Wireless Networks
(CSC-7602)
Lecture 2
(24 Sept. 2007)

Seung-Jong Park (Jay)
http://www.csc.lsu.edu/~sjpark

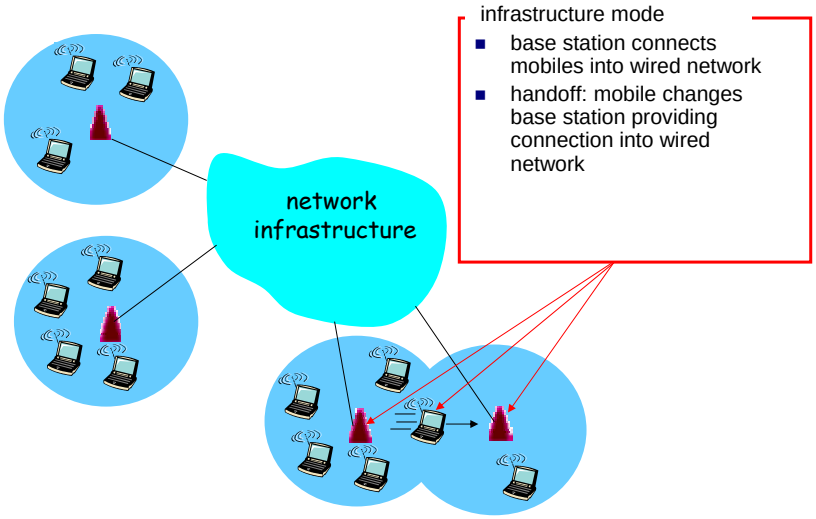
1

Wireless Technologies

Classification (1)

- View of Infrastructure/-less
  - Infrastructure
    - Cellular networks
    - Satellite networks
    - WLAN
  - Infrastructureless
    - Mobile ad-hoc networks (MANET)
    - Sensor networks
  - Hybrid
    - Wireless mesh networks

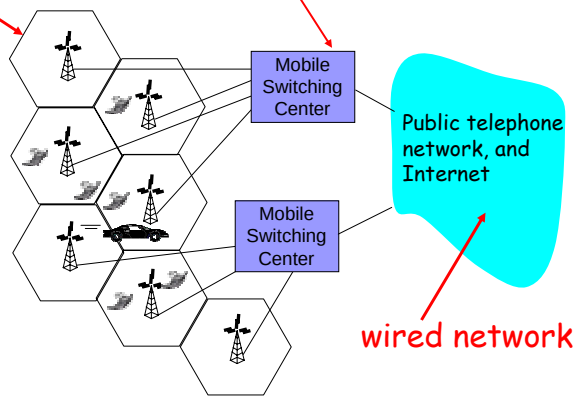
Infrastructure



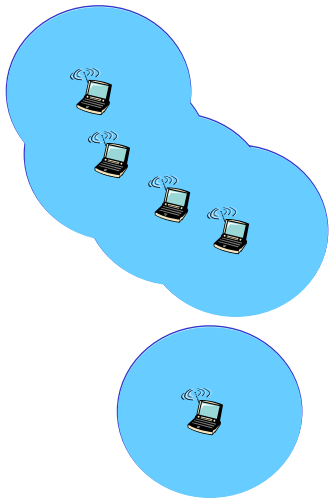
Cellular network architecture

MSC
- connects cells to wide area net
- manages call setup (more later!)
- handles mobility (more later!)

cell
- covers geographical region
- *base station* (BS) analogous to 802.11 AP
- *mobile users* attach to network through BS
- *air-interface:* physical and link layer protocol between mobile and BS



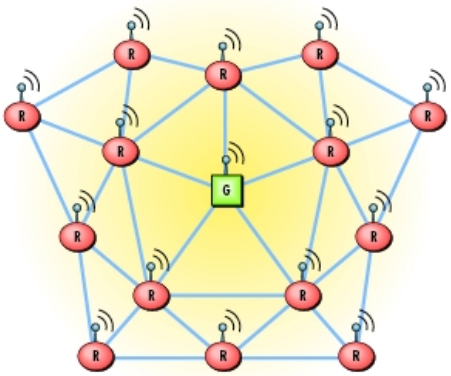
Infrastructureless



Ad hoc mode
- no base stations
- nodes can only transmit to other nodes within link coverage
- nodes organize themselves into a network: route among themselves

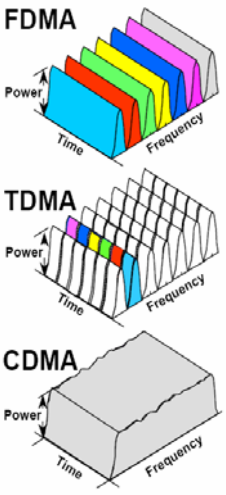
Wireless Mesh Networks

- Wireless Mesh Network
  - Enable integration of various existing networks such as Wi-Fi, the Internet, cellular and sensor networks through gateway/bridge functionalities in the mesh routers

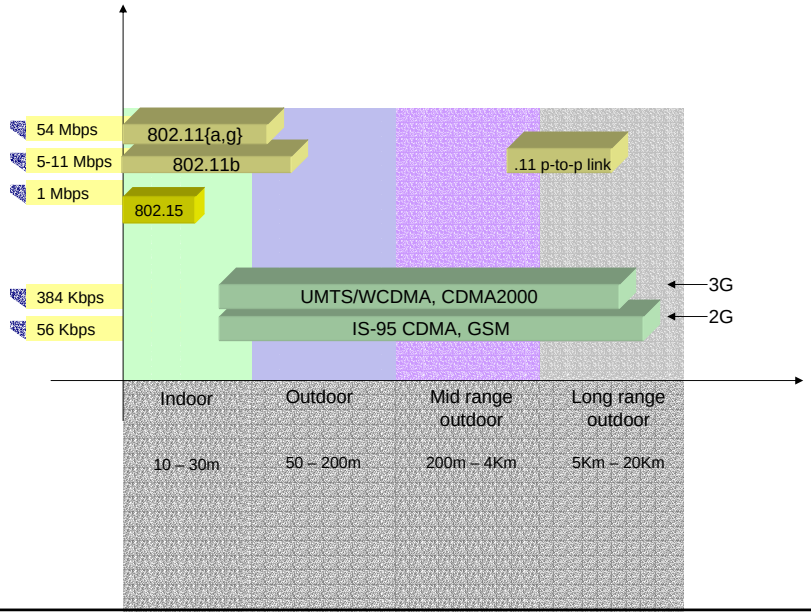


Classification (2)

- View of multiple access technologies at MAC layer
  - FDMA (Frequency Division Multiple Access)
    - Each user has a private frequency
    - AMPS
  - TDMA (Time Division Multiple Access)
    - Each user has a private time on a private frequency
    - GSM, IS-54
  - CDMA (Code Division Multiple Access)
    - Users share time and frequency with a private code
    - IS-95



Characteristics of selected wireless link standards



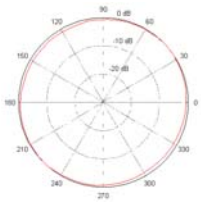
Wireless LAN

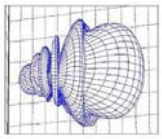
- 802.11b:
  - Most common wireless protocol.
  - Uses 2.4GHz frequency, with 11 Mbps bandwidth. (5 Mbps is more typical).
- 802.11a:
  - Uses 5.5GHz range, 54 Mbps bandwidth (~20 Mbps is typical performance).
  - Produces to much radio power to be certified in medical areas.
- 802.11g:
  - Uses 2.4GHz band and is compatible with 802.11b.
  - Also 54 Mbps bandwidth (~20 Mbps typical).

Characteristics of Wireless Communication

Wireless Transmission

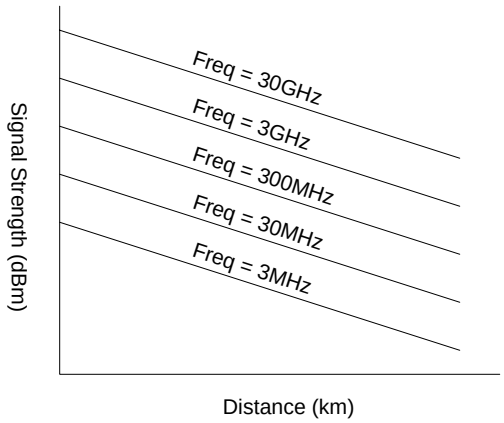
- Unguided media
- Transmission and reception via antenna
- Omnidirectional
  - Signal spreads in all directions
  - Can be received by many antennae
- Directional
  - Focused beam
  - Careful alignment required





Wireless Link Characteristics

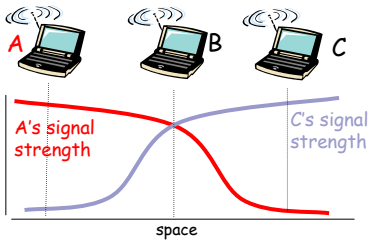
Free Space Loss

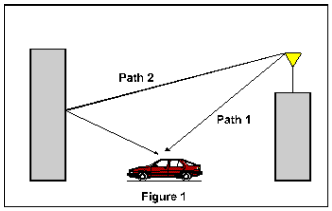


Wireless Link Characteristics

Differences from wired link ….

- interference from other sources: standardized wireless network frequencies (e.g., 2.4 GHz) shared by other devices (e.g., phone); devices (motors) interfere as well
- multipath propagation: radio signal reflects off objects ground, arriving ad destination at slightly different times

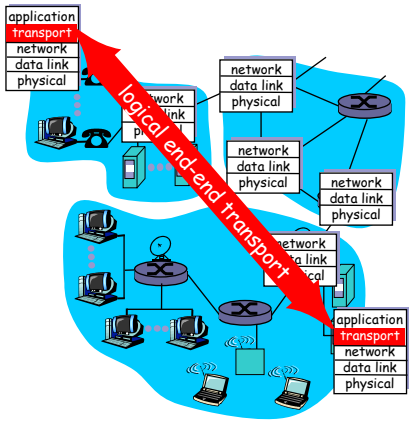




Overview
TCP

Transport services and protocols

- provide *logical communication* between app processes running on different hosts
- transport protocols run in end systems
  - send side: breaks app messages into segments, passes to network layer
  - rcv side: reassembles segments into messages, passes to app layer
- more than one transport protocol available to apps
  - Internet: TCP and UDP



Transport vs. network layer

- *network layer:* logical communication between hosts
- *transport layer:* logical communication between processes
  - relies on, enhances, network layer services

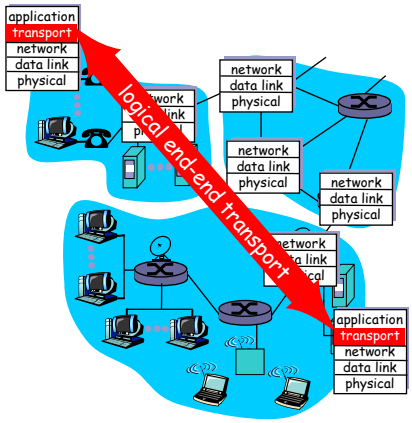
Household analogy:

12 kids sending letters to 12 kids

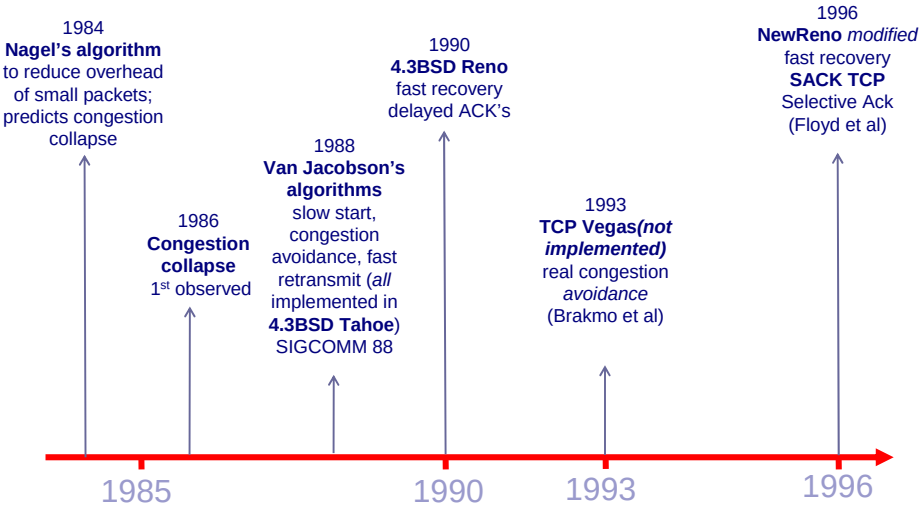
- processes = kids
- app messages = letters in envelopes
- hosts = houses
- transport protocol = Ann and Bill
- network-layer protocol = postal service

Internet transport-layer protocols

- reliable, in-order delivery (TCP)
  - congestion control
  - flow control
  - connection setup
- unreliable, unordered delivery: UDP
  - no-frills extension of "best-effort" IP
- services not available:
  - delay guarantees
  - bandwidth guarantees



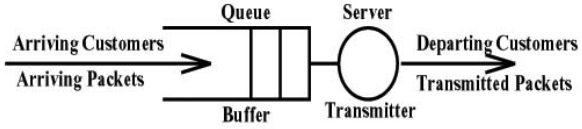
Evolution of TCP



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)
- a top-10 problem!



TCP Congestion Control

- end-end control (no network assistance)
- sender limits transmission:
  `LastByteSent-LastByteAcked`
  `≤ CongWin`
- Roughly,

$$\text{rate} = \frac{\text{CongWin}}{\text{RTT}} \text{ Bytes/sec}$$

- **`CongWin`** is dynamic, function of perceived network congestion

How does sender perceive congestion?

- loss event = timeout *or* 3 duplicate acks
- TCP sender reduces rate (**`CongWin`**) after loss event

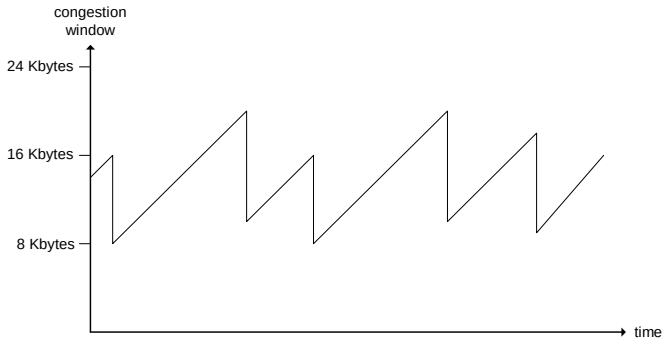
three mechanisms:

- AIMD
- slow start
- conservative after timeout events

TCP AIMD

multiplicative decrease: cut **`CongWin`** in half after loss event

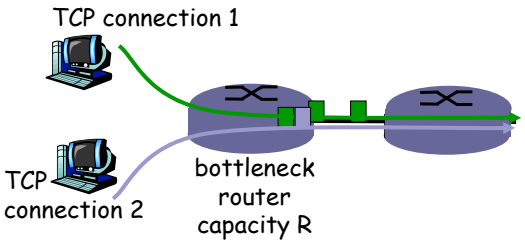
additive increase: increase **`CongWin`** by 1 MSS every RTT in the absence of loss events: *probing*



Long-lived TCP connection

TCP Fairness

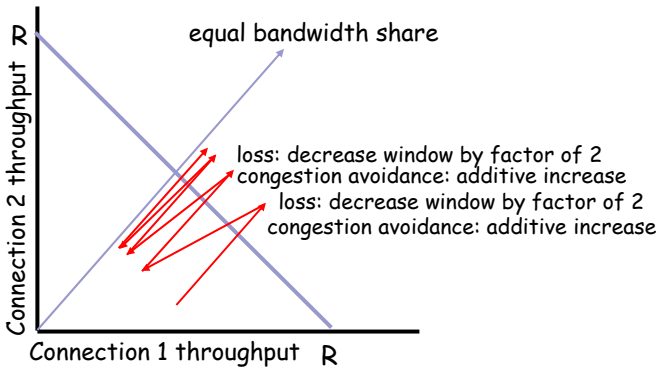
Fairness goal: if K TCP sessions share same bottleneck link of bandwidth R, each should have average rate of R/K



Why is TCP fair?

Two competing sessions:

- Additive increase gives slope of 1, as throughout increases
- multiplicative decrease decreases throughput in the direction of 0 throughput



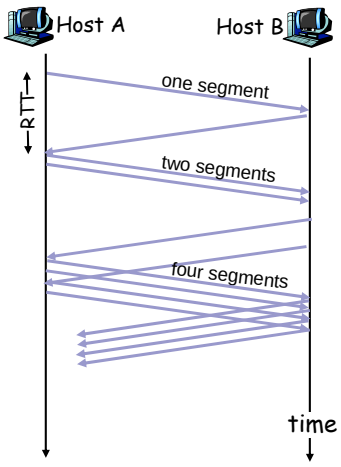
TCP Slow Start

- When connection begins, **CongWin** = 1 MSS
  - Example: MSS = 500 bytes & RTT = 200 msec
  - initial rate = 20 kbps
- available bandwidth may be >> MSS/RTT
  - desirable to quickly ramp up to respectable rate

- When connection begins, increase rate exponentially fast until first loss event

TCP Slow Start (more)

- When connection begins, increase rate exponentially until first loss event:
  - double **CongWin** every RTT
  - done by incrementing **CongWin** for every ACK received
- Summary: initial rate is slow but ramps up exponentially fast



Refinement

- After 3 dup ACKs:
  - **CongWin** is cut in half
  - window then grows linearly
- But after timeout event:
  - **CongWin** instead set to 1 MSS;
  - window then grows exponentially
  - to a threshold, then grows linearly

Philosophy:

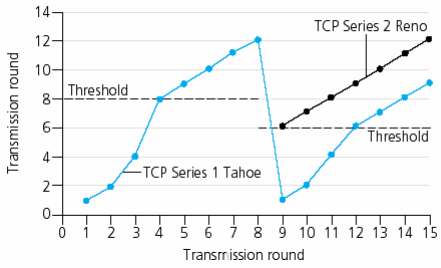
• 3 dup ACKs indicates network capable of delivering some segments
• timeout before 3 dup ACKs is "more alarming"

•TCP Reno started to use "**FAST RECOVERY**"

Refinement (more)

Q: When should the exponential increase switch to linear?

A: When **CongWin** gets to ssthreshold (= 1/2 of its value before timeout).



Implementation:

- Variable Threshold
- At loss event, Threshold is set to 1/2 of CongWin just before loss event

Summary: TCP Congestion Control

- When **`CongWin`** is below **`Threshold`**, sender in slow-start phase, window grows exponentially.
- When **`CongWin`** is above **`Threshold`**, sender is in congestion-avoidance phase, window grows linearly.
- When a triple duplicate ACK occurs, **`Threshold`** set to **`CongWin/2`** and **`CongWin`** set to **`Threshold`**.
- When timeout occurs, **`Threshold`** set to **`CongWin/2`** and **`CongWin`** is set to 1 MSS.

TCP sender congestion control

Event	State	TCP Sender Action	Commentary
ACK receipt for previously unacked data	Slow Start (SS)	CongWin = CongWin + MSS, If (CongWin > Threshold) set state to "Congestion Avoidance"	Resulting in a doubling of CongWin every RTT
ACK receipt for previously unacked data	Congestion Avoidance (CA)	CongWin = CongWin+MSS * (MSS/CongWin)	Additive increase, resulting in increase of CongWin by 1 MSS every RTT
Loss event detected by triple duplicate ACK	SS or CA	Threshold = CongWin/2, CongWin = Threshold, Set state to "Congestion Avoidance"	Fast recovery, implementing multiplicative decrease. CongWin will not drop below 1 MSS.
Timeout	SS or CA	Threshold = CongWin/2, CongWin = 1 MSS, Set state to "Slow Start"	Enter slow start
Duplicate ACK	SS or CA	Increment duplicate ACK count for segment being acked	CongWin and Threshold not changed

TCP Reno

- Tahoe + Fast re-transmit
- Packet loss detected both through timeouts, and through DUP-ACKs
- Sender reduces window by half, the ssthresh is set to half of current window, and congestion avoidance is performed (window increases only by 1 every round-trip time)
- Fast recovery ensures that pipe does not become empty
- Window cut-down to 1 (and subsequent slow-start) performed only on time-out

TCP New-Reno

- TCP-Reno with more intelligence during fast recovery
- In TCP-Reno, the first partial ACK will bring the sender out of the fast recovery phase
- Results in timeouts when there are multiple losses
- In TCP New-Reno, partial ACK is taken as an indication of another lost packet (which is immediately retransmitted).
- Sender comes out of fast recovery only after all outstanding packets (at the time of first loss) are ACKed

TCP SACK

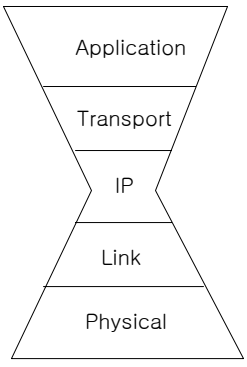
- TCP (Tahoe, Reno, and New-Reno) uses cumulative acknowledgements
- When there are multiple losses, TCP Reno and New-Reno can retransmit only one lost packet per round-trip time
- What about TCP-Tahoe?
- SACK enables receiver to give more information to sender about received packets allowing sender to recover from multiple-packet losses faster

Overview
IP

- What is difference among
  - Router
  - Switch
  - Hub

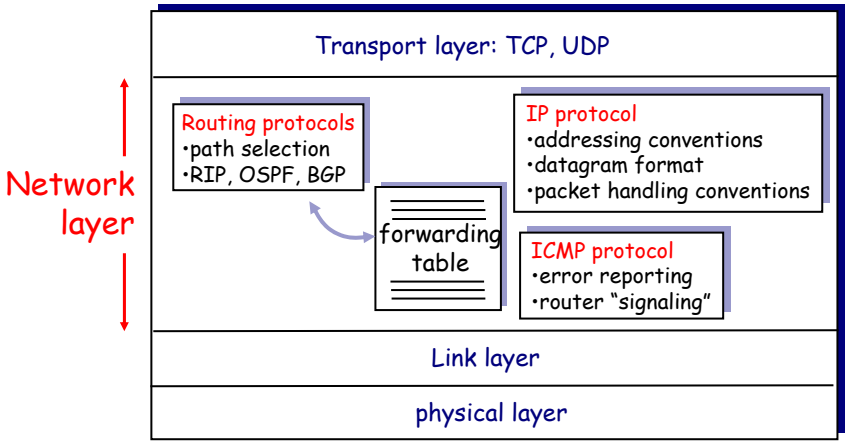
Internet Protocol (IP)

- Addressing
- Routing
- Fragmentation and Reassembly
- Quality of Service
- Multiplexing and Demultiplexing



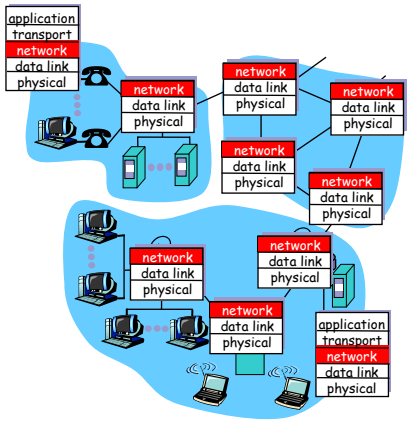
The Internet Network layer

Host, router network layer functions:



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



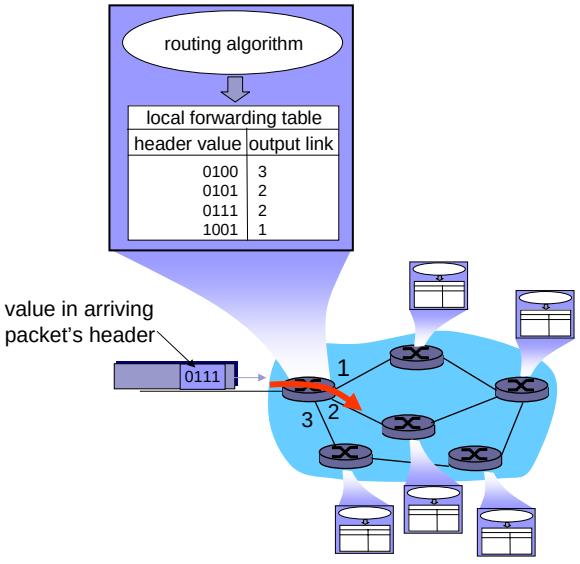
Key Network-Layer Functions

- *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

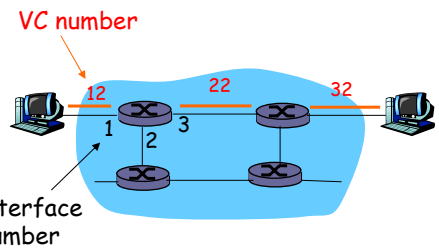
Interplay between routing and forwarding



Network layer connection and connectionless service

- Datagram network provides network-layer connectionless service
- VC network provides network-layer connection service
- Analogous to the transport-layer services, but:
  - Service: host-to-host
  - No choice: network provides one or the other
  - Implementation: in the core

Forwarding table



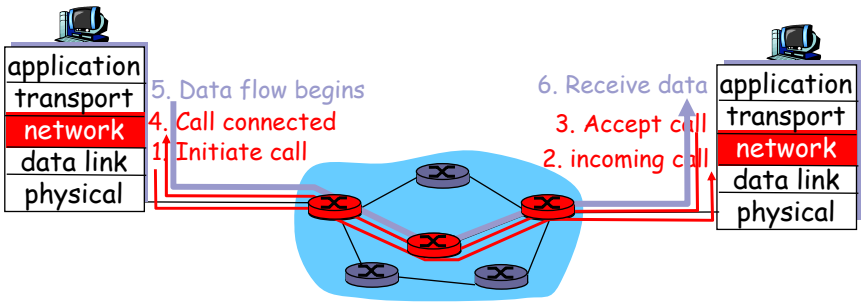
Forwarding table in northwest router:

Incoming interface	Incoming VC #	Outgoing interface	Outgoing VC #
1	12	2	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

Routers maintain connection state information!

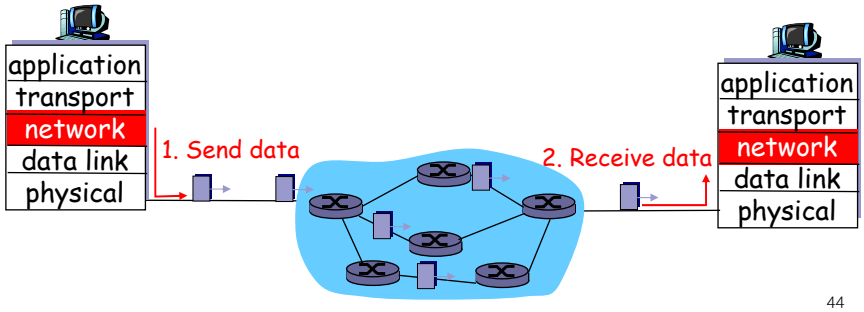
Virtual circuits: signaling protocols

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



Datagram networks

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



Comparison of Virtual-Circuit and Datagram Subnets

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Forwarding table

4 billion possible entries

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

Huge size of table

Longest prefix matching

Prefix Match	Link Interface
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
otherwise	3

Examples

DA: 11001000 00010111 00010110 10100001 Which interface?

DA: 11001000 00010111 00011000 10101010 Which interface?

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

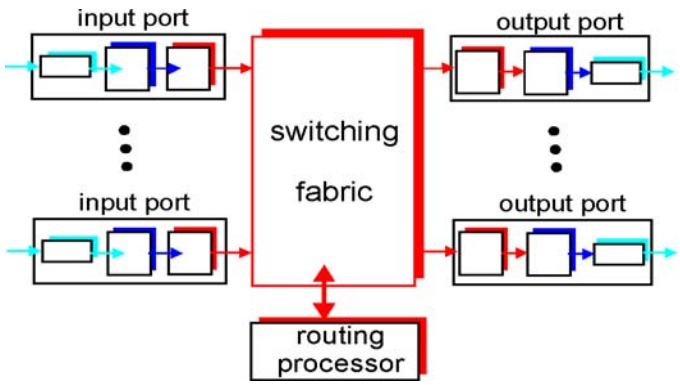
ATM

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

Router Architecture Overview

Two key router functions:

- run routing algorithms/protocol (RIP, OSPF, BGP)
- *forwarding* datagrams from incoming to outgoing link



IP addressing: the last word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

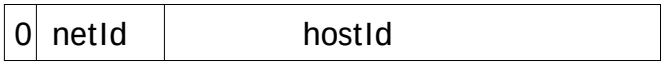
- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

Addressing

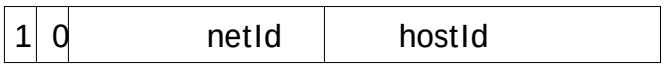
- Need unique identifier for every host in the Internet (analogous to postal address)
- IP addresses are 32 bits long
- Hierarchical addressing scheme
- Conceptually ...
  - IPaddress =(NetworkAddress,HostAddress)

Address Classes

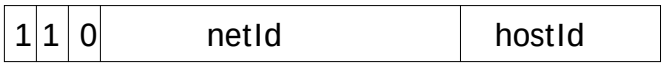
- Class A



- Class B



- Class C



- Two more classes
  - 1110 : multicast addressing
  - 1111 : reserved

Addresses and Hosts

- Since netId is encoded into IP address, each host will have a unique IP address for each of its network connections
- Hence, IP addresses refer to network connections and not hosts
- Why will hosts have multiple network connections?

Special Addresses

- hostId of 0 : network address
- hostId of all 1's: directed broadcast
- All 1's : limited broadcast
- netId of 0 : this network
- Loopback : 127.0.0.0

Dotted decimal notation: IP addresses are written as four decimal integers separated by decimal points, where each integer gives the value of one octet of the IP address.

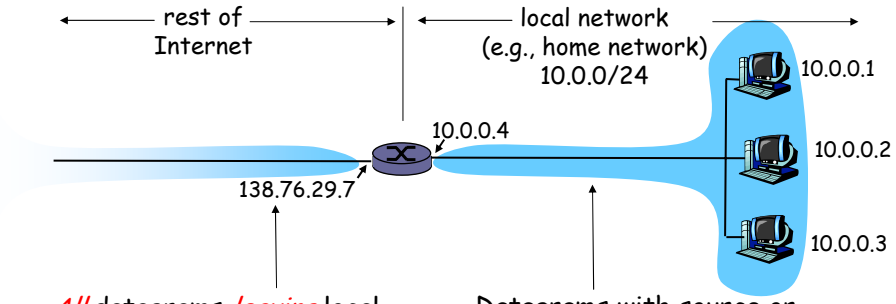
Exceptions to Addressing

- Subnetting
  - Splitting hostId into subnetId and hostId
  - Achieved using subnet masks
  - Useful for?
- Supernetting (Classless Inter-domain Routing or CIDR)
  - Combining multiple lower class address ranges into one range
  - Achieved using 32 bit masks and max prefix routing
  - Useful for?

Examples

- Subnetting
  - 192.168.1.0/24 – class C network
  - 192.168.1.64/26 and 192.168.1.128/26 – 2 subnetworks with upto 62 stations each!
- Supernetting
  - 192.168.2.0/24 and 192.168.3.0/24 – 2 class C networks
  - 192.168.2.0/23 – 1 super network with upto 126 stations!!

NAT: Network Address Translation



All datagrams *leaving* local network have same single source NAT IP address: 138.76.29.7, different source port numbers

Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: Network Address Translation

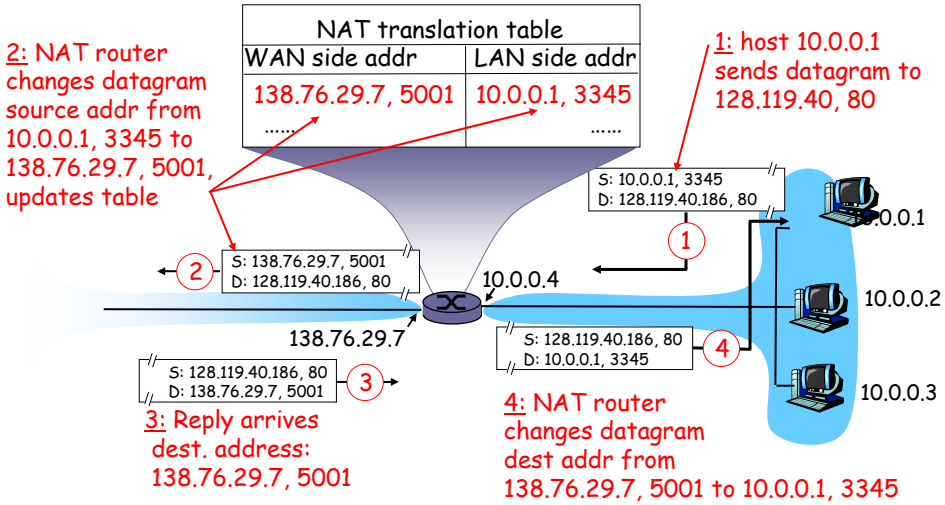
- Motivation: local network uses just one IP address as far as outside word is concerned:
  - no need to be allocated range of addresses from ISP: - just one IP address is used for all devices
  - can change addresses of devices in local network without notifying outside world
  - can change ISP without changing addresses of devices in local network
  - devices inside local net not explicitly addressable, visible by outside world (a security plus).

NAT: Network Address Translation

Implementation: NAT router must:

- *outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  . . . remote clients/servers will respond using (NAT IP address, new port #) as destination addr.
- *remember (in NAT translation table)* every (source IP address, port #) to (NAT IP address, new port #) translation pair
- *incoming datagrams: replace* (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: Network Address Translation



IP Routing

- Direct
  - If source and destination hosts are connected directly
  - Still need to perform IP address to physical address translation. Why?
- Indirect
  - Table driven routing
  - Each entry: (NetId, RouterId)
    - Default router
    - Host-specific routes

IP Routing Algorithm

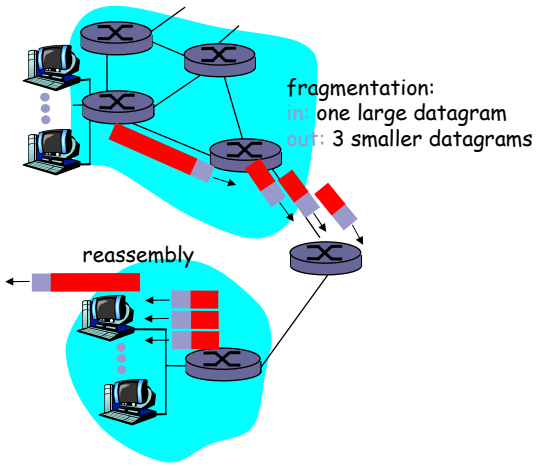
- RouteDatagram(Datagram, RoutingTable)
- Extract destination IP address, D, from the datagram and compute the netID N
  - If N matches any directly connected network address deliver datagram to destination D over that network
  - Else if the table contains a host-specific route for D, send datagram to next-hop specified in table
  - Else if the table contains a route for network N send datagram to next-hop specified in table
  - Else if the table contains a default route send datagram to the default router specified in table
  - Else declare a routing error

Routing Protocols

- Interior Gateway Protocol (IGP)
  - Within an autonomous domain
  - RIP (distance vector protocol), OSPF (link state protocol)
- Exterior Gateway Protocol (EGP)
  - Across autonomous domains
  - BGP (border gateway protocol)

IP Fragmentation & Reassembly

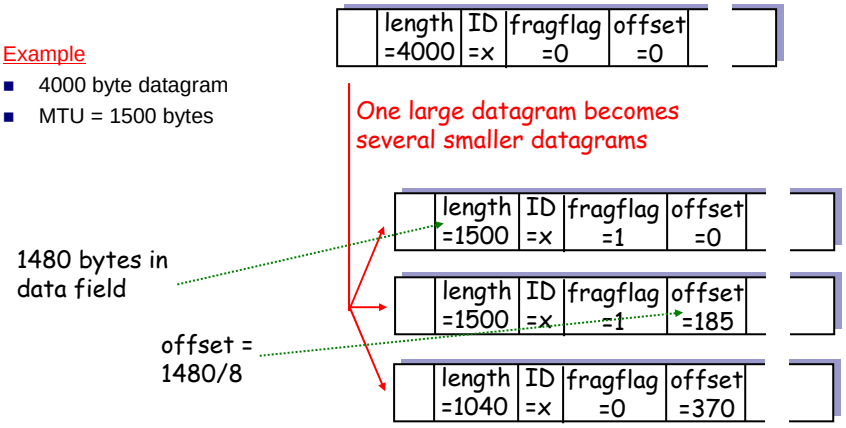
- network links have MTU (max.transfer size) - largest possible link-level frame.
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments



IP Fragmentation and Reassembly

Example

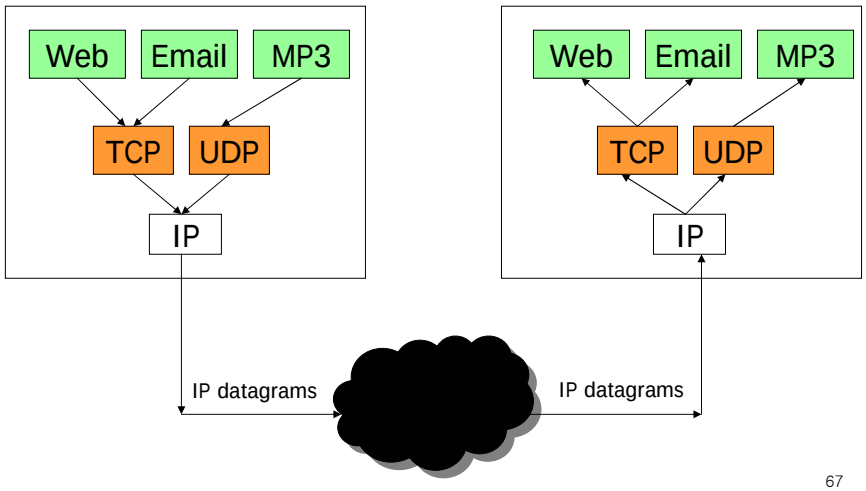
- 4000 byte datagram
- MTU = 1500 bytes



IP Reassembly

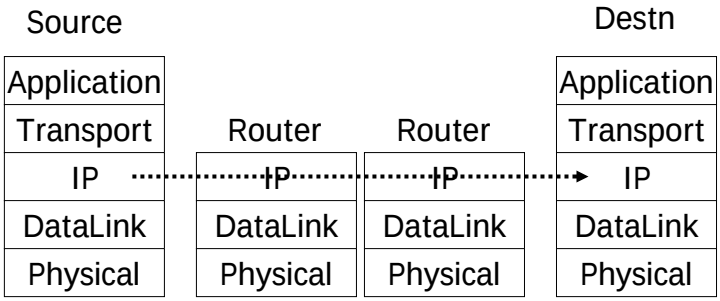
- Fragmented packets need to be put together
- Where does reassembly occur?
- What are the trade-offs?

Multiplexing

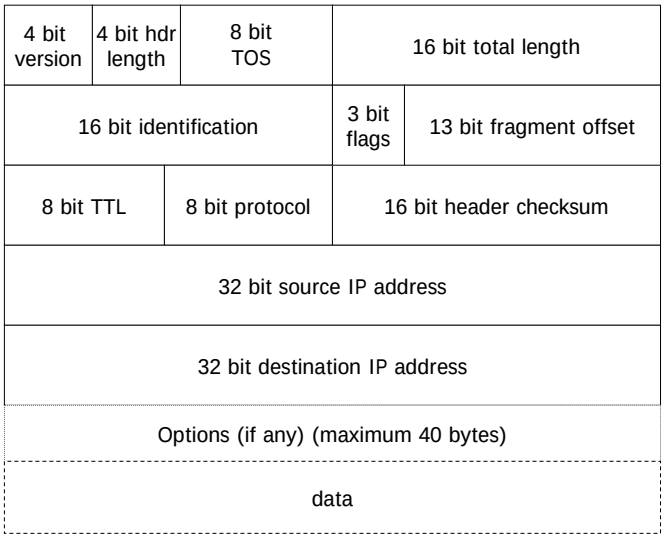


IP Header

- Used for conveying information to peer IP layers



IP Header (contd.)



69