TCP/IP (Transmitsion Control Protocol/Internet Protocol)

TCP/IP (Transmitsion Control Protocol/Internet Protocol) เป็นชุดของโปรโตคอลที่ถูกใช้ในการสื่อสารผ่านเครือข่ายอินเทอร์เน็ต โดยมีวัตถุประสงค์เพื่อให้สามารถใช้สื่อสารจากต้นทางข้ามเครือข่ายไปยังปลายทางได้ และสามารถหาเส้นทางที่จะส่งข้อมูลไปได้เองโดยอัตโนมัติ ถึงแม้ว่าในระหว่างทางอาจจะผ่านเครือข่ายที่มีปัญหา โปรโตคอลก็ยังคงหาเส้นทางอื่นในการส่งผ่านข้อมูลไปให้ถึงปลายทางได้

ชุดโปรโตคอลนี้ได้รับการพัฒนามาตั้งแต่ปี 1960 ซึ่งถูกใช้เป็นครั้งแรกในเครือข่าย ARPANET ซึ่งต่อมาได้ขยายการเชื่อมต่อไปทั่วโลกเป็นเครือข่ายอินเตอร์เน็ต ทำให้ TCP/IP เป็นที่ยอมรับอย่างกว้างขวางจนถึงปัจจุบัน

TCP/IP Protocol

- การ Encapsulation/Demultiplexing

1. ชั้นโฮสต์-เครือข่าย (Host-to-network)
2. ชั้นสื่อสารอินเตอร์เน็ต (The Internet Layer)
    a. IP (Internet Protocol)
    b. ICMP (Internet Control Message Protocol)
3. ชั้นสื่อสารนำส่งข้อมูล (Transport Layer)
    a. UDP (User Datagram Protocol)
    b. TCP (Transmission Control Protocol)
        i. การสื่อสารของ TCP
        ii. การสื่อสารแบบ Three-ways handshake
4. ชั้นสื่อสารการประยุกต์ (Application Layer)

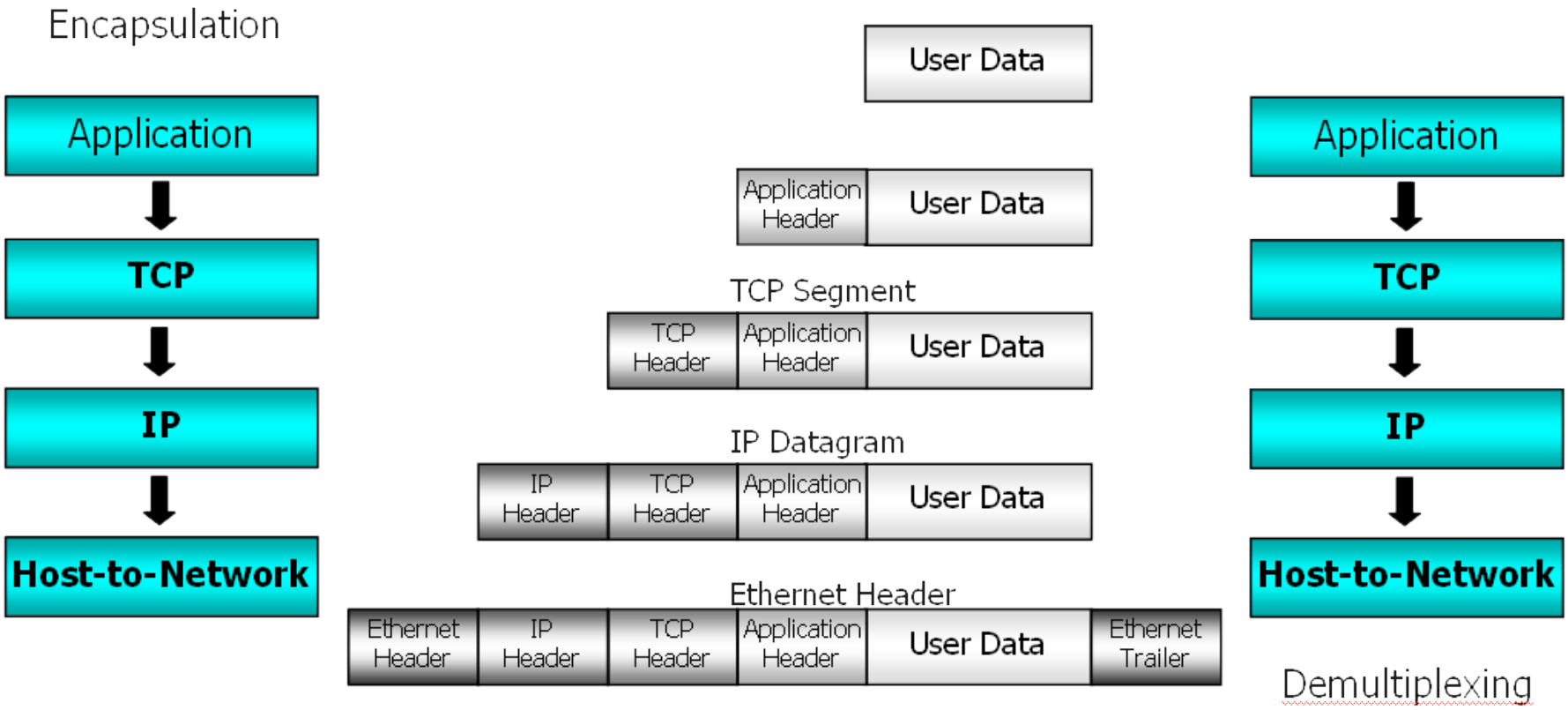
TCP/IP Protocol

TCP/IP มีจุดประสงค์ของการสื่อสารตามมาตรฐาน สามประการคือ

1. เพื่อใช้ติดต่อสื่อสารระหว่างระบบที่มีความแตกต่างกัน
2. ความสามารถในการแก้ไขปัญหาที่เกิดขึ้นในระบบเครือข่าย เช่นในกรณีที่ผู้ส่งและผู้รับยังคงมีการติดต่อกันอยู่ แต่โหนดกลางทีใช้เป็นผู้ช่วยรับ-ส่งเกิดเสียหายใช้การไม่ได้ หรือสายสื่อสารบางช่วงถูกตัดขาด กฎการสื่อสารนี้จะต้องสามารถจัดหาทางเลือกอื่นเพื่อทำให้การสื่อสารดำเนินต่อไปได้โดยอัตโนมัติ
3. มีความคล่องตัวต่อการสื่อสารข้อมูลได้หลายชนิดทั้งแบบที่ไม่มีความเร่งด่วน เช่น การจัดส่งแฟ้มข้อมูล และแบบที่ต้องการรับประกันความเร่งด่วนของข้อมูล เช่น การสื่อสารแบบ real-time และทั้งการสื่อสารแบบเสียง (Voice) และข้อมูล (data)

Encapsulation/Demultiplexing

การส่งข้อมูลผ่านในแต่ละเลเยอร์ แต่ละเลเยอร์จะทำการประกอบข้อมูลที่ได้รับมา กับข้อมูลส่วนควบคุมซึ่งถูกนำมาไว้ในส่วนหัวของข้อมูลเรียกว่า Header ภายใน Header จะบรรจุข้อมูลที่สำคัญของโปรโตคอลที่ทำการ Encapsulate เมื่อผู้รับได้รับข้อมูล ก็จะเกิดกระบวนการทำงานย้อนกลับคือโปรโตคอลเดียวกัน ทางฝั่งผู้รับก็จะได้รับข้อมูลส่วนที่เป็น Header ก่อนและนำไปประมวลและทราบว่าข้อมูลที่ตามมามีลักษณะอย่างไร ซึ่งกระบวนการย้อนกลับนี้เรียกว่า Demultiplexing

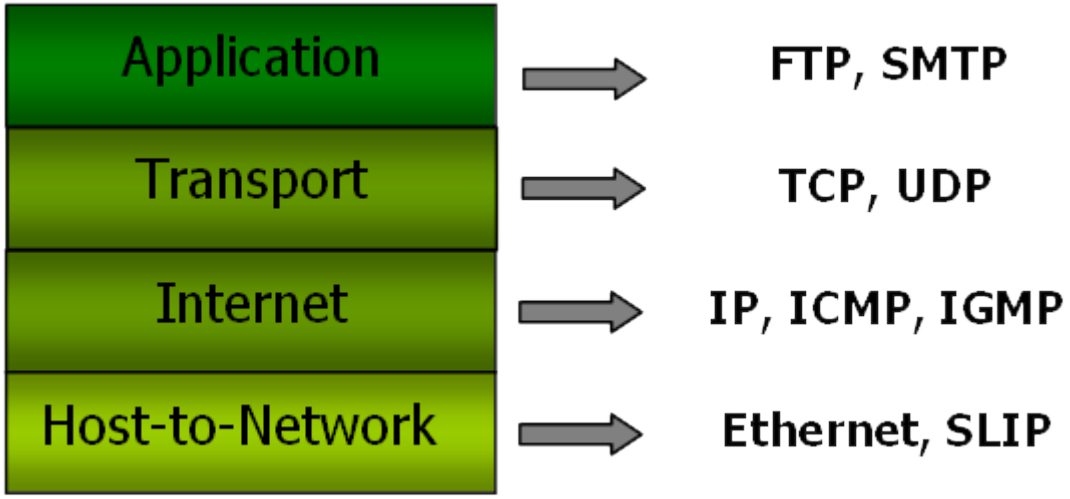


รูปที่1 ขั้นตอนการ Encapsulation และ Demultiplexing

ข้อมูลที่ผ่านการ Encapsulate ในแต่ละเลเยอร์มีชื่อเรียกแตกต่างกัน ดังนี้

- ข้อมูลที่มาจาก User หรือก็คือข้อมูลที่ User เป็นผู้ป้อนให้กับ Application เรียกว่า User Data
- เมื่อแอพพลิเคชั่นได้รับข้อมูลจาก user ก็จะนำมาประกอบกับส่วนหัวของแอพพลิเคชั่น เรียกว่า Application Data และส่งต่อไปยังโปรโตคอล TCP
- เมื่อโปรโตคอล TCP ได้รับ Application Data ก็จะนำมารวมกับ Header ของ โปรโตคอล TCP เรียกว่า TCP Segment และส่งต่อไปยังโปรโตคอล IP
- เมื่อโปรโตคอล IP ได้รับ TCP Segment ก็จะนำมารวมกับ Header ของ โปรโตคอล IP เรียกว่า IP Datagram และส่งต่อไปยังเลเยอร์ Host-to-Network Layer
- ในระดับ Host-to-Network จะนำ IP Datagram มาเพิ่มส่วน Error Correction และ flag เรียกว่า Ethernet Frame ก่อนจะแปลงข้อมูลเป็นสัญญาณไฟฟ้า ส่งผ่านสายสัญญาณที่เชื่อมโยงอยู่ต่อไป

ในแต่ละเลเยอร์ของโครงสร้าง TCP/IP สามารถอธิบายได้ดังนี้



รูปที่2 โครงสร้าง TCP/IP

1. ชั้นโฮสต์-เครือข่าย (Host-to-Network Layer)

โพรโตคอลสำหรับการควบคุมการสื่อสารในชั้นนี้เป็นสิ่งที่ไม่มีการกำหนดรายละเอียดอย่างเป็นทางการ หน้าที่หลักคือการรับข้อมูลจากชั้นสื่อสาร IP มาแล้วส่งไปยังโหนดที่ระบุไว้ในเส้นทางเดินข้อมูลทางด้านผู้รับก็จะทำงานในทางกลับกัน คือรับข้อมูลจากสายสื่อสารแล้วนำส่งให้กับโปรแกรมในชั้นสื่อสาร

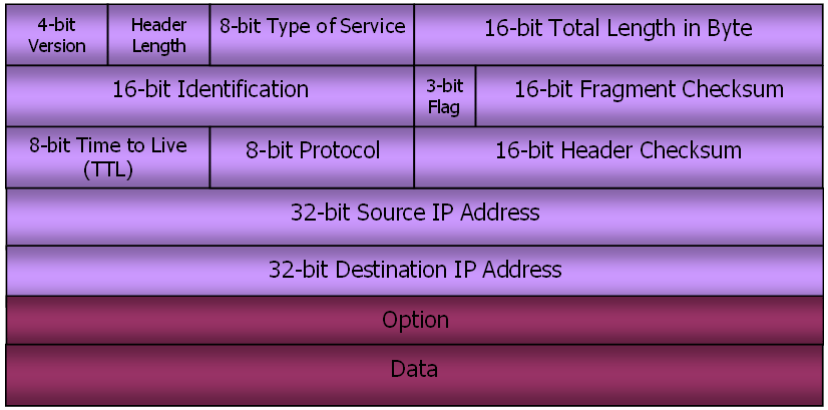
2. ชั้นสื่อสารอินเทอร์เน็ต (The Internet Layer)

ใช้ประเภทของระบบการสื่อสารที่เรียกว่า ระบบเครือข่ายแบบสลับช่องสื่อสารระดับแพ็กเก็ต (packet-switching network) ซึ่งเป็นการติดต่อแบบไม่ต่อเนื่อง (Connectionless) หลักการทำงานคือการปล่อยให้ข้อมูลขนาดเล็กที่เรียกว่า แพ็กเก็ต (Packet) สามารถไหลจากโหนดผู้ส่งไปตามโหนดต่างๆ ในระบบจนถึงจุดหมายปลายทางได้โดยอิสระ หากว่ามีการส่งแพ็กเก็ตออกมาเป็นชุดโดยมีจุดหมายปลายทางเดียวกันในระหว่างการเดินทางในเครือข่าย แพ็กเก็ตแต่ละตัวในชุดนี้ก็จะเป็นอิสระแก่กันและกัน ดังนั้น แพ็กเก็ตที่ส่งไปถึงปลายทางอาจจะไม่เป็นไปตามลำดับก็ได้

a. IP (Internet Protocol)

IP เป็นโปรโตคอลในระดับเน็ตเวิร์คเลเยอร์ ทำหน้าที่จัดการเกี่ยวกับแอดเดรสและข้อมูล และควบคุมการส่งข้อมูลบางอย่างที่ใช้ในการหาเส้นทางของแพ็กเก็ต ซึ่งกลไกในการหาเส้นทางของ IP จะมีความสามารถในการหาเส้นทางที่ดีที่สุด และสามารถเปลี่ยนแปลงเส้นทางได้ในระหว่างการส่งข้อมูล และมีระบบการแยกและประกอบดาต้าแกรม (datagram) เพื่อรองรับการส่งข้อมูลระดับ data link ที่มีขนาด MTU (Maximum Transmission Unit) ทีแตกต่างกัน ทำให้สามารถนำ IP ไปใช้บนโปรโตคอลอื่นได้หลากหลาย เช่น Ethernet ,Token Ring หรือ Apple Talk

การเชื่อมต่อของ IP เพื่อทำการส่งข้อมูล จะเป็นแบบ connectionless หรือเกิดเส้นทางการเชื่อมต่อในทุกๆครั้งของการส่งข้อมูล 1 ดาต้าแกรม โดยจะไม่ทราบถึงข้อมูลดาต้าแกรมที่ส่งก่อนหน้าหรือส่งตามมา แต่การส่งข้อมูลใน 1 ดาต้าแกรม อาจจะเกิดการส่งได้หลายครั้งในกรณีที่มีการแบ่งข้อมูลออกเป็นส่วนย่อยๆ (fragmentation) และถูกนำไปรวมเป็นดาต้าแกรมเดิมเมื่อถึงปลายทาง



รูปที่ 3 IP Header

เฮดเดอร์ของ IP โดยปกติจะมีขนาด 20 bytes ยกเว้นในกรณีที่มีการเพิ่ม option บางอย่าง ฟิลด์ของเฮดเดอร์ IP จะมีความหมายดังนี้

a. **Version** : หมายเลขเวอร์ชันของโปรโตคอล ที่ใช้งานในปัจจุบันคือ เวอร์ชัน 4 (IPv4) และเวอร์ชัน 6 (IPv6)
b. **Header Length** : ความยาวของเฮดเดอร์ โดยทั่วไปถ้าไม่มีส่วน option จะมีค่าเป็น 5 (5*32 bit)
c. **Type of Service (TOS)** : ใช้เป็นข้อมูลสำหรับเราเตอร์ในการตัดสินใจเลือกการเราต์ข้อมูลในแต่ละดาต้าแกรม แต่ในปัจจุบันไม่ได้มีการนำไปใช้งานแล้ว
d. **Length** : ความยาวทั้งหมดเป็นจำนวนไบต์ของดาต้าแกรม ซึ่งด้วยขนาด 16 บิตของฟิลด์ จะหมายถึงความยาวสูงสุดของดาต้าแกรม คือ 65535 byte (64k) แต่ในการส่งข้อมูลจริง ข้อมูลจะถูกแยกเป็นส่วนๆตามขนาดของ MTU ที่กำหนดในลิงค์เลเยอร์ และนำมารวมกันอีกครั้งเมื่อส่งถึงปลายทาง แอพพลิเคชั่นส่วนใหญ่จะมีขนาดของดาต้าแกรมไม่เกิน 512 byte
e. **Identification** : เป็นหมายเลขของดาต้าแกรมในกรณีที่มีการแยกดาต้าแกรมเมื่อข้อมูลส่งถึงปลายทางจะนำข้อมูลที่มี identification เดียวกันมารวมกัน

f. **Flag** : ใช้ในกรณีที่มีการแยกดาต้าแกรม

g. **Fragment offset** : ใช้ในการกำหนดตำแหน่งข้อมูลในดาต้าแกรมที่มีการแยกส่วน เพื่อให้สามารถนำกลับมาเรียงต่อกันได้อย่างถูกต้อง

h. **Time to live (TTL)** : กำหนดจำนวนครั้งที่มากที่สุดที่ดาต้าแกรมจะถูกส่งระหว่าง hop (การส่งผ่านข้อมูลระหว่างเน็ตเวิร์ค) เพื่อป้องกันไม่ให้เกิดการส่งข้อมูลโดยไม่สิ้นสุด โดยเมื่อข้อมูลถูกส่งไป 1 hop จะทำการลดค่า TTL ลง 1 เมื่อค่าของ TTL เป็น 0 และข้อมูลยังไม่ถึงปลายทาง ข้อมูลนั้นจะถูกยกเลิก และเราเตอร์สุดท้ายจะส่งข้อมูล ICMP แจ้งกลับมายังต้นทางว่าเกิด time out ในระหว่างการส่งข้อมูล

i. **Protocol** : ระบุโปรโตคอลที่ส่งในดาต้าแกรม เช่น TCP ,UDP หรือ ICMP

j. **Header checksum** : ใช้ในการตรวจสอบความถูกต้องของข้อมูลในเฮดเดอร์

k. **Source IP address** : หมายเลข IP ของผู้ส่งข้อมูล

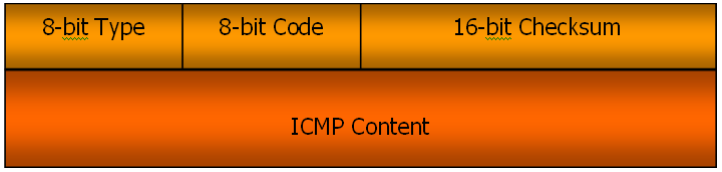
l. **Destination IP address** : หมายเลข IP ของผู้รับข้อมูล

m. **Data** : ข้อมูลจากโปรโตคอลระดับบน

b. ICMP (Internet Control Message Protocol)

ICMP เป็นโปรโตคอลที่ใช้ในการตรวจสอบและรายงานสถานภาพของดาต้าแกรม (Datagram) ในกรณีที่เกิดปัญหากับดาต้าแกรม เช่น เราเตอร์ไม่สามารถส่งดาต้าแกรมไปถึงปลายทางได้ ICMP จะถูกส่งออกไปยังโฮสต้นทางเพื่อรายงานข้อผิดพลาด ที่เกิดขึ้น อย่างไรก็ดี ไม่มีอะไรรับประกันได้ว่า ICMP Message ที่ส่งไปจะถึงผู้รับจริงหรือไม่ หากมีการส่งดาต้าแกรมออกไปแล้วไม่มี ICMP Message ฟ้อง Error กลับมา ก็แปลความหมายได้สองกรณีคือ ข้อมูลถูกส่งไปถึงปลายทางอย่างเรียบร้อย หรืออาจจะมีปัญหา ในการสื่อสารทั้งการส่งดาต้าแกรม และ ICMP Message ที่ส่งกลับมาก็มีปัญหาระว่างทางก็ได้ ICMP จึงเป็นโปรโตคอลที่ไม่มีความน่าเชื่อถือ (unreliable) ซึ่งจะเป็นหน้าที่ของ โปรโตคอลในระดับสูงกว่า Network Layer ในการจัดการให้การสื่อสารนั้นๆ มีความน่าเชื่อถือ

ในส่วนของ ICMP Message จะประกอบด้วย Type ขนาด 8 บิต Checksum ขนาด 16 บิต และส่วนของ Content ซึ่งจะมีขนาดแตกต่างกันไปตาม Type และ Code ดังรูป



รูปที่ 4 ICMP Header

3. ชั้นสื่อสารนำส่งข้อมูล (Transport Layer)

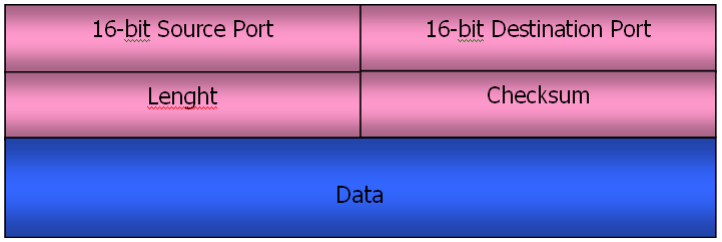
แบ่งเป็นโพรโตคอล 2 ชนิดตามลักษณะ ลักษณะแรกเรียกว่า Transmission Control Protocol (TCP) เป็นแบบที่มีการกำหนดช่วงการสื่อสารตลอดระยะเวลาการสื่อสาร (connection-oriented) ซึ่งจะยอมให้มีการส่งข้อมูลเป็นแบบ Byte stream ที่ไว้ใจได้โดยไม่มีข้อผิดพลาด ข้อมูลที่มีปริมาณมากจะถูกแบ่งออกเป็นส่วนเล็กๆ เรียกว่า message ซึ่งจะถูกส่งไปยังผู้รับผ่านทางชั้นสื่อสารของอินเทอร์เน็ต ทางฝ่ายผู้รับจะนำ message มาเรียงต่อกันตามลำดับเป็นข้อมูลตัวเดิม TCP ยังมีความสามารถในการควบคุมการไหลของข้อมูลเพื่อป้องกันไม่ให้ผู้ส่ง ส่งข้อมูลเร็วเกินกว่าที่ผู้รับจะทำงานได้ทันอีกด้วย

โปรโตคอลการนำส่งข้อมูลแบบที่สองเรียกว่า UDP (User Datagram Protocol) เป็นการติดต่อแบบไม่ต่อเนื่อง (connectionless) มีการตรวจสอบความถูกต้องของข้อมูลแต่จะไม่มีการแจ้งกลับไปยังผู้ส่ง จึงถือได้ว่าไม่มีการตรวจสอบความถูกต้องของข้อมูล อย่างไรก็ตาม วิธีการนี้มีข้อดีในด้านความรวดเร็วในการส่งข้อมูล จึงนิยมใช้ในระบบผู้ให้และผู้ใช้บริการ (client/server system) ซึ่งมีการสื่อสารแบบ ถาม/ตอบ (request/reply) นอกจากนั้นยังใช้ในการส่งข้อมูลประเภทภาพเคลื่อนไหวหรือการส่งเสียง (voice) ทางอินเทอร์เน็ต

a. UDP : (User Datagram Protocol)

เป็นโปรโตคอลที่อยู่ใน Transport Layer เมื่อเทียบกับโมเดล OSI โดยการส่งข้อมูลของ UDP นั้นจะเป็นการส่งครั้งละ 1 ชุดข้อมูล เรียกว่า UDP datagram ซึ่งจะไม่มีความสัมพันธ์กันระหว่างดาต้าแกรมและจะไม่มีกลไกการตรวจสอบความสำเร็จในการรับส่งข้อมูล

กลไกการตรวจสอบโดย checksum ของ UDP นั้นเพื่อเป็นการป้องกันข้อมูลที่อาจจะถูกแก้ไข หรือมีความผิดพลาดระหว่างการส่ง และหากเกิดเหตุการณ์ดังกล่าว ปลายทางจะได้รู้ว่ามีข้อผิดพลาดเกิดขึ้น แต่มันจะเป็นการตรวจสอบเพียงฝ่ายเดียวเท่านั้น โดยในข้อกำหนดของ UDP หากพบว่า Checksum Error ก็ให้ผู้รับปลายทางทำการทิ้งข้อมูลนั้น แต่จะไม่มีการแจ้งกลับไปยังผู้ส่งแต่อย่างใด การรับส่งข้อมูลแต่ละครั้งหากเกิดข้อผิดพลาดในระดับ IP เช่น ส่งไม่ถึง, หมดเวลา ผู้ส่งจะได้รับ Error Message จากระดับ IP เป็น ICMP Error Message แต่เมื่อข้อมูลส่งถึงปลายทางถูกต้อง แต่เกิดข้อผิดพลาดในส่วนของ UDP เอง จะไม่มีการยืนยัน หรือแจ้งให้ผู้ส่งทราบแต่อย่างใด



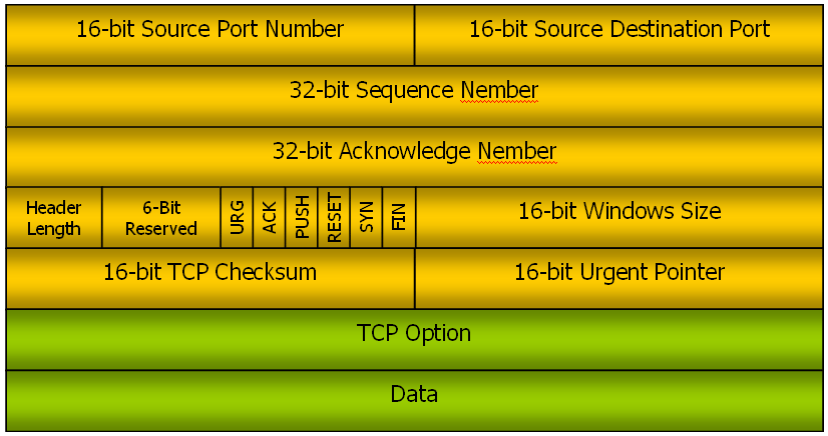
รูปที่ 5 UDP Header

มีรายละเอียด ดังนี้

- **Source Port Number** : หมายเลขพอร์ตต้นทางที่ส่งดาต้าแกรมนี้
- **Destination Port Number** : หมายเลขพอร์ตปลายทางที่จะเป็นผู้รับดาต้าแกรม
- **UDP Length** : ความยาวของดาต้าแกรม ทั้งส่วน Header และ data นั่นหมายความว่า ค่าที่น้อยที่สุดในฟิลด์นี้คือ 8 ซึ่งเป็นขนาดของ Header
- **Checksum** : เป็นตัวตรวจสอบความถูกต้องของ UDP datagram และจะนำข้อมูลบางส่วนใน IP Header มาคำนวณด้วย

b. TCP : (Transmission Control Protocol)

อยู่ใน Transport Layer เช่นเดียวกับ UDP ทำหน้าที่จัดการและควบคุมการรับส่งข้อมูล ซึ่งมีความสามารถและรายละเอียดมากกว่า UDP โดยดาต้าแกรมของ TCP จะมีความสัมพันธ์ต่อเนื่องกัน และมีกลไกควบคุมการรับส่งข้อมูลให้มีความถูกต้อง (reliable) และมีการสื่อสารอย่างเป็นกระบวนการ (connection-oriented)



รูปที่ 6 TCP Header

มีรายละเอียด ดังนี้

- **Source Port Number** : หมายเลขพอร์ตต้นทางที่ส่งดาต้าแกรมนี้
- **Destination Port Number** : หมายเลขพอร์ตปลายทางที่จะเป็นผู้รับดาต้าแกรม
- **Sequence Number** : ฟิลด์ที่ระบุหมายเลขลำดับอ้างอิงในการสื่อสารข้อมูลแต่ละครั้ง เพื่อใช้ในการแยกแยะว่าเป็นข้อมูลของชุดใด และนำมาจัดลำดับได้ถูกต้อง
- **Acknowledgment Number** : ทำหน้าที่เช่นเดียวกับ Sequence Number แต่จะใช้ในการตอบรับ
- **Header Length** : โดยปกติความยาวของเฮดเดอร์ TCP จะมีความยาว 20 ไบต์ แต่อาจจะมากกว่านั้น ถ้ามีข้อมูลในฟิลด์ option แต่ต้องไม่เกิน 60 ไบต์

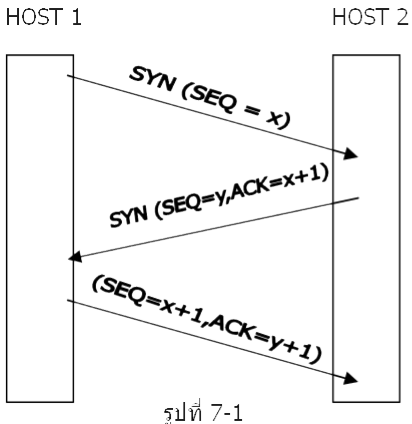
- **Flag** : เป็นข้อมูลระดับบิตที่อยู่ในเฮดเดอร์ TCP โดยใช้เป็นตัวบอกคุณสมบัติของแพ็กเก็ต TCP ขณะนั้นๆ และใช้เป็นตัวควบคุมจังหวะการรับส่งข้อมูลด้วย ซึ่ง Flag มีอยู่ทั้งหมด 6 บิต แบ่งได้ดังนี้

Type	Description
URG	ใช้บอกความหมายว่าเป็นข้อมูลด่วน และมีข้อมูลพิเศษมาด้วย (อยู่ใน Urgent pointer)
ACK	แสดงว่าข้อมูลในฟิลด์ Acknowledge Number นำมาใช้งานได้
DSH	เป็นการแจ้งให้ผู้รับข้อมูลทราบว่าควรจะส่งข้อมูล Segment นี้ไปยัง Application ที่กำลังรออยู่โดยเร็ว
RST	ยกเลิกการติดต่อ (reset) เนื่องจากในกรณีที่เกิดการสับสนขึ้นด้วยเหตุผลต่างๆ เช่น โฮสต์มีปัญหา ให้เริ่มต้นสื่อสารกันใหม่
SYN	ใช้ในการเริ่มต้นขอติดต่อกับปลายทาง
FIN	ใช้ส่งเพื่อแจ้งให้ปลายทางทราบว่ายุติการติดต่อ

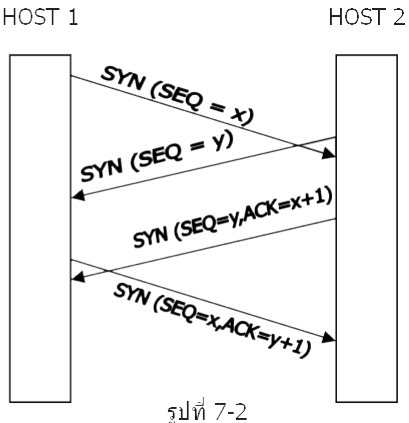
Flag ในเฮดเดอร์ของ TCP มีความสำคัญในการกำหนดการทำงานของ TCP segment เนื่องจากข้อมูลในเฮดเดอร์ของ TCP จะมีข้อมูลครบถ้วนทั้งการรับและการส่งข้อมูล ซึ่งในการสทำงานแต่ละอย่างจะมีการใช้งานฟิลด์ไม่เหมือนกัน flag จะเป็นตัวกำหนดว่าให้ใช้งานฟิลด์ไหน เช่น ฟิลด์ Acknowledgment number จะไม่ถูกใช้ในขั้นตอนการเริ่มต้นการเชื่อมต่อ แต่จะมีข้อมูลในฟิลด์ ซึ่งเป็นข้อมูลที่ไม่มีความหมายใดๆ ซึ่งถ้าไม่มี flag เป็นตัวกำหนดก้ออาจจะมีการนำข้อมูลมาใช้ และก่อให้เกิดความผิดพลาดได้

---

i. การสื่อสารของ TCP



รูปที่ 7-1



รูปที่ 7-2

เมื่อเซกเมนต์ CONNECT (SYN = “1” และ ACK = “0”) เดินทางมาถึง Entity TCP ที่โฮสต์ปลายทางจะค้นหาโพรเซสตามหมายเลขพอร์ตที่กำหนดในเขตข้อมูล Destination port ซึ่งถ้าหากไม่พบก็จะตอบปฏิเสธด้วยเซกเมนต์ที่มี RST = “1” กลับไปยังผู้ส่ง
เซกเมนต์ CONNECT ของผู้ส่งจะถูกส่งต่อไปยังโพรเซส ตามพอร์ตที่ระบุซึ่งอาจจะตอบรับหรือตอบปฏิเสธก็ได้ ถ้าโพรเซสนั้นต้องการสื่อสารด้วยก็จะส่งเซกเมนต์ตอบรับกลับไป รูปที่ 6-1 แสดงลำดับขั้นตอนการส่ง TCP เซกเมนต์ในการสร้างการเชื่อมต่อในสภาวะปกติระหว่างผู้ส่งและผู้รับ

ในกรณีที่โฮสต์สองแห่งพยายามสร้างการเชื่อมต่อระหว่างซ็อคเก็ตคู่เดียวกันจะเกิดเป็นลำดับขั้นตอนแสดงในรูปที่ 6-2 ผลสุดท้ายจะมีการเชื่อมต่อเกิดขึ้นเพียงหนึ่งช่องทางเท่านั้นเนื่องจากการเชื่อมต่อในแต่ละช่องทางจะถูกกำหนดขึ้นโดยใช้หมายเลขซ็อคเก็ตผู้ส่งและผู้รับ ถ้าการเชื่อมต่อลำดับแรกสำเร็จก็จะถูกบันทึกไว้ในตารางการสื่อสาร เช่น (x, y) ถ้าการเชื่อมต่อลำดับที่สองสำเร็จในเวลาต่อมา ข้อมูลนี้ก็จะถูกบันทึกไว้ที่เดียวกันคือ (x, y)

ขั้นตอนในการสร้างการเชื่อมต่อและการยกเลิกสามารถเขียนอธิบายด้วยไฟไนท์สเตทแมชชีนที่มีการทำงาน 11 สถานะ ดังแสดงในตารางข้างล่าง ในแต่ละสถานะจะมีเหตุการณ์บางอย่างที่เป็นไปได้ซึ่งจะได้รับการตอบสนองด้วยการกระทำที่เหมาะสม ในทางตรงกันข้าม เหตุการณ์ที่เป็นไปไม่ได้จะกลายเป็นข้อผิดพลาดที่จะต้องรายงานให้ทราบ

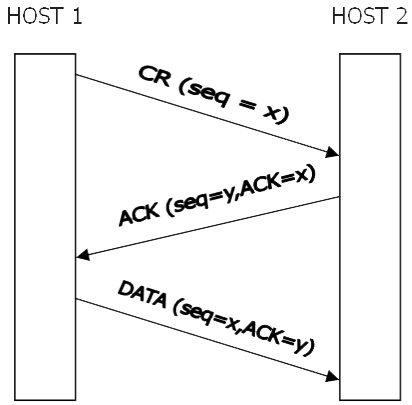
การเชื่อมต่อเริ่มต้นจากสถานะ CLOSED เมื่อเรียกใช้บริการ LISTEN หรือ CONNECT ก็จะมีการเปลี่ยนสถานะไปจากเดิม และถ้าอีกฝ่ายตองการเชื่อมต่อด้วย การเชื่อมต่อก็จะเกิดขึ้นและย้ายไปอยู่ในสถานะ ESTABLISHED คือการเชื่อต่อสมบูรณ์ และเมื่อยกเลิกการติดต่อก็จะกลับไปสู่สถานะ CLOSED อย่างเดิม

ii. การเริ่มต้นการสื่อสารของ TCP โดยใช้การบันทึกเวลาแบบ Three-way handshake

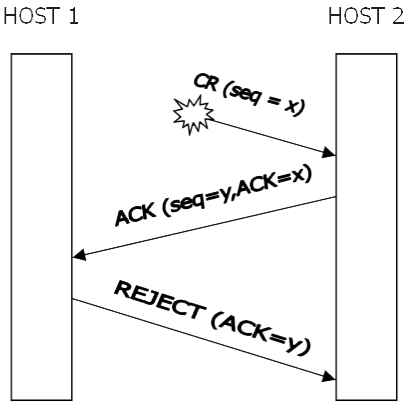
Three-way Handshake เป็นวิธีการส่งแพ็กเก็ตที่สามารถช่วยแก้ปัญหาในเรื่องแพ็กเก็ตซ้ำซ้อนได้ดี แต่วิธีนี้จำเป็นจะต้องสร้างช่องสื่อสารให้ได้ก่อนที่จะเริ่มรับ-ส่งข้อมูล อย่างไรก็ตาม แพ็กเก็ตควบคุมที่ใช้ในการต่อรองค่าตัวแปรสำหรับการสื่อสารต่างๆ อาจเกิดการตกค้างอยู่ในระบบได้ ทำให้การกำหนดค่าหมายเลขลำดับมีปัญหาไปด้วย เช่นการสร้างช่องสื่อสารระหว่างโฮสต์1 และ โฮสต์2 เริ่มจาก โฮสต์1 ขอเริ่มการเชื่อต่อด้วยการส่งแพ็กเก็ต CR (Connection Request) ไปยังโฮสต์2 ซึ่งจะมีค่าตัวแปรต่างๆสำหรับการสื่อสารรวมทั้งหมายเลขลำดับและหมายเลขช่องสื่อสารไปด้วย ผู้รับคือโฮสต์2 ก็จะส่ง ACK (Acknowledge) กลับมายังโฮสต์1 แต่ถ้าแพ็กเก็ต จากผู้ส่งเกิดสูญหายระหว่างทางและสำเนาแพ็กเก็ตที่ยังตกค้างอยู่ระบบเกิดเดินทางไปถึงผู้รับในภายหลังก็จะทำให้การสร้างช่องสื่อสารใช้การไม่ได้เนื่องจากมีค่าตัวแปรต่างๆไม่ตรงกัน

การใช้ Three-way handshake เป็นการไม่บังคับให้ผู้ส่งและผู้รับข้อมูลจะต้องกำหนดค่าเริ่มต้นของหมายเลขลำดับเป็นเลขเดียวกัน ทำให้สามารถนำวิธีนี้มาใช้ร่วมกับวิธีการจัดจังหวะการทำงานให้พร้อมกัน

(Synchronization) แบบต่างๆได้ แทนที่จะเป็นการใช้วิธีการบันทึกเวลา ดังรูปที่ 7-1 แสดงขั้นตอนการเริ่มต้นการทำงานจากโฮสต์ 1 ไปยังโฮสต์ 2 สมมุติให้โฮสต์ 1 เลือกหมายเลขลำดับเป็น “x” และส่งแพ็กเก็ต CONNECTION REQUEST ไปยังโฮสต์ 2 โฮสต์ 2 ตอบรับด้วยแพ็กเก็ต CONNECTION ACCEPTED ซึ่งจะยอมรับหมายเลขลำดับ “x” พร้อมกับประกาศหมายเลขลำดับ “y” ที่เป็นของตนเอง จากนั้นโฮสต์ 1 ก็จะตอบรับค่าตัวเลือกของโฮสต์ 2 ผ่านทางเขตข้อมูลสำหรับการควบคุมในแพ็กเก็ตข้อมูลแรกที่ส่งมา



รูปที่ 8-1



รูปที่ 8-2

สมมติว่าได้เกิดปัญหาการสูญหายของแพ็กเก็ตในขณะที่สำเนาแพ็กเก็ตที่ค้างในระบบเดินทางไปถึงผู้รับแทน รูปที่7-2 แสดงเหตุการณ์ที่แพ็กเก็ตTPDU (ตัวแรกในรูป) เป็นสำเนาแพ็กเก็ตเก่าที่พึ่งจะเดินทางไปถึงโฮสต์ 2 โดยที่โฮสต์ 1 ไม่ทราบ โฮสต์ 2 ก็จะทำงานตามปกติคือจะตอบรับด้วยการส่งแพ็กเก็ต CONNECTION ACCEPTED TPDU กลับมา ที่โฮสต์ 1 ซึ่งโฮสต์1 จะสามารถตรวจสอบได้ว่า หมายเลขลำดับโฮสต์2 ตอบกลับมานั้นเป็นหมายเลขลำดับที่ได้เลิกใช้ไปแล้ว จึงมีการส่งแพ็กเก็ต REJECTกลับมายังโฮสต์ 2 เพื่อบอกยกเลิกการทำงาน จะเห็นว่าวิธีการนี้อาศัยการสื่อสารผ่านแพ็กเก็ต 3 ตัวซึ่งเป็นที่มาของคำว่า “การจับมือร่วมสามขั้นตอน” ผลสุดท้าย ทั้งโฮสต์ 1 และโฮสต์ 2 ก็จะไม่มีการสร้างช่องสื่อสารขึ้นมาจากข้อมูลในสำเนาแพ็กเก็ตเก่าแต่อย่างใด

4. ชั้นสื่อสารการประยุกต์ (Application Layer)

มีโพรโตคอลสำหรับสร้างจอเทอร์มินัลเสมือน เรียกว่า TELNET โพรโตคอลสำหรับการจัดการแฟ้มข้อมูล เรียกว่า FTP และโพรโตคอลสำหรับการให้บริการจดหมายอิเล็กทรอนิกส์ เรียกว่า SMTP โดยโพรโตคอลสำหรับสร้างจอเทอร์มินัลเสมือนช่วยให้ผู้ใช้สามารถติดต่อกับเครื่องโฮสต์ที่อยู่ไกลออกไปโดยผ่านอินเทอร์เน็ต และสามารถทำงานได้เสมือนกับว่ากำลังนั่งทำงานอยู่ที่เครื่องโฮสต์นั้น โพรโตคอลสำหรับการจัดการแฟ้มข้อมูลช่วยในการคัดลอกแฟ้มข้อมูลมาจากเครื่องอื่นที่อยู่ในระบบเครือข่ายหรือส่งสำเนาแฟ้มข้อมูลไปยังเครื่องใดๆก็ได้ โพรโตคอลสำหรับให้บริการจดหมายอิเล็กทรอนิกส์ช่วยในการจัดส่งข้อความไปยังผู้ใช้ในระบบ หรือรับข้อความที่มีผู้ส่งเข้ามา