแรกของการผลิตคอมพิวเตอร์ ในยุคนั้นผู้ผลิตแต่ละรายก็ได้ผลิตคอมพิวเตอร์ตามมาตรฐานของตนเองขึ้นมา ซึ่งสามารถทำงานได้เฉพาะกับเครื่องๆ เดียวเท่านั้น จากการที่มีการพัฒนาเครื่องคอมพิวเตอร์และระบบการทำงานที่ทันสมัย ทำให้การติดต่อระหว่างเครื่องคอมพิวเตอร์เป็นเรื่องที่สำคัญขึ้นมา แต่เนื่องจากการที่เครื่องแต่ละเครื่องผลิตขึ้นมาโดยมาตรฐานที่ไม่เหมือนกัน ทำให้การส่งข้อมูลระหว่างเครื่องนั้นเป็นไปได้ยาก ดังนั้นจากจุดนี้ทำให้เกิดการสร้างมาตรฐานของข้อมูลขึ้น เพื่อความสะดวกของเครื่องคอมพิวเตอร์ในการที่จะติดต่อสื่อสารกัน โปรโตคอลการสื่อสารมีมากมายหลายแบบ เช่น **X.25, NetBEUI, IPX/SPX, TCP/IP** เป็นต้น

กระบวนการที่โปรโตคอลในระดับต่างๆ ถูกรวมเข้าโปรโตคอลอื่นๆ เรียกว่า "**Binding**" การดำเนินการนี้จะเกิดขึ้นเพื่อจัดเตรียมให้ข้อมูลได้รับการกำหนดเส้นทางการขนส่งตั้งแต่ระดับ **Application Layer** ลงมาจนถึงการ์ดเชื่อมต่อระบบเครือข่าย และทำให้มั่นใจได้ว่าข้อมูลได้ถูกจัดเตรียมไว้ให้สามารถใช้บริการจากโปรโตคอลที่อยู่ในระดับสูงกว่าและต่ำกว่า และการ์ดเชื่อมต่อระบบเครือข่ายสามารถทำการขนส่งข้อมูลเข้าและออกจากโหนดนั้นๆ ได้

## 5.1 หน้าที่ของโปรโตคอล

โปรโตคอล คือกฎเกณฑ์และกระบวนการในการสื่อสาร ซึ่งกฎของการสื่อสารนี้สามารถนำมาประยุกต์ใช้กับการสื่อสารข้อมูลในระบบเครือข่ายคอมพิวเตอร์ได้ โดยที่โปรโตคอลจะมีอยู่หลายชนิด โปรโตคอลแต่ละชนิดจะมีจุดประสงค์ในการทำงานที่แตกต่างกันแต่จะช่วยในการสื่อสารบนระบบเครือข่าย โดยโปรโตคอลแต่ละตัวจะทำงานร่วมกันเป็นลำดับชั้นในรูปแบบของชุดโปรโตคอล (**Protocol Stack**) เช่น ในโครงสร้างแบบ **OSI Model** จะมีโปรโตคอลต่างๆ ทำงานอยู่ในเลเยอร์แต่ละระดับชั้น ในการสื่อสารข้อมูล การทำงานในแต่ละเลเยอร์จะเป็นการทำงานหนึ่งขั้นตอน โดยการทำงานแต่ละขั้นตอนจะมีกระบวนการที่แตกต่างกันไป

กฎเกณฑ์การสื่อสารก็มีลักษณะเดียวกับการพูด หากพูดคนละภาษาก็ไม่สามารถเข้าใจซึ่งกันและกันได้ ดังนั้นการใช้กฎเกณฑ์ที่แตกต่างกันจะทำให้ไม่สามารถสื่อสารกันได้ ในทำนองเดียวกันเครื่องคอมพิวเตอร์ที่ใช้โปรโตคอลแตกต่างกันจะไม่สามารถทำการสื่อสารข้อมูลระหว่างกันได้

### *5.1.1 Routable Protocol*

ในราวกลางปี **1980** ระบบเครือข่าย **LAN** ส่วนใหญ่ถูกจัดตั้งแยกจากกัน โดย **LAN** หนึ่งวงจะถูกใช้โดยองค์กรหรือหน่วยงานใดหน่วยงานหนึ่งเท่านั้น จึงยังไม่เกิดระบบเครือข่ายขนาดใหญ่ขึ้นมา ต่อมาการทำงานบนระบบเครือข่ายเริ่มที่จะได้รับความนิยมเพิ่มมากขึ้น จึงมีความต้องการในการเชื่อมต่อวง **LAN** ต่างๆ เหล่านี้เข้าด้วยกันเป็นระบบเครือข่าย **WAN** เพื่อแลกเปลี่ยนข้อมูลทางธุรกิจ หรือข้อมูลอื่นๆ ระหว่างกัน จนเป็นเครือข่ายขนาดใหญ่และซับซ้อน ในการขนส่งข้อมูลระหว่างวง **LAN** เหล่านี้ จึงมีเส้นทางในการขนส่งข้อมูลได้หลายเส้นทาง จำเป็นที่จะต้องใช้โปรโตคอลที่สามารถระบุเส้นทางการขนส่งข้อมูลได้ หรือที่เรียกว่า **Routable Protocol** สำหรับช่วยในการขนส่งแพ็กเก็ตข้อมูลข้ามไปมาระหว่างวง **LAN** ที่เชื่อมโยงเข้าด้วยกันนี้ได้อย่างเหมาะสม

### *5.1.2 Binding Process*

กระบวนการในการขนส่งข้อมูลเป็นกระบวนการที่โปรโตคอลต่างๆ ทำการติดต่อระหว่างกันและกัน และส่งไปยังการ์ดเชื่อมต่อระบบเครือข่าย เพื่อทำการขนส่งข้อมูลทางกายภาพจริงๆ การที่โปรโตคอล และการ์ดเชื่อมต่อระบบเครือข่ายทำงานร่วมกันจะต้องมีกระบวนการที่เรียกว่า "**Binding**" เช่นถ้าต้องการให้ใช้โปรโตคอล 2 ตัวในการทำงาน (**IPX/SPX** หรือ **TCP/IP**) จะต้องรวมโปรโตคอลนี้เข้ากับไดร์ฟเวอร์ของการ์ดเชื่อมต่อระบบเครือข่าย โดยทั่วไปกระบวนการ **Binding** จะเริ่มตั้งแต่การติดตั้งระบบปฏิบัติการหรือติดตั้งโปรโตคอล หรือการเรียกใช้โปรโตคอล การดำเนินการเช่นนี้จะเอื้ออำนวยต่อความสำเร็จในการจัดตั้งการเชื่อมต่อ

ในกระบวนการ **Binding** มีขั้นตอนมากกว่าการรวมโปรโตคอลที่เกี่ยวข้องเข้ากับไดร์ฟเวอร์ของการ์ดเชื่อมต่อระบบเครือข่าย แต่โปรโตคอลแต่ละตัวใน **Stack** จะต้องมีส่วนเกี่ยวข้องกับองค์ประกอบของโปรโตคอลในเลเยอร์ลำดับชั้นที่สูงกว่าหรือต่ำกว่า เพื่อให้สามารถทำงานได้อย่างราบรื่น เช่นโปรโตคอล **TCP/IP** อาจจะต้องติดต่อกับโปรโตคอล **NetBIOS** ใน **Session Layer** รวมไปถึงส่วนของไดร์ฟเวอร์ที่อยู่ในลำดับต่ำกว่า เป็นต้น

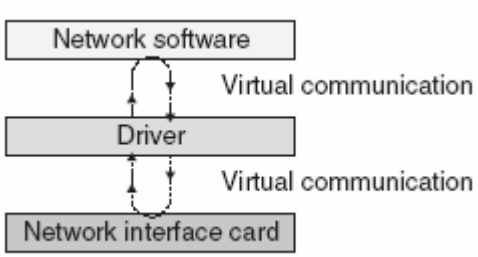
### *5.1.3 Device Driver*

**Device Driver** หรือบางครั้งอาจจะเรียกว่า "ไดร์ฟเวอร์ (**Driver**)" คือซอฟต์แวร์ที่ทำให้เครื่องคอมพิวเตอร์สามารถทำงานร่วมกับอุปกรณ์ที่ต้องการได้ เมื่อติดตั้งอุปกรณ์ต่างๆ เข้ากับเครื่องคอมพิวเตอร์ ระบบปฏิบัติการจะยังไม่สามารถทำงานร่วมกับอุปกรณ์เหล่านั้นได้จนกว่าจะมีการติดตั้งไดร์ฟเวอร์และตั้งค่าต่างๆ ให้เหมาะสม ทำนองเดียวกันการ์ดเชื่อมต่อระบบเครือข่าย ซึ่งใช้ในการเชื่อมต่อเครื่องคอมพิวเตอร์เข้ากับระบบเครือข่ายยังคงเป็นเพียงฮาร์ดแวร์ การที่จะทำให้สามารถทำงานกับระบบเครือข่ายได้อย่างเหมาะสม การ์ดเหล่านี้จะต้องอาศัยไดร์ฟเวอร์ที่ช่วยให้สามารถทำงานกับระบบปฏิบัติการและโปรโตคอลซึ่งมีส่วนร่วมในระบบเครือข่ายได้

ในความหมายของคำว่า "ไดร์ฟเวอร์" นั้น สิ่งสำคัญที่จะต้องบันทึกไว้คือว่ามีอุปกรณ์ที่แตกต่างกันหลายชนิดที่แตกต่างกันในระบบเปิด เช่น เครื่อง **Wintel (Windows/Intel based)** ที่ได้รับการออกแบบมาให้ทำงานได้กับการเพิ่มเติมอุปกรณ์ที่ผลิตจากบริษัทต่างๆ ถึงแม้ว่าอุปกรณ์เหล่านี้จะสามารถจัดแบ่งตามประเภทได้ เช่นเครื่องพิมพ์ ดิสก์ไดร์ฟ อุปกรณ์ตัวชี้ และโมเด็ม แต่อุปกรณ์แต่ละประเภทเหล่านี้จะมีการทำงานตามวิถีทางของตัวเอง โดยมีความสามารถของตัวเอง วิธีการติดตั้งอุปกรณ์ของตัวเอง และวิธีการทำงานที่เป็นของตัวเอง ดังนั้นงานของไดร์ฟเวอร์ก็คือจะต้องรับรู้และทำให้อุปกรณ์นั้นๆ มีความสามารถเหล่านี้อย่างครบถ้วนบนระบบปฏิบัติการและโปรโตคอลที่อุปกรณ์เหล่านี้ทำงานด้วย (ในกรณีของการ์ดเชื่อมต่อระบบเครือข่าย) ผลก็คือบริษัทผู้ผลิตอุปกรณ์จะเขียนไดร์ฟเวอร์สำหรับอุปกรณ์ของตนเอง ปรับแต่งไดร์ฟเวอร์เพื่อทำให้อุปกรณ์นั้นๆ มีสมรรถนะสูงสุดและใช้ได้กับระบบที่ต้องการให้อุปกรณ์นั้นๆ เข้ามาทำงานร่วม

### *5.1.4 Driver กับ OSI*

ไดร์ฟเวอร์ของการ์ดเชื่อมต่อระบบเครือข่ายจะทำงานใน **MAC Sub-layer** ใน **Data Link Layer** ของ **OSI Reference Model** ซึ่ง **Mac Sub-layer** จะเป็นผู้ส่งข้อมูลให้กับ **Physical Layer** โดยไดร์ฟเวอร์จะสนับสนุนการทำงานของการ์ดเชื่อมต่อระบบเครือข่ายกับ **Redirector** ซึ่งเป็นส่วนหนึ่งของซอฟต์แวร์ระบบเครือข่ายซึ่งทำงานอยู่ในเครื่องคอมพิวเตอร์บนระบบเครือข่าย ดังแสดงตามรูปที่ **5 – 1**




รูปที่ 5 – 1 การสื่อสารระหว่าง NIC กับซอฟต์แวร์ระบบเครือข่ายผ่านไดร์ฟเวอร์

### *5.1.5 NDIS and ODI*

เมื่อกล่าวถึงการ์ดเชื่อมต่อระบบเครือข่าย มีการพัฒนามาตรฐานขึ้นมา 2 แบบ เพื่อช่วยเหลืองานในการสร้างไดร์ฟเวอร์ให้สามารถสนับสนุนระบบปฏิบัติการและโปรโตคอลที่แตกต่างกันซึ่งมีใช้บนระบบเครือข่ายเป็นจำนวนมาก หนึ่งในมาตรฐานนี้คือ **NDIS (Network Device Interface Specification)** ที่ได้รับการพัฒนาโดยบริษัทไมโครซอฟต์ และบริษัท **3Com** ส่วนอีกมาตรฐานหนึ่ง คือ **ODI (Open Data-Link Interface)** ที่ได้รับการพัฒนาโดยบริษัท **Novell** และ **Apple** โดยทั้งคู่เป็นซอฟต์แวร์การต่อเชื่อมซึ่งกำหนดขอบเขตร่วมกัน ระหว่างการ์ดเชื่อมต่อระบบเครือข่ายกับโปรโตคอลในระดับสูงเพื่อให้ทำงานได้อย่างมีประสิทธิภาพ โดยจะจัดให้มีจุดเชื่อมต่อ **(linkage point)** ระบบเครือข่ายกับโปรโตคอล มาตรฐานเหล่านี้จะทำให้สามารถเขียนไดร์ฟเวอร์สำหรับการ์ดเชื่อมต่อระบบเครือข่ายให้เป็นไปตามคุณลักษณะเฉพาะของตนเอง เพื่อสนับสนุนโปรโตคอลได้มากกว่า 1 ตัวในการ์ดเดียว นั่นคือ **NDIS** และ **ODI** จะจัดให้มีบางสิ่งที่เป็นระเบียบแบบแผนที่สามารถเข้าใจได้ทั้ง 2 ฝ่าย ให้กับการ์ดเชื่อมต่อระบบเครือข่ายและชุดโปรโตคอล **(protocol stack)** จึงเป็นเหมือนบางสิ่งบางอย่างที่เปรียบได้กับระเบียบแบบแผนที่ซึ่งใช้สัญลักษณ์สากลที่นักท่องเที่ยวทั้งหมดสามารถเข้าใจได้ โดยไม่คำนึงถึงภาษาพื้นเมืองของแต่ละคน

## 5.2 โปรโตคอลสแต็กมาตรฐาน (Standard Protocol Stack)

อุตสาหกรรมคอมพิวเตอร์ได้พัฒนาโปรโตคอลมาตรฐานสำหรับการสื่อสารข้อมูลไว้หลายชนิด จึงทำให้ผลิตภัณฑ์ฮาร์ดแวร์และซอฟต์แวร์ต่างๆ ที่ผลิตออกจำหน่ายสามารถทำงานร่วมกับมาตรฐานเหล่านี้ได้ ต้นแบบในการกำหนดมาตรฐานต่างๆ เหล่านี้ได้แก่

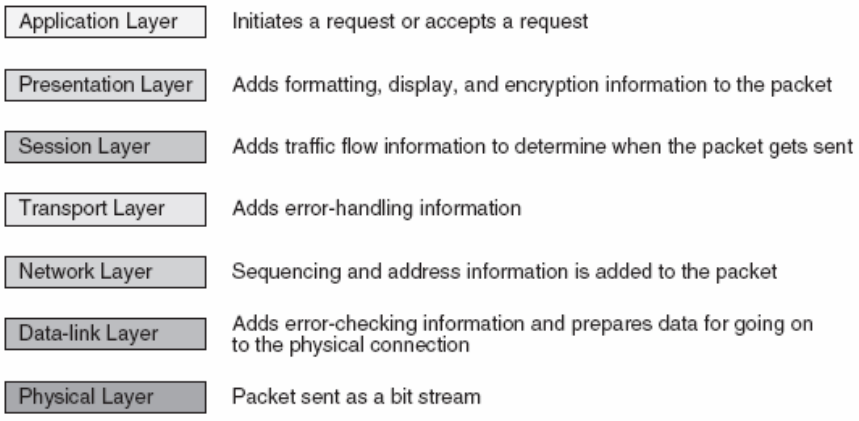
- มาตรฐานกลางของ **OSI Reference Model**
- สถาปัตยกรรมเครือข่าย **SNA** ของบริษัท **IBM**
- **DECnet** ของบริษัท **Digital**
- **NetWare** ของบริษัท **Novell**
- **AppleTalk** ของบริษัท **Apple**
- โปรโตคอลมาตรฐานของอินเตอร์เน็ต คือ **TCP/IP**

ในสแต็กแต่ละชนิดจะมีโปรโตคอลที่รองรับการทำงานในแต่ละระดับชั้นอยู่เป็นจำนวนมาก ซึ่งโปรโตคอลแต่ละตัวในแต่ละเลเยอร์ก็จะมีหน้าที่การทำงานที่แตกต่างกัน โดยจะทำงานตามหน้าที่ซึ่งถูกกำหนดไว้ในเลเยอร์นั้นๆ อย่างไรก็ตามงานในการสื่อสารของระบบเครือข่ายจะต้องใช้โปรโตคอลตัวใดตัวหนึ่งใน **Application, Transport** และ **Network Layer** เป็นหลักในการสื่อสารข้อมูล ซึ่งจะต้องเป็นโปรโตคอลที่สามารถทำงานได้มากกว่า 1 เลเยอร์ขึ้นไป

## 5.3 โครงสร้างระบบเครือข่ายกับโปรโตคอล (Network Models and Protocols)

การมั่นใจว่าโครงสร้างระบบเครือข่ายที่มีเป็นสิ่งที่วิเศษและหรูหรา จะดูได้จากมุมมองของระบบเครือข่ายในรูปแบบของเลเยอร์ที่บรรจุอยู่ภายในทำให้ง่ายต่อการจัดเข้าชุดและง่ายต่อการทำความเข้าใจความสัมพันธ์ระหว่างการปฏิบัติงานต่างๆ และการให้บริการที่จัดให้มีโดยระบบเครือข่าย อย่างไรก็ตามคำอธิบายโดยตลอดของโครงสร้างเหล่านี้จะได้พบกับการอ้างถึงโปรโตคอลนี้ทำงานในเลเยอร์นี้ โปรโตคอลนั้นทำงานที่เลเยอร์นั้นอยู่ตลอดเวลา

หลังจากที่ได้อธิบายแนวความคิดของโครงสร้างระบบเครือข่ายโดยการแบ่งการทำงานของระบบเครือข่ายเป็นหลายเลเยอร์ โดยมีโปรโตคอลซึ่งถูกแสดงให้เห็นในลักษณะของซอฟต์แวร์ที่ทำให้ระบบเครือข่ายทำงานได้อย่างแท้จริง อย่างไรก็ตามในความเป็นจริงแล้วความสัมพันธ์นี้นับว่าง่ายมาก คือโครงสร้างระบบเครือข่ายจะอธิบายว่าต้องการที่จะทำอะไรและโปรโตคอลก็จะเป็นตัวที่ทำให้เกิดขึ้น มากไปกว่านั้นต้องขอบคุณการอธิบายโครงสร้างของระบบเครือข่ายเป็นเลเยอร์ เพราะว่าโปรโตคอลที่เกี่ยวข้องกับการให้บริการโดยเฉพาะ เช่นจัดเตรียมการเข้าถึงระบบเครือข่าย หรือกำหนดแอดเดรสให้กับเฟรมข้อมูล อาจจะสมมติว่าโปรโตคอลอื่นที่ทำงานบนเลเยอร์อื่น กำลังดูแลการให้บริการเพื่อให้มั่นใจว่ากระบวนการการสื่อสารทั้งหมดเกิดขึ้นอย่างถูกต้อง นั่นคือโปรโตคอลที่กำหนดแอดเดรสให้กับเฟรมข้อมูลจะไม่รู้เกี่ยวกับว่าเฟรมข้อมูลนั้นจะถูกนำเข้าไปบนสายเคเบิลระบบเครือข่ายได้อย่างไร ไม่ว่าจะเป็นการส่งเฟรมข้อมูลโดยตรงหรือหากจำเป็นก็จะส่งออกไปใหม่ ซึ่งจะมีโปรโตคอลอื่นที่ทำงานบนเลเยอร์อื่นซึ่งจะคอยดูแลกระบวนการต่างๆ เหล่านี้ ดังนั้นโปรโตคอลซึ่งทำหน้าที่กำหนดแอดเดรสให้กับเฟรมข้อมูล จะมีความกังวลเพียงเฉพาะการทำงานของตัวเองให้ถูกต้องเท่านั้น การจัดแบ่งโครงสร้างเป็นเลเยอร์ ทำให้โปรโตคอลถูกใช้อย่างฟุ่มเฟือยในการพัฒนาในวงแคบเพื่อมุ่งในการทำงานของตนเองและคาดว่างานอย่างอื่นจะได้รับการดูแลอย่างเหมาะสมที่ซึ่งโปรโตคอลขึ้นอยู่กับโครงสร้าง เช่นใน **ISO/OSI Reference Model** ซึ่งเป็นส่วนของการกำหนดมาตรฐาน โดยการแยกและกำหนดมาตรฐานที่จัดให้มีในแต่ละเลเยอร์ โครงสร้างของระบบเครือข่ายจะอธิบายการกำหนดมาตรฐานของโปรโตคอลซึ่งทำงานในแต่ละเลเยอร์ได้อย่างมีประสิทธิภาพ โดยที่โครงสร้างจะไม่สามารถกำหนดโปรโตคอลได้ด้วยตัวเอง นั่นคือ **ISO/OSI Model** จะไม่กำหนดโปรโตคอล ใน **Application Layer** เช่นโปรโตคอลที่ใช้โดยโปรแกรมการโยกย้ายไฟล์ เพื่อให้ได้รับการ **access** เข้าไปยังระบบเครือข่าย แล้วโครงสร้างระบบเครือข่ายทำอะไร? แทนที่จะกำหนดมาตรฐานการให้บริการและสร้างการเชื่อมต่อซึ่งทำโดยโปรโตคอลที่ทำงานใน **Application Layer** ด้วยวิธีนี้โครงสร้างระบบเครือข่ายจะจัดให้มีโครงร่างของโปรโตคอลมาตรฐานที่โปรแกมประยุกต์ต้องใช้




รูปที่ 5 – 2 หน้าที่ของโปรโตคอลในเลเยอร์ต่างๆ ของโครงสร้าง **OSI Reference Model**

**ISO/OSI** และโครงสร้างระบบเครือข่ายแบบอื่นกำหนดเลเยอร์ที่แบ่งออกอย่างเห็นได้เด่นชัดที่เกี่ยวข้องกับการจัดหีบห่อข้อมูล การส่งและการรับสัญญาณข้อมูล คำถามคือโปรโตคอลตัวใดที่จัดให้มีการบริการที่จำเป็นในแต่ละเลเยอร์ และมีโปรโตคอลเพียงตัวเดียวที่ทำงานในทุกเลเยอร์ซึ่งรองรับการปฏิบัติงานที่เหมาะสมในแต่ละระดับ ใช่หรือไม่ คำตอบคือไม่ใช่ หรือมีชุดของโปรโตคอลที่มีความเกี่ยวข้องกัน ซึ่งโปรโตคอลแต่ละตัวจะทำงานในเลเยอร์ต่างๆ ที่ร่วมกันรองรับการปฏิบัติงานทั้งหมดที่ต้องทำในการจัดหีบห่อแพ็กเก็ตข้อมูล การส่งและรับสัญญาณข้อมูล ใช่หรือไม่ คำตอบคือใช่ และมีชุดของโปรโตคอลที่มีความเกี่ยวข้องกันมากกว่า **1** ชุด ใช่หรือไม่ คำตอบก็คือใช่

ตอนนี้ก็ถึงเวลาที่จะเข้าสู่โลกของ **Protocol stacks** และ **Protocol suites** ถึงแม้ว่าคำทั้งสองจะสามารถใช้แทนกันได้แต่ก็มีบางสิ่งเป็นนัยยะที่แตกต่างกันระหว่าง **stack** กับ **suite**

### *5.3.1 Protocol Suites*

**Protocol Suite** กล่าวถึงชุดของโปรโตคอลที่ได้รับการออกแบบและสร้าง (โดยมากจะมีโปรโตคอลมากกว่า **1** ตัวในแต่ละเลเยอร์) ให้เป็นส่วนประกอบที่สมบูรณ์ของชุดฟังก์ชันการทำงานอย่างราบเรียบ ดังนั้นชุดของโปรโตคอล **TCP/IP** ที่ได้รับการออกแบบโดยผู้จำหน่ายแต่ละรายจะเป็นตัวแทนของ **suites** ซึ่งเมื่อนำไปใช้งานบนระบบเครือข่าย **TCP/IP** ของผู้จำหน่ายนั้นๆ ก็จะเป็น **network stack**

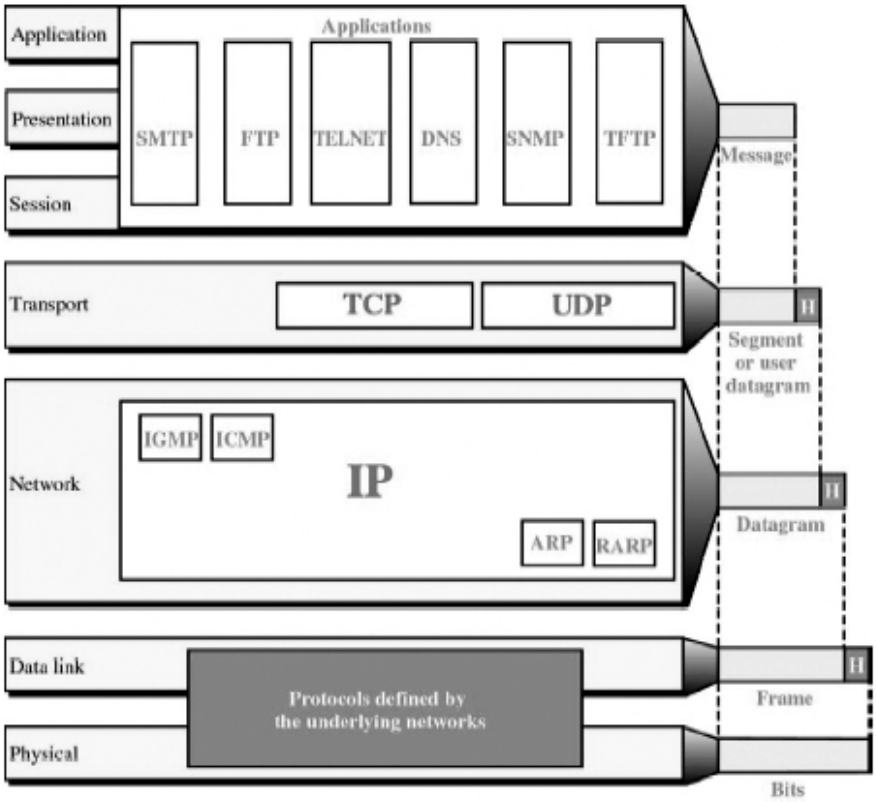
### *5.3.2 Protocol Stack*

ในปัจจุบัน ที่มีการใช้เครือข่ายอินเตอร์เน็ตอย่างแพร่หลาย และการเติบโตของระบบเครือข่ายสากล **TCP/IP** สำหรับ **ISO/OSI model** จึงเกิดขึ้น ถึงแม้ว่า **ISO/OSI** จะยังคงเป็นการอธิบายโครงสร้างของระบบเครือข่ายที่ก่อตั้งมานานแล้ว แต่ก็ทำงานอยู่บนระบบเครือข่ายเป็นจำนวนมาก และเนื่องจากขณะนี้จะเน้นไปที่โปรโตคอล **TCP/IP** ในส่วนนี้จะใช้ **TCP/IP model** เป็นพื้นฐานสำหรับทำความเข้าใจตัวอย่างของ **protocol stack**

**Stack** จะกล่าวถึงชุดของโปรโตคอลที่สมบูรณ์ ซึ่งโดยมากจะมีโปรโตคอล **1** ตัวในแต่ละเลเยอร์ที่ทำงานบนระบบเครือข่าย และระบบเครือข่ายอาจจะต้องอาศัย **protocol stack** มากกว่า **1** ชุดในการทำงาน ตัวอย่างเช่นระบบเครือข่ายอาจจะ **run Novell Netware stack** และ **TCP/IP stack** อย่างไรก็ตาม **stacks** ก็เป็นอิสระต่อกัน นั่นคือโปรโตคอลในแต่ละ **stack** จะได้รับการออกแบบมาให้โต้ตอบกับและอาศัยโปรโตคอลที่เป็นเครือญาติใน **stack** ของตนเอง แทนที่จะต้องอาศัยโปรโตคอลใน **stack** อื่น ให้ลองคิดว่า **stack** เป็นชุดลำดับของโปรโตคอลที่รันบนระบบเครือข่าย หรือเครื่องคอมพิวเตอร์ระบบเครือข่าย และให้คิดว่า **suite** เป็นชุดของโปรโตคอลที่สร้างโดยผู้จำหน่ายโดยเฉพาะหรือองค์กรใดองค์กรหนึ่ง ในหัวข้อถัดไปจะอธิบายรายละเอียดของชุดโปรโตคอลแต่ละชุดอย่างชัดเจน และโปรโตคอลบางตัวที่สำคัญ

## 5.4 โปรโตคอลของไมโครซอฟต์

**TCP/IP Protocol Suit** เป็นชุดโปรโตคอลหลักของบริษัท ไมโครซอฟต์ ชุดของโปรโตคอล **TCP/IP** มีการจัดแบ่งออกเป็น **4** ระดับชั้น ซึ่งไม่ตรงกับ **OSI Model** พอดี โดยแต่ละระดับชั้นจะประกอบด้วยเลเยอร์ต่างๆ ใน **OSI Model** มากกว่า **1** เลเยอร์ สำหรับชุดโปรโตคอลที่ได้รับการพัฒนาให้มาใช้ร่วมกับ **TCP/IP** ในเลเยอร์ต่างๆ เริ่มจากด้านบน ที่ซึ่งเป็นจุดเริ่มของการส่งสัญญาณข้อมูล ตามรูปที่ **5 – 3**




รูปที่ 5 – 3 โปรโตคอลใน **TCP/IP Suit**

### *5.4.1 Applications Layer*

**Application Layer** เป็นเลเยอร์ระดับสูงที่สุดที่แทบเหมือนกับไม่มีตัวตนในความหมายของระยะห่างจากกายภาพของระบบเครือข่าย เลเยอร์นี้เป็นที่ซึ่งโปรแกรมประยุกต์ต่างๆ **access** เข้าไปยังระบบเครือข่ายโดยการใช้โปรโตคอล **Application Layer** ของชุดโปรโตคอล **TCP/IP** จะประกอบด้วย **Application, Presentation** และ **Session Layer** ของ **OSI Model** โดยมีโปรโตคอลต่างๆ ที่ได้รับการพัฒนาให้มาใช้งานร่วมกับ **TCP/IP** ดังนี้

- **SMTP (Simple Mail Transfer Protocol)** ใช้สำหรับการส่งอีเมลล์ **(e-mail)**
- **FTP (File Transfer Protocol)** ใช้สำหรับถ่ายโอนไฟล์ข้อมูลระหว่างเครื่องที่ใช้โปรโตคอล **TCP/IP**
- **TELNET** ใช้สำหรับติดต่อกับอุปกรณ์ระบบเครือข่าย
- **DNS** ใช้สำหรับแปลงชื่อของ **IP Address**
- **SNMP (Simple Network Management Protocol)** ใช้สำหรับการบริหารเครือข่าย
- **HTTP (Hypertext Transfer Protocol)** ซึ่งเป็นโปรโตคอลซึ่งใช้ในการโยกย้ายไฟล์ที่เป็น **hypertext** ซึ่งโปรแกรมเบราส์เซอร์ **(browser)** อาศัยในการจัดส่งเว็บเพจที่บรรจุข้อมูลผสมกันของ ข้อความ รูปภาพ สัญญาณเสียง และสัญญาณภาพ

### *5.4.2 Transport Layer*

โปรโตคอลใน **Transport layer** มีหน้าที่เป็นสื่อกลางในการสื่อสารระหว่าง **Application Layer** กับ **Internetwork layer** โปรโตคอลในเลเยอร์นี้มีอยู่ 2 ตัวคือ **TCP (Transmission Control Protocol)** และ **UDP (User Datagram Protocol)**

#### *5.4.2.1 TCP (Transmission Control Protocol)*

เป็นโปรโตคอลการสื่อสารข้อมูลแบบต้องการให้มีการจัดตั้งการเชื่อมต่อ **(Connection Oriented)** คือมีลักษณะเหมือนการส่งข้อมูลเสียงทางโทรศัพท์ ผู้ใช้ต้องจัดตั้งการเชื่อมต่อก่อนแล้วจึงจะสามารถทำการส่งข้อมูลได้ และเมื่อเลิกใช้แล้วก็ทำการยกเลิกการติดต่อ โดยในขั้นตอนของการติดต่อนั้น จะทำการเชื่อมต่อเครื่องคอมพิวเตอร์ผู้ส่งและเครื่องคอมพิวเตอร์ผู้รับเข้าด้วยกัน ขั้นตอนแรกคือขั้นตอนที่ทำการเรียกไปยังเครื่องปลายทาง เพื่อขอทำการติดต่อนั้นเป็นช่วงที่ใช้เวลามากที่สุด **TCP** เป็นโปรโตคอลในระดับชั้นที่ **4** เมื่อเทียบกับ **OSI** มีลักษณะการทำงานเป็นวงจรเสมือน **(Virtual Circuit)** คือจะมีการทำวงจรขึ้นมาก่อนที่จะรับส่งข้อมูลกัน นั่นคือแต่ละโหนดต้องมีตารางของแอดเดรสและเส้นทางการขนส่งข้อมูลไปยังปลายทาง เพื่อให้รู้ว่าจะต่อกับใครจึงจะได้วงจรเสมือนตามต้องการ เมื่อจัดตั้งการเชื่อมต่อเสร็จแล้วก็จะรับส่งข้อมูลกันโดยใช้เส้นทางนี้ตลอด ดังนั้นจะไม่มีปัญหาเรื่องการเรียงลำดับของชุดข้อมูลผิดพลาด หรือ เกิดการซ้ำซ้อนของข้อมูล การส่งผ่านข้อมูลบน **TCP** เป็น **byte stream-oriented** สำหรับหน้าที่ของ **TCP** ก็คือ จัดการเรื่องตรวจสอบความผิดพลาด ทำ **flow control** ทำการ **multiplex** หรือ **demultiplex application layer connection** นอกจากนี้ก็ยังทำหน้าที่ควบคุมแลกเปลี่ยนสถานะและทำ **Synchronization** ด้วย และถ้าจำเป็นจะจัดให้มีการส่งสัญญาณข้อมูลออกไปใหม่ในเหตุการณ์ที่เกิดความผิดพลาดในการส่งสัญญาณข้อมูล ตามตัวอย่างที่กล่าวมานี้ **TCP** จึงถูกใช้ในระบบเครือข่ายของไมโครซอฟต์สำหรับการแบ่งปันการใช้ไฟล์และเครื่องพิมพ์ หมายเลขพอร์ตจะถูกใช้ในการอ้างอิงที่อยู่ของโปรแกรมใน **Application Layer** เพื่อบอกให้ทราบช่องทางในการติดต่อระหว่างโปรแกรมบนเครื่องหนึ่งกับโปรแกรมในเครื่องอื่น โดยหมายเลขพอร์ตและแอดเดรสจะถูกนำมารวมกันเพื่อสร้างเป็นซ็อกเก็ต **(Socket)** เพื่อเป็นช่องทางใช้ในการติดต่อไปยังโฮสต์อื่น

#### *5.4.2.2 UDP (User Datagram Protocol)*

**UDP** เป็นโปรโตคอลซึ่งไม่ได้จัดตั้งการเชื่อมต่อ **(Connectionless)** แต่จะรับผิดชอบการส่งข้อมูลแบบ **end-to-end** ที่เหมือนกับไม่ค่อยมีความน่าเชื่อถือ เพราะจะทำการส่งสัญญาณข้อมูลโดยไม่มีการตรวจสอบว่าข้อมูลได้ถูกจัดส่งไปถึงอีกฝ่ายหนึ่งอย่างถูกต้องหรือไม่ (โปรแกรมประยุกต์จะมีหน้าที่ในการตรวจสอบนี้) จึงเหมาะสำหรับการส่งข้อมูลขนาดเล็ก นอกจากนั้นการใช้พอร์ตของ **UDP** จะแตกต่างจากการใช้พอร์ตของ **TCP** จึงทำให้ทั้ง **TCP** และ **UDP** สามารถใช้พอร์ตเดียวกันได้โดยไม่รบกวนการทำงานซึ่งกันและกัน การใช้ **UDP** ในโลกแห่งความเป็นจริงจึงประกอบด้วย การ **browse** การ **logon** การแพร่กระจายข้อมูล **(broadcast)** หรือส่งสัญญาณข้อมูลแบบ **multicast** ไปยังผู้รับจำนวนมากในเวลาเดียวกัน

### *5.4.3 Internetwork Layer*

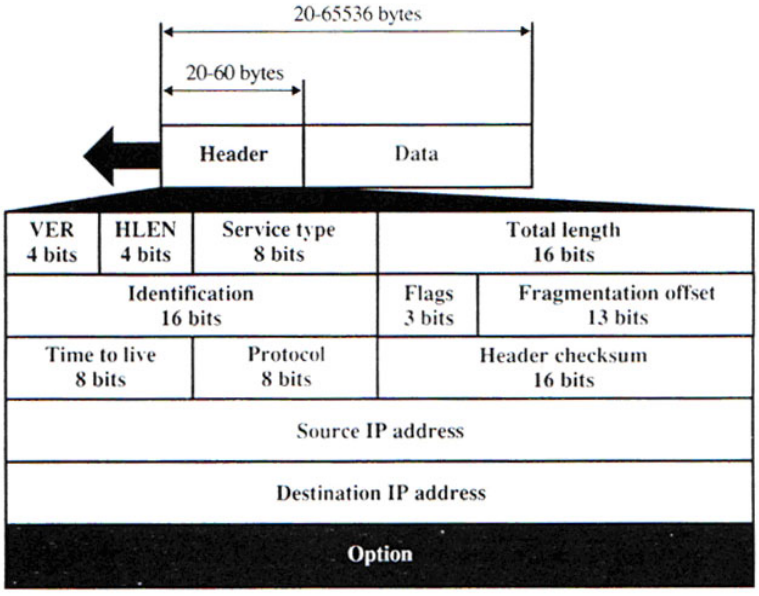
เลเยอร์นี้ตรงกับ **Network Layer** ของ **OSI Model** ซึ่งในเลเยอร์นี้จะเกี่ยวข้องกับการใช้โปรโตคอลหลายตัวในการกำหนดเส้นทางการขนส่งข้อมูลผ่านเราท์เตอร์ ซึ่งเป็นการขนส่งข้อมูลจากเครือข่ายหนึ่งไปยังอีก

เครือข่ายหนึ่งผ่านระบบอินเตอร์เน็ต **Internetwork Layer** เป็นเหมือนทางด่วนของข้อมูล ที่ซึ่งแพ็กเก็ตข้อมูลจะถูกจัดเรียงลำดับ กำหนดเส้นทางการขนส่ง และจัดส่งผ่านเครือข่ายแพ็กเก็ตสวิตชิ่ง **(packet-switching)** และเลเยอร์นี้ยังเป็นเหมือนที่อยู่ของโปรโตคอล **IP (Internet Protocol)** ซึ่งเป็นอีกครึ่งหนึ่งของ **TCP/IP** โปรโตคอลต่างๆ ที่ได้รับการพัฒนาให้รับผิดชอบการทำงานต่างๆ ในเลเยอร์นี้ มีดังนี้

#### *5.4.3.1 IP (Internet Protocol)*

เป็นโปรโตคอลการรับส่งข้อมูลใน **Network layer** แบบไม่ต้องการจัดตั้งการเชื่อมต่อ **(Connectionless Oriented)** ที่ถูกใช้โดยชุดโปรโตคอล **TCP/IP** ซึ่งโดยทั่วไปสำหรับระดับชั้นที่ **3** นี้มีทางเลือกสองแบบคือ สร้างเป็นวงจรเสมือน **(virtual circuit)** หรือจัดส่ง **datagram** ในที่นี้ **IP** ได้เลือกสภาพแวดล้อมแบบ **datagram** ซึ่งหลักการทำงานต่างๆ จะตรงข้ามกับวงจรเสมือนที่ชุด **TCP** ใช้อยู่บน **Transport Layer** โดยสรุปแล้วชุดโปรโตคอล **TCP/IP** นี้ออกแบบมาเพื่อใช้กับสภาพแวดล้อมแบบ **datagram** แต่ได้เพิ่มความน่าเชื่อถือเข้าไปไว้ด้วย โดย **IP** จะทำการขนส่งข้อมูลเป็นแพ็กเก็ตพิเศษที่เรียกว่า **Datagram** ซึ่งมีภาระหน้าที่ดังนี้

- กำหนดแอดเดรส โดยรวมจาก **network ID** และ **local host ID** จัดการ **status messages** ต่างๆ ซึ่งกำหนดไว้ **4** แบบคือ **destination unreachable/invalid, time out, parameter error** และ **redirect request**
- จัดการทำ **routing** โดยดำเนินตาม **Gateway- Gateway Protocol (GGP)** และยังกำหนดเวลาที่เรียกว่า **time to live** โดยจะลดค่าลงเรื่อยๆ เมื่อ **IP datagram** ได้ผ่านเข้าไปในเราท์เตอร์แต่ละตัว เพื่อป้องกันการเกิดขยะบนเครือข่าย
- อีกหน้าที่หนึ่งคือกำหนดชนิดของบริการเพื่อบอกว่า **datagram** จะเลือกใช้เส้นทางแบบใด ระหว่างเส้นทางที่มีการหน่วงเวลาต่ำ หรือเส้นทางที่มีแบนด์วิดธ์ **(bandwidth)** สูง หรือเส้นทางที่มีความน่าเชื่อถือสูง




รูปที่ **5 – 4 IP Datagram**

**IP** เป็นโปรโตคอลแบบ **Packet Switched** ที่ไม่ต้องการการเชื่อมต่อเช่นเดียวกับ **UDP** ที่ไม่มีการตรวจสอบความผิดพลาดในการส่งข้อมูล **IP** มีหน้าที่หลักในการกำหนดเส้นทางการส่งให้กับ **datagram** ซึ่งทำได้โดยการตรวจสอบแอดเดรสของผู้ส่งและผู้รับ และรับข้อมูลแอดเดรสปลายทางที่ติดมากับแต่ละ **datagram** แล้วนำไปเปรียบเทียบกับ **routing table** ซึ่งช่วยในการพิจารณาว่าจะส่ง **datagram** นั้นไปตามเส้นทางใด ตัวอย่างเช่นไปยังเครื่องคอมพิวเตอร์เครื่องใดเครื่องหนึ่งหรือไปยังระบบเครือข่ายอื่น นอกจากนั้นถ้าจำเป็นต้องแตก **datagram** ให้เป็นหน่วยที่เล็กลง งานในการแยก **datagram** และประกอบขึ้นมาใหม่ก็ตกเป็นหน้าที่ของโปรโตคอล **IP (Internet Protocol)** ซึ่งทำหน้าที่หาที่อยู่ปลายทางและเส้นทางที่ใช้ขนส่งข้อมูลใส่เข้าไปในส่วนหัวของแพ็กเก็ตข้อมูล เพื่อให้สามารถส่งไปถึงปลายทางได้ด้วยการใช้ **Dynamic Routing Table**

#### *5.4.3.2 ARP (Address Resolution Protocol)*

ก่อนที่ **IP packet** จะถูกส่งไปยัง **Host** อื่น จะต้องทราบที่อยู่ปลายทางโดยโปรโตคอล **ARP** จะทำการตรวจสอบ **MAC Address** ในหน่วยความจำแคชของเราท์เตอร์ก่อน ถ้าไม่พบแสดงว่าแอดเดรสปลายทางเป็นโฮสต์ใหม่ที่ยังไม่เคยส่งข้อมูลไปก่อน ดังนั้น **ARP** จะทำการส่งสัญญาณร้องขอแอดเดรสจากโฮสต์ต่างๆ บนระบบเครือข่าย หากตรวจพบว่ามีแอดเดรสตรงกับของตนเองจะทำการส่งแอดเดรสตอบกลับไปยังเครื่องคอมพิวเตอร์ที่ส่ง **ARP** ออกมา เพื่อนำไปบันทึกใน **Route table** ของเราท์เตอร์

#### *5.4.3.3 RARP (Reverse Address Resolution Protocol)*

โปรโตคอล **RARP** จะทำในทางกลับกันกับโปรโตคอล **ARP** คือ **RARP** จะให้หมายเลข **IP** แก่เครื่องที่ร้องขอ เมื่อ **RARP** ได้รับการร้องขอ **IP Address** จากโหนด จะทำการตรวจสอบ **IP Address** ใน **Route table** ของเราท์เตอร์ เพื่อทำการส่งค่า **IP Address** กลับไปยังเครื่องที่ร้องขอ และในทำนองเดียวกันถ้าไม่พบในหน่วยความจำแคชของเราท์เตอร์จะทำการส่งคำร้องขอออกไปบนระบบเครือข่าย เพื่อให้เครื่องปลายทางส่ง **IP Address** กลับมาให้

#### *5.4.3.4 ICPM (Internet Control Message Protocol)*

เป็นโปรโตคอลที่ถูกใช้ในการรับ-ส่ง สถานภาพในการขนส่งข้อมูล โดยทั่วไปเราท์เตอร์จะใช้ **ICPM** ในการควบคุมการไหลของกระแสข้อมูล หรือควบคุมอัตราเร็วในการขนส่งข้อมูลระหว่างเราท์เตอร์ด้วยกัน สำหรับข้อความในการรายงานสถานภาพของการขนส่งข้อมูลมีอยู่ 2 ชนิด คือรายงานความผิดพลาด **(Reporting Error)** และข้อมูลลำดับการส่ง **(Sending Query)**

### *5.4.4 Network Access Layer*

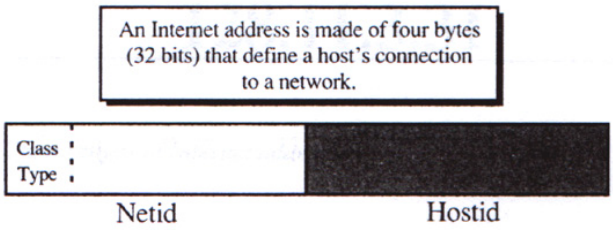
ประกอบด้วย **Data Link** และ **Physical Layer** ของ **OSI Model** เป็นการเชื่อมต่อทางกายภาพระหว่างระบบเครือข่ายที่มีสถาปัตยกรรมแตกต่างกัน เช่นระหว่างเครือข่ายโทเก้นริงกับเครือข่ายอีเธอร์เน็ต เป็นต้น เมื่อ **Network Access Layer** ขยายออกไปรวมกับการสื่อสารโดยทั่วไป ก็จะต้องมีโปรโตคอลที่เกี่ยวข้องกับโมเด็ม การเข้ารหัสข้อมูล การโยกย้ายไฟล์ และอื่นๆ อีกมาก

## 5.5 TCP/IP (Transmission Control Protocol/Internet Protocol)

**Transmission Control Protocol / Internet Protocol (TCP/IP)** เป็นชุดของโปรโตคอลตามมาตรฐานอุตสาหกรรมที่ได้รับการออกแบบมาสำหรับการสื่อสารในระบบเครือข่ายคอมพิวเตอร์ ในสภาวะแวดล้อมที่แตกต่างจากกัน และยิ่งไปกว่านั้น **TCP/IP** ยังเป็นโปรโตคอลที่สามารถกำหนดเส้นทางในการขนส่งข้อมูลได้ จึงถูกกำหนดให้เป็นโปรโตคอลมาตรฐานในการสื่อสารบนระบบอินเตอร์เน็ต ตามที่อธิบายไว้ข้างต้น

### *5.5.1 การกำหนดแอดเดรส (Addressing)*

การที่จะทราบได้ว่าเครื่องปลายทางอยู่ที่ใดในการติดต่อสื่อสารบนเครือข่ายอินเตอร์เน็ตจำเป็นที่จะต้องทราบที่อยู่ของเครื่องปลายทางว่าอยู่ที่ใด การกำหนดที่อยู่บนเครือข่ายอินเตอร์เน็ตทำโดยการใช้ **IP Address** โดยที่เครื่องทุกเครื่องที่ต่อมายังเครือข่ายอินเตอร์เน็ตจำเป็นต้องมี **IP Address** เป็นของตัวเอง และที่สำคัญ **IP Address** นั้นต้องไม่ซ้ำกับ **IP Address** ของเครื่องอื่นด้วย จึงสามารถสรุปได้ว่าเครื่องทุกเครื่องจะมีหมายเลย **IP Address** ที่ไม่เหมือนกันเลย และเป็นแอดเดรสที่แตกต่างจากแอดเดรสจริง (**MAC Address** ใน **NICs**) ที่สามารถชี้เฉพาะอุปกรณ์แต่ละชนิดเฉพาะเครื่อง ในการกำหนดแอดเดรสโดยใช้ **TCP/IP** นั้น แอดเดรสที่ถูกจัดตั้งขึ้นมาจะมีความยาว **32** บิต (**4** ไบต์) และประกอบด้วย **3** ฟิลด์คือ **Address Type, Network Identifier** และ **Host Identifier**




รูปที่ **5 – 5 TCP/IP Packet Format**

**Class Type** จะเป็นส่วนที่บอกว่า แอดเดรสที่ถูกกำหนดให้นั้นอยู่ในคลาส (**Class**) ใด

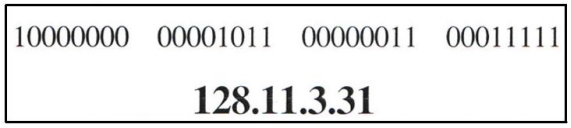
**Network Identifier** เป็นหมายเลขที่ระบุว่าเครื่องแม่ข่าย **TCP/IP** ถูกกำหนดให้อยู่ในระบบเครือข่ายทางกายภาพเดียวกัน หรือไม่ โดยเครื่องแม่ข่ายทั้งหมดที่อยู่ในระบบเครือข่ายเดียวกันจะมี **Network ID** หมายเลขเดียวกัน เพื่อให้สามารถสื่อสารซึ่งกันและกันได้

**Host Identifier** เป็นหมายเลขที่ระบุเครื่องแม่ข่ายภายในระบบเครือข่าย โดย **Host ID** จะต้องเป็นค่าโดยเฉพาะสำหรับเครื่องแม่ข่ายแต่ละเครื่อง ที่ได้รับการกำหนดโดย **Network ID** และ **IP Address** จะระบุตำแหน่งของส่วนประกอบต่างๆ บนระบบเครือข่าย ด้วยวิธีเดียวกันกับการกำหนดเลขที่ของบ้าน ที่ระบุว่าเป็นบ้านหลังใดในเมือง

### *5.5.2 Address Class*

**InterNIC (Internet Network Information Center)** เป็นองค์กรที่บริหารจัดการการใช้ **IP Address** ให้กับคอมพิวเตอร์ต่าง ๆ บนเครือข่ายอินเตอร์เน็ต ในปัจจุบันเราใช้ **IPv4** ซึ่งจะมีหมายเลขทั้งหมด **4,294,967,296** หมายเลข (**256 x 4** ชุด) แต่เมื่อ เก็บเข้าไว้ในเครื่องคอมพิวเตอร์จะเก็บเป็นรูปฐาน **2** สำหรับสำหรับรูปแบบของ **IP Address** ในมุมมองของผู้ใช้จะเห็นเป็นตัวเลข **4** ชุดที่คั่นกันด้วยจุด เช่น **255.255.255.0** เป็นต้น

และแต่ละชุดมี ตัวเลขอยู่ ระหว่าง **0 – 255** ในการสังเกตว่าแอดเดรสที่ได้รับนั้นอยู่ในคลาส (**Class**) ใด สามารถทำได้โดยการสังเกตที่บิตแรกว่าขึ้นต้นด้วยหมายเลขอะไร และเพื่อให้ง่ายต่อการใช้งาน **IP Address** จึงทำการจัดแบ่งตัวเลขประจำบิตต่างๆ ออกเป็น **4** กลุ่ม และแปลงให้เป็นเลขฐานสิบ (**Decimal**) ดังรูป




รูปที่ **5 – 6** การเปลี่ยน **IP Address** เป็นเลข **Decimal**

**Class A Addresses**

ในการกำหนดแอดเดรส หากบิตแรกของ **IP Address** เป็นเลข **0** แสดงว่าแอดเดรสที่ได้รับนั้นอยู่ใน **Class A** ซึ่งจากการที่มีแอดเดรสอยู่ใน **Class A** นั้น **7** บิตถัดมาจะถูกใช้ในการสร้างแอดเดรสของระบบเครือข่าย ซึ่งสามารถสร้างได้ถึง **128 Network** ส่วนบิตที่เหลือนั้นจะถูกกระจายเป็นแอดเดรสของเครื่องคอมพิวเตอร์นั่นเอง **Class A** เป็น **IP** ชุดแรกมีหมายเลข **IP** อยู่ที่ **1.0.0.1 – 126.255.255.254** โดย **1** ไบต์แรกจะเป็น **Network Address** และ **3** ไบต์ที่เหลือเป็น **Host Address** เหมาะสำหรับเครือข่ายขนาดใหญ่ เพราะสามารรองรับการใช้งานได้ **16** ล้านเครื่อง

**Class B Addresses**

ในการกำหนดแอดเดรส หาก **2** บิตแรกคือ **01** แสดงว่า **IP Address** นั้อยู่ใน **Class B** และ **14** บิตถัดมาจะถูกใช้เป็น **Physical Network** และ **16** บิตที่เหลือจะถูกแบ่งเป็นแอดเดรสย่อยได้อีก **Class B** มีหมายเลข **IP** อยู่ที่ **128.0.0.1-191.255.255.254** โดยแบ่ง **2** ไบต์แรกจะเป็น **Network Address** ส่วน **2** ไบต์ที่เหลือเป็น **Host Address** เหมาะสำหรับเครือข่ายขนาดใหญ่ แต่เล็กกว่า **Class A** เพราะสามารรองรับการใช้งานได้ **65,534** เครื่อง

**Class C Addresses**

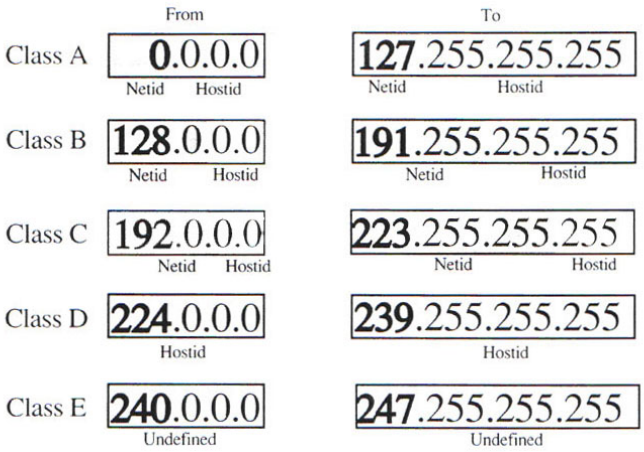
สำหรับ **Class C** นั้น **3** บิตแรกจะเป็น **110** และ **21** บิตถัดมาจะเป็น **Physical Network** ส่วน **8** บิตที่เหลือจะถูกใช้ในการกำหนดแอดเดรสย่อยต่อไป **Class C** มีหมายเลข **IP** อยู่ที่ **192.0.0.1-223.255.255.254** โดย **3** ไบต์แรกเป็น **Network Address** และ **1** ไบต์ที่เหลือเป็น **Host Address** เหมาะสำหรับเครือข่ายขนาดเล็ก แต่ละกลุ่มมี **256** หมายเลข และสามารรองรับจำนวนเครื่องได้ไม่เกิน **254** เครื่อง

**Class D Addresses**

ใน **Class D** จะขึ้นต้นด้วย **1110** และบิตที่เหลือจะถูกนำมาใช้ในการกำหนดแอดเดรสย่อยเลย **Class D** เป็น **IP** สำหรับ **Multiclass** กล่าวคือส่งแพ็กเก็ตข้อมูลกระจายให้กับกลุ่มคอมพิวเตอร์ ซึ่งประกอบไปด้วยหมายเลข **IP** ตั้งแต่ **225.0.0.0 – 239.255.255.255**

**Class E Addresses**

สำหรับ **Class E** ยังคงสำรองไว้สำหรับการใช้งานในอนาคต มีหมายเลข **IP** ตั้งแต่ **240.0.0.0 – 247.255.255.255**




รูปที่ 5 – 7 การกำหนดคลาสของแอดเดรสบนเครือข่ายอินเตอร์เน็ต

ในการใช้งานชุดโปรโตคอล **TCP/IP** นี้ ถ้าเป็นระบบเครือข่าย **WAN** จะสามารถใช้ได้บน **X.25, Frame Relay** และ **Switched Multi – Megabit Data Service (SMDS)** แต่ถ้าเป็นสภาพแวดล้อมแบบ **LAN** ก็จะเข้ากันได้ดีกับเครือข่ายอีเธอร์เน็ต ส่วนโปรโตคอลใน **Application Layer** ที่ **TCP/IP** รองรับได้เช่น **ISO File Transfer and Management (FTAM) , X.400** ซึ่งเป็นมาตรฐานในการทำ **message exchange** และ **X.500** ซึ่งเป็นมาตรฐานในการทำ **Directory Services**

### *5.5.3 Network Mask และ Subnet Mask*

**Network Mask** หรือ **Net Mask** เป็นการระบุว่า **IP Address** ที่เราใช้มีกี่บิต หรือกี่ไบต์ ที่เป็นส่วนของหมายเลขเครือข่าย อธิบายคร่าว ๆ ก็คือ หากจะใช้บิตใดเป็นหมายเลขเครือข่ายก็ตั้งค่าบิตนั้นของ **Net Mask** เป็น **1** ให้เสมือนเป็นหน้ากาก และบิตที่เหลือให้เป็น **0** ซึ่งก็คือจำนวนบิตที่ใช้ในส่วนของหมายเลขเครื่องนั้นเอง ซึ่งมีการกำหนด **Net Mask** ดังนี้

- **Class A Network Mask 255.0.0.0** ใช้ **1** ไบต์แรกเป็น **Network Address**
- **Class B Network Mask 255.255.0.0** ใช้ **2** ไบต์แรกเป็น **Network Address**
- **Class C Network Mask 255.255.255.0** ใช้ **3** ไบต์แรกเป็น **Network Address**

สำหรับ **Subnet Mask** จะใช้ในกรณีที่ต้องการขอยืมบางส่วนที่ใช้บริการในการกำหนด **Network Address** มาเพิ่มในการกำหนด **Host Address** เพิ่มเติม ทั้งนี้เพื่อให้เกิดมี **Subnet** เพิ่มเติม ยกตัวอย่างเช่นเครือข่ายใน **Class B** หมายเลข **128.1. x.x** (โดยที่ **x** มีค่า **1-254**) ดังนั้น **Network Mask** ก็คือ **255.255.0.0** สำหรับการหา **Subnet Mask** นั้น ยกตัวอย่างเช่น **IP Address** หมายเลข **193.127.6.0** จะสามารถหา **subnet** ได้ตามวิธีดังนี้

**1. IP Address 193.127.6.0**

**2. xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx** ต้องการหา **subnet** อย่างน้อย **16 host**

ดังนั้น **16 host = 24, Host Address = 4 bit**

**Network Address = 32 – 4 = 28 bit**

**Subnet Mask** = จำนวน **bit** ของ **network Address = 28 bit**

เขียนเป็น **11111111 .11111111. 11111111. 11110000**

3. จากตัวเลข $1-3$ ชุดแรก จะสามารถคำนวณแต่ละชุดได้เป็น

$= (1\times27)+(1\times26)+(1\times25)+(1\times24)+(1\times23)+(1\times22)+(1\times21)+(1\times20)=255$

5. จากตัวเลขชุดสุดท้ายสามารถคำนวณได้เป็น

$= (1\times27)+(1\times26)+(1\times25)+(1\times24)+(0\times23)+(0\times22)+(0\times21)+(0\times20)=240$

ดังนั้น **Subnet Mask** ของ **IP Address 193.127.6.0 0** จึงมีค่าเป็น = **255.255.255.240** นั้นเอง

เนื่องจากขณะนี้ชุดโปรโตคอล **TCP/IP** กำลังเผชิญปัญหาหลัก 2 ข้อคือ **IP address space** กำลังจะเต็ม เนื่องจากมีผู้นิยมใช้อินเตอร์เน็ตมาก ในขณะที่ **IP address** นี้จะไม่สามารถใช้ซ้ำกันได้ และอีกปัญหาหนึ่งคือ **routing table** โดยเฉพาะใน **backbone router** จะต้องใหญ่ขึ้นเรื่อยๆ เนื่องจากใช้ระบบ **flat address space** นั้นคือจะยิ่งต้องใช้เวลาในการทำ **routing** มากขึ้นดังนั้น **Internet Engineering Task Force (IETF)** จึงได้จัดทำกลุ่มวิจัยขึ้นมาเรียกว่า **IP - The Next Generation (IPng)** ซึ่งขณะนี้ได้เตรียมแนวทางแก้ไขปัญหาไว้ 2 ทางคือ **TCP & UDP with Bigger Address (TUBA)** โดยกำหนดโฮสต์เป็น 2 คลาส คือคลาสหนึ่งรับเฉพาะ **IP** อย่างเดียว และอีกคลาสหนึ่งเป็น **dual stacked host** ซึ่งจะรองรับทั้ง **IP** และ **ISO Connectionless Network Protocol (CLNP)** แต่อย่างไรก็ตามจะต้องไม่มีผลกระทบต่อ การใช้แอพพลิเคชั่นต่างๆ เช่น **telnet, SMTP** หรือ **FTP** สำหรับอีกแนวทางหนึ่งเรียกว่า **SIP-P** เป็นการรวมข้อเสนอ 2 ข้อคือ "**P**" **Internet Protocol (PIP)** และ **Simple IP (SIP)** รวมทั้งกำลังมีการพัฒนา **IPv6** ขึ้นมาใช้งานแทน **IPv4** ซึ่งใช้งานอยู่ในปัจจุบันเพื่อแก้ปัญหา **IP Address** ไม่เพียงพอต่อความต้องการ

### 5.6 NetBIOS

**NetBIOS** เป็นโปรโตคอลที่ได้รับการออกแบบให้เป็นโปรโตคอลตัวเชื่อม (**interface**) ระหว่างฮาร์ดแวร์กับซอฟต์แวร์ระบบปฏิบัติการ เพื่อให้โปรแกรมประยุกต์สามารถสื่อสารกับเครือข่ายได้โดยเป็นอิสระจากฮาร์ดแวร์ ทั้งนี้โปรแกรมประยุกต์ต่างๆ จะสามารถเข้าถึงเลเยอร์สูงสุดของ **OSI model** ได้เท่านั้น ซึ่งทำให้โปรแกรมประยุกต์ที่สร้างขึ้นมาสามารถทำงานได้ในเครือข่ายที่มีสภาพแวดล้อมของระบบเครือข่ายไม่เหมือนกัน ทั้งนี้ **NetBIOS** จะทำหน้าที่ขนส่งข้อมูลไปยังโปรแกรมประยุกต์ที่อยู่บนเครื่องอื่นในเครือข่ายให้ ในช่วงเริ่มต้นนั้น **NetBIOS** ได้รับการออกแบบให้ทำงานได้กับเครื่องคอมพิวเตอร์ **PC** ของ **IBM** ในระบบเครือข่าย **LAN** เท่านั้น แต่ปัจจุบัน **NetBIOS** ได้กลายเป็นพื้นฐานการทำงานของโปรแกรมประยุกต์บนระบบเครือข่ายไปแล้ว โดย **NetBIOS** เป็นโปรโตคอลที่ถูกใช้งานอย่างแพร่หลาย และสามารถทำงานได้ทั้งบนระบบเครือข่ายอีเธอร์เน็ตและเครือข่ายโทเก้นริง

**NetBIOS** ได้รับการออกแบบมาให้เป็นตัวเชื่อม โดยเป็นส่วนขยายของ **BIOS** ที่ช่วยให้สามารถติดต่อใช้งานบริการบนเครือข่ายได้ จึงกล่าวได้ว่า **NetBIOS** ถูกออกแบบให้เป็น **Application Program Interface (API)** ในขณะเดียวกัน **NetBIOS** ก็ถือว่าเป็นโปรโตคอลได้เช่นเดียวกันกับ **TCP/IP** เพราะมีชุดของโปรโตคอลชั้นล่างลงไปที่สามารถทำงานร่วมกันได้ แรกเริ่ม **NetBIOS** ถูกออกแบบให้ทำงานกับเครือข่ายขนาดเล็กที่เป็นเครือข่ายระดับท้องถิ่น ดังนั้น **NetBIOS** จึงถูกออกแบบให้ทำงานร่วมกับ **NetBEUI (NetBIOS Extended User Interface)** ซึ่งเป็น **network-transport protocol** ดังนี้

NetBIOS over NetBEUI

| Layer | Protocol | Description |
|---|---|---|
| 7 Application | Redirector | เป็นตัว redirect คำสั่งให้ออกไปยังเป้าหมายผ่านทางเครือข่าย |
| 6 Presentation | SMB | Server Message Blocks ช่วยให้สามารถทำ file sharing, print sharing และ user-based messaging |
| 5 Session | NetBIOS | ให้บริการ name service, datagram service และ session service (อ่านรายละเอียดในหัวข้อ NetBIOS service) |
| 4 Transport | NetBEUI | ให้บริการขนส่งข้อมูล |
| 3 Network | | |
| 2 Data link | NIC Driver, NDIS | |
| 1 Physical | NIC Adapter | |

### *5.6.1 NetBIOS name*

การที่โหนดแต่ละโหนดจะสื่อสารกันได้จะต้องมี NetBIOS name ที่ไม่ซ้ำกันในเครือข่าย ซึ่งโหนดอาจจะหมายถึงเครื่องคอมพิวเตอร์ เราท์เตอร์ เครื่องพิมพ์ โดย NetBIOS Name จะมีชื่อยาวได้สูงสุด 16 ไบต์ หรือ 16 ตัวอักษร แต่สำหรับระบบปฏิบัติการของไมโครซอฟต์ สามารถตั้ง NetBIOS name ได้สูงสุดเพียง 15 ตัวอักษรเท่านั้น เพราะไบต์ที่ 16 จะถูกนำไปใช้เพื่อบ่งบอกชนิดของ NetBIOS name นั้นๆ เช่น domain name, group name, computer name หรืออื่นๆ NetBIOS name โดยทั่วไป สามารถแบ่งออกได้เป็น 2 ประเภทคือ

- Unique name คือชื่อที่ไม่สามารถซ้ำกันได้ในเครือข่ายวงเดียวกัน โดยส่วนใหญ่จะเป็น Computer name เช่น MYMACHINE
- Group name คือชื่อของโดเมนหรือเวิร์คกรุ๊ป (workgroup) ที่เครื่องนั้นๆ สังกัดอยู่ เช่น MYWORKGROUP

เนื่องจาก NetBIOS ทำงานบน Session Layer ซึ่งอยู่เหนือ Network Layer ดังนั้น NetBIOS จะไม่มีข้อมูลที่เกี่ยวข้องกับแอดเดรสบนระบบเครือข่าย เช่น ข้อมูล IP ของ NetBIOS name

### *5.6.2 วิธีการส่งข้อมูลของ NetBIOS*

NetBIOS ได้รับการออกแบบให้ทำงานกับกลุ่มของเครื่องคอมพิวเตอร์ที่ใช้สื่อการแพร่กระจายข้อมูลเดียวกัน ซึ่งสามารถทำงานได้ทั้งแบบ connection-oriented และแบบ connectionless นอกจากนี้ยังสนับสนุนการทำงานแบบ broadcast และ multicast อีกด้วย เมื่อโหนดต้องการสื่อสารผ่านเครือข่าย สามารถทำได้ 2 วิธี คือ

#### *5.6.2.1 Session mode*

เป็นการรับส่งข้อมูลที่มีขนาดใหญ่มีระบบตรวจสอบข้อผิดพลาดและการกู้คืน (recovery) แต่มีข้อเสียที่เป็นการสื่อสารแบบ 1 ต่อ 1 เท่านั้น ขนาดของข้อมูลสามารถขยายได้สูงสุดถึง 64 Kbytes นอกจากนี้ยังมี NetBIOS session control command และ NetBIOS session data transfer command ซึ่งช่วยให้สามารถทำการสื่อสารผ่านการจัดตั้งการสนทนาได้ (connection-oriented connection)

*5.6.2.2 Datagram mode (รวมถึงการส่งข้อมูลแบบ broadcast)*

สามารถรับส่งข้อมูลได้เร็วที่สุด แต่ไม่มีการรับประกันว่าข้อมูลจะไปถึงปลายทางหรือไม่ และไม่สามารถส่งข้อมูลซ้ำในกรณีที่ข้อมูลไปไม่ถึงปลายทางได้ โดยปกติจะมีขนาด **512** ไบต์ แต่สามารถสื่อสารกับเครื่องคอมพิวเตอร์อื่นได้หลายเครื่องในเวลาเดียวกัน (**connectionless connection**)

***5.6.3 NetBIOS Service***

หลายคนอาจจะเคยใช้งานบริการ **Browsing** (ผ่านทาง **network neighborhood**), **domain authentication, trust, file sharing** หรือ **printer sharing** ซึ่งโปรแกรมประยุกต์เหล่านี้ล้วนทำงานได้โดยใช้พื้นฐานจาก บริการพื้นฐานของ **NetBIOS** มี **3** ชนิดด้วยกัน คือ

*5.6.3.1 Name Service*

เป็นบริการที่ช่วยให้ **NetBIOS node** สามารถลงทะเบียน **NetBIOS name** ที่เป็นเอกลักษณ์ (**unique name** หรือ **group name**) ในระบบเครือข่ายได้ เช่น เมื่อเครื่องคอมพิวเตอร์เปิดเครื่องขึ้นมาใหม่ มันจะส่ง **ADD NAME QUERY** ออกไปเพื่อตรวจสอบว่ามีเครื่องคอมพิวเตอร์หรือ **NetBIOS node** อื่นใดใช้ชื่อ **NetBIOS name** ซ้ำกันหรือไม่ ซึ่งโดยปกติแล้วหากไม่มีการตอบกลับ (**response**) กลับมาภายหลังการส่งดิวรีดังกล่าวออกไปแล้วหกครั้งใน **0.5** วินาที ถือได้ว่า **NetBIOS name** ดังกล่าวไม่ซ้ำกับใคร (ในขณะนั้น) สามารถใช้งานเป็น **NetBIOS name** ได้ ทั้งนี้บริการ **Name service** นี้สามารถตรวจสอบ **NetBIOS name** ได้ทั้ง **unique name** และ **Group name** ด้วย

*5.6.3.2 Session Service*

เป็นบริการที่ใช้การสื่อสารแบบ **connection-oriented** ซึ่งมีความเชื่อถือได้ และเป็นการสื่อสารแบบ **full-duplex** ทั้งนี้ **NetBIOS** ต้องการอย่างน้อย **1** โพรเซส เพื่อทำหน้าที่เป็นไคลเอนต์ และตัวอื่นๆ เป็นเซิร์ฟเวอร์ การที่จะจัดตั้งการสนทนาได้จะต้องมีการเตรียมการทั้งสองฝ่าย ฝ่ายแรกจะเป็นตัวผู้รับ (**Listen**) ในขณะที่อีกฝ่ายจะเป็นฝ่ายผู้เรียก (**Call**) ทั้งนี้ฝ่ายผู้รับจะอ้างอิงชื่อจากตาราง **NetBIOS name** ของตัวเองและยังต้องตรวจสอบชื่อของฝั่งตรงข้ามอีกด้วย ถ้าผู้ฝ่ายรับไม่พร้อมที่จะรับการติดต่อ การติดต่อที่เกิดขึ้นจากฝ่ายผู้เรียกก็จะล้มเหลว หากสามารถจัดตั้งการสนทนาได้สำเร็จ โปรแกรมประยุกต์ของทั้งสองฝ่ายจะได้รับข้อมูล **session-id** จากนั้นโปรแกรมประยุกต์จะเริ่มการรับส่งข้อมูล และในตอนท้ายของการติดต่อทั้งสองฝ่ายสามารถเป็นฝ่ายส่งคำสั่ง **Hang-Up** เพื่อยกเลิกการเชื่อมต่อได้ การสื่อสารแบบ **Connection-Oriented** นี้ไม่มีการควบคุมอัตราการขนส่งข้อมูล เพราะถือว่า **LAN** มีความเร็วสูงเพียงพอที่จะขนส่งข้อมูลได้โดยไม่มีปัญหา

*5.6.3.3 Datagram Service*

เป็นการสื่อสารที่สามารถส่งข้อมูลไปยังเครื่องที่ระบุหรือส่งไปยังทุกเครื่องในกลุ่มที่ระบุหรือแพร่กระจายข้อมูลไปยังวง **LAN** ได้ บริการนี้ใช้การสื่อสารแบบ **connectionless** เช่นเดียวกันกับการรับส่งข้อมูลแบบ **Datagram** แบบอื่นๆ เช่น **UDP/IP** โดยผู้ส่งจะใช้คำสั่ง **Send_Datagram** ซึ่งต้องระบุผู้รับปลายทางด้วย ซึ่งอาจจะเป็นกลุ่ม หรือ **NetBIOS node** เดี่ยวๆ ก็ได้ ส่วนทางด้านผู้ที่เรียกใช้คำสั่ง **Receive_Datagram** นั้นจะต้องระบุ **local name** หรือชื่อของปลายทางที่ต้องการรับ นอกจากนี้ยังมีคำสั่ง **Send_Broadcast_Datagram**

ซึ่งจะส่งข้อความไปยังทุกเครื่องใน LAN ซึ่งหากมีโพรเซสที่รันคำสั่ง Receive_Broadcast_Datagram ไว้ก็จะได้รับ datagram ที่ถูกส่งออกมานั้นไป

### *5.6.4 NetBIOS Encapsulation*

โดยปกติแล้ว NetBIOS ทำงานได้เป็นอย่างดีบนโปรโตคอล NetBEUI แต่เพื่อให้ NetBIOS สามารถทำงานข้ามเครือข่ายได้ จึงได้มีการนำ NetBIOS ไปใช้งานบน routable protocol อื่น เช่น TCP/IP และ IPX/SPX ดังนั้นเพื่อให้แน่ใจได้ว่า NetBIOS สามารถทำงานได้บน TCP/IP และ IPX/SPX ได้ดีเช่นเดียวกับการรัน NetBIOS บน NetBEUI หรือ NBF จึงได้มีการนำ encapsulation มาใช้ ดังนี้

#### *5.6.4.1 NetBIOS over IPX/SPX*

IPX เป็นโปรโตคอลที่พัฒนาขึ้นโดย บริษัท Novell และได้มีการเผยแพร่การใช้งาน NetBIOS over IPX ในปี 1986 ตารางด้านล่างแสดงกลไกการทำงานของ NetBIOS over IPX (ในหัวข้อนี้ไม่ได้เจาะลึกในรายละเอียดของ NetBIOS over IPX)

NetBIOS over IPX

<table>
<tr><th>Layer</th><th>Protocol</th><th>Description</th></tr>
<tr><td>7 Application</td><td rowspan="2">Higher level protocols e.g. SMB / CIFS</td><td>e.g. Browser Service</td></tr>
<tr><td>6 Presentation</td><td></td></tr>
<tr><td>5 Session</td><td rowspan="2"></td><td rowspan="3">Session Management Protocol</td></tr>
<tr><td>4 Transport</td></tr>
<tr><td rowspan="2">3 Network</td><td>User Datagram Protocol, Name Management Protocol, NetBIOS Diagnostic and Monitoring Protocol</td></tr>
<tr><td colspan="2">IPX</td></tr>
<tr><td>2 Data link</td><td colspan="2">e.g. IEEE 802.2</td></tr>
<tr><td>1 Physical</td><td colspan="2">Token Ring / Ethernet etc</td></tr>
</table>

#### *5.6.4.2 NetBIOS over TCP/IP (NBT)*

ภายหลังจากที่เครือข่ายอินเตอร์เน็ตได้รับความนิยม มีผู้นำระบบเข้ามาเชื่อมต่อจำนวนมาก และเพื่อให้การใช้งานเป็นไปในแนวทางเดียวกัน จึงได้มีการเผยแพร่ RFC (request for comment) 2 ฉบับ ดังนี้

1. RFC 1001 (PROTOCOL STANDARD FOR A NetBIOS SERVICE ON A TCP/UDP TRANSPORT: CONCEPTS AND METHODS)
2. RFC 1002 (PROTOCOL STANDARD FOR A NetBIOS SERVICE ON A TCP/UDP TRANSPORT: DETAILED SPECIFICATIONS)

ทั้งนี้เพื่อกำหนดมาตรฐานในการนำ NetBIOS มาใช้งานบนระบบเครือข่าย TCP/IP ให้สามารถใช้งาน NetBIOS service ได้ครบ โดยพยายามให้มีการเปลี่ยนแปลงให้น้อยที่สุดและสามารถเข้ากับมาตรฐานเดิมได้ และยังทำงานได้อย่างยืดหยุ่นและมีประสิทธิภาพ โดยไม่จำเป็นต้องมีศูนย์กลางในการควบคุม และสามารถทำงานได้โดยไม่จำเป็นต้องใช้สิ่งอำนวยความสะดวกอื่นใดเพิ่มเติม

มีการใช้งาน **NetBIOS over TCP/IP** ที่เห็นได้ชัดเจนคือ ระบบปฏิบัติการ **Windows** ของไมโครซอฟต์ และ **Samba** ที่สามารถทำงานได้บนยูนิกซ์และลีนุกซ์ ทั้งนี้การนำไปใช้ของทั้งสองค่ายอาจจะแตกต่างไปจาก **RFC** ที่ได้กำหนดไว้บ้าง แต่ก็ดำเนินไปในแนวทางคล้ายๆ กัน อย่างไรก็ตามในที่นี้จะเน้นถึงการอิมพลีเมนต์ **NetBIOS over TCP/IP** ของไมโครซอฟต์มากกว่า เพราะมีการนำไปใช้งานมากกว่า

**NetBIOS over TCP/IP**

| Layer | Protocol | | Description |
|---|---|---|---|
| 7 Application | Higher level protocols e.g. SMB/CIFS | | e.g. Browser Service |
| 6 Presentation | | | |
| 5 Session | Name Service | datagram service | Session Service |
| 4 Transport | UDP, TCP | | |
| 3 Network | IP | | |
| 2 Datalink | e.g. IEEE 802.2 | | |
| 1 Physical | Token Ring / Ethernet etc | | |

### *5.6.5 NetBIOS Service over TCP/IP*

บริการของ **NetBIOS** บน **TCP/IP** มีด้วยกัน **3** บริการเหมือนกับ **NetBIOS** ที่รันบน **NBF** ซึ่งมีรายละเอียดเพิ่มเติมของแต่ละบริการดังนี้

#### *5.6.5.1 Name Service*

ให้บริการลงทะเบียนและยกเลิกการใช้งาน **NetBIOS name** ภายในระบบเครือข่ายที่อยู่ในเซ็กเมนต์เดียวกัน โดยใช้โปรโตคอล **UDP** พอร์ต **137 (broadcast packet)** ทั้งนี้ **Name service** สามารถใช้ได้ในวง **LAN** เท่านั้นเนื่องจากเราท์เตอร์ส่วนใหญ่จะถูกตั้งค่าให้ไม่อนุญาตให้แพ็กเก็ต **UDP** ที่แพร่กระจายออกมา ผ่านไปได้ การที่บริการนี้ทำงานบนโปรโตคอล **UDP** ทำให้มีข้อดีตรงที่ส่วนหัว **(header)** ของแพ็กเก็ตมีขนาดเล็กและใช้เวลาในการสื่อสารน้อยกว่าการใช้โปรโตคอล **TCP** ส่วนข้อเสียก็คือเมื่อส่งข้อมูลออกไปแล้วจะไม่สามารถรู้ได้เลยว่าข้อมูลถูกส่งออกไปถึงเป้าหมายที่ต้องการจริงหรือไม่

#### *5.6.5.2 Datagram Service*

เป็นบริการที่ทำให้สามารถสื่อสารกับโหนดอื่นๆ ได้ โดยสามารถส่งข้อมูลได้ทั้งแบบ **connectionless** และแบบ **broadcast** โดยใช้โปรโตคอล **UDP** พอร์ต **138** ตัวอย่างการนำไปใช้งานที่เห็นได้ชัดเจนคือโปรแกรม **browser** ซึ่งจะถูกเรียกใช้เมื่อผู้ใช้รัน **network neighborhood** จากเดสก์ท็อปของวินโดวส์ ซึ่ง **browser service** จะเรียกใช้งาน **datagram service** โดยการแพร่กระจายข้อความออกไป ทั้งนี้ **Datagram service** ซึ่งรันอยู่บน **UDP** นั้นก็มีข้อดีและข้อเสียเหมือนกับ **Name service** เช่นเดียวกัน

#### *5.6.5.3 Session Service*

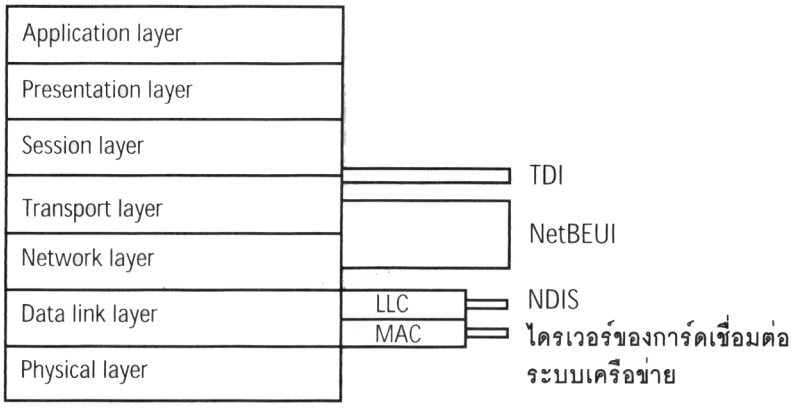
เป็นบริการที่ให้การเชื่อมต่อแบบ **Connection-oriented** โดยใช้โปรโตคอล **TCP** พอร์ต **139** ตัวอย่างการใช้งานที่เห็นได้ชัดเจนคือ **file sharing, printer sharing** นอกจากนี้ยังมี **network application**

ของ **Windows** ทำงานโดยอาศัยบริการนี้ เช่น **Server Manager, Event Viewer, Register Editor** และ **Performance Monitor**

**Session service** มีความซับซ้อนมากกว่า **name** หรือ **datagram service** เพราะ **session service** ทำงานบนพอร์ต **TCP** ซึ่งต้องมีการจัดตั้งการเชื่อมต่อ ตรวจสอบความถูกต้องของเครื่องคอมพิวเตอร์ในระบบ ตรวจสอบความถูกต้องของผู้ใช้ และยกเลิกการเชื่อมต่อ อย่างไรก็ตามหากกล่าวถึงบริการ (**service**) หรือแอพพลิเคชั่น (**application**) ที่ทำงานอยู่ในเลเยอร์ที่สูงกว่า **NetBIOS** มักจะพบว่าบริการดังกล่าวมักจะอาศัยบริการพื้นฐานทั้ง **3** อย่างของ **NetBIOS** จึงจะสามารถทำงานได้อย่างสมบูรณ์ เช่น **Messenger Service** ซึ่งสามารถใช้งานได้โดยการใช้คำสั่ง **net send ‹destination› message** ซึ่งสามารถระบุปลายทางเป็นชื่อผู้ใช้ (**user**) เครื่องคอมพิวเตอร์หนึ่งเครื่องหรือทุกเครื่องในโดเมนก็ได้ หากเป้าหมายเป็นเครื่องคอมพิวเตอร์ **1** เครื่อง **NetBIOS name service** ก็จะแพร่กระจายข้อความออกไปเพื่อค้นหาเครื่องคอมพิวเตอร์ดังกล่าว หรือในกรณีที่ปลายทางเป็นผู้ใช้ **NetBIOS name service** ก็จะแพร่กระจายข้อความออกไปทั้งเครือข่ายเพื่อค้นหาผู้ใช้ซึ่งมีชื่อตามที่ระบุ (ไบต์ที่ **16** มีค่าเป็น **‹03h›**) หลังจากนั้นจะเป็นขั้นตอนการส่งข้อความที่ต้องการส่ง หากปลายทางเป็นกลุ่มของเครื่องคอมพิวเตอร์ **NetBIOS datagram service** จะเป็นตัวส่งแพร่กระจายข้อความออกไปทั้งเครือข่าย แต่ถ้าเป็นเครื่องคอมพิวเตอร์เครื่องเดียว **NetBIOS session service** จะเชื่อมต่อไปหาเครื่องคอมพิวเตอร์นั้นโดยตรง

## 5.7 NetBEUI (Network Basic End User Interface)

โปรโตคอล **NetBEUI** เป็นมาตรฐานของการสื่อสารข้อมูลที่ออกแบบขึ้นมาอย่างพิเศษ โดยการร่วมมือกันระหว่างไมโครซอฟต์ และไอบีเอ็ม เพื่อใช้ในการสื่อสารขอมูลในระบบเครือข่ายที่เป็นเครื่องไมโครคอมพิวเตอร์ของไอบีเอ็ม ซึ่งเริ่มต้นใช้ในปี **1985** โดย **NetBEUI** เป็นโปรโตคอลที่ไม่มีส่วนในการระบุเส้นทางส่งผ่านข้อมูล (**Non-routable Protocol**) โดยจะใช้วิธีการแพร่กระจายข้อมูลออกไปในเครือข่าย และหากใครเป็นผู้รับที่ถูกต้องก็จะนำข้อมูลที่ได้รับไปประมวลผล ข้อจำกัดของโปรโตคอลนี้คือ ไม่สามารถทำการแพร่กระจายข้อมูลข้ามไปยังเซ็กเมนต์อื่น ๆ ที่ไม่ใช่เซ็กเมนต์เดียวกันได้ เนื่องจากอุปกรณ์ระบบเครือข่าย เช่นเราท์เตอร์ไม่สามารถจะแพร่กระจายข้อมูลออกไปยังเครือข่ายอื่นๆได้ เพราะถ้าหากยอมให้การสื่อสารระหว่างเครือข่ายเต็มไปด้วยข้อมูลที่เกิดจากการ **Broadcast** จนทำให้เครือข่ายต่าง ๆ ไม่สามารถติดต่อสื่อสารกันได้อย่างมีประสิทธิภาพ โปรโตคอล **NetBEUI** จึงเหมาะที่จะใช้งานบนเครือข่ายขนาดเล็กที่มีเครื่องคอมพิวเตอร์ไม่เกิน **50** เครื่องเท่านั้น




รูปที่ **5 – 8** การทำงานของ **NetBEUI** ในโครงสร้าง **OSI Model**

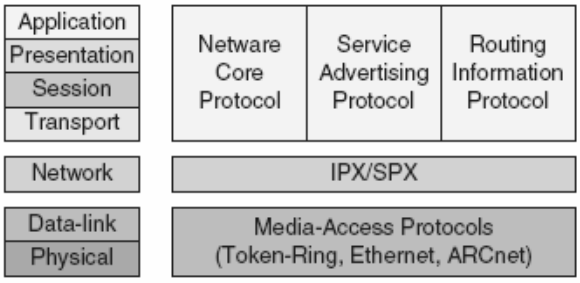
**NetBEUI** ถูกพัฒนาขึ้นโดยบริษัท **IBM** เพื่อเป็นโปรโตคอลระบบเครือข่ายของเครื่อง **PC** และไมโครซอฟต์ได้นำมาใช้ในผลิตภัณฑ์หลายตัวด้วยกัน โดย **NetBEUI** ทำงานอยู่บน **Data-link layer** และเนื่องจากโปรโตคอลใน **data-link layer** เป็นโปรโตคอลที่ไม่มีความสามารถในการกำหนดเส้นทางการขนส่งข้อมูล **(nonroutable protocol)** ดังนั้น **NetBEUI** ก็เป็น **nonroutable protocol** ด้วยเช่นกัน ซึ่งถือว่าเป็นข้อจำกัดของโปรโตคอล **NetBEUI**

**NetBEUI** สามารถทำงานได้เป็นอย่างดีและทำงานได้เร็วกับเครือข่ายขนาดเล็กที่มีจำนวนเครื่องคอมพิวเตอร์ตั้งแต่ **20 – 200** เครื่อง และยังสามารถทำงานข้ามเซ็กเมนต์ของ **LAN** ได้ แต่ต้องมีเกตเวย์เป็นตัวควบคุมเซ็กเมนต์อีกที ในความเป็นจริงแล้ว ไม่อาจเรียก **NetBEUI** เวอร์ชัน **3.0** ว่าเป็นโปรโตคอล **NetBEUI** ได้เต็มที่นัก แต่ถือว่าเป็น **NetBIOS Frame (NBF) format** เสียมากกว่า เพราะ **NetBEUI** จะใช้งาน **NetBIOS interface** และ **interface** อื่นที่อยู่สูงกว่า แต่ **NBF** นำ **Transport Driver Interface (TDI)** มาใช้งานแทน ซึ่ง **NBF** ก็สามารถทำงานร่วมกันและเข้ากันได้ดีกับ **NetBEUI** ที่ไมโครซอฟต์ได้นำไปใช้งานในผลิตภัณฑ์ตัวก่อนหน้านี้ และเนื่องจาก **NetBEUI** ก็เป็น **nonroutable protocol** ซึ่งไม่สามารถส่งต่อแพ็กเก็ตข้อมูลผ่าน **routed network** ได้ แต่ **NetBIOS** ก็สามารถทำงานร่วมกับ **routable protocol** ตัวอื่นๆ ได้ เช่น **IPX** และ **TCP/IP**

เมื่อ **NetBEUI** เป็นโปรโตคอลที่ทำงานได้ดีกว่าโปรโตคอลตัวอื่นใน **LAN** แต่ทำงานได้แย่มากสำหรับ **WAN** จึงมีการแนะนำให้ใช้ทั้ง **NetBEUI** และ **TCP/IP** ใน **Windows NT** เป็นต้นไป ทั้งนี้จะต้องมีการติดตั้ง **NetBEUI** ในทั้ง **2** ฝั่งของการสื่อสาร และตั้งค่าให้ **NetBEUI** เป็นโปรโตคอลแรกที่จะถูกเรียกใช้ (ให้ลำดับความสำคัญมากกว่า **TCP/IP**) โดย **Windows NT** จะเลือกใช้ **NetBEUI** สำหรับการสื่อสารภายในเซ็กเมนต์ของระบบเครือข่าย **LAN** และใช้ **TCP/IP** สำหรับการสื่อสารไปยังเราท์เตอร์หรือ **WAN** ส่วนอื่นๆ

## 5.8 Netware Protocol

บริษัท **Novell** ได้พัฒนาโปรโตคอลสำหรับใช้กับระบบเครือข่าย **Netware** ในทำนองเดียวกับ **TCP/IP** ได้แก่ **Media Access Control, IPX/SPX (Internetwork Packet Exchanger/Sequence Packet Exchanger), RIP (Routing Implement Protocol), SAP (Service Advertising)** และ **NCP (Netware Cone Protocol)** แต่เนื่องจากโปรโตคอลเหล่านี้ มีใช้ก่อนที่จะมีการกำหนดมาตรฐาน **OSI** จึงมีบางส่วนไม่ค่อยตรงกับ **OSI Model** รูปที่ **5 – 9** แสดงให้เห็นการเปรียบเทียบโปรโตคอลของ **NetWare** กับ **OSI Model**



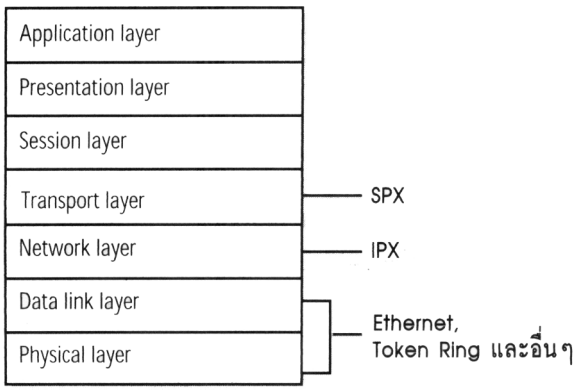

รูปที่ **5 – 9** เปรียบเทียบ **NetWare** กับ **OSI Reference Model**

### *5.8.1 Media Access Control*

โปรโตคอลนี้จะทำการกำหนดแอดเดรสให้แต่ละโหนดในระบบเครือข่าย **NetWare** โดยสร้างเป็นแอดเดรสในการ์ดเชื่อมต่อระบบเครือข่าย โปรโคอลนี้มีหน้าที่ในการจัดเก็บส่วนหัวของแพ็กเก็ตข้อมูล ซึ่งเป็นแอดเดรสต้นทางและแอดเดรสปลายทาง เมื่อแพ็กเก็ตข้อมูลถูกส่งออกไปยังเครื่องคอมพิวเตอร์ต่างๆ ในระบบเครือข่าย เครื่องคอมพิวเตอร์แต่ละเครื่องจะทำการตรวจสอบว่าแอดเดรสนั้นเป็นแอดเดรสของตนเองหรือไม่ หรือในกรณีที่ต้องการแพร่กระจายข้อมูล การ์ดเชื่อมต่อระบบเครือข่ายจะทำการคัดลอกข้อมูลนั้นส่งให้ **Protocol Stack**

### *5.8.2 IPX/SPX (Internetwork Packet Exchange and Sequence Packet Exchange)*

เป็นโปรโตคอลที่ออกแบบโดยบริษัท **Novell** ซึ่งพัฒนามาจากโปรโตคอล **XNS (Xerox Network System)** ของบริษัท **Xerox** โปรโตคอล **IPX (Internetwork Packet Exchange)** เป็นโปรโตคอลที่ทำงานอยู่ใน **Network Layer** ใช้จัดการการแลกเปลี่ยนแพ็กเก็ตภายในเครือข่ายทั้งในส่วนของการหาปลายทางและการจัดส่งแพ็กเก็ต โปรโตคอล **IPX** มีลักษณะการเชื่อมต่อแบบ **Connectionless** จึงไม่ค่อยมีความน่าเชื่อถือ ส่วน **SPX (Sequenced Packet Exchange)** จะเป็นโปรโตคอลที่ทำงานอยู่ใน **Transport Layer** โดยมีหน้าที่ช่วยในการจัดการรักษาความปลอดภัยให้กับข้อมูล และเพิ่มความน่าเชื่อถือให้กับโปรโตคอล **IPX** โดยจัดการให้ส่งข้อมูลไปถึงจุดหมายได้อย่างแน่นอน




รูปที่ **5 – 10** การทำงานของ **IPX/SPX** ในโครงสร้าง **OSI Model**

**Novell** ได้ใช้โปรโตคอลในโครงสร้าง **XNS (Xerox Network System)** ในการปรับปรุง **Internet Datagram** ของ **IPX** ให้มีการจัดเก็บแอดเดรสใน **2** รูปแบบ คือ

- **Internetwork Addressing** แอดเดรสของกลุ่มเครื่องคอมพิวเตอร์ในระบบเครือข่าย ถูกกำหนดโดยหมายเลขเครือข่ายที่ระบุให้ในขณะทำการติดตั้ง
- **Internode Addressing** แอดเดรสของบริการภายในโหนด ถูกกำหนดโดยหมายเลข **Socket**

### *5.8.3 RIP (Routing Information Protocol)*

โปรโตคอล **RIP** ช่วยในการแลกเปลี่ยนข้อมูลในระบบเครือข่าย **NetWare** เป็นโปรโตคอลที่ได้รับการพัฒนาบนระบบ **XNS** เช่นเดียวกับ **IPX** แต่ในการใช้ **RIP** จะมีการเพิ่มข้อมูลบางฟิลด์เข้าไปในแพ็กเก็ตเพื่อช่วยในการเลือกเส้นทางในการขนส่งข้อมูล ในการ **broadcast** ของโปรโตคอล **RIP** จะเกิดสิ่งต่างๆ ดังต่อไปนี้

- เครื่องเวิร์กสเตชั่นสามารถค้นหาเส้นทางในการขนส่งข้อมูลที่เร็วที่สุดได้
- เราท์เตอร์สามารถร้องขอข้อมูลจากเราท์เตอร์ตัวอื่นๆ เพื่ออัพเดทข้อมูล **Route Table** ให้ทันสมัยอยู่ตลอดเวลา
- เราท์เตอร์สามารถตอบสนองการร้องขอข้อมูลจากเครื่องเวิร์คสเตชั่น และเราท์เตอร์ตัวอื่นๆ ได้
- เราท์เตอร์มั่นใจได้ว่าสามารถติดต่อถึงกันได้
- เราท์เตอร์สามารถตรวจพบความเปลี่ยนแปลงโครงสร้างในระบบเครือข่าย

### *5.8.4 SAP (Service Advertising Protocol)*

โปรโตคอล **SAP** อนุญาตให้โหนดที่ให้บริการ เช่น **File Service, Print Service, Gateway Service** และ **Application Service** สามารถประกาศการให้บริการเหล่านั้นพร้อมทั้งระบุแอดเดรสของโหนดที่ให้บริการ ออกไปบนระบบเครือข่าย ทำให้เครื่องลูกข่ายสามารถ **access** เข้าไปยังทรัพยากรระบบเครือข่ายเหล่านั้นได้ และจากการใช้โปรโตคอล **SAP** ทำให้สามารถทำการเพิ่มหรือลดส่วนของการให้บริการได้อย่างคล่องตัว โดยปกติแล้ว **SAP Server** จะทำการบอร์ดคาสต์ข้อมูลเหล่านี้ออกไปทุกๆ **60** วินาที โดยแพ็กเก็ตของ **SAP** จะประกอบด้วย

- **Operating Information** ทำให้ทราบถึงกิจกรรมที่แพ็กเก็ตกำลังทำ
- **Service Type** ทำให้ทราบชนิดของบริการที่ให้โดยเครื่องเซิร์ฟเวอร์
- **Service Name** ทำให้ทราบชื่อของเครื่องเซิร์ฟเวอร์ที่ให้บริการ
- **Network Address** ระบุจำนวนระบบเครือข่ายที่มีการให้บริการนั้นๆ
- **Node Address** ระบุจำนวนเครื่องเซิร์ฟเวอร์ที่ **broadcast** การให้บริการนั้นๆ
- **Socket Address** ทำให้ทราบหมายเลข **Socket** ของเครื่องเซิร์ฟเวอร์ที่ให้บริการ
- **Total Hops to Server** เป็นจำนวน **Hop** ที่จะเดินทางไปถึงเครื่องเซิร์ฟเวอร์
- **Operation Field** ระบุประเภทของการร้องขอ
- **Addition Information** เป็นข้อมูล **1 – 2** ฟิลด์ที่ต่อท้าย ที่บอกข้อมูลเพิ่มเติมอย่างอื่นของเครื่องเซิร์ฟเวอร์

### *5.8.5 NCP (NetWare Core Protocol)*

โปรโตคอล **NCP** กำหนดการควบคุมการเชื่อมต่อ และสร้างการร้องขอใช้บริการ ทำให้เครื่องเซิร์ฟเวอร์และเครื่องลูกข่ายสามารถติดต่อสื่อสารระหว่างกันได้ อย่างปลอดภัย

## 5.9 X.25 Product Switching

กลุ่มของโปรโตคอลสำหรับระบบเครือข่าย **WAN** จะประกอบด้วยโปรโตคอล **X.25** ซึ่งให้บริการสลับวงจร **(Switching Service)** มีการให้บริการสวิตชิ่งเป็นครั้งแรกในการเชื่อมต่อเครื่องคอมพิวเตอร์จากระยะไกล **(Remote)** เข้าสู่เครื่องคอมพิวเตอร์เมนเฟรม โดยจะทำการแยกข้อมูลออกเป็นส่วนๆ ส่งผ่านเครือข่ายสายโทรศัพท์ โดยเส้นทางระหว่างโหนดจะกระทำผ่านวงจรเสมือน **(Virtual Circuit)** ข้อมูลแต่ละส่วนจึงถูกส่งผ่านเส้นทางต่างๆ ไปยังจุดหมายปลายทาง และเมื่อถึงปลายทางแพ็กเก็ตข้อมูลเหล่านั้นจะถูกนำมารวมกันเพื่อนำไปใช้งาน

โดยปกติโปรโตคอล X.25 จะประกอบด้วยข้อมูล 128 ไบต์ แต่อย่างไรก็ตามเมื่อเครื่องคอมพิวเตอร์ต้นทางและเครื่องคอมพิวเตอร์ปลายทางทำการเชื่อมต่อกันได้แล้ว จะทำความตกลงระหว่างกันในเรื่องขนาดของแพ็กเก็ตข้อมูลได้ ตามทฤษฎีแล้วโปรโตคอล X.25 สามารถมีเส้นทางในการขนส่งข้อมูลได้ 4096 เส้นทาง และจะทำการขนส่งข้อมูลด้วยความเร็ว 64 Kbps ที่จัดว่ามีความน่าเชื่อถือ โปรโตคอล X.25 จะมีการทำงานใน Physical, Data link และ Network Layer ของโครงสร้าง OSI Model อย่างไรก็ตามโปรโตคอล X.25 ก็มีข้อเสียอยู่ 2 ประการคือ

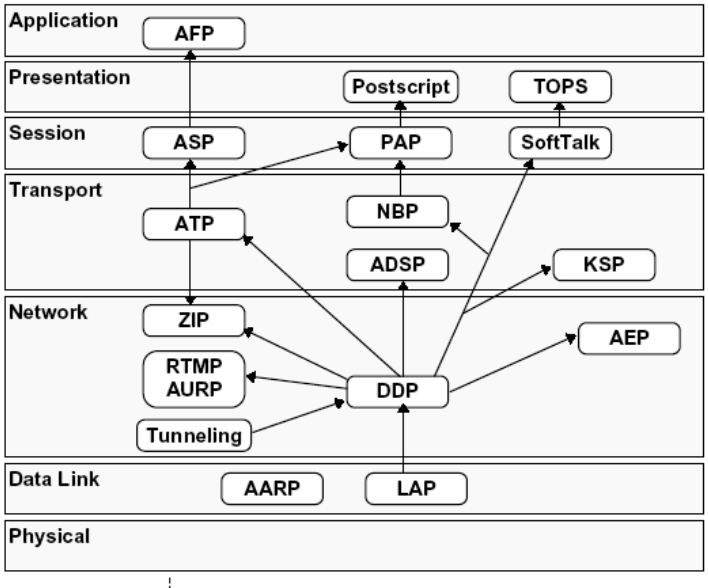
- กระบวนการ Store and Forward ในระหว่างเส้นทางการขนส่งข้อมูล ทำให้เกิดความล่าช้าโดยทั่วไปจะเสียเวลาประมาณ 0.6 วินาที จึงไม่ส่งผลกระทบกับข้อมูลขนาดใหญ่
- มีความต้องการบัฟเฟอร์ขนาดใหญ่เพื่อรองรับ กระบวนการ Store and Forward

โปรโตคอล X.25 และ TCP/IP มีความเหมือนกันตรงที่ต่างเป็นโปรโตคอลแบบ Packet Switching แต่ก็มีความแตกต่างกันบางอย่างคือ

- TCP/IP มีการตรวจสอบความผิดพลาดของข้อมูลในลักษณะ End-to-End ส่วน X.25 จะมีการตรวจสอบความผิดพลาดของข้อมูลในลักษณะ Node-to-Node
- TCP/IP การจัดเรียงข้อมูลด้วยการควบคุมการไหลของกระแสข้อมูล (Flow Control) ที่มีกลไกซับซอนมากกว่า X.25
- X.25 ผูกติดอยู่กับลักษณะของการเชื่อมต่อโดยเฉพาะ ส่วน TCP/IP ได้รับการออกแบบมาให้สามารถใช้รูปแบบในการเชื่อมต่อได้หลายชนิด

## 5.10 AppleTalk

AppleTalk ใช้ในการกล่าวถึงฮาร์ดแวร์ และซอฟต์แวร์ระบบเครือข่าย LAN ของบริษัท Apple โดยใช้เครื่องคอมพิวเตอร์ Macintosh โดยที่ protocol stack ของ AppleTalk เป็นชุดของโปรโตคอลที่เปรียบเทียบได้กับ OSI/ISO Reference Model จำนวน 5 เลเยอร์ดังแสดงตามรูปที่ 5 – 11




รูปที่ 5 – 11 AppleTalk Protocol Suit

การจัดส่งข้อมูลในระบบเครือข่าย **AppleTalk** อยู่บนพื้นฐานของการให้บริการแบบไม่ต้องการการเชื่อมต่อด้วยโปรโตคอล **DDP** ถึงแม้ว่าโปรโตคอลในระดับสูงจะเป็นโปรโตคอลที่ต้องการการเชื่อมต่อในรูปของการจัดตั้งการสนทนาระหว่างเครื่องคอมพิวเตอร์เพื่อทำให้มั่นใจในความน่าเชื่อถือในการจัดส่งข้อมูล โดยโปรโตคอลในแต่ละเลเยอร์จะจัดให้มีการให้บริการกับโปรโตคอลที่อยู่ในเลเยอร์ระดับเหนือขึ้นไปและเลเยอร์ระดับต่ำลงมา รายการต่อไปนี้จะอธิบายโปรโตคอลที่มีในแต่ละเลเยอร์ โดยเริ่มจากเลเยอร์ระดับสูงที่สุด

**5.10.1 Application Layer**

มีโปรโตคอล **AFP (AppleTalk Filing Protocol)** ซึ่งเป็นโปรโตคอลการแบ่งปันการใช้ไฟล์ในโครงสร้าง **AppleTalk** ในโหมด **Native**

**5.10.2 Session Layer** ประกอบด้วย

- **ASP (AppleTalk Session Protocol)** จะบริหารจักการการเชื่อมต่อทางตรรกะกับโปรโตคอลในเลเยอร์ระดับที่สูงกว่า
- **PAP (Printer Access Protocol)** ทำงานกับ **AppleTalk Transaction Protocol** (อยู่ใน **transport layer**) เพื่อส่งคำสั่งจากเครื่องคอมพิวเตอร์ไปยังเครื่องเซิร์ฟเวอร์

**5.10.3 Transport Layer** ประกอบด้วย

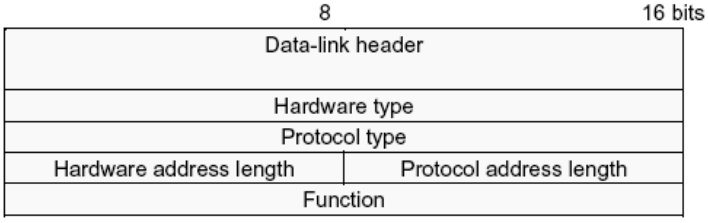
- **ATP (AppleTalk Transaction Protocol)** เป็นโปรโตคอลที่ดูแลการขนส่งแพ็กเก็ตข้อมูล
- **NBP (Name-Binding Protocol)** โปรโตคอลนี้จะรับผิดชอบในการสร้างการเชื่อมต่อระหว่างอุปกรณ์ กับชื่อของระบบเครือข่าย ซึ่งโปรโตคอล **ATP** และ **NBP** จะโต้ตอบกับโปรโตคอล **DDP** ในระดับที่อยู่ต่ำลงมา
- **ADSP (AppleTalk Data Stream Protocol)** ซึ่งทำงานกับ **Datagram Delivery Protocol (DDP – อยู่ใน network layer)** เพื่ออนุญาตให้เครื่องคอมพิวเตอร์จัดตั้งการสนทนาโดยการสื่อสารแบบ 2 ทิศทาง

**5.10.4 Network Layer** ประกอบด้วย

- **DDP (Datagram Delivery Protocol)** ทำหน้าที่ดูแลการจัดส่งข้อมูลของระบบเครือข่ายโดยการจัดเตรียม **datagram** และกำหนดเส้นทางการขนส่งให้
- **ZIP (Zone Information Protocol)** ทำงานกับโปรโตคอล **DDP** ในการกำหนดตำแหน่งของ **nodes** บนระบบเครือข่าย
- **AEP (AppleTalk Echo Protocol)** จัดให้มีบริการ **echo** กับเครื่องแม่ข่ายในระบบเครือข่าย **AppleTalk** ซึ่งสามารถสร้างข้อมูล **echo** ได้มากถึง **585** ไบต์
- **RTMP (Routing Table Maintenance Protocol)** จัดการเส้นทางการขนส่งข้อมูลให้กับระบบเครือข่าย **AppleTalk** โดยจะสื่อสารกับระบบเครือข่ายที่รู้จัก และทำการติดต่อกันโดยตรงจึงช่วยลดความหนาแน่นในการขนส่งข้อมูลได้

**5.10.5 Data link layer** ประกอบด้วย

- **LAP (Link Access Protocol)** คือโปรโตคอลซึ่งบริษัท **Apple** จัดให้มีเพื่อสนับสนุนโครงสร้างสถาปัตยกรรมใน **Physical layer** โดยโปรโตคอลนี้จะรวมการสนับสนุนสำหรับ **EtherTalk, LocalTalk, TokenTalk** และ **FDDITalk** (ระบบเครือข่ายความเร็วสูง ซึ่งมีพื้นฐานการเดินสายเคเบิลด้วยเคเบิลใยแก้วนำแสง และใช้วิธี **token passing** ในการ **access**)
- **AARP (AppleTalk Address Resolution Protocol)** ที่ใช้ในการ **map** แอดเดรสให้กับโปรโตคอลในเลเยอร์ต่างๆ โดยมีโครงสร้างของ **packet** แสดงตามรูปที่ **5 – 12**




รูปที่ **5 – 12 AARP Packet Structure**

## 5.11 APPC (Advanced Program to Program Communication)

**APPC** เป็นชุดโปรโตคอลของบริษัท **IBM** ที่ขยายมาจาก **SNA (Systems Network Architecture)** ซึ่งเป็นสภาวะแวดล้อมของระบบเครือข่ายที่ใช้การประยุกต์บนคอมพิวเตอร์เครื่องอื่นเป็น **peer** ในการสื่อสารกันโดยตรงผ่านระบบเครือข่าย โดยไม่ต้องอาศัยเครื่องแม่ข่ายที่เป็นคอมพิวเตอร์เมนเฟรมเป็นตัวกลาง **APPC** เรียกอีกอย่างหนึ่งว่า **LU 6.2 (Logical Unit)** เพื่ออ้างถึงชื่อ (อย่างสั้น) ที่โปรแกรมประยุกต์ใช้ในการทำให้อุปกรณ์ต่างๆ ทำการแลกเปลี่ยนข้อมูลกันในสภาวะแวดล้อม **SNA**

**LU 6.2** ถูกพัฒนาขึ้นสำหรับนักพัฒนาโปรแกรม **LU 6.2** ที่รันภายใต้ระบบปฏิบัติการดอส สามารถติดต่อสื่อสารกับเครื่องเมนเฟรมซึ่งรันภายใต้ระบบอื่นได้ โดยหลักการโปรแกรมที่ไม่เหมือนกันสามารถ ติดต่อซึ่งกันและกันได้ **IBM** ได้พัฒนา **SNA** ก่อนที่จะมีการใช้เครื่องไมโครคอมพิวเตอร์ หลักการ **SNA** คือ การประมวลผลแบบกระจายงาน (**distributed processing**) การติดต่อภายใต้ **SNA** จะผ่านเครื่องเมนเฟรม แต่ **LU 6.2** จะเป็นสื่อสารแบบจุด-ต่อ-จุด (**peer-to-peer communication**) **LU 6.2** จึงเป็นตัวลดขีดจำกัดต่าง ๆ ของ **SNA** ลง โดย **LU 6.2** มีการเชื่อมต่อแบบ **API (Application Program Interface)** ซึ่งมีคุณสมบัติทางฮาร์ดแวร์ ดั้งนั้น **SNA** จึงกลายเป็นอุปกรณ์ที่เป็นอิสระ ไม่ขึ้นกับฮาร์ดแวร์

ใน **APPC** จะใช้ **LU** เป็นชื่อในการสื่อสารกับระบบและโปรแกรมอื่นในระบบเครือข่าย โดย **APPC** จะทำงานใน **transport layer** และถูกออกแบบมาให้ยอมรับการโต้ตอบระหว่างอุปกรณ์ระบบเครือข่าย ตั้งแต่เครื่อง **Workstation** แบบตั้งโต๊ะ จนถึงเครื่องคอมพิวเตอร์แม่ข่าย ซึ่งมีใน **platform** ของระบบเครือข่ายหลายแบบ ซึ่งประกอบด้วย **Apple, UNIX** และ **Windows**

**แบบฝึกหัดท้ายบท**

1. จงอธิบายว่าโปรโตคอล (**Protocol**) คืออะไร และมีหน้าที่อะไร
2. **Routable Protocol** คืออะไร มีประโยชน์อย่างไร จงอธิบายมาพอสังเขป
3. ให้อธิบายงานในการสื่อสารของเลเยอร์ต่างๆ ใน **OSI Reference Model** ตามตารางด้านล่าง

| OSI Layer | Communication Task |
|---|---|
| Application Layer | |
| Presentation Layer | |
| Session Layer | |
| Transport Layer | |
| Network Layer | |
| Data link Layer | |
| Physical Layer | |

4. คอลัมน์ทางด้านซ้ายเป็นรายการของเลเยอร์ทั้ง 7 ของ **OSI Model** ให้เติมชื่อเลเยอร์ของ **TCP/IP** ลงในคอลัมน์ด้านข้างให้สอดคล้องกับแต่ละเลเยอร์ของ **OSI Model**

| OSI Layer | TCP/IP Layer |
|---|---|
| Application Layer | |
| Presentation Layer | |
| Session Layer | |
| Transport Layer | |
| Network Layer | |
| Data link Layer | |
| Physical Layer | |

5. จงเติมชื่อของโปรโตคอลของ **NetWare** ซึ่งทำงานบนแต่ละเลเยอร์ของ **OSI Model**

| OSI Layer | NetWare Protocol |
|---|---|
| Application Layer | |
| Presentation Layer | |
| Session Layer | |
| Transport Layer | |
| Network Layer | |
| Data link Layer | |
| Physical Layer | |

โปรโตคอลการสื่อสารอื่นๆ ซึ่งเป็นที่นิยมใช้มีดังต่อไปนี้

- **AppleTalk**
- **DECnet**
- **NetBEUI**
- **NetBIOS**
- **X.25**

จงเติมชื่อโปรโตคอลดังกล่าวข้างต้นลงในช่องว่างด้านหลังโจทย์ข้อ **6 – 12** (อาจมีได้มากกว่า **1** คำตอบ)

**6.** โปรโตคอลซึ่งมักนิยมใช้ในระบบเครือข่ายแบบ **Peer-to-Peer** ของไมโครซอฟต์ คือ _____

**7.** โปรโตคอลซึ่งถูกนำมาใช้สำหรับ **packet switching** คือ _____

**8.** โปรโตคอลที่ถูกใช้โดยทั่วไปในระบบเครือข่ายของเครื่องคอมพิวเตอร์ **Macintosh** คือ _____

**9.** โปรโตคอลที่ได้รับการออกแบบโดยบริษัท **Digital Equipment Corporation** คือ _____

**10.** โปรโตคอลที่ได้รับการออกแบบโดยบริษัท **IBM** คือ _____

**11.** โปรโตคอลที่ไม่มีความสามารถในการกำหนดเส้นทางการขนส่งข้อมูล คือ _____

**12.** โปรโตคอลที่มีขนาดเล็กและขนส่งข้อมูลได้อย่างรวดเร็วใน **Transport Layer** คือ _____

**13.** จงอธิบายความแตกต่างระหว่าง **Protocol Suit** กับ **Protocol Stack**

**14.** โปรโตคอลใน **TCP/IP Suit** ซึ่งทำงานใน **Application Layer** มีอะไรบ้าง จงอธิบาย

**15.** จงอธิบายการกำหนดที่อยู่ของเครื่องคอมพิวเตอร์บนเครือข่ายอินเตอร์เน็ตมาพอสังเขป

**16.** จงอธิบายข้อแตกต่างระหว่าง **IP Address Class A** กับ **Class C**

**17.** **Net Mask** และ **Subnet Mask** มีประโยชน์อย่างไรในการกำหนดแอดเดรส

**18.** **Subnet** ของ **IP Address 193.127.6.0** มีค่าเท่าใด

**19.** จงอธิบายการทำงานของโปรโตคอล **IPX/SPX** มาพอสังเขป

**20.** **LU 6.2** คืออะไร มีประโยชน์อย่างไรในการสื่อสารข้อมูลบนระบบเครือข่าย

<u>จงเติมคำลงในช่องว่างให้ถูกต้อง</u>

**21.** บริษัทผู้ผลิตเครื่องพิมพ์มีหน้าที่รับผิดชอบในการเขียน _____ สำหรับผลิตภัณฑ์เครื่องพิมพ์ของตนเอง เพื่อให้เครื่องพิมพ์สามารถใช้งานร่วมกับเครื่องคอมพิวเตอร์ได้อย่างมีประสิทธิภาพ

**22.** ไดร์ฟเวอร์ของอุปกรณ์ที่มีอยู่ในระบบปฏิบัติการ _____ โดยบริษัทผู้ผลิตระบบปฏิบัติการ

**23.** ไดร์ฟเวอร์ของการ์ดเชื่อมต่อระบบเครือข่ายจะถูกรวมอยู่ใน _____ ของเครื่องคอมพิวเตอร์

**24.** ไดร์ฟเวอร์ของการ์ดเชื่อมต่อระบบเครือข่ายจะทำงานใน _____ ใน _____ ของ **ISO/OSI Reference Model**

**25.** ซอฟต์แวร์การแปลความหมายจะเป็นไปตาม _____ **NDIS** และ **ODI.**

**26.** **NDIS** กำหนดการเชื่อมต่อสำหรับการสื่อสารระหว่าง _____ กับ_____

27. **ODI** มีการทำงานเหมือน **NDIS** แต่ได้รับการพัฒนาโดย _____และ _____ สำหรับเชื่อมต่อฮาร์ดแวร์เข้ากับโปรโตคอล
28. _____ คือกระบวนการในการรวมการทำงานของโปรโตคอลและ **NIC** เข้าด้วยกัน
29. เครื่องคอมพิวเตอร์ผู้ส่งจะแตกข้อมูลออกเป็นชิ้นเล็กๆ เรียกว่า _____
30. โปรโตคอลที่รองรับการสื่อสารหลายทิศทางระหว่าง **LAN–to–LAN** รู้จักในชื่อว่าโปรโตคอล _____
31. เพื่อหลีกเลี่ยงการเกิดความขัดแย้ง หรือส่งข้อมูลไม่สมบูรณ์ โปรโตคอลจะ _____
32. กฎเกณฑ์การสื่อสารในสภาพแวดล้อมของระบบเครือข่าย **LAN** โดยเฉพาะ เช่นอีเธอร์เน็ตหรือโทเก้นริงมีชื่อเรียกว่า _____
33. **TCP/IP** เป็นโปรโตคอลที่รองรับการกำหนดเส้นทางขนส่งข้อมูล จึงมักถูกใช้เป็น _____
34. **NetBIOS** เป็นโปรโตคอลใน **Session Layer** ของ **IBM** ซึ่งทำหน้าที่เป็น _____ การเชื่อมต่อระบบเครือข่าย
35. **APPC** เป็นโปรโตคอล _____ ของบริษัท **IBM**
36. **NetBEUI** ไม่เหมาะสมที่จะใช้บนระบบเครือข่ายขนาดใหญ่ เนื่องจาก _____
37. **X.25** เป็นโปรโตคอลซึ่งถูกนำมาใช้กับระบบเครือข่าย _____
38. **X.25** ทำงานใน _____, _____ และ _____ **Layer** ใน โครงสร้าง **OSI Reference Model**
39. **AppleTalk** เป็นชุดของโปรโตคอลที่ได้รับการออกแบบมาสำหรับใช้กับเครื่องคอมพิวเตอร์ _____
40. **EtherTalk** อนุญาตให้เครื่องคอมพิวเตอร์ **Macintosh** สื่อสารกับระบบเครือข่าย _____