CPE 426 Computer Networks

Chapter 8:
Text Chapter 26: TCP

TOPICS

- **Chapter 26: TCP** (Transmission Control Protocol)
  - **Transport Services**
  - **Transport Protocol Techniques**
  - **Congestion Control/Avoidance Techniques**
  - **Packet Loss and Adaptive Retransmission**
  - **Buffers, Flow Control and Windows**
  - **TCP Three-Way Handshake**
  - **TCP Congestion Control**
  - **TCP Segment Format**

Chapter 26

- TCP
- Transport Control Protocol
- Reliable Protocol for the Internet
- TCP/IP (TCP over IP)

Chapter 26 TCP: Reliable Transport Service

- **ในกรณีที่เราต้องการความมั่นใจในการส่งข้อมูลได้อย่างถูกต้อง ผ่าน IP ที่มีการทำงานแบบ Datagram เราต้องใส่ Mechanism เพื่อให้ความมั่นใจดังกล่าวลงใน Protocol ของ Layer 4 (Host-to-Host หรือ Transport)**
  - เราจะต้องเลือกใช้ TCP (Transmission Control Protocol)
  - เรียกรวมๆว่า TCP/IP
  - การใช้ TCP+IP ทำให้ผู้เขียน Application ไม่ต้องกังวลเรื่องของการส่งข้อมูลอีกต่อไป
    - เพียงแต่นำข้อมูลส่งให้ พร้อมกำหนด IP Address และ Port Number จากนั้น ระบบ TCP/IP จะจัดการที่เหลือให้
    - ผิดกับกรณีที่ใช้ UDP/IP ในบทที่แล้ว
    - กล่าวคือ TCP จะให้ Reliability ในการสื่อสาร

Chapter 26 TCP: 26.3 TCP Service

- **TCP ให้บริการของ 7 ส่วนใหญ่ๆ ดังนี้**
  - Connection Orientation
    - TCP ให้บริการแบบ Connection-Oriented ซึ่ง Application จะต้องร้องขอการเชื่อมต่อก่อนที่จะมีการส่งข้อมูล
  - Point-to-Point Communication
    - แต่ละ Connection ของ TCP จะมีสอง End Points เท่านั้น (กำหนด Port ต้นทาง-ปลายทาง และ IP ต้นทาง-ปลายทาง)
  - Complete Reliability
    - TCP จะ Guarantee ว่าข้อมูลที่ส่ง จะไปถึงที่หมายได้อย่างถูกต้อง สมบูรณ์ และเป็นลำดับ
  - Full Duplex Communication
    - แต่ละ TCP Connection จะยอมให้ข้อมูลส่งได้สองทิศทางตลอดเวลา

Chapter 26 TCP: 26.3 TCP Service(2)

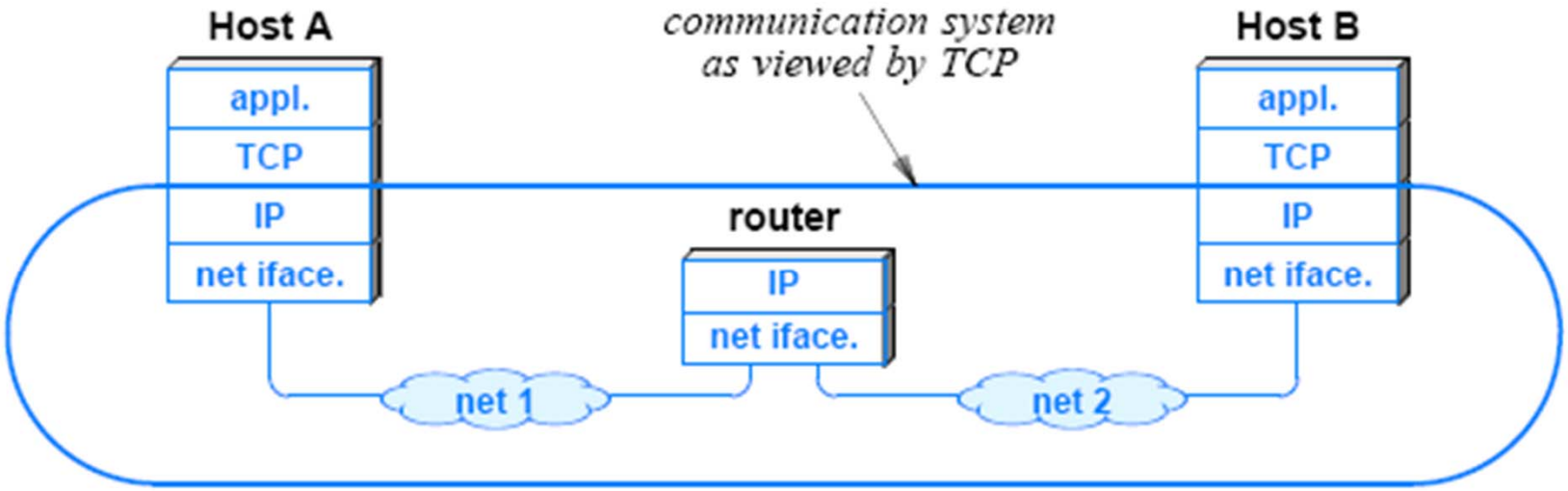
- **TCP ให้บริการของ 7 ส่วนใหญ่ๆ ดังนี้(ต่อ)**
  - Stream Interface
    - Application สามารถส่งข้อมูลได้อย่างต่อเนื่อง Octet ต่อ Octet ผ่าน TCP Connection โดย TCP จะไม่มีการรวมกลุ่มของ Data ให้เป็น Record หรือ Message และจะไม่ Guarantee ว่าแต่ละส่วนของข้อมูลที่ไปถึงมีขนาดเท่ากับที่ Application ส่งให้(มีการทำ Segmentation)
  - Reliable Connection Startup
    - TCP ให้สอง Application สามารถเริ่มต้นการสื่อสารได้อย่างมั่นใจ
  - Graceful Connection Shutdown
    - ก่อนที่จะจบ Connection นั้น TCP จะให้ความมั่นใจว่าข้อมูลได้ถูกส่งอย่างครบถ้วนทั้งสองฝั่ง และทั้งสองฝั่งยินยอมให้มีการจบการเชื่อมต่อ

Chapter 26 TCP: 26.4 End-to-End Service and Virtual Connection

- **TCP จัดว่าเป็น End-to-End Protocol เหมือน UDP**
  - เนื่องจากมันให้บริการการสื่อสารระหว่าง Application ของสอง Computer
- **แต่มันเป็น Connection-Oriented**
  - เนื่องจากต้องมีการทำ Connection ก่อนส่งข้อมูล
  - และต้องมีการทำ Disconnection
- **Connection ของ TCP จัดว่าเป็น Virtual Connection**
  - เนื่องจากกระทำผ่าน Software เพราะตัว Network (IP) นั้นเป็น Connectionless(Datagram)

Chapter 26 TCP: 26.4 End-to-End Service and Virtual Connection

- **แต่ละ TCP Message(Segment) จะถูกบรรจุใน IP Datagram และส่งผ่าน Internet เมื่อถึงปลายทาง IP จะปลอก Datagram Payload และส่งต่อไปยัง TCP**
  - IP จะมอง TCP เป็นแค่ Data ที่จะต้องส่ง
  - TCP จะมอง IP เป็นพาหนะสำหรับส่งข้อมูล ไปยัง TCP อีกฝั่งหนึ่ง



Chapter 26 TCP: 26.5 Transport Protocol Techniques

- **ปัญหาที่ End-to-End Transport Protocol จะต้องเจอ ในการได้มาซึ่ง Reliable Service**
  - Unreliable Communication
    - Message อาจจะสูญหาย ซ้ำซ้อน ผิดพลาด ถูกหน่วงเวลา หรือส่งมาไม่เป็นลำดับ (Lost, Duplicated, Corrupted, Delayed, Out of Order)
  - End System Reboot
    - ถ้า Host เกิด Crash หรือมีการ Reboot ข้อมูลที่ค้างอยู่ จะต้องไม่สับสนกับ Session ที่ถูกสร้างขึ้นมาใหม่
  - Heterogeneous End System
    - แต่ละ Host ที่เชื่อมต่อใน Internet มีความสามารถรับ-ส่งข้อมูลได้ไม่เท่ากัน
  - Congestion in the Internet
    - เมื่อมีการส่งข้อมูลมากเกินไป จะเกิดความคับคั่งภายใน Network ได้
- **ปกติแล้ว Transport Protocol จะใช้หลายๆวิธีร่วมกันในการจัดการกับปัญหาเหล่านี้ เราจะกล่าวพื้นฐานแต่ละอันต่อไป**

Chapter 26 TCP: 26.5.1 Sequencing to Handle Duplicates and Out-of-Order Delivery

- **Transport Protocol จะทำ Sequencing (กำหนดหมายเลขลำดับของข้อมูล) ในการจัดการกับปัญหาเรื่อง Duplicate Data และข้อมูลที่ไปถึงอย่างไม่เป็นลำดับ**
  - คล้ายกับวิธีที่กระทำใน Layer 2
  - หมายเลขข้อมูลทำให้เรารู้ว่าข้อมูลนั้นมาซ้ำกันหรือไม่
  - ทำให้รู้ว่าข้อมูลมาเป็นลำดับหรือไม่

Chapter 26 TCP: 26.5.2 Retransmission to Handle Lost Packet

- **ในการจัดการกับ Packet Lost นั้น Transport Protocol จะใช้ Positive Acknowledgement ร่วมกับการ Retransmission**
  - คล้ายกับ Mechanism ใน Layer 2 เช่นกัน
  - เมื่อ Frame มาถึงอย่างถูกต้อง จะมีการส่ง ACK Message กลับไป
  - เมื่อผู้ส่ง ส่งข้อมูลแต่ละ Packet จะมีการจับเวลาโดยใช้ Timer
    - ถ้า Timer Expire (คือไม่ได้รับ ACK ในเวลาที่กำหนด) ผู้ส่งจะทำการ Retransmission ข้อมูลนั้นไปใหม่
  - ถ้า Packet มีการ Delay มาก อาจจะทำให้เกิด Retransmission ยังผลให้เกิด Duplicate Packet

Chapter 26 TCP: 26.5.3 Techniques to Avoid Replay

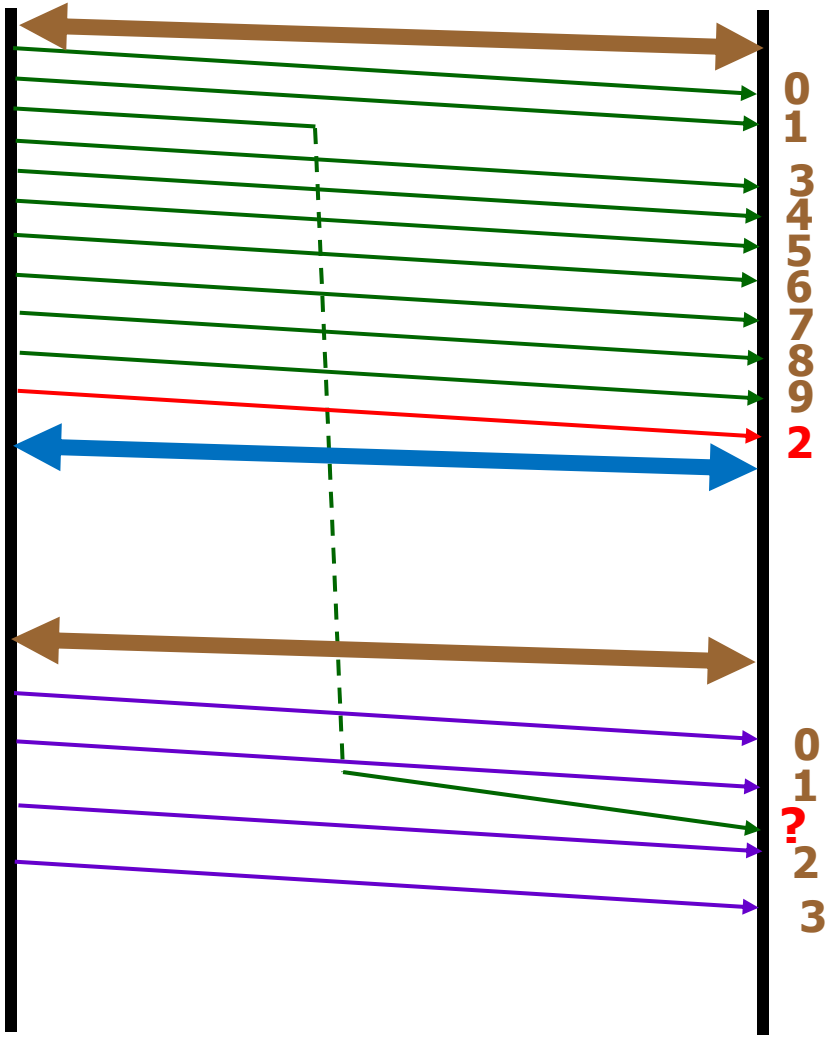
- **ในกรณีที่ Network มี Delay สูง อาจจะทำให้เกิด 'Replay Error' โดยที่ Packet ที่ Delay นั้นจะส่งผลต่อการสื่อสารในตอนหลัง**
  ยกตัวอย่าง
  - สองคอมพิวเตอร์ตกลงจะสื่อสารกันเมื่อเวลา 13.00 น.
  - คอมพิวเตอร์เครื่องหนึ่ง ส่ง 10 Packet ติดต่อกันไปยังฝั่งตรงข้าม
  - ปัญหาด้าน Hardware ทำให้ Packet 3 ถูก Delay ไป
  - เส้นทางการส่งข้อมูลถูกเปลี่ยน เพื่อใช้เส้นทางที่ไม่ผ่านอุปกรณ์ที่มีปัญหา
  - Protocol ที่คอมพิวเตอร์ต้นทางส่ง Packet 3 ไปใหม่ จากนั้นส่ง Packet อื่นๆตาม จากนั้นจบการสื่อสาร

Chapter 26 TCP: 26.5.3 Techniques to Avoid Replay

- สองคอมพิวเตอร์ตกลงจะสื่อสารกันเมื่อเวลา 13.00 น.
- คอมพิวเตอร์เครื่องหนึ่ง ส่ง 10 Packet ติดต่อกันไปยังฝั่งตรงข้าม
- ปัญหาด้าน Hardware ทำให้ Packet 3 ถูก Delay ไป
- เส้นทางการส่งข้อมูลถูกเปลี่ยน เพื่อใช้เส้นทางที่ไม่ผ่านอุปกรณ์ที่มีปัญหา
- Protocol ที่คอมพิวเตอร์ต้นทางส่ง Packet 3 ไปใหม่ จากนั้นส่ง Packet อื่นๆตาม จากนั้นจบการสื่อสาร
- เวลา 13.05 น. คอมพิวเตอร์ทั้งสองตกลงที่จะสื่อสารกันใหม่
- หลังจากผู้ส่งคนเดิม ส่งไปได้สอง Packet แล้ว ตัว Packet 3 จากการสื่อสารครั้งแรกที่ถูก Delay มาถึงยังผู้รับ
- ต่อจากนั้น Packet ที่ 3 จากการสื่อสารครั้งที่สองมาถึงยังผู้รับ

- **Transport Protocol ต้องออกแบบเพื่อจัดการกับเรื่องนี้ มิฉะนั้นผู้รับจะได้ Packet 3 ที่ผิดพลาด ในขณะที่โยนทิ้ง Packet ที่ถูกต้อง**
  - Protocol จะต้อง Mark แต่ละ Session โดยใช้ ID ที่เฉพาะ ไม่ซ้ำกัน โดยการนำ ID กลับมาใช้ใหม่ต้องให้แน่ใจว่าจะไม่เกิด Replay (เวลาต้องห่างกัน)

Replay (Selective Reject)



Chapter 26 TCP: 26.5.3 Techniques to Avoid Replay

- **Replay ยังสามารถเกิดกับ Control Packet ได้เช่นกัน**
  - เช่น Connection Closing Packet ถูก Delay และไปถึงหลังจากมีการทำ Connection ใหม่ครั้งที่สอง

Chapter 26 TCP: 26.5.4 Flow Control to Prevent Data Overrun

- **ในการจัดการเรื่อง Heterogeneous End System ที่มีความสามารถไม่เท่ากัน ตัว Transport Protocol จะใช้วิธีการของ Flow Control เพื่อควบคุม**
  - Stop-and-go Protocol มักจะไม่ใช้ เนื่องจากมีประสิทธิภาพต่ำ
  - ปกติจะใช้ Sliding Window Flowcontrol
  - เช่นเดียวกัน จะคล้ายกับ Flow Control ใน Layer 2
    - การคำนวณ Efficiency จะคล้ายกับที่กล่าวมาแล้วใน Layer 2
    - ให้ดูใน Slide ก่อนหน้า หรือดูจาก CPE 326 Course Notes

Chapter 26 TCP: 26.5.4 Flow Control to Prevent Data Overrun

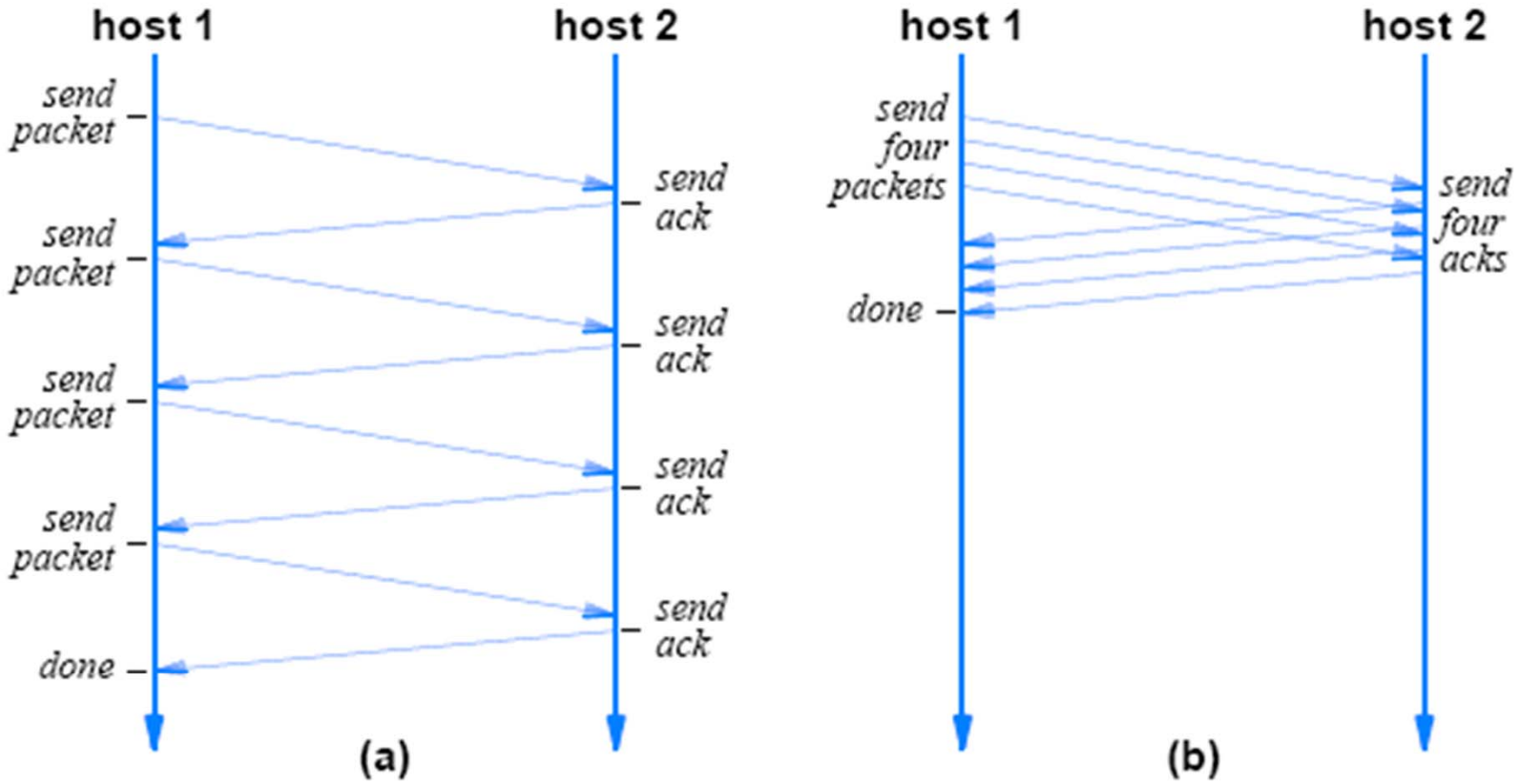
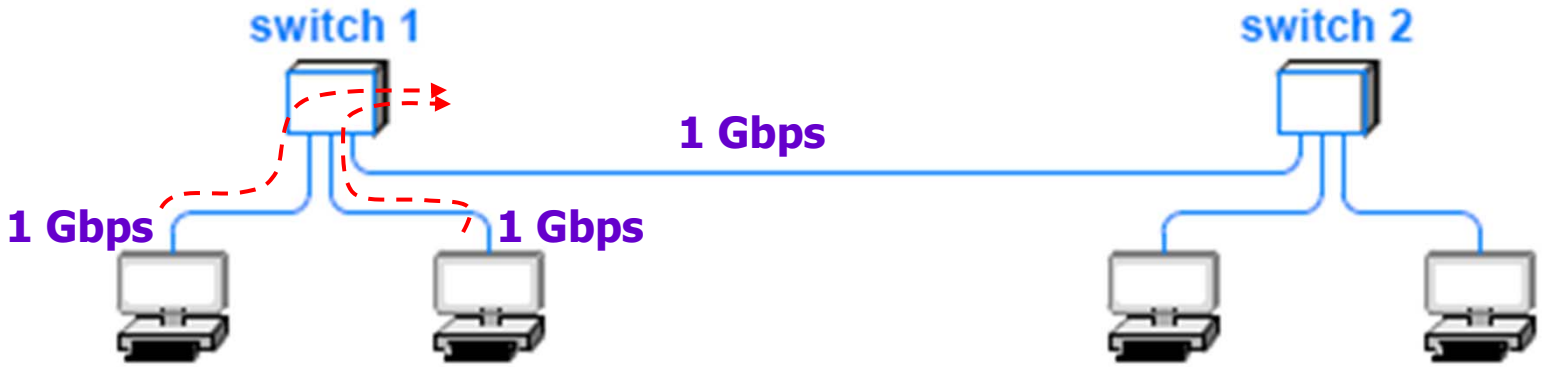


Figure 26.3 Comparison of transmission using (a) stop-and-go, and (b) sliding window.

Chapter 26 TCP: 26.6 Techniques To Avoid Congestion

- **Congestion เกิดเมื่อมีข้อมูลที่จะส่งมากเกินกว่าที่ Network จะรับได้**
  - อธิบายได้โดยใช้ Queuing Theory
  - Packet จะถูก Delay ไปมาก ในลักษณะ Exponential
  - ถ้า Queue ล้น Packet จะสูญหาย



Chapter 26 TCP: 26.6 Techniques To Avoid Congestion

- **Retransmission เนื่องจาก Queue ล้น จะทำให้เกิด Congestion Collapse**
  - Packet ที่ Retransmit จะถูกโยนทิ้งอีก เพราะ Queue ล้น
- **Congestion จะต้องตรวจจับได้อย่างทันท่วงที และทำการแก้ไข มิฉะนั้นจะเกิดผลเสียหายต่อทั้ง Network ซึ่งการตรวจจับของ Transport Protocol จะใช้พื้นฐานจากสองวิธี**
  - Intermediate System(Router) จะต้องแจ้งไปยังผู้ส่ง เมื่อมี Congestion
  - ผู้ส่งตรวจจับการเกิด Congestion เอง โดยดูจากค่า Delay หรือจำนวน Loss ของ Packet
    - ใน Internet จะใช้วิธีการนี้ เนื่องจาก Delay และ Loss ส่วนใหญ่ (ถ้าไม่ใช่เกิดจาก Hardware Failure) จะเป็นผลมาจาก Congestion
    - การแก้คือลดการส่งลง ทำได้โดยการควบคุมที่ Flow Control (ลดขนาดของ Window ลง)

Chapter 26 TCP: 26.7 The Art of Protocol Design

- แต่ละ **Technique** ที่กล่าวมาจะต้องถูกเลือกใช้อย่าระวัง และส่งผลต่อการออกแบบ **Protocol**
- **Protocol Header** จะต้องมี **Field** ที่จะรองรับข้อมูลต่างๆเพื่อควบคุมการทำงาน
- จะต้องแก้ปัญหาเรื่อง **Computer Reboot** ด้วย เพื่อป้องกัน **Duplicate Connection** หรือการทำ **Connection** ที่ไม่ถูกต้อง
- ในหัวข้อที่จะกล่าวต่อไป จะเป็นรายละเอียดเฉพาะที่ **TCP** เลือกใช้

Chapter 26 TCP: 26.8 Techniques Used In TCP To Handle Packet Loss

- **TCP ใช้การ Retransmission ในกรณีที่ Packet Loss**
  - ผ่านการทำ Positive Acknowledge ในการสื่อสารทั้งสองทิศทาง
  - มีการใช้ Transmission Timer
    - Transmission Timer ควรตั้งไว้เท่าไร
    - ต่ำเกินไป จะมีผลให้มี Duplicate Data ใน Network ที่ช้าเช่น Satellite Network
    - สูงเกินไป จะทำให้ประสิทธิภาพลดลง เช่นใน LAN
    - Congestion ที่เกิดขึ้นจะทำให้เกิด Delay และมีผลต่อการตั้ง Retransmission Timer
      - Ack อาจจะมี Delay เพิ่มในระดับ Magnitude ถ้ามี Congestion
  - Retransmission Timer ใน TCP จะต้องมีความสามารถปรับให้เหมาะกับ Delay ใน Network
    - ไม่เหมือนกับ Layer 2 Retransmission Timer ซึ่งเป็นระดับ Point-to-Point (Link) ที่สามารถคำนวณและกำหนดล่วงหน้าได้

Chapter 26 TCP: 26.8 Techniques Used In TCP To Handle Packet Loss

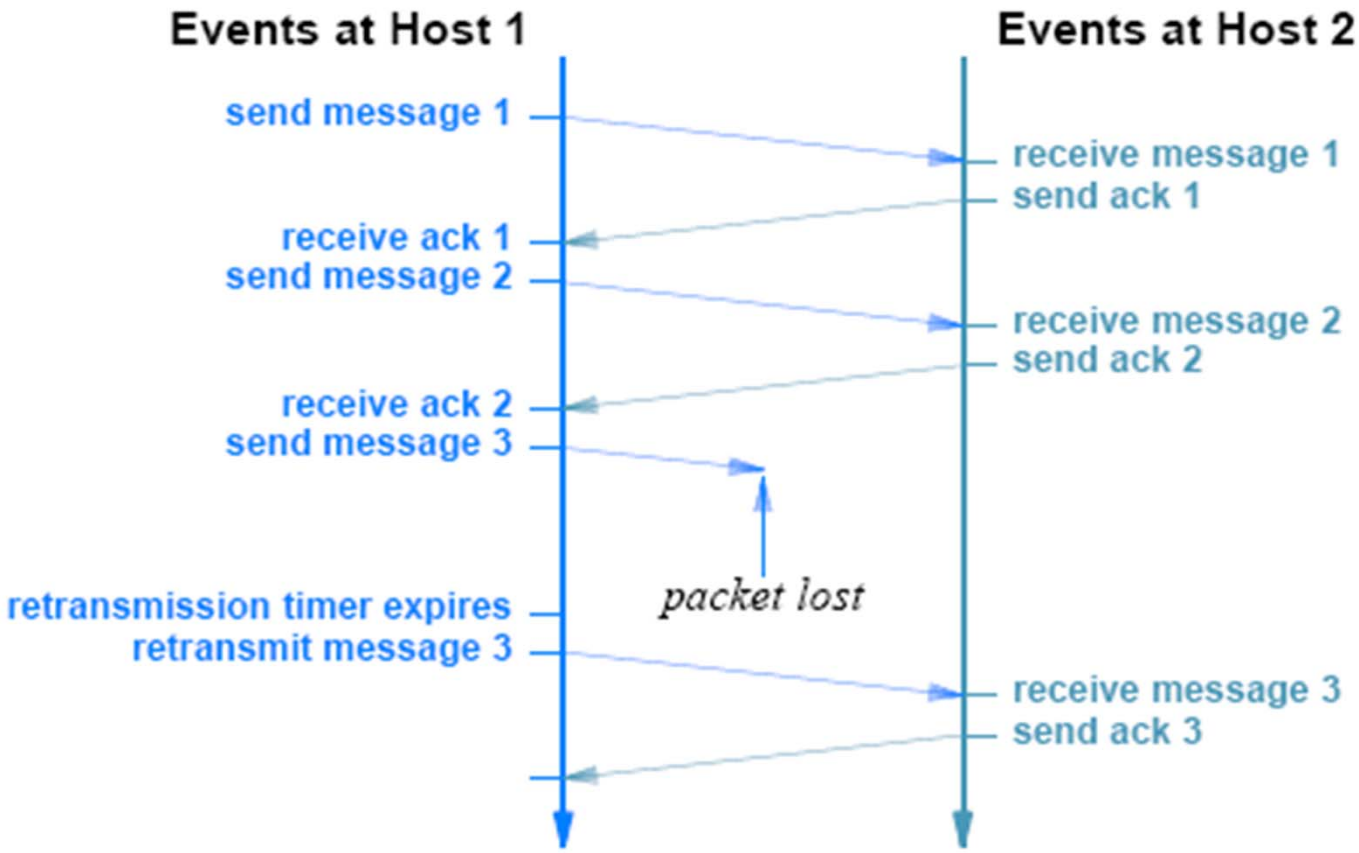


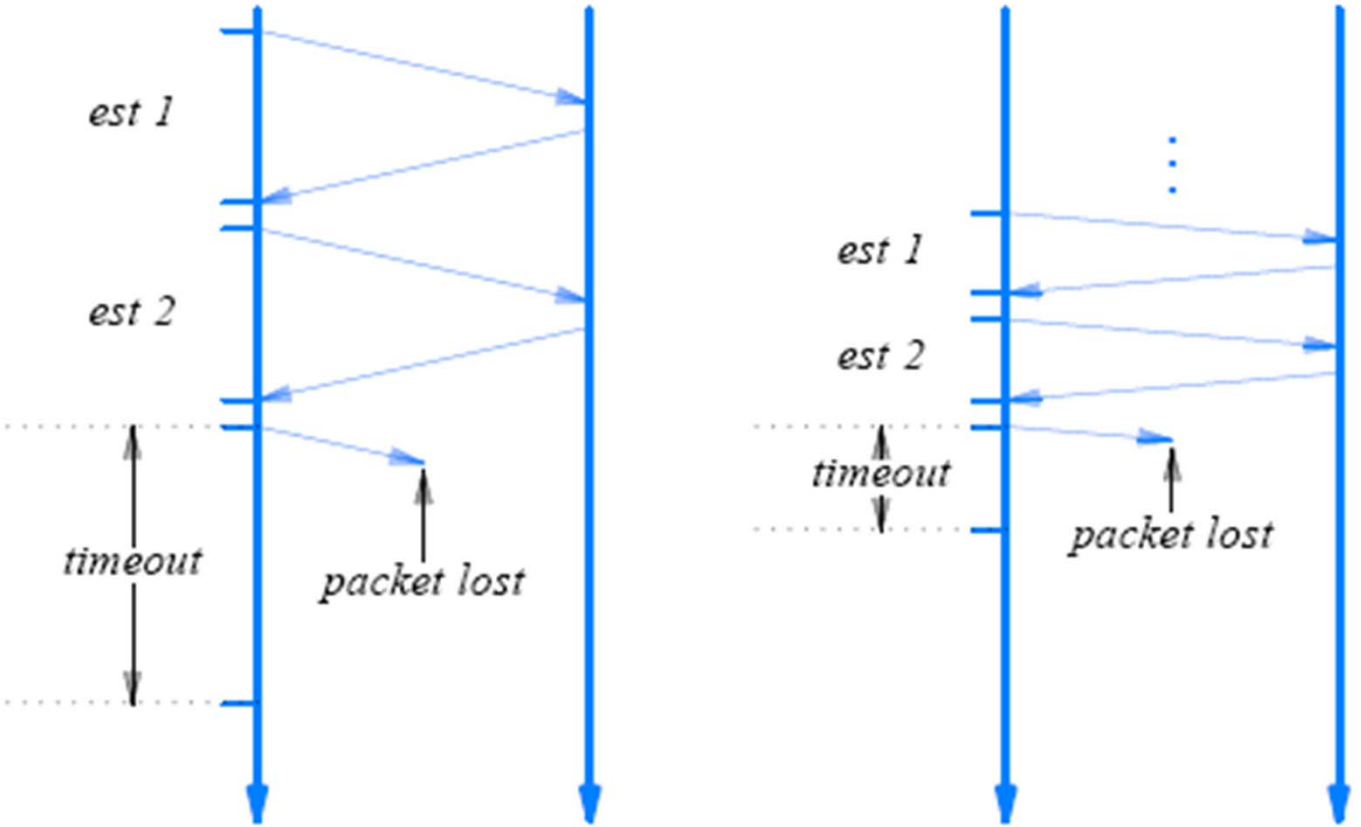
Figure 26.5 Illustration of TCP retransmission after a packet loss.

Chapter 26 TCP: 26.9 Adaptive Retransmission

- **ผู้ออกแบบ TCP เล็งเห็นว่า การตั้ง Retransmission Timer ที่คงที่จะมีผลต่อประสิทธิภาพที่ลดลงของ Internet**
  - เนื่องจาก Internet ประกอบด้วย Network หลายๆ Technology ที่มีค่า Delay ต่างกัน
- **TCP จะใช้ Adaptive Retransmission Timer ที่สามารถปรับค่าตามค่า Delay ของ Network**
  - TCP จะประมาณค่า Round-trip Delay จากเวลาที่ส่ง Packet ออกไปและได้รับคำตอบกลับมา เช่นตอนทำ Connection หรือช่วงการส่งข้อมูลและได้รับ ACK
    - มันจะคำนวณค่า Round-Trip Delay โดยใช้ Weighted Average ร่วมกับค่า Variance
    - ปกติ มันจะตั้งค่า Timer สูงกว่า Round Trip เล็กน้อย
    - เมื่อ Delay เริ่มมีการเปลี่ยนแปลง มันจะตั้งค่า Timer เพิ่มขึ้นที่สูงกว่าค่า Mean ที่คำนวณได้ ร่วมกับค่า Variance
    - การใช้ Weight Average จะทำให้สามารถ Reset Timer เมื่อ Delay กลับเข้าสู่ค่าปกติ

Chapter 26 TCP: 26.10 Comparison of Retransmission Timer

- พิจารณาจากการเกิด Packet Loss ของสอง Network ที่มี Delay ต่างกัน
  - ถ้า Delay มีค่าสูง TCP จะตั้งค่า Timer ที่สูงตาม



Chapter 26 TCP: 26.11 Buffers, Flow Control and Windows

- **TCP ใช้ Window Mechanism ในการควบคุมการไหลข้อมูล**
  - ต่างจาก Window Flow Control ใน Layer 2 ที่นับเป็น Frame แต่ TCP จะกำหนด Window เป็น Byte ของ Data ที่ส่งได้
  - หลังจากทำ Connection คอมพิวเตอร์ทั้งสองฝั่งจะกำหนดค่า Buffer (Memory) เพื่อใช้ในการรับ-ส่งข้อมูล
    - จากนั้นมันจะส่งขนาดของ Buffer ที่เหลือไปยังฝั่งตรงข้าม
    - เมื่อได้รับข้อมูล มันจะ Acknowledge ด้วยค่าของ Buffer ที่ยังสามารถรับได้ (Buffer ที่เหลือ)
    - ดังนั้นคำว่า Window ในความหมายของ TCP คือ จำนวน Buffer เป็น Byte ที่ยังเหลืออยู่ที่จะรับข้อมูลได้
      - เรียก Window Advertisement
    - ถ้าผู้ส่ง ส่งข้อมูลเร็วเกินไป ผู้รับไม่สามารถ Process ทัน มันจะ Acknowledge กลับมาด้วย Zero Window
      - ผู้ส่งต้องหยุดส่ง

Chapter 26 TCP: 26.11 Buffers, Flow Control and Windows(2)

- **ในตัวอย่าง ผู้ส่งใช้ขนาดของ Segment สูงสุด 1000 Bytes โดยกำหนด Initial Window Size 2500 Bytes**
  - ผู้ส่งส่งสาม Segment ขนาด 1000, 1000 และ 500 Bytes
  - ผู้รับ Process ไม่ทัน ส่ง Zero Window Advertisement
  - ผู้ส่งหยุดส่ง
  - ต่อมาผู้รัยทำการ Process 2000 Byte ของ Data จากนั้นทำการ Advertise ว่ารับได้อีก 2000 Byte เหนือจากที่รับแล้ว 2500 Byte
    - Window Size จะวัดต่อจาก Data ที่ทำการ Acknowledge เสมอ
  - ผู้ส่งส่งอีกสอง Segment แต่ละอันมีขนาด 1000 Byte
  - Window Size ลดลงเหลือศูนย์อีก ทำให้ผู้ส่งหยุดส่ง

Chapter 26 TCP: 26.11 Buffers, Flow Control and Windows(3)

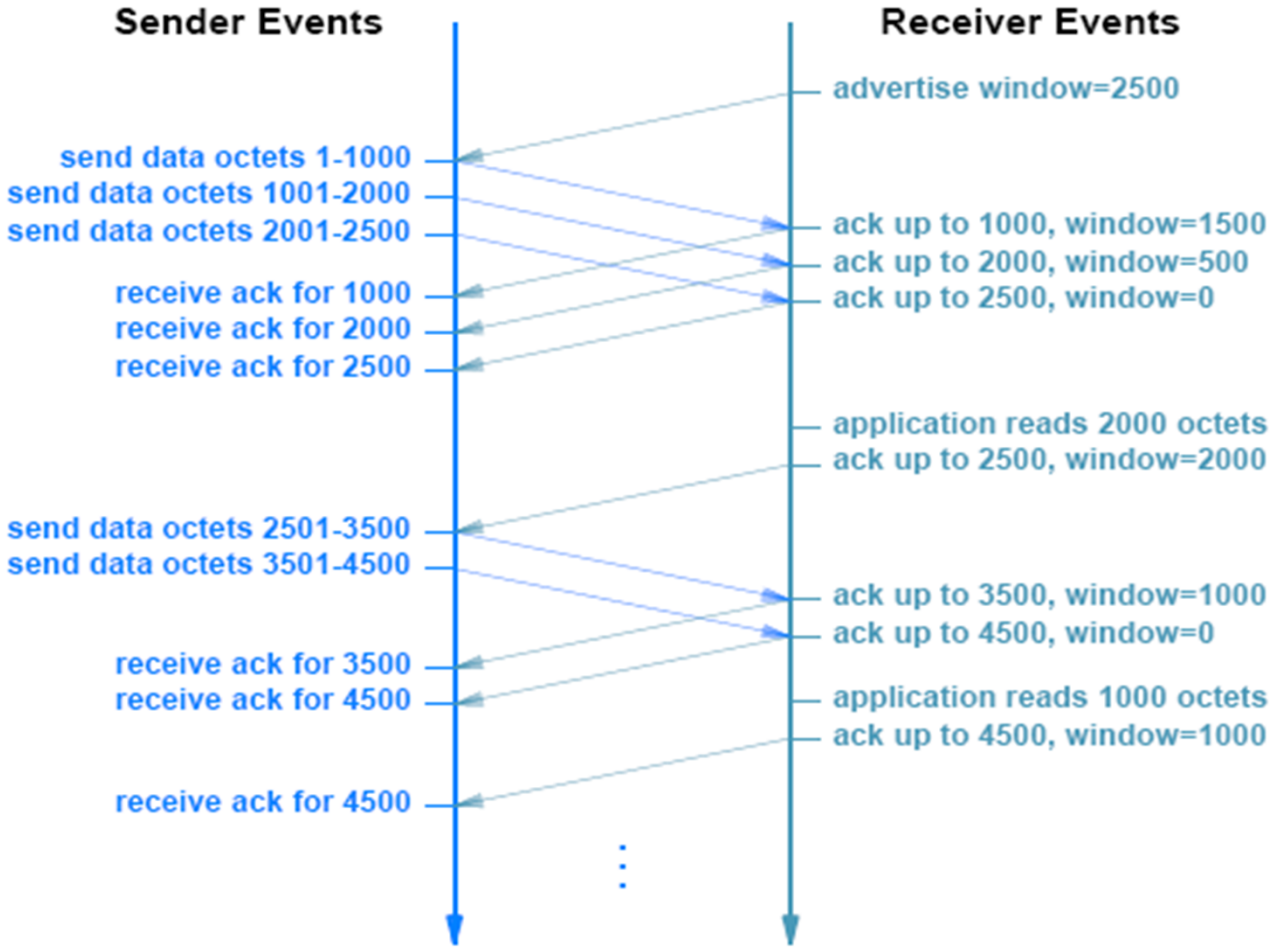


Figure 26.7 A sequence of messages that illustrates TCP flow for a maximum segment size of *1000* bytes.

Chapter 26 TCP: 26.12 TCP Three-Way Handshake

- **ในการที่จะให้แน่ใจว่าการทำ Connection และการ Terminate Connection ไม่ผิดพลาด TCP จะใช้วิธีการของ Three-Way Handshake**
  - ระหว่างการทำ Three-way Handshake จะมีการแลกเปลี่ยน Control Message และกำหนดขนาดของ Buffer
  - Three-Way Handshake จะสามารถจัดการกับกรณีที่เกิด Packet Loss, Duplicate, Delay และ Replay ได้ และจะ Guarantee ว่า Connection จะสร้างและจบลง เมื่อทั้งสองฝ่ายตกลงกัน
- **Control Message ที่ใช้ระหว่างทำ Three-Way Handshake Connection เรียก Synchronization Segment (SYN Segment) และเรียก Finish Segment(FIN Segment) สำหรับ Control Message ตอนทำ Termination**
- **ตอนทำ Three-Way Handshake จะมีการเลือก Sequence Number**
  - แต่ละด้านจะเลือกเลข 32 Bit แบบ Random เป็น Sequence เริ่มต้น ของ Data
  - ถ้าเกิดการ Reboot และ Application พยายามจะทำ Connection ใหม่ โอกาสที่จะได้ Sequence เดิมจะน้อยมาก ดังนั้นปัญหา Replay จะเลี่ยงได้
- **ในกรณีที่ทำ Close Connection จะมีการส่ง FIN ร่วมกับ Acknowledge เพื่อให้แน่ใจว่า Data ได้รับครบถ้วนก่อนจะปิด Connection**

Chapter 26 TCP: 26.12 TCP Three-Way Handshake(Open Connection)

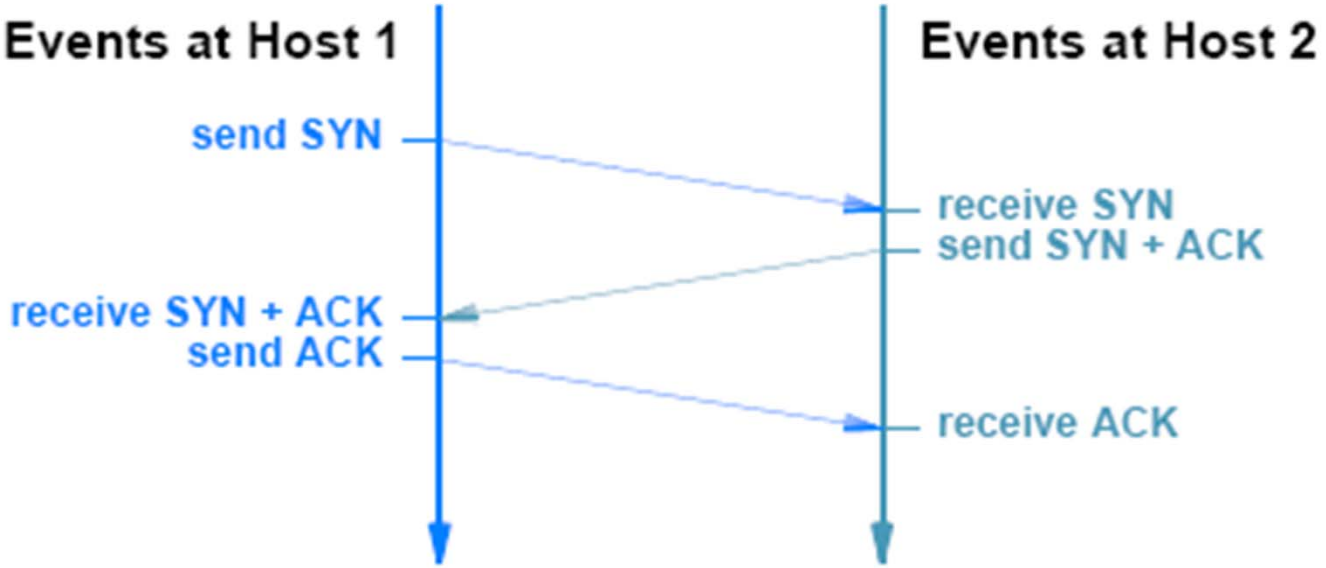


Figure 26.8 The 3-way handshake used to create a TCP connection.

Chapter 26 TCP: 26.12 TCP Three-Way Handshake(Close Connection)

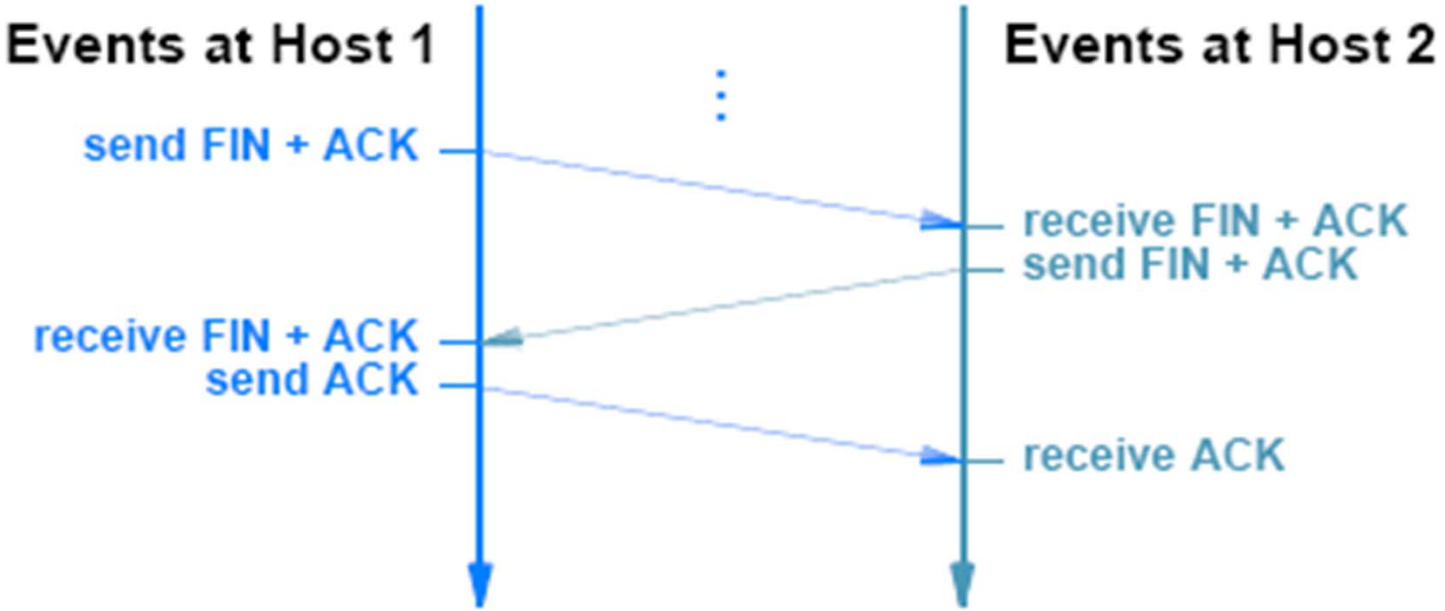


Figure 26.9 The 3-way handshake used to close a connection.

Chapter 26 TCP: 26.13 TCP Congestion Control

- **TCP มี Congestion Control ที่ดูเหมือนประหลาดกว่าทั่วไป**
  - เนื่องจาก Congestion ทำให้เกิด Delay ส่งผลให้เกิด Retransmission ส่งผลกลับมาให้ Congestion หนักขึ้น คือเกิด Congestion Collapse
  - แม้ว่าการจัดการกับ Congestion ทั่วไปจะใช้วิธีลดอัตราการส่ง แต่ TCP ไม่สามารถคำนวณอัตราการส่งได้ เพราะวัดการส่งจาก Buffer ที่ได้รับจากอีกฝั่งหนึ่ง
  - ดังนั้นการควบคุมอัตราการส่งใน TCP จะควบคุมจากการเปลี่ยนแปลงขนาดของ Window แทน
    - การลดขนาด Window ลงชั่วคราว จะเป็นการลดอัตราการส่งข้อมูลไปในตัว
    - เมื่อมี Data Loss (คือ Congestion ในมุมมอง TCP) ขนาด Window จะถูกลดลง

Chapter 26 TCP: 26.13 TCP Congestion Control(2)

- **TCP มี Congestion Control หลาย Algorithm ที่นิยมคือวิธีของ 'Slow Start'**
  - เมื่อเริ่ม Connection ใหม่ หรือกรณีที่มี Message Loss มันจะเริ่มจากการส่งหนึ่ง Message
    - ถ้ามันได้รับ ACK มันจะส่งสอง Message
    - ถ้าได้รับ ACK ทั้งสอง มันจะส่ง 4 Messgae
    - และจะส่งเป็นสองเท่าตราบใดที่ได้รับ ACK ครบ เรื่อยไปจนถึงส่งเท่ากับครึ่งหนึ่งของ Window Size ที่ Advertise จากผู้รับ
      - จากนั้นมันจะเพิ่มการส่งแบบ Linear ตราบเท่าที่ไม่มีการ Congestion
    - ถ้ามี Congestion จะเริ่มส่งทีละ Message ใหม่
      - ดังนั้นทุกคนจะหยุดส่งถ้าเกิด Congestion และป้องกันการเกิด Congestion Collapse
  - Congestion Control จะใช้ร่วมกับ Congestion Avoidance ซึ่งมี Algorithm มากมาย

Chapter 26 TCP: 26.14 TCP Segment Format

- มี **Format** เดียว ใช้กับ **Message, Acknowledge, SYN, FIN**
  - แต่ละ Message ของ TCP เรียก Segment
  - แต่ละ Segment สามารถส่ง Data, Acknowledge ของ Data ที่ได้รับ, Window Advertisement
    - Acknowledgement Number และ Window Field หมายถึง Data ที่ได้รับ คือ Sequence Number เริ่มต้นของ Data ถัดไปที่ต้องการ และ Buffer ที่เหลือ ถ้า Data ที่ได้ไม่เป็น Order มันจะส่ง Ack Number ของ Data ที่คาดหวังซ้ำๆจนกว่าจะได้รับ
    - Sequence Number หมายถึง Sequence Number ของ Byte แรกของ Data ที่ส่ง
    - Checksum จะเป็น Checksum ของ Header และ Data
    - Code Bit บ่งบอกว่าเป็น Segment ชนิดใหน (Data, SYN, FIN)

Chapter 26 TCP: 26.14 TCP Segment Format

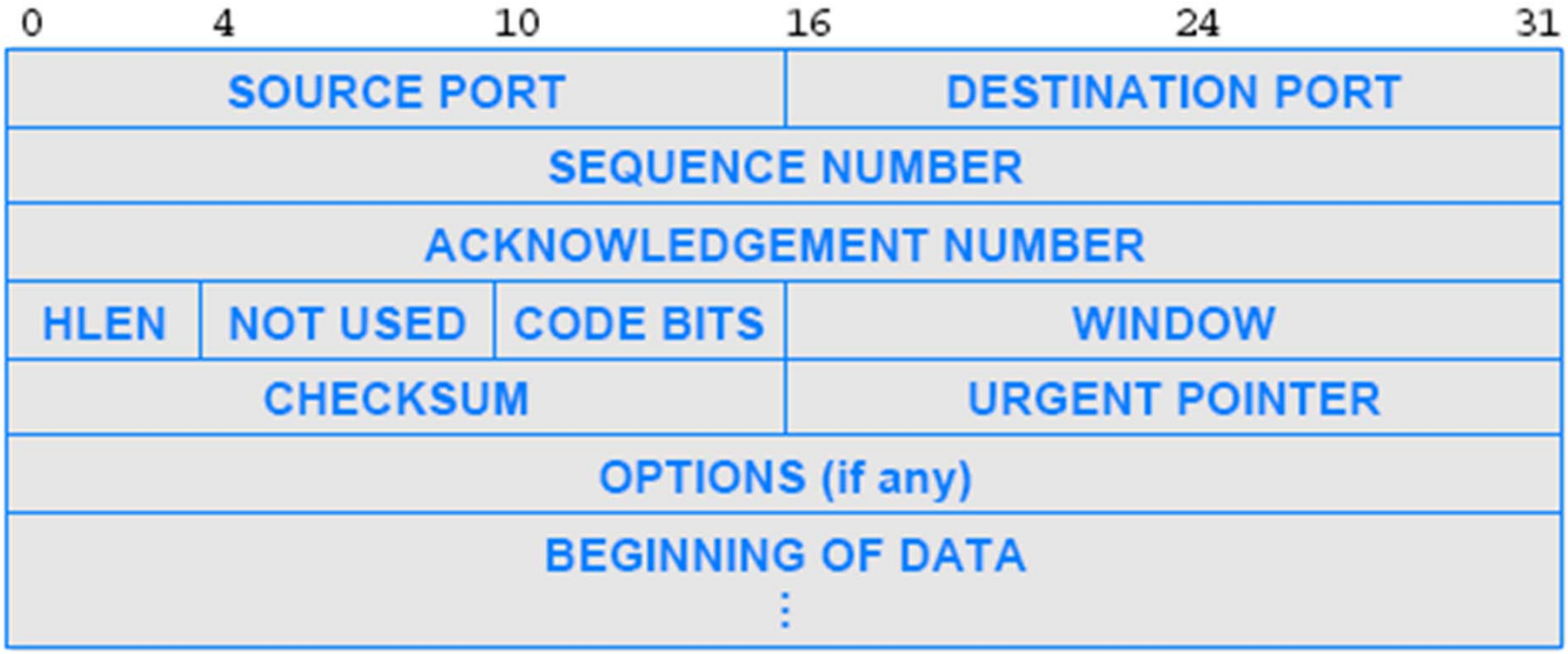


Figure 26.10 The TCP segment format used for both data and control messages.

TCP Port

- **เช่นเดียวกันกับ UDP Port**
  - Well-Known Port: 0-1023
    - FTP: 20/21 (Data/Control)
    - Telnet: 23
    - SMTP: 25
    - DNS: 53 (UDP/TCP)
    - BOOTP: 67/68 (Server/Client) (Mostly UDP)
    - HTTP: 80
  - Registered Port: 1024-49151
  - Dynamic Port: 49152-65535

End of Week 11

- **HW 6: UDP และ TCP**
  - ให้ Download Question จาก Web
  - ส่ง สัปดาห์หน้า
- **Next Week, Week 12**
  - Routing Part I: Routing Mechanism