CSC4200/5200 – COMPUTER NETWORKING

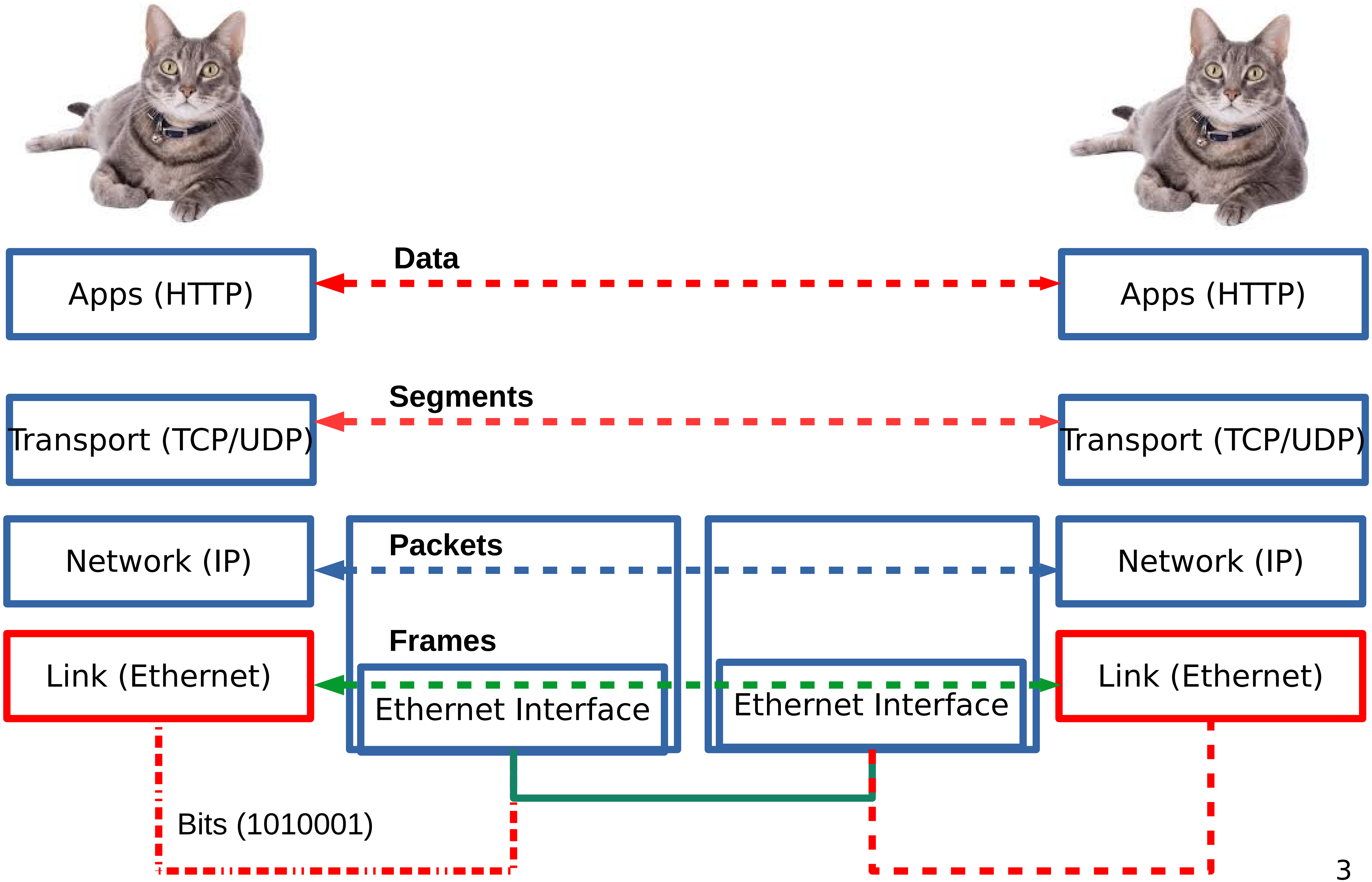
CONNECTING MACHINES TO A NETWORK

Instructor: Susmit Shannigrahi
sshannigrahi@tntech.edu

Tennessee TECH

Recap

- What is a network?
- What is the Internet?
- What is a packet?
- What is packet switching?
- How many layers in the current Internet stack?
- Architecture vs Protocol
- Three components of latency?
- Bandwidth-delay product?
- What is jitter?


Data
Apps (HTTP)
Apps (HTTP)
Segments
Transport (TCP/UDP)
Transport (TCP/UDP)
Packets
Network (IP)
Network (IP)
Frames
Link (Ethernet)
Ethernet Interface
Ethernet Interface
Link (Ethernet)
Bits (1010001)

Network Adaptors and Mac Addresses

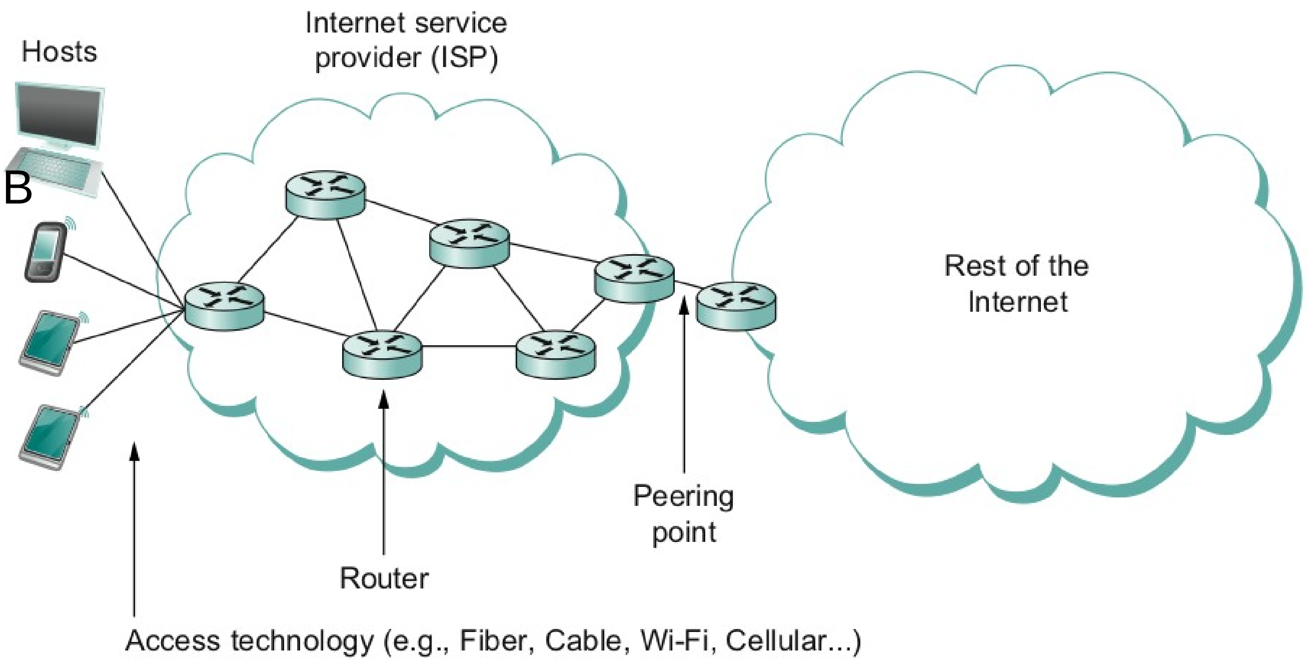
Media access control address (MAC)

A unique identifier assigned to a network interface controller (NIC)

Globally unique

What does it take to create a link?

Send a number of bits from Host A to Router B over Interface XX:YY:ZZ:AA:BB:CC



- Common abstractions
  - Why?

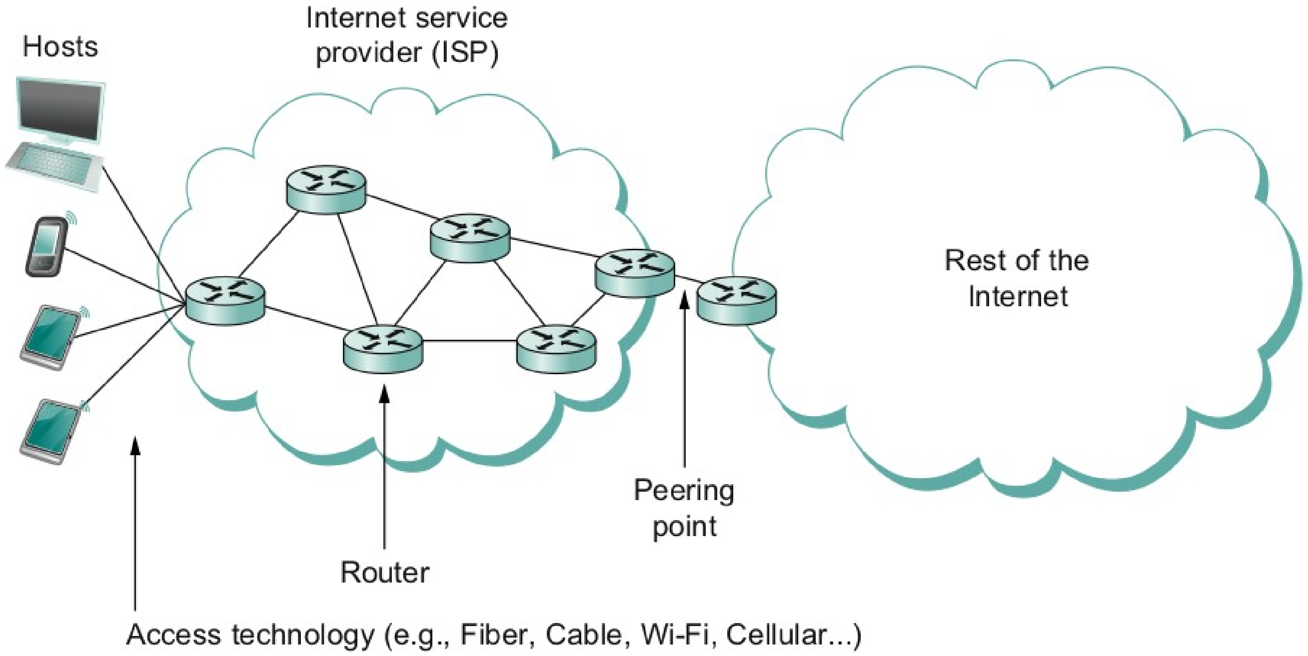
Two Steps to a Link

- Create a physical medium between nodes **(wire, fiber, air!)**

- Make it carry bits

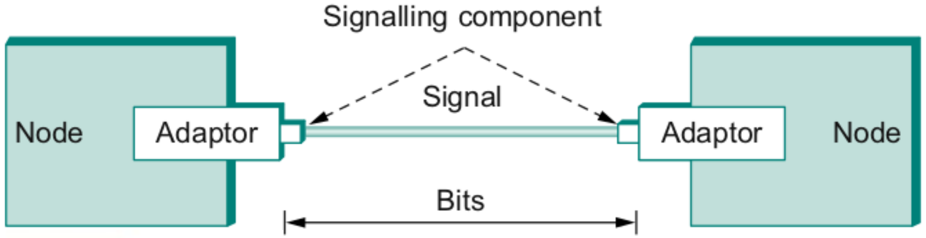
  - **Encoding** bits so that the other end understands **(encoding)**
  - Create bag of bits to create messages (**framing**)
  - Detect errors in frames (**error detection**)
  - Deal with lost frames **(reliable delivery)**
  - Create shared access to link, e.g, WiFi **(media access)**

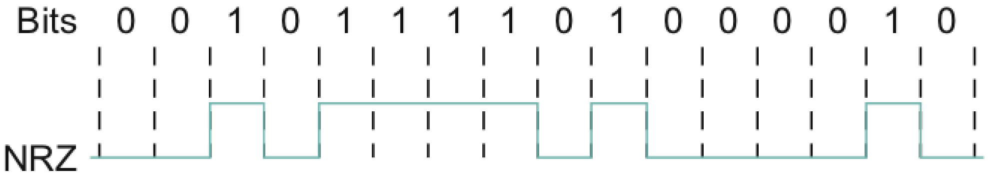
Step 1 - Create the Physical Link



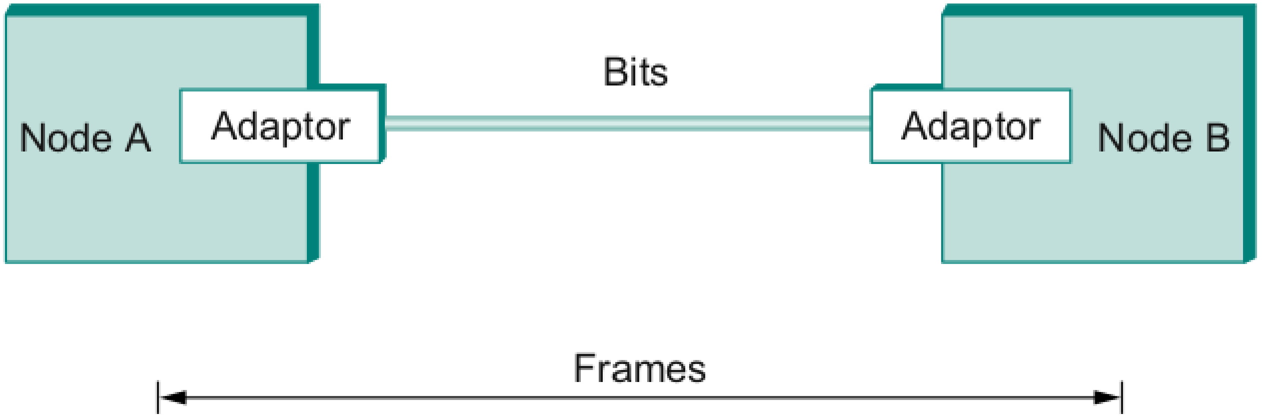
Step 2.1 - Encoding

- Bit pattern - 0101001
  - Must encode it into electrical signals and then decode it on the other end!





Step 2.2 – Create Frames – bag of bits



- Bits - between adaptors
- Frames – between hosts (two computers want to exchange messages)
  - The job of an adaptor is to find frames in a bit sequence

- Frames are link layer protocols

Step 2.2 - Framing

- Point-to-point
  - Special start of text character denoted as Flag
    - 0 1 1 1 1 1 1 0
  - Address, control : default numbers
  - Protocol for demux : IP / IPX
  - Payload : negotiated (1500 bytes)
  - Checksum : for error detection

Step 2.3 - Error Detection

- Bit errors are introduced into frames
  - Because of electrical interference and thermal noises
- Detecting Error
- Correction Error
- Two approaches when the recipient detects an error
  - Notify the sender that the message was corrupted, so the sender can send again.
    - If the error is rare, then the retransmitted message will be error-free
  - Using some error correct detection and correction algorithm, the receiver reconstructs the message

Error Detection

- Common technique for detecting transmission error
  - CRC (Cyclic Redundancy Check)
    - Used in HDLC, DDCMP, CSMA/CD, Token Ring
  - Other approaches
    - Two Dimensional Parity (BISYNC)
    - Checksum (IP)

Error Detection

- Basic Idea of Error Detection
  - To add redundant information to a frame that can be used to determine if errors have been introduced

0	1	0	1	0	**0**

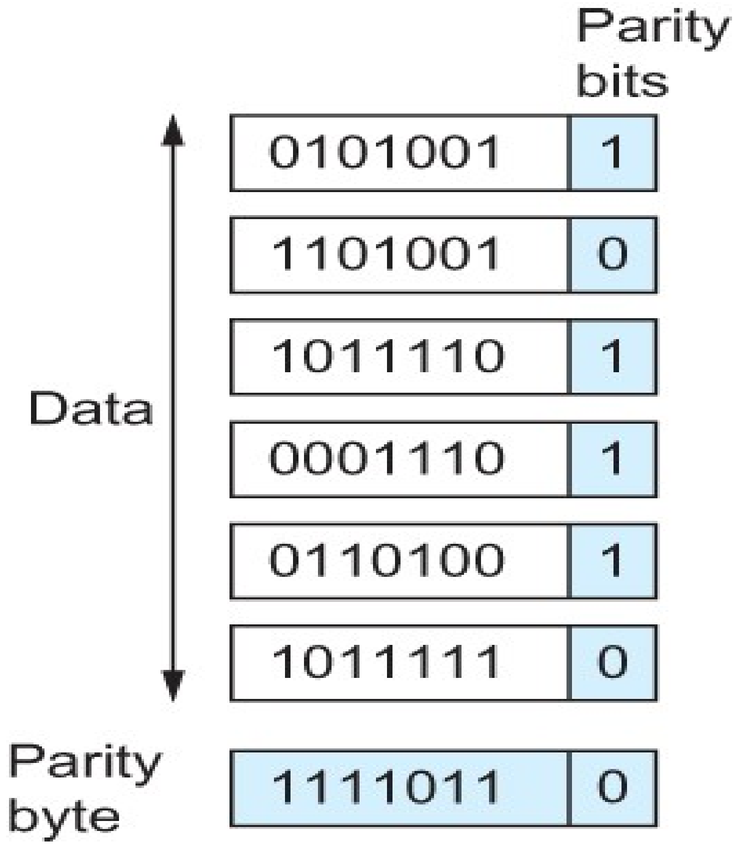
0	1	0	1	1	**1**

Number of 1s
- Odd 1s = Parity bit 1
- Even 1s = Parity bit 0

Two-dimensional parity

- Two-dimensional parity does a similar calculation
- Extra parity byte for the entire frame, in addition to a parity bit for each byte
- Two-dimensional parity catches all 1-, 2-, and 3-bit errors and most 4-bit errors

Two-dimensional parity



Two Dimensional Parity

Number of 1s

- Odd 1s = Parity bit 1
- Even 1s = Parity bit 0

Do it both horizontally and vertically

Others - Cyclic Redundancy Check (CRC)

- Reduce the number of extra bits and maximize protection
- N+1 bit message is N degree polynomial

  10011010 →

  $Msg(x) = (1 \times x^7) + (0 \times x^6) + (0 \times x^5) + (1 \times x^4) + (1 \times x^3) + (0 \times x^2) + (1 \times x^1) + (0 \times x^0)$

- $Msg(x) = x^7 + x^4 + x^3 + x^1$

Others - Cyclic Redundancy Check (CRC)

- Msg(x)=$x^7+x^4+x^3+x^1$
- Pick a divisor polynomial (from a table)
  C(x) = $x^{3+}x^2+1$
- Divide M(X) by C(x) → subtract the reminder from M(x)
  - Gives you M’(X)
  - You can do this by performing a logical XOR
- Send M’(x)and C(x) to the recipient
  - If the result is 0, you received a good copy

Others - Cyclic Redundancy Check (CRC)

- Msg(x)=$x^7+x^4+x^3+x^1$
- Pick a divisor polynomial (from a table)
  C(x) = $x^{3+}x^2+1$
- Divide M(X) by C(x) → subtract the reminder from M(x)
  - Gives you M’(X)
  - You can do this by performing a logical XOR
- Send M’(x)and C(x) to the recipient
  - If the result is 0, you received a good copy

Others - Cyclic Redundancy Check (CRC)

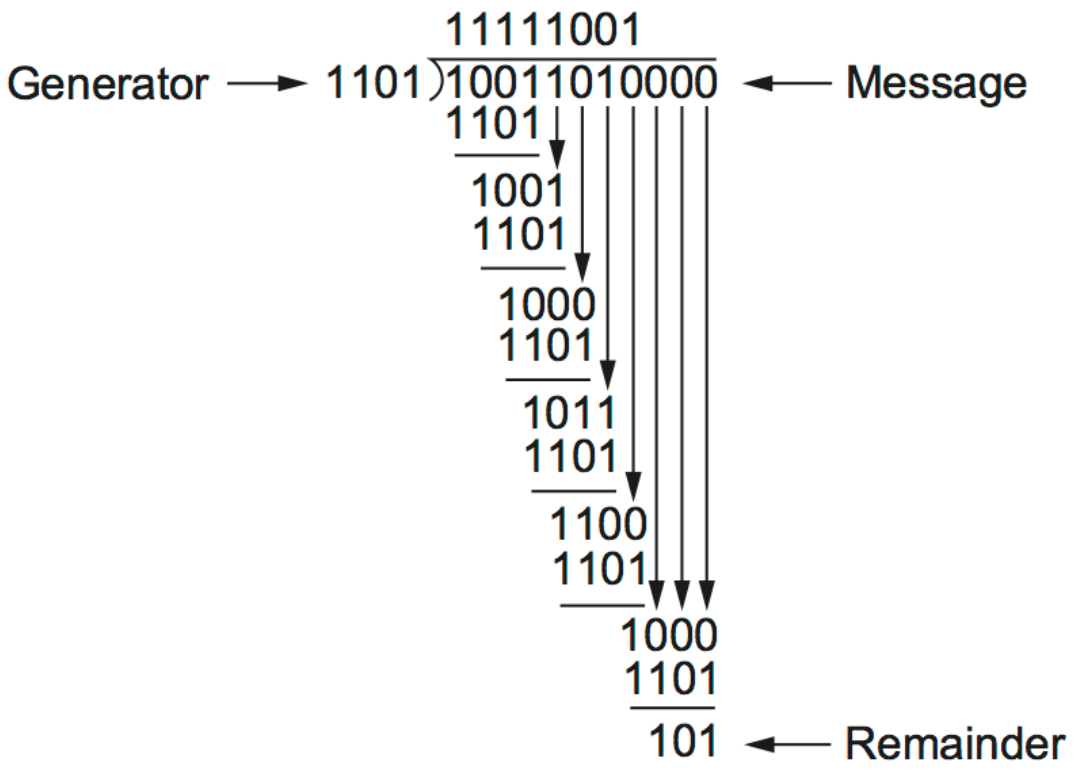
1. Msg(x)=10011010 = $x^7+x^4+x^3+x^1$

2. add k zeros at the end of the message, 3 in this case.
10011010000 ← T(x)

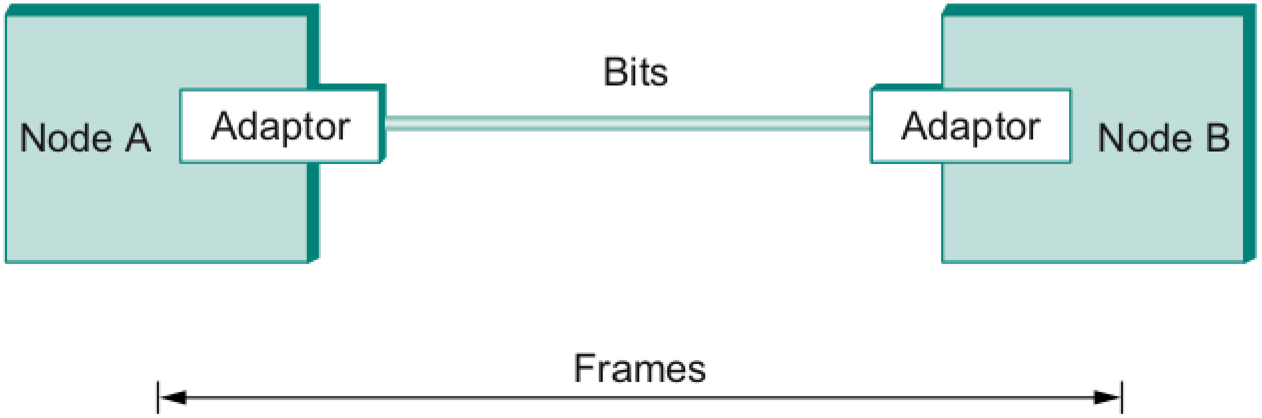
3. Pick a c(x) → x^3+x^2+1.

4. T(x)/c(x) → Reminder 101.

5. 101 will act as the value of CRC (generally)



Frames



- We are still sending frames between hosts!
- Shortcomings of error correction/detection?

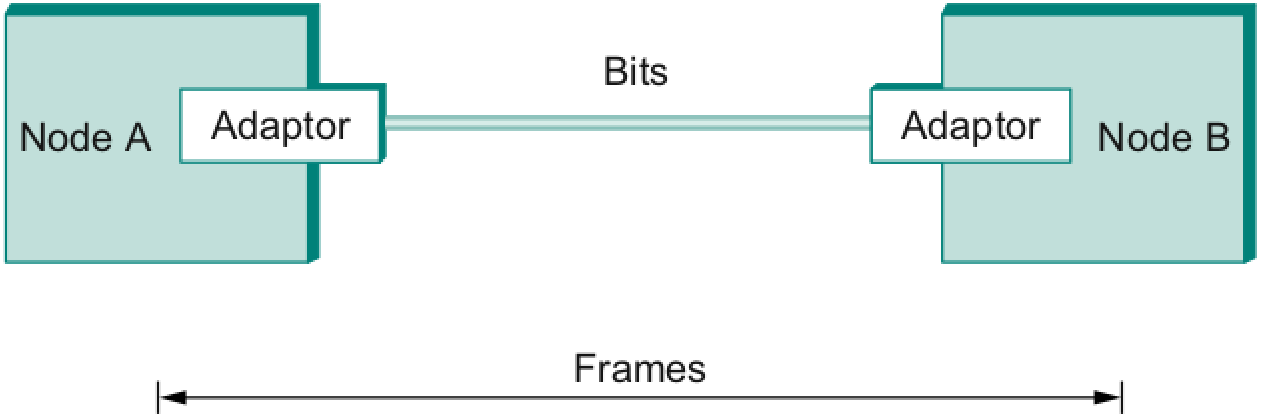
Step 2.4 - Reliable Delivery

- Frames might get lost
  - Too many bits lost
  - Clock did not sync properly
  - Error detected but the report got lost

- Can we build links that does not have errors?
  - Not possible

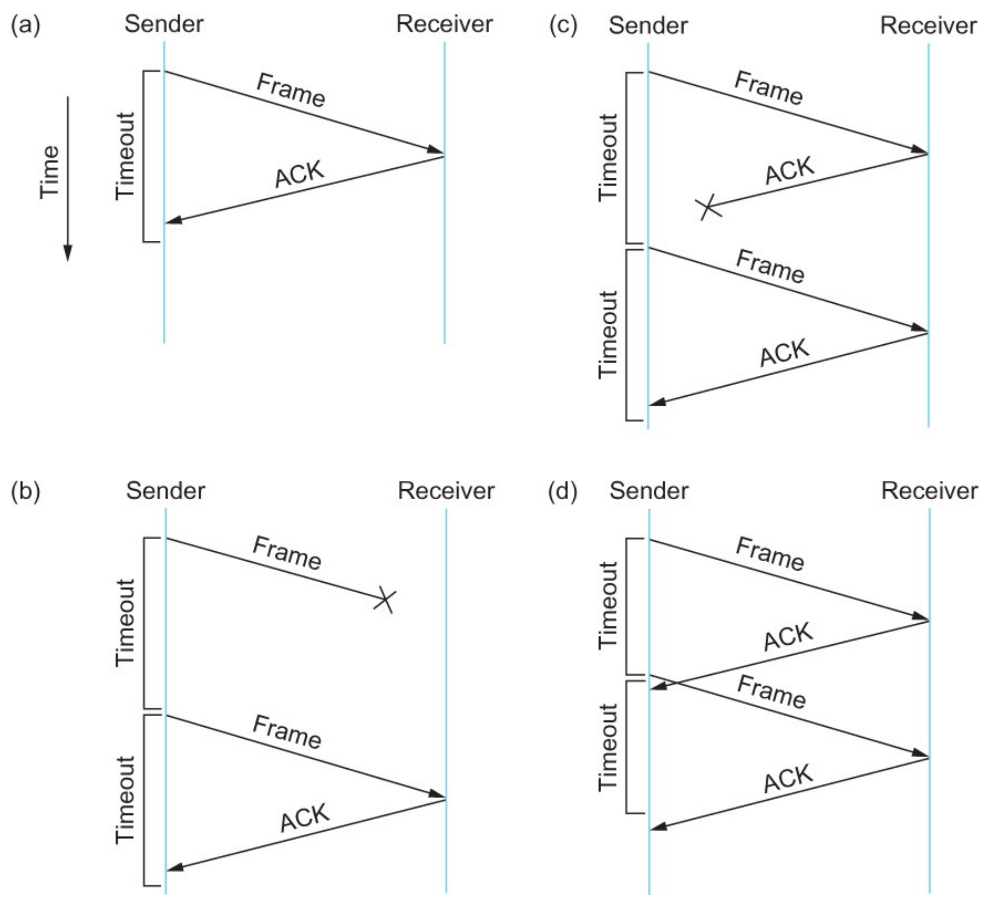
- How about all those error correction stuff we learned?
  - Can we add them to frames?
  - We could, but think of the overhead
  - What happens when the entire frame is lost?

Frames – bag of bits



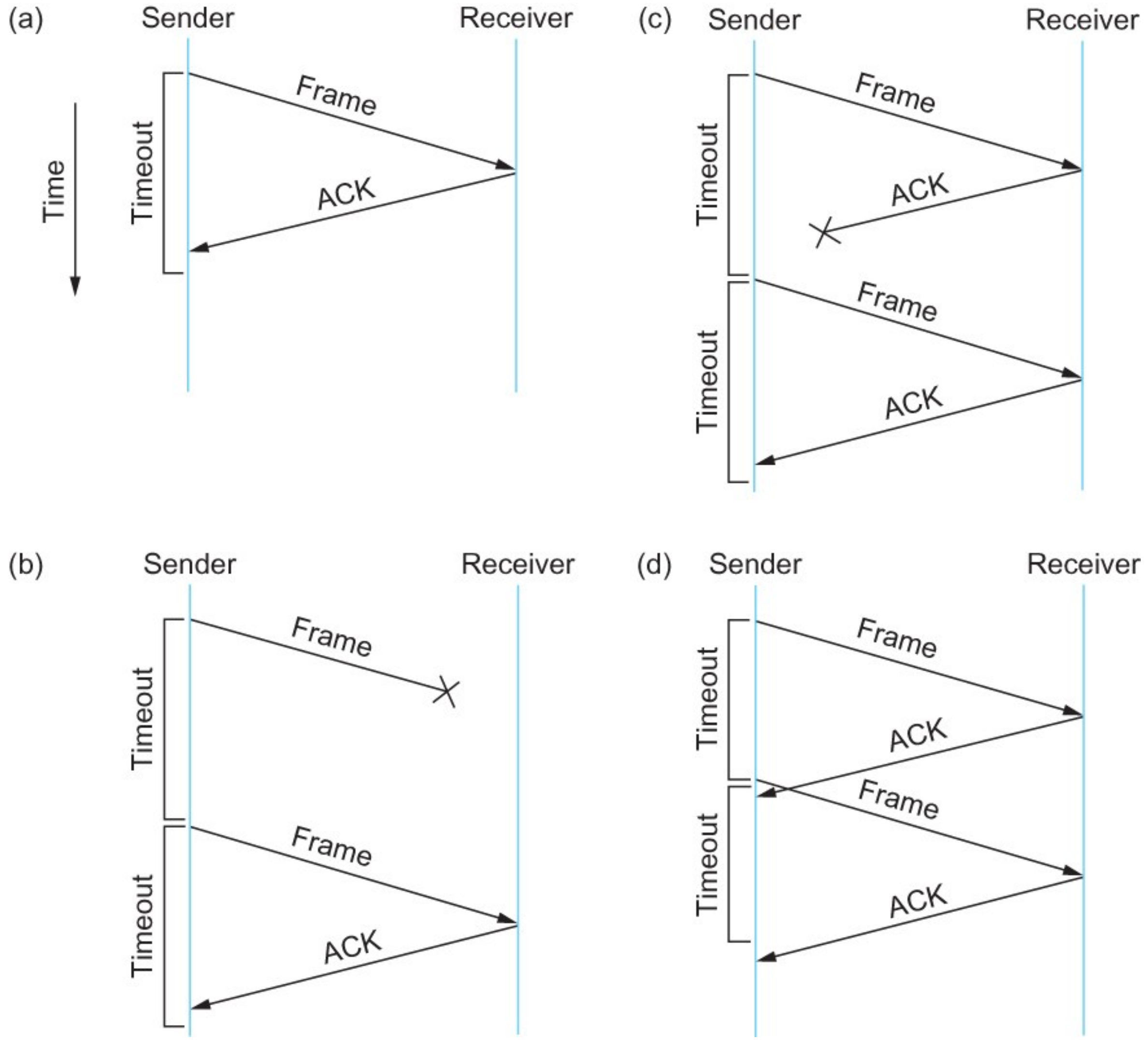
- Sending side – encapsulation, add error check bits, flow control
- Receiving side – extract frames, check for error, flow control

Stop and Wait



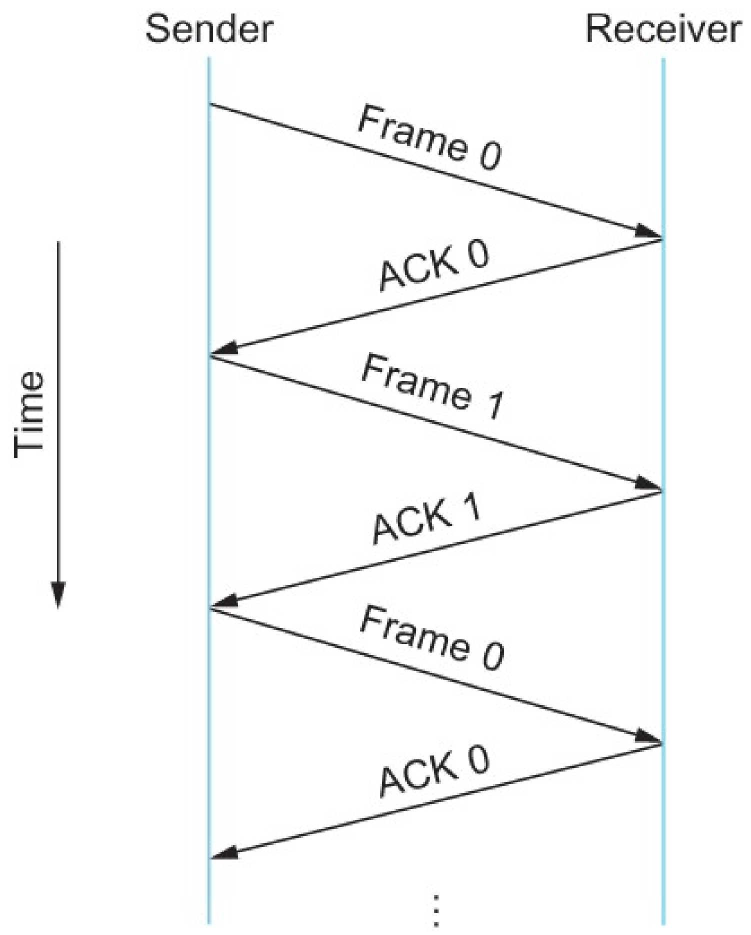
- Sender sends a frame, sets a timeout (e.g., 1 sec)
- Receiver receives the frame, sends an ACK
- Sender
  - sends the next frame on ACK
  - retransmits the same frame if timeout happens
- **Spot the bugs in the protocol**

Stop and Wait – Bugs (C and D)

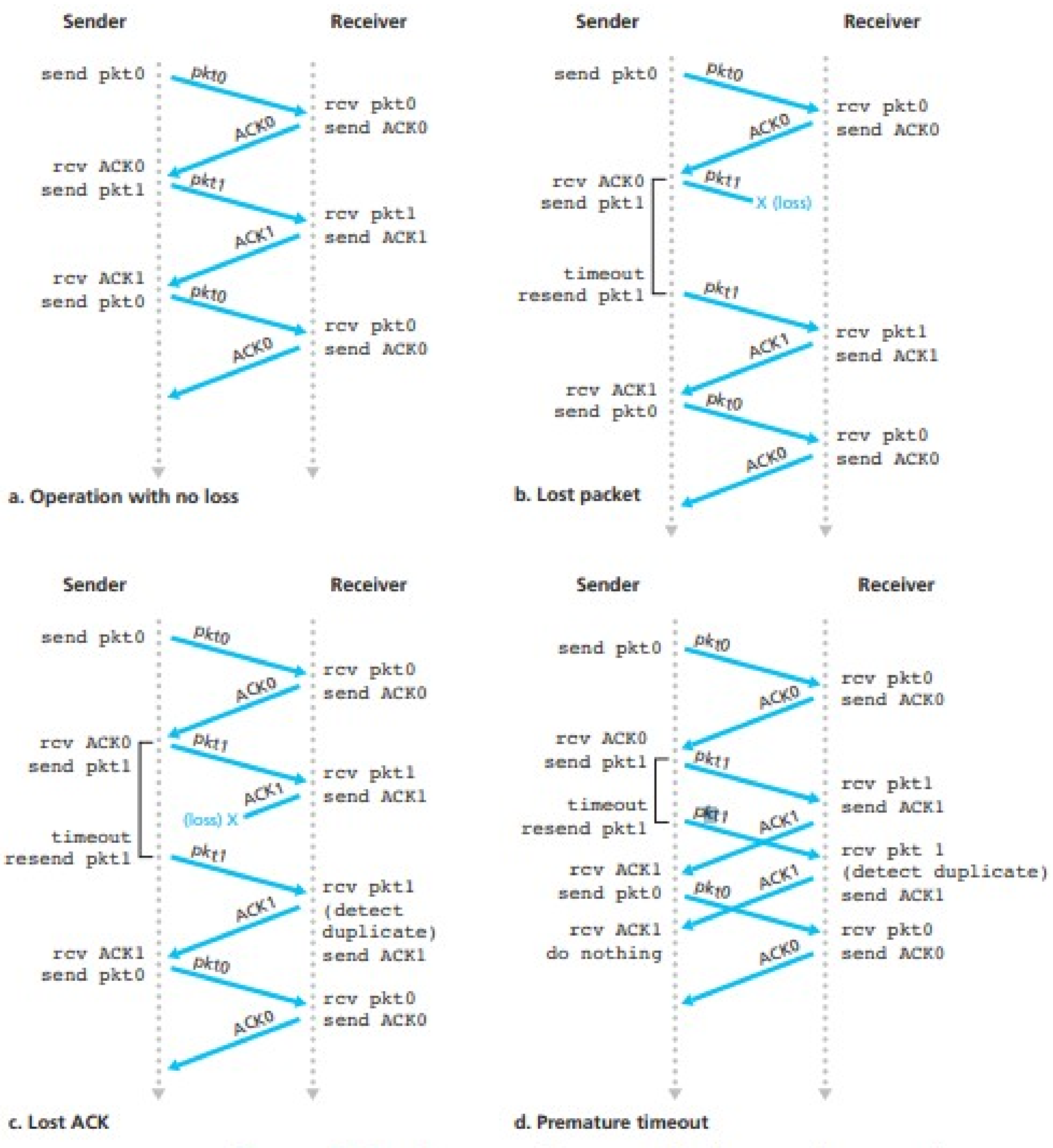


Stop and Wait – How to fix the bug?

Hint: Uniquely identify each packet



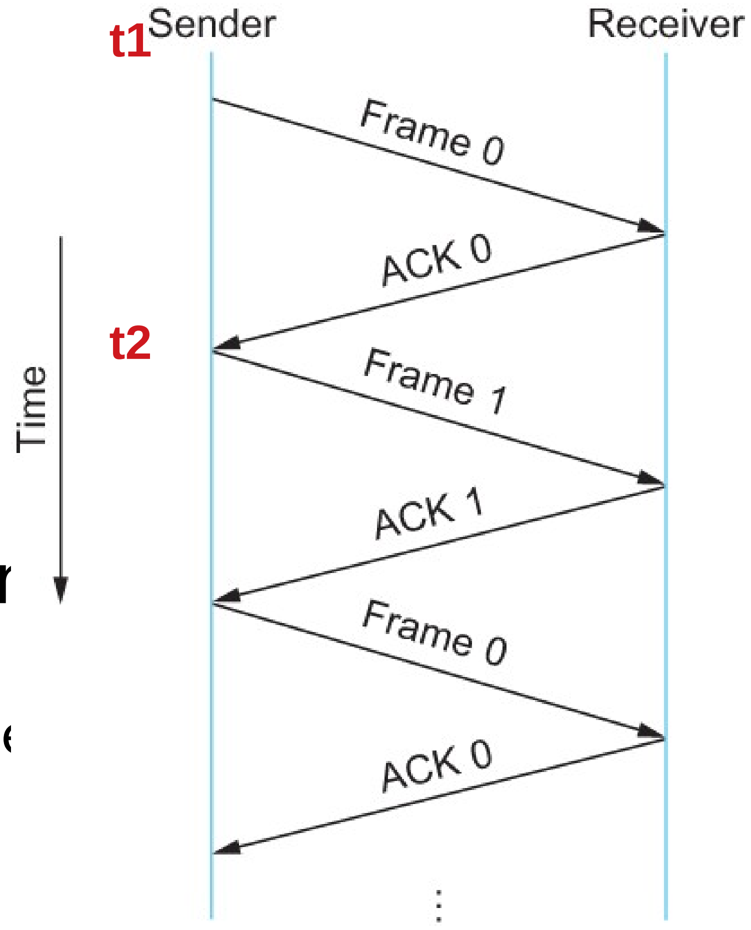
Stop and Wait v2



Stop and Wait - V2 Problems

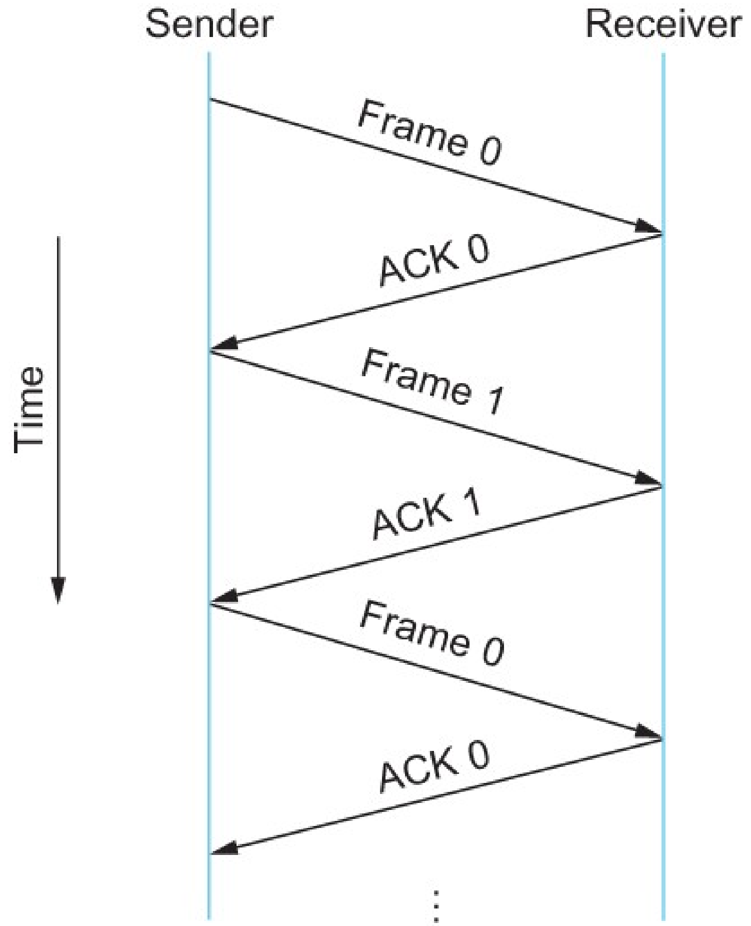
- Sender sets a timeout to wait for an ACK
  - Too small – retransmissions
  - Too large – long wait if frames are lost

- Solution:
  - Keep a running average of Round Trip Tir
  - EstimatedRTT = (1 – α) • EstimatedRTT + α • Sampl
  - Timeout = 2*EstimatedRTT
  - Value of α = 0.125
  - Where does α come from? RFC 6928 (for now)



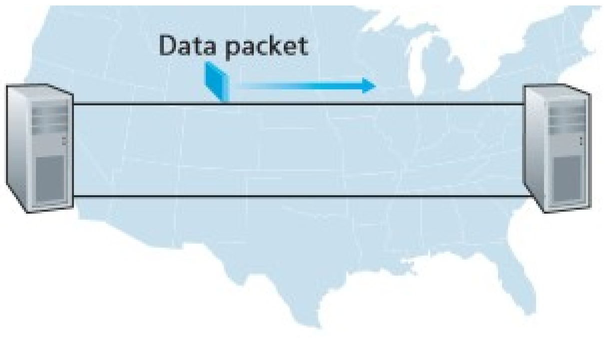
Stop and Wait – How to fix the bug?

Hint: Uniquely identify each packet



Stop and Wait – How does it perform?

- Bandwidth (R)= 1Gbps
- Packet size (L) = 1000 bytes
- RTT = 30ms
- T_{trans} = L/R = 8000bits/10^9bits/sec = 8microsecond
- T_{prop} = 15ms
- Total Delay = 15.008 ms

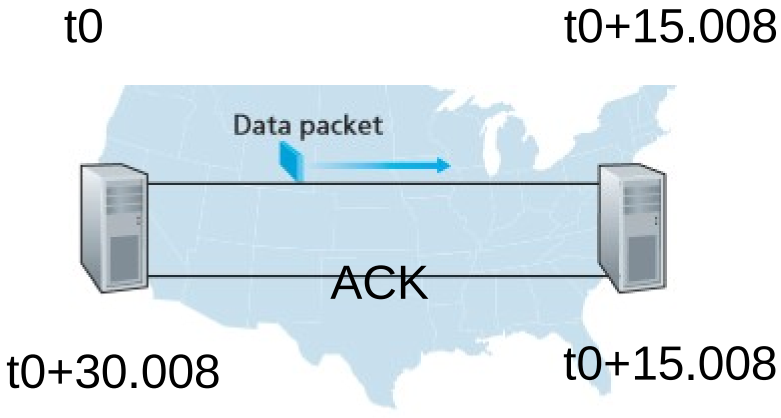


a. A stop-and-wait protocol in operation

Kurose/Ross

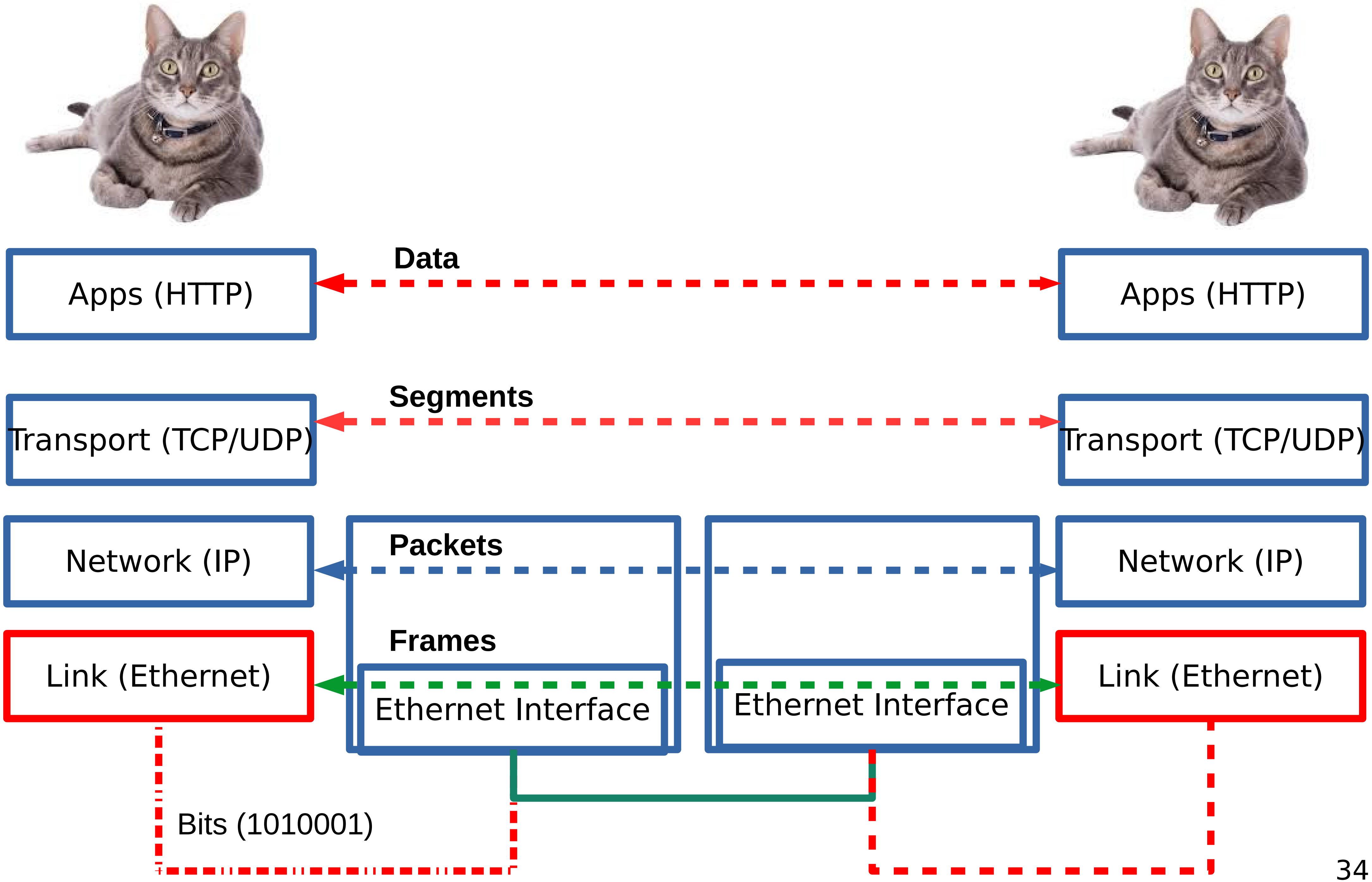
Stop and Wait – How does it perform?

- Sender transmits for only 0.008 ms in 30.008ms
- Utilization = 0.008/30.008 = 0.00027
- One bit at a time
- Worse when loss happens!



a. A stop-and-wait protocol in operation

Kurose/Ross


Data
Apps (HTTP)
Apps (HTTP)
Segments
Transport (TCP/UDP)
Transport (TCP/UDP)
Packets
Network (IP)
Network (IP)
Frames
Link (Ethernet)
Ethernet Interface
Ethernet Interface
Link (Ethernet)
Bits (1010001)

Reading Assignment

- Chapter 2.4 –Error detection and CRC-
  - https://book.systemsapproach.org/direct/error.html#internet-checksum-algorithm
  - https://book.systemsapproach.org/direct/error.html#cyclic-redundancy-check
  - **About 45 minutes read**

- https://book.systemsapproach.org/direct/reliable.html#reliable-transmission
  - Until Sliding window
  - **10 minutes read**