Department of
Information Sciences and Telecommunications

Computer Networks – TELCOM 2310

Lecture 2

Network Architecture and Protocols

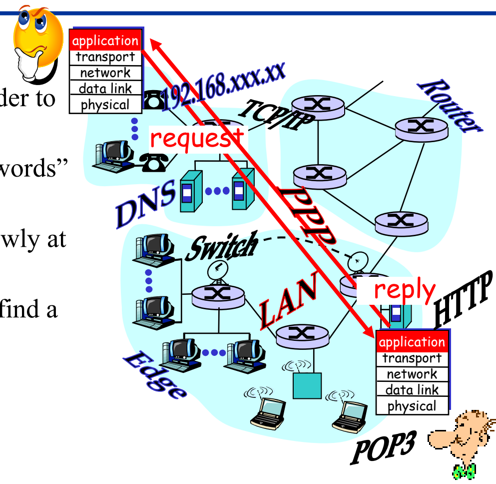
Prof. Taieb Znati

Outline

- Architecture Design Issues and Protocols
  - Network edge and core
  - Layering and Protocols
- OSI Layered Architecture
- Internet Architecture and Design Principles
  - Internet Structure
  - User, networking and service views
  - Delay and Loss in Packet Switching

Internet – A User's View

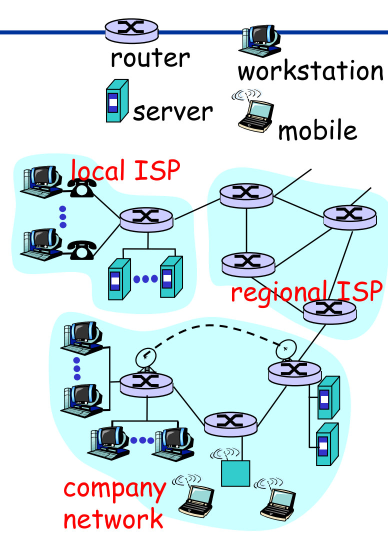
- What really happens when …?
- How does email get from the sender to the receiver?
- What do all these network "buzzwords" mean?
- Why do my browsers respond slowly at times?
- How does an IP address actually find a web site?



Internet – A "Nuts and Bolts" View

- Millions of connected computing devices: *hosts, end-systems*
  - PCs workstations, servers
  - PDAs, phones, toasters, ….

  running *network applications*
- *Communication links*
  - Fiber, copper, radio, satellite
  - Transmission rate = ***bandwidth***
- *Routers:* forward packets (chunks of data)



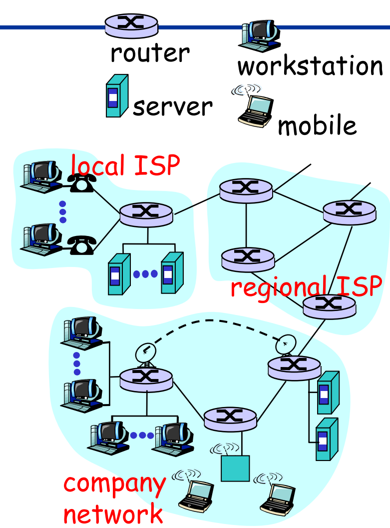
Internet of Appliances

IP picture frame
http://www.ceiva.com/

Web-enabled toaster+weather forecaster

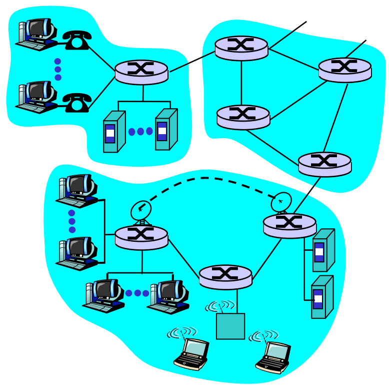
Internet – A Networking View

- *Protocols* control sending, receiving of msgs
  - e.g., TCP, IP, HTTP, FTP, PPP
- *Internet:* "network of networks"
  - loosely hierarchical
  - public Internet versus private intranet
- Internet standards
  - RFC: Request for comments
  - IETF: Internet Engineering Task Force



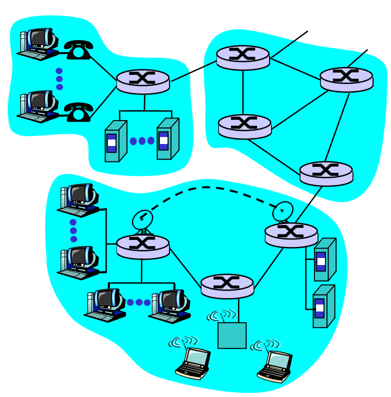
Internet – A Service View

- Communication *infrastructure:*
  - It enables distributed applications:
    - Web, email, games, e-commerce, database access, voting, file sharing (MP3), …
- Communication services provided to applications:
  - Connectionless
  - Connection-oriented



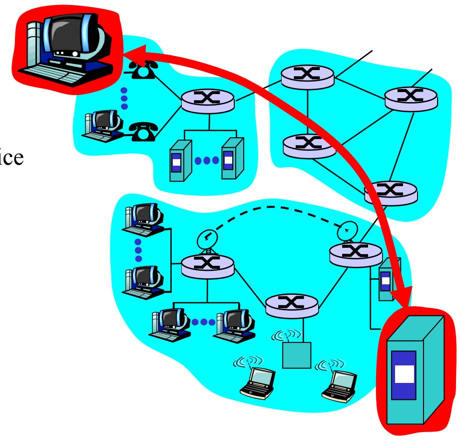
A Closer Look at Network Structure

- Network edge: applications and hosts
- Network core:
  - routers
  - network of networks
- Network access, physical media: communication links



The Network Edge

- End systems (hosts):
  - run application programs
  - e.g. Web, email
  - at "edge of network"
- Client/Server model
  - client host requests, receives service from always-on server
  - e.g. Web browser/server; email client/server
- Peer-to-peer model:
  - minimal (or no) use of dedicated servers
  - e.g. Gnutella, KaZaA, …



Network Edge
Connection-oriented Service

Goal: data transfer between end systems

- *Handshaking:* setup (prepare for) data transfer ahead of time
  - Hello, hello back human protocol
  - *set up "state"* in two communicating hosts
- TCP - Transmission Control Protocol
  - Internet's connection-oriented service

TCP service [RFC 793]

- *Reliable, in-order* byte-stream data transfer
  - loss: acknowledgements and retransmissions
- *Flow control:*
  - sender won't overwhelm receiver
- *Congestion control:*
  - senders "slow down sending rate" when network congested

Network Edge
Connectionless Service

Goal: data transfer between end systems
- same as before!

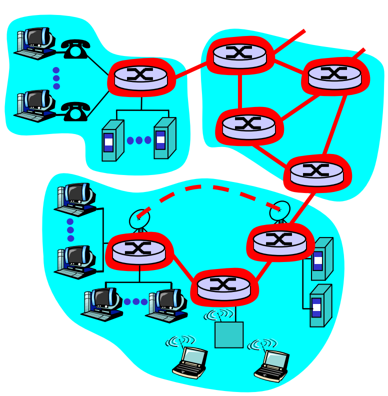
- UDP - User Datagram Protocol [RFC 768]: Internet's connectionless service
  - Unreliable data transfer
  - No flow control
  - No congestion control

Applications using TCP:
- HTTP (Web), FTP (file transfer), Telnet (remote login), SMTP (email)

Applications using UDP:
- Streaming media, teleconferencing, DNS, Internet telephony

The Network Core

- Mesh of interconnected routers
- *The* fundamental question: how is data transferred through net?
  - Circuit switching: Dedicated circuit per call
    - Telephone network
  - Packet-switching: Data sent through the network in discrete "chunks"



Network Core: Packet Switching

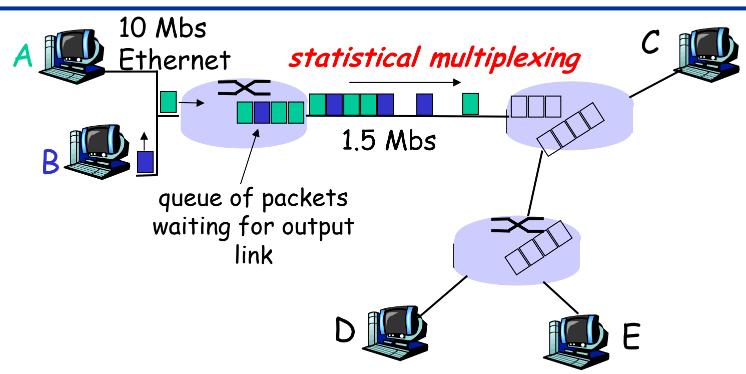
Each end-to-end data stream is divided into *packets*

- user A, B packets *share* network resources
- each packet uses full link bandwidth
- resources used *as needed*

Resource contention:

- Aggregate resource demand can exceed amount available
- Congestion: packets queue, wait for link use
- Store and forward: packets move one hop at a time
  - Transmit over link
  - Wait turn at next link

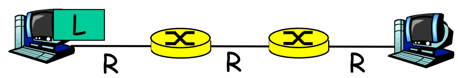
Packet Switching
Statistical Multiplexing



Sequence of A & B packets does not have fixed pattern ➔ ***statistical multiplexing***.

In TDM each host gets same slot in revolving TDM frame.

Packet-switching
Store-and-forward

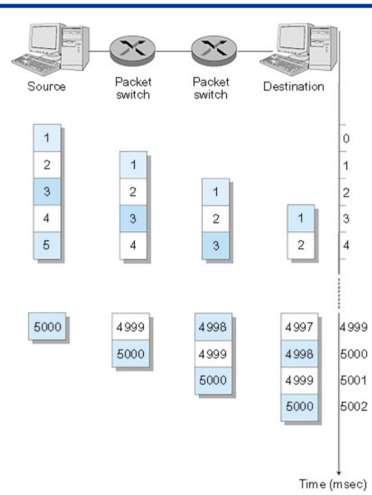


- Takes L/R seconds to transmit (push out) packet of L bits on to link or R bps
- Entire packet must arrive at router before it can be transmitted on next link: *store and forward*
- delay = 3L/R

Example:

- L = 7.5 Mbits
- R = 1.5 Mbps
- delay = 15 sec

Packet Switching
Message Segmenting



Now break up the message into 5000 packets

- Each packet 1,500 bits
- 1 msec to transmit packet on one link
- *Pipelining:* each link works in parallel
- Delay reduced from 15 sec to 5.002 sec

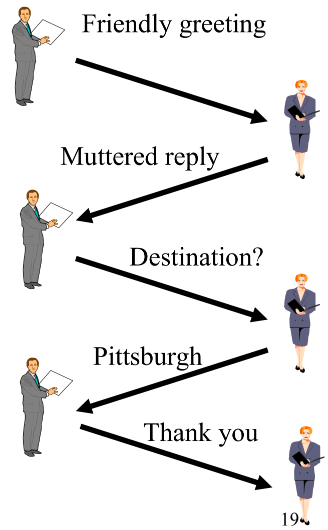
Packet-switched Networks
Forwarding

- *Goal:* Move packets through routers from source to destination
  - Path selection problem
    - Routing algorithms for datagram network:
  - *Destination address* in packet determines next hop
  - Routes may change during session
  - Analogy: asking directions while driving,
- Virtual circuit network:
  - Each packet carries tag (virtual circuit ID), tag determines next hop
  - Fixed path determined at *call setup time*, remains fixed thru call
  - *Routers maintain per-call state*

Network Architecture and Protocols
Design Principles

Protocols

- Module in layered structure
- An agreement between parties on how communication should take place
- Protocols define:
  - Interface to higher layers (API)
  - Interface to peer (syntax & semantics)
    - Actions taken on receipt of a messages
    - Format and order of messages
    - Error handling, termination, ordering of requests, etc.
- Example: Buying airline ticket

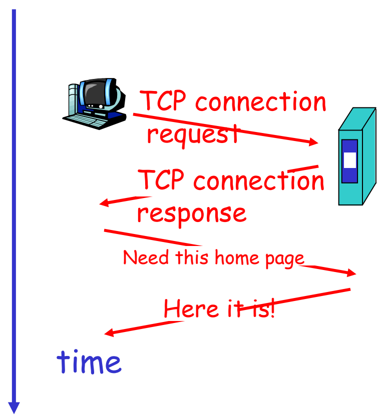


Network Protocols

- Machine, rather than human, oriented
- All communication activity in Internet governed by protocols

Protocols define format, order of messages sent and received among network entities, and actions taken on message transmission and/or reception

Network Protocols -- TCP



Protocol "Layers"

Networks are complex!

- many "pieces":
  - hosts
  - routers
  - links of various media
  - applications
  - protocols
  - hardware, software

Question:

Is there any hope of *organizing* structure of network?

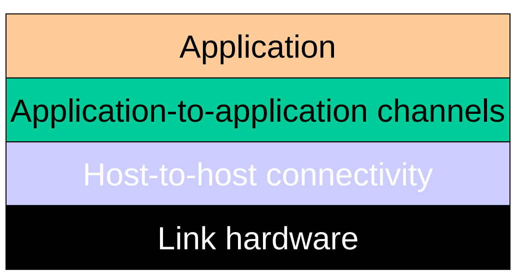
Or at least our discussion of networks?

Why layering?

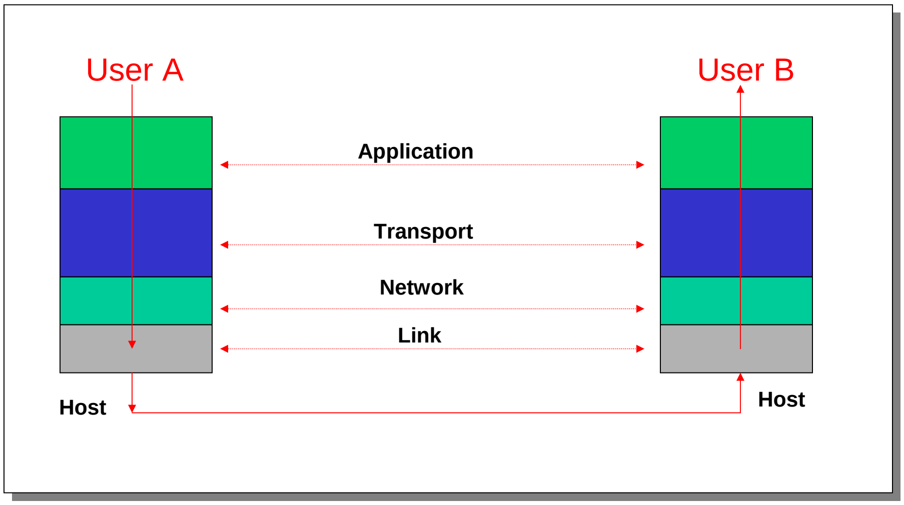
Dealing with complex systems:

- Explicit structure allows identification, relationship of complex system's pieces
  - Layered reference model for discussion
- Modularization eases maintenance, updating of system
  - Change of implementation of layer's service transparent to rest of system
  - For example, change in gate procedure doesn't affect rest of system

Layering Example



Peer Layer Communications



Layering Characteristics

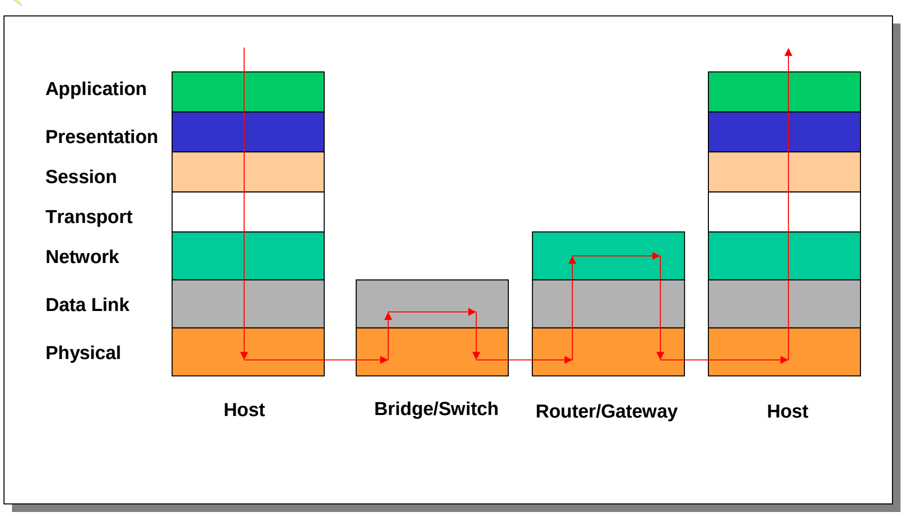
- Each layer relies on services from layer below and exports services to layer above
- Interface defines interaction
- Hides implementation - layers can change without disturbing other layers (black box)

OSI Reference Model

- Physical: how to transmit bits
- Data link: how to transmit frames
- Network: how to route packets
- Transport: how to send packets end2end
- Session: how to tie flows together
- Presentation: byte ordering, security
- Application: everything else

- TCP/IP has been amazingly successful, and it's not based on a rigid OSI model. The OSI model has been very successful at shaping thought

OSI Layers and Locations



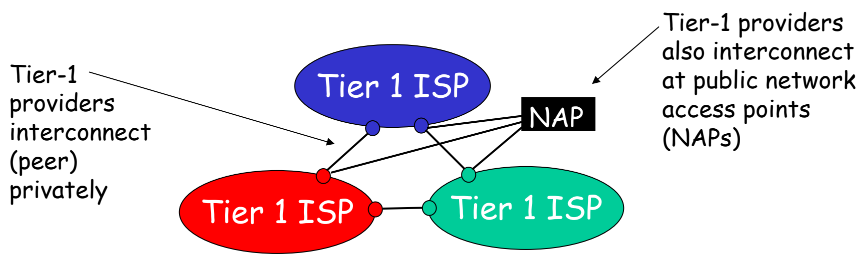
Is Layering Harmful?

- Layer N may duplicate lower level functionality (e.g., error recovery)
- Layers may need same info (timestamp, MTU)
- Strict adherence to layering may hurt performance
- Some layers are not always cleanly separated.
  - Inter-layer dependencies in implementations for performance reasons
  - Some dependencies in the standards (header checksums)
- Interfaces are not really standardized.
  - It would be hard to mix and match layers from independent implementations, e.g., windows network apps on unix (w/out compatibility library)
  - Many cross-layer assumptions, e.g. buffer management

Internet Architecture and Protocols Design Principles

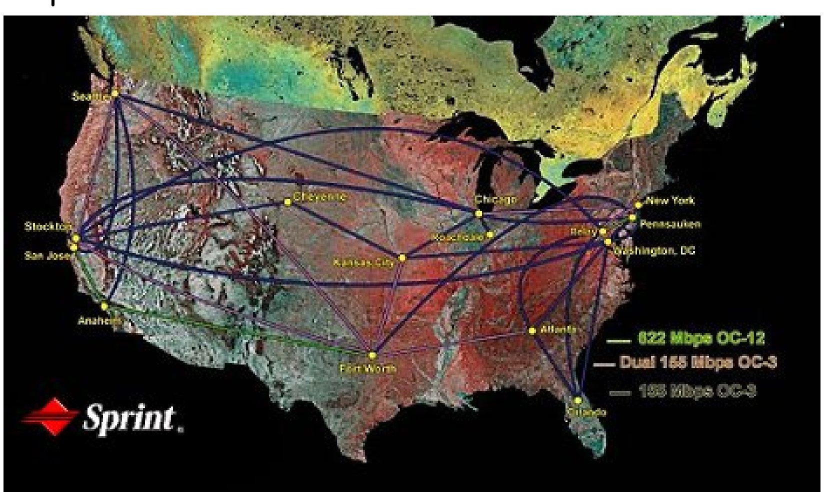
Internet structure: network of networks

- roughly hierarchical
- at center: "tier-1" ISPs (e.g., UUNet, BBN/Genuity, Sprint, AT&T), national/international coverage
  - treat each other as equals



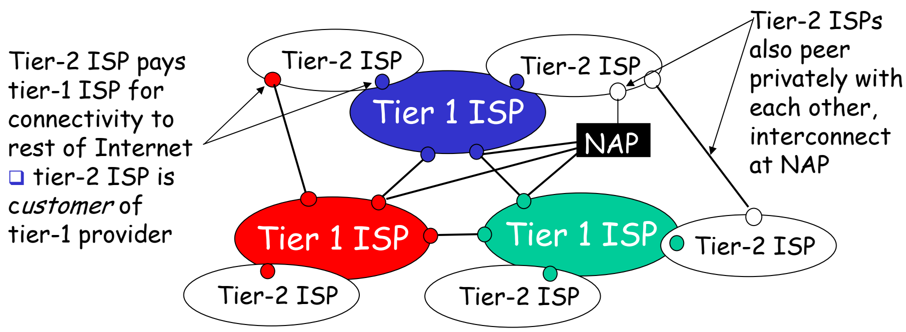
Tier-1 ISP: e.g., Sprint

Sprint US backbone network



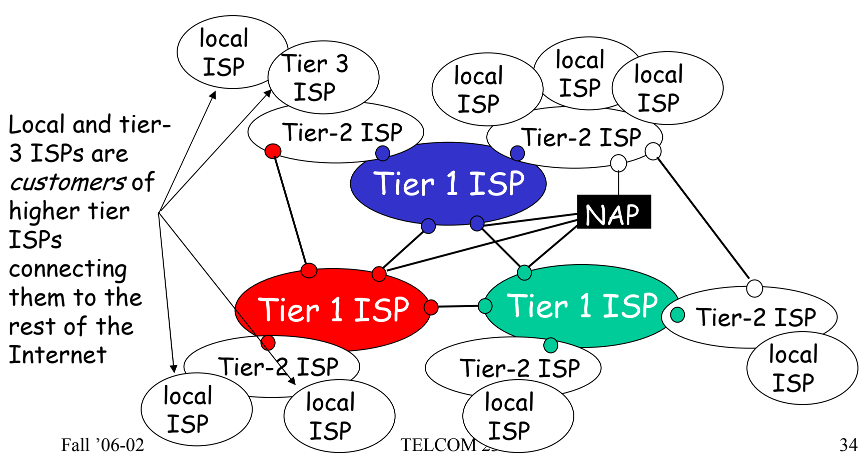
Internet Structure
A Network of Networks

- “Tier-2” ISPs: smaller (often regional) ISPs
  - Connect to one or more tier-1 ISPs, possibly other tier-2 ISPs



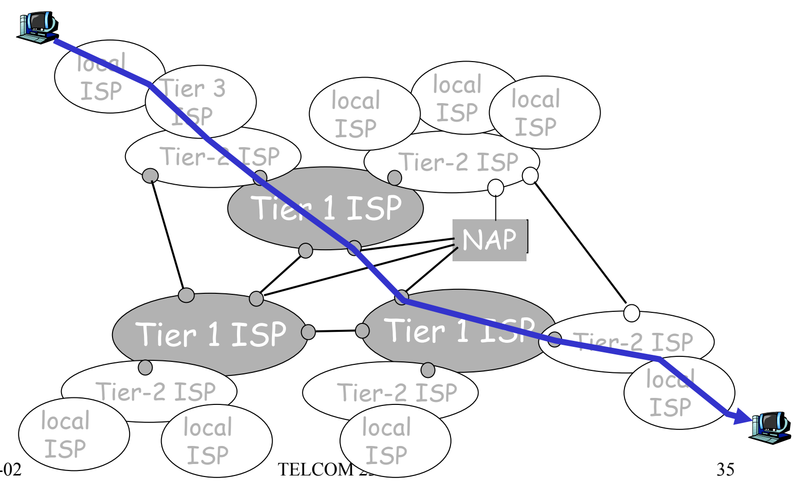
Internet Structure
A Network of Networks

- “Tier-3” ISPs and local ISPs
  - last hop (“access”) network (closest to end systems)



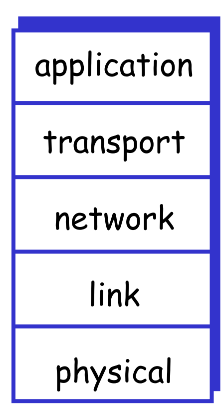
Internet structure: network of networks

- A packet passes through many networks!



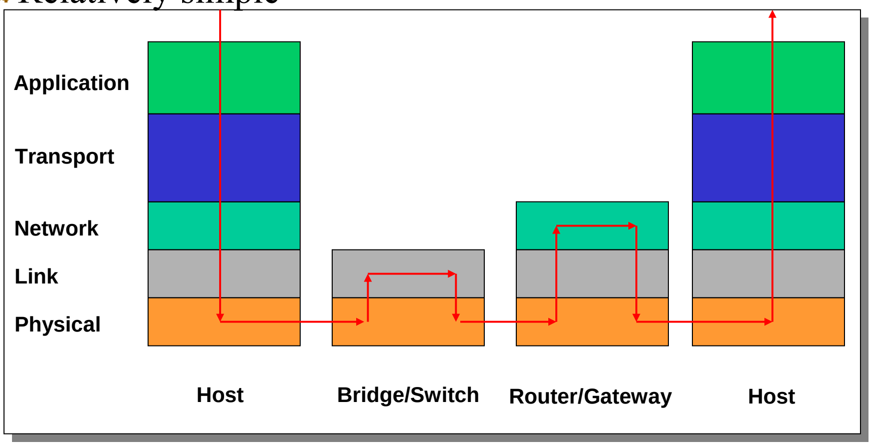
Internet Protocol Stack

- Application: supporting network applications
  - FTP, SMTP, STTP
- Transport: host-host data transfer
  - TCP, UDP
- Network: routing of datagrams from source to destination
  - IP, routing protocols
- Link: data transfer between neighboring network elements
  - PPP, Ethernet
- Physical: bits "on the wire"

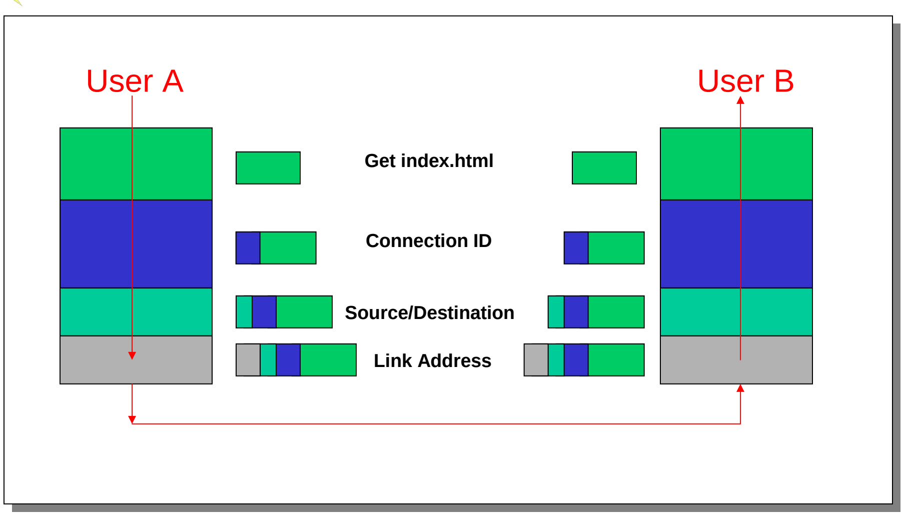


IP Layering

- Relatively simple



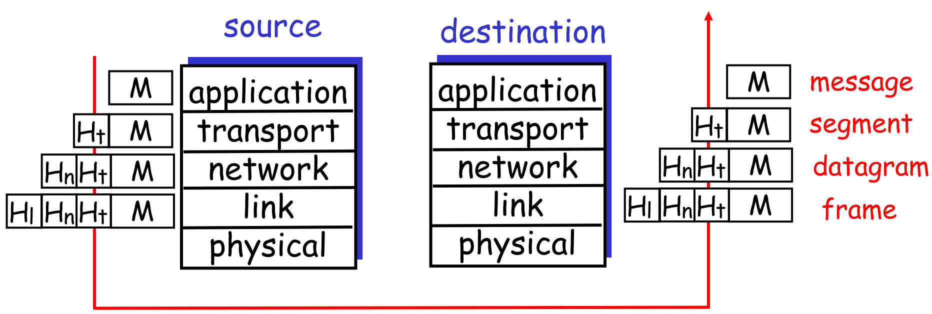
Layer Encapsulation



Protocol layering and data

Each layer takes data from above

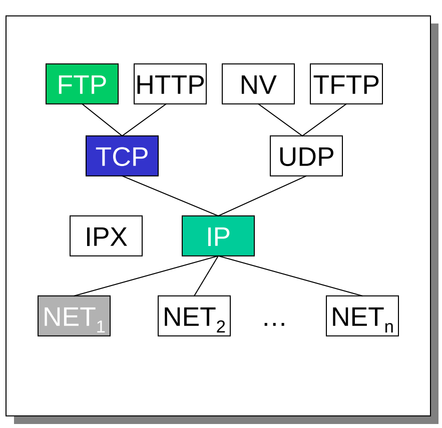
- adds header information to create new data unit
- passes new data unit to layer below

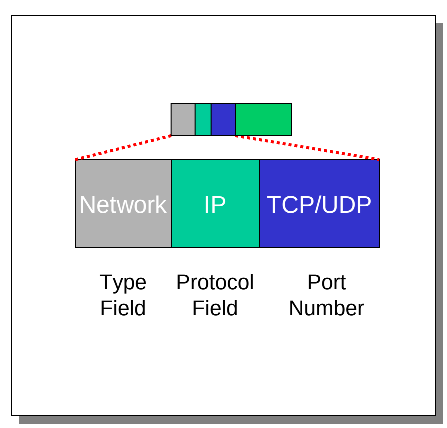


Fall '06-02 TELCOM 2310 39

Protocol Demultiplexing

- Multiple choices at each layer

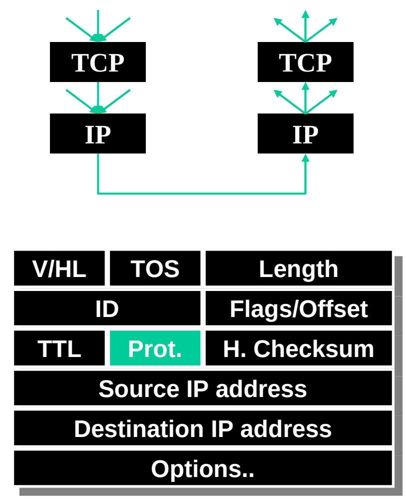




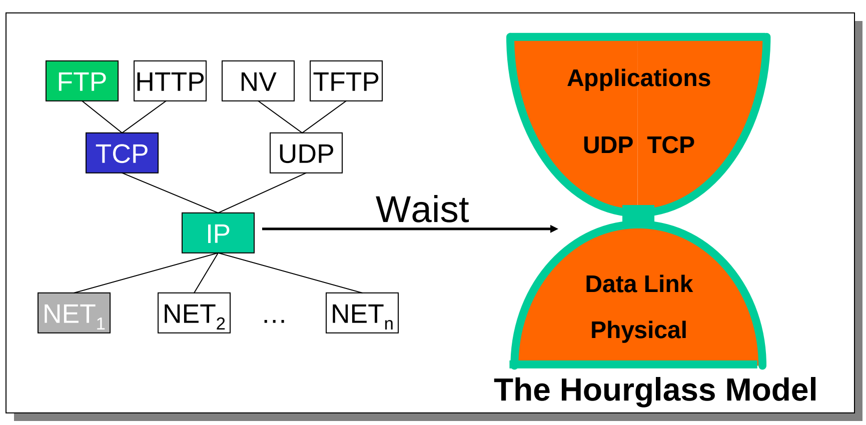
Fall '06-02 TELCOM 2310 40

Multiplexing and Demultiplexing

- There may be multiple implementations of each layer.
  - How does the receiver know what version of a layer to use?
- Each header includes a demultiplexing field that is used to identify the next layer.
  - Filled in by the sender
  - Used by the receiver
- Multiplexing occurs at multiple layers. E.g., IP, TCP, …



The Internet Protocol Suite

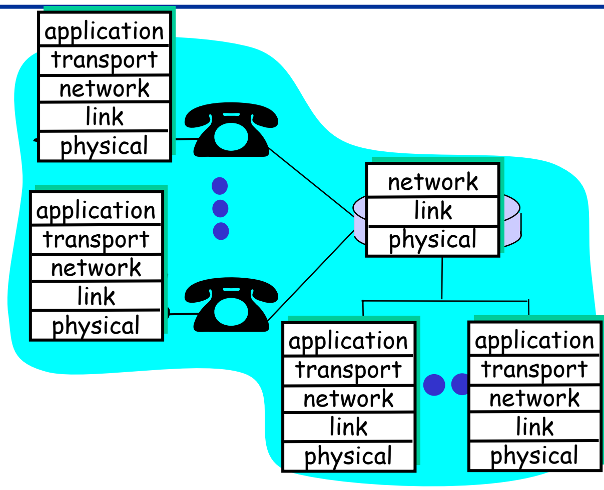


The waist facilitates interoperability

Logical Communication

Each layer:

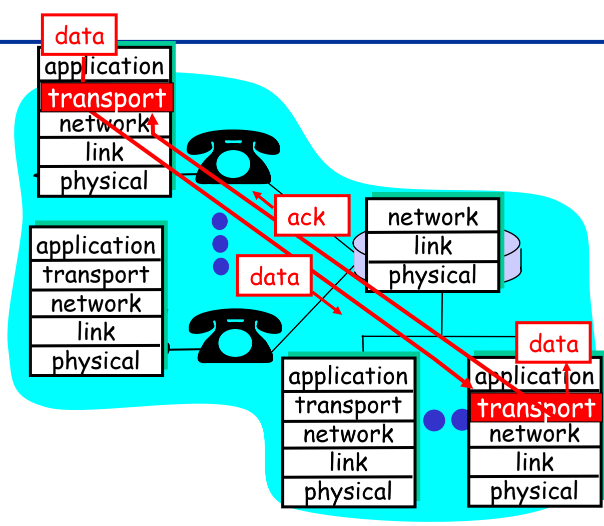
- Distributed "entities" implement layer functions at each node
- Entities perform actions, exchange messages with peers



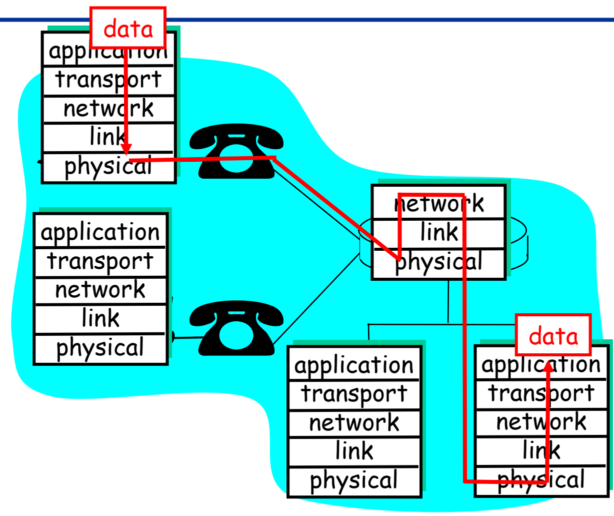
Layering: *logical* communication

E.g.: Transport Layer

- Take data from apps
- Add addressing, reliability check info to form "datagram"
- Send datagram to peer
- Wait for peer to acknowledge receipt



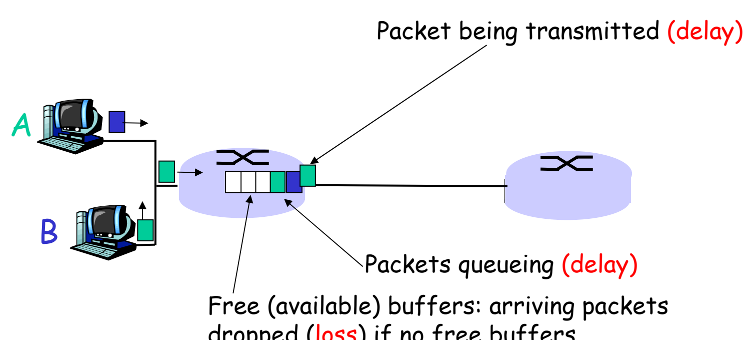
Physical communication



How Do loss and Delay Occur?

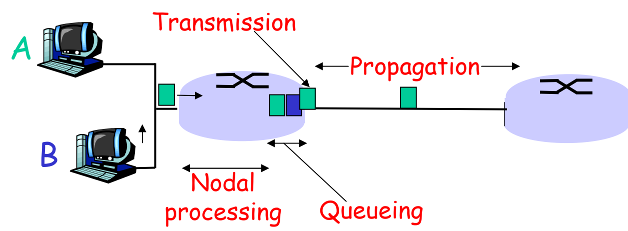
Packets *queue* in router buffers

- Packet arrival rate to link exceeds output link capacity
- Packets queue, wait for turn



Four sources of packet delay

- 1. Nodal processing:
  - Check bit errors
  - Determine output link
- 2. Queueing
  - Time waiting at output link for transmission
  - Depends on congestion level of router



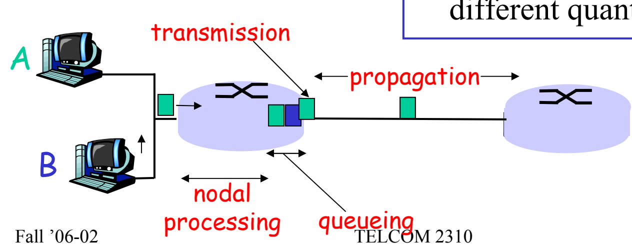
Fall ’06-02 TELCOM 2310 47

Delay in packet-switched networks

3. Transmission delay:
- R=link bandwidth (bps)
- L=packet length (bits)
- time to send bits into link = L/R

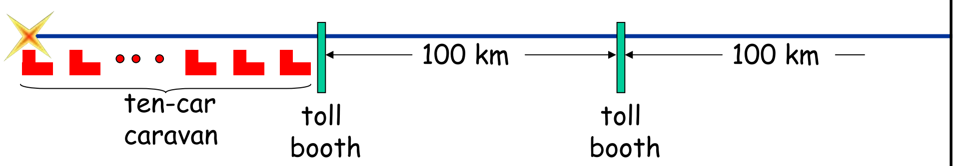
4. Propagation delay:
- d = length of physical link
- s = propagation speed in medium (~2×10^8 m/sec)
- propagation delay = d/s

Note: s and R are *very* different quantities!



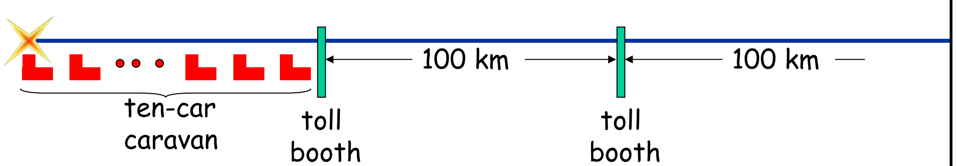
Fall ’06-02 TELCOM 2310 48

Caravan analogy



- Cars "propagate" at 100 km/hr
- Toll booth takes 12 sec to service a car (transmission time)
- car~bit; caravan ~ packet
- Q: How long until caravan is lined up before 2nd toll booth?
- Time to "push" entire caravan through toll booth onto highway = 12*10 = 120 sec
- Time for last car to propagate from 1st to 2nd toll both: 100km/(100km/hr)= 1 hr
- A: 62 minutes

Caravan analogy (more)



- Cars now "propagate" at 1000 km/hr
- Toll booth now takes 1 min to service a car
- Q: Will cars arrive to 2nd booth before all cars serviced at 1st booth?
- Yes! After 7 min, 1st car at 2nd booth and 3 cars still at 1st booth.
- 1st bit of packet can arrive at 2nd router before packet is fully transmitted at 1st router!
  - See Ethernet applet at AWL Web site

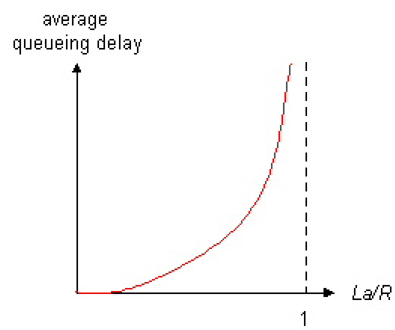
Nodal delay

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

- d_{proc} = processing delay
  - typically a few microsecs or less
- d_{queue} = queuing delay
  - depends on congestion
- d_{trans} = transmission delay
  - = L/R, significant for low-speed links
- d_{prop} = propagation delay
  - a few microsecs to hundreds of msecs

Queueing delay (revisited)

- R=link bandwidth (bps)
- L=packet length (bits)
- a=average packet arrival rate

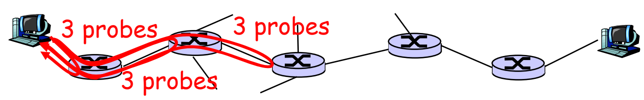


traffic intensity = La/R

- La/R ~ 0: average queueing delay small
- La/R -> 1: delays become large
- La/R > 1: more "work" arriving than can be serviced, average delay infinite!

"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
  - router *i* will return packets to sender
  - sender times interval between transmission and reply.



"Real" Internet delays and routes

traceroute: gaia.cs.umass.edu to www.eurecom.fr

Three delay measements from gaia.cs.umass.edu to cs-gw.cs.umass.edu

```
1  cs-gw (128.119.240.254)  1 ms  1 ms  2 ms
2  border1-rt-fa5-1-0.gw.umass.edu (128.119.3.145)  1 ms  1 ms  2 ms
3  cht-vbns.gw.umass.edu (128.119.3.130)  6 ms 5 ms 5 ms
4  jn1-at1-0-0-19.wor.vbns.net (204.147.132.129)  16 ms 11 ms 13 ms
5  jn1-so7-0-0-0.wae.vbns.net (204.147.136.136)  21 ms 18 ms 18 ms
6  abilene-vbns.abilene.ucaid.edu (198.32.11.9)  22 ms  18 ms  22 ms
7  nycm-wash.abilene.ucaid.edu (198.32.8.46)  22 ms  22 ms  22 ms
8  62.40.103.253 (62.40.103.253)  104 ms 109 ms 106 ms
9  de2-1.de1.de.geant.net (62.40.96.129)  109 ms 102 ms 104 ms
10  de.fr1.fr.geant.net (62.40.96.50)  113 ms 121 ms 114 ms
11  renater-gw.fr1.fr.geant.net (62.40.103.54)  112 ms  114 ms  112 ms
12  nio-n2.cssi.renater.fr (193.51.206.13)  111 ms  114 ms  116 ms
13  nice.cssi.renater.fr (195.220.98.102)  123 ms  125 ms  124 ms
14  r3t2-nice.cssi.renater.fr (195.220.98.110)  126 ms  126 ms  124 ms
15  eurecom-valbonne.r3t2.ft.net (193.48.50.54)  135 ms  128 ms  133 ms
16  194.214.211.25 (194.214.211.25)  126 ms  128 ms  126 ms
17  * * *
18  * * *
19  fantasia.eurecom.fr (193.55.113.142)  132 ms  128 ms  136 ms
```

trans-oceanic link (between hops 7 and 8)

* means no reponse (probe lost, router not replying)

Packet loss

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

Conclusion

- Architecture Design Issues and Protocols
  - Network edge and core
  - Layering and Protocols
- OSI Layered Architecture
- Internet Architecture and Design Principles
  - Internet Structure
  - User, networking and service views
  - Delay and Loss in Packet Switching