CS 54001-1: Large-Scale Networked Systems

Professor: Ian Foster

TA: Xuehai Zhang

Winter Quarter

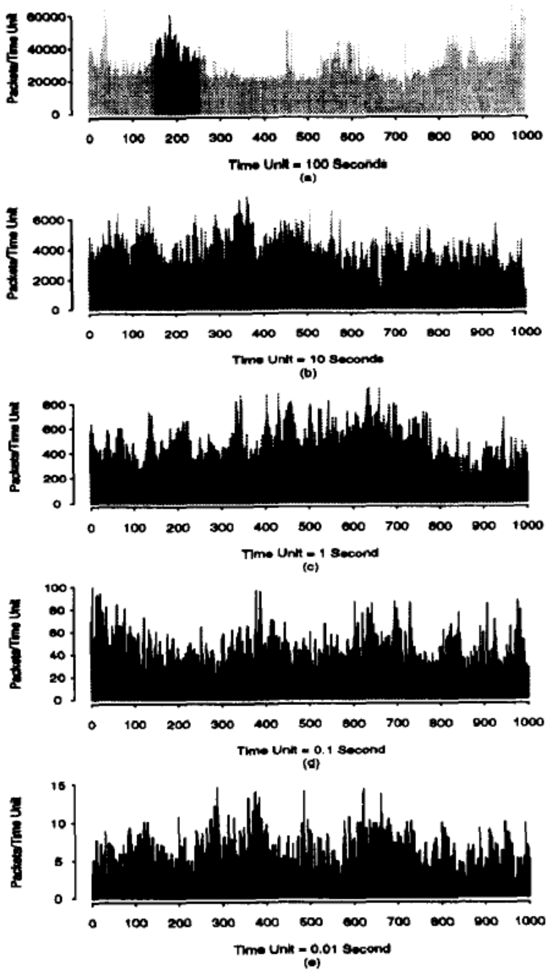
www.classes.cs.uchicago.edu/classes/archive/2003/winter/54001-1

Overview

- Introductions
- What is a network?
- Course format and content
- Internet design principles and protocols

What is a Network?

A Collection of Nodes and Links with Interesting Properties



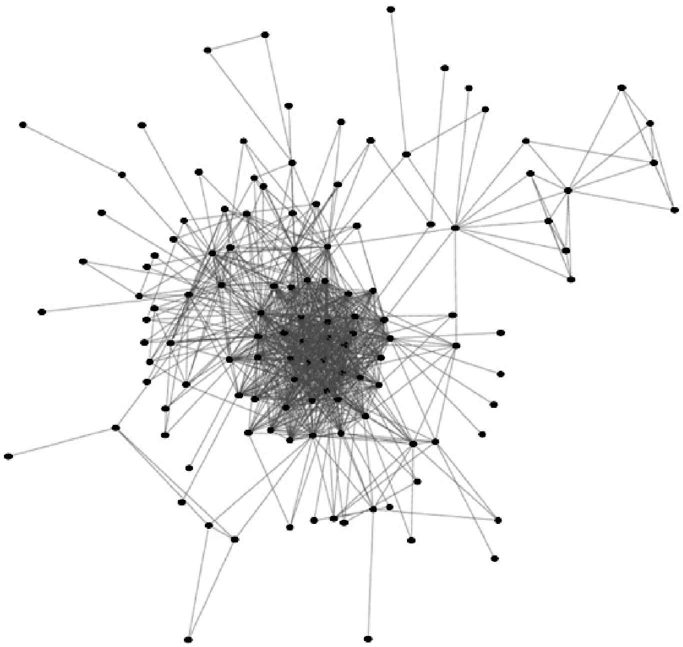
Time Unit = 100 Seconds
(a)

Time Unit = 10 Seconds
(b)

Time Unit = 1 Second
(c)

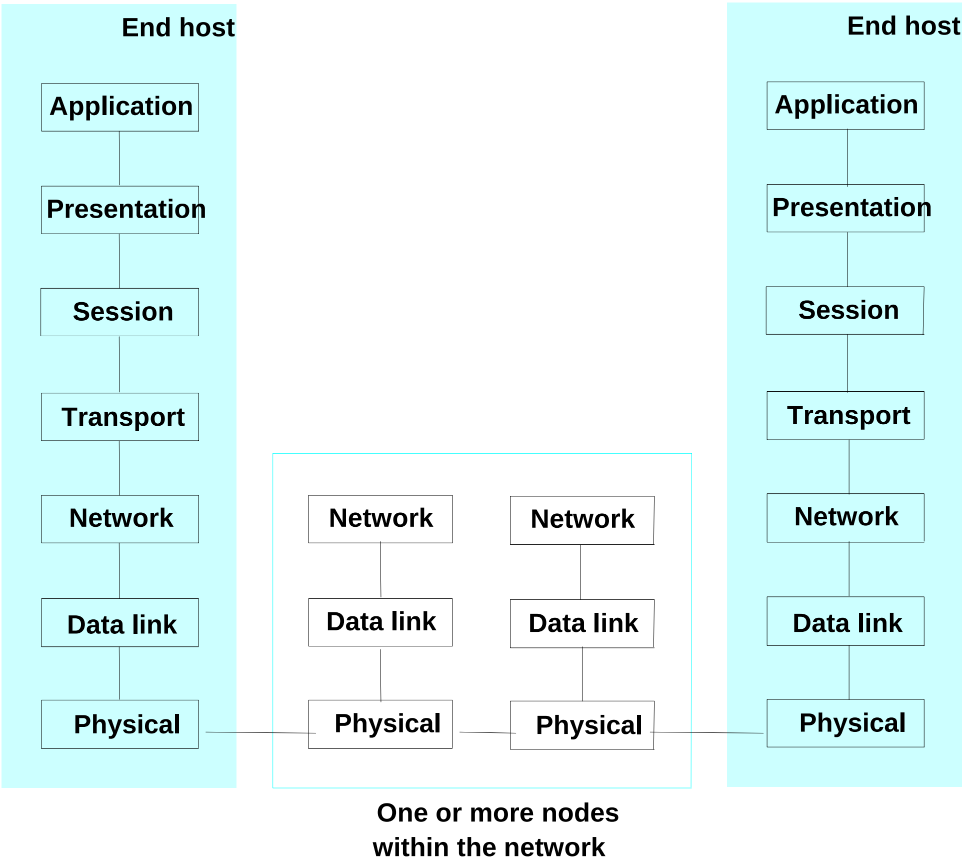
Time Unit = 0.1 Second
(d)

Time Unit = 0.01 Second
(e)



Time	Whole graph		Largest Connected component				Random Graph	
Interval	# Nodes	# Links	# Nodes	# Links	Clustering	Path length	Clustering	Path length
1 day	20	38	12	34	0.827	1.61	0.236	2.39
2 days	20	77	15	75	0.859	1.29	0.333	1.68
7 days	63	331	58	327	0.816	2.21	0.097	2.35
14 days	87	561	81	546	0.777	2.56	0.083	2.3
30 days	128	1046	126	1045	0.794	2.45	0.067	2.29

Communication Protocols



End host

Application

Presentation

Session

Transport

Network

Data link

Physical

Network

Data link

Physical

Network

Data link

Physical

End host

Application

Presentation

Session

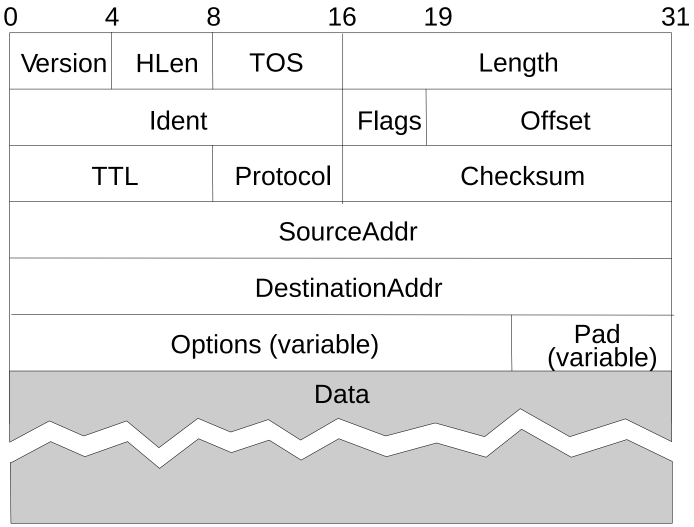
Transport

Network

Data link

Physical

One or more nodes within the network



0 4 8 16 19 31

Version HLen TOS Length

Ident Flags Offset

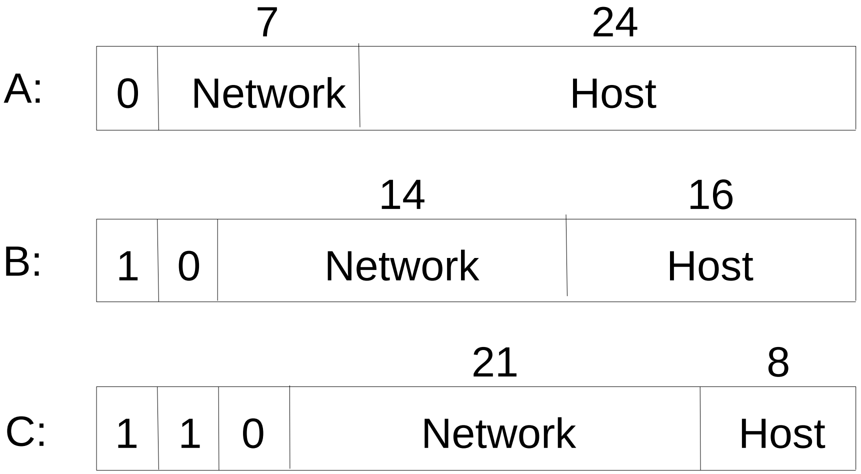
TTL Protocol Checksum

SourceAddr

DestinationAddr

Options (variable) Pad (variable)

Data

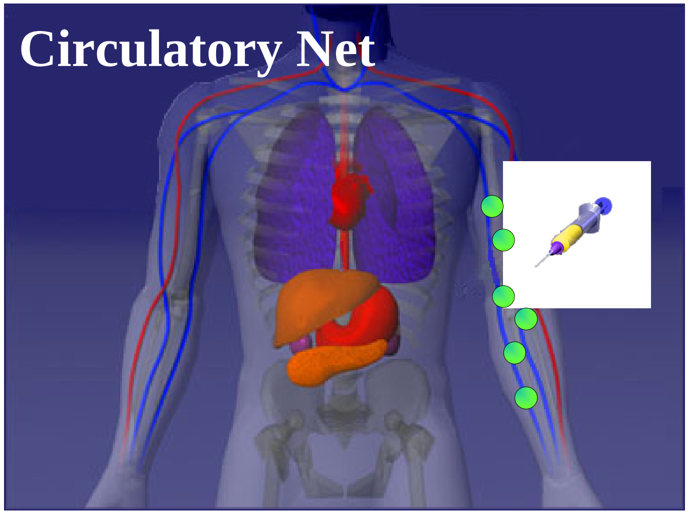


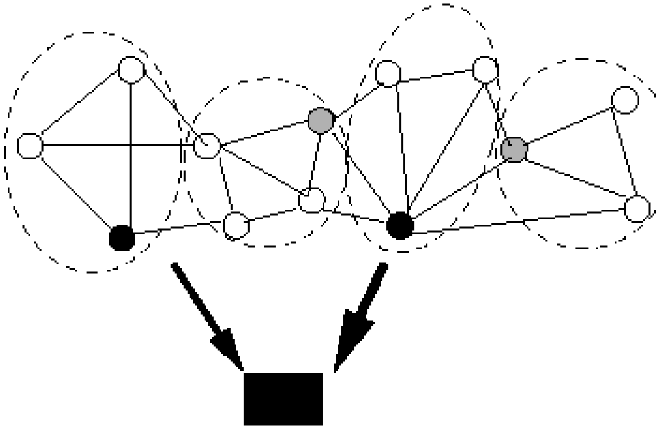
A: 0 | Network (7) | Host (24)

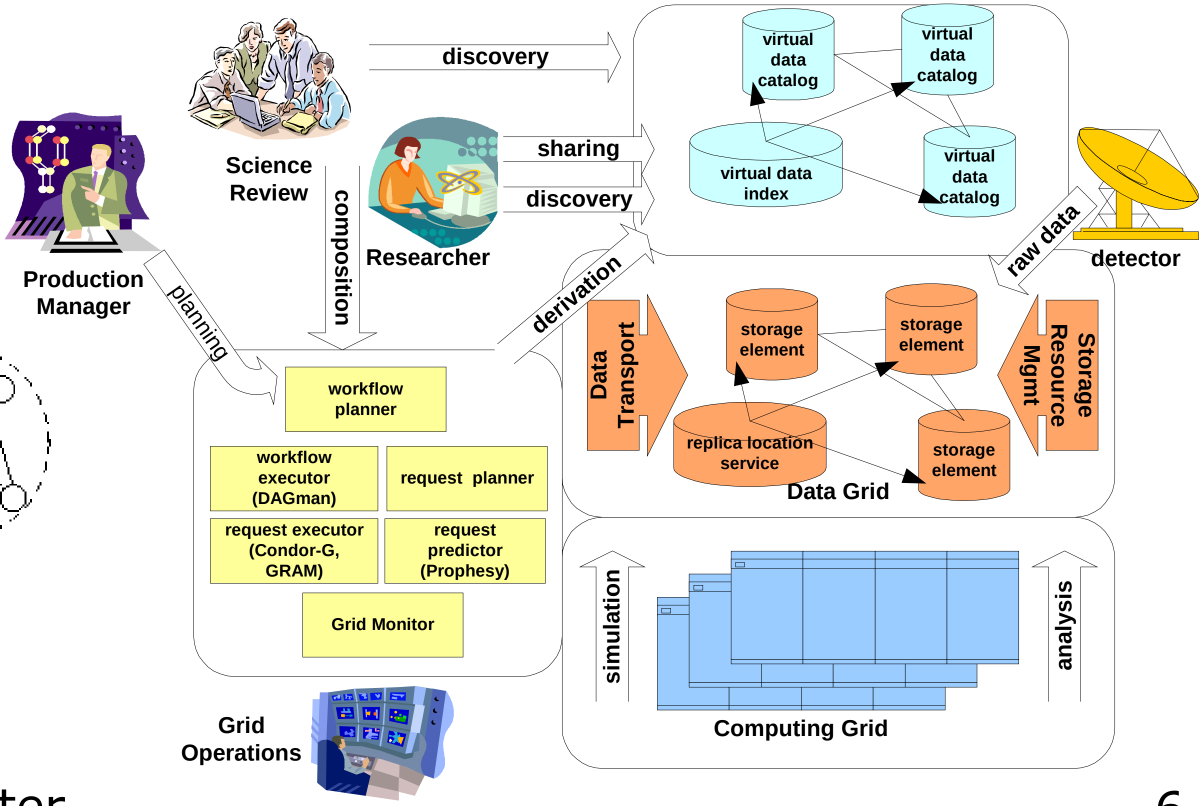
B: 1 0 | Network (14) | Host (16)

C: 1 1 0 | Network (21) | Host (8)

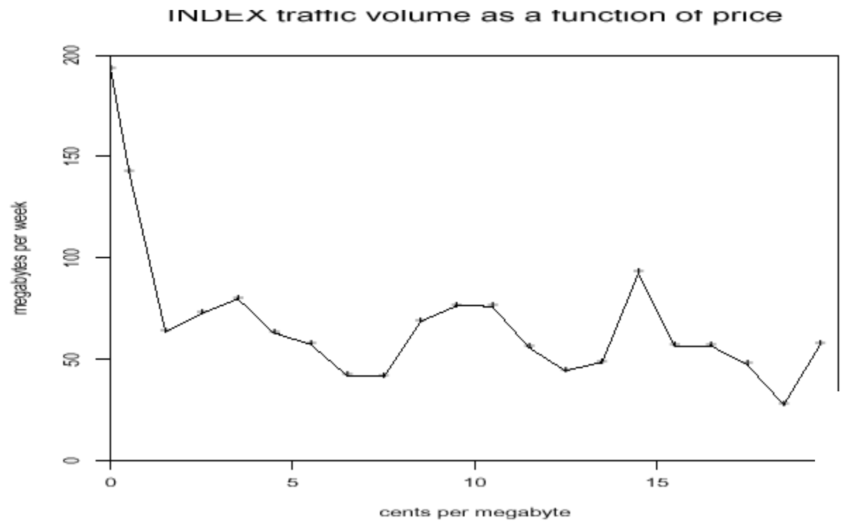
Applications

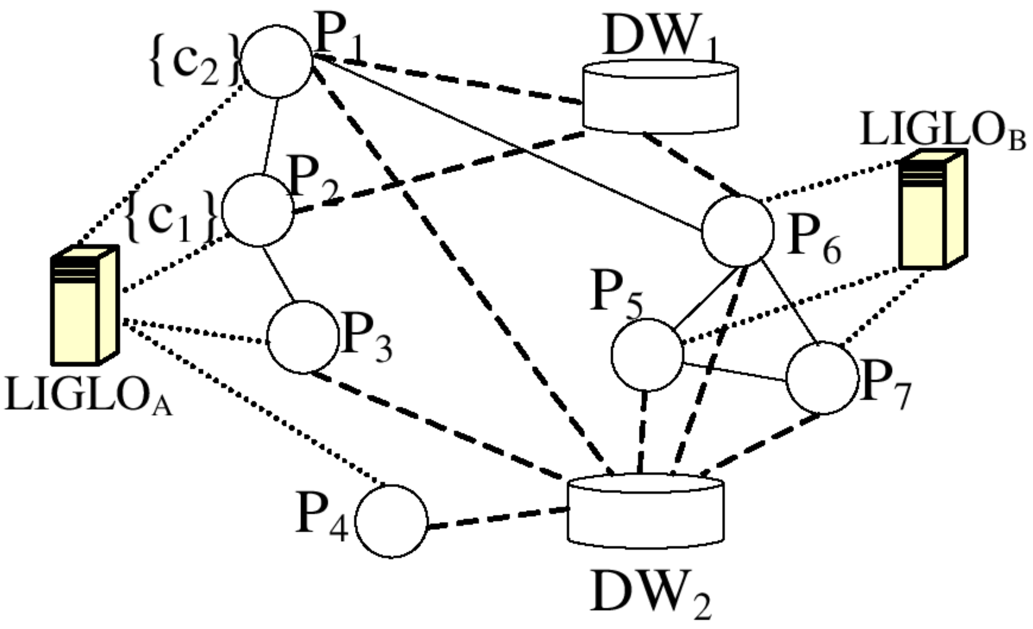






Organizational Structures





What is a “Network”?

- A collection of nodes and links with interesting emergent properties
  - Internet, Gnutella, citations, disease, …
- A collection of devices that use some protocol to communicate
  - Internet protocols: TCP/IP and friends
- Applications enabled by existence of those basic protocols
  - Web, Grid, Napster, …
- Organizational structures that allow such systems to function
  - Security, management, policy, …

CS 54001-1 Course Goals

- Yes
  - Gain understanding of fundamental issues that effect design, construction, and operation of large-scale networked systems
  - Gain understanding of some significant future trends in network design and use
- No
  - Learn how to write network applications

Course Outline (Subject to Change)

1. (January 9th) Internet design principles and protocols
2. (January 16th) Internetworking, transport, routing
3. (January 23rd) Mapping the Internet and other networks
4. (January 30th) Security (with guest lecturer: Gene Spafford)
5. (February 6th) P2P technologies & applications (Matei Ripeanu)

   (plus midterm)
6. (February 13th) Optical networks (Charlie Catlett)
7. *(February 20th) Web and Grid Services (Steve Tuecke)
8. (February 27th) Advanced applications (with guest lecturers: Terry Disz, Mike Wilde)
9. *(March 6th) Network operations (Greg Jackson)
10. (March 13th) Final exam

* Ian Foster is out of town.

Approach

- Prior to each lecture, I will assign reading:
  - Chapters from *Computer Networks: A Systems Approach*, 2nd Edition, Larry Peterson and Bruce Davie, Morgan Kaugrman, 1999.
  - Other sources.
- I'll also assign exercises of various sorts, for which answers will be provided later
- Evaluation will be based on a midterm plus a final

Course Details

- Thursdays, 5:30-8:30 Ryerson 251
  - 9 weeks lectures, one final exam
  - Also midterm
- Evaluation
  - Attendance: 10%
  - Mid-term: 30%
  - Final: 60%

Policies

- Collaboration
  - We encourage you to discuss the course material with fellow students. However, submitted assignments must be your own work.
  - If you discuss in details specific problems or assignments with other people, you must acknowledge this on the front of the work that you turn in.

For More Information

- Contact me
  - Ian Foster, foster@cs.uchicago.edu
  - Email or set up a meeting
- Contact my trusty TA
  - Xuehai Zhang, hai@cs.uchicago.edu
- Monitor the class web page
  - www.classes.cs.uchicago.edu/classes/archive/2003/winter/54001-1
- Post questions to the mailing list
  - http://mailman.cs.uchicago.edu/mailman/listinfo/cspp54001

Can you please provide Xuehai with …

- Photo
- Name
- Educational background
- What courses you have taken in CSPP
- A few sentences on what you know about networks
- A few sentences on what you want to get out of this course

Internet Design Principles & Protocols

- An introduction to the mail system
- An introduction to the Internet
- Internet design principles and layering
- Brief history of the Internet
- Packet switching and circuit switching
- Protocols
- Addressing and routing
- Performance metrics
- A detailed FTP example

Sources of Overheads

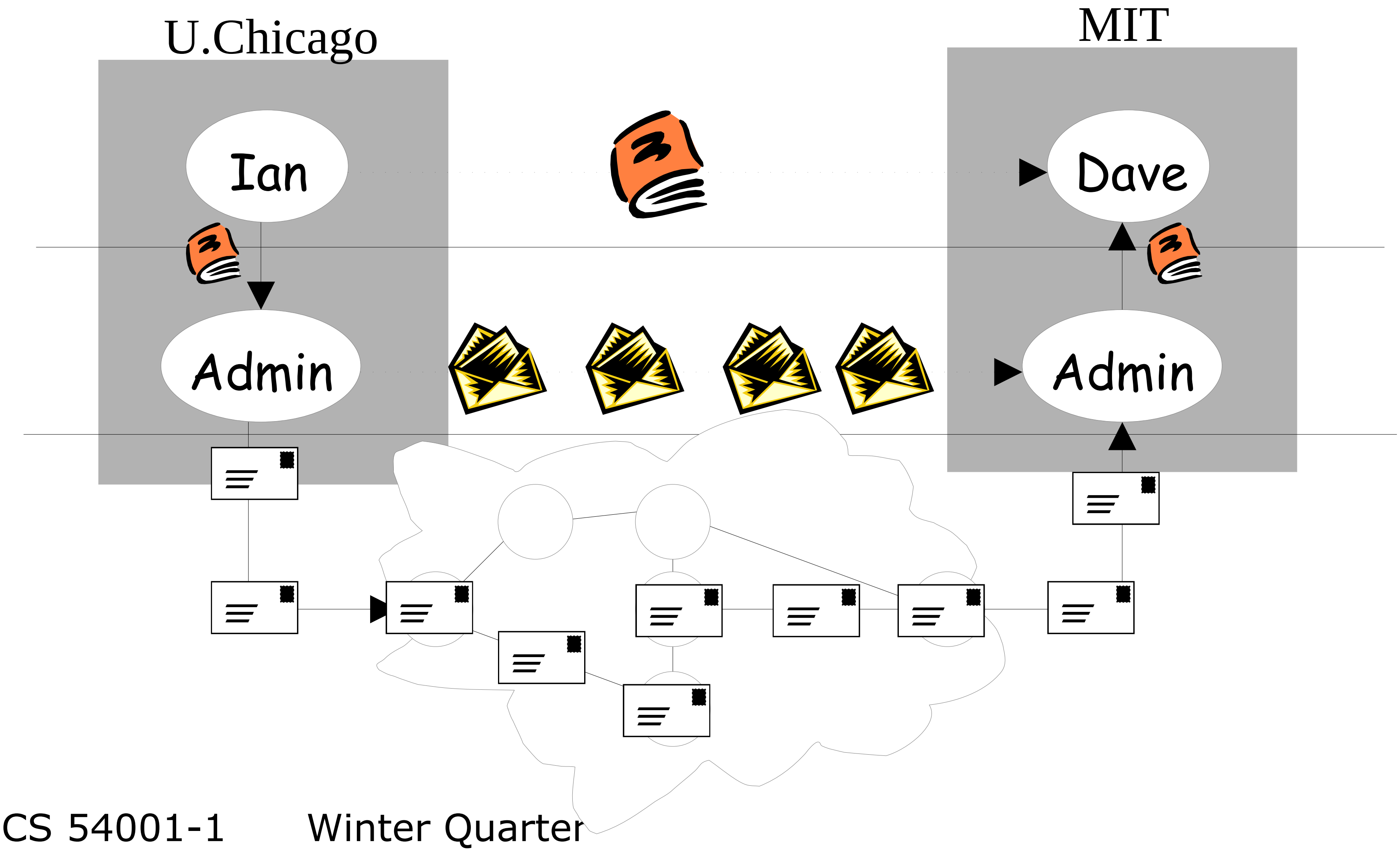
Gratefully Acknowledged!

http://www.stanford.edu/class/cs244a

http://www.cs.wisc.edu/~pb/cs640.html

http://walrandpc.eecs.berkeley.edu/122S03.html

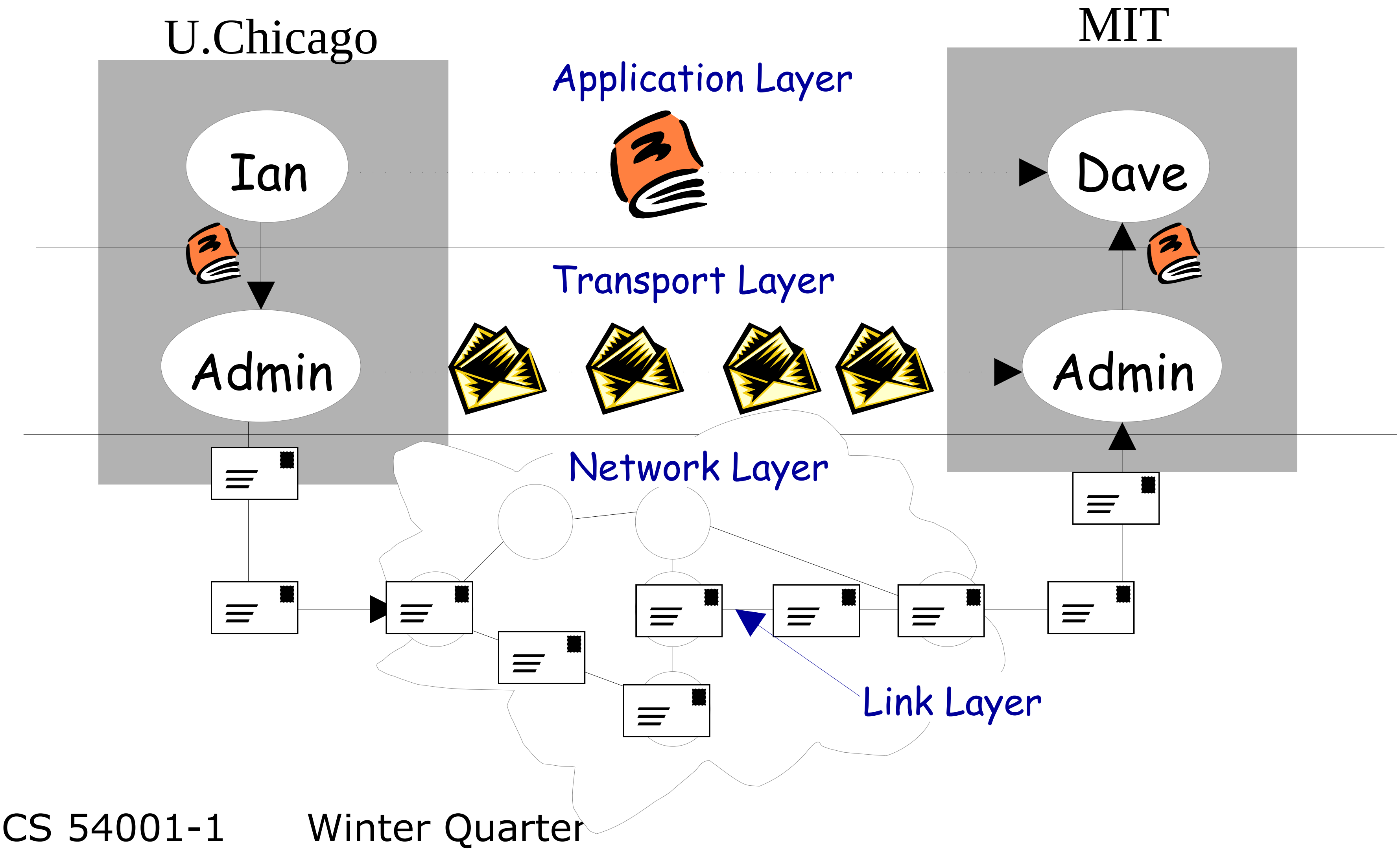
An Introduction to the mail system



Characteristics of the mail system

- Each envelope is individually routed
- No time guarantee for delivery
- No guarantee of delivery in sequence
- No guarantee of delivery at all!
  - Things get lost
  - How can we acknowledge delivery?
  - Retransmission
    - How to determine when to retransmit? Timeout?
    - Need local copies of contents of each envelope
    - How long to keep each copy
    - What if an acknowledgement is lost?

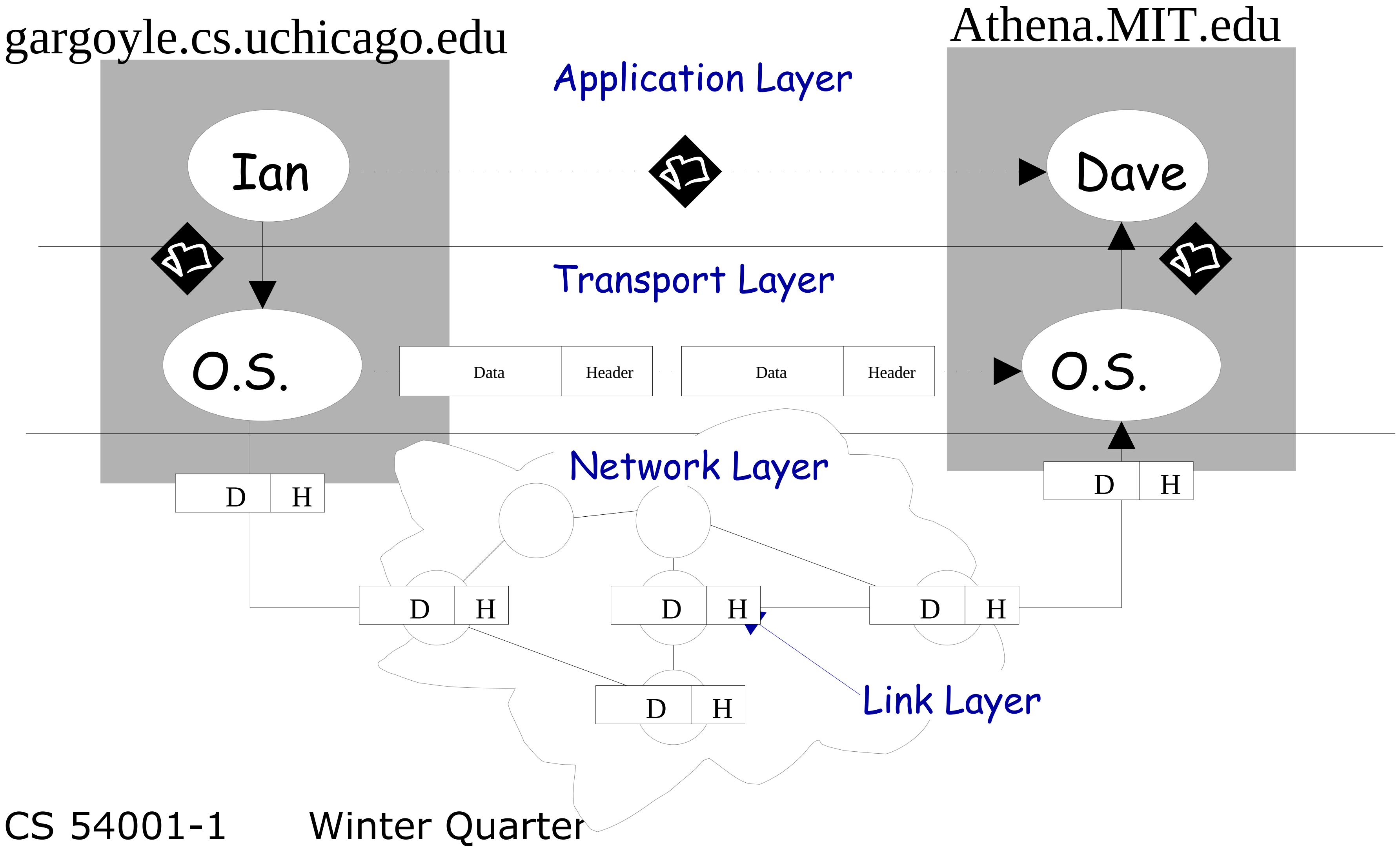
An Introduction to the Mail System



Internet Design Principles & Protocols

- An introduction to the mail system
- **An introduction to the Internet**
- Internet design principles and layering
- Brief history of the Internet
- Packet switching and circuit switching
- Protocols
- Addressing and routing
- Performance metrics
- A detailed FTP example

An Introduction to the Internet



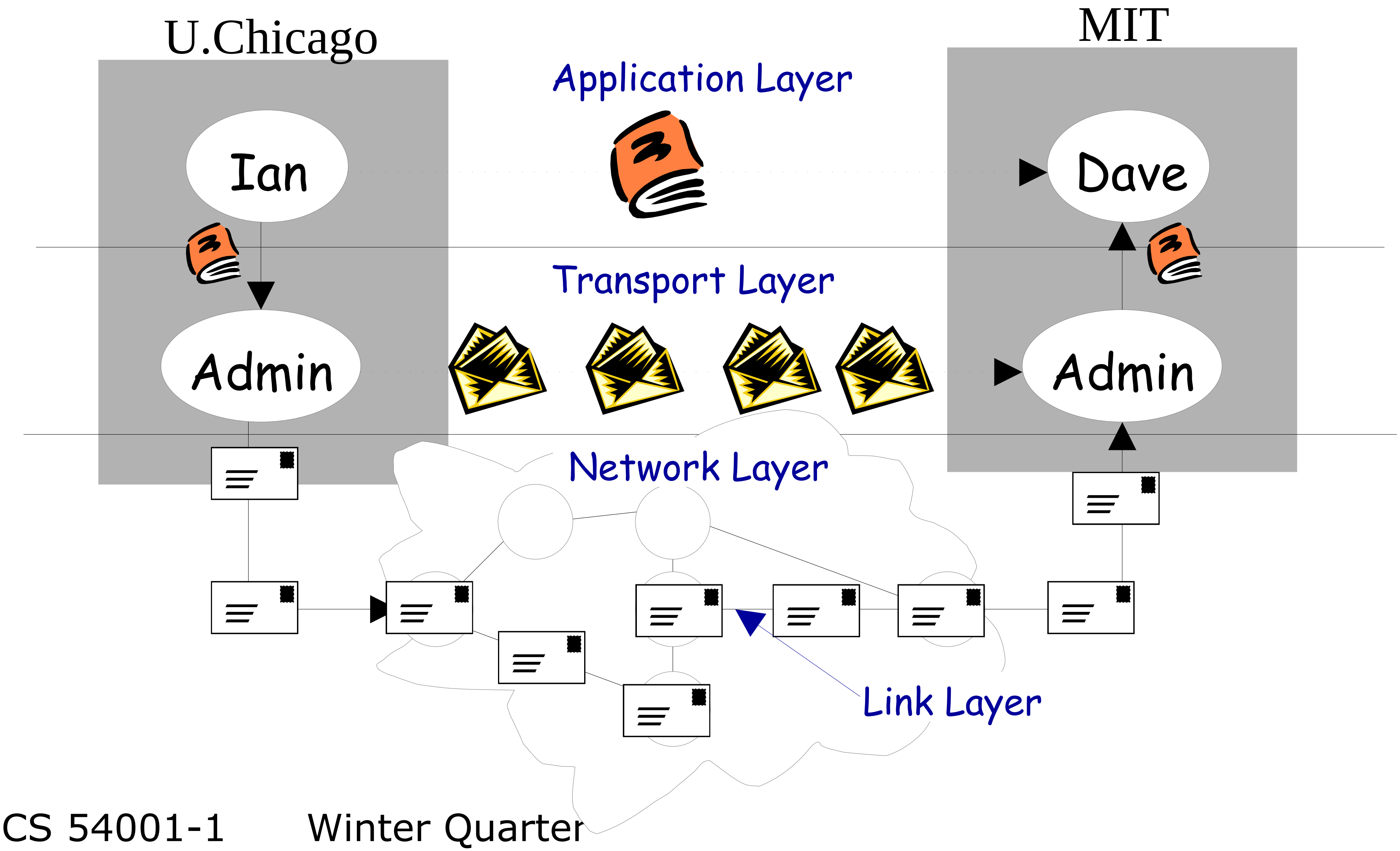
Characteristics of the Internet

- Each packet is individually routed
- No time guarantee for delivery
- No guarantee of delivery in sequence
- No guarantee of delivery at all!
  - Things get lost
  - Acknowledgements
  - Retransmission
    - How to determine when to retransmit? Timeout?
    - Need local copies of contents of each packet.
    - How long to keep each copy?
    - What if an acknowledgement is lost?

Characteristics of the Internet (2)

- No guarantee of integrity of data.
- Packets can be fragmented.
- Packets may be duplicated.

An Introduction to the Mail System



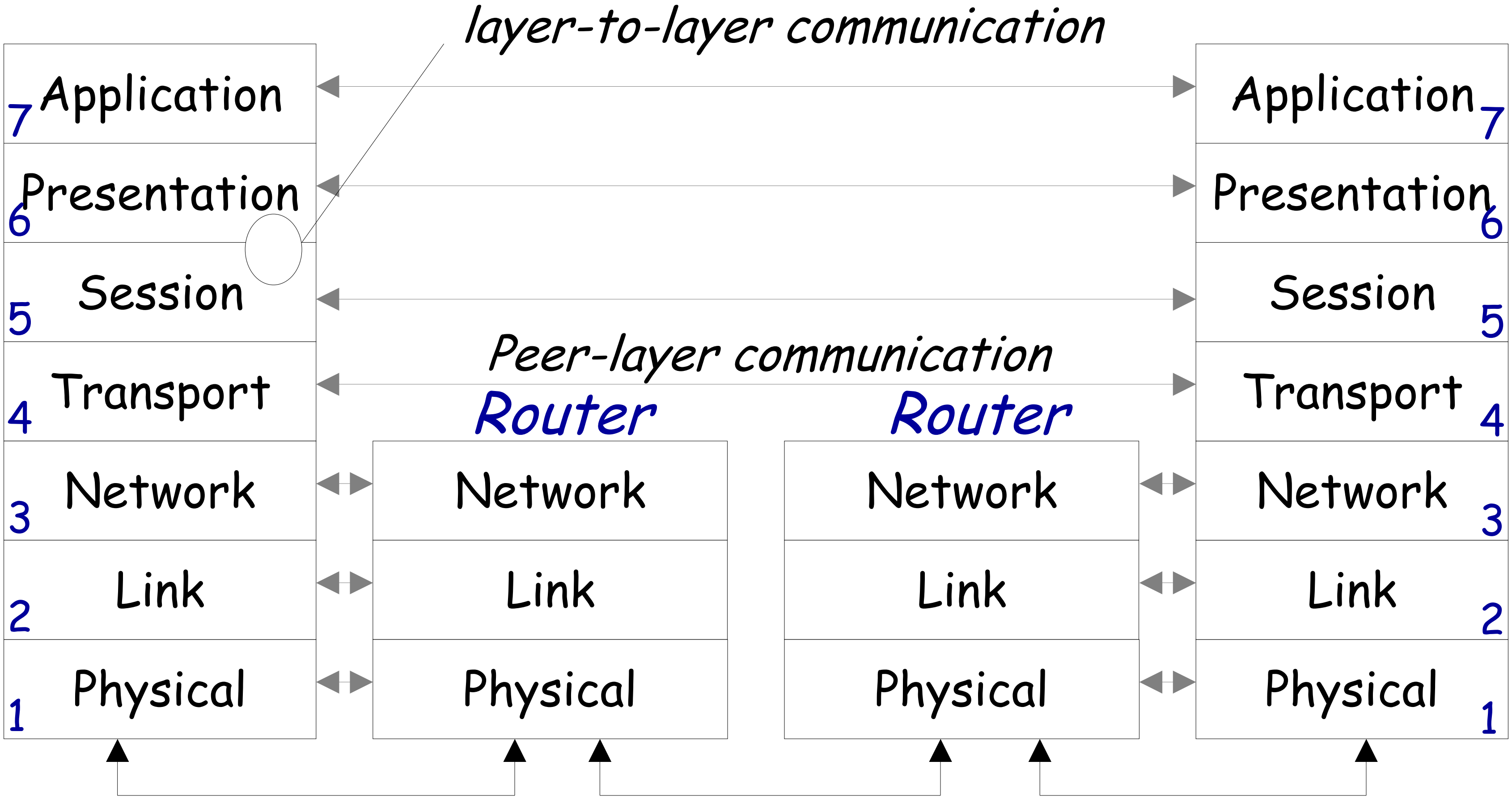
Some Questions about the Mail System

- How many sorting offices are needed and where should they be located?
- How much sorting capacity is needed?
  - Should we allocate for Mother’s Day?
- How can we guarantee timely delivery?
  - What prevents delay guarantees?
  - Or delay variation guarantees?
- How do we protect against fraudulent mail deliverers, or fraudulent senders?

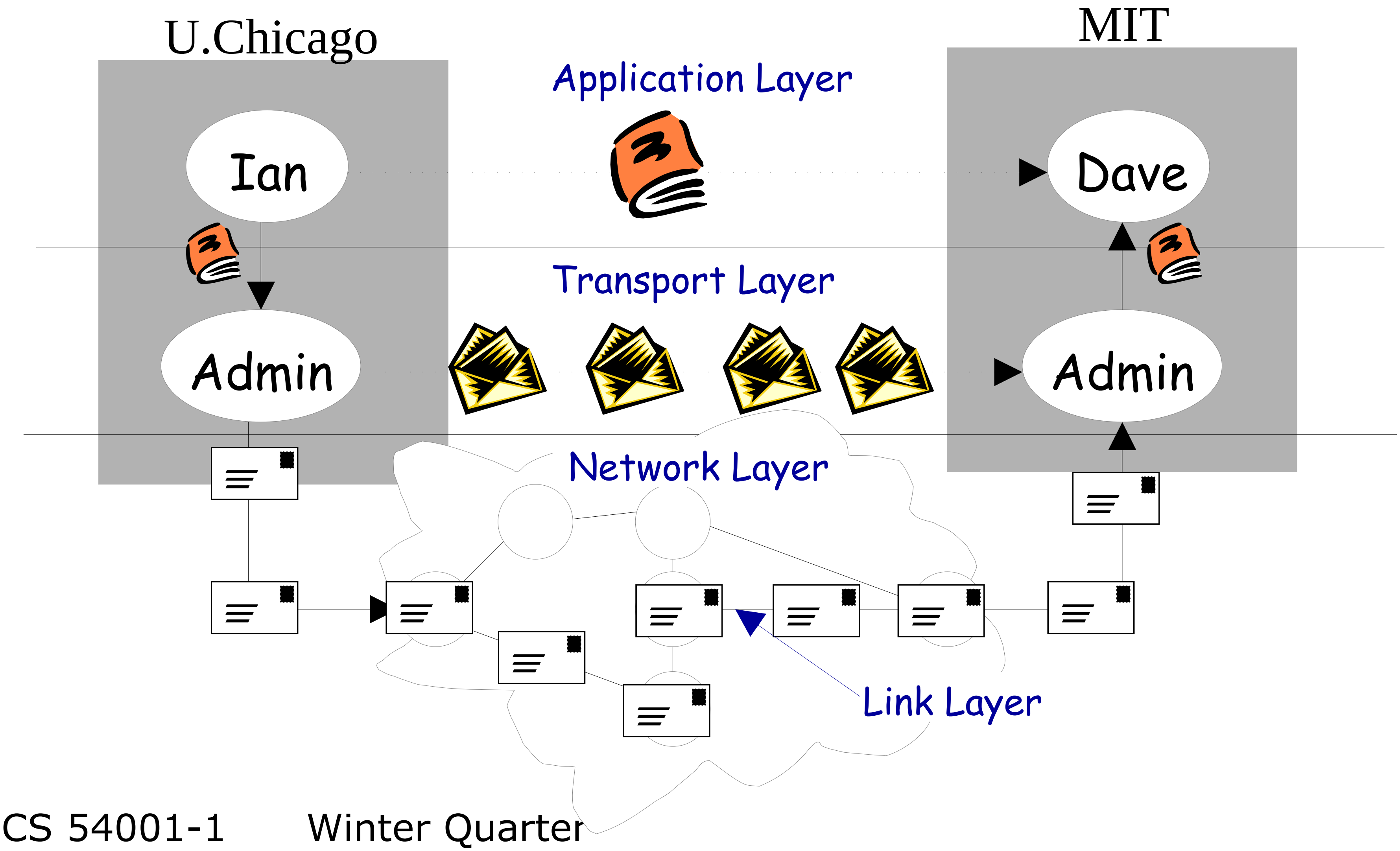
Internet Design Principles & Protocols

- An introduction to the mail system
- An introduction to the Internet
- **Internet design principles and layering**
- Brief history of the Internet
- Packet switching and circuit switching
- Protocols
- Addressing and routing
- Performance metrics
- A detailed FTP example

Layering: The OSI Model



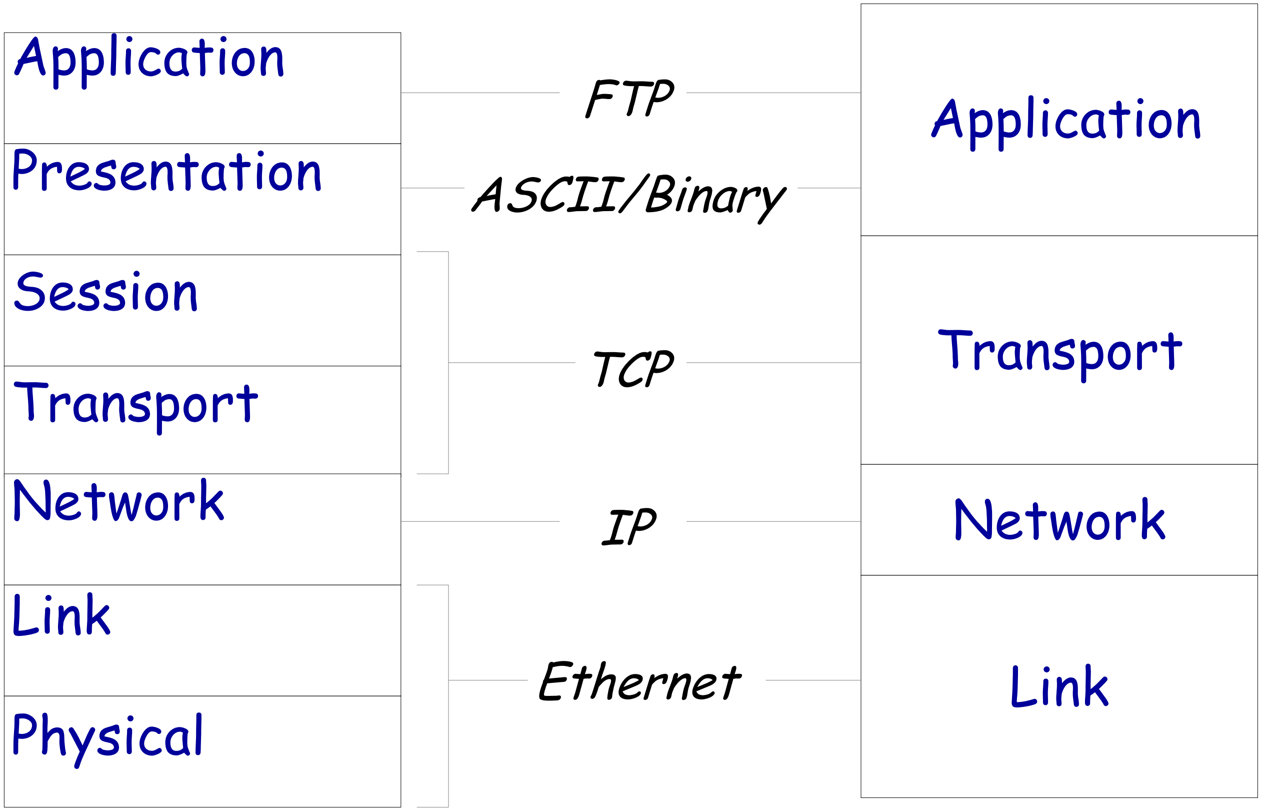
An Introduction to the Mail System



Layering in the Internet

- Transport Layer
  - Provides reliable, in-sequence delivery of data from end-to-end on behalf of application
- Network Layer
  - Provides “best-effort”, but unreliable, delivery of datagrams
- Link Layer
  - Carries data over (usually) point-to-point links between hosts and routers; or between routers and routers.

Layering: FTP



The 7-layer OSI Model

The 4-layer Internet model

Internet Architecture

- Defined by Internet Engineering Task Force (IETF)
  1. Application: interacts with user to initiate data transfers (e.g., browser, media player, command line)
  2. Transport: reliable, in-order delivery of data (TCP and UDP)
  3. Network: addressing and routing (IP)
  4. Data Link: defines how hosts access physical media (Ethernet)
  5. Physical: defines how bits are represented on wire (Manchester)
- Information is passed between layers via encapsulation
  - Header information is attached to data passed down layers
- Multiplexing between layers
- Layers access other layers via APIs (e.g., sockets)
- Communication at a specific layer is enabled by a protocol

Internet Design Goals

- **Scope**: support a wide range of approaches
- **Scalability**: work well with very large networks (encourages simplicity)
- **Robustness**: operate (well) under partial failures
- **Incremental deployment**: compatibility with existing system(s)

The End-to-End Argument

- See “End-To-End Arguments in System Design”
  - The function in question can completely and correctly be implemented only with the knowledge of the application standing at the endpoints of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)

For Example: File Transfer

- Goal: to transfer a file correctly between peers
- Method: break up file into messages, transfer messages
- Threats: network may drop, reorder, duplicate, or corrupt messages
- What if we have hop-by-hop reliability?
- Where must correct delivery be checked?

Placing Functionality: Encryption

- Which layer should encrypt data?
- Higher: data is in the clear in fewer places, keys are nearest the user, every application must encrypt
- Lower: more opportunity to intercept, how to provide key material, applications are simpler (don’t worry about crypto)
- User vs Administrator locus of control

Placing Functionality: Reliability

- Consider reliability… assume a link has probability p of losing a packet; (1-p) of not losing a packet
- Traversing n hops give (1-p)^n prob of delivery and 1- (1-p)^n prob of drop
- Assume “typical” Internet path of n = 15

Placing Functionality: Performance Impact

- For a low loss rate (p = 10^-5), e2e Prob(loss) = 1.5x10^-3=.0015 (<1%)
- But for a higher rate (p = .01, say, for wireless), Ploss = 1-(1-.01)15=0.14 !!
- Internet was designed with < 1% path loss in mind; unfortunately, some parts today have much higher rates (later)

Internet Design Principles & Protocols

- An introduction to the mail system
- An introduction to the Internet
- Internet design principles and layering
- **Brief history of the Internet**
- Packet switching and circuit switching
- Protocols
- Addressing and routing
- Performance metrics
- A detailed FTP example

A Brief History of Networking: early years

- Roots traced to public telephone network of the 60s
  - How can computers be connected together?
- Three groups were working on packet switching as an efficient alternative to circuit switching
- L. Kleinrock had first published work in 1961
  - Showed packet switching was effective for bursty traffic
- P. Baran had been developing packet switching at Rand Institute and plan was published in 1967
  - Basis for ARPAnet
- First contract to build network switches awarded to BBN
- First network had four nodes in 1969

History of the Internet contd.

- By 1972, network had grown to 15 nodes
  - Network Control Protocol: first end-to-end protocol (RFC001)
  - Email was first app: R. Tomlinson, 1972
- In 1973, R. Metcalfe invented Ethernet
- In 1974, V. Cerf and R. Kahn developed open architecture for Internet
  - TCP and IP

History of the Internet contd.

- By 79 the Internet had grown to 200 nodes and by the end of 89 to over 100K
  - Much growth fueled by connecting universities
- Major developments
  - TCP/IP as standard; DNS
- 89: V. Jacobson made major improvements to TCP
- 91: T. Berners-Lee invented the Web
- 93: M. Andreesen invented Mosaic
- The rest should be pretty familiar…

Internet Design Principles & Protocols

- An introduction to the mail system
- An introduction to the Internet
- Internet design principles and layering
- Brief history of the Internet
- **Packet switching and circuit switching**
- Protocols
- Addressing and routing
- Performance metrics
- A detailed FTP example

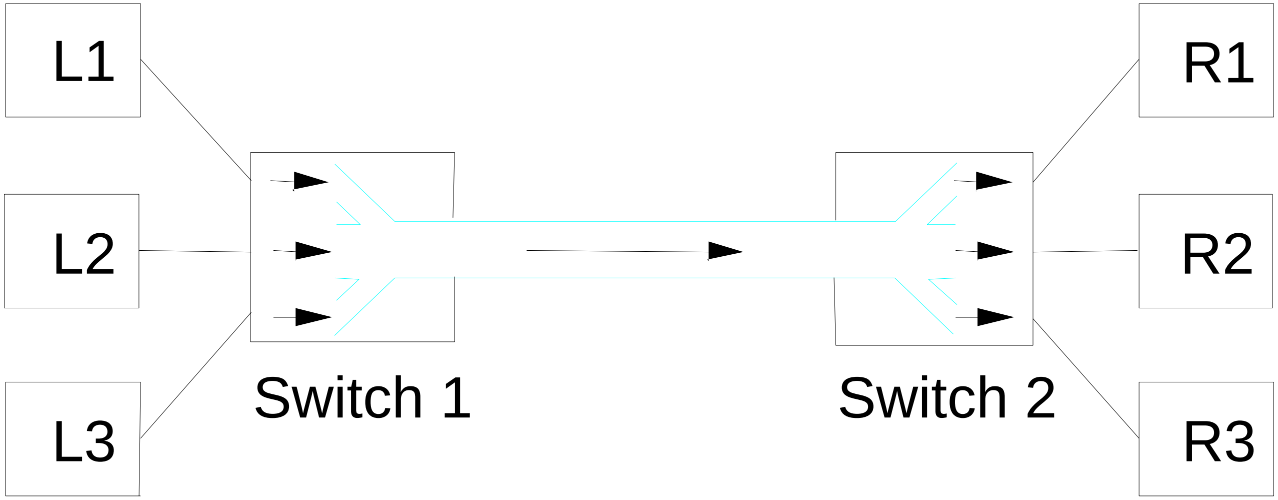
Switching Strategies

- Circuit switching: carry bit streams
  - original telephone network

- Packet switching: store-and-forward messages
  - Internet

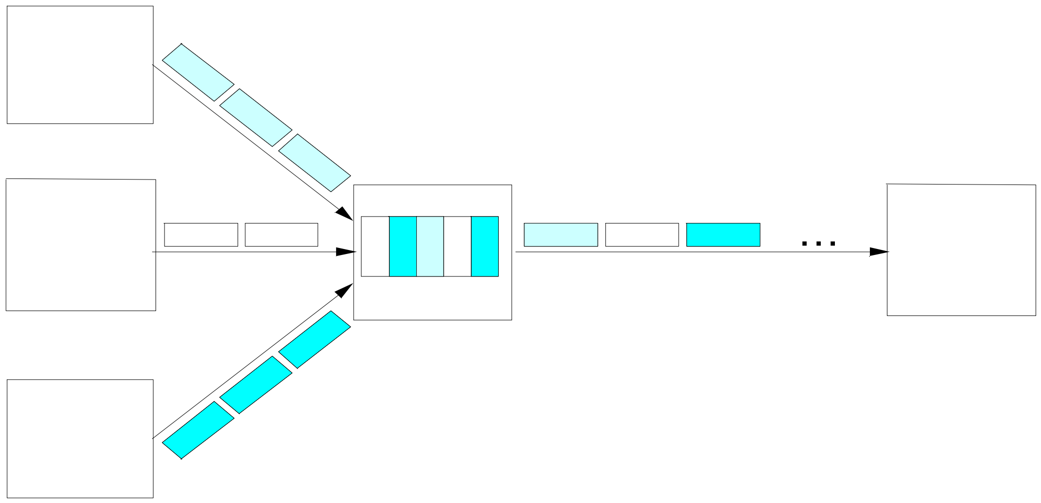
Multiplexing

- Time-Division Multiplexing (TDM)
- Frequency-Division Multiplexing (FDM)



Statistical Multiplexing

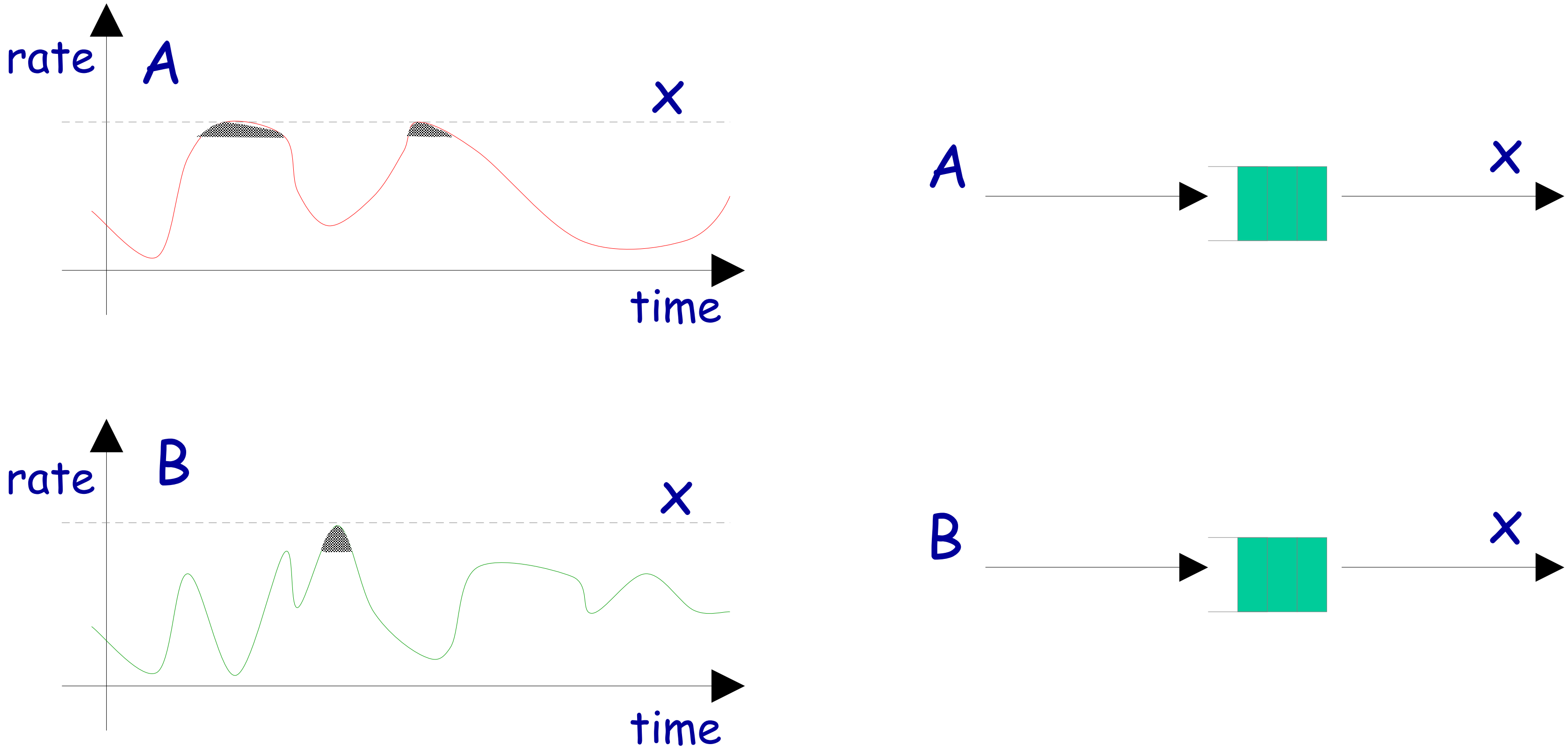
- On-demand time-division
- Schedule link on a per-packet basis
- Packets from different sources interleaved on link
- Buffer packets that are *contending* for the link
- Buffer (queue) overflow is called *congestion*



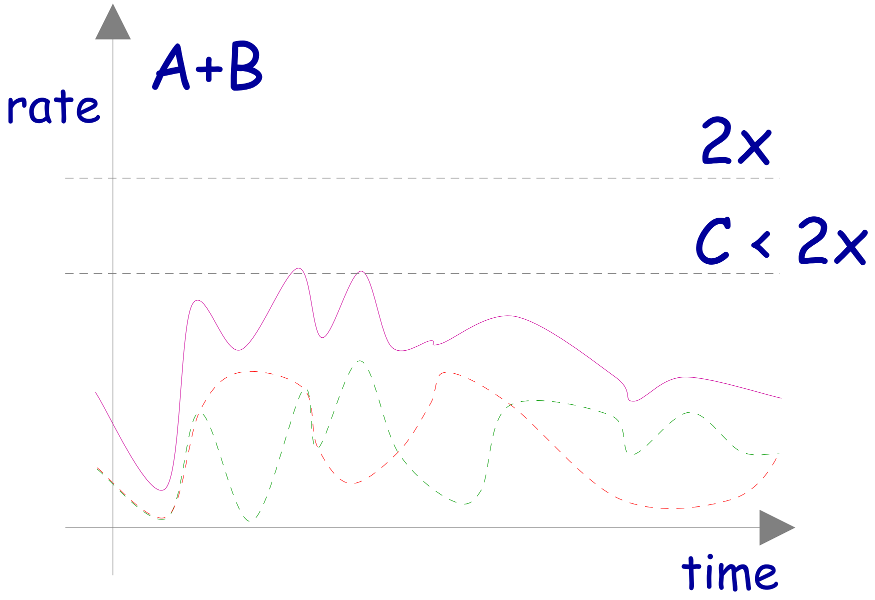
Example: Circuit vs. Packet Switching

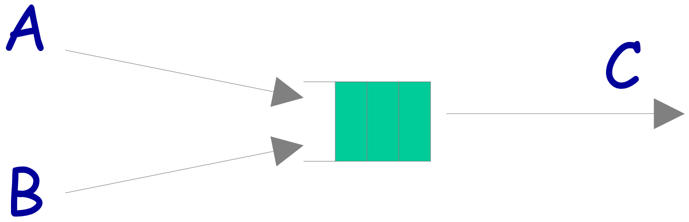
- Suppose host A sends data to host B in a bursty manner such that 1/10th of the time A actively generates 100Kbps and 9/10th of the time A sleeps
  - Under circuit switching, given a 1Mbps link, how many users can be supported?
    - Answer: 10 with no delays for any user
  - Under packet switching given a 1Mbps links how many users can be supported?
    - Answer: about 30 with low probability of delay
  - **Point: 3 times more users can be supported!**

Statistical Multiplexing



Statistical Multiplexing Gain





Statistical multiplexing gain = 2x/C

Note: the gain could be defined for a particular loss probability (in this case, x and C were chosen so that there were no losses).

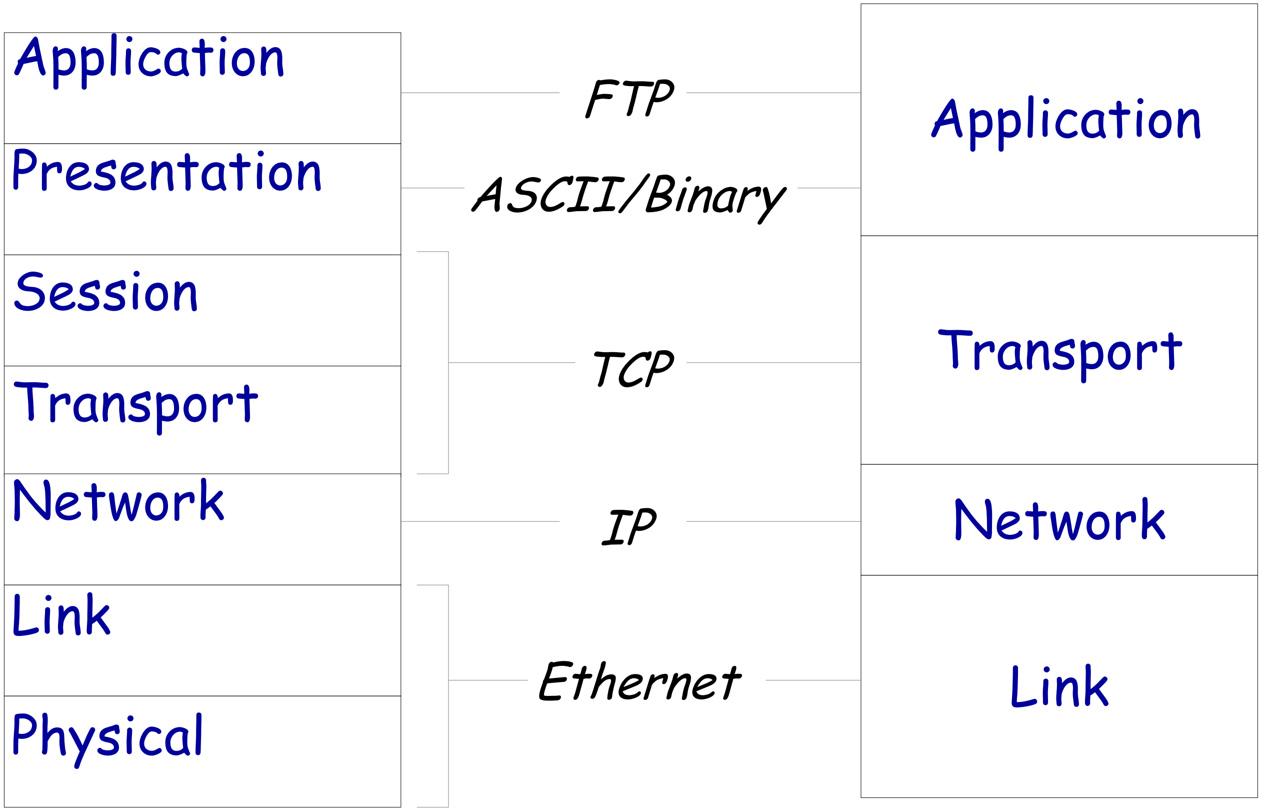
Why does the Internet use packet switching?

1. Efficient use of expensive links:
   - The links are assumed to be expensive and scarce.
   - Packet switching allows many, bursty flows to share the same link efficiently.
   - “Circuit switching is rarely used for data networks, ... because of very inefficient use of the links” - *Gallager*
2. Resilience to failure of links & routers:
   - ”For high reliability, ... [the Internet] was to be a datagram subnet, so if some lines and [routers] were destroyed, messages could be ... rerouted” - *Tanenbaum*

Internet Design Principles & Protocols

- An introduction to the mail system
- An introduction to the Internet
- Internet design principles and layering
- Brief history of the Internet
- Packet switching and circuit switching
- **Protocols**
- Addressing and routing
- Performance metrics
- A detailed FTP example

Layering and Protocols Revisited



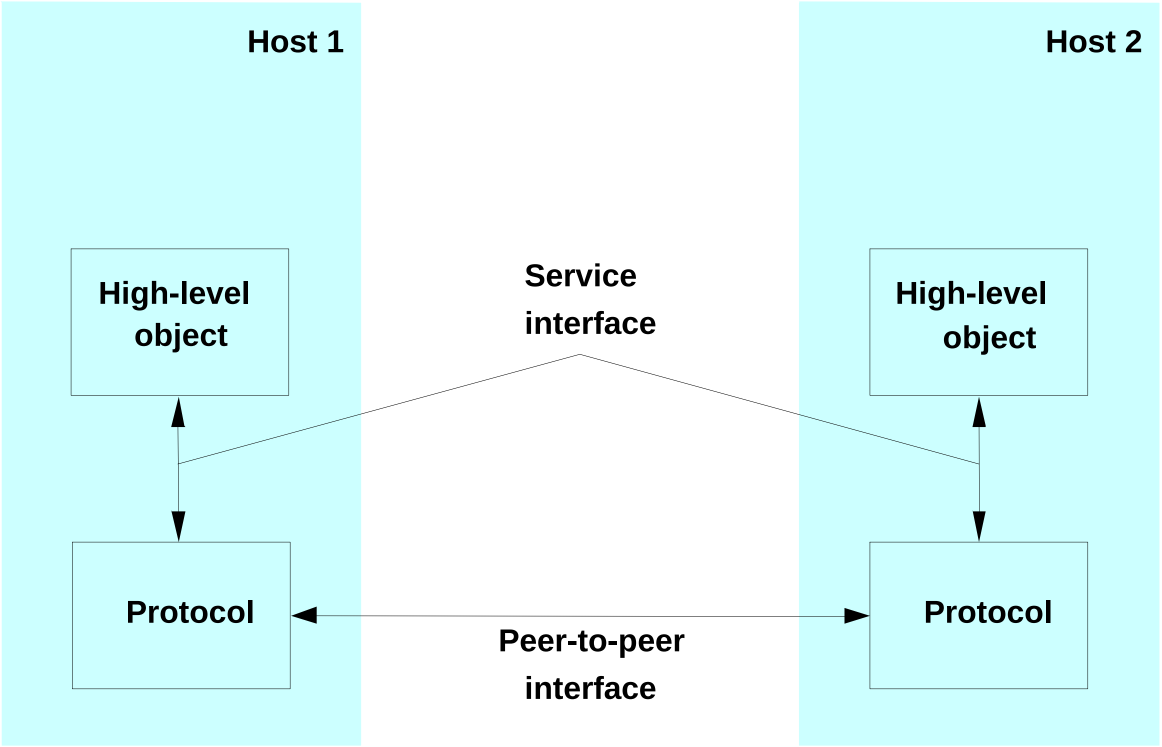
The 7-layer OSI Model

The 4-layer Internet model

Protocols

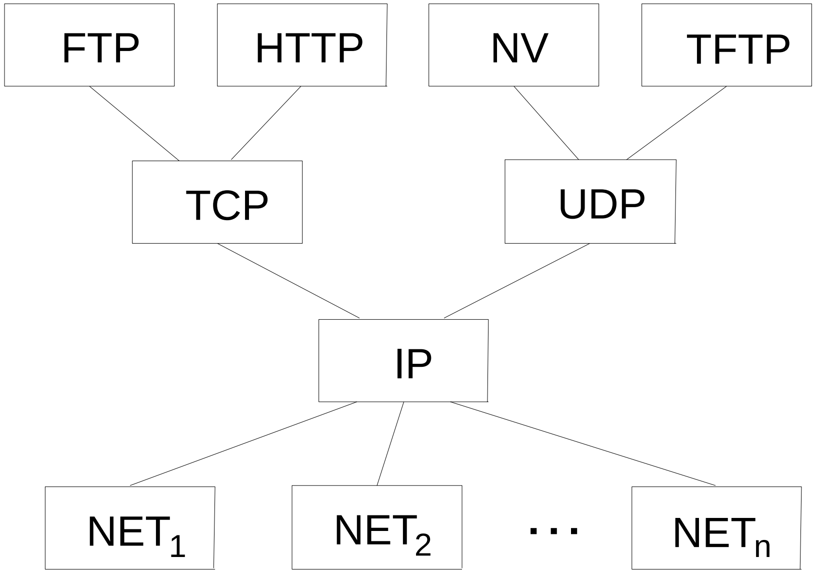
- Building blocks of a network architecture
- Each protocol object has two different interfaces
  - *service interface*: operations on this protocol
  - *peer-to-peer interface*: messages exchanged with peer
- Term “protocol” is overloaded
  - specification of peer-to-peer interface
  - module that implements this interface

Interfaces

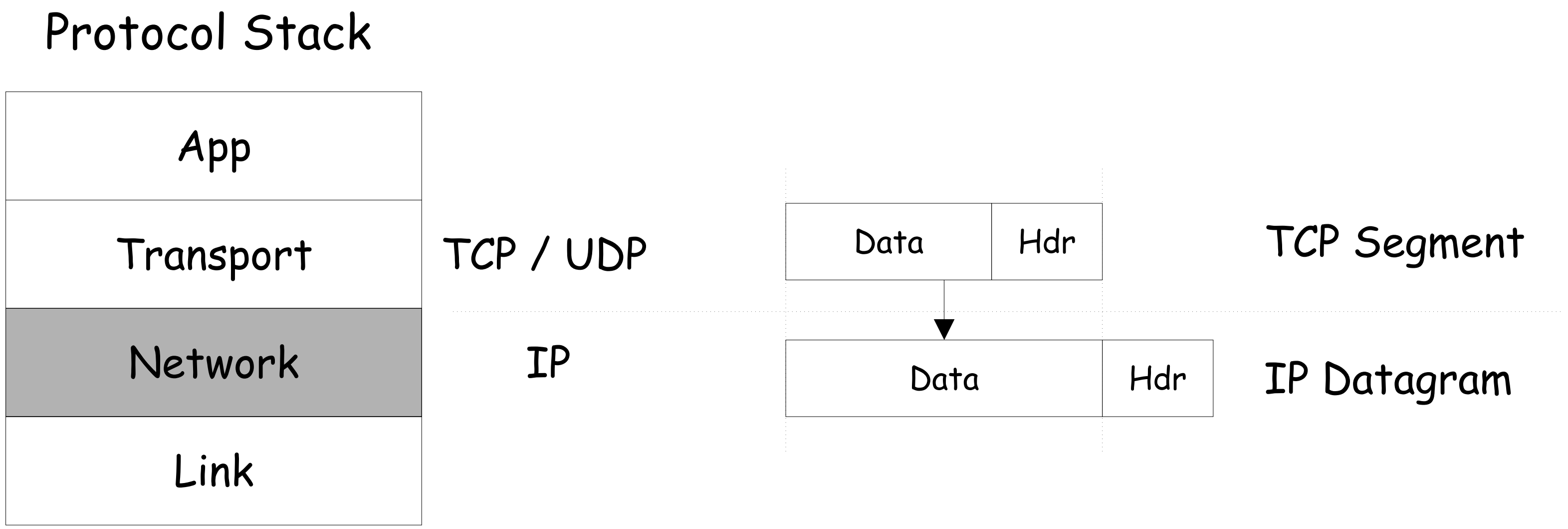

Host 1
Host 2
High-level object
High-level object
Service interface
Protocol
Protocol
Peer-to-peer interface

Hourglass Design

- Single protocol at network level insures packets will get from source to destination while allowing for flexibility



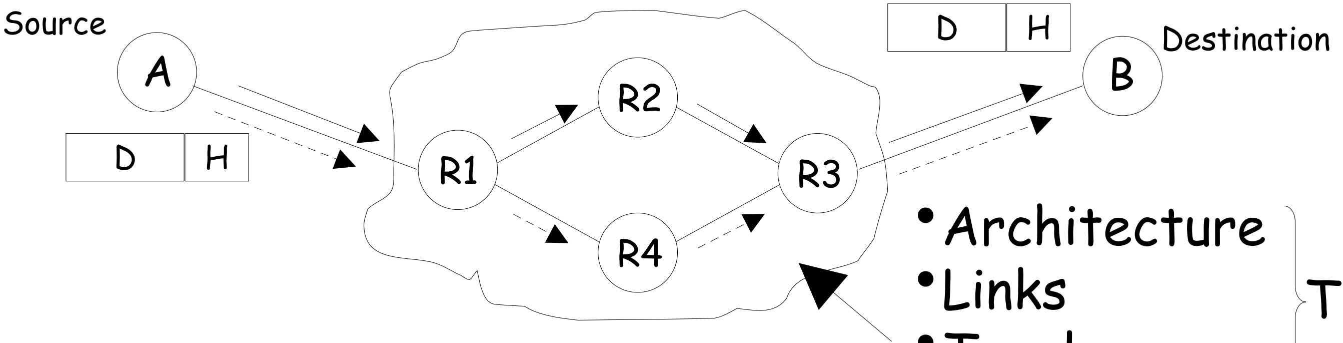
The Internet Protocol (IP)



The Internet Protocol (IP)

- Characteristics of IP

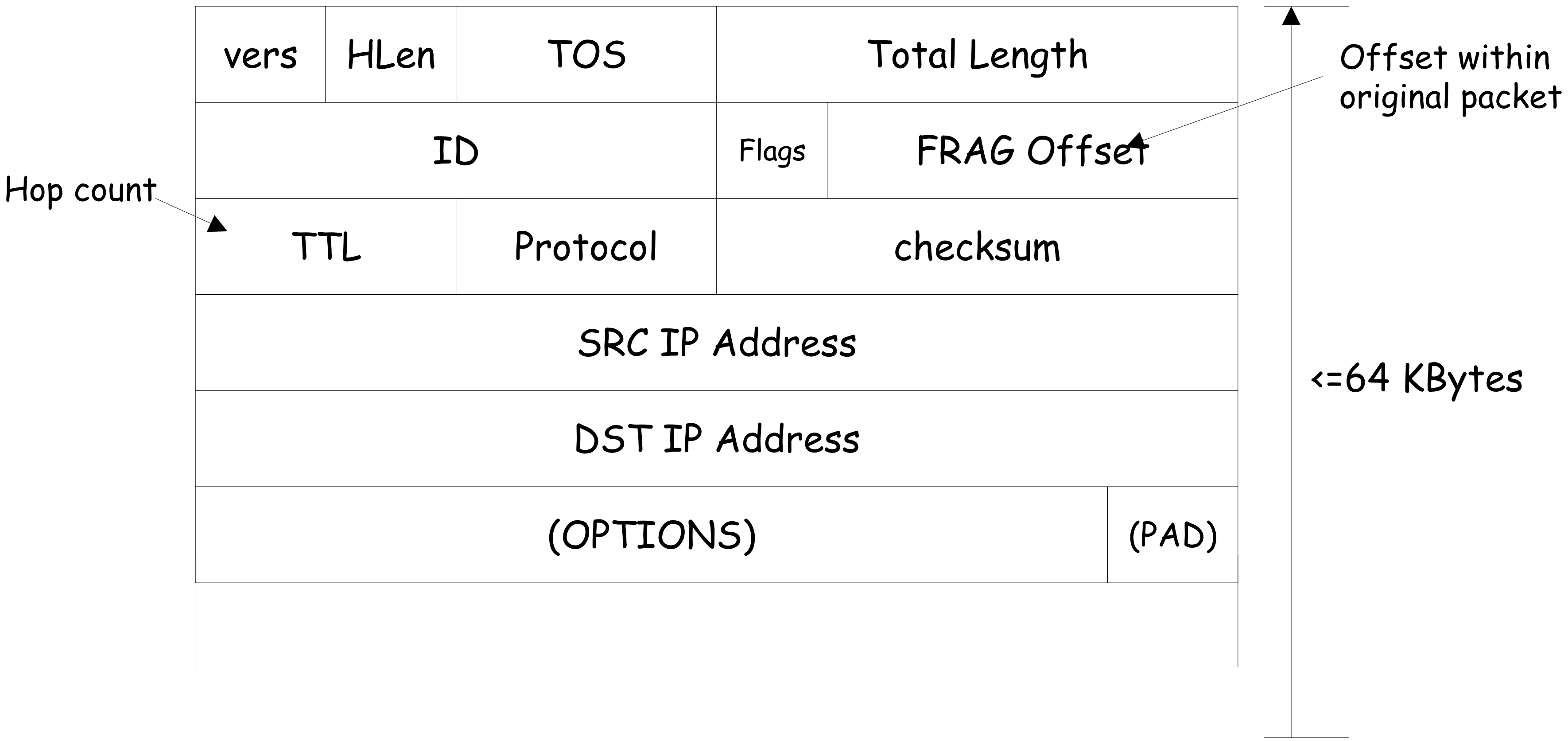
• CONNECTIONLESS:	mis-sequencing
• UNRELIABLE:	may drop packets…
• BEST EFFORT:	… but only if necessary
• DATAGRAM:	individually routed



- Architecture
- Links
- Topology

Transparent

The IP Datagram



vers | HLen | TOS | Total Length

ID | Flags | FRAG Offset

TTL | Protocol | checksum

SRC IP Address

DST IP Address

(OPTIONS) | (PAD)

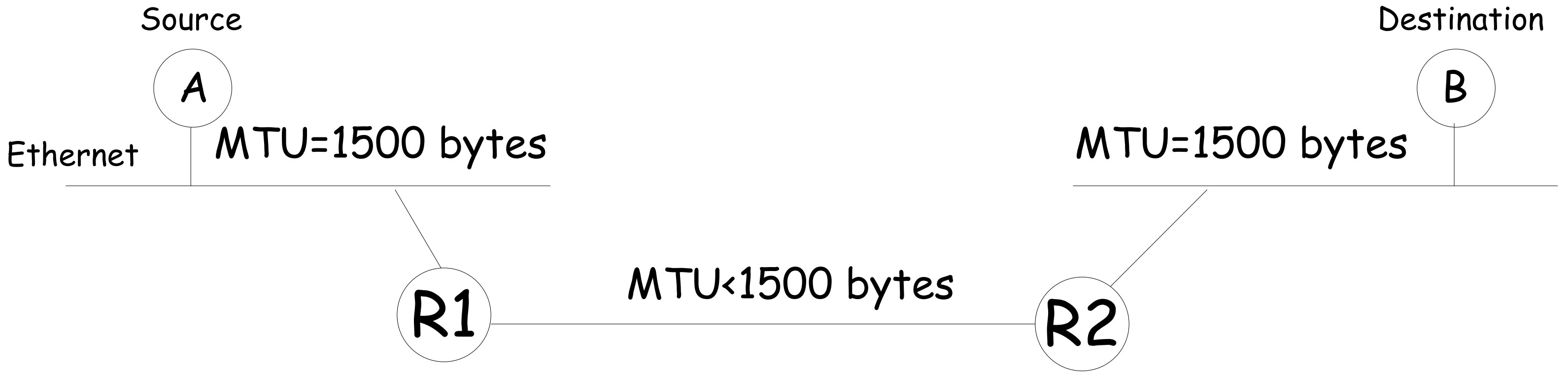
Hop count

Offset within original packet

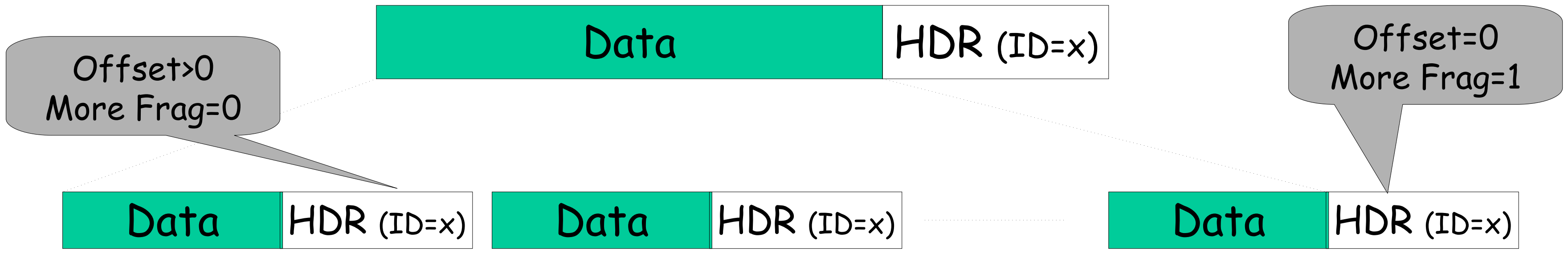
<=64 KBytes

Fragmentation

Problem: A router may receive a packet larger than the maximum transmission unit (MTU) of the outgoing link.



Solution: R1 fragments the IP datagram into mutiple, self-contained datagrams.



Fragmentation

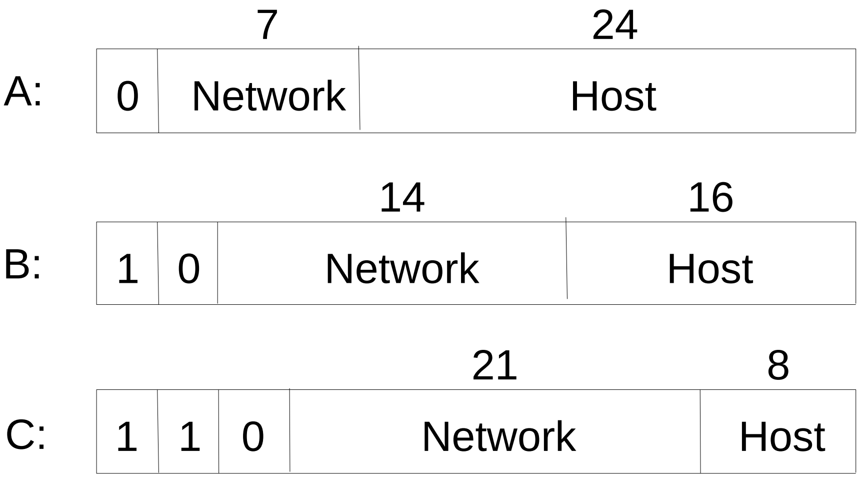
- Fragments are re-assembled by the destination host; not by intermediate routers.
- To avoid fragmentation, hosts commonly use path MTU discovery to find the smallest MTU along the path.
- Path MTU discovery involves sending various size datagrams until they do not require fragmentation along the path.
- Most links use MTU>=1500bytes today.
- Try: `traceroute –f www.mit.edu 1500` and
  `traceroute –f www.mit.edu 1501`
- (DF=1 set in IP header; routers send "ICMP" error message, which is shown as "!F").
- Can you find a destination for which the path MTU < 1500 bytes?

Internet Design Principles & Protocols

- An introduction to the mail system
- An introduction to the Internet
- Internet design principles and layering
- Brief history of the Internet
- Packet switching and circuit switching
- Protocols
- **Addressing and routing**
- Performance metrics
- A detailed FTP example

Global Addresses

- Properties
  - globally unique
  - hierarchical: network + host
- Dot Notation
  - 10.3.2.4
  - 128.96.33.81
  - 192.12.69.77



A:
7 24
0 Network Host

B:
14 16
1 0 Network Host

C:
21 8
1 1 0 Network Host

Mapping Computer Names to IP Addresses The Domain Naming System (DNS)

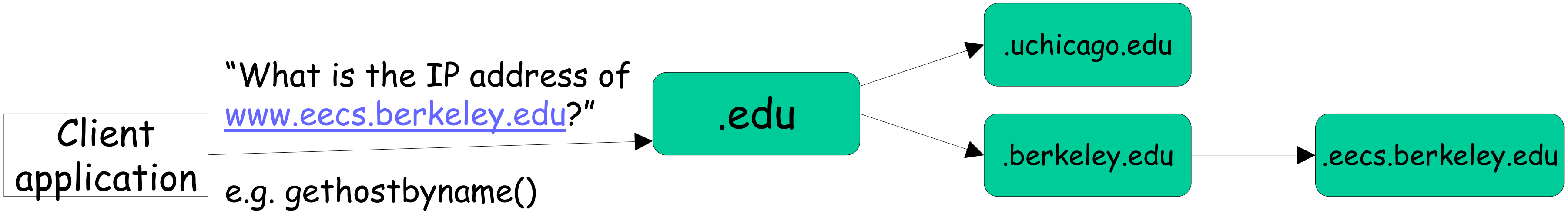
- Names are hierarchical and belong to a domain:
  - e.g. gargoyle.cs.uchicago.edu
  - Common domain names: .com, .edu, .gov, .org, .net, .uk (or other country-specific domain)
  - Top-level names are assigned by the Internet Corporation for Assigned Names and Numbers (ICANN)
  - A unique name is assigned to each organization
- DNS Client-Server Model
  - DNS maintains a hierarchical, distributed database of names
  - Servers are arranged in a hierarchy
  - Each domain has a “root” server
  - An application needing an IP address is a DNS client

Mapping Computer Names to IP Addresses
The Domain Naming System (DNS)

A DNS Query

1. Client asks local server.
2. If local server does not have address, it asks the root server of the requested domain.
3. Addresses are cached in case they are requested again.

E.g. www.eecs.berkeley.edu



Example: Try "host www.mit.edu" or "nslookup www.mit.edu"

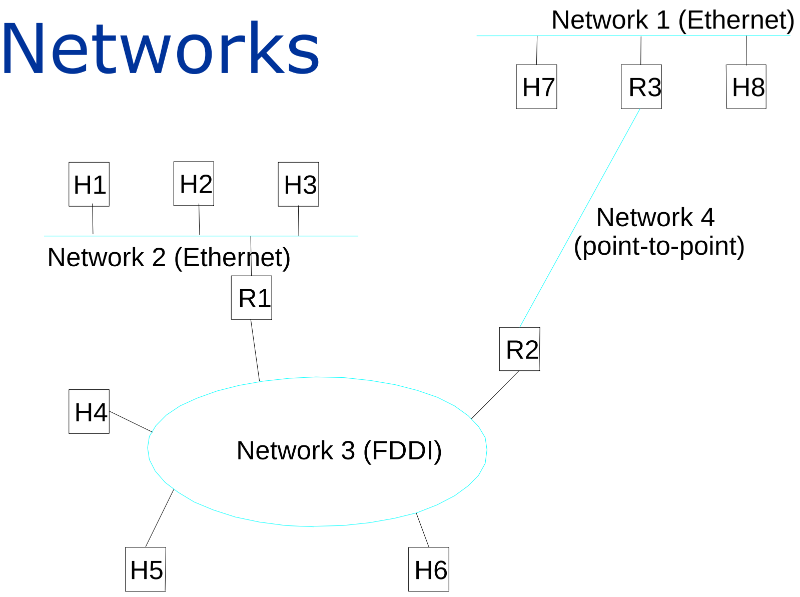
An Example of Names and Addresses

Mapping the path between two hosts

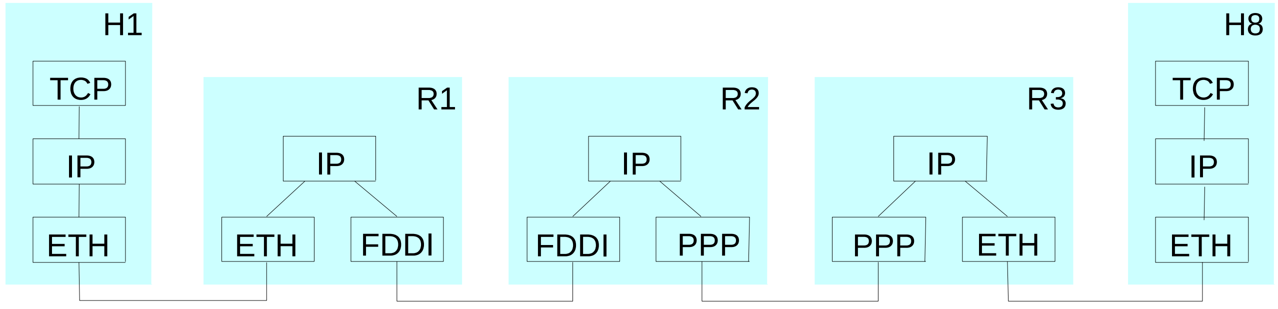
```
[11:53am] foster@gargoyle:~ 30% host gargoyle
gargoyle.cs.uchicago.edu has address 128.135.11.238
[11:54am] foster@gargoyle:~ 26% /usr/sbin/traceroute www.mit.edu
traceroute to DANDELION-PATCH.mit.edu (18.181.0.31), 30 hops max, 40 byte packets
 1  msfc-jones-v11.uchicago.edu (128.135.11.30)  0.976 ms  0.660 ms  0.543 ms
 2  msfc-1155-v903.uchicago.edu (128.135.247.62)  0.783 ms  0.782 ms  0.715 ms
 3  c12012-1155-g00.uchicago.edu (128.135.249.130)  0.782 ms  0.829 ms  0.753 ms
 4  128.135.247.98 (128.135.247.98)  1.673 ms  1.874 ms  1.974 ms
 5  mren-m10-lsd6509.startap.net (206.220.240.86)  1.868 ms  1.961 ms  1.658 ms
 6  chin-mren-ge.abilene.ucaid.edu (198.32.11.97)  17.073 ms  2.313 ms  1.892 ms
 7  nycmng-chinng.abilene.ucaid.edu (198.32.8.83)  22.313 ms  22.322 ms  24.267 ms
 8  ATM10-420-OC12-GIGAPOPNE.NOX.ORG (192.5.89.9)  27.166 ms  26.956 ms  27.390 ms
 9  192.5.89.90 (192.5.89.90)  27.407 ms  27.683 ms  27.471 ms
10  NW12-RTR-2-BACKBONE.MIT.EDU (18.168.0.21)  27.603 ms  27.502 ms  27.205 ms
11  DANDELION-PATCH.MIT.EDU (18.181.0.31)  28.309 ms *  27.996 ms
```

IP Internet

- Concatenation of Networks



- Protocol Stack

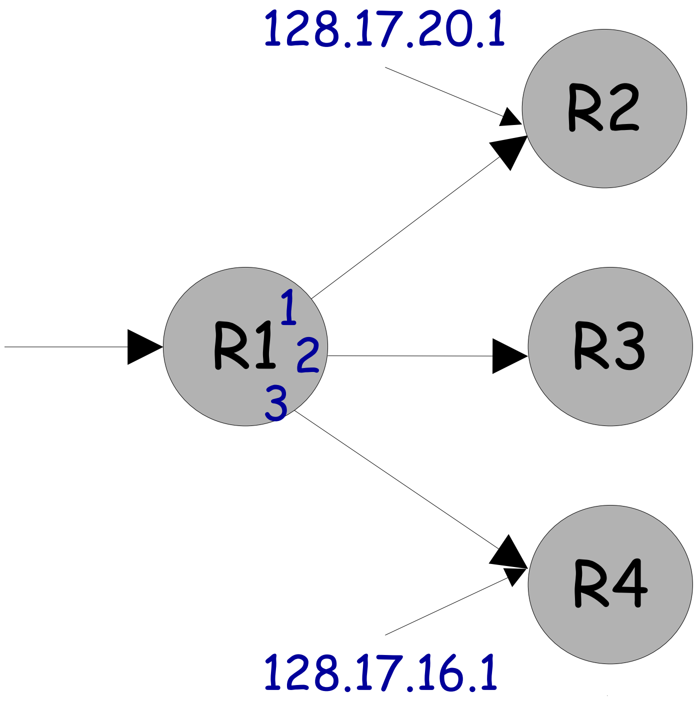


Datagram Forwarding

- Strategy
  - every datagram contains destination's address
  - if directly connected to destination network, then forward to host
  - if not directly connected to destination network, then forward to some router
  - forwarding table maps network number into next hop
  - each host has a default router
  - each router maintains a forwarding table
- Example

Network Number	Next Hop
1	R3
2	R1
3	interface 1
4	interface 0

How a Router Forwards Datagrams



e.g. 128.9.16.14 => Port 2

Prefix	Next-hop	Port
65/8	128.17.16.1	3
128.9/16	128.17.14.1	2
128.9.16/20	128.17.14.1	2
128.9.19/24	128.17.10.1	7
128.9.25/24	128.17.14.1	2
128.9.176/20	128.17.20.1	1
142.12/19	128.17.16.1	3

Forwarding/routing table

Forwarding Tables

- Suppose there are *n* possible destinations, how many bits are needed to represent addresses in a routing table?
  - – $\log_2 n$
- So, we need to store and search $n * \log_2 n$ bits in routing tables?
  - – We're smarter than that!

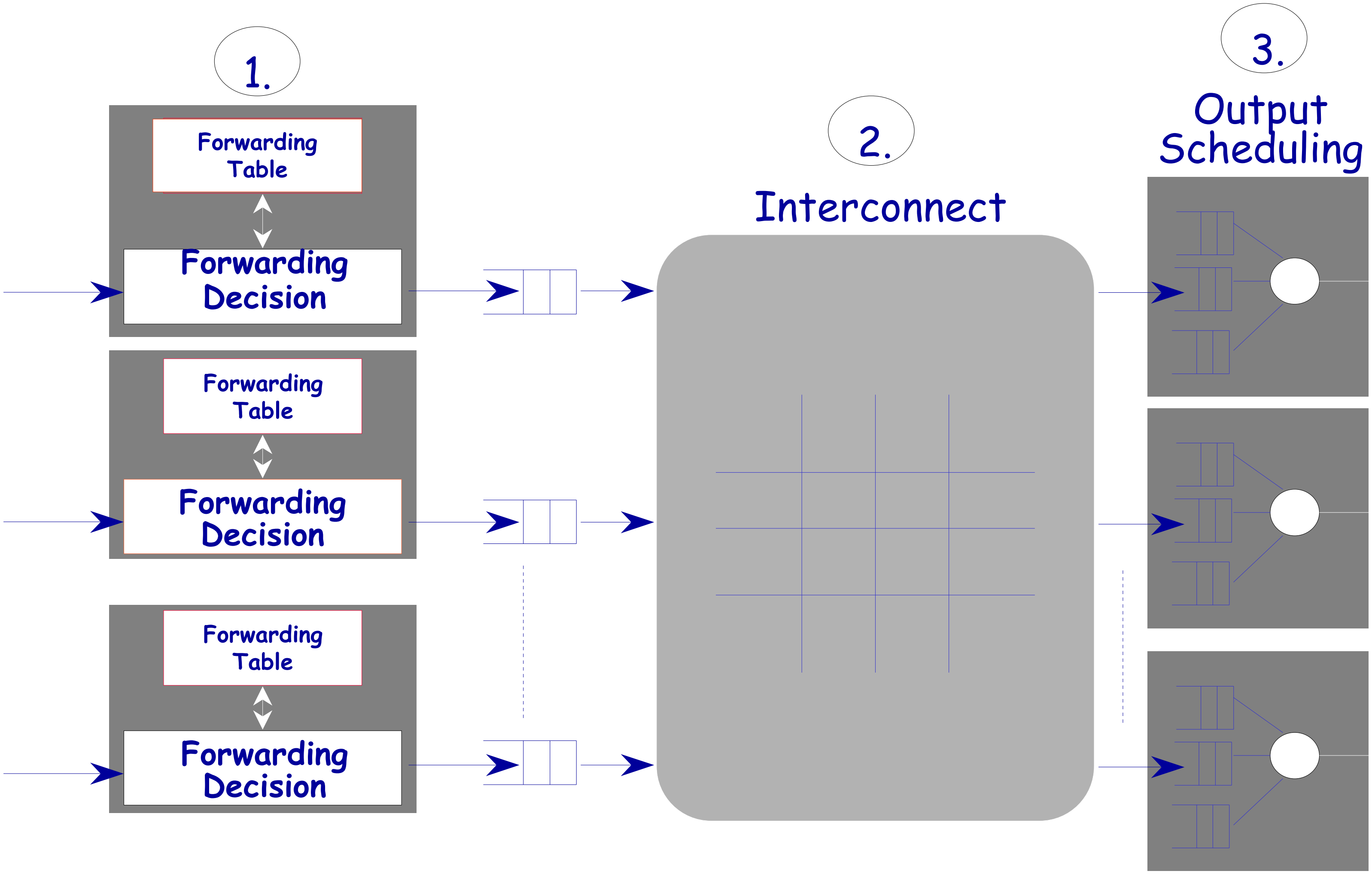
How a Router Forwards Datagrams

- Every datagram contains a destination address.
- The router determines the prefix to which the address belongs, and routes it to the“Network ID” uniquely identifies a physical network.
- All hosts and routers sharing a Network ID share same physical network.

Forwarding Datagrams

- Is the datagram for a host on directly attached network?
- If no, consult forwarding table to find next-hop.

Inside a Router



Internet Design Principles & Protocols

- An introduction to the mail system
- An introduction to the Internet
- Internet design principles and layering
- Brief history of the Internet
- Packet switching and circuit switching
- Protocols
- Addressing and routing
- **Performance metrics**
- A detailed FTP example

Performance Metrics

- Bandwidth (throughput)
  - data transmitted per time unit
  - link versus end-to-end
  - notation
    - KB = 2^{10} bytes
    - Mbps = 10^6 bits per second
- Latency (delay)
  - time to send message from point A to point B
  - one-way versus round-trip time (RTT)
  - components

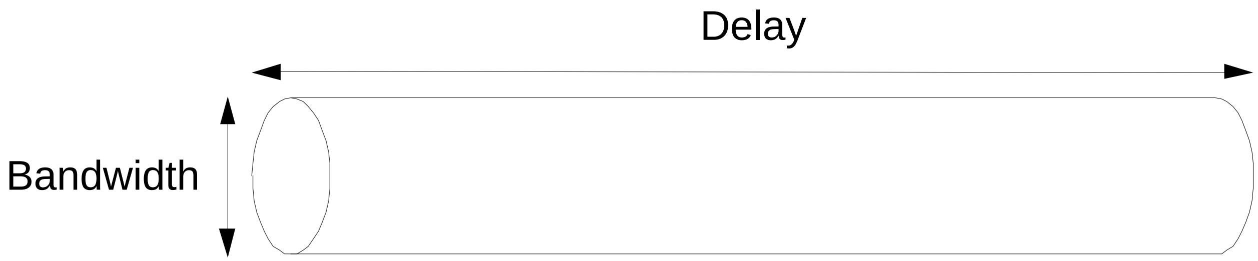
    Latency = Propagation + Transmit + Queue
    Propagation = Distance / c
    Transmit = Size / Bandwidth
  - Speed of light in fiber: 5 usec/km

Bandwidth versus Latency

- Relative importance
  - 1 byte: 1ms vs 100ms dominates 1Mbps vs 100Mbps
  - 25 MB: 1Mbps vs 100Mbps dominates 1ms vs 100ms
- Infinite bandwidth
  - RTT dominates
    - Throughput = TransferSize / TransferTime
    - TransferTime = RTT + 1/Bandwidth x TransferSize
- It’s a big planet!

Delay x Bandwidth Product

- Amount of data “in flight” or “in the pipe”
- Example: 100ms x 45Mbps = 560KB



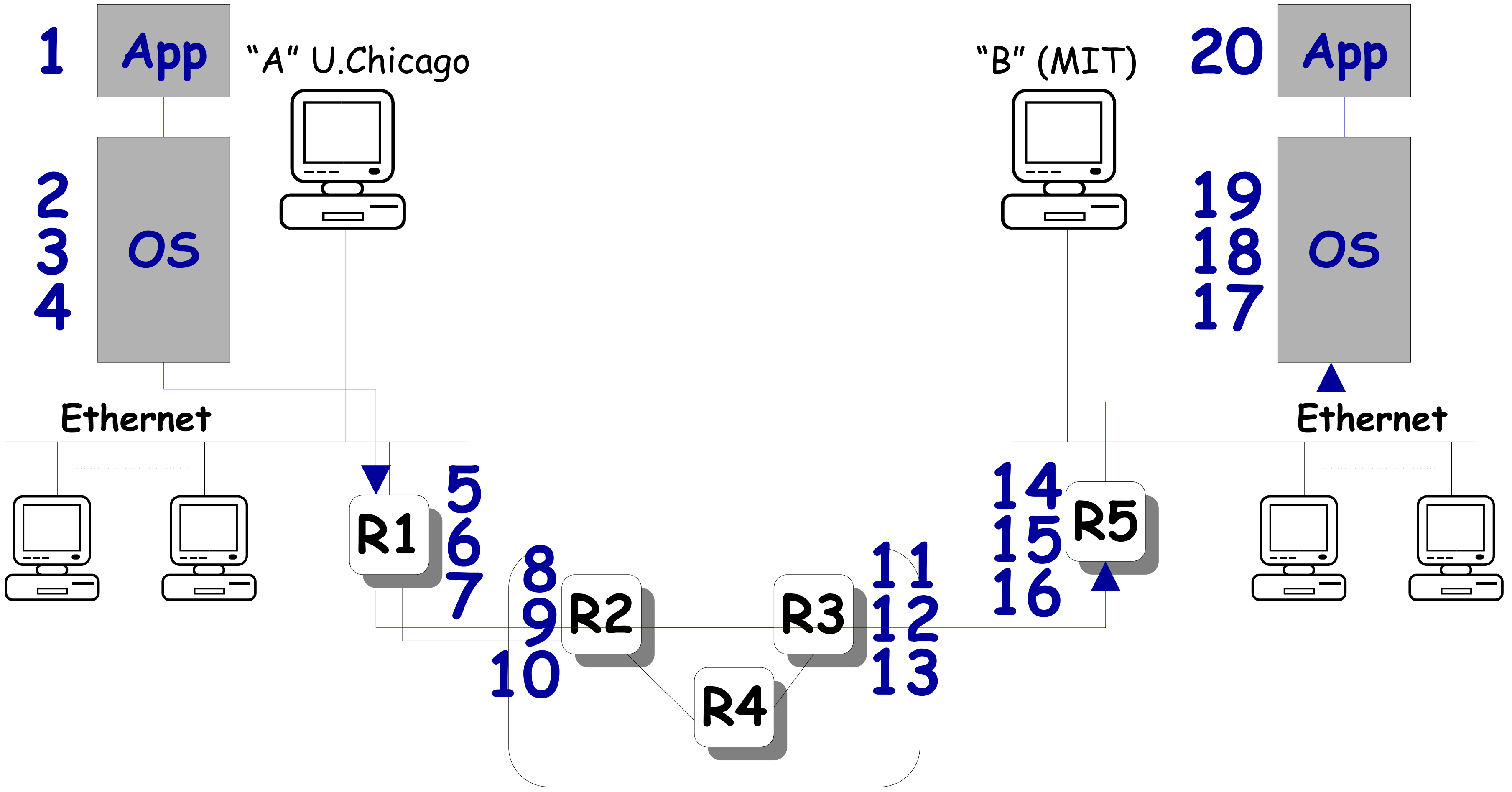
Delay

Bandwidth

Internet Design
Principles & Protocols

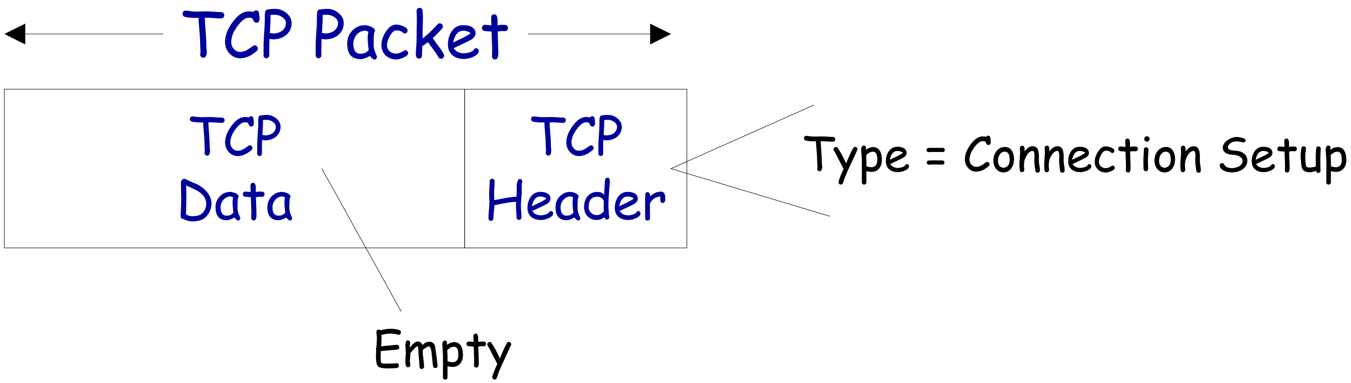
- An introduction to the mail system
- An introduction to the Internet
- Internet design principles and layering
- Brief history of the Internet
- Packet switching and circuit switching
- Protocols
- Addressing and routing
- Performance metrics
- **A detailed FTP example**

Example: FTP over the Internet Using TCP/IP and Ethernet



In the Sending Host

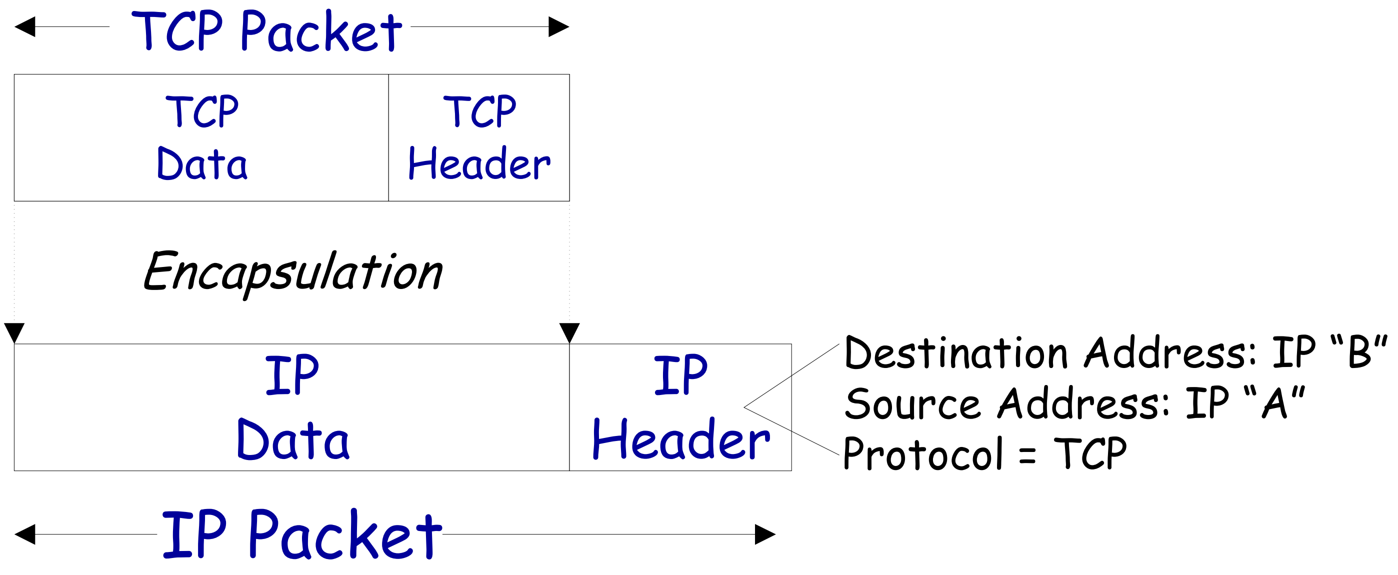
1. **Application-Programming Interface (API)**
   - Application requests TCP connection with "B"
2. **Transmission Control Protocol (TCP)**
   - Creates TCP "Connection setup" packet
   - TCP requests IP packet to be sent to "B"



In the Sending Host (2)

3. Internet Protocol (IP)

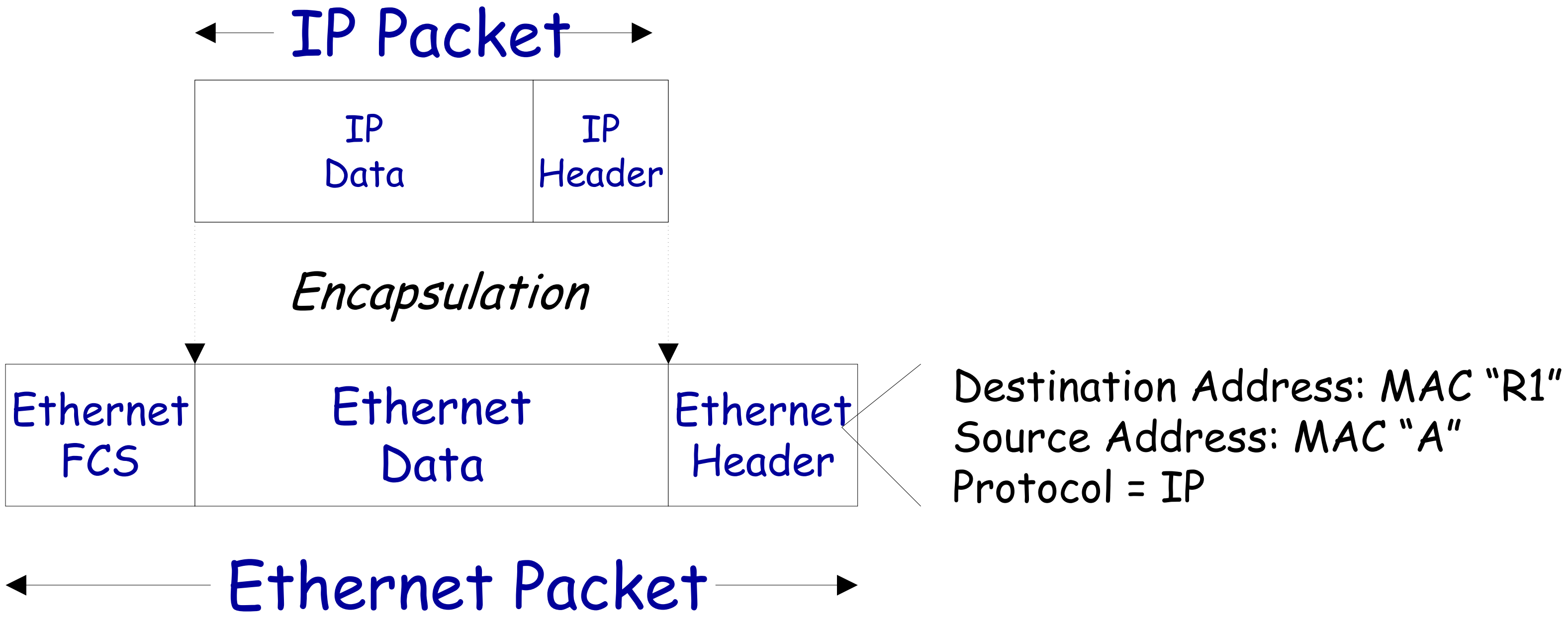
- Creates IP packet with correct addresses
- IP requests packet to be sent to router



In the Sending Host (3)

4. Link ("MAC" or Ethernet) Protocol

- Creates MAC frame with Frame Check Sequence
- Wait for Access to the line.
- MAC requests PHY to send each bit of the frame.

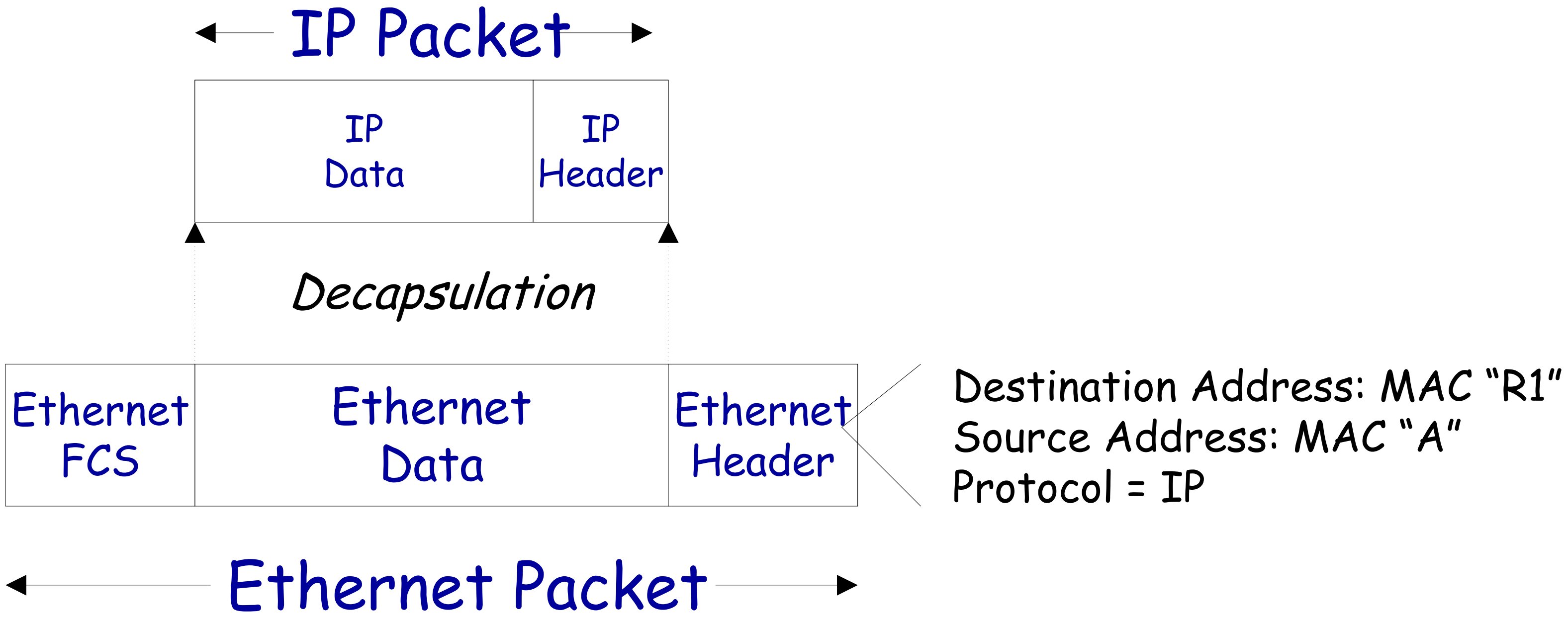


Destination Address: MAC "R1"
Source Address: MAC "A"
Protocol = IP

In Router R1

5. Link (“MAC” or Ethernet) Protocol

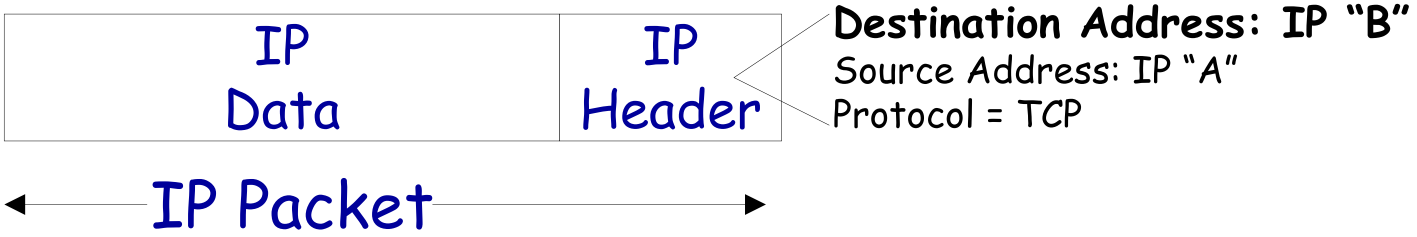
- Accept MAC frame, check address and Frame Check Sequence (FCS).
- Pass data to IP Protocol.



In Router R1

6. Internet Protocol (IP)

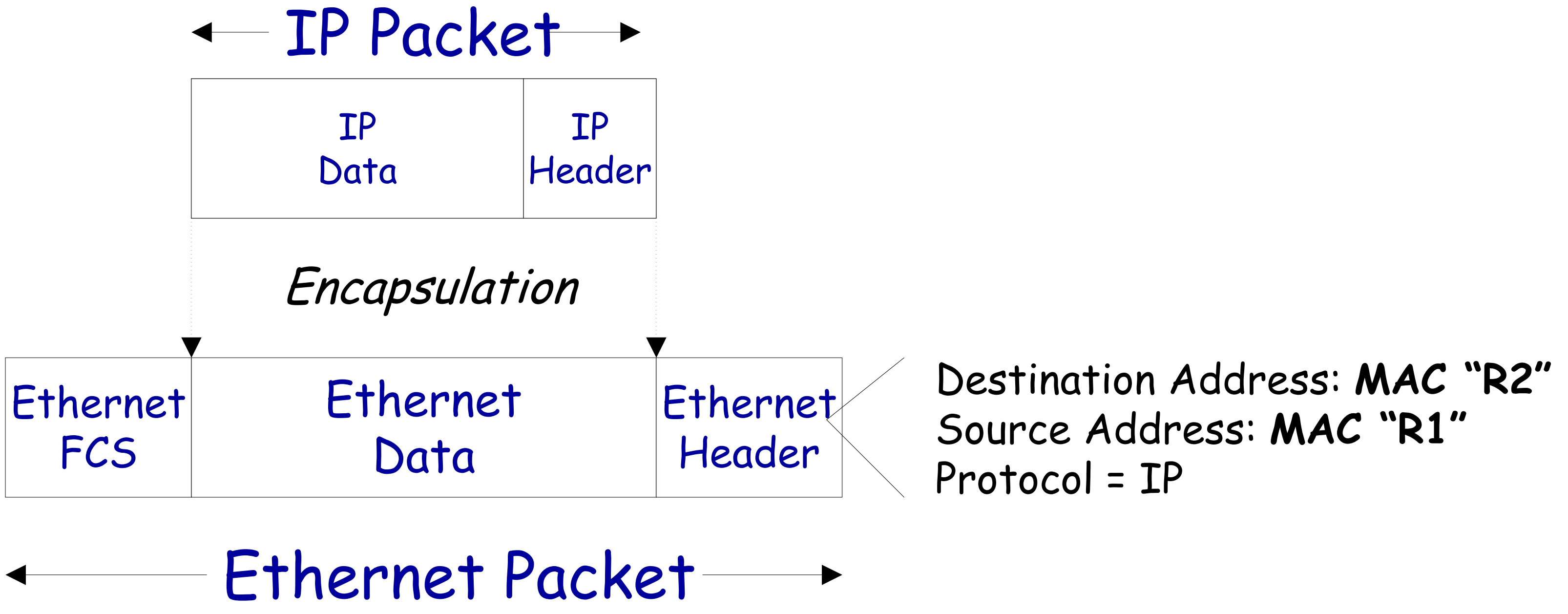
- Use IP destination address to decide where to send packet next ("next-hop routing")
- Request Link Protocol to transmit packet



In Router R1

7. Link ("MAC" or Ethernet) Protocol

- Creates MAC frame with Frame Check Sequence
- Wait for Access to the line.
- MAC requests PHY to send each bit of the frame.



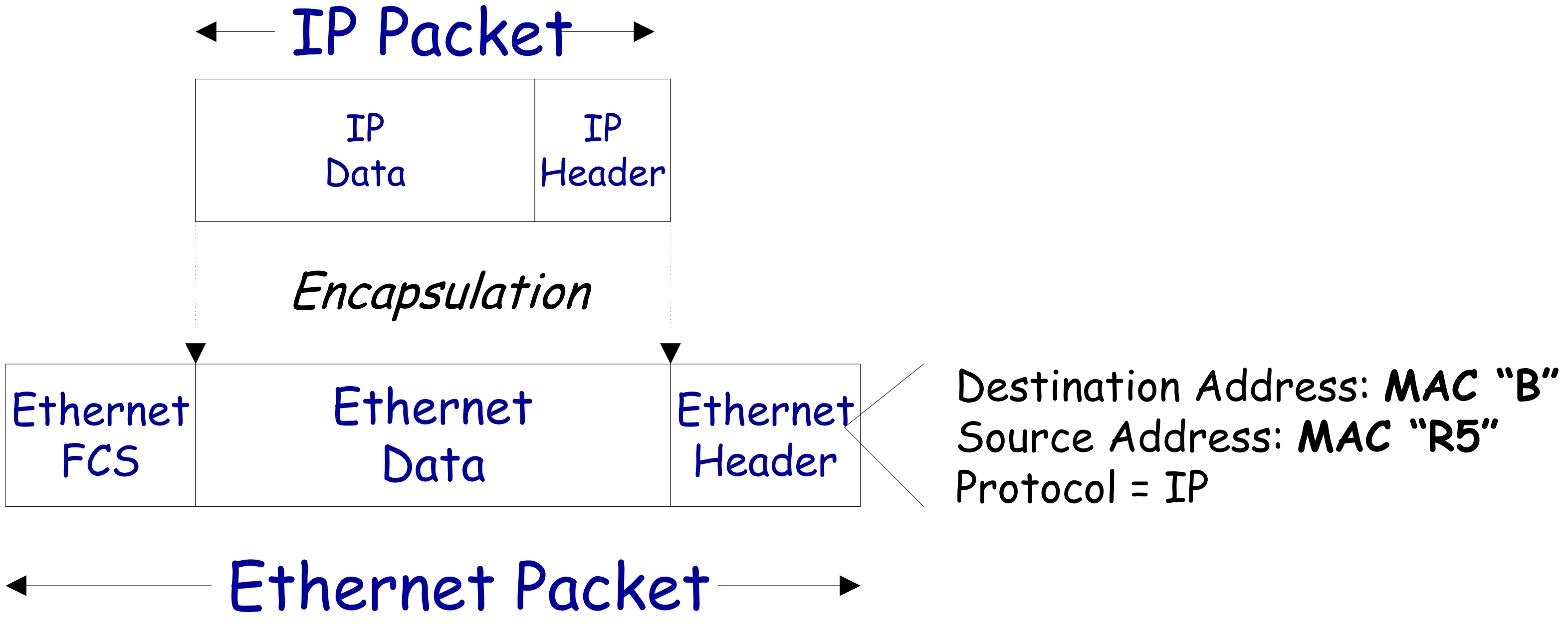
Destination Address: ***MAC "R2"***
Source Address: ***MAC "R1"***
Protocol = IP

In Routers R2, R3, R5

Same operations as Router R1

16. Link (“MAC” or Ethernet) Protocol

- Creates MAC frame with Frame Check Sequence
- Wait for Access to the line.
- MAC requests PHY to send each bit of the frame.

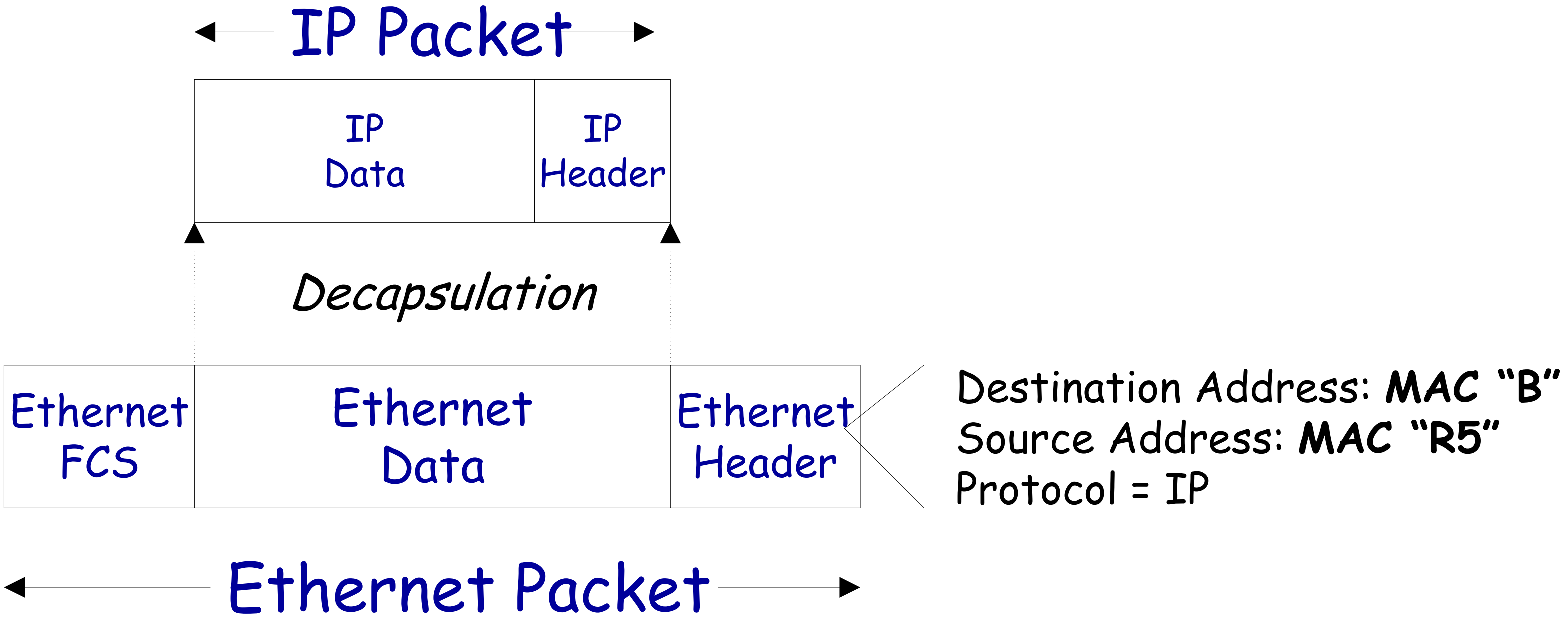


Destination Address: **MAC “B”**
Source Address: **MAC “R5”**
Protocol = IP

In the receiving host

17. Link ("MAC" or Ethernet) Protocol

- Accept MAC frame, check address and Frame Check Sequence (FCS).
- Pass data to IP Protocol.

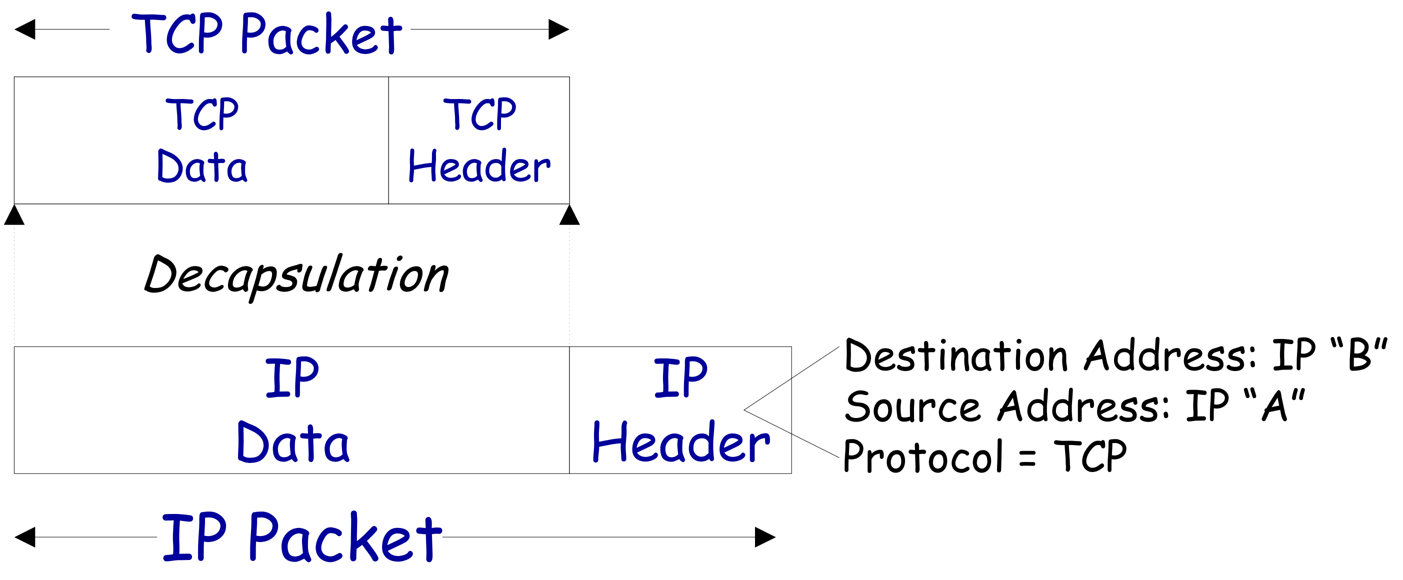


Destination Address: **MAC "B"**
Source Address: **MAC "R5"**
Protocol = IP

In the receiving host (2)

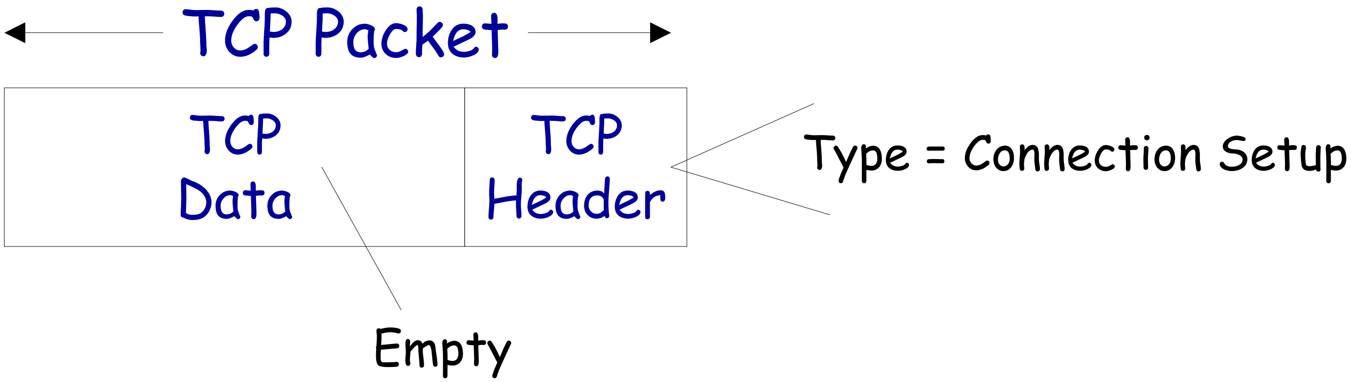
18. Internet Protocol (IP)

- Verify IP address.
- Extract/decapsulate TCP packet from IP packet.
- Pass TCP packet to TCP Protocol.



In the receiving host (3)

19. **Transmission Control Protocol (TCP)**
    - Accepts TCP “Connection setup” packet
    - Establishes connection by sending “Ack”.
20. **Application-Programming Interface (API)**
    - Application receives request for TCP connection with “A”.



Next Week

- We’ll cover:
  - Internetworking
  - Transport
  - Routing
- I want you to:
  - Read Peterson and Davies Ch 1 and 2
  - Read “End to End Arguments in System Design”
  - Use traceroute to determine paths to following locations & build map of network
    - ANL, IIT, NWU, UIC, Loyola, UIUC, Purdue, Indiana