

End-to-end transport protocols, TCP, congestion control, and DNS

George Porter
CSE 124
January 15, 2015

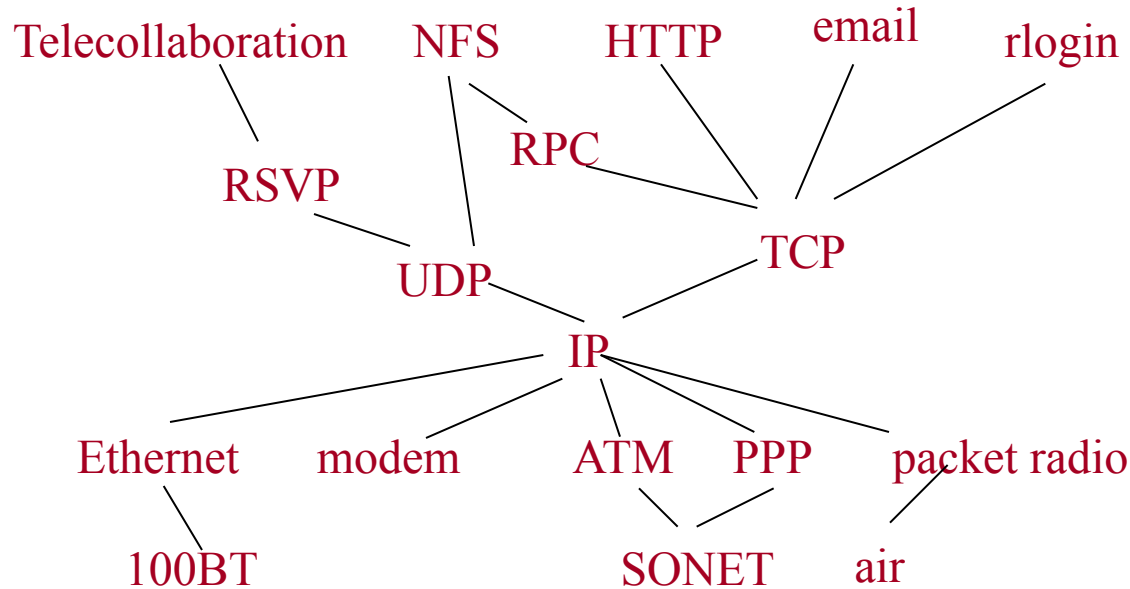
Announcements

- First assignment available
 - HTTP Server: due Feb 3, before the start of class
- First homework due Jan 27, 2015
- My office hours have moved slightly earlier
 - Wed 2:00-3:00pm, CSE 3104
 - No office hours on Jan/21/2015

Internetworking and layering

Internet Architecture

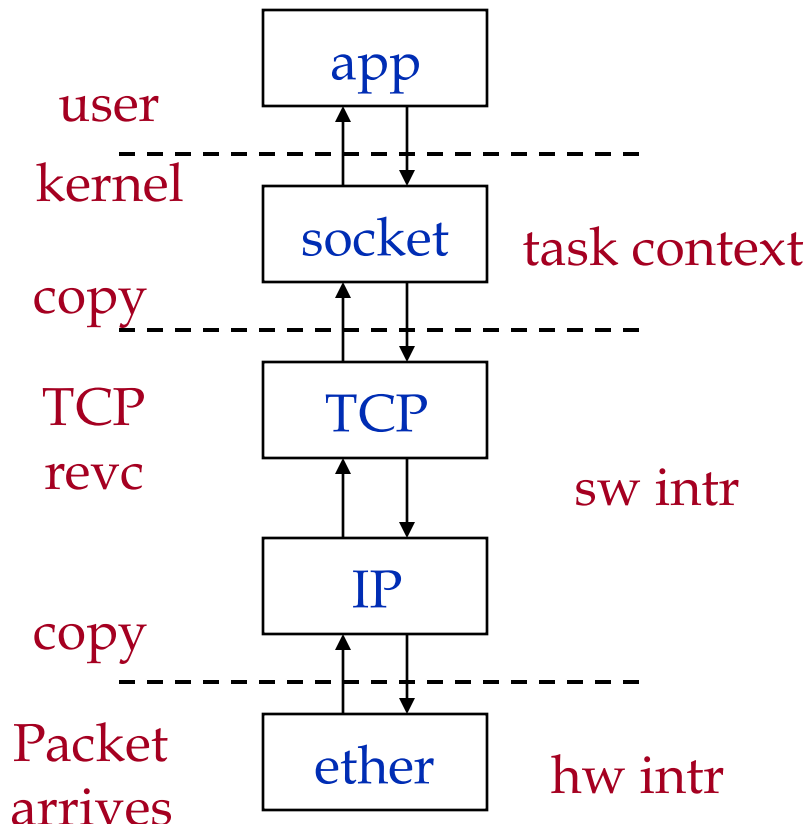
■ IP Hourglass:



● Layering not strict

- Can define new abstractions on any existing protocol

Layering in Internet Applications



■ Bottlenecks

- Boundary crossings
- Copies
- Context switches

■ Layering nice way to logically consider protocols

- May not lead to fastest implementation
- But! Processors are getting faster... people are getting more expensive

IP Address Allocation

- 32-bit values, consisting of four 8-bit ‘octets’
- E.g., 206.109.3.23, 172.22.16.1
- Originally, 4 address classes
 - “A”: 0 | 7 bit network | 24 bit host (1M each)
 - “B”: 10 | 14 bit network | 16 bit host (64K)
 - “C”: 110 | 21 bit network | 8 bit host (255)
 - “D”: 1110 | 28 bit multicast group #
- Assign net # centrally, host # locally

IP Address Issues

- We can run out
 - 4B IP addresses; 4B micros in 1997
- We will run out faster if sparsely allocated
 - Rigid structure causes internal fragmenting
 - E.g., assign a class C address to site with 2 computers
Waste 99% of assigned address space
- Need address aggregation to keep tables small
 - 2 million class C networks
 - Entry per network in IP forwarding tables
Scalability?

Subnetting

- Subnetting: introduce *subnet masks*
 - All hosts on same network already have same network #
 - Subnet mask: hosts on one network have same subnet #
 - Subnet mask: 255.255.255.128, IP: 128.96.34.15
 - Mask is a series of 1s that ‘mask out’ the part of the IP address that refers to the subnet number
- Variable subnet sizes
 - Computer Science department needs 4096 addresses
 $255.255.240.0 = 11111111.11111111.11110000.00000000$
 - Small lab on campus that needs 32 IP addresses
 $255.255.255.224 = 11111111.11111111.11111111.11100000$
Signifies last 5 bits are the host ($2^5 = 32$)

Subnet examples

■ 172.22.16.0/24

- /24 corresponds to

11111111 11111111 11111111 00000000

- 172.22.16.0 && /24 → network is 172.22.16.XX
- Allows for 256 hosts to live on that “subnet”

■ 206.109.1.128/28

- /28 corresponds to

11111111 11111111 11111111 11110000

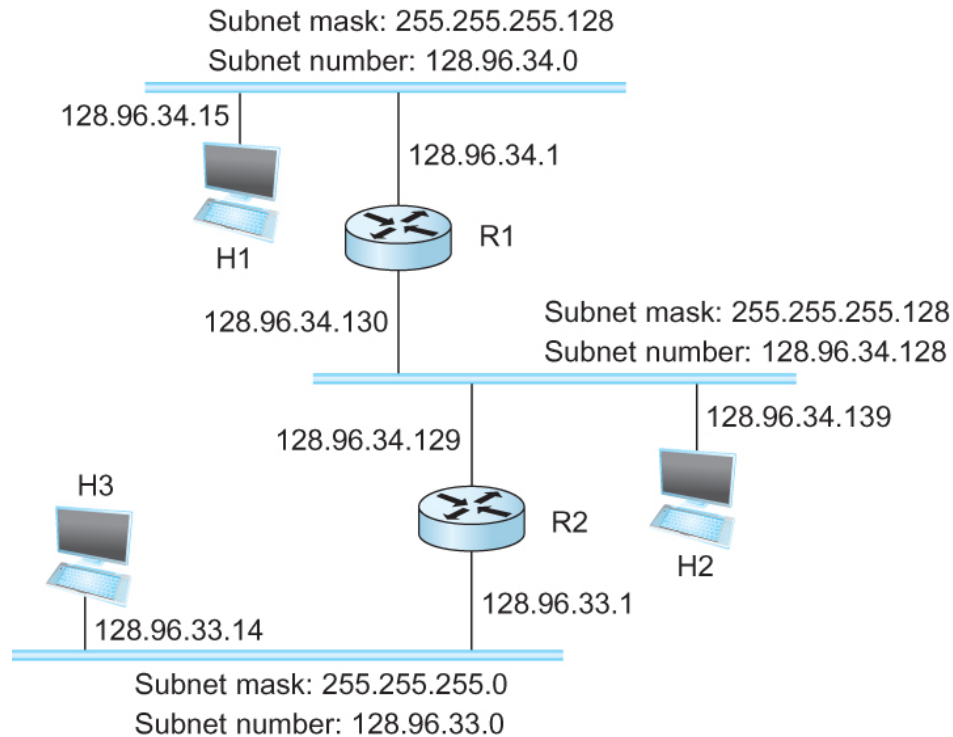
- 206.109.1.128 →

11001110 01101101 00000001 10000000

- ANDing these together results in a
network of 206.109.1.128

16 hosts on that network (last 4 bits)

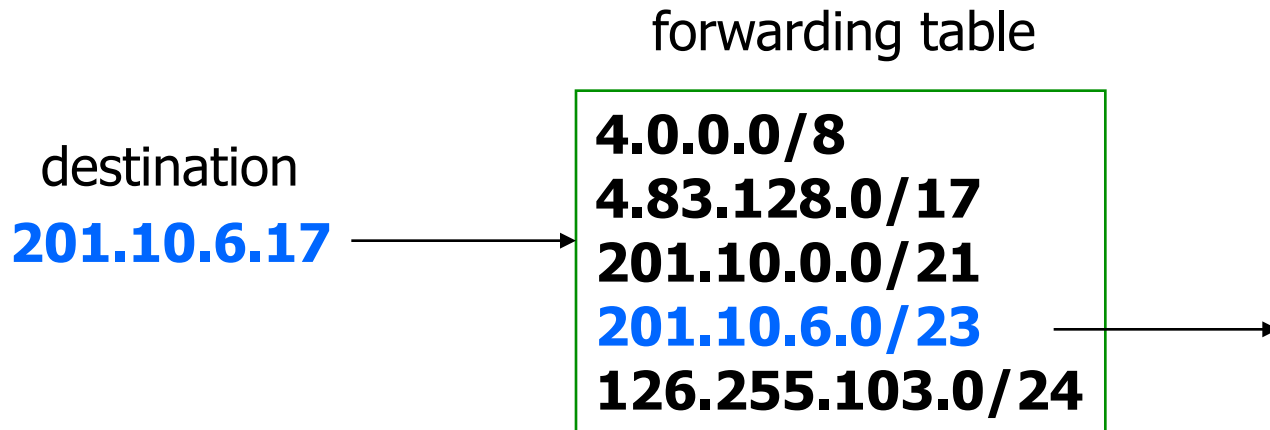
Subnetting Example



SubnetNumber	SubnetMask	NextHop
128.96.34.0	255.255.255.128	Interface 0
128.96.34.128	255.255.255.128	Interface 1
128.96.33.0	255.255.255.0	R2

- Forwarding Table at Router R1

Longest-prefix matching example



201.10.6.17: 11001001 00001010 00000110 00010001
/21: 11111111 11111111 11111000 00000000
/23: 11111111 11111111 11111110 00000000

Different Kinds of Addresses

- Domain name (e.g. *www.google.com*)
 - Global, human readable
- IP Address (e.g. 66.102.7.104)
 - Global, works across all networks
- Ethernet (e.g. *08-00-2b-18-bc-65*)
 - Local, works on a particular network

Address Resolution Protocol (ARP)

- IP forwarding tables: one entry per *network* not *host*
 - Thus, routes designed to get packets to proper network
 - Network needs to take over from there to get to proper host
- Address resolution protocol (ARP) translates IP addresses to link-level addresses (e.g., 48-bit Ethernet addr)
 - Broadcast request over network for IP→link-level mapping
 - Maintain local cache (with timeout)

Address Resolution Protocol (ARP)

(152.3.140.5)



(152.3.145.240)



*Broadcast: Anyone know the
Ethernet address for 152.3.145.240?*

Ethernet



*Reply: Yes, I'm at
08-00-2b-18-bc-65*



Ethernet

Dynamic Host Configuration Protocol (DHCP)

- IP cannot be configured into hosts when they are produced
 - Depends upon the attached network
 - Difficult for admins to manually configure hosts
- DHCP allows for dynamic assignment of IP addresses
 - Option one: store config info at DHCP server (indexed by unique MAC address)
 - Option two: DHCP server maintains pool of available addrs

*Provides **lease** for individual IP addresses*

UCSD-PROTECTED, probably your home/dorm/etc
- When host needs config info, broadcast DHCPDISCOVER
 - One host per physical network acts as relay to DHCP server

Example: What Happens When You Click on a Web Link?

Putting things together: What Happens When You Click on a Web Link?

Your computer



www.google.com



?

Internet

Steps in Retrieving a Web Page

- Extract hostname from URL
 - <http://www.google.com/foo> → www.google.com
- Use DNS to translate www.google.com to IP address
 - Used for Internet routing, covered next lecture
- Establish a TCP (socket) connection to 66.102.7.104
 - Protocol agreement for browser and server to speak HTTP
 - TCP handle network problems (drops, corruption, etc.)
 - TCP layered on top of IP/Ethernet
- Routers determine efficient path to 66.102.7.104

Finding the right IP address: Domain Naming System (DNS)

Your computer
(152.3.140.5)

Local
DNS server
(152.3.145.240)

What's the IP address for www.google.com?



Oh, you can find it at 66.102.7.104

Finding the Right Ether Address: Address Resolution (ARP)

(152.3.140.5)



(152.3.145.240)



*Broadcast: Anyone know the
Ethernet address for 152.3.145.240?*

Ethernet

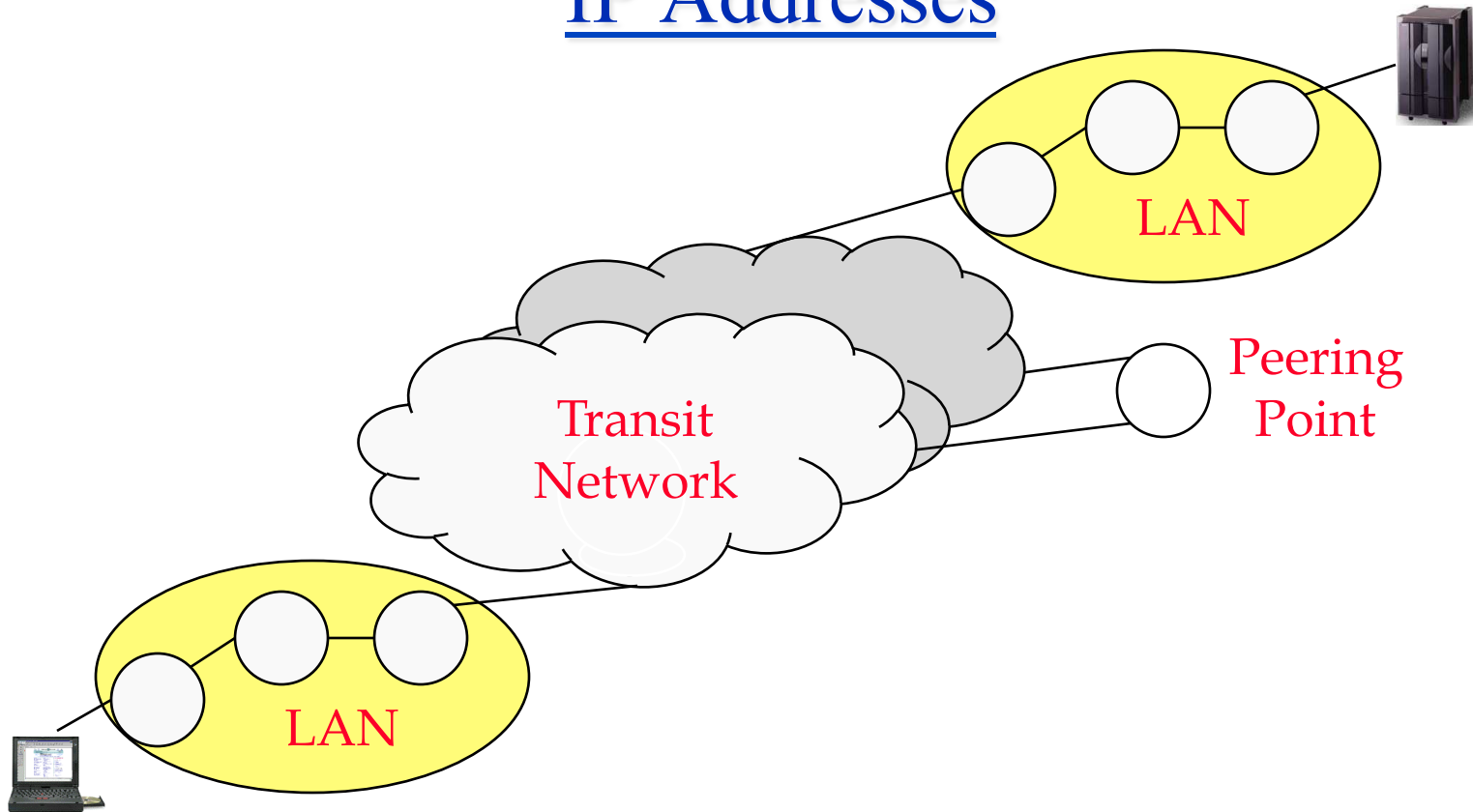


*Unicast: Yes, I'm at
08-00-2b-18-bc-65*



Ethernet

IP Addresses

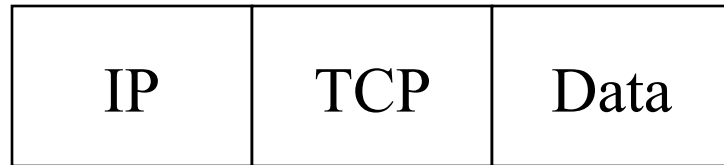
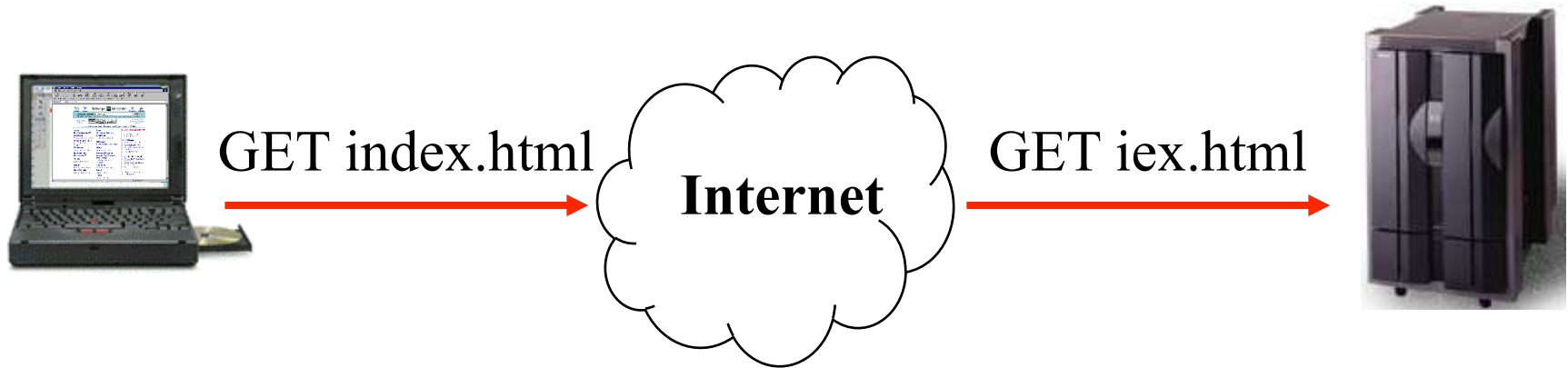


- At each step, figure out how to get closer to destination

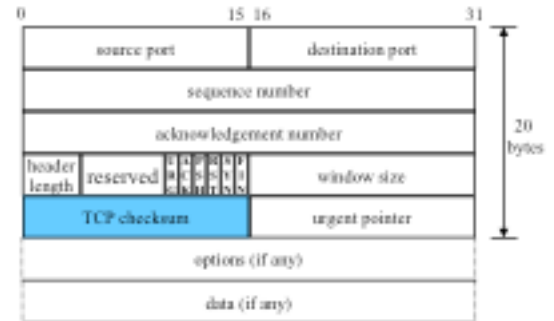
Have Address, Now Send Data

- Real networks must deal with congestion, bit errors, etc.
 - Data can be corrupted
 - Data can get lost
 - Data might not fit in a single packet
 - Data can be delivered in the wrong order
 - ...

What If the Data Gets Corrupted?

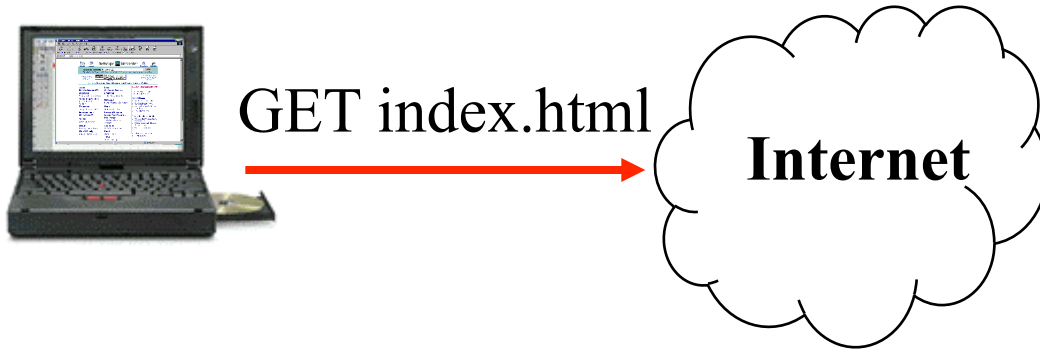


TCP Packet Format

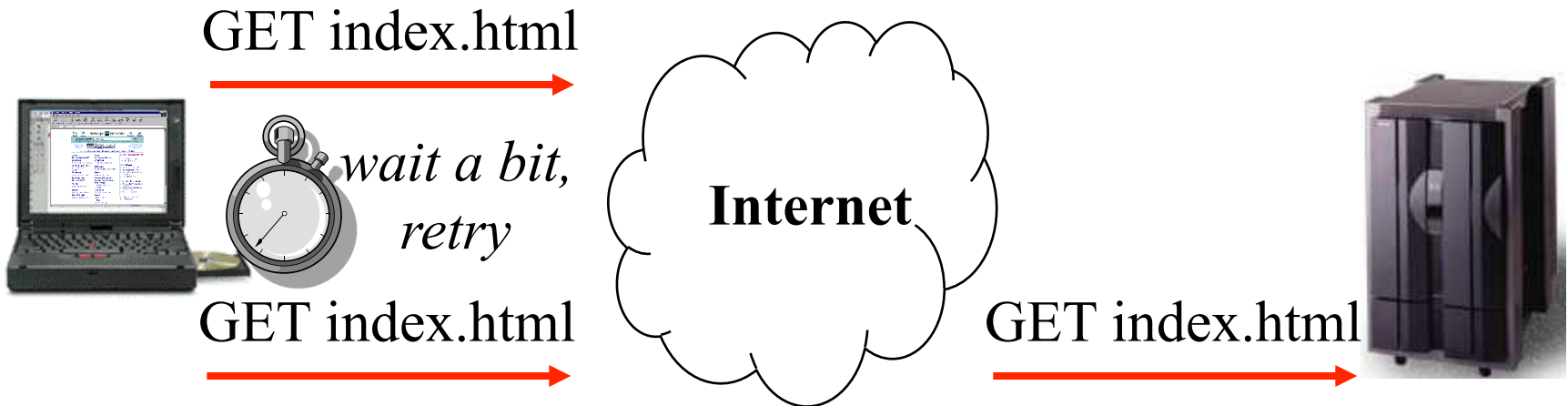


Solution: Add a *checksum*

What If the Data Gets Lost?



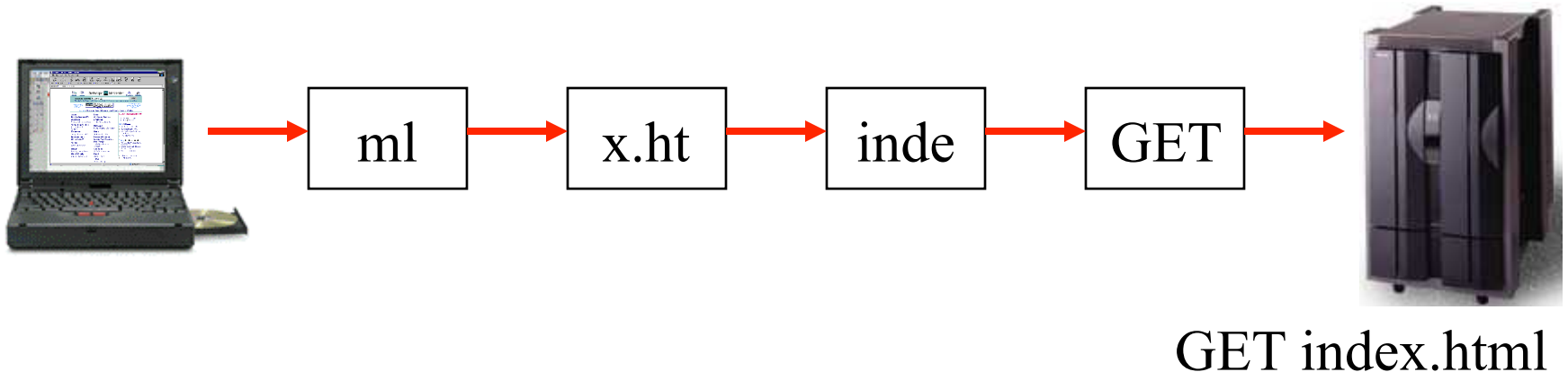
Solution: *Timeout* and *retransmit*



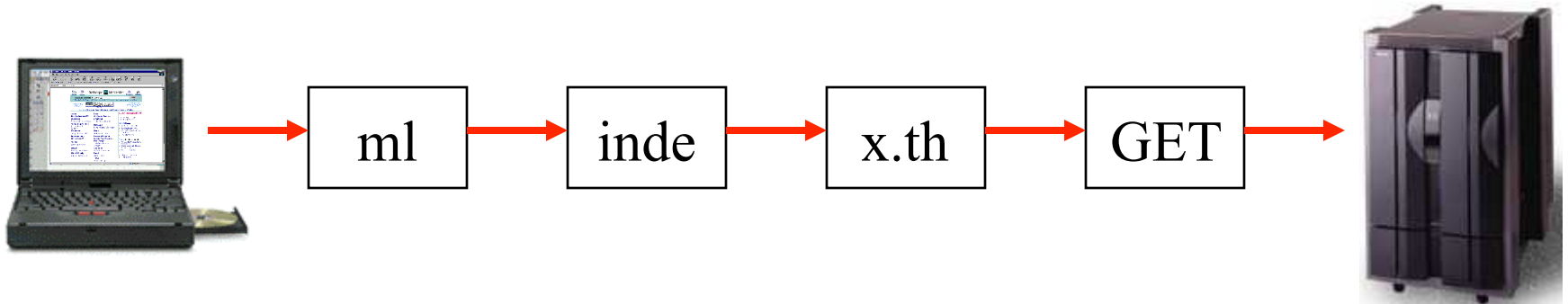
What If the Data Does Not Fit in a Single Packet?

On Ethernet, max IP packet is 1.5kbytes
Typical web page is 10kbytes

Solution: *Fragment* data across packets

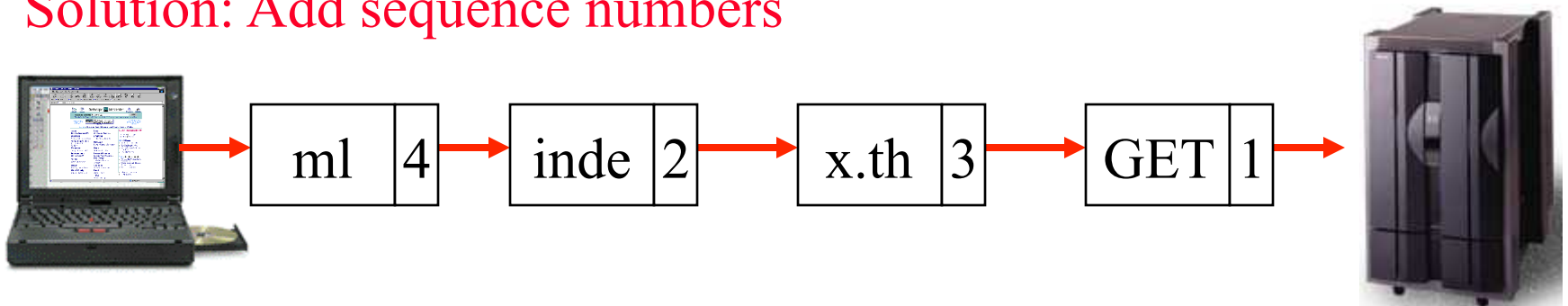


What If the Data is Out of Order?



GET x.thindexm1
???

Solution: Add sequence numbers



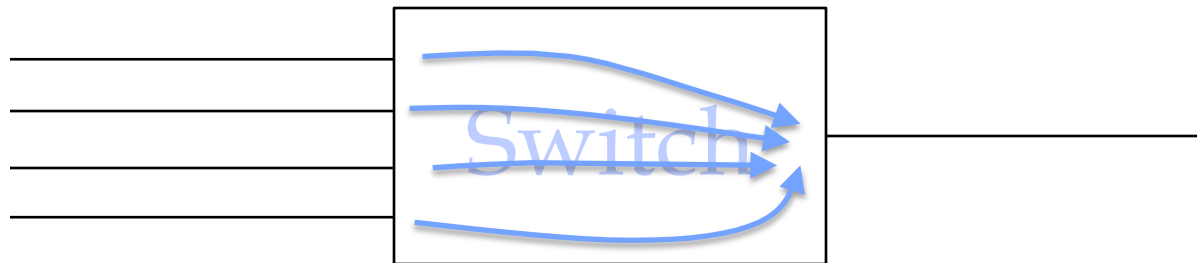
GET index.html

What If Network is Overloaded?

- Data can arrive at router faster than it can be forwarded
- How?
 1. Links can be different speeds



2. “Incast”



What If Network is Overloaded?

- Data can arrive at router faster than it can be forwarded
- Short bursts: buffer at router
- What if buffer overflows?
 - Packets dropped and retransmitted
 - Sender adjusts rate until load matches available resources
- Called “Congestion control”
 - Broadcast network (Ethernet): bus arbitration
 - Wide-area networks: typically handled by TCP
 - Necessary for global fairness, but in the interests of individuals to ignore

End-to-End Protocols

Host-to-Host vs. Process-to-Process Communication

- Until now, we have focused on delivering packets between arbitrary *hosts* connected to Internet
 - Routing protocols
 - IP best effort delivery model
 - Scalability and robustness through hierarchy and soft state
- Transition to arbitrary *processes* communicating together
 - One goal: provide illusion that all processes located on one large computer
 - Can address (*name*) and reliably communicate with any process

UDP

■ User Datagram Protocol (UDP)

- Simple demultiplexing

No guarantees about reliability, in-order delivery

- Thin veneer on top of IP adds src/dest port numbers

16 bit port number allows for identification of 65536 unique communication endpoints per host

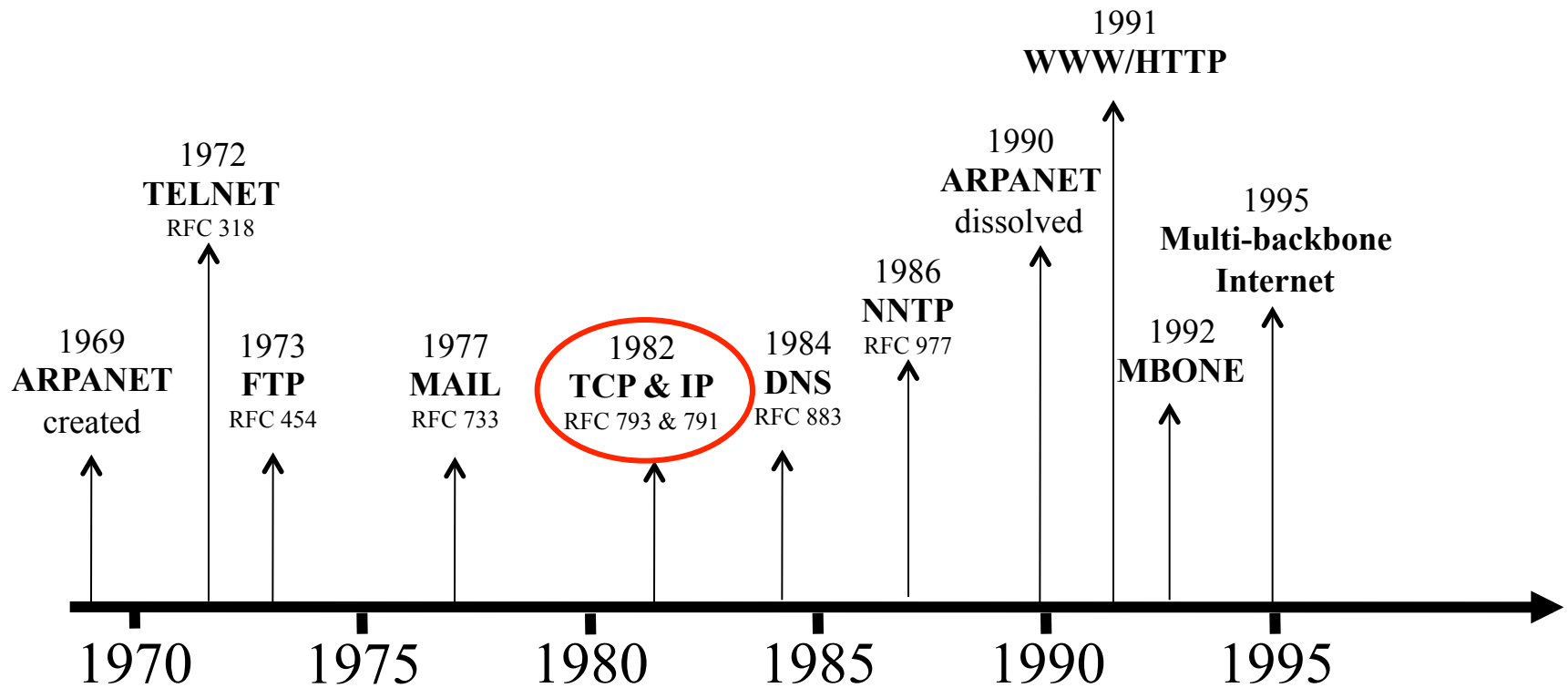
Note that a single process can utilize multiple ports

IP addr + port number uniquely identifies all Internet endpoints

- UDP Packet



A Brief Internet History



TCP

- Transmission Control Protocol (TCP)
 - Reliable in-order delivery of byte stream
 - Full duplex (endpoints simultaneously send/receive)
e.g., single socket for web browser talking to web server
 - Flow-control
To ensure that sender does not overrun *receiver*
Fast server talking to slow client
 - Congestion control
Keep the sender from overrunning the *network*
Many simultaneous connections across routers (cross traffic)

TCP Flow Control and Congestion Control

- Sender must determine maximum amount of data in transit that will not overrun *either* receiver or network
- Solutions?

TCP Flow Control and Congestion Control

- Sender must determine maximum amount of data in transit that will not overrun *either* receiver or network
- Solutions:
 - Maintain “sliding window” to track data in transit
 - Size of window determined by minimum of “flow window” and “congestion window”
 - Receiver ACKs “slide” left side of window forward

Opening up another “slot” at right side of window for transmission

