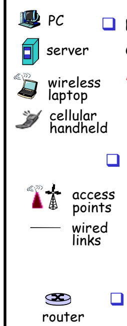
Networking Review

Goals:
- review key topics from intro networks course
  - equalize backgrounds
  - identify remedial work
  - ease into course

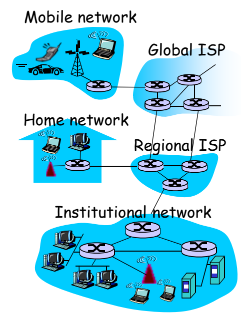
Overview:
- overview
- error control
- flow control
- congestion control
- routing
- LANs
- addressing
- synthesis:
  - control timescales

Based on slides from Prof. Jim Kurose 1

What's the Internet: "nuts and bolts" view



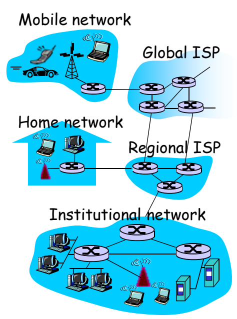
- millions of connected computing devices: *hosts = end systems*
  - running *network apps*
- *communication links*
  - fiber, copper, radio, satellite
  - transmission rate = *bandwidth*
- *routers:* forward packets (chunks of data)



2

What's the Internet: "nuts and bolts" view

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



3

What's a protocol?

human protocols:
- "what's the time?"
- "I have a question"
- introductions

… specific msgs sent

… specific actions taken when msgs received, or other events

network protocols:
- machines rather than humans
- all communication activity in Internet governed by protocols

protocols define format, order of msgs sent and received among network entities, and actions taken on msg transmission, receipt

4

Types of Communication Networks

- Traditional
  - Traditional local area network (LAN)
  - Traditional wide area network (WAN)
- Higher-speed
  - High-speed local area network (LAN)
  - Metropolitan area network (MAN)
  - High-speed wide area network (WAN)

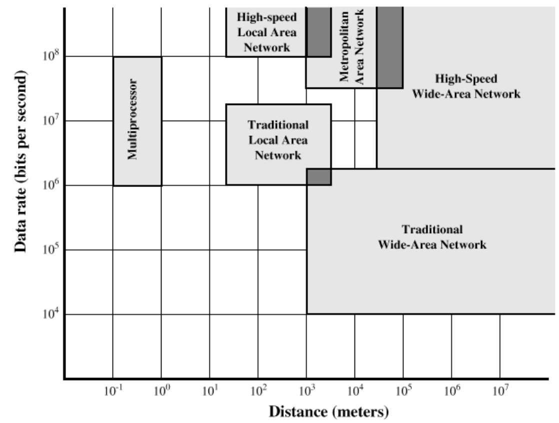


Figure 3.1 Comparison of Multiprocessor Systems, LANs, MANs, and WANs

Characteristics of WANs

- Covers large geographical areas
- Circuits provided by a common carrier
- Consists of interconnected switching nodes
- Traditional WANs provide modest capacity
  - 64000 bps common
  - Business subscribers using T-1 service – 1.544 Mbps common
- Higher-speed WANs use optical fiber and transmission technique known as asynchronous transfer mode (ATM)
  - 10s and 100s of Mbps common

Characteristics of LANs

- Like WAN, LAN interconnects a variety of devices and provides a means for information exchange among them
- Traditional LANs
  - Provide data rates of 1 to 20 Mbps
- High-speed LANS
  - Provide data rates of 100 Mbps to 1 Gbps

Differences between LANs and WANs

- Scope of a LAN is smaller
  - LAN interconnects devices within a single building or cluster of buildings
- LAN usually owned by organization that owns the attached devices
  - For WANs, most of network assets are not owned by same organization
- Internal data rate of LAN is much greater

9

The Need for MANs

- Traditional point-to-point and switched network techniques used in WANs are inadequate for growing needs of organizations
- Need for high capacity and low costs over large area
- MAN provides:
  - Service to customers in metropolitan areas
  - Required capacity
  - Lower cost and greater efficiency than equivalent service from telephone company

10

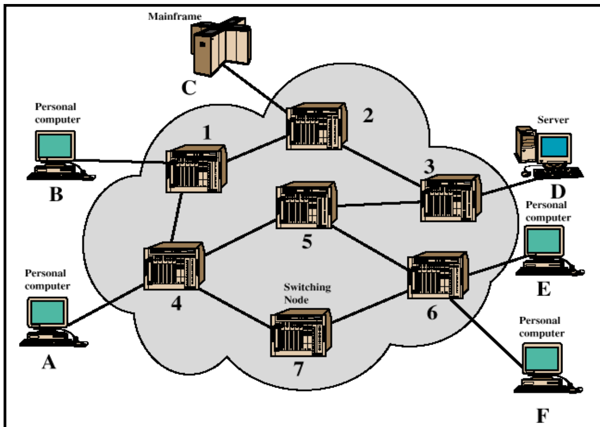


Figure 3.3 Simple Switching Network

11

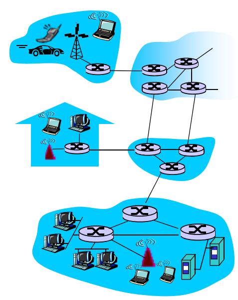
Observations of Figure 3.3

- Some nodes connect only to other nodes (e.g., 5 and 7)
- Some nodes connect to one or more stations
- Node-station links usually dedicated point-to-point links
- Node-node links usually multiplexed links
  - Frequency-division multiplexing (FDM)
  - Time-division multiplexing (TDM)
- Not a direct link between every node pair

12

A closer look at network structure:

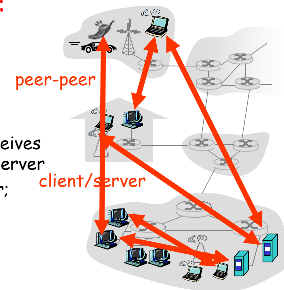
- network edge: applications and hosts
- access networks, physical media: wired, wireless communication links
- network core:
  - interconnected routers
  - network of networks



13

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-peer model:
  - minimal (or no) use of dedicated servers
  - e.g. Skype, BitTorrent



14

Network edge: reliable data transfer 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 reliable data transfer 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

15

Network edge: best effort (unreliable) data transfer service

Goal: data transfer between end systems

- same as before!

- UDP - User Datagram Protocol [RFC 768]:
  - connectionless
  - unreliable data transfer
  - no flow control
  - no congestion control

App's using TCP:

- HTTP (Web), FTP (file transfer), Telnet (remote login), SMTP (email)

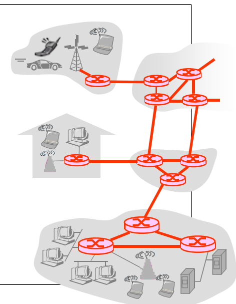
App's using UDP:

- streaming media, teleconferencing, DNS, Internet telephony

16

The Network Core

- mesh of interconnected routers
- *the* fundamental question: how is data transferred through net?
  - circuit switching: dedicated circuit per call: telephone net
  - packet-switching: data sent thru net in discrete "chunks"



17

Techniques Used in Switched Networks

- Circuit switching
  - Dedicated communications path between two stations
  - E.g., public telephone network
- Packet switching
  - Message is broken into a series of packets
  - Each node determines next leg of transmission for each packet

18

Phases of Circuit Switching

- Circuit establishment
  - An end to end circuit is established through switching nodes
- Information Transfer
  - Information transmitted through the network
  - Data may be analog voice, digitized voice, or binary data
- Circuit disconnect
  - Circuit is terminated
  - Each node deallocates dedicated resources

19

Network Core: Circuit Switching

network resources (e.g., bandwidth) divided into "pieces"

- pieces allocated to calls
- resource piece *idle* if not used by owning call *(no sharing)*

- Qiestion: how is bandwidth divided into "pieces"
  -
  -

20

Characteristics of Circuit Switching

- Can be inefficient
  - Channel capacity dedicated for duration of connection
  - Utilization not 100%
  - Delay prior to signal transfer for establishment
- Once established, network is transparent to users
- Information transmitted at fixed data rate with only propagation delay

21

Components of Public Telecommunications Network

- Subscribers - devices that attach to the network; mostly telephones
- Subscriber line - link between subscriber and network
  - Also called subscriber loop or local loop
- Exchanges - switching centers in the network
  - A switching centers that support subscribers is an end office
- Trunks - branches between exchanges

22

How Packet Switching Works

- Data is transmitted in blocks, called packets
- Before sending, the message is broken into a series of packets
  - Typical packet length is 1000 octets (bytes)
  - Packets consists of a portion of data plus a packet header that includes control information
- At each node en route, packet is received, stored briefly and passed to the next node

23

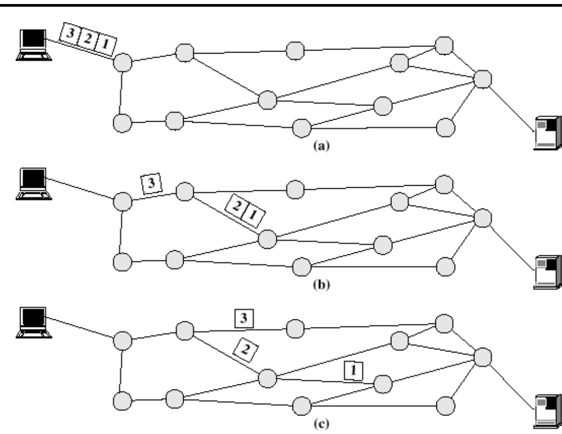


Figure 3.7 Packet Switching: Datagram Approach

24

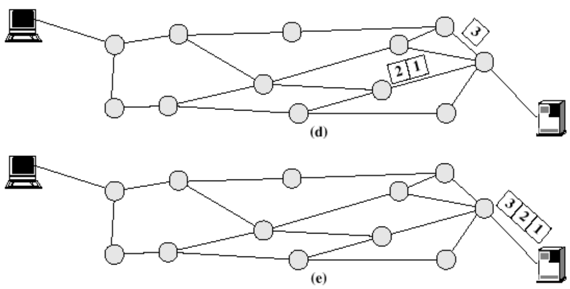


Figure 3.7 Packet Switching: Datagram Approach

Network Core: Packet Switching

each end-end data stream divided into *packets*

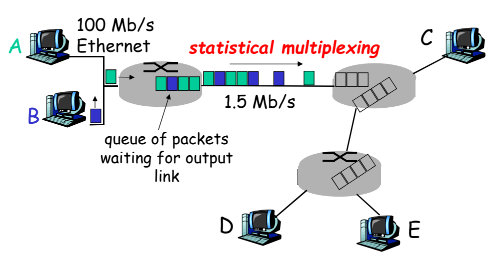
- user A, B packets *share* network resources
- each packet uses full link bandwidth
- resources used *as needed*

Bandwidth division into "pieces"
Dedicated allocation
Resource reservation

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
  - Node receives complete packet before forwarding

Packet Switching: Statistical Multiplexing



Question: why packet switching?

-
-

Packet Switching Advantages

- Line efficiency is greater
  - Many packets over time can dynamically share the same node to node link
- Packet-switching networks can carry out data-rate conversion
  - Two stations with different data rates can exchange information
- Unlike circuit-switching networks that block calls when traffic is heavy, packet-switching still accepts packets, but with increased delivery delay
- Priorities can be used

Disadvantages of Packet Switching

- Each packet switching node introduces a delay
- Overall packet delay can vary substantially
  - This is referred to as jitter
  - Caused by differing packet sizes, routes taken and varying delay in the switches
- Each packet requires overhead information
  - Includes destination and sequencing information
  - Reduces communication capacity
- More processing required at each node

Effect of Packet Size on Transmission

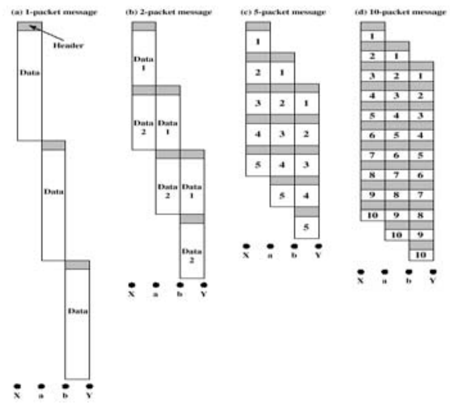


Figure 3.9 Effect of Packet Size on Transmission Time

Effect of Packet Size on Transmission

- Breaking up packets decreases transmission time because transmission is allowed to overlap
- Figure 3.9a
  - Entire message (40 octets) + header information (3 octets) sent at once
  - Transmission time: 129 octet-times
- Figure 3.9b
  - Message broken into 2 packets (20 octets) + header (3 octets)
  - Transmission time: 92 octet-times

Effect of Packet Size on Transmission

- Figure 3.9c
  - Message broken into 5 packets (8 octets) + header (3 octets)
  - Transmission time: 77 octet-times
- Figure 3.9d
  - Making the packets too small, transmission time starts increases
  - Each packet requires a fixed header; the more packets, the more headers

Packet Switching Networks - Datagram

- Each packet treated independently, without reference to previous packets
- Each node chooses next node on packet's path
- Packets don't necessarily follow same route and may arrive out of sequence
- Exit node restores packets to original order
- Responsibility of exit node or destination to detect loss of packet and how to recover

33

Packet Switching Networks – Datagram

- Advantages:
  - Call setup phase is avoided
  - Because it's more primitive, it's more flexible
  - Datagram delivery is more reliable

34

Packet Switching Networks – Virtual Circuit

- Preplanned route established before packets sent
- All packets between source and destination follow this route
- Routing decision not required by nodes for each packet
- Emulates a circuit in a circuit switching network but is not a dedicated path
  - Packets still buffered at each node and queued for output over a line

35

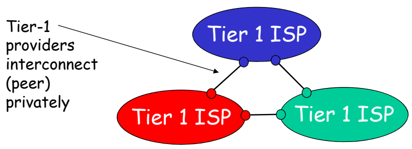
Packet Switching Networks – Virtual Circuit

- Advantages:
  - Packets arrive in original order
  - Packets arrive correctly
  - Packets transmitted more rapidly without routing decisions made at each node

36

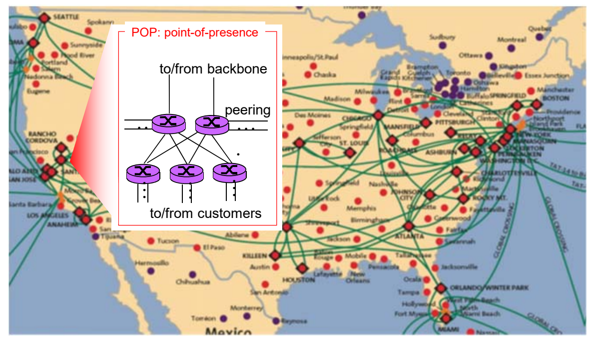
Internet structure: network of networks

- roughly hierarchical
- at center: “tier-1” ISPs (e.g., Verizon, Sprint, AT&T, Cable and Wireless), national/international coverage
  - treat each other as equals



37

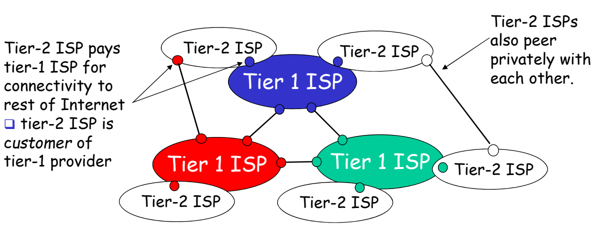
Tier-1 ISP: e.g., Sprint



38

Internet structure: network of networks

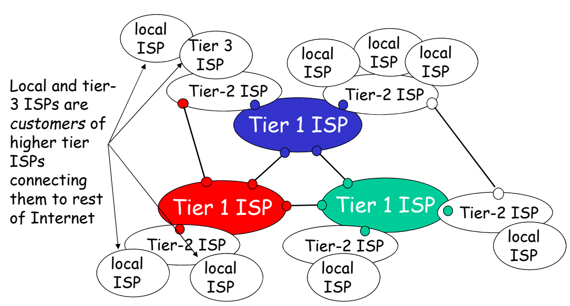
- “Tier-2” ISPs: smaller (often regional) ISPs
  - Connect to one or more tier-1 ISPs, possibly other tier-2 ISPs



39

Internet structure: network of networks

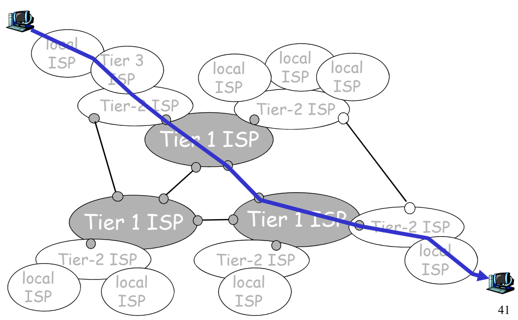
- “Tier-3” ISPs and local ISPs
  - last hop (“access”) network (closest to end systems)



40

Internet structure: network of networks

- a packet passes through many networks!



Protocol "Layers"

Networks are complex!

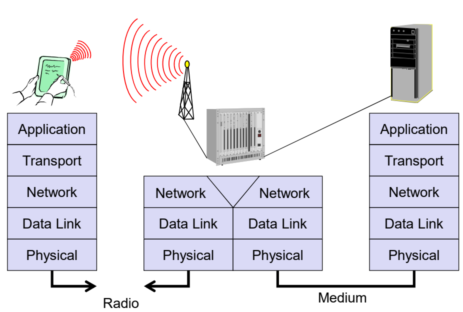
- many "pieces":
  - hosts
  - routers
  - links of various media
  - applications
  - protocols
  - hardware, software

Question:

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

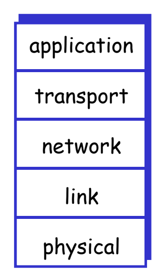
Or at least our discussion of networks?

TCP/IP Model



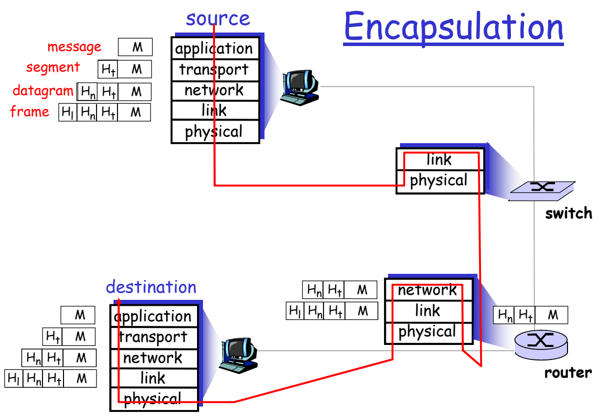
Internet protocol stack

- application: supporting network applications (FTP, SMTP, HTTP)
- transport: process-process 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"



Question: anything missing?

Encapsulation



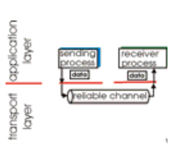
Networking Review

Goals:
- review key topics from intro networks course
  - equalize backgrounds
  - identify remedial work
  - ease into course

Overview:
- overview
- error control
- flow control
- congestion control
- routing
- LANs
- addressing
- synthesis:
  - control timescales

Error control

- reliable point-point communication
  - generic problem: app-to-app, over path, over link
- error model?
  - bits flipped in packet
  - packets "lost
  - packets delayed or reordered

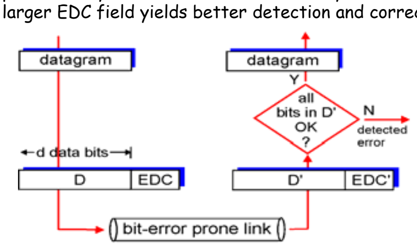


provided service

Bit level error detection

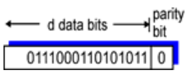
EDC= Error Detection and Correction bits (redundancy)
D = Data protected by error checking, may include header fields

- Error detection not 100% reliable!
  - protocol may miss some errors, but rarely
  - larger EDC field yields better detection and correction

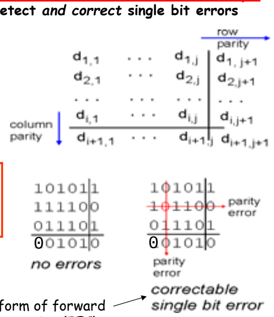


Parity Checking

Single Bit Parity:
Detect single bit errors



Two Dimensional Bit Parity:
Detect *and correct* single bit errors



Much more powerful error detection/correction schemes: Cyclic Redundancy Check (CRC)

Simple form of forward error correction (FEC)

Internet checksum

Goal: detect "errors" (e.g., flipped bits) in transmitted segment (note: used at transport layer *only*)

Sender:
- treat segment contents as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into segment checksum field

Receiver:
- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - NO - error detected
  - YES - no error detected. *But maybe errors nonetheless?*

Recovering from lost packets

- why are packets lost?
  - limited storage, discarded in congestion
  - outages: eventually reroute around failure (~sec recovery times hopefully)
  - dropped at end system e.g., on NIC
- ARQ: automatic request repeat
  - sender puts sequence numbers on packets (why)
  - receiver positively or negatively acknowledges correct receipt of packet
  - sender starts (logical) timer for each packet, timeout and retransmits

Reference: section 3.4 in K&R

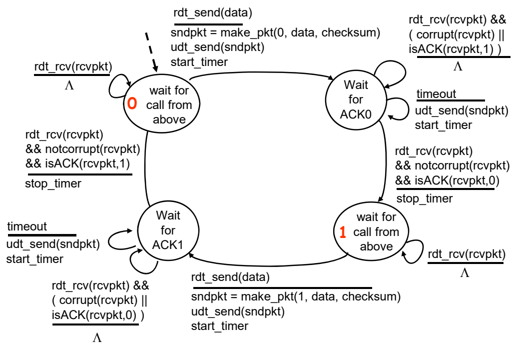
rdt3.0: channels with errors *and* loss

Assumption: underlying channel can corrupt, lose packets (data or ACKs)
- need checksum, seq. #, ACKs, retransmissions, timer

- seq #s
  - detect reordering
  - ACK, NAKing
  - detect missing packet
  - duplicate detection due to retransmissions

Approach: sender waits "reasonable" amount of time for ACK
- retransmits if no ACK received in this time
- if pkt (or ACK) just delayed (not lost):
  - retransmission will be duplicate, but use of 0,1 seq. #'s already handles this
  - receiver must specify seq # of pkt being ACKed
- requires countdown timer

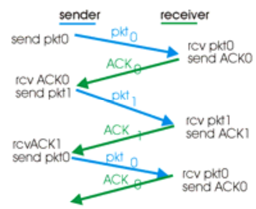
rdt3.0 sender



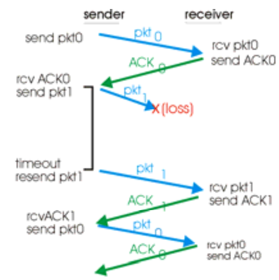
FSM specification of sender (details not important)

53

rdt3.0 in action



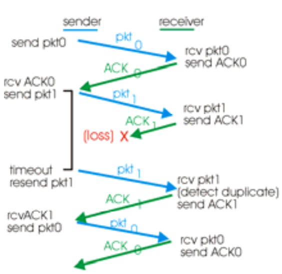
(a) operation with no loss



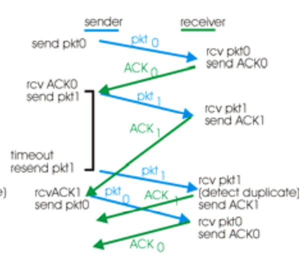
(b) lost packet

54

rdt3.0 in action



(c) lost ACK

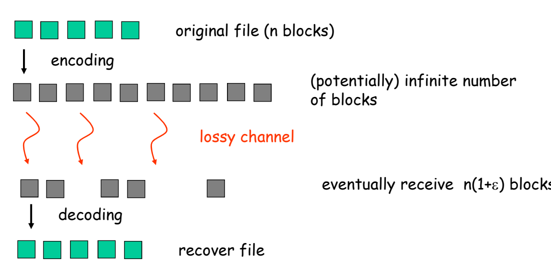


(d) premature timeout

55

Forward error control

- add redundancy to recover from losses



56

Forward error control

- rateless codes allow infinite code blocks
  - LT/Raptor codes
- ε controls computation cost, BW usage
- used for video delivery; large file transfers

57

Networking Review

Goals:
- review key topics from intro networks course
  - equalize backgrounds
  - identify remedial work
  - ease into course

Overview:
- overview
- error control
- flow control
- congestion control
- routing
- LANs
- addressing
- synthesis:
  - "a day in the life"
  - control timescales

58

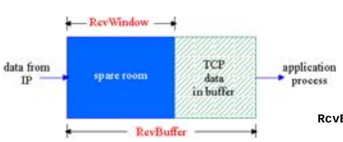
Flow Control (in TCP)

flow control

sender won't overrun receiver's buffers by transmitting too much, too fast

receiver: explicitly informs sender of (dynamically changing) amount of free buffer space
- **RcvWindow field** in TCP segment

sender: keeps the amount of transmitted, unACKed data less than most recently received **RcvWindow**



receiver buffering

`RcvBuffer` = size of TCP Receive Buffer

`RcvWindow` = amount of spare room in Buffer

59

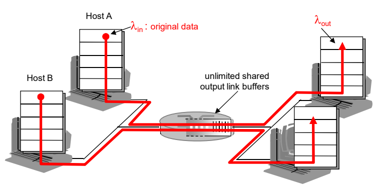
Principles of Congestion Control

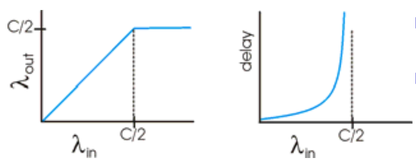
Congestion:
- informally: "too many sources sending too much data too fast for *network* to handle"
- different from flow control!
- manifestations:
  - lost packets (buffer overflow at routers)
  - long delays (queueing in router buffers)

60

Causes/costs of congestion: scenario 1

- two senders, two receivers
- one router, infinite buffers
- no retransmission



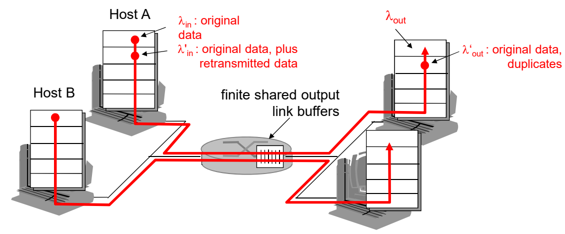


- large delays when congested
- maximum achievable throughput

61

Causes/costs of congestion: scenario 2

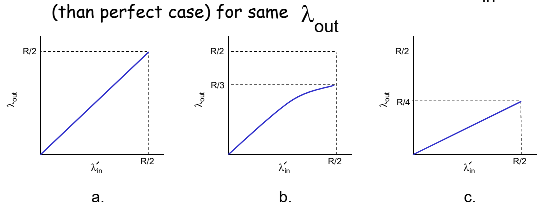
- one router, *finite* buffers
- sender retransmission of lost packet



62

Causes/costs of congestion: scenario 2

- always: $\lambda_{in} = \lambda_{out}$ (goodput)
- "perfect" retransmission only when loss: $\lambda'_{in} > \lambda_{out}$
- retransmission of delayed (not lost) packet makes λ'_{in} larger (than perfect case) for same λ_{out}



"costs" of congestion:

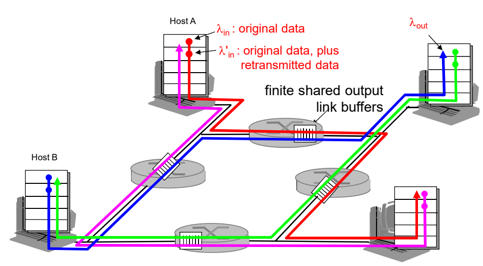
- more work (retrans) for given "goodput"
- unneeded retransmissions: link carries multiple copies of pkt

63

Causes/costs of congestion: scenario 3

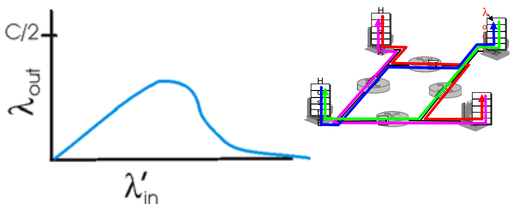
- four senders
- multihop paths
- timeout/retransmit

Q: what happens as λ_{in} and λ'_{in} increase ?



64

Causes/costs of congestion: scenario 3



Another "cost" of congestion:

- when packet dropped, any "upstream transmission capacity used for that packet was wasted!

65

Approaches towards congestion control

Two broad approaches towards congestion control:

End-end congestion control:

- no explicit feedback from network
- congestion inferred from end-system observed loss, delay
- approach taken by TCP

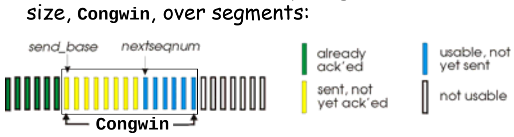
Network-assisted congestion control:

- routers provide feedback to end systems
  - single bit indicating congestion (SNA, DECbit, TCP/IP ECN, ATM)
  - explicit rate sender should send at

66

TCP Congestion Control

- end-end control (no network assistance)
- transmission rate limited by congestion window size, **Congwin**, over segments:



67

TCP congestion control:

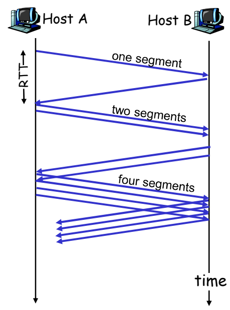
- "probing" for usable bandwidth:
  - ideally: transmit as fast as possible (**Congwin** as large as possible) without loss
  - *increase* **Congwin** until loss (congestion)
  - loss: *decrease* **Congwin**, then begin probing (increasing) again
- two "phases"
  - slow start
  - congestion avoidance
- important variables:
  - **Congwin**
  - **threshold:** defines threshold between two slow start phase, congestion control phase

68

TCP Slowstart

Slowstart algorithm

initialize: Congwin = 1
for (each segment ACKed)
    Congwin++
until (loss event OR
    CongWin > threshold)

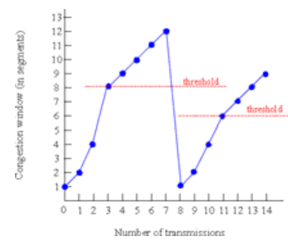


- exponential increase (per RTT) in window size (not so slow!)
- loss event: timeout (Tahoe TCP) and/or or three duplicate ACKs (Reno TCP)

TCP Congestion Avoidance: Tahoe

TCP Tahoe Congestion avoidance

/* slowstart is over */
/* Congwin > threshold */
Until (loss event) {
  every Congwin segments ACKed: Congwin++
  }
threshold = Congwin/2
Congwin = 1
perform slowstart



Numerous improvements: TCP Reno, SACK

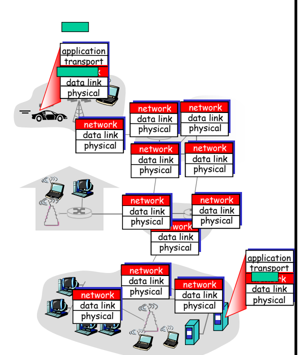
Part 0: Networking Review

Goals:
- review key topics from intro networks course
  - equalize backgrounds
  - identify remedial work
  - ease into course

Overview:
- overview
- error control
- flow control
- congestion control
- routing (and network layer services)
- LANs
- addressing
- synthesis:
  - control timescales

Network layer

- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on rcving side, delivers segments to transport layer
- network layer protocols in *every* host, router
- router examines header fields in all IP datagrams passing through it



Two Key Network-Layer Functions

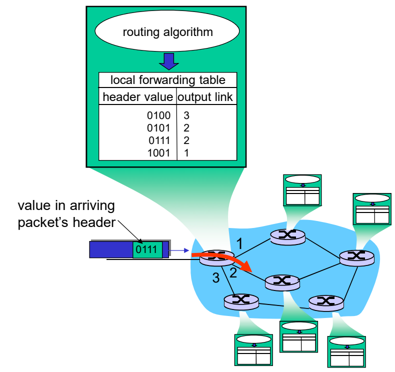
- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to dest.
  - *routing algorithms*

analogy:

- routing: process of planning trip from source to dest
- forwarding: process of getting through single interchange

73

Interplay between routing and forwarding



74

Network service model

CRUCIAL question!

Q: What *service model* for "channel" transporting packets from sender to receiver?

service abstraction

- guaranteed bandwidth?
- preservation of inter-packet timing (no jitter)?
- loss-free delivery?
- in-order delivery?
- congestion feedback to sender?

The most important abstraction provided by network layer:

virtual circuit
or
datagram?

75

Virtual circuits

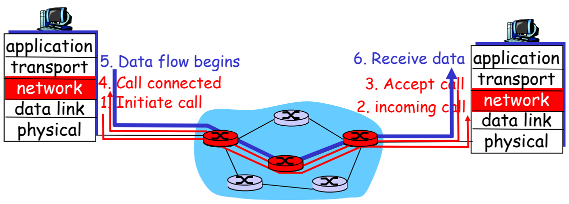
"source-to-dest path behaves much like telephone circuit"
- performance-wise
- network actions along source-to-dest path

- call setup, teardown for each call *before* data can flow
- each packet carries VC identifier (not destination host ID)
- *every* router on source-dest path maintains "state" for each passing connection
  - transport-layer connection only involved two end systems
- link, router resources (bandwidth, buffers) may be *allocated* to VC
  - to get circuit-like perf.

76

Virtual circuits: signaling protocols

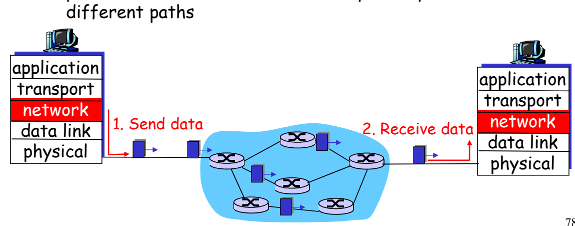
- used to set up, maintain teardown VC
- used in ATM, frame-relay, X.25
- not used in today's Internet



77

Datagram networks: the Internet model

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets typically routed using destination host ID
  - packets between same source-dest pair may take different paths



78

Datagram or VC network: why?

Internet

- data exchange among computers
  - "elastic" service, no strict timing req.
- "smart" end systems (computers)
  - can adapt, perform control, error recovery
  - simple inside network, complexity at "edge"
- many link types
  - different characteristics
  - uniform service difficult

ATM

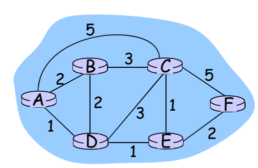
- evolved from telephony
- human conversation:
  - strict timing, reliability requirements
  - need for guaranteed service
- "dumb" end systems
  - telephones
  - complexity inside network

79

Routing

Routing protocol

Goal: determine "good" path (sequence of routers) thru network from source to dest.



Graph abstraction for routing algorithms:

- graph nodes are routers
- graph edges are physical links
  - link cost: delay, $ cost, or congestion level

- "good" path:
  - typically means minimum cost path
  - other def's possible

80

Routing: only two approaches used in practice

Global:
- all routers have complete topology, link cost info
- "link state" algorithms: use Dijkstra's algorithm to find shortest path from given router to all destinations

Decentralized:
- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms
- a 'self-stabilizing algorithm' (we'll see these later)

Distance Vector Routing Algorithm

iterative:
- continues until no nodes exchange info.
- *self-terminating*: no "signal" to stop

asynchronous:
- nodes need *not* exchange info/iterate in lock step!

distributed:
- each node communicates *only* with directly-attached neighbors

Each node:

wait for (change in local link cost of msg from neighbor)

recompute distance table

if least cost path to any dest has changed, *notify* neighbors

Hierarchical Routing

Our routing review thus far - idealization
- all routers identical
- network "flat"

… *not* true in practice

scale: with 200 million destinations:
- can't store all dest's in routing tables!
- routing table exchange would swamp links!

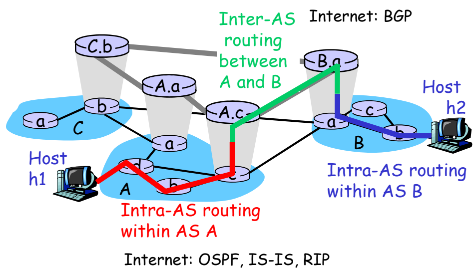
administrative autonomy
- internet = network of networks
- each network admin may want to control routing in its own network

Hierarchical Routing

- aggregate routers into regions, "autonomous systems" (AS)
- routers in same AS run same routing protocol
  - "intra-AS" routing protocol
  - routers in different AS can run different intra-AS routing protocol

gateway routers
- special routers in AS
- run intra-AS routing protocol with all other routers in AS
- *also* responsible for routing to destinations outside AS
  - run *inter-AS routing* protocol with other gateway routers

Intra-AS and Inter-AS routing

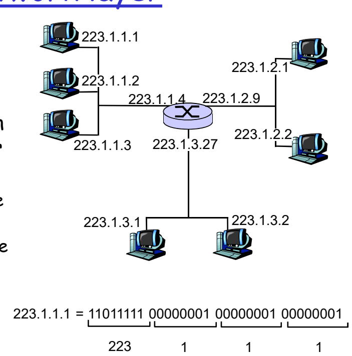


Addressing

- what's an address?
  - identifier that differentiates between me and someone else, and also helps route data to/from me
- real world examples of addressing?
  - mailing address
  - office #, floor, etc
  - phone

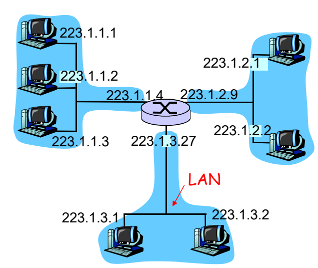
Addressing: network layer

- *IP address:* 32-bit identifier for host, router *interface*
- *interface:* connection between host, router and physical link
  - router's typically have multiple interfaces
  - host may have multiple interfaces
  - IP addresses associated with interface, not host, router



IP Addressing

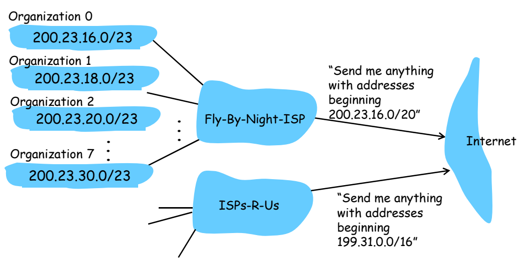
- *IP address:*
  - network part (high order bits)
  - host part (low order bits)
- *what's a network ?*
  (from IP address perspective)
  - device interfaces with same network part of IP address
  - can physically reach each other without intervening router



network consisting of 3 IP networks (for IP addresses starting with 223, first 24 bits are network address)

Hierarchical addressing: route aggregation

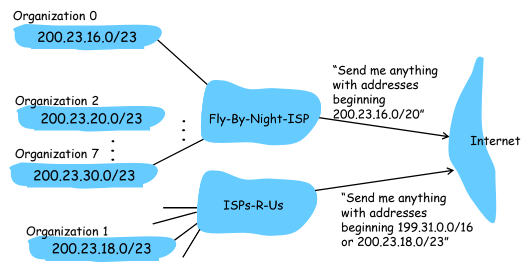
Hierarchical addressing allows efficient advertisement of routing information:



89

Hierarchical addressing: more specific routes

ISPs-R-Us has a more specific route to Organization 1



90

IP addresses: how to get one?

Q: How does host get IP address?

- hard-coded by system admin in a file
  - Wintel: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address: "plug-and-play"
  - host broadcasts "DHCP discover" msg
  - DHCP server responds with "DHCP offer" msg
  - host requests IP address: "DHCP request" msg
  - DHCP server sends address: "DHCP ack" msg

91

Part 0: Networking Review

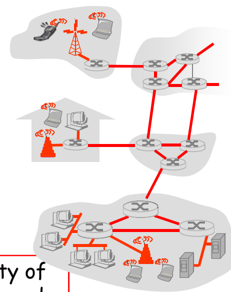
Goals:
- review key topics from intro networks course
  - equalize backgrounds
  - identify remedial work
  - ease into course

Overview:
- overview
- error control
- flow control
- congestion control
- routing
- LANs
- addressing (cont.)
- synthesis:
  - control timescales

92

Link Layer: Introduction

Some terminology:
- hosts and routers are **nodes**
- communication channels that connect adjacent nodes along communication path are **links**
  - wired links
  - wireless links
  - LANs
- layer-2 packet is a **frame**, encapsulates datagram

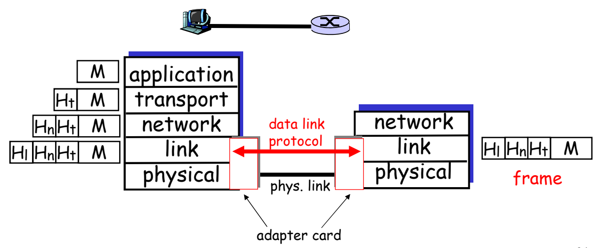


data-link layer has responsibility of transferring datagram from one node to adjacent node over a link

93

Link Layer: setting the context

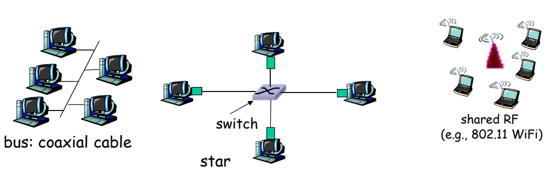
- two *physically connected* devices:
  - host-router, router-router, host-host
- unit of data: *frame*



94

LANs

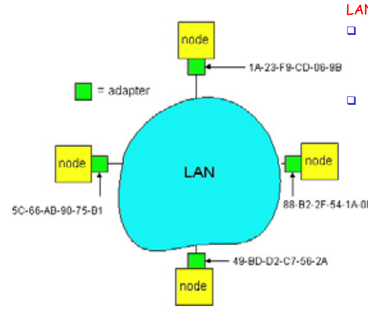
- bus topology popular through mid 90s
- today: star topology prevails
  - active *switch* in center, each "spoke" runs a (separate) Ethernet protocol
- wireless LANS: 802.11



95

LAN Addresses

Each adapter on LAN has unique LAN address (also has an IP address)



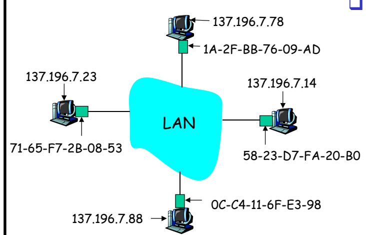
LAN (or MAC or physical) address:
- used to get datagram from one interface to another physically-connected interface (same network)
- 48 bit MAC address (for most LANs) burned in the adapter ROM

Question: why separate MAC and IP addresses?

96

ARP: Address Resolution Protocol

Question: how to determine MAC address of B knowing B's IP address?



- Each IP node (host, router) on LAN has ARP table
- ARP table: IP/MAC address mappings for some LAN nodes

  < IP address; MAC address; TTL>

  - TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

97

ARP protocol: Same LAN (network)

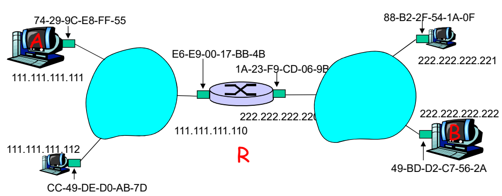
- A wants to send datagram to B, and B's MAC address not in A's ARP table.
- A broadcasts ARP query packet, containing B's IP address
  - dest MAC address = FF-FF-FF-FF-FF-FF
  - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) MAC address
  - frame sent to A's MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play":
  - nodes create their ARP tables *without intervention from net administrator*

98

Addressing: routing to another LAN

walkthrough: send datagram from A to B via R

assume A knows B's IP address

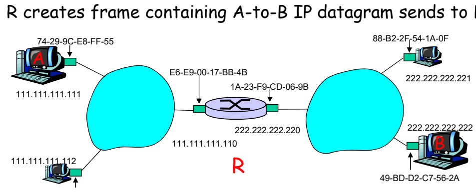


- two ARP tables in router R, one for each IP network (LAN)

99

- A creates IP datagram with source A, destination B
- A uses ARP to get R's MAC address for 111.111.111.110
- A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram
- A's NIC sends frame
- R's NIC receives frame
- R removes IP datagram from Ethernet frame, sees its destined to B
- R uses ARP to get B's MAC address
- R creates frame containing A-to-B IP datagram sends to B

This is a really important example – make sure you understand!



100

Part 0: Networking Review

Goals:
- review key topics from intro networks course
  - equalize backgrounds
  - identify remedial work
  - ease into course

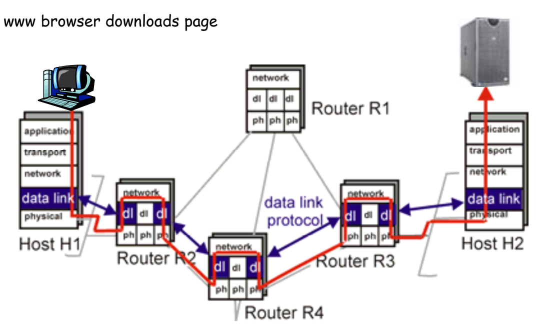
Overview:
- overview
- error control
- flow control
- congestion control
- routing
- LANs
- addressing (cont.)
- synthesis:
  - control timescales

What are the different control timescales in a network?

- transport layer
- network layer
- link layer
- other important timescales?

Synthesis: which protocols involved?

www browser downloads page



Protocols involved in http GET

- user types in a URL, what happens?
- DNS: translate hostname to IP address
  - via DHCP, source has IP address of DNS server (suppose DNS server on same network segment)
  - create DNS query, pass to UDP, create UDP segment containing DNS query, pass to IP on host
  - look in routing table (DHCP gave me default router), recognize that DNS server on same network.
  - use ARP to determine MAC address of DNS server
  - Ethernet used to send frame to DNS server on physically connected "wire" (network segment, ethernet "cable")
  - on DNS machine ethernet->IP->UDP. UDP looks at dest port #, sees it is DNS, passes DNS query to DNS application. (assume DNS knows IP addresses of hostname in original URL - address found!)
  - DNS server sends UDP reply back to orginating machine

Protocols involved in http GET

- browser now has IP address of GET destination server
- need to establish TCP connection to server, send SYN packet (will get an SYNACK back, eventuallly….)
- SYN packet down to network layer, with IP address of server. Since server destined "off my network", SYN packet goes through router.
- look in routing table, see that destination off network, need to send to "default gateway" (to get off my net)
- use ARP to get MAC address of default gateway, create Ethernet frame with gateway MAC address, containing IP packet containing TCP segment, containing SYN
- IMPORTANT to realize that while the Ethernet frame containing the IP datagram that contains the TCP SYN has as its destination address the MAC address of the router, the IP datagram (still) has as destination address the IP address of the remote www server

105

Protocols involved in http GET

- Router receives Ethernet frame (frame addressed to router), looks at IP datagram, sees that IP datagram not addressed to itself (IP datagram addressed to server). Router knows it must forward IP datagram to next hop router along path to eventual destination.
- Router checks routing tables (table values populated using intra, possibly inter-, domain routing protocols like OSPF, RIP, IS-IS, BGP (inter). Get IP address of next hop router.
- Router puts IP packets in Ethernet frame, Ethernet frame addressed to next hop router. MAC address of next hop router determined by ARP. Frame sent to next hop router.
- Network management shoehorn: arriving packets at interface cause SNMP MIB variable for # arriving IP datagrams to be incremented
- Forwarding continues until IP datagram containing TCP SYN eventually arrives at destination, gaia.cs.umass.edu (128.119.30.30)
- Up to IP, demultiplex from Ethernet to IP using Ethernet TYPE field to identify IP as upper layer protocol
- From IP to TCP using protocol field of IP datagram,
- SYN packet arrives at gaia TCP (FINALLY)

106

Protocols involved in http GET

- So …. SYN has arrived at gaia. Gaia returns SYNACK to initial sender
- Gaia gets synack, ready to send data.
- HTTP GET message now sent to gaia.cs.umass.edu in TCP segment, in IP datagram, in Ethernet frame, along hops to gaia.cs.umass.edu
- GET arrives! REPLY formulated by http server … and sent

107