Computer Networks

Introduction

Outline

- Why the internet?
- What is the Internet? What is a protocol?
- History.
- Network structure.
- Layered architecture.
- Encapsulation.

Warm Up Discussions

Question

- Your friend has some new MP3 files for you and you want them as soon as possible. If your friend lives 20 minutes away, and you have a cable model with 2Mbps, which way you prefer to get the CD?
  - (a) Ask your friend to bring you the CD-ROMs (containing 650 MB) immediately
  - (b) Download the contents from the Internet?

Several Facts

- 1 Byte = 8 bits
- KB = 2^{10} bytes
- Mbps = 10^6 bits per second

Solution to The Question

- *CD-ROM contains 650 MB = 5200 Mb.*
- *Carrying only one CD-ROM and given a travel time 20 min = 1200 sec.*
- *Time needed to download from internet*

  5200 Mb / 2Mbps = *2600 sec.*

It Sounds Counter-Intuitive

- *Why Internet then?*

It Sounds Counter-Intuitive

- *Why Internet then?*
- *What if your friend lives in Paris?*
  - *7 hours of flight time + 2 hour to and from airport + 1 hour of waiting = 10 hours*
  - *Internet speed reduces to 200 Kbps*

To Paris

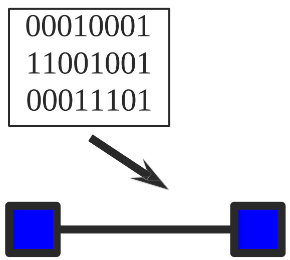
- *CD-ROM contains 650 MB = 5200 Mb.*
- *Carrying only one CD-ROM and given a travel time 10 hours = 36,000 sec.*
- *Time needed to download from internet*

  5200 Mb / 200 Kbps = 26,000 sec.

Internet Wins!

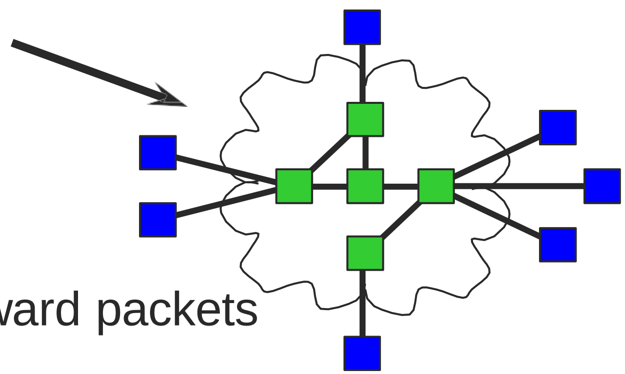
- 2600 sec (Internet) < 3600 (flight)
- In addition, …
  - Have you considered the money?
    - Flight tickets, fuel cost, …
  - Have you considered the time spent by your friend?

The Big Picture

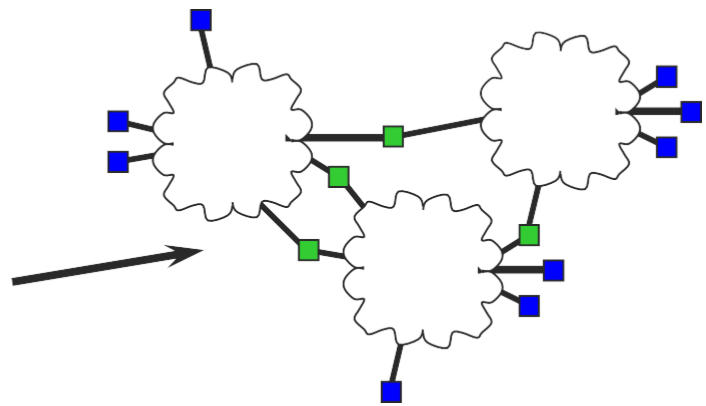


Nodes & Links

Direct link networks
Local area networks:
collection of devices, routers, links managed by an organization



Network of networks,
Internet



Packet switches: forward packets (chunks of data)

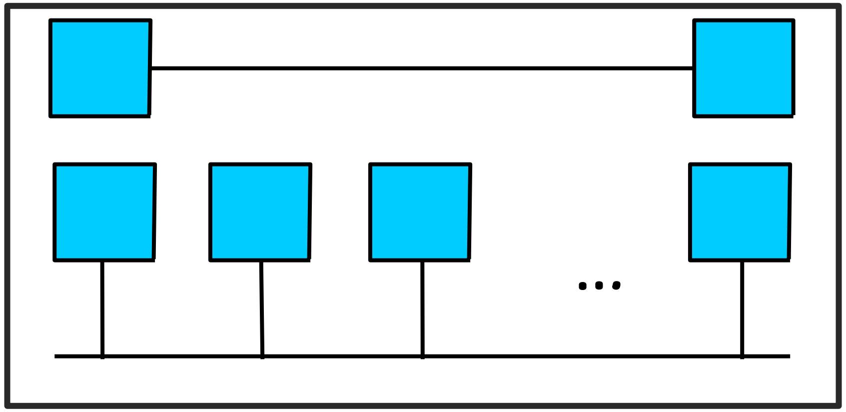
- routers, switches

Computer Networks



- Building Blocks
  - Communication links: coax(ial) cable, Twisted pair, optical fiber, Radio link types…
    - Transmission rate: bandwidth.
  - Point-to-point
  - Multiple access

    Example?



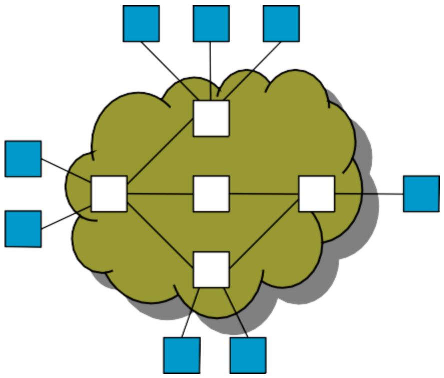
Computer Networks

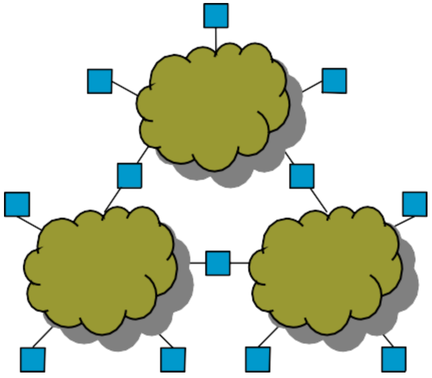
- Nodes: ■ ■
  - Hosts: End systems running network apps at Internet’s “edge”. E.g., workstations, security cam, gaming device, cars, etc.

  - Packet switches: forward packets (chunk of data) such as routers, switches.

A Definition

- Recursive definition of a network
  - Two or more nodes connected by a physical link
  - Two or more networks connected by one or more nodes (i.e. router/gateway)
  - i.e. a network can be constructed from a nesting of networks, where at the bottom level, the network is implemented by some physical medium.





History of Network Research

Computer Networks

1960 ARPANET

1970 First Email Sent

NCP

1980 ARPANET, MILNET

Internet

TCP

1990 WWW was born

2000 Home Broadband

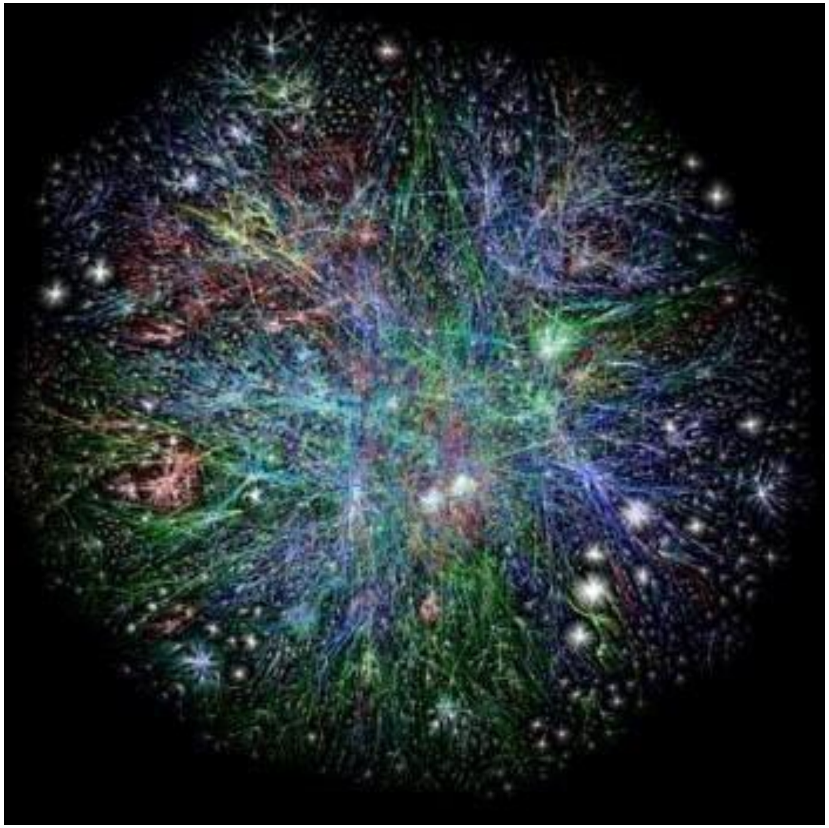
P2P: Napster

Sensor Networks

SDN

2010 Enriched apps & services

Pervasive mobile devices



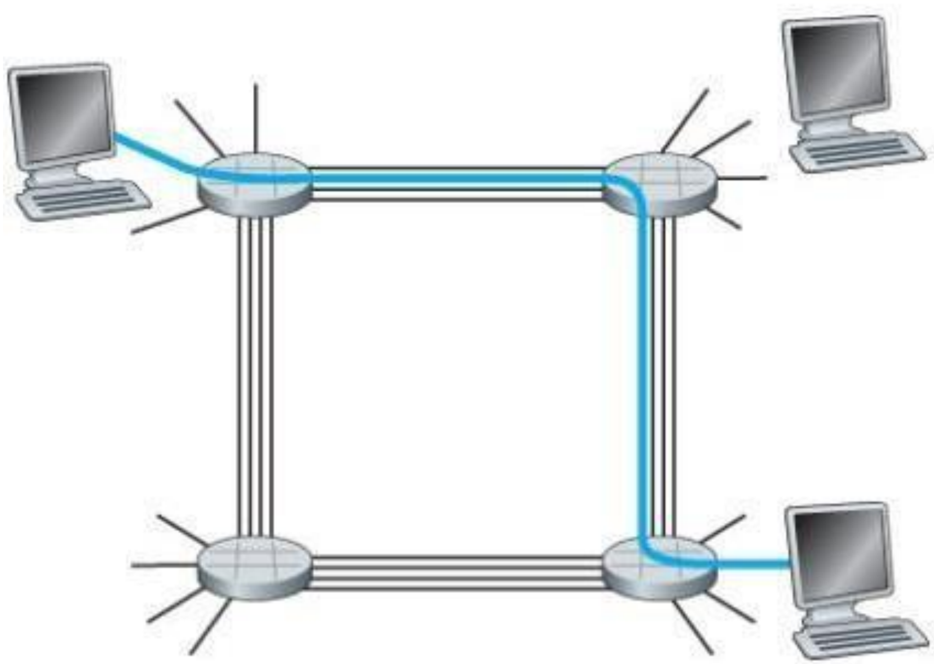
Computer Networks

- What must a network provide?
  - Scalable Connectivity
  - Cost-Effective Resource Sharing
  - Support for Common Services
  - Manageability

Computer Networks

- What must a network provide?
  - Connectivity   (and scalable)
    - Packet switched (store and forward) v.s. circuit switched
    - a set of independent networks (clouds) are interconnected to form an internetwork, or internet for short.
    - internet v.s. Internet (with TCP/IP)
    - A node that is connected to two or more networks is commonly called a router or gateway.

Circuit-switched network



A simple circuit-switched network consisting of four switches and four links

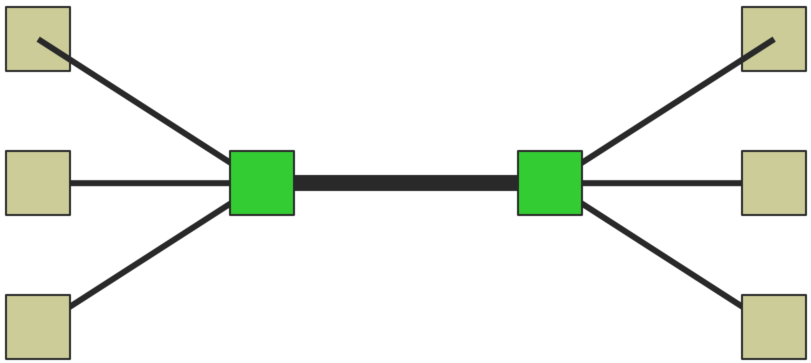
Computer Networks

- What must a network provide?
  - Scalable Connectivity
  - Cost-Effective Resource Sharing
  - Support for Common Services
  - Manageability

Sharing of Resources

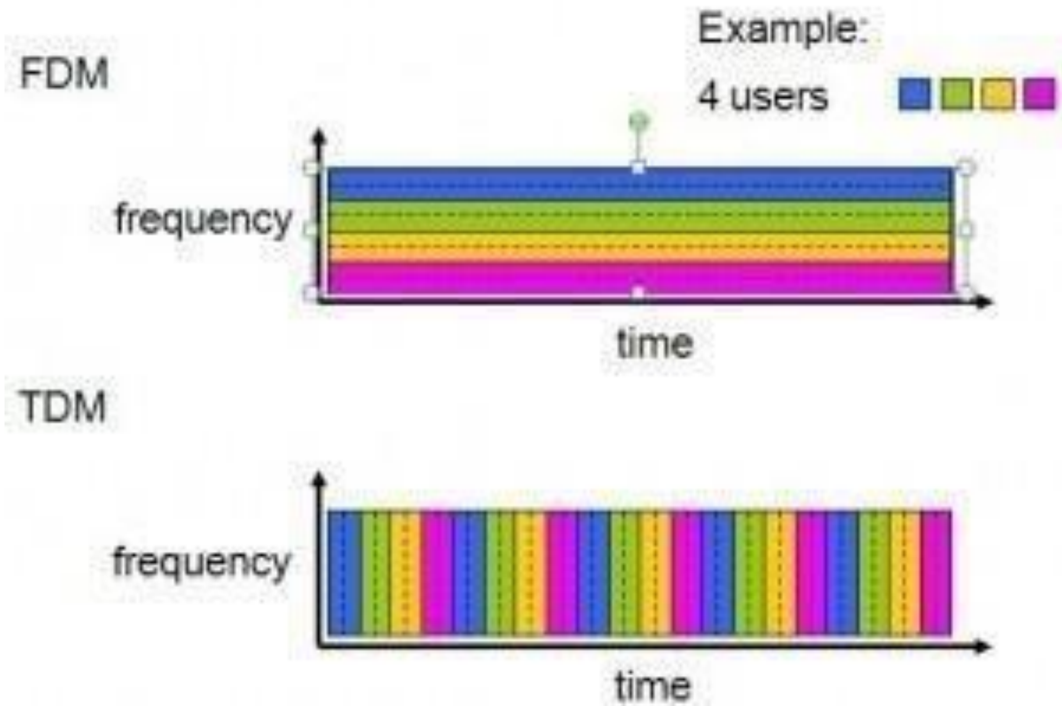
How do several hosts share the same link when they all want to use it at the same time?

- Physical links and switches must be shared among many users



- Common multiplexing strategies
  - (Synchronous) time-division multiplexing (TDM)
  - Frequency-division multiplexing (FDM)

TDM v.s. FDM

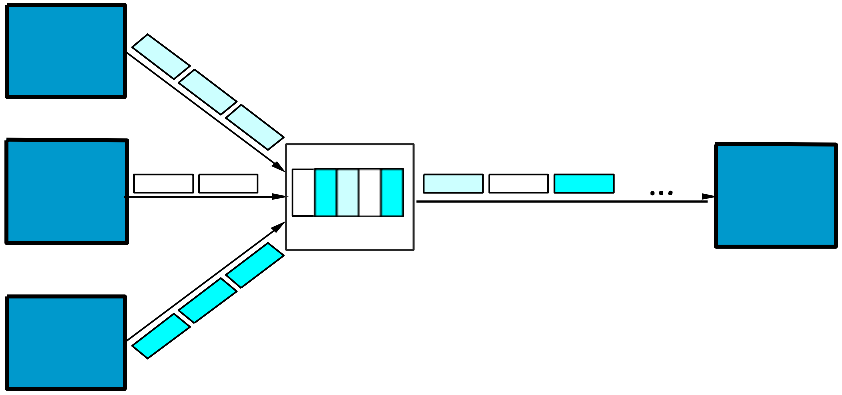


Drawbacks

- Waste of resources
  - If one of the flows (host pairs) does not have any data to send, its share of the physical link—that is, its time quantum or its frequency—remains idle, even if one of the other flows has data to transmit.
- Flexibility
  - both STDM and FDM are limited to situations in which the maximum number of flows is fixed and known ahead of time. It is not practical to resize the quantum or to add additional quanta.

Statistical Multiplexing in a Switch

- Packets buffered in switch until forwarded
  - On-demand transmission, instead of a predetermined time slot
- Selection of next packet depends on policy
  - How do we make these decisions in a fair manner? Round Robin? FIFO?
  - How should the switch handle congestion?



Computer Networks

- What must a network provide?
  - Scalable Connectivity
  - Cost-Effective Resource Sharing
  - Support for Common Services
  - Manageability

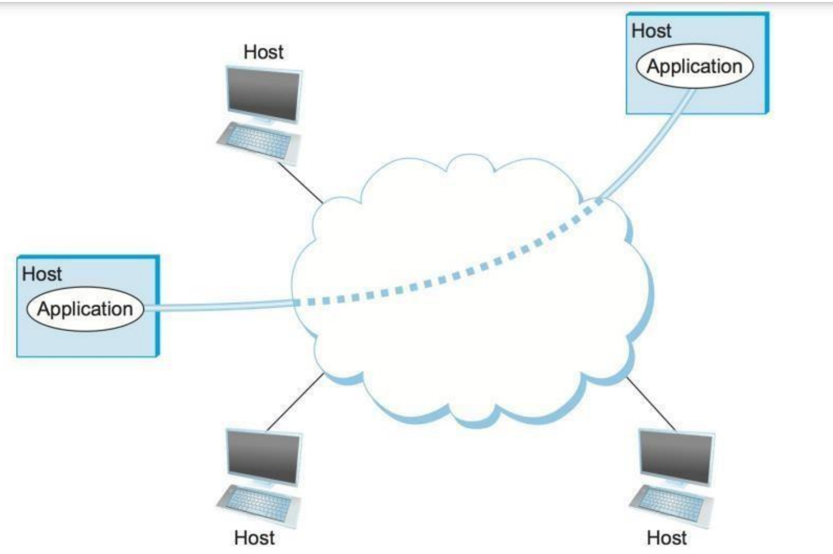
Support For Common Services

- Provide meaningful communication between hosts on a network
- Common services simplify the role of applications
- Hide the complexity of the network without overly constraining the application designer
- The challenge for a network designer is to identify the right set of common services.

Support For Common Services

- Packet delivery failure? Same order?
- Privacy of data.

Network provides a variety of different types of channels, with each application selecting the type that best meets its needs.



Computer Networks

- What must a network provide?
  - Scalable Connectivity
  - Cost-Effective Resource Sharing
  - Support for Common Services
  - Manageability

Network Performance

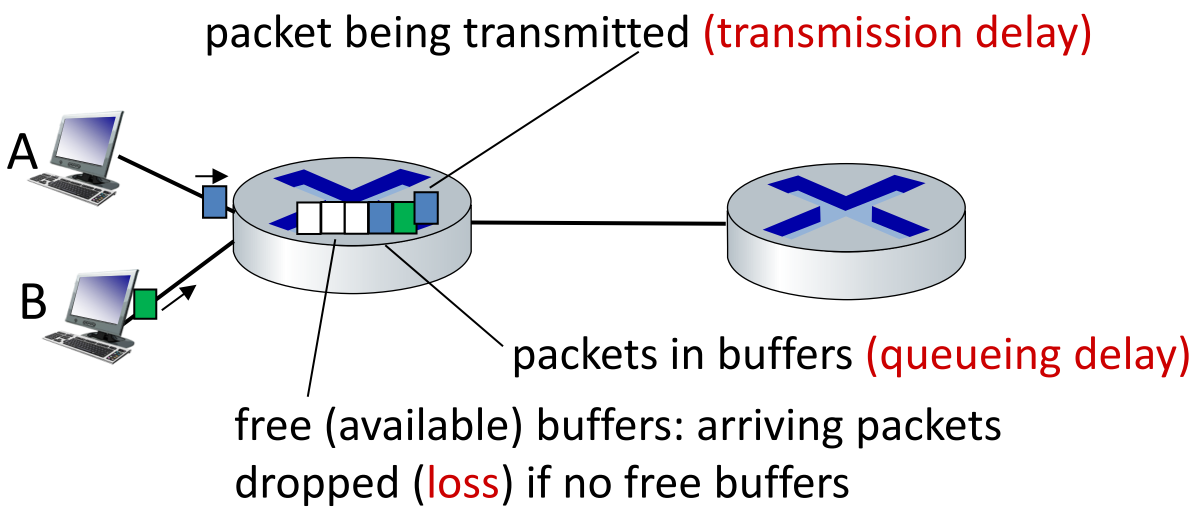
- Packet loss
- Packet delay
- Throughput

How do packet delay and loss occur?

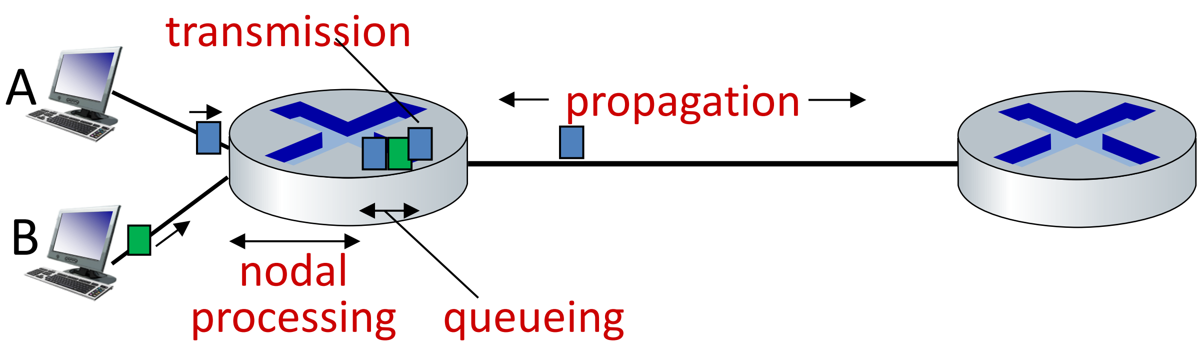
packets *queue* in router buffers, waiting for turn for transmission

- queue length grows when arrival rate to link (temporarily) exceeds output link capacity

- packet *loss* occurs when memory to hold queued packets fills up



Packet delay: four sources



$$d_{total} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}}$$

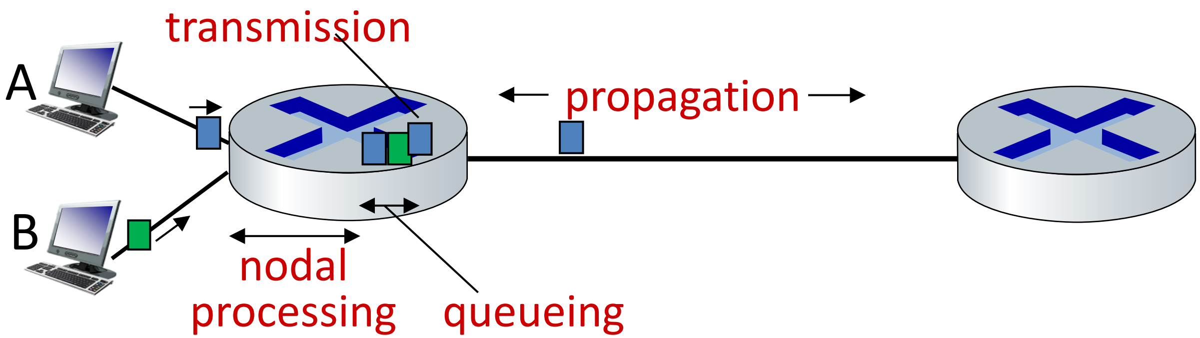
d_{proc}: nodal processing

- check bit errors
- determine output link
- typically < microsecs

d_{queue}: queueing delay

- time waiting at output link for transmission
- depends on congestion level of router

Packet delay: four sources



$$d_{total} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}}$$

d_{trans}: transmission delay:

- *L*: packet length (bits)
- *R*: link *transmission rate (bps)*
- $d_{trans} = L/R$

d_{prop}: propagation delay:

- *d*: length of physical link
- *s*: propagation speed (~2x10^8 m/sec)
- $d_{\text{prop}} = d/s$

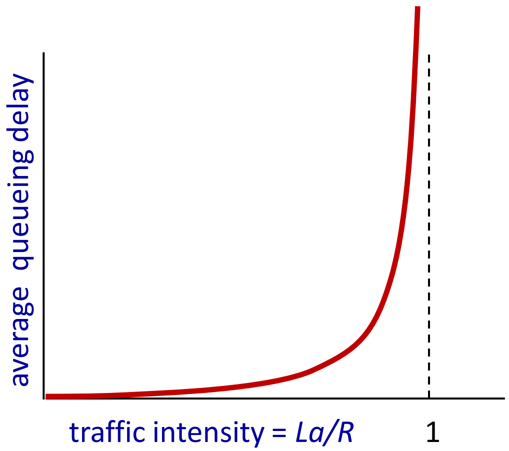
d_{trans} and d_{prop}
very different

Packet queueing delay (revisited)

- *a:* average packet arrival rate
- *L:* packet length (bits)
- *R:* bit transmission rate

$$\frac{L \cdot a}{R} : \quad \frac{\text{arrival rate of bits}}{\text{Transmission rate of bits}}$$

"traffic intensity"



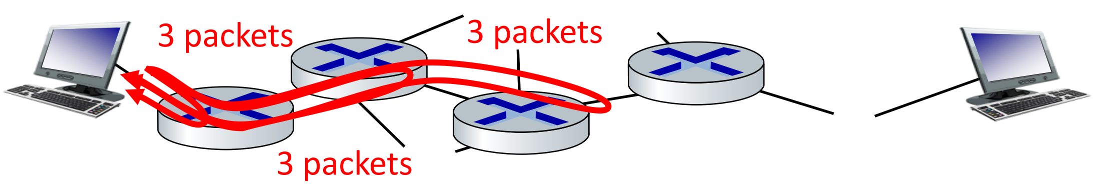
- *La/R* ~ 0: avg. queueing delay small
- *La/R* -> 1: avg. queueing delay large
- *La/R* > 1: more "work" arriving is more than can be serviced - average delay infinite!

La/R ~ 0

La/R -> 1

"Real" Internet delays and routes

- what do "real" Internet delay & loss look like?
- **traceroute** program: provides delay measurement from source to router along end-end Internet path towards destination. For all *i:*
  - sends three packets that will reach router *i* on path towards destination (with time-to-live field value of *i*)
  - router *i* will return packets to sender
  - sender measures time interval between transmission and reply



Try it out:

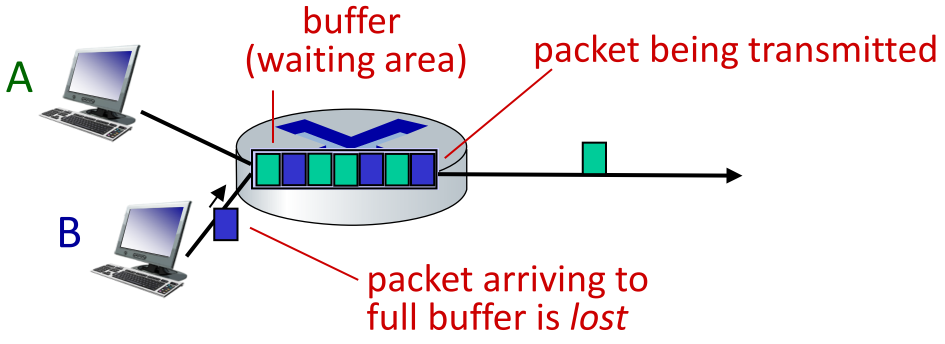
Windows command: tracert <Domain name or IP address to trace>

Linux and MAC: traceroute <Domain name or IP address to trace>

* Do some traceroutes from exotic countries at www.traceroute.org

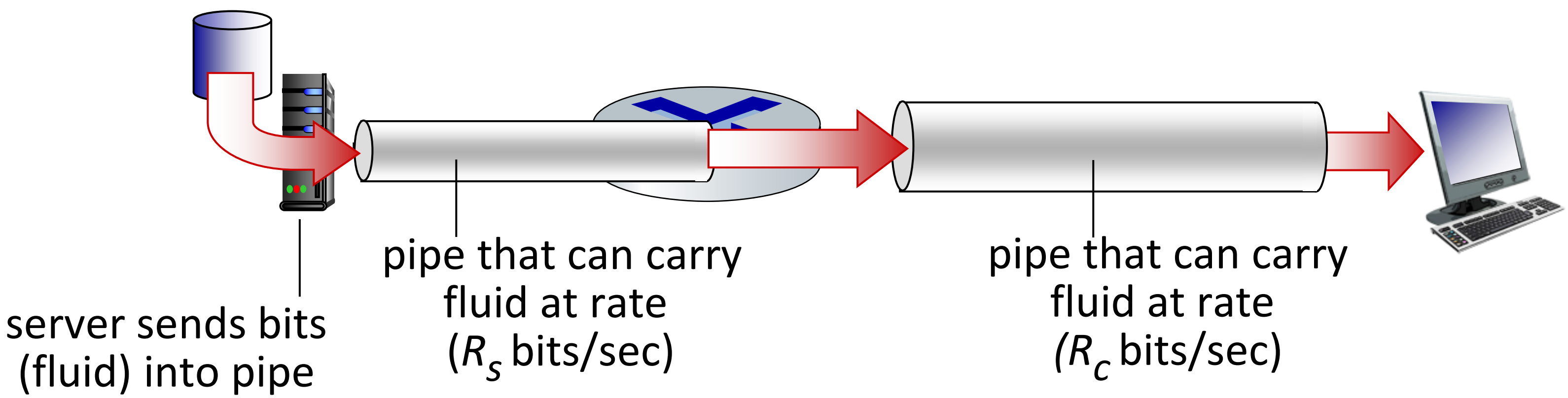
Packet loss

- queue (aka buffer) preceding link in buffer has finite capacity
- packet arriving to full queue dropped (aka lost)
- lost packet may be retransmitted by previous node, by source end system, or not at all



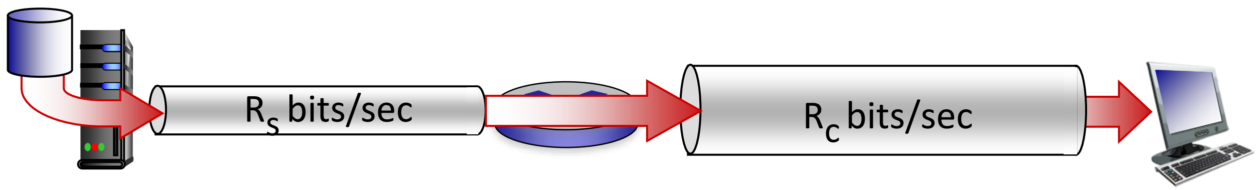
Throughput

- *Throughput:* the actual rate (bits/time unit)) at which bits are being sent from sender to receiver over the network.

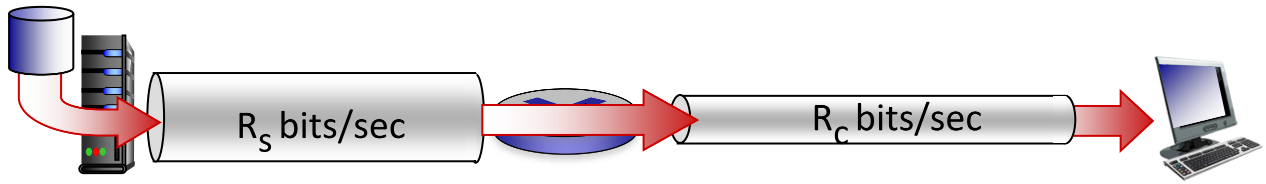


Throughput

$R_s < R_c$ What is average end-end throughput?



$R_s > R_c$ What is average end-end throughput?



bottleneck link

link on end-end path that constrains end-end throughput

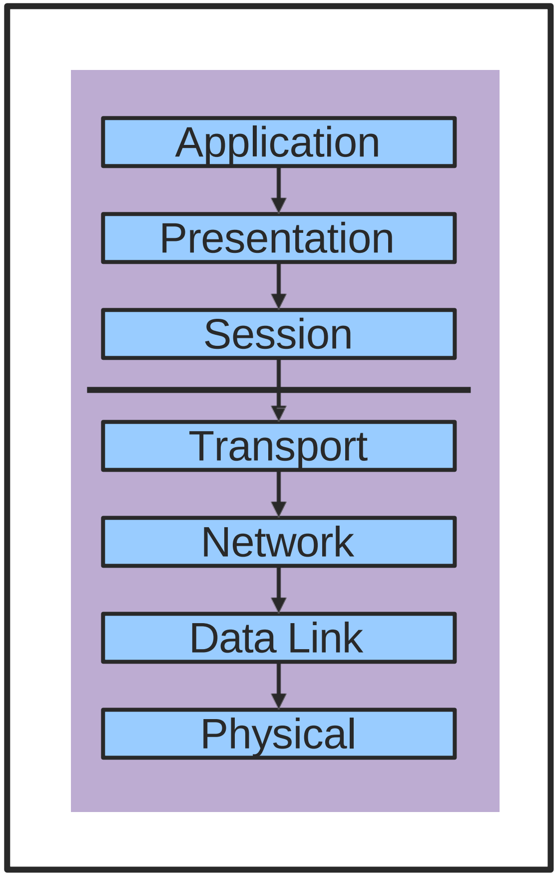
Network architecture

- How are networks designed and built?
  - Layering
  - Protocols
  - Standards
    - IETF: Internet Engineering Task Force -- > RFC: Request for Comments
    - ISO: International Standards Organization.

Why layering?

- Layering provides two nice features
  - First, it decomposes the problem of building a network into more manageable components.
    - Not monolithic software, by layering
  - Second, it provides a more modular design.
    - If add some new service, you only need to modify the functionality at one layer, reusing the functions provided at all the other layers.

OSI Protocol Stack



- OSI: Open Systems Interconnection
- Application: Application specific protocols
- Presentation: Format of exchanged data, data translator (convert ***coded file to ASCII-coded file)
- Session: Name space for connection mgmt (request/response. Eg. RPC)
- Transport: Process-to-process channel (connection-oriented, flow control, reliability, e.g. TCP)
- Network: Host-to-host packet delivery (Packet forwarding, routing)
- Data Link: Framing of data bits (including error correction, e.g MAC: CSMA/CD)
- Physical: Transmission of raw bits

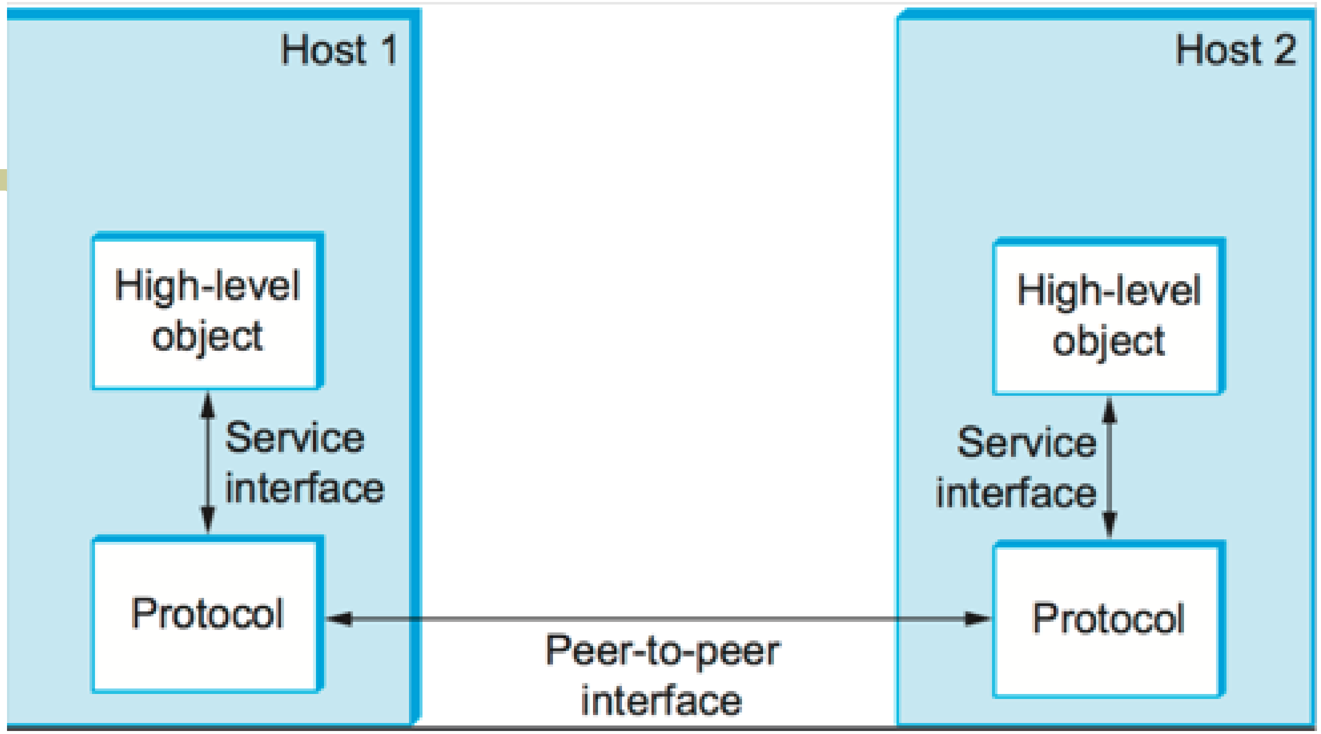
Protocol

- A protocol defines the *format, order of messages sent and received* among network entities, and *actions taken* on msg transmission and receipt.
- Example:
  - Request/reply
  - Message streaming protocol

Protocol

- Each protocol defines two different interfaces
  - 1st, it defines *a service interface* to the other objects on the same computer that want to use its communication services.
  - 2nd, a protocol defines *a peer interface* to its counterpart (peer) on another machine.

protocol defines a communication service that it exports locally (the service interface), along with a set of rules governing the messages that the protocol exchanges with its peer(s) to implement this service (the peer interface).

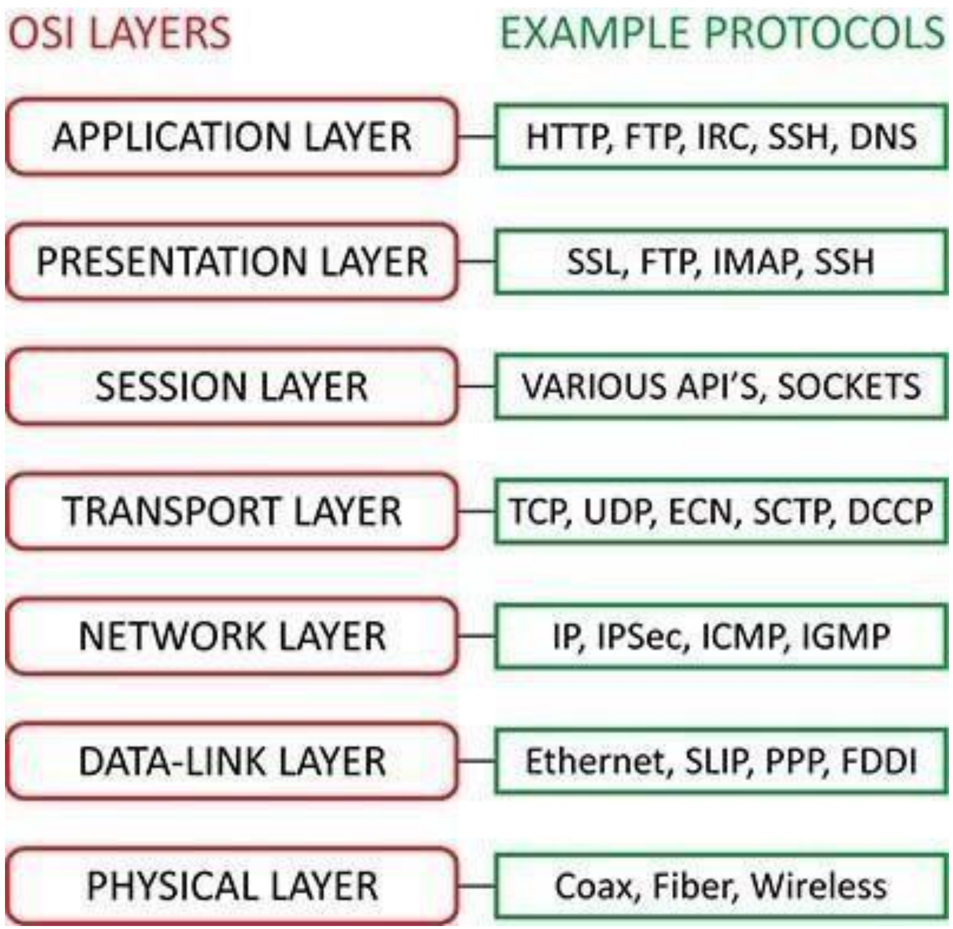


- HTTP protocol specification defines in detail how a GET command is formatted, what arguments can be used with the command, and how a web server should respond when it receives such a command.

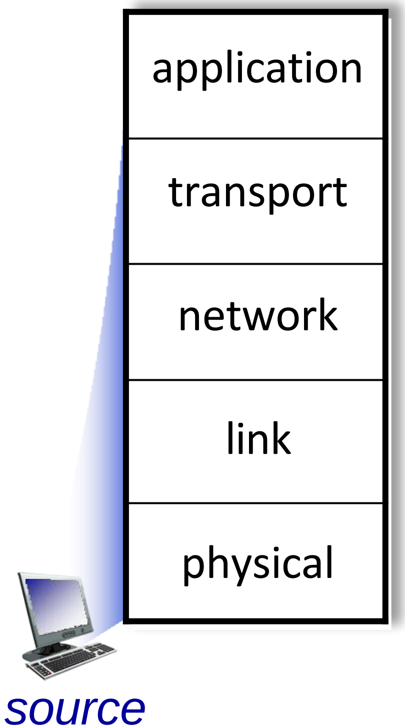
Protocol Acronyms

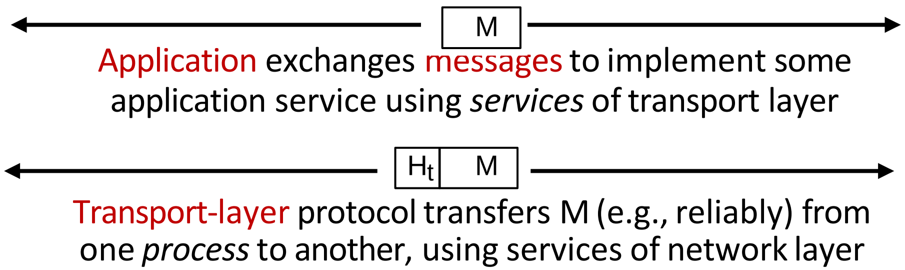
- FTP - (File Transfer Protocol
- HTTP - HyperText Transfer Protocol
- SMTP – Simple Mail Transfer Protocol
- DHCP - Dynamic Host Configuration Protocol
- TCP - Transmission Control Protocol
- UDP - User Datagram Protocol
- IP - Internet Protocol
- ARP - Address Resolution Protocol
- DNS - Domain Name Server/System

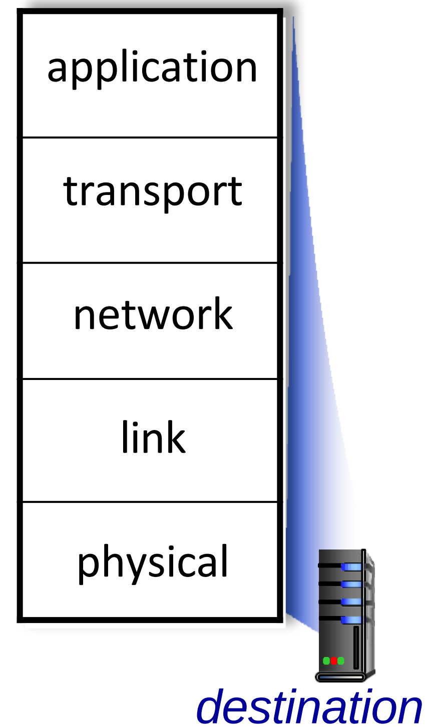
OSI Layers & Protocols



Layering and Encapsulation







- transport-layer protocol encapsulates application-layer message, M, with *transport* layer-layer header H_t to create a transport-layer segment
  - H_t used by transport layer protocol to implement its service

Layering and Encapsulation

application | transport | network | link | physical

M

H_t | M

Transport-layer protocol transfers M (e.g., reliably) from one *process* to another, using services of network layer

H_n | H_t | M

Network-layer protocol transfers transport-layer segment [H_t | M] from one *host* to another, using link layer services

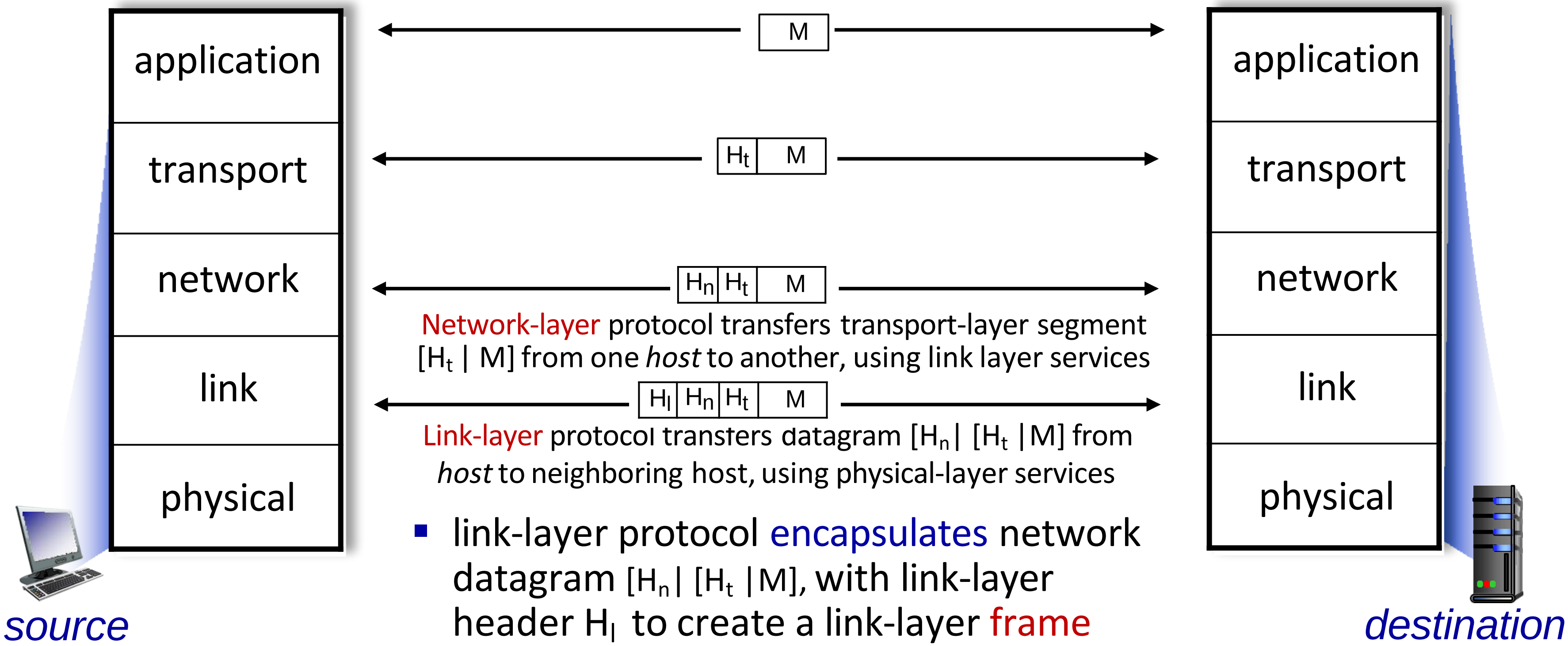
- network-layer protocol encapsulates transport-layer segment [H_t | M] with network layer-layer header H_n to create a network-layer datagram
  - H_n used by network layer protocol to implement its service

application | transport | network | link | physical

source

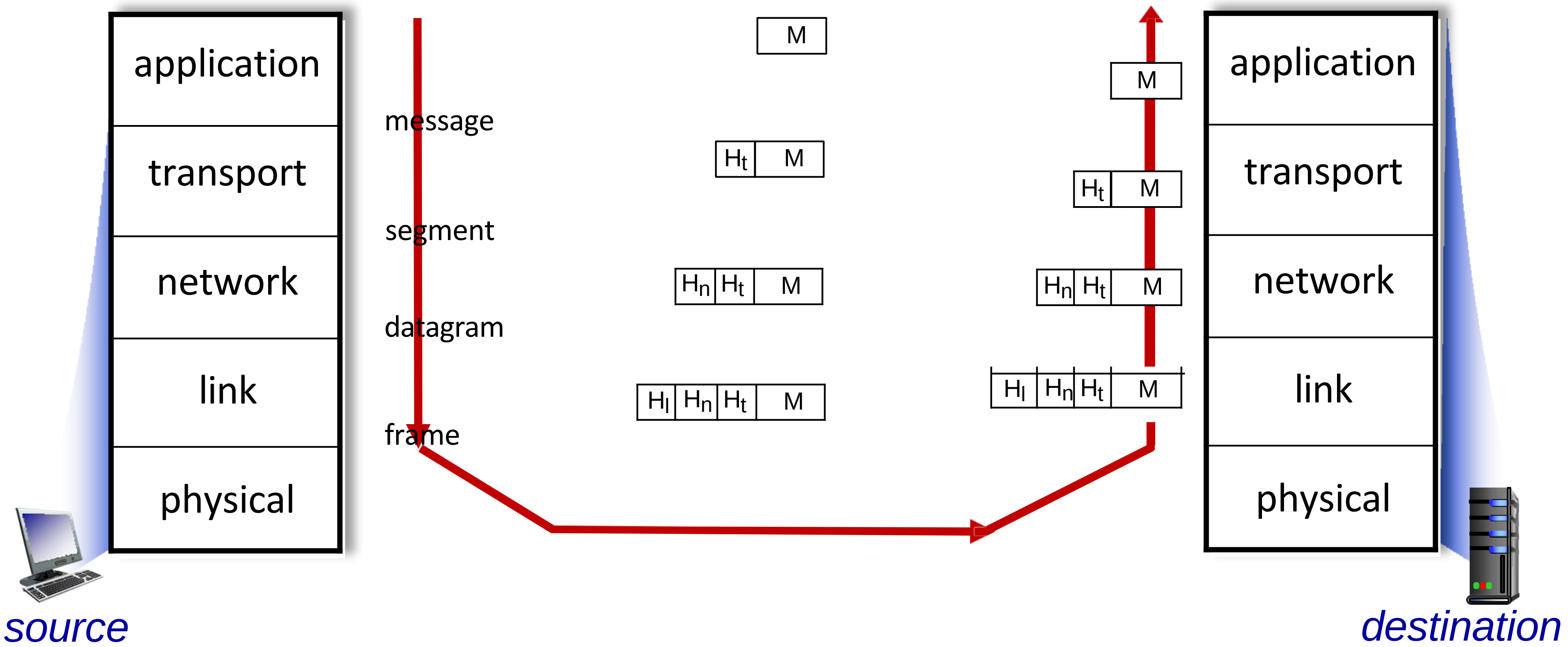
destination

Layering and Encapsulation



- link-layer protocol encapsulates network datagram $[H_n|\ [H_t\ |M]$, with link-layer header H_l to create a link-layer frame

Layering and Encapsulation



Encapsulation: an end-end view