# CS3600 — Systems and Networks

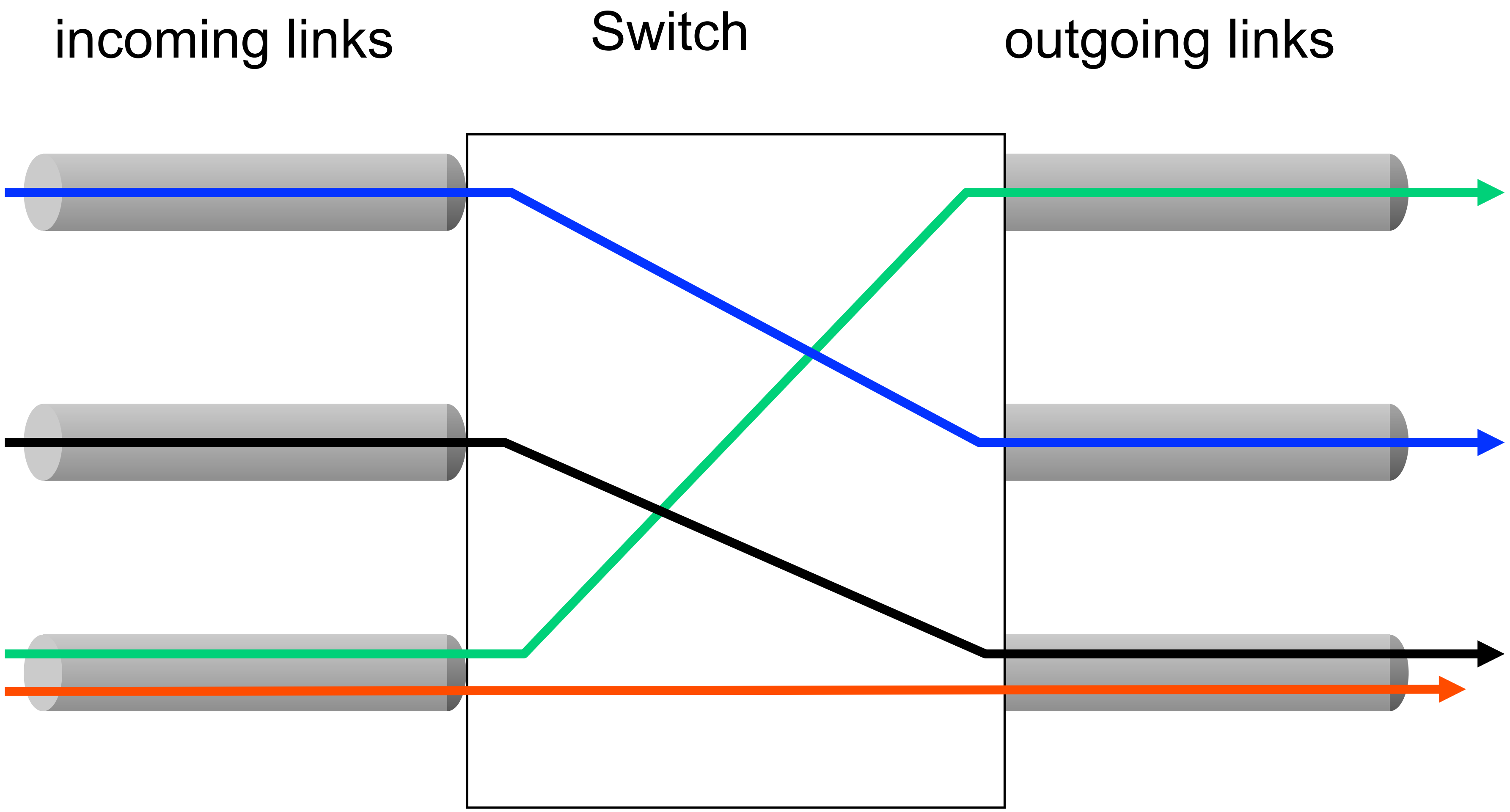
## NORTHEASTERN UNIVERSITY

Lecture 17: Internet architecture

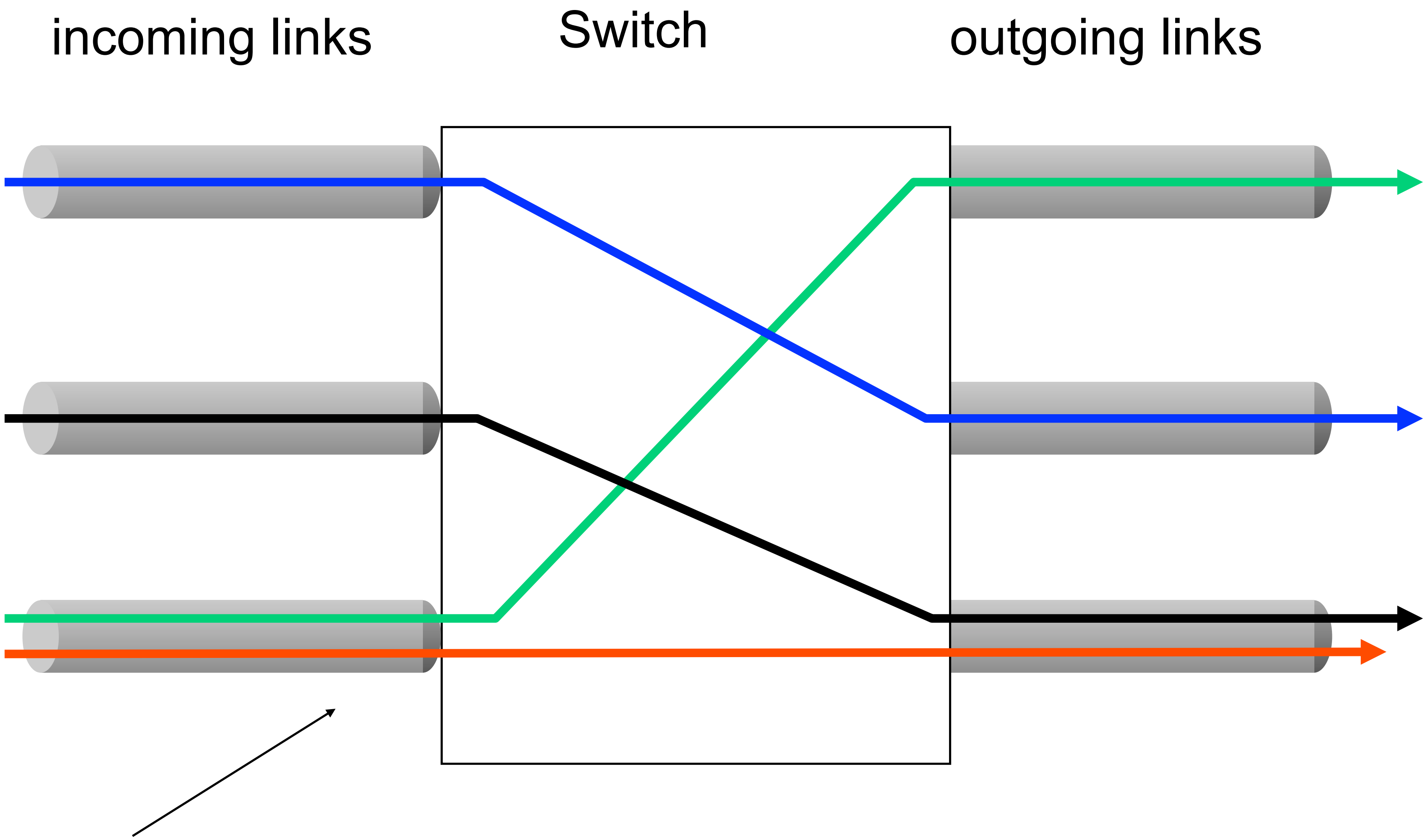
Prof. Alan Mislove (amislove@ccs.neu.edu)

Slides used with permissions from Edward W. Knightly,
T. S. Eugene Ng, Ion Stoica, Hui Zhang

# A Generic Switch



# A Generic Switch



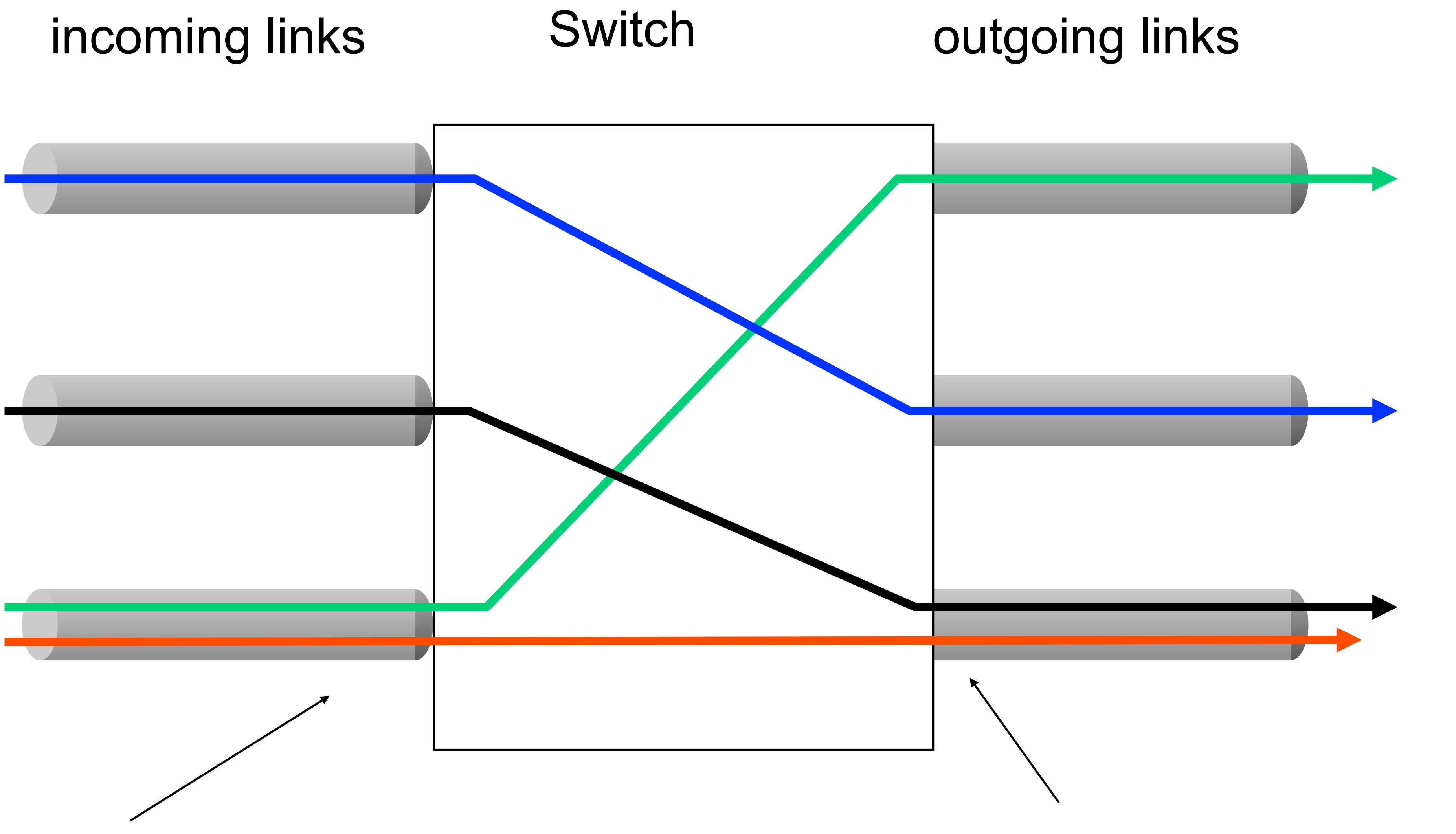
incoming links

Switch

outgoing links

How to Demultiplex?

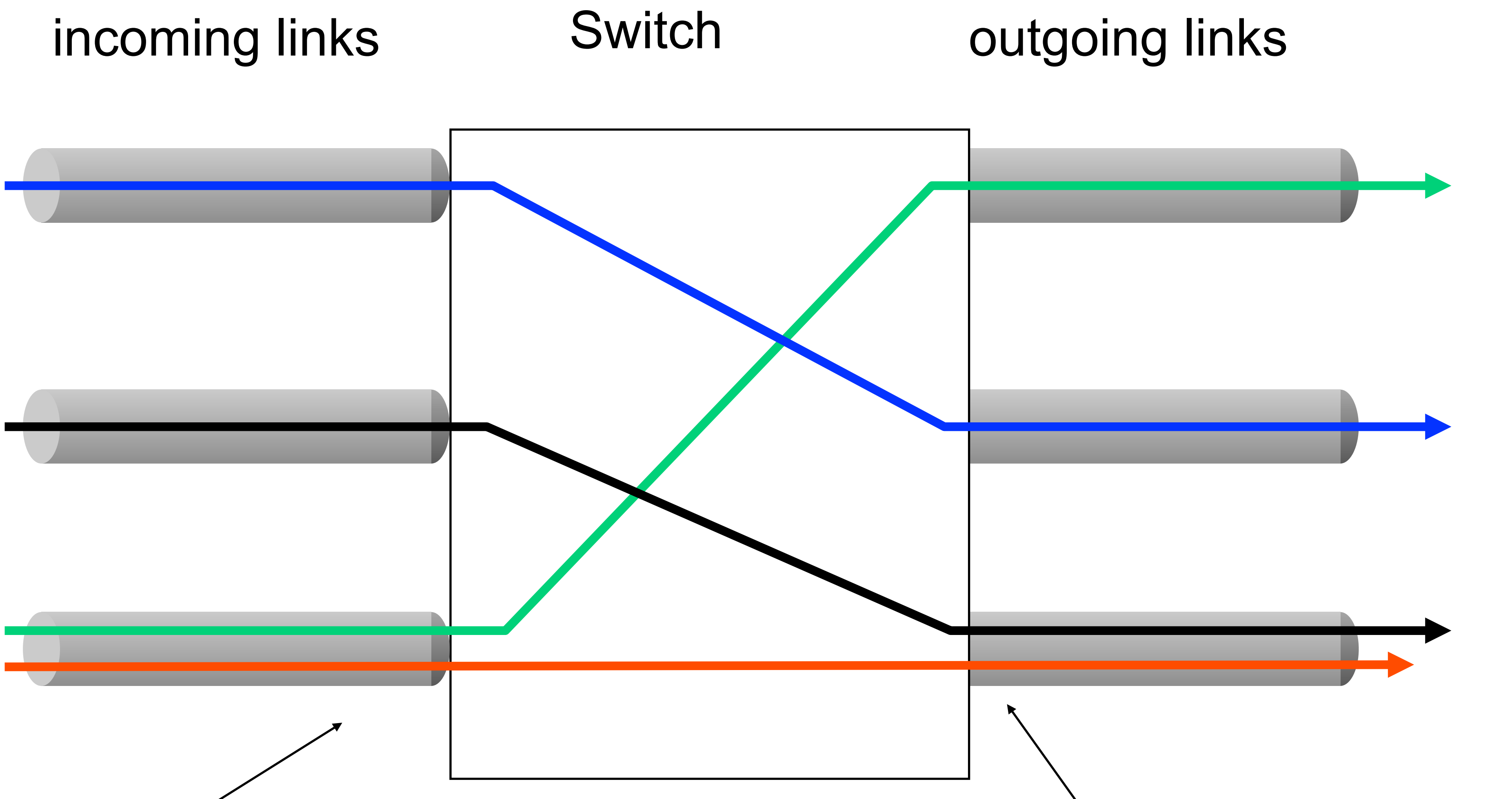
# A Generic Switch



How to Demultiplex?

How to Multiplex?

# A Generic Switch



incoming links

Switch

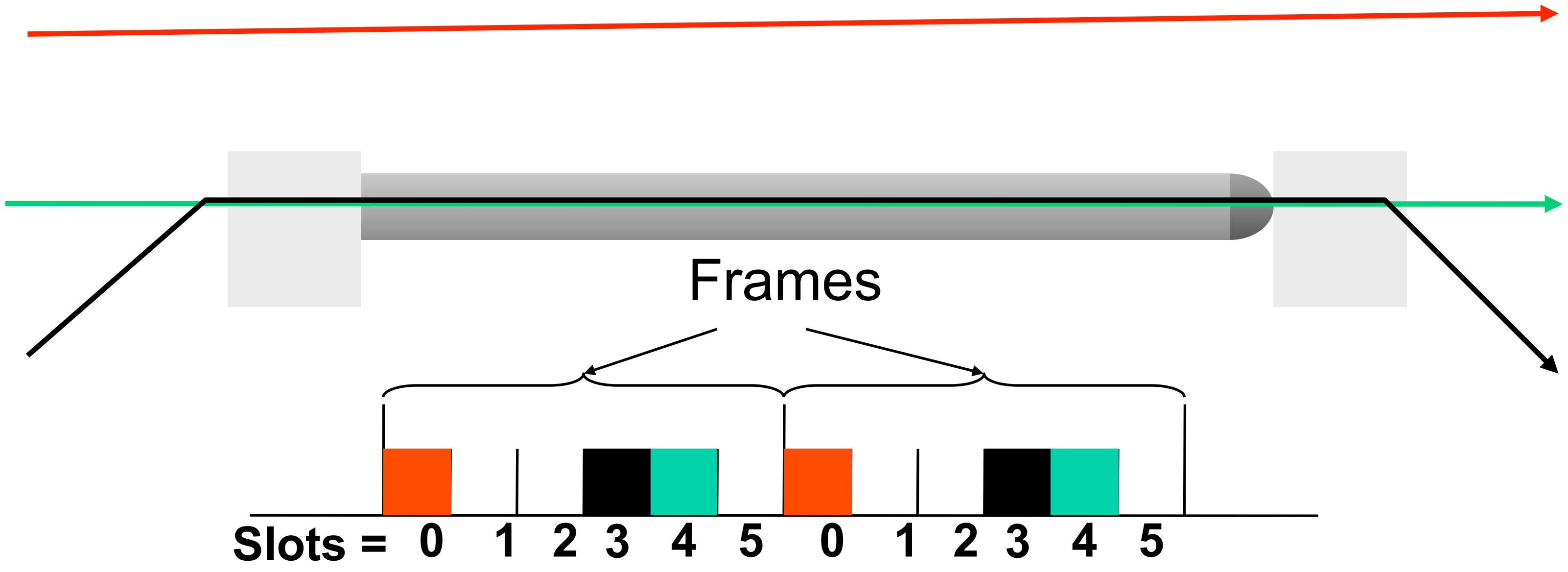
outgoing links

How to Demultiplex?

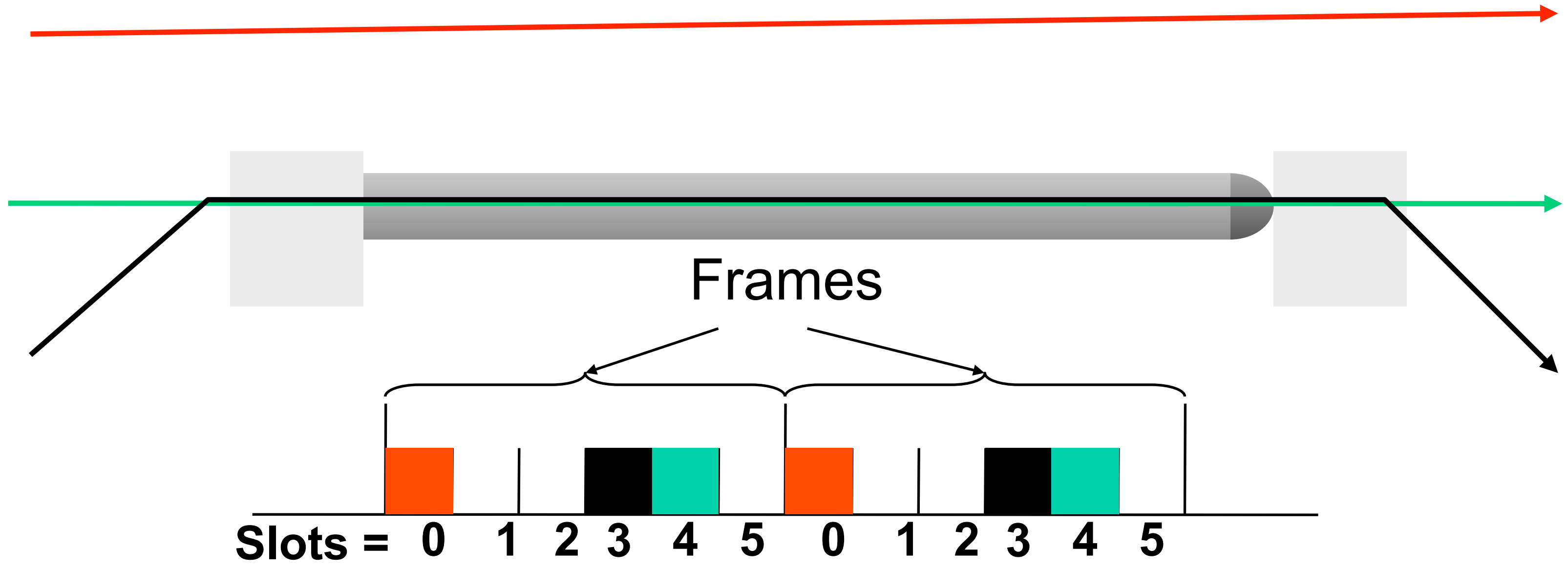
How to Switch?

How to Multiplex?

# Circuit Switching: Multiplexing/ Demultiplexing

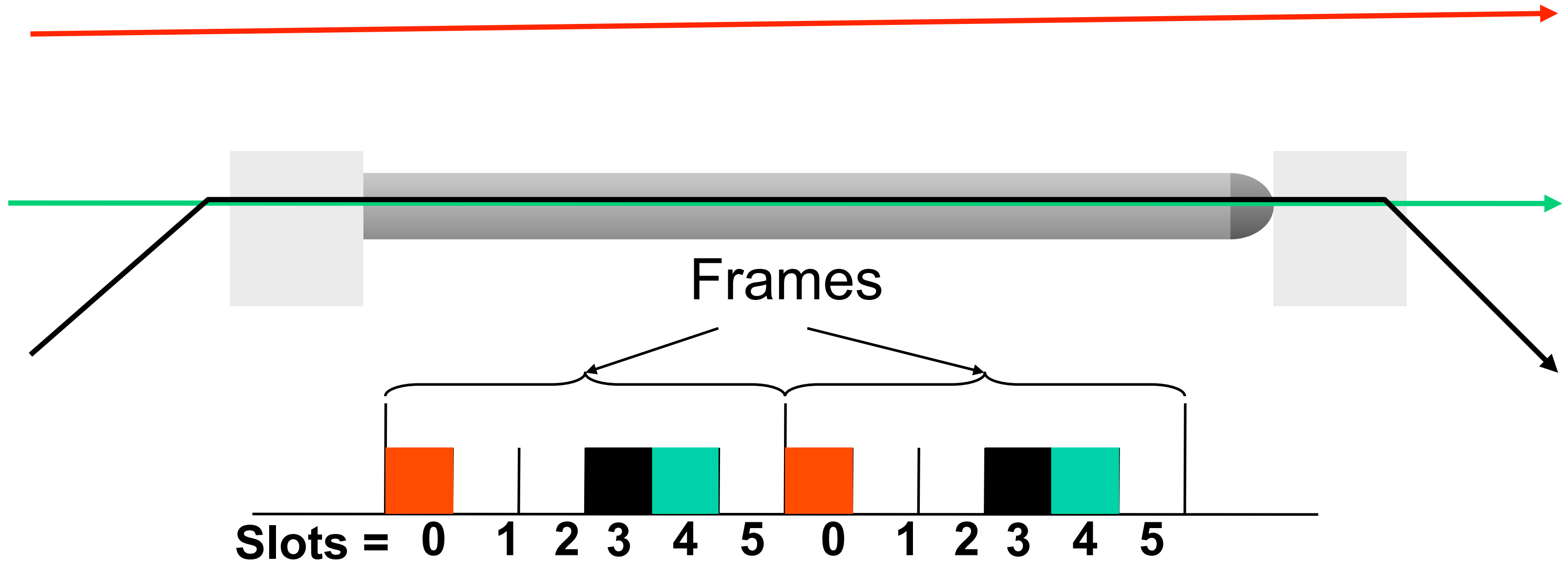


# Circuit Switching: Multiplexing/ Demultiplexing



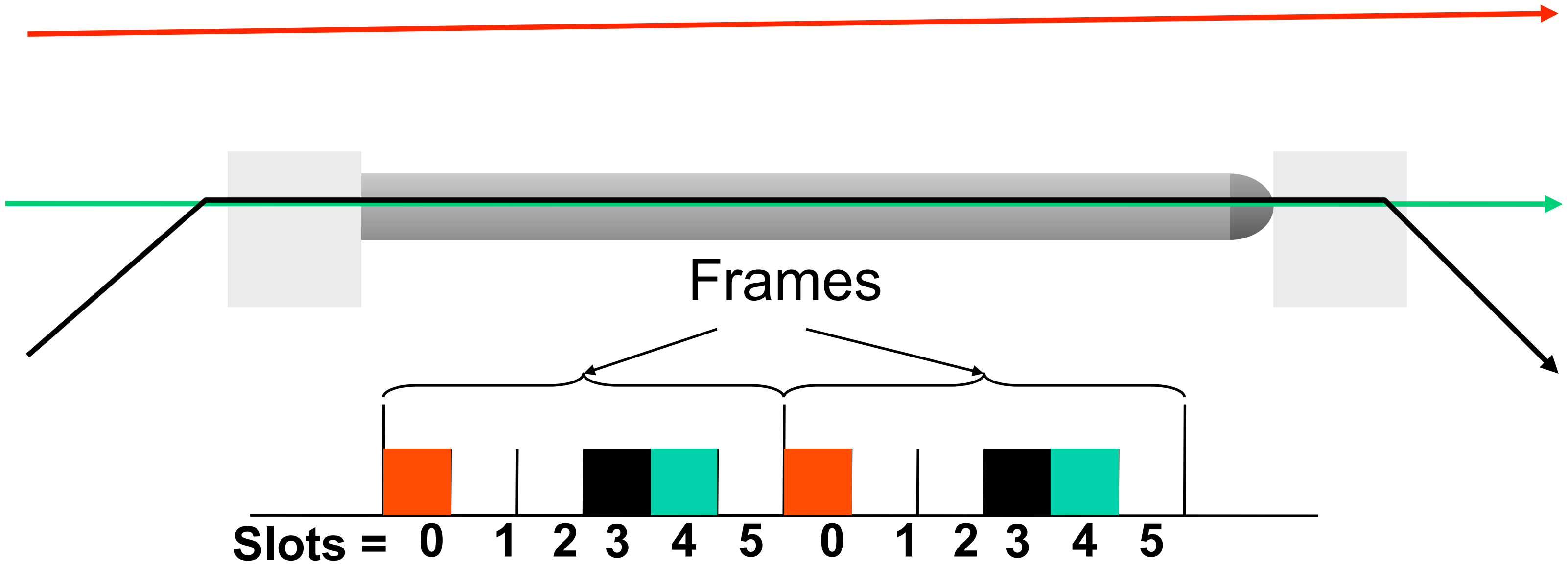
- Time divided in frames and frames divided in slots

# Circuit Switching: Multiplexing/ Demultiplexing



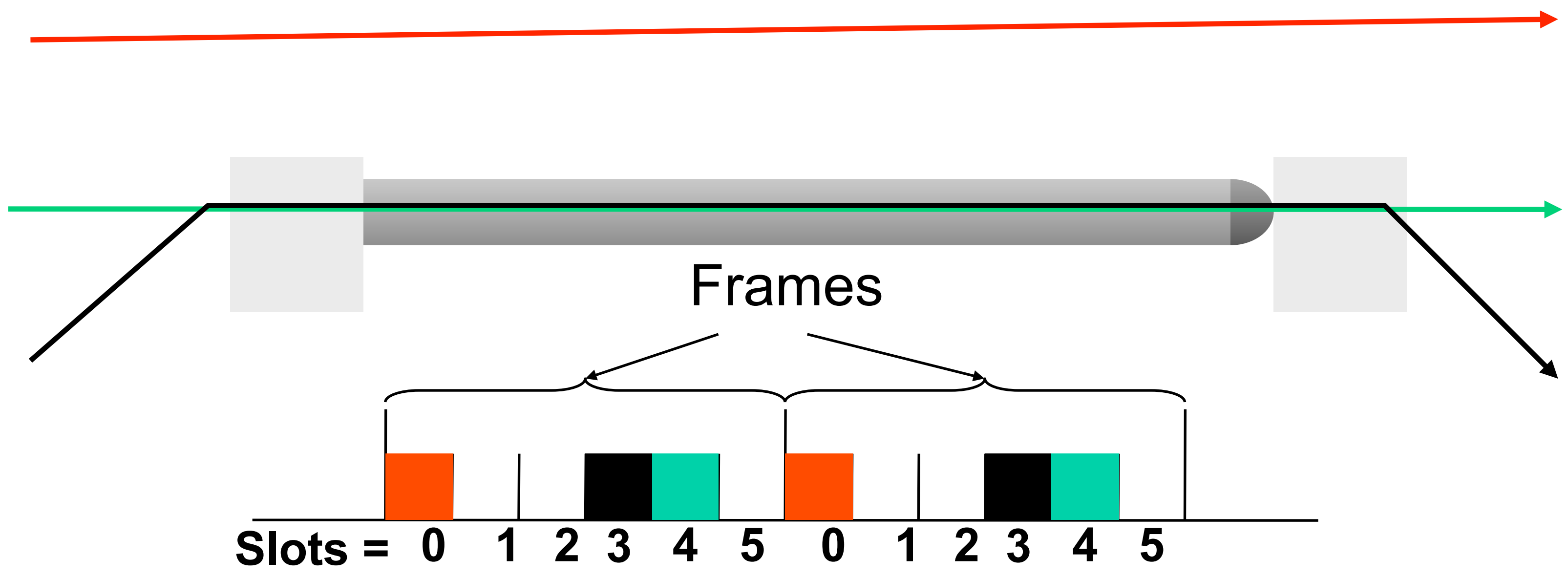
- Time divided in frames and frames divided in slots
- Relative slot position inside a frame determines which conversation the data belongs to
  - E.g., slot 0 belongs to red conversation
- Needs synchronization between sender and receiver

# Circuit Switching: Multiplexing/ Demultiplexing



- Time divided in frames and frames divided in slots
- Relative slot position inside a frame determines which conversation the data belongs to
  - E.g., slot 0 belongs to red conversation
- Needs synchronization between sender and receiver
- In case of non-permanent conversations
  - Needs to dynamic bind a slot to a conservation
  - How to do this?

# Circuit Switching: Multiplexing/ Demultiplexing

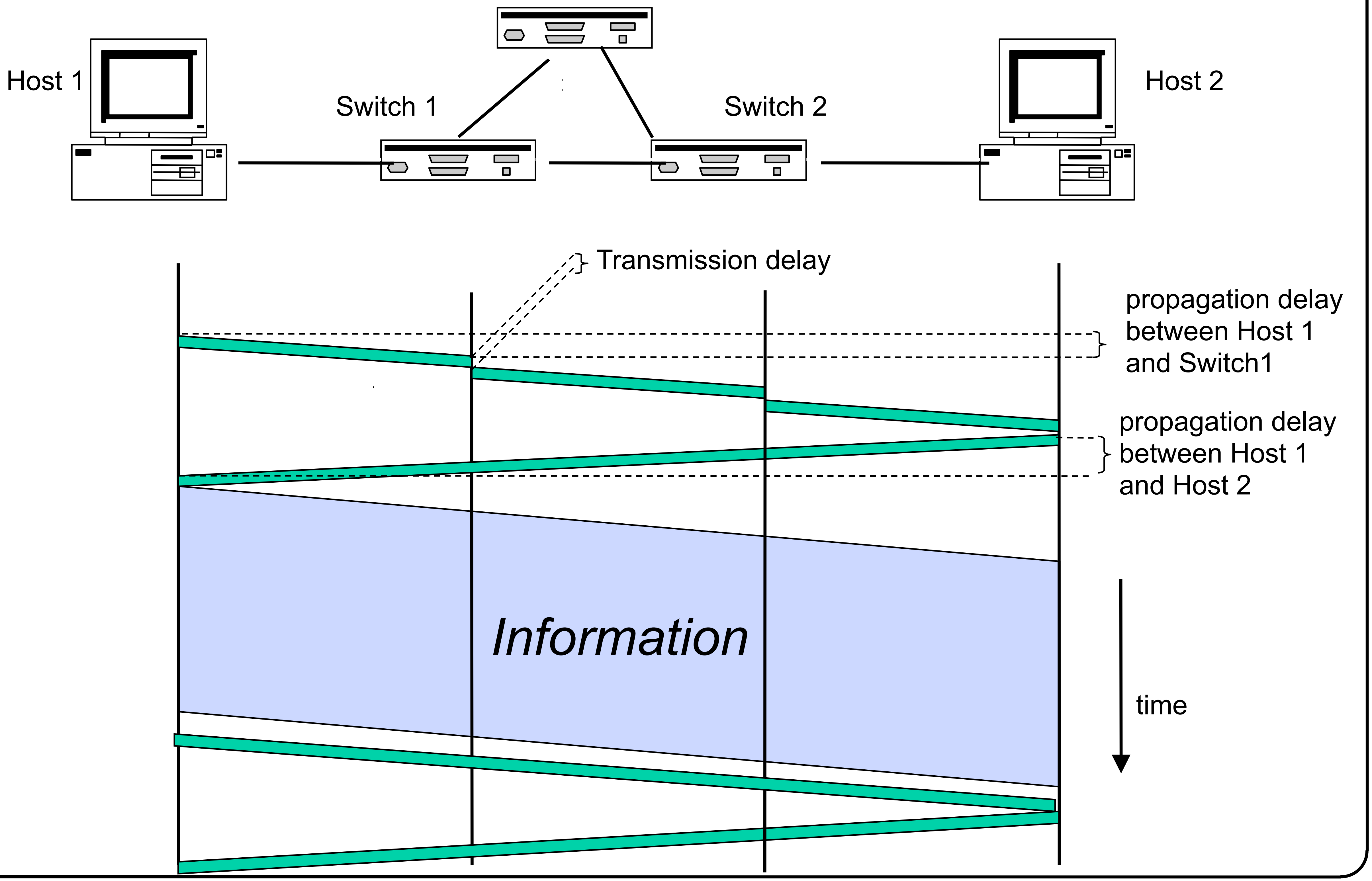


- Time divided in frames and frames divided in slots
- Relative slot position inside a frame determines which conversation the data belongs to
  - E.g., slot 0 belongs to red conversation
- Needs synchronization between sender and receiver
- In case of non-permanent conversations
  - Needs to dynamic bind a slot to a conservation
  - How to do this?
- If a conversation does not use its circuit the capacity is lost!

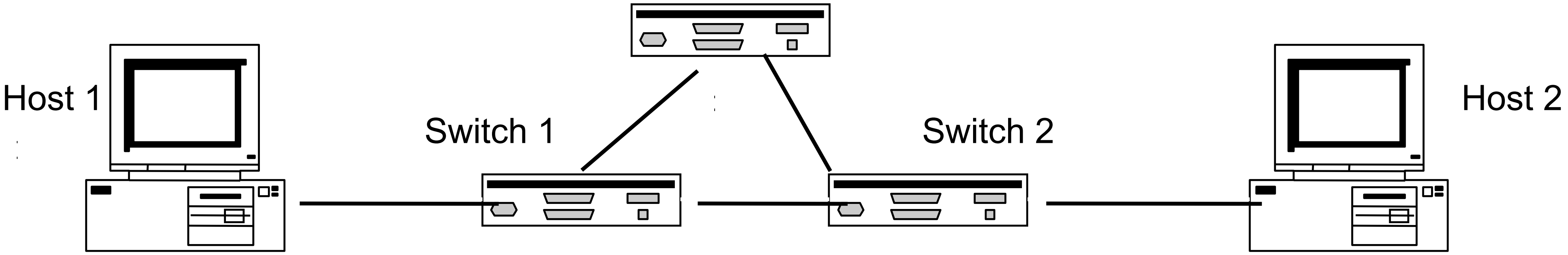
# Circuit Switching

- Three phases
  1. circuit establishment
  2. data transfer
  3. circuit termination
- If circuit not available: busy
- Examples
  - Telephone networks
  - ISDN (Integrated Services Digital Networks)

# Timing in Circuit Switching


Host 1
Switch 1
Switch 2
Host 2
Transmission delay
propagation delay between Host 1 and Switch1
propagation delay between Host 1 and Host 2
Information
time

# Timing in Circuit Switching

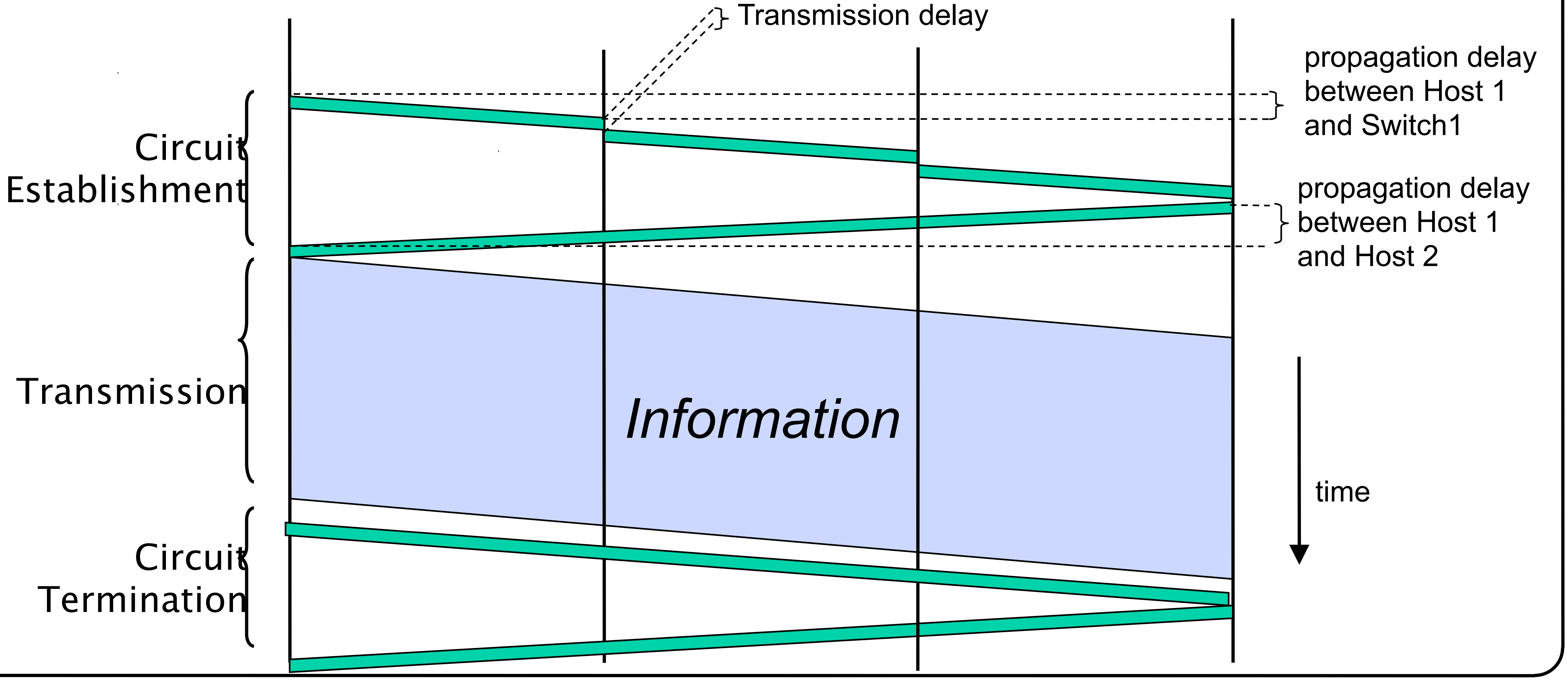


Host 1

Switch 1

Switch 2

Host 2



Transmission delay

propagation delay
between Host 1
and Switch1

propagation delay
between Host 1
and Host 2

Circuit
Establishment

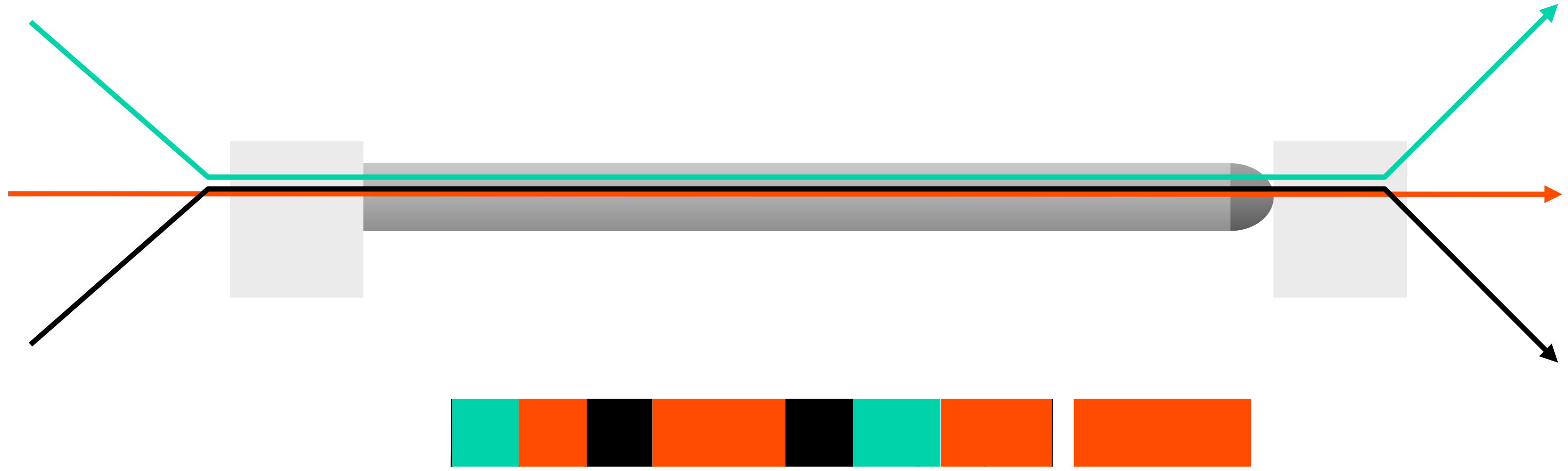
Transmission

*Information*

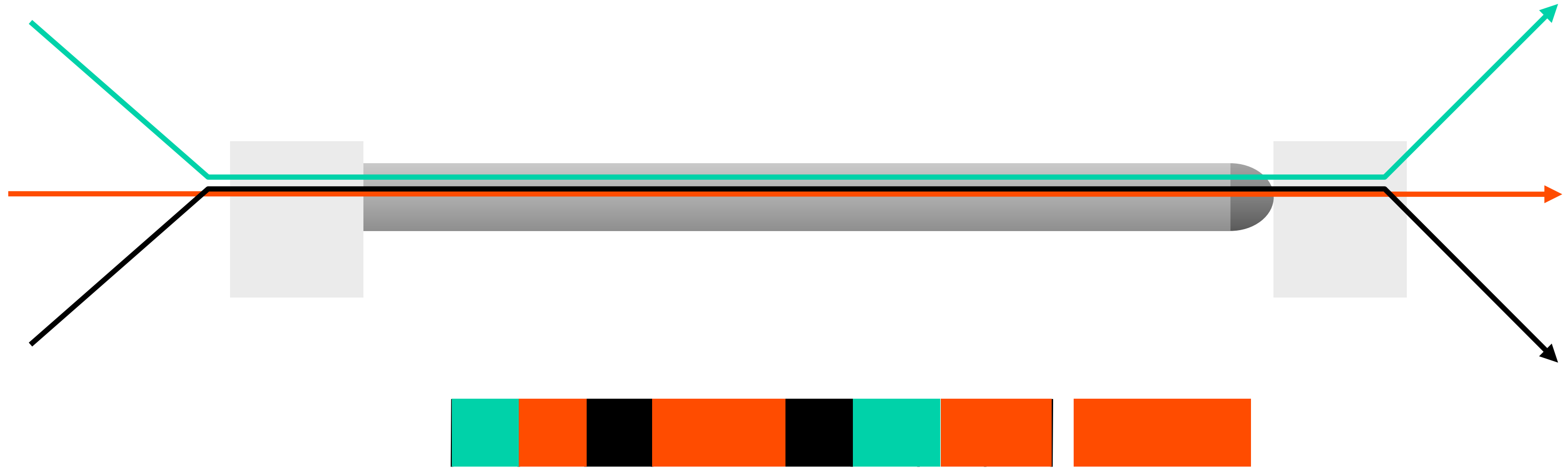
time

Circuit
Termination

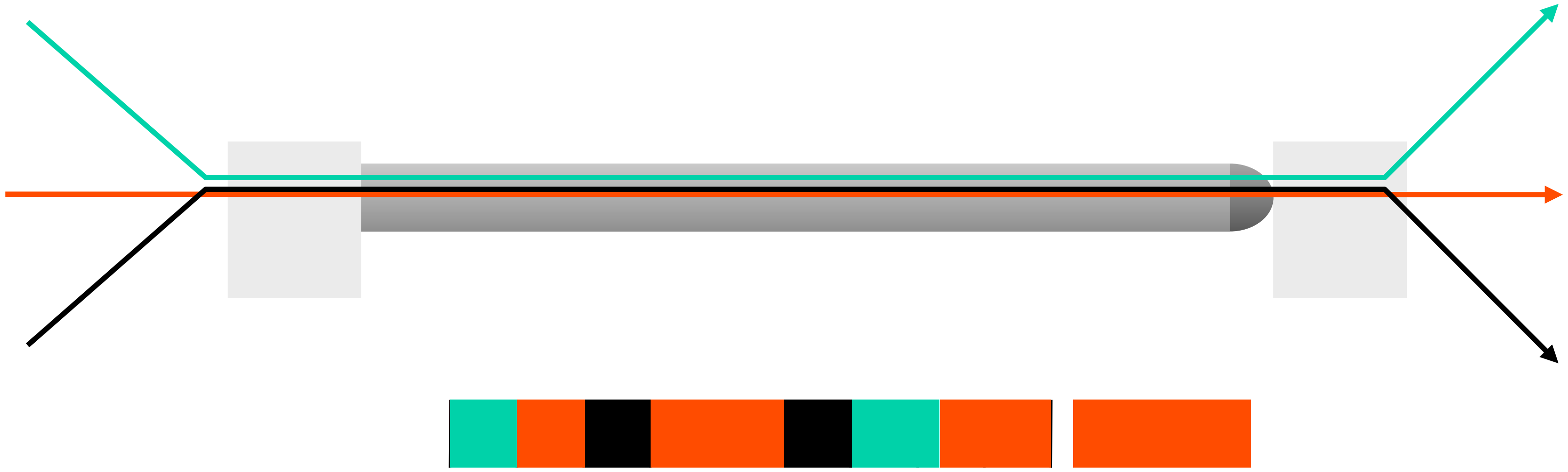
# Packet Switching: Multiplexing/ Demultiplexing



# Packet Switching: Multiplexing/ Demultiplexing

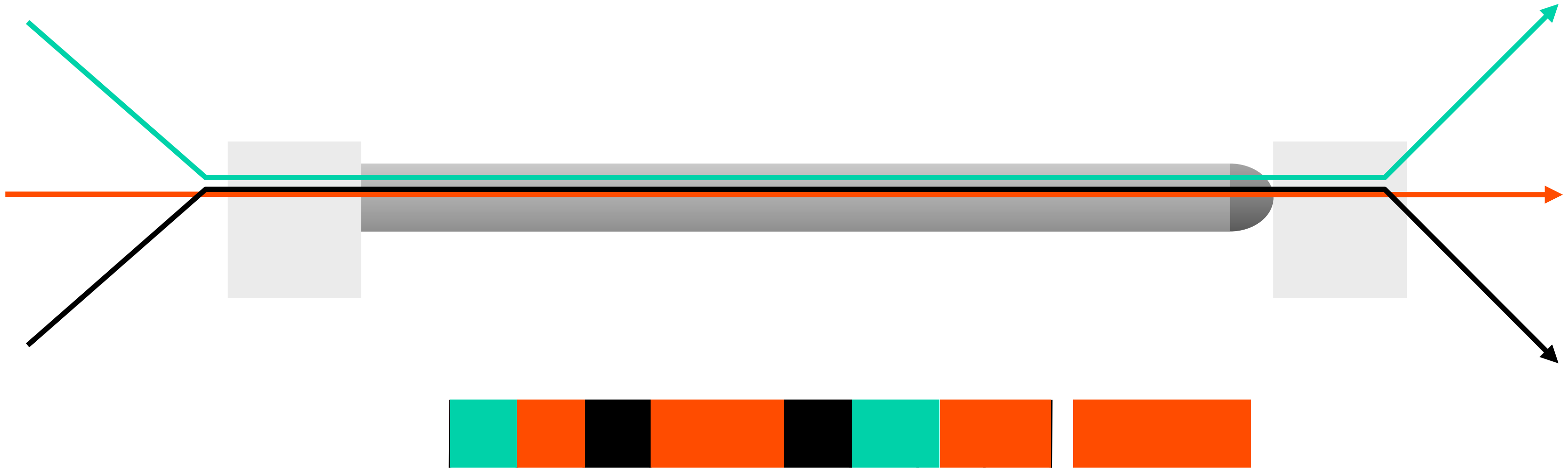


# Packet Switching: Multiplexing/ Demultiplexing



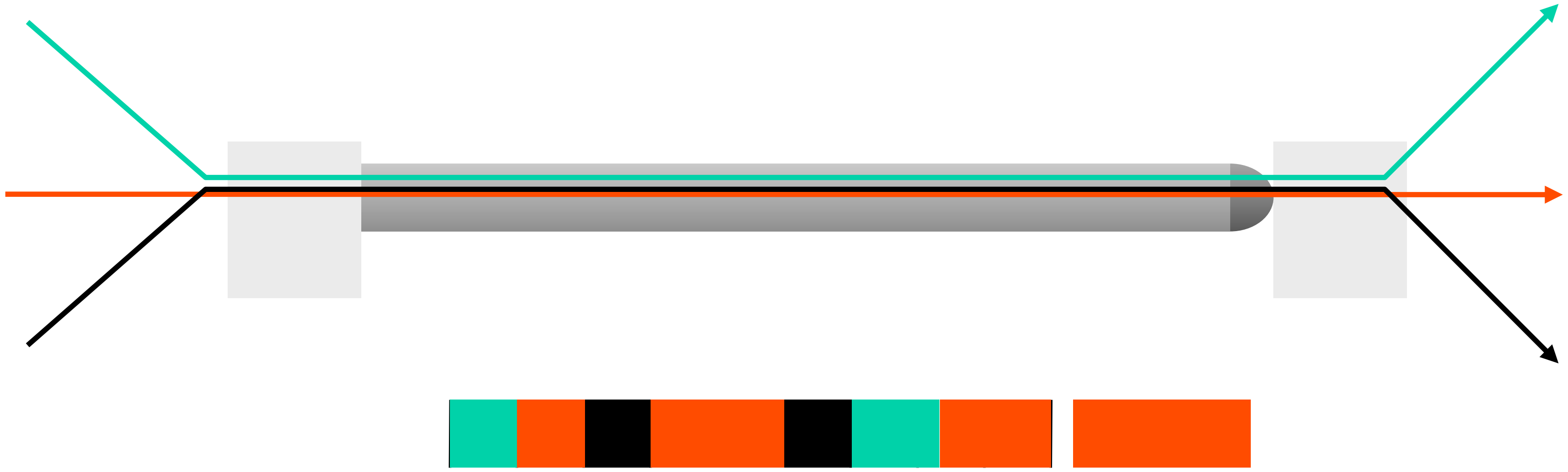
- Data from any conversation can be transmitted at any given time

# Packet Switching: Multiplexing/ Demultiplexing



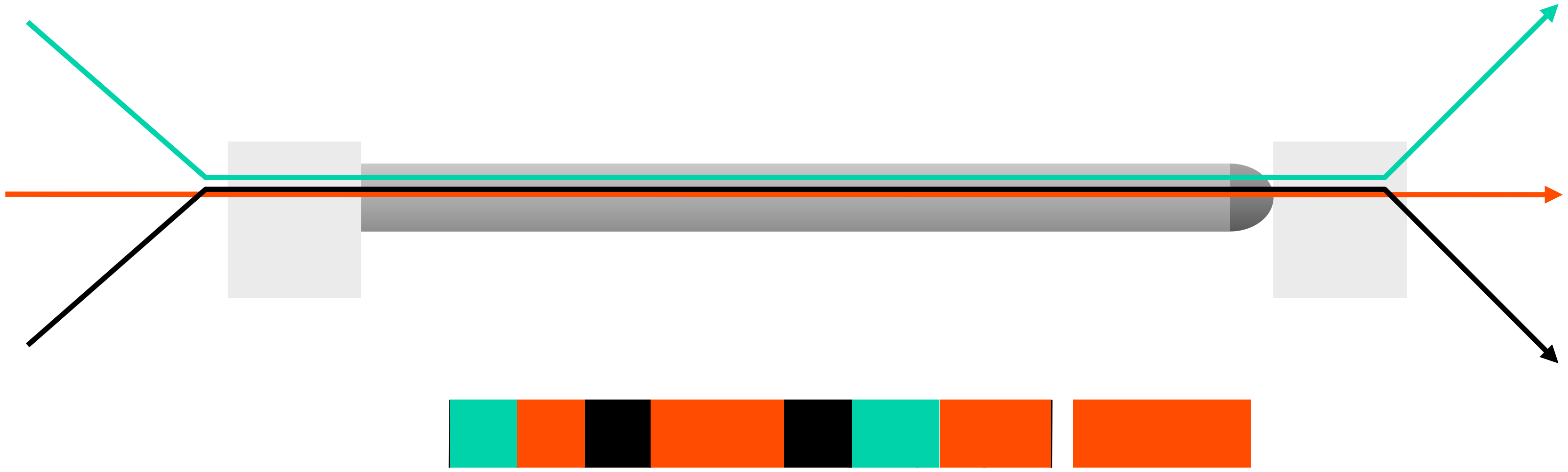
- Data from any conversation can be transmitted at any given time
  - A single conversation can use the entire link capacity if it is alone

# Packet Switching: Multiplexing/ Demultiplexing



- Data from any conversation can be transmitted at any given time
  - A single conversation can use the entire link capacity if it is alone
- How to demultiplex?

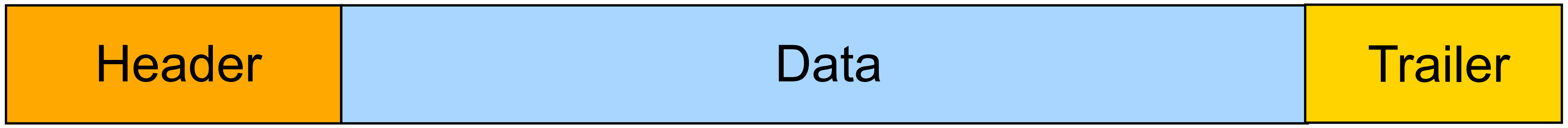
# Packet Switching: Multiplexing/ Demultiplexing



- Data from any conversation can be transmitted at any given time
    - A single conversation can use the entire link capacity if it is alone
- How to demultiplex?
    - Use meta-data (header) to describe data

# Packet Switching

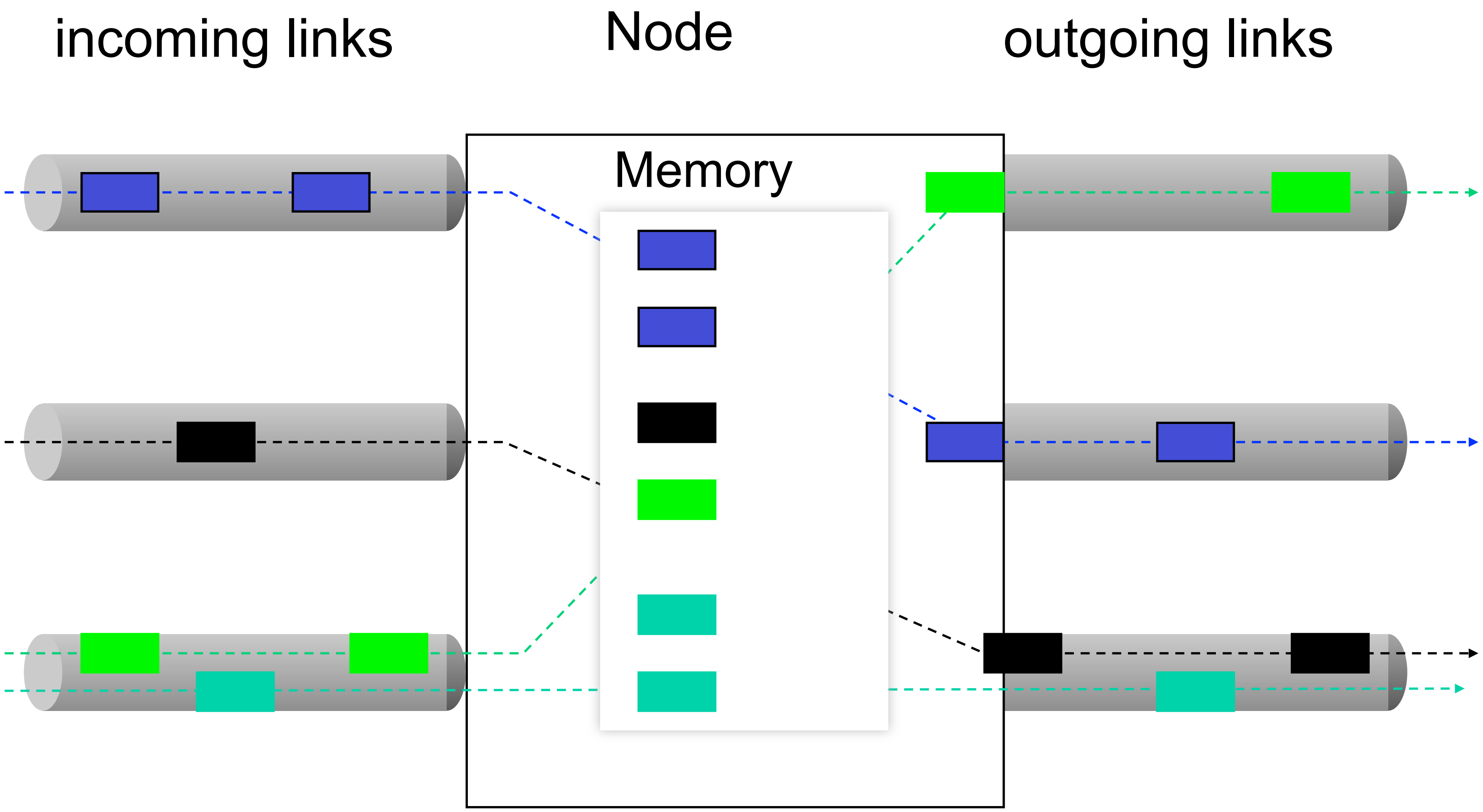
- Data are sent as formatted bit-sequences, so-called packets.
- Packets have the following structure:



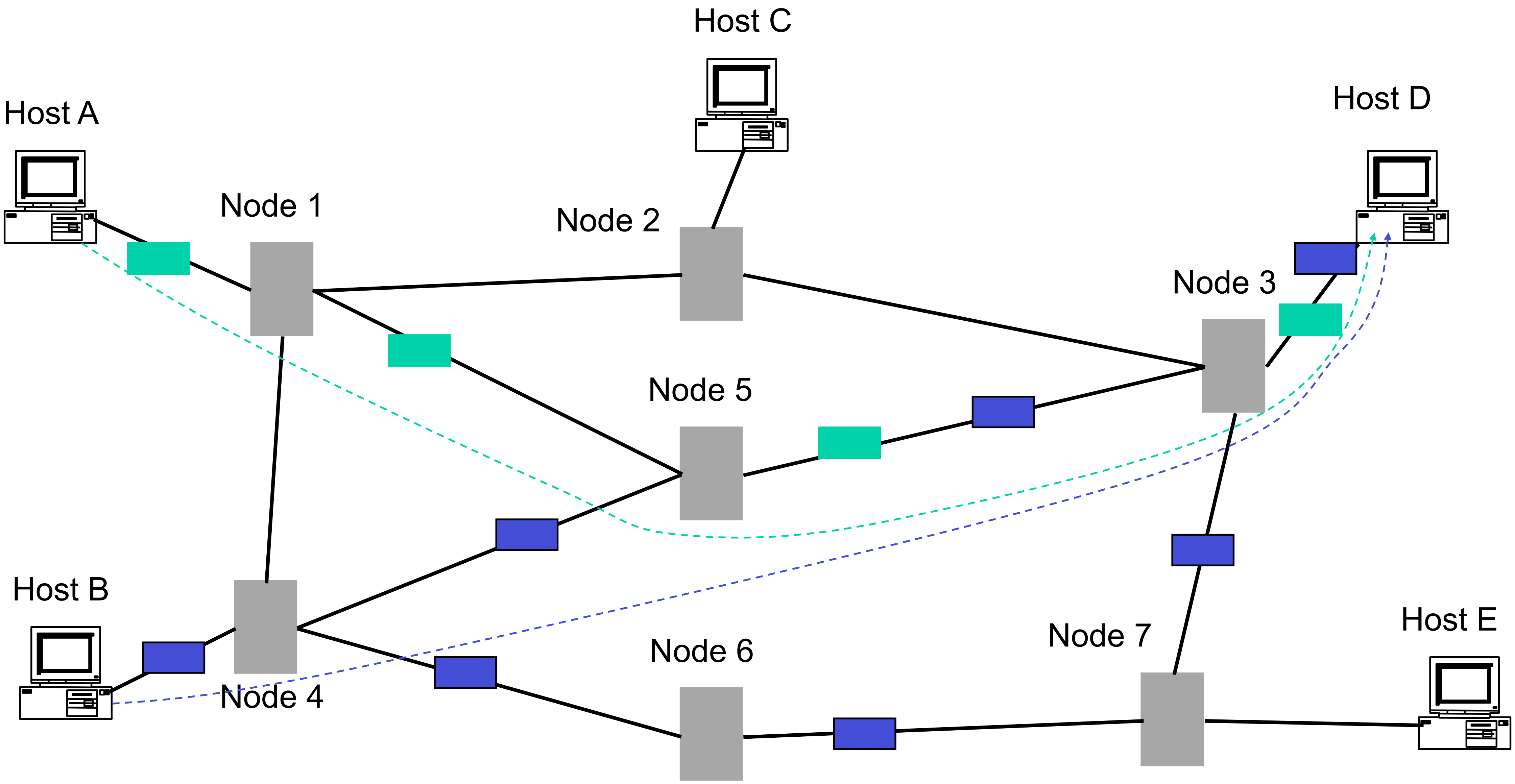
Header Data Trailer

  - Header and Trailer carry control information (e.g., destination address, check sum)
- At each node the entire packet is received, stored briefly, and then forwarded to the next node based on the header information (**Store-and-Forward Networks**)
- Allows statistical multiplexing

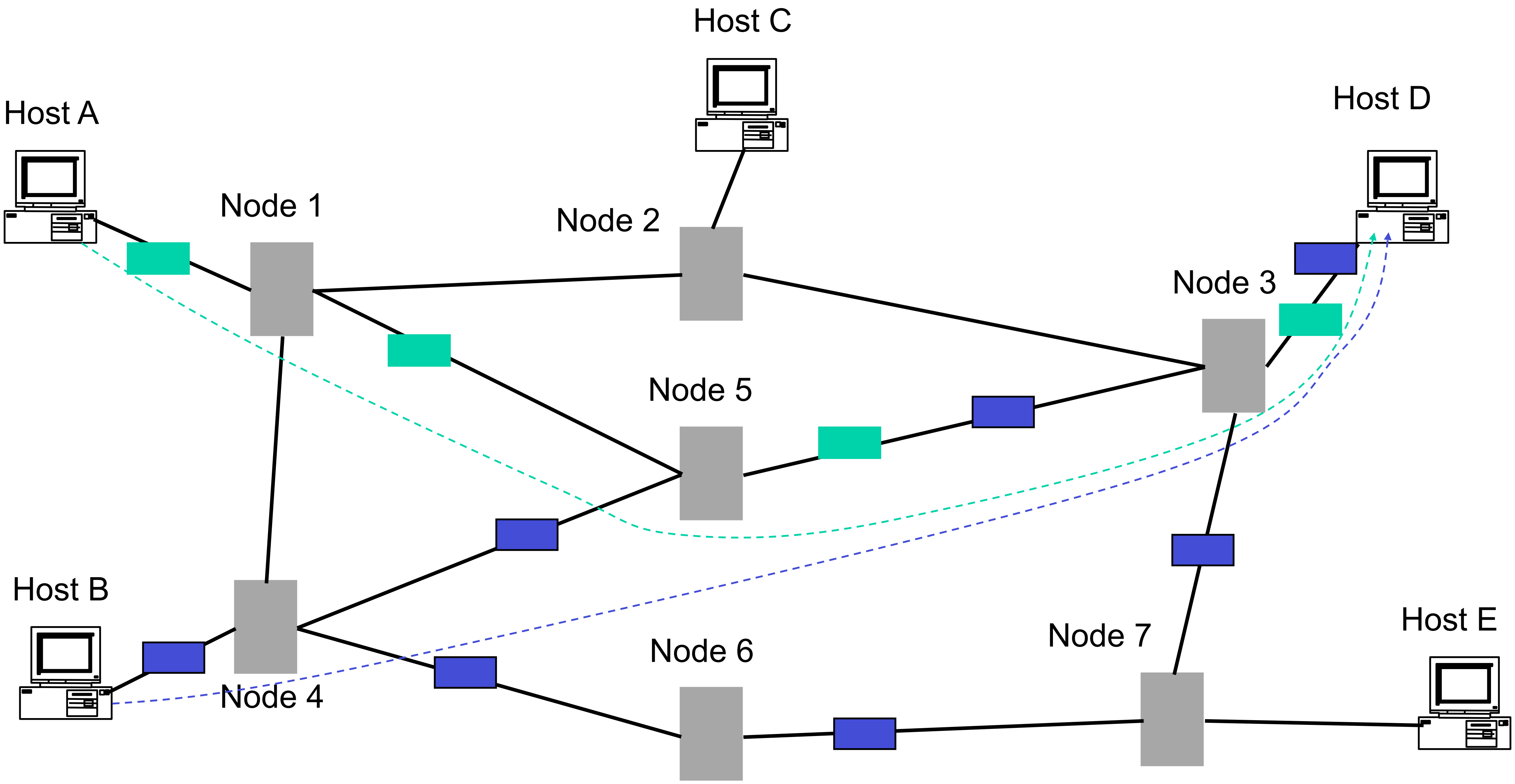
# Packet Switch



# Datagram Packet Switching

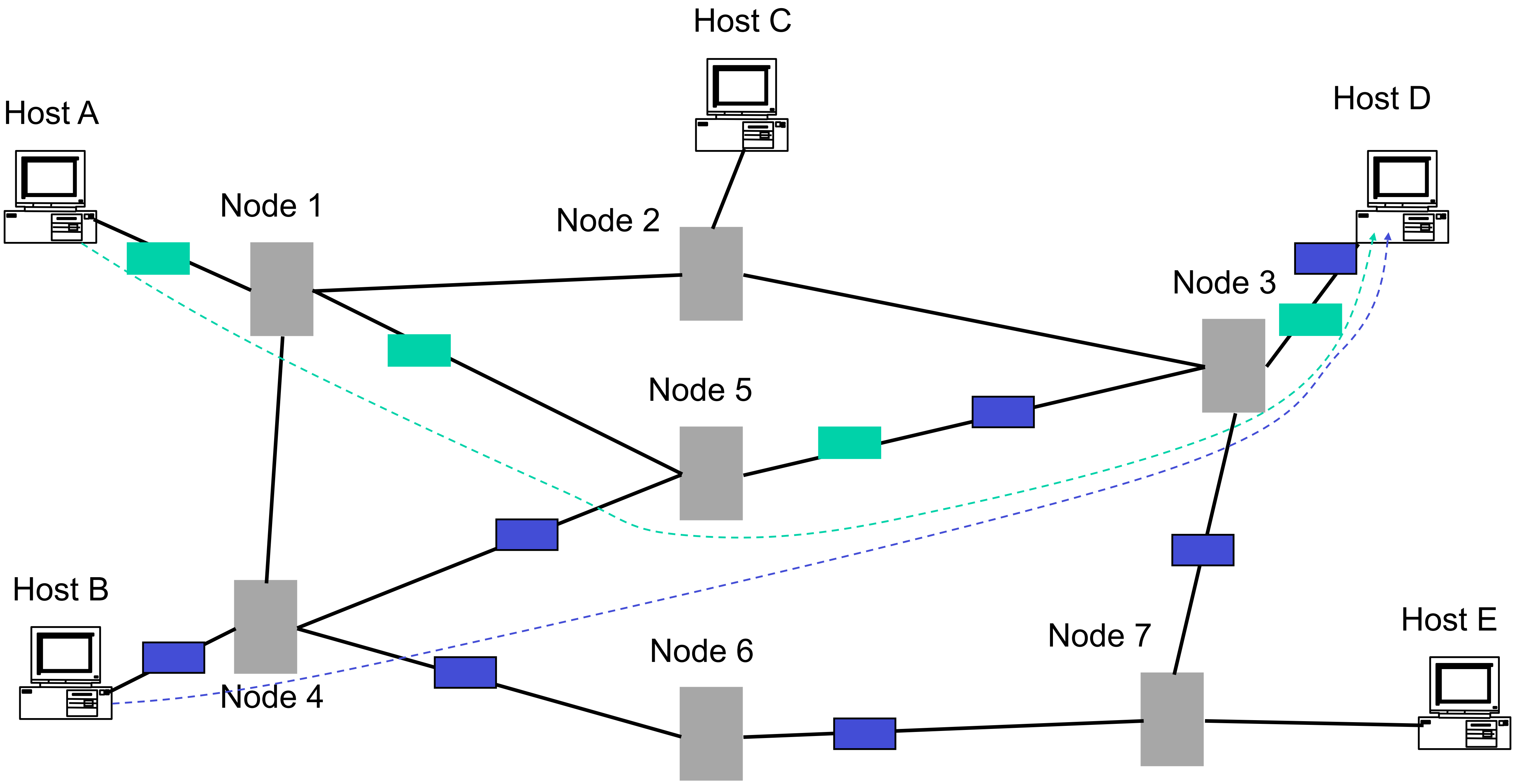


# Datagram Packet Switching



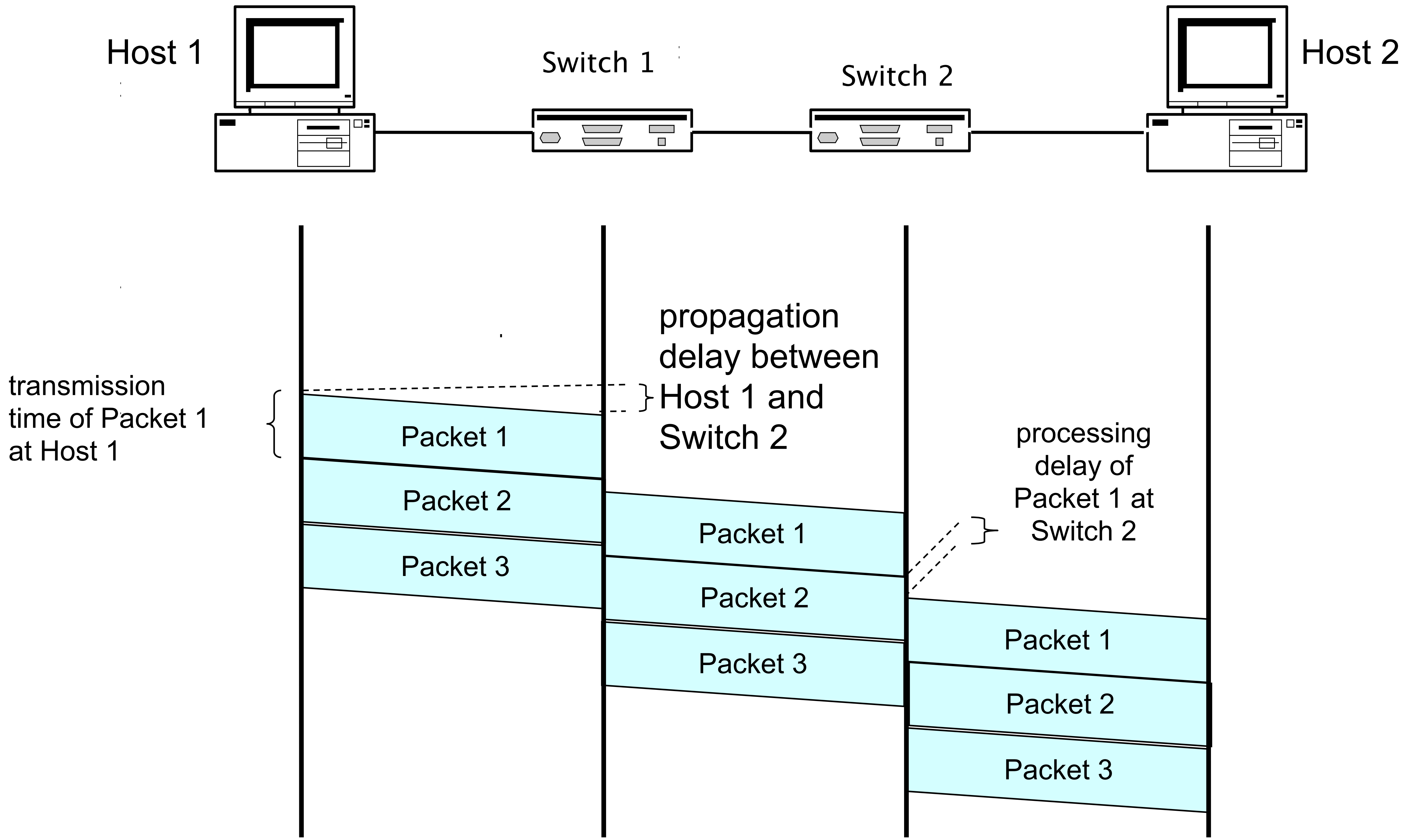
- Each packet is independently switched

# Datagram Packet Switching



- Each packet is independently switched
  - Each packet header contains destination address

# Timing of Datagram Packet Switching


Host 1
Switch 1
Switch 2
Host 2
transmission time of Packet 1 at Host 1
propagation delay between Host 1 and Switch 2
processing delay of Packet 1 at Switch 2
Packet 1
Packet 2
Packet 3
Packet 1
Packet 2
Packet 3
Packet 1
Packet 2
Packet 3

# Packet-Switching vs. Circuit-Switching

# Packet-Switching vs. Circuit-Switching

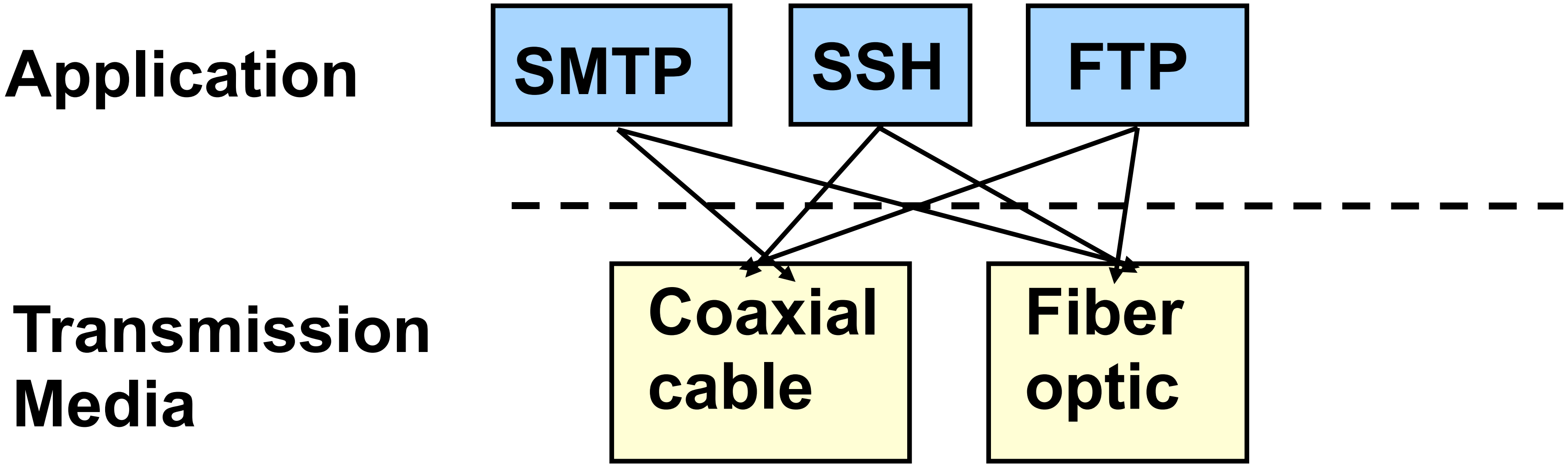
- Most important advantage of packet-switching over circuit switching: ability to exploit statistical multiplexing
  - More efficient bandwidth usage
- However, packet-switching needs to buffer and deal with congestion
  - More complex switches
  - Harder to provide good network services (e.g., delay and bandwidth guarantees)

# Organizing Network Functionality

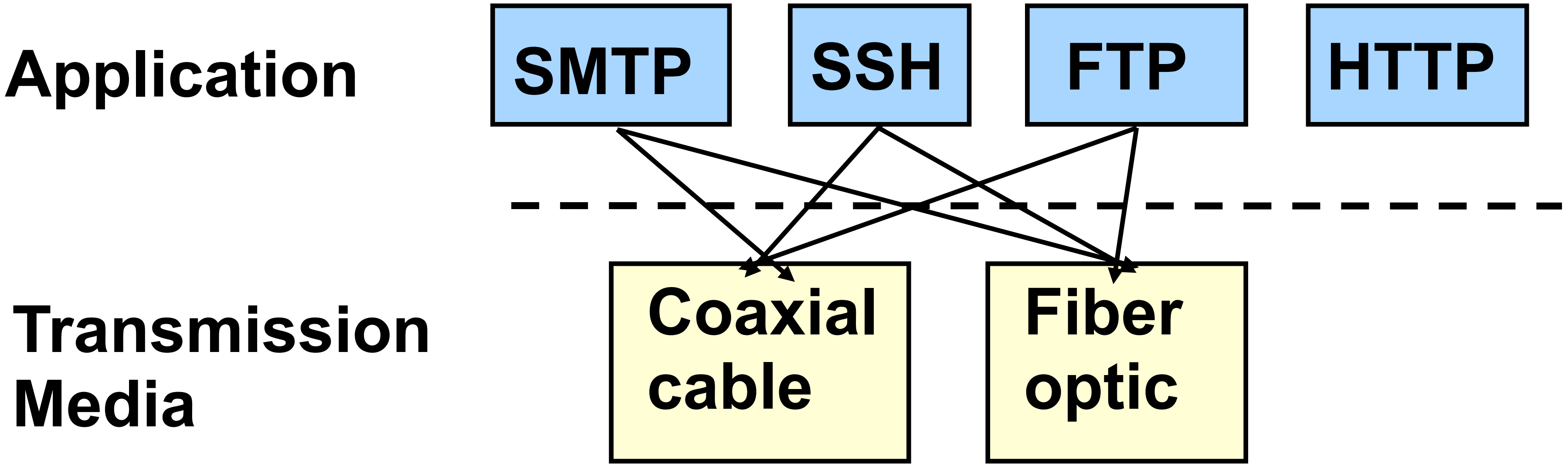
- Many kinds of networking functionality
  - e.g., encoding, framing, routing, addressing, reliability, etc.
- Many different network styles and technologies
  - circuit-switched vs packet-switched, etc.
  - wireless vs wired vs optical, etc.
- Many different applications
  - ftp, email, web, P2P, etc.

- Network architecture
  - How should different pieces be organized?
  - How should different pieces interact?

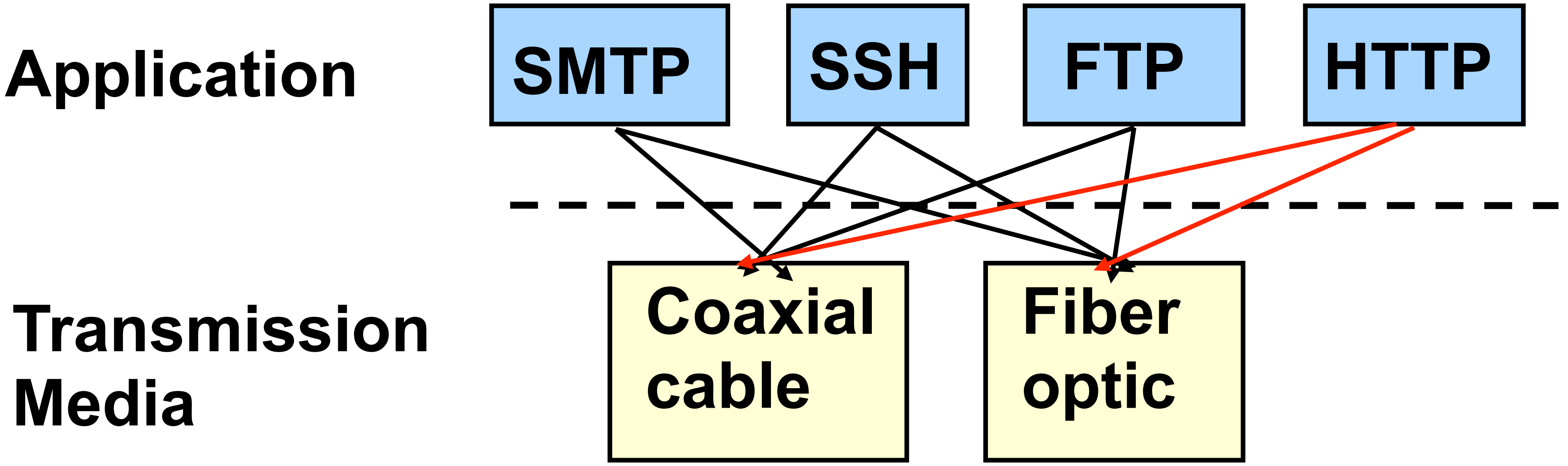
# Problem


Application
SMTP
SSH
FTP
Transmission
Media
Coaxial
cable
Fiber
optic

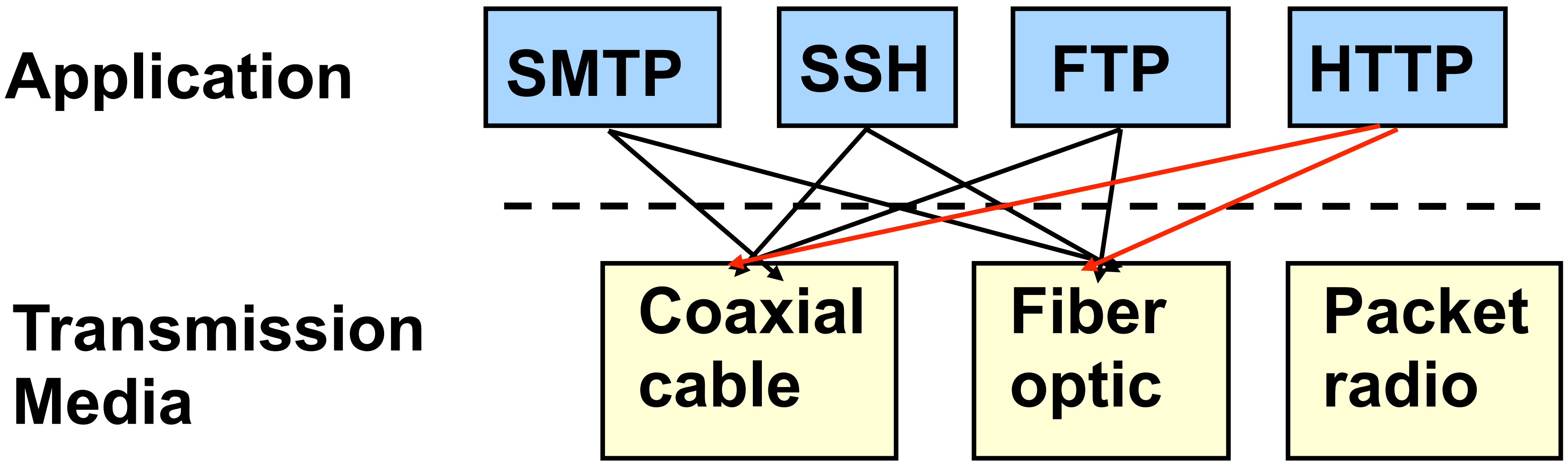
# Problem


Application
SMTP
SSH
FTP
HTTP
Transmission Media
Coaxial cable
Fiber optic

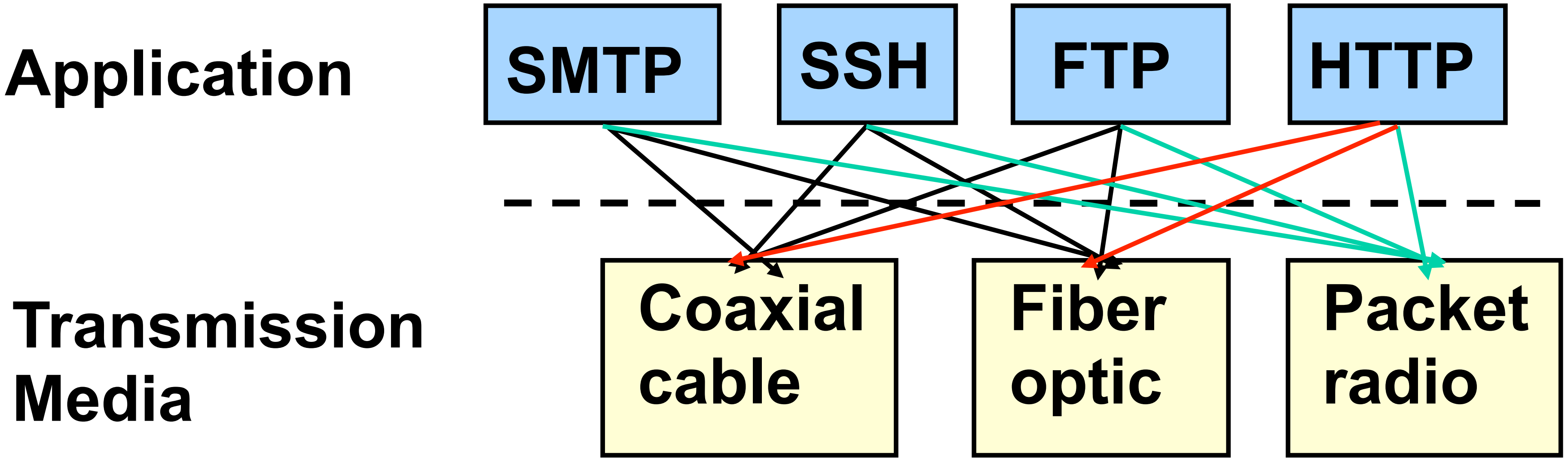
# Problem


Application
SMTP
SSH
FTP
HTTP
Transmission Media
Coaxial cable
Fiber optic

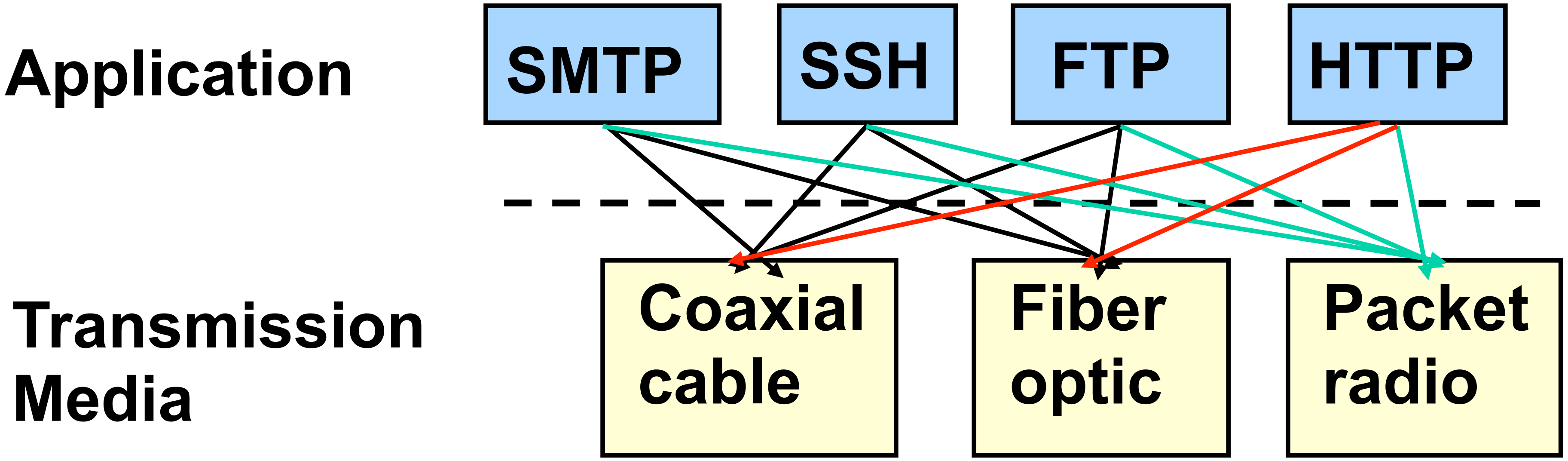
# Problem


Application
SMTP
SSH
FTP
HTTP
Transmission Media
Coaxial cable
Fiber optic
Packet radio

# Problem

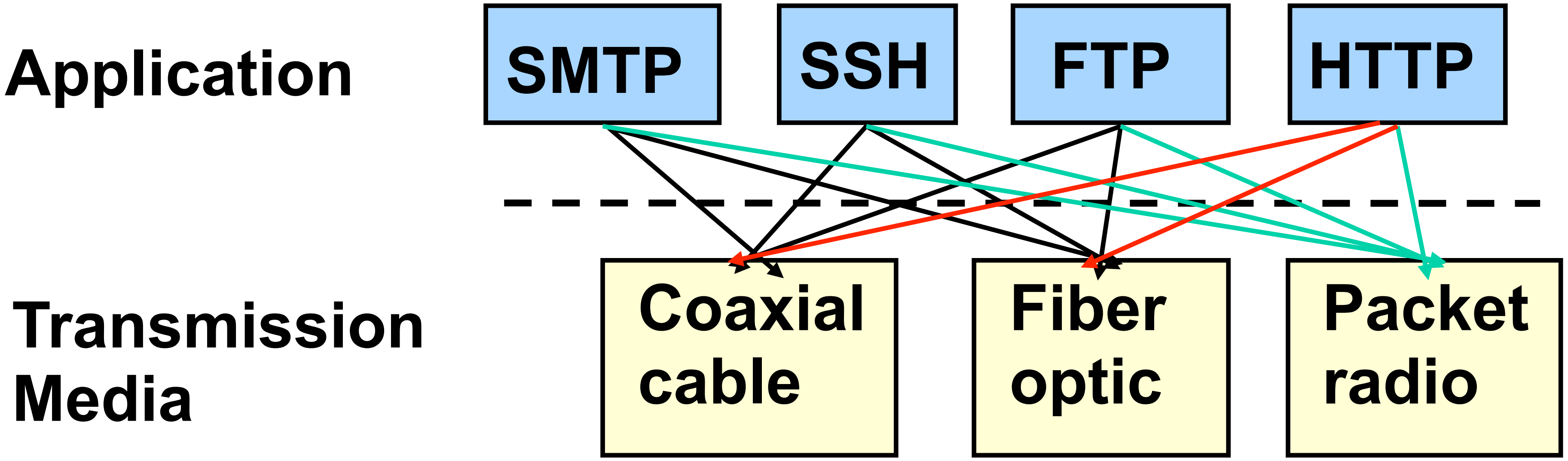

Application
SMTP
SSH
FTP
HTTP
Transmission Media
Coaxial cable
Fiber optic
Packet radio

# Problem



- new application has to interface to all existing media
  - adding new application requires O(m) work, m = number of media
- new media requires all existing applications be modified
  - adding new media requires O(a) work, a = number of applications

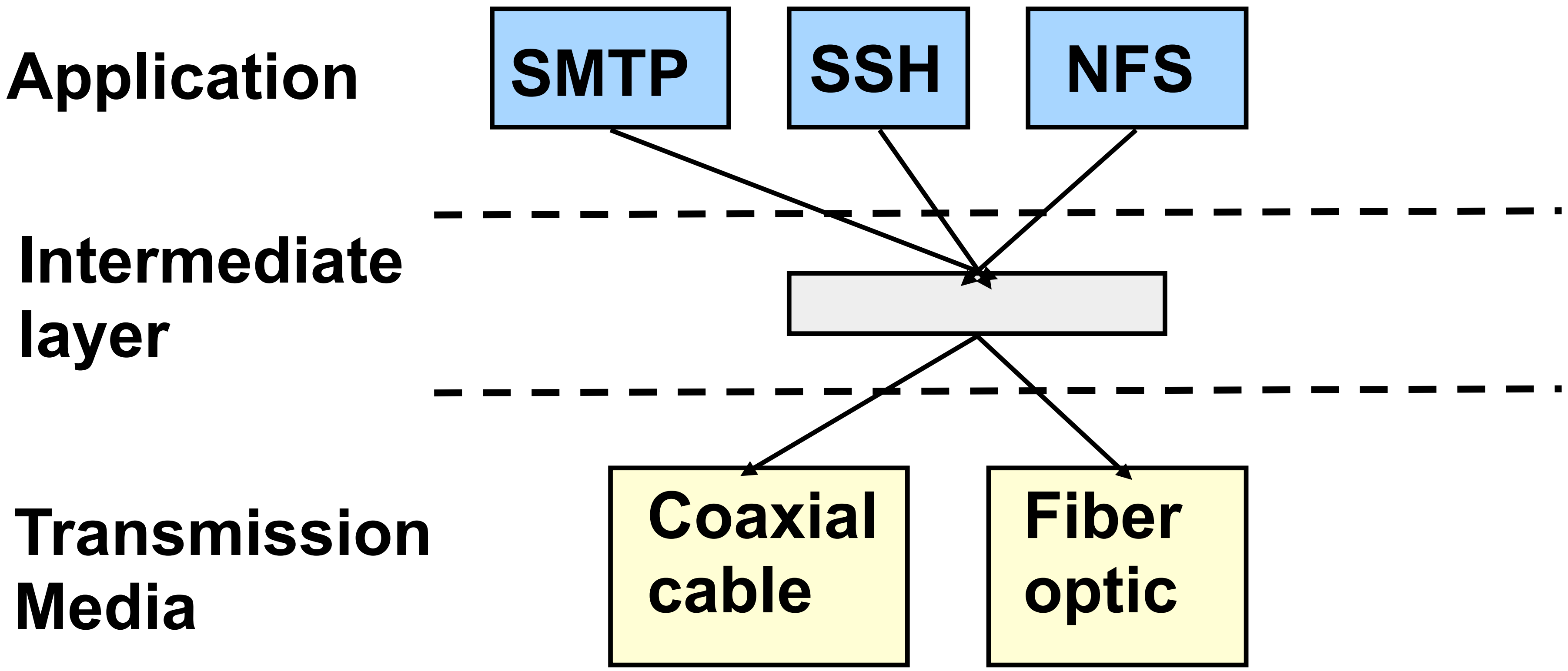
# Problem



- new application has to interface to all existing media
  - adding new application requires O(m) work, m = number of media
- new media requires all existing applications be modified
  - adding new media requires O(a) work, a = number of applications
- total work in system O(ma) → eventually too much work to add apps/media
- Application end points may not be on the same media!

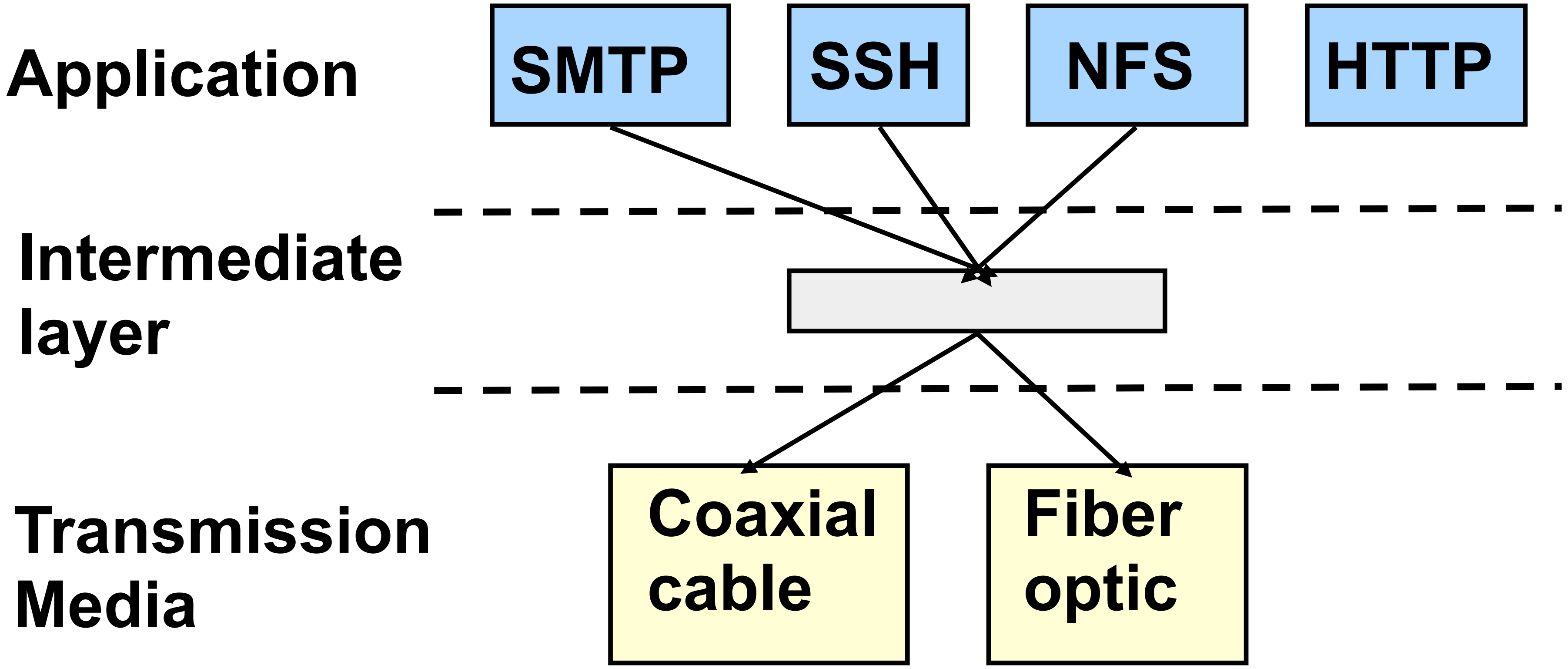
# Solution: Indirection

- Solution: introduce an intermediate layer that provides a single abstraction for various network technologies
  - O(1) work to add app/media
  - Indirection is an often used technique in computer science



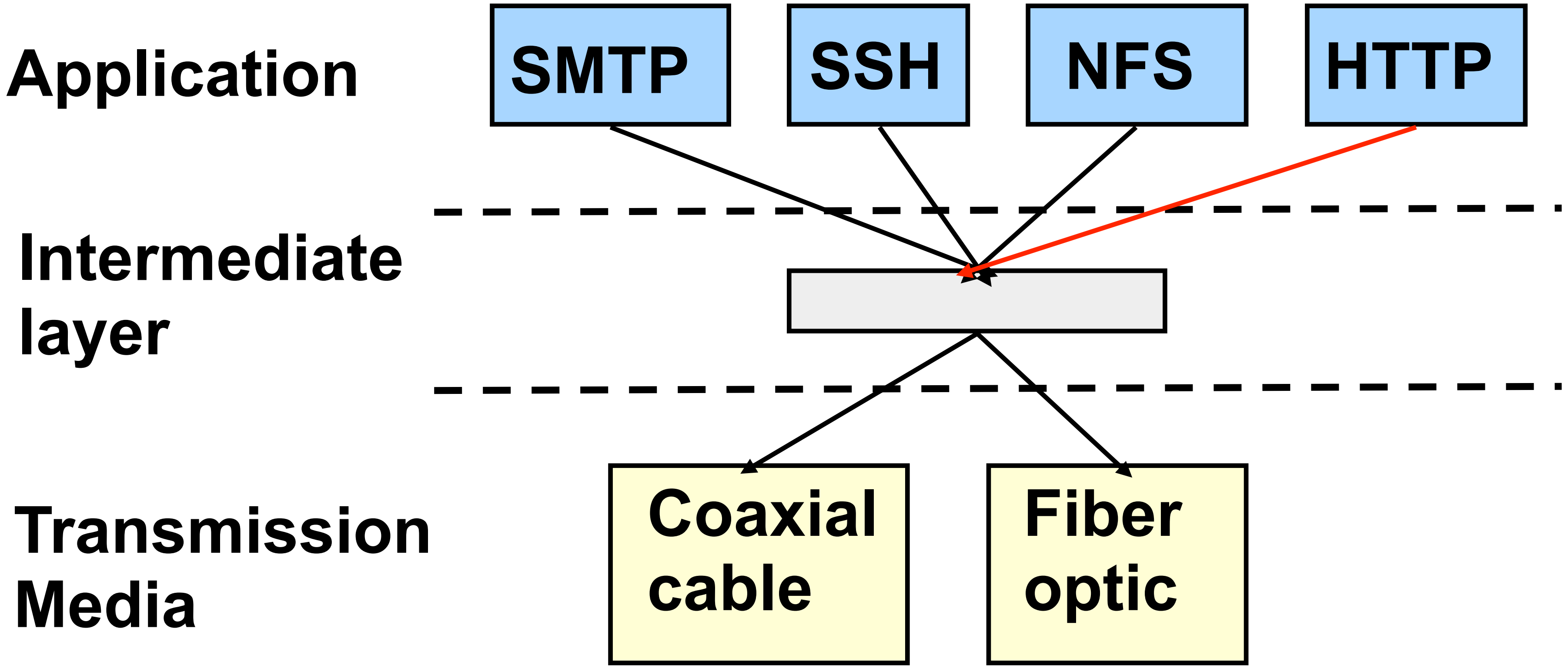
# Solution: Indirection

- Solution: introduce an intermediate layer that provides a single abstraction for various network technologies
  - O(1) work to add app/media
  - Indirection is an often used technique in computer science



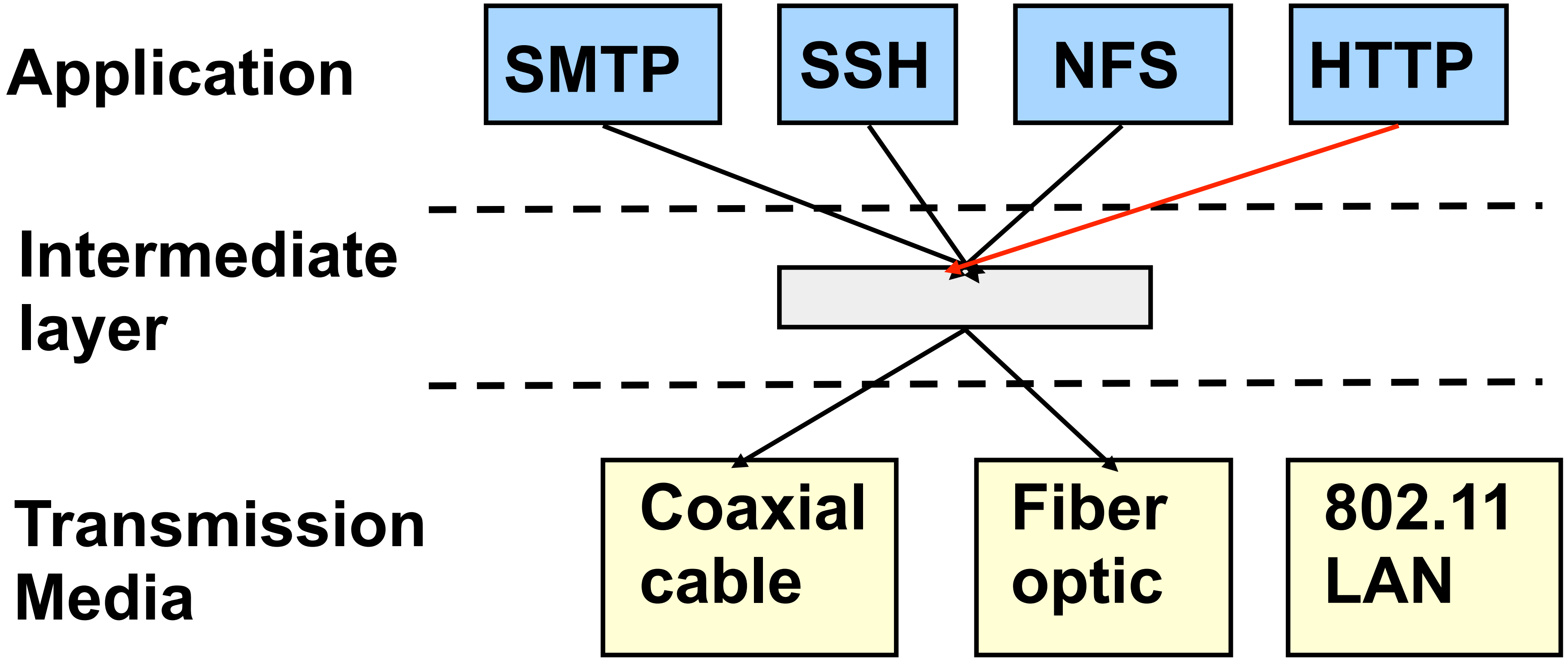
# Solution: Indirection

- Solution: introduce an intermediate layer that provides a single abstraction for various network technologies
  - O(1) work to add app/media
  - Indirection is an often used technique in computer science



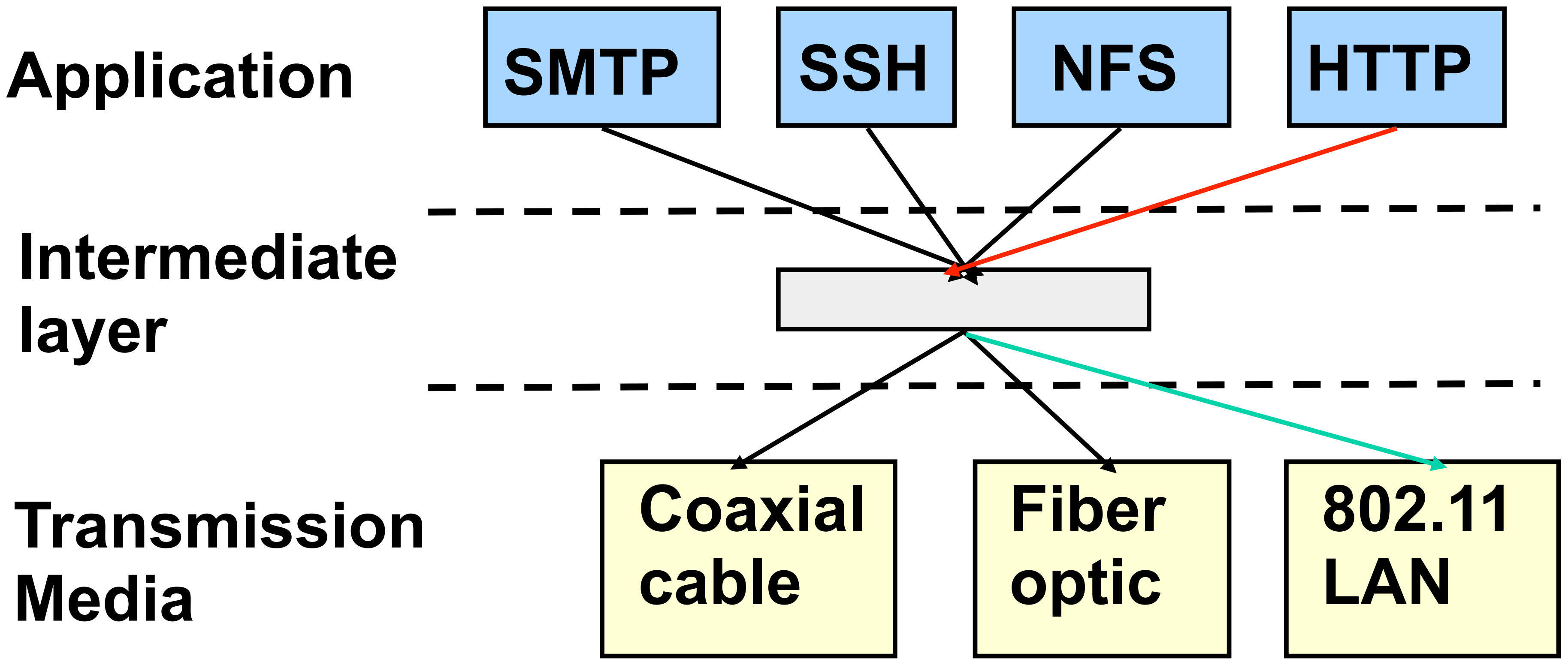
# Solution: Indirection

- Solution: introduce an intermediate layer that provides a single abstraction for various network technologies
  - O(1) work to add app/media
  - Indirection is an often used technique in computer science



# Solution: Indirection

- Solution: introduce an intermediate layer that provides a single abstraction for various network technologies
  - O(1) work to add app/media
  - Indirection is an often used technique in computer science



# Network Architecture

- Architecture is not the implementation itself

- Architecture is how to “organize” implementations
  - what interfaces are supported
  - where functionality is implemented

- Architecture is the modular design of the network

# Software Modularity

Break system into modules:

- Well-defined interfaces gives flexibility
  - can change implementation of modules
  - can extend functionality of system by adding new modules
- Interfaces hide information
  - allows for flexibility
  - but can hurt performance

# Network Modularity

Like software modularity, but with a twist:

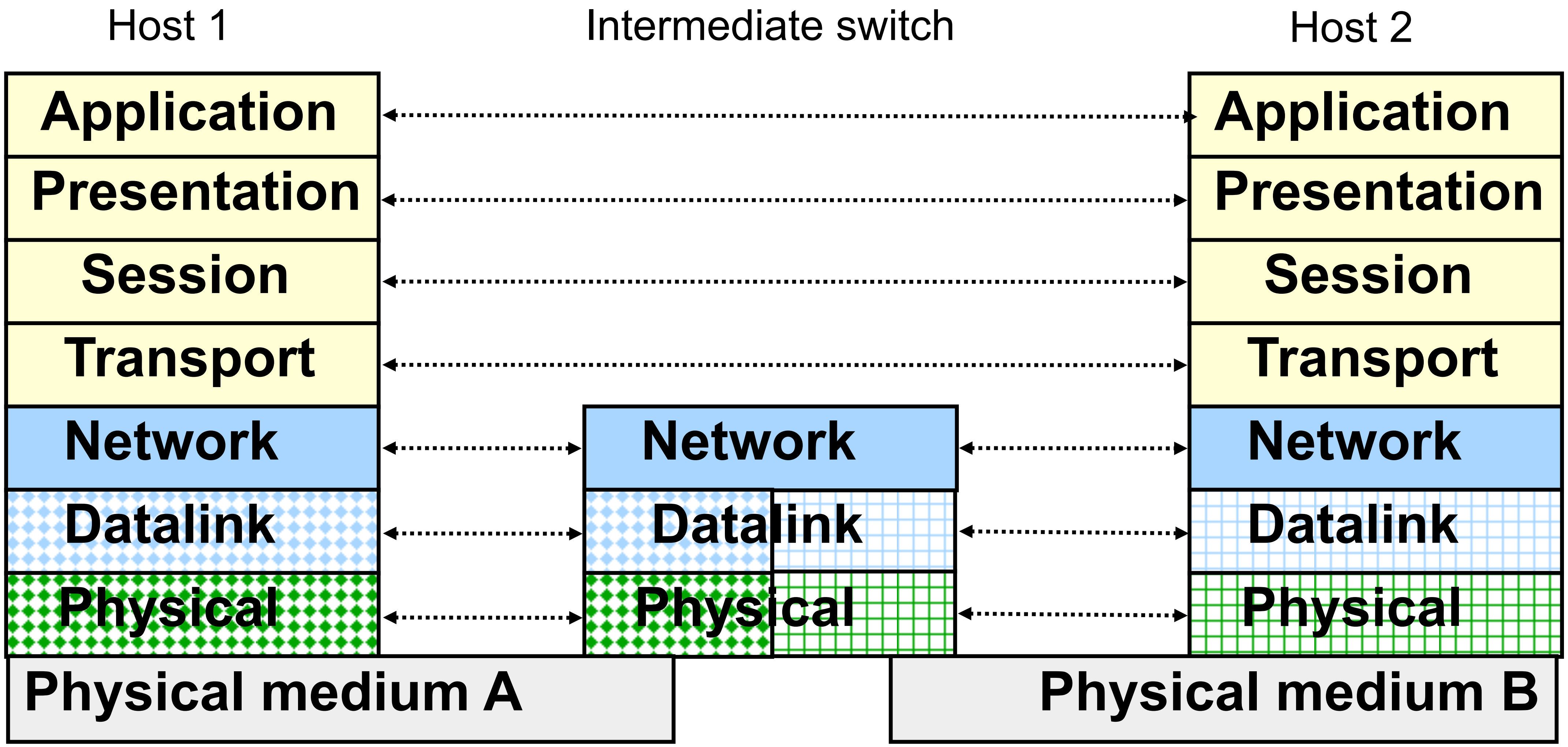
- Implementation distributed across routers and hosts
- Must decide both:
  - how to break system into modules
  - where modules are implemented

# Layering

- Layering is a particular form of modularization
- The system is broken into a vertical hierarchy of logically distinct entities (layers)
- The service provided by one layer is based solely on the service provided by layer below
- Rigid structure: easy reuse, performance suffers

# ISO OSI Reference Model

- Seven layers
  - Lower two layers are peer-to-peer
  - Network layer involves multiple switches
  - Next four layers are end-to-end



# Key Concepts

# Key Concepts

- Service – says what a layer does
  - Ethernet: unreliable subnet unicast/multicast/broadcast datagram service
  - IP: unreliable end-to-end unicast datagram service
  - TCP: reliable end-to-end bi-directional byte stream service
  - Guaranteed bandwidth/latency unicast service
- Service Interface – says how to access the service
  - E.g. UNIX socket interface
- Protocol – says how is the service implemented
  - a set of rules and formats that govern the communication between two peers

# Physical Layer (1)

- **Service**: move information between two systems connected by a physical link
- **Interface**: specifies how to send a bit
- **Protocol**: coding scheme used to represent a bit, voltage levels, duration of a bit
- Examples: coaxial cable, optical fiber links; transmitters, receivers

# Datalink Layer (2)

- **Service**:
  - framing (attach frame separators)
  - send data frames between peers
  - others:
    - arbitrate the access to common physical media
    - per-hop reliable transmission
    - per-hop flow control
- **Interface**: send a data unit (packet) to a machine connected to the same physical media
- **Protocol**: layer addresses, implement Medium Access Control (MAC) (e.g., CSMA/CD)…

# Network Layer (3)

- **Service**:
  - –deliver a packet to specified network destination
  - –perform segmentation/reassemble
  - –others:
    - packet scheduling
    - buffer management
- **Interface**: send a packet to a specified destination
- **Protocol**: define global unique addresses; construct routing tables

# Transport Layer (4)

- **Service**:
  - Multiplexing/demultiplexing
  - optional: error-free and flow-controlled delivery
- **Interface**: send message to specific destination
- **Protocol**: implements reliability and flow control
- Examples: TCP and UDP

# Session Layer (5)

- **Service**:
  - full-duplex
  - access management (e.g., token control)
  - synchronization (e.g., provide check points for long transfers)
- **Interface**: depends on service
- **Protocol**: token management; insert checkpoints, implement roll-back functions

# Presentation Layer (6)

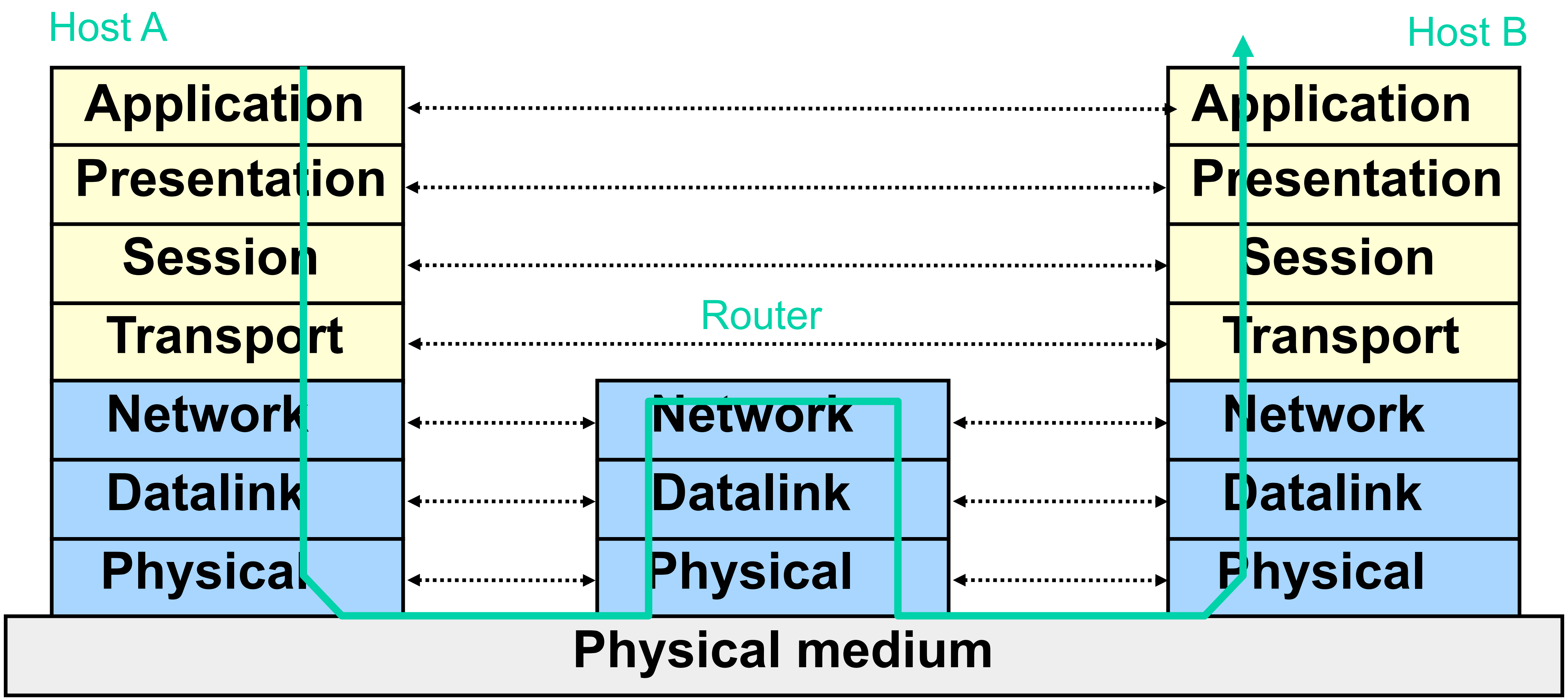
- **Service**: convert data between various representations
- **Interface**: depends on service
- **Protocol**: define data formats, and rules to convert from one format to another

# Application Layer (7)

- **Service**: any service provided to the end user
- **Interface**: depends on the application
- **Protocol**: depends on the application
- Examples: FTP, Telnet, WWW browser

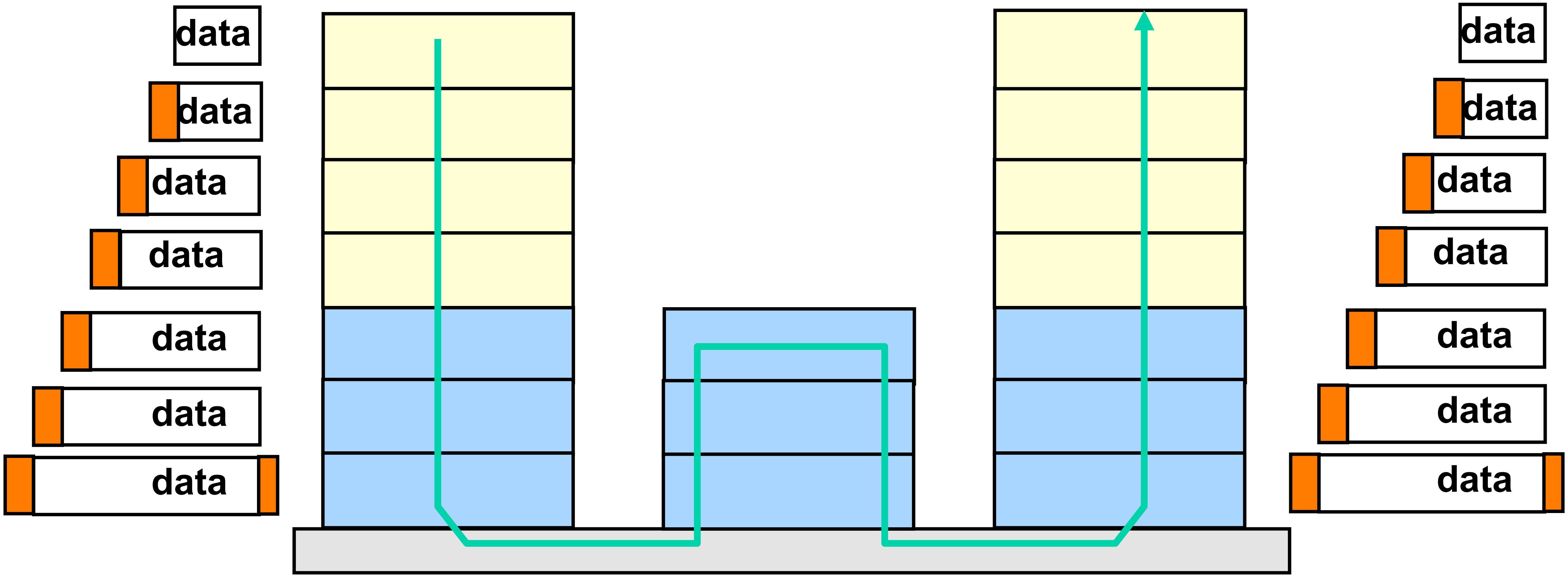
# Physical Communication

- Communication goes down to physical network, then to peer, then up to relevant layer



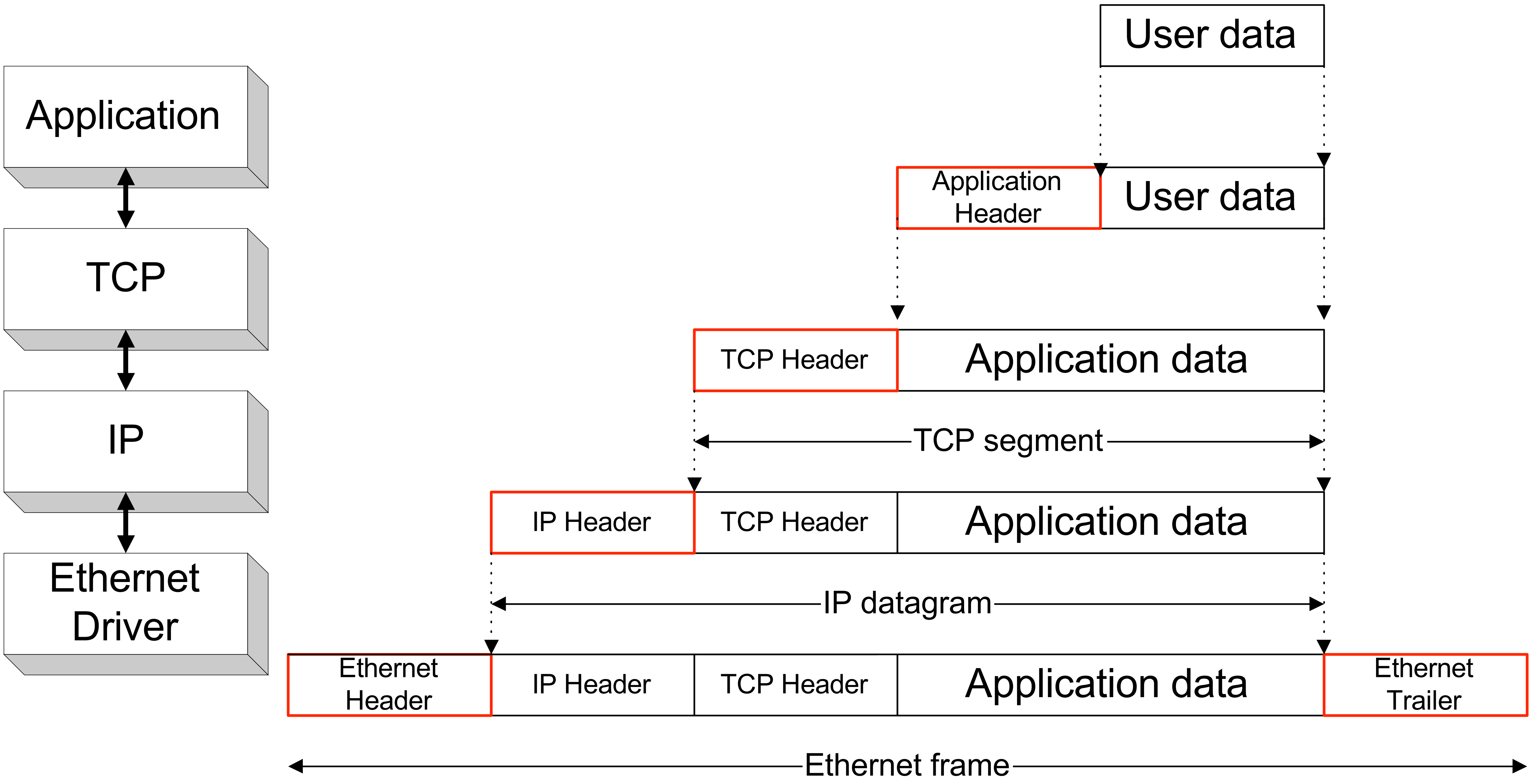
# Encapsulation

- A layer can use only the service provided by the layer immediate below it
- Each layer may change and add a header to data packet



# Encapsulation

- As data is moving down the protocol stack, each protocol is adding layer-specific control information.



# Example: Postal System

Standard process (historical):

- Write letter
- Drop an addressed letter off in your local mailbox
- Postal service delivers to address
- Addressee reads letter (and perhaps responds)

# Postal Service as Layered System

Layers:

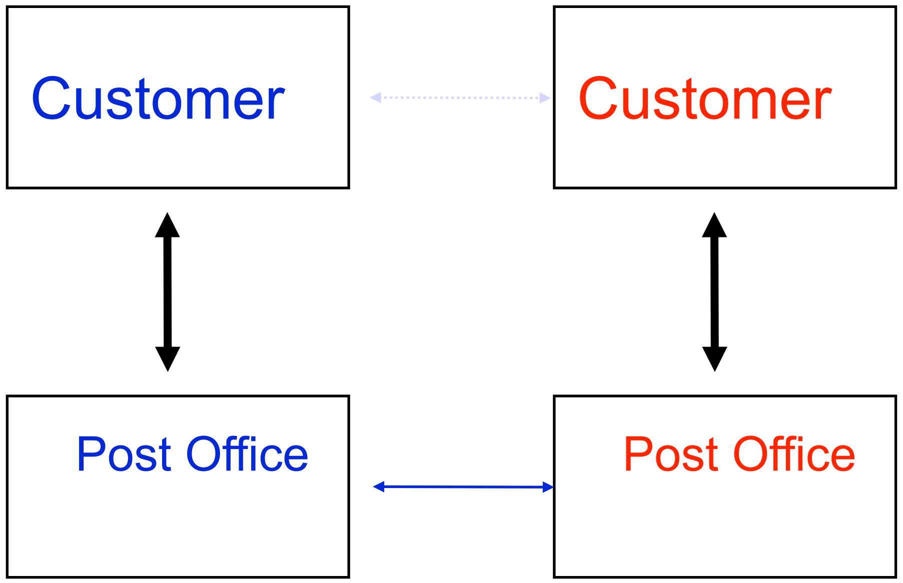
- Letter writing/reading
- Delivery

Information Hiding:

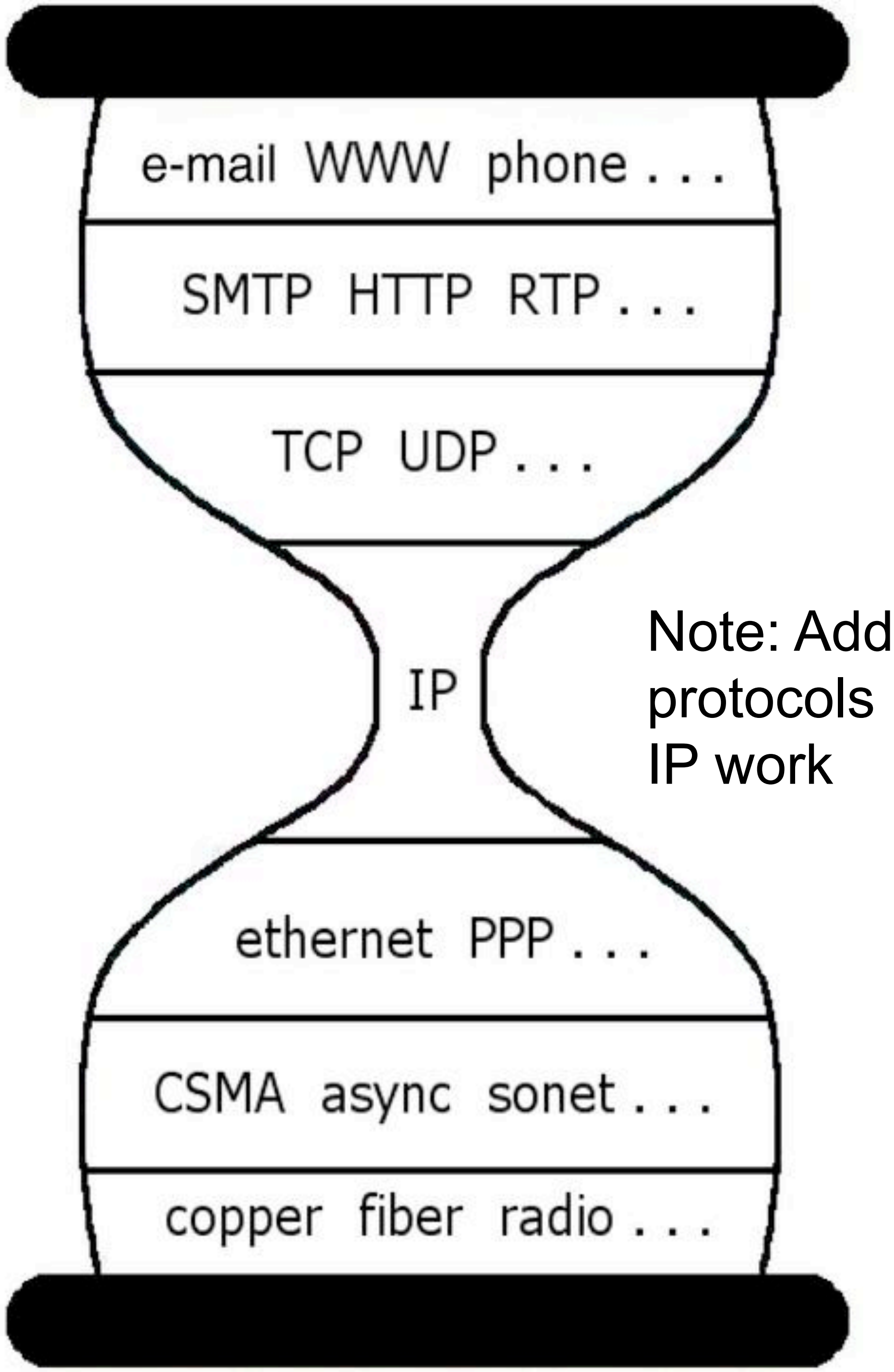
- Network need not know letter contents
- Customer need not know how the postal network works

Encapsulation:

- Envelope



# Hourglass



Note: Additional protocols like routing protocols (RIP, OSPF) needed to make IP work

# Implications of Hourglass

A single Internet layer module:

- Allows all networks to interoperate
  - all networks technologies that support IP can exchange packets
- Allows all applications to function on all networks
  - all applications that can run on IP can use any network
- Simultaneous developments above and below IP