**CH1 Intro to Networking 1.1 What is the internet:**

Millions of connected devices

Host = end system - Anything that runs network apps

Communication links

Physical infrastructure - Transmission rate = bandwidth

Packet switches: Devices that forwards packets

Internet: “network of networks” = Interconnected ISPs

Protocol - control sending, receiving of msgs. Defines format, order of msgs among network entities, and actions taken on msg transmission, receipt standards.

e.g., TCP, IP, HTTP, Skype, 802.11

RFC: Request for comments

IETF: Internet Engineering Task Force

Internet can be seen as a service:

Infrastructure that provides a service to applications

Ex: websites, games, e-commerce. Or as a service to devs Don’t need to worry, just plug into the API

**1.2 Network Edge**

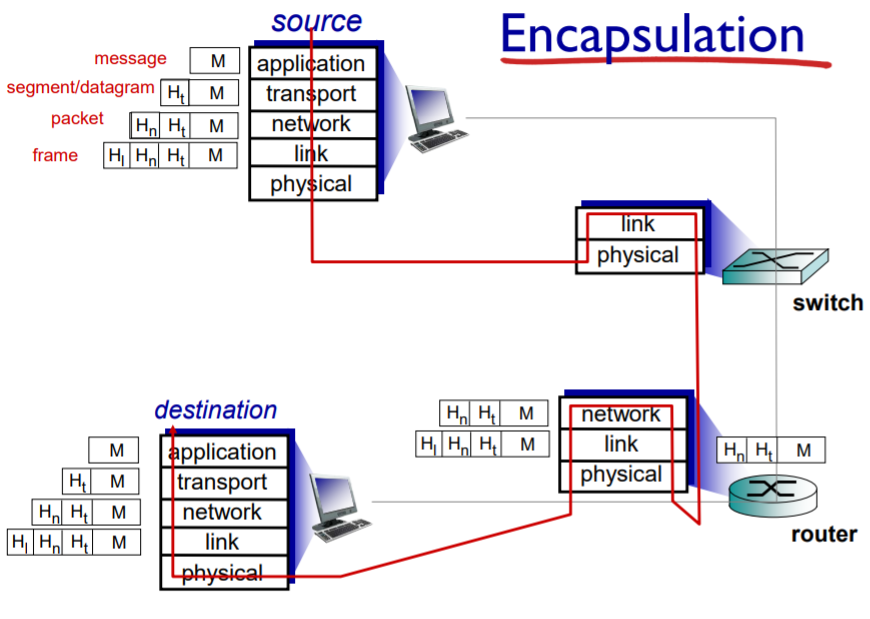
network edge: End user side hosts: clients and servers

Hosts sends packets of length L bits, Transmits at transmission rate R, Transmission Delay = L/R

Access networks: in-between networks that connect users through the carrier network, to other networks such as the Internet. Types of Access Networks: residential access nets, institutional access networks (school, company),mobile access networks.

Physical Media:Wires(DSL/Cable),Radio(WLAN/Cellular)

**1.3 Network core:** interconnected routers that connect everything, network of networks, mesh of routers Forward packets through routers from source to destination. Packet Switching - forward packets through routers, takes L/R seconds to transmit (push out) L-bit packet into link at R bps

store and forward: entire packet must arrive at router before it can be transmitted. end-end delay = 2L/R (assuming zero propagation delay)

queuing and loss: If arrival rate (in bits) to link exceeds transmission rate:

packets will queue, wait to be transmitted on link leading to delay packets can be dropped (lost) if memory (buffer) fills up.

Two key network-core functions

Forwarding: move packets from router’s input to appropriate router output. Routing: determines source destination route taken by packets via routing algrtms.

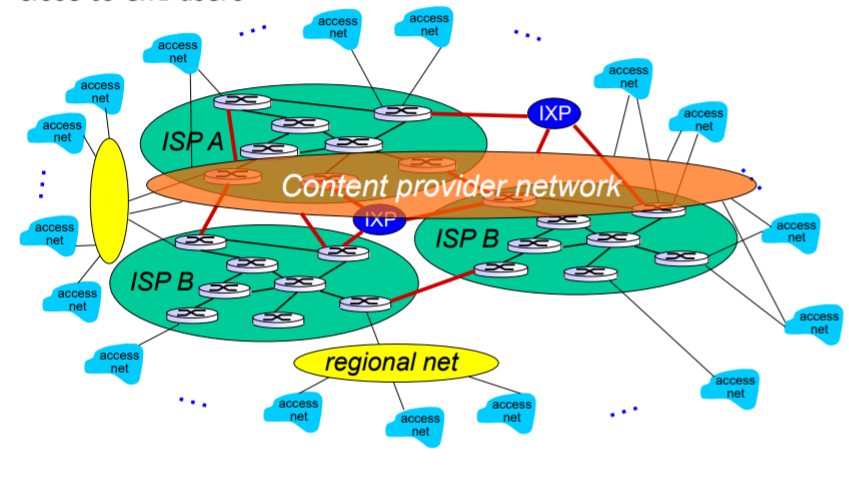
Circuit Switching: Alternative switching method to packet. end-end resources allocated to, reserved for “call” between source & dest

Packet switch vs circuit switch: Packet switching allows more users to use network.

example: 1 Mb/s link each user: • 100 kb/s when “active” • active 10% of time

circuit-switching: 10 users

packet switching: with 35 users, probability > 10 active at same time is less than .0004%

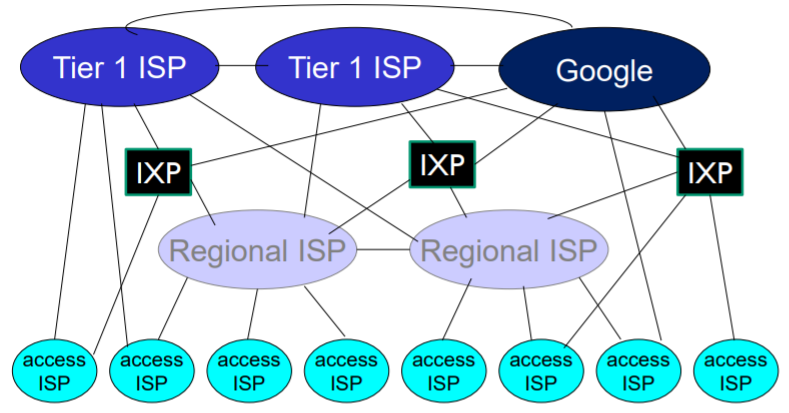
Packet is not “slam dunk winner”, excessive congestion possible. Good for bursty data.

Internet structure: network of networks

End systems connect to Internet via access ISPs. (Residential, company and university ISPs)

Access ISPs must be interconnected so that any two hosts can send packets to each other.

Current structure in place because of economics - connecting directly is not scalable.

**1.4 Performance: delay, loss, and throughput in Netwks**

How do loss and delay occur: packets queue in router buffers packet arrival rate to link (temporarily) exceed output link capacity packets queue, wait for turn.

Four sources of packet delay: dproc: nodal processing - check bit errors - determine output link - typically small, < msec. dqueue: queueing delay time waiting at output link for transmission depends on congestion level of router. dtrans: transmission delay: L: packet length (bits) R: link bandwidth (bps) dtrans = L/R

dprop: propagation delay: d: length of physical link s: prop speed in medium (~2x108 m/sec) d prop = d/s

Make sure not to mix transmission and propagation, transmission is amount of time to leave router, propagation is amount of time to get to the router

dtotal = dproc + dqueue + dtrans + dprop

Throughput: rate (bits/time unit) at which bits transferred between sender/receiver.

instantaneous: rate at given point in time

average: rate over longer period of time

Effective rate is bottleneck - the slower link that constrains throughput. **1.5 Protocol Layers, Service models**: Protocol Layers: 1)App 2) Presentation 3)Session 4)Transport 5)Network 6)Data-Link 7)Physical. In networking, Presentation and Session are absent. Layering helps dealing with complex systems thanks to explicit structure, modularization eases maintenance. 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. Ethernet, 802.11 (WiFi), PPP. physical: bits “on the wire”

Encapsulation: as data is processed, headers are added to the data to inform where it goes. **1.6 Networks under attack: Security** Internet not originally designed with any security in mind. Internet protocol designers playing catch-up. security considerations must be in all layers! Malware: Virus: self-replicating infection that needs to be ran.

Worm: self-replicating infection by passively replicate

Spyware - can track user can record keystrokes, web sites visited, upload info to collection site.

Botnet - Attack that controls other computers,

DOS - Denial of Service Attack, load resources so server is unavailable for actual traffic.

DDOS - Distributed Denial of Service Attack (using many types of computers). Packet Sniffing - attack that reads packets. IP Spoofing - spoofs address so source computer sends packet to wrong destination (man in the middle attack) **CH2 App layer**  **2.1 Principles**

Creating a network app - when you create app layer, you write programs that: run on (different) end systems,communicate over network.

Don’t need to write software for network-core

- network-core devices do not run user applications -allows for rapid app development, propagation

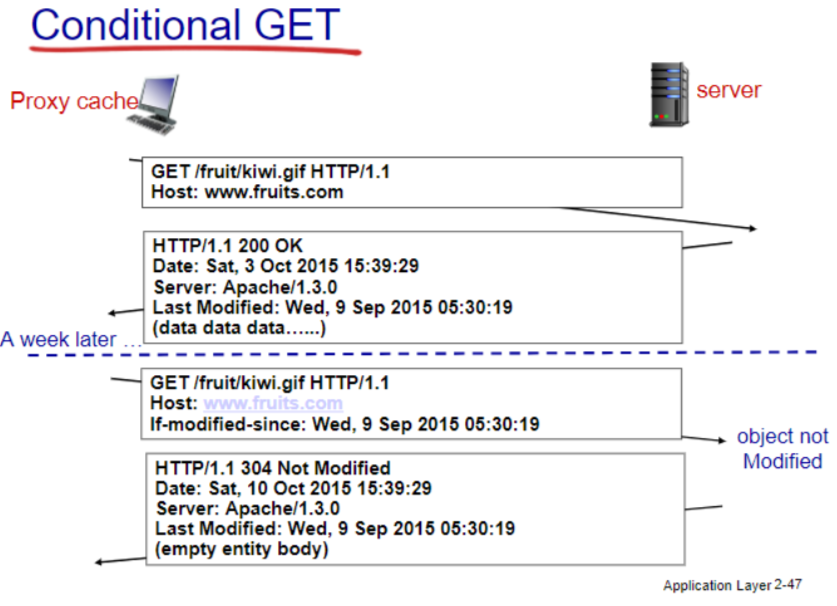
Process communication - process: program running within a host, Port number sends to specific process

client process:process that initiates communication

server process:process that waits to be contacted

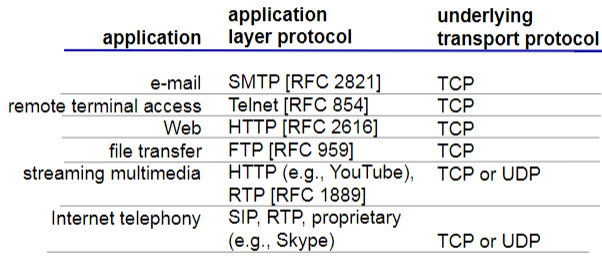
Sockets- process sends/receives messages to/from socket, socket like a mailbox.

What does app-layer protocol define?

types of messages exchanged - request, response message syntax: what fields in messages & how fields are portrayed. message semantics: meaning of information in fields rules for when and how processes send & respond to messages.

Transport layer services-Needs to be reliable, fast/time sensitive,have throughput, secure.

TCP (Transmission Control Protocol) - is reliable, has flow control, congestion control, does not provide timing, security, requires a setup between client and server. Congestion Control: throttle sender when network is overloaded. Flow Control: sender won’t overwhelm receiver.

UDP (User Datagram Protocol) service- Unreliable, provides timing does not provide, flow control, congestion control, throughput guarantee, security.

UDP exists because: no connection saves time, faster when there's no need to check reliability, and header is smaller than tcp.

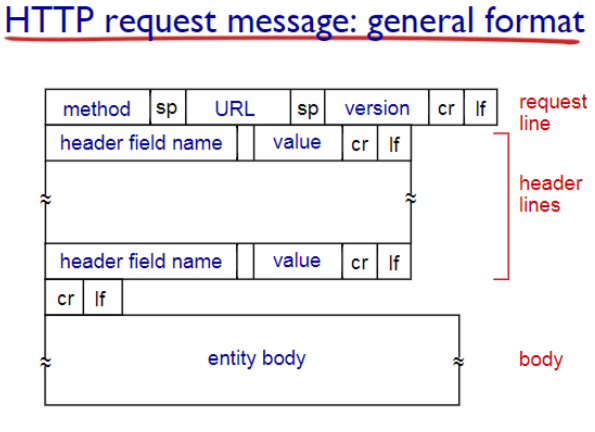
Securing TCP: \*SSL provides encrypted TCP connection, data integrity, end-point authentication. SSL is at app layer, Apps use SSL libraries, which “talk” w TCP

**2.2 Web and HTTP:** Web page consists of objects:

object can be any asset/file ...

Web page consists of base HTML-file which includes referenced objects. Each object is addressable by a URL

HTTP: hypertext transfer protocol - Web’s application layer protocol Client/server model (through HTTP Protocol): client: browser that requests, receives and “displays” Web objects server: Web server sends objs in response HTTP is “stateless” : server maintains no info about past client requests. non-persistent HTTP -at most one object sent over TCP connection, connection then closed. Persistent HTTP -multiple objects can be sent over single TCP connection. HTTP request message: request line, header lines, carriage return: line feed at start indicate end.



HTTP response:

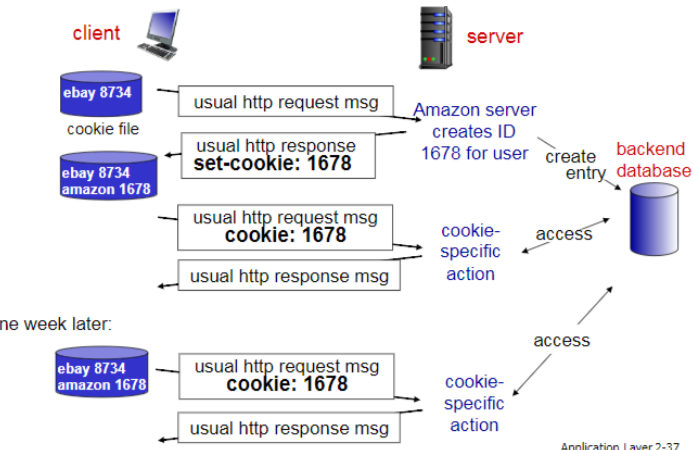
Status codes: 200 OK- request succeeded

301 Moved Permanently -requested object moved,

400 Bad Request request msg not understood.

404 Not Found requested document not found on this server 505 HTTP Version Not Supported

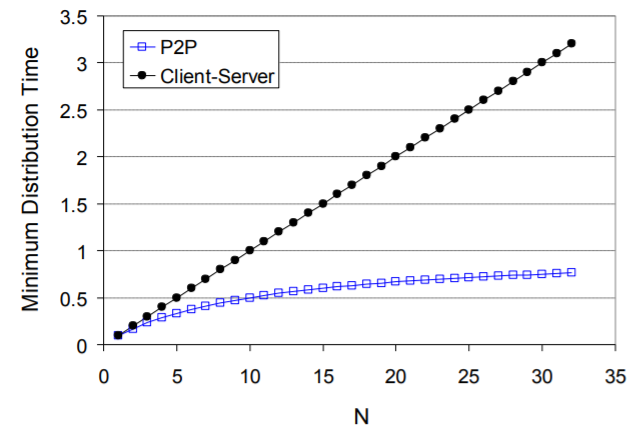
Format: Status line w/ protocol response, header lines, data/requested file. Response Methods: HTTP/1.0: GET, POST | HTTP/1.1: GET, POST, HEAD, PUT, DELETE POST -input is uploaded to server in entity body URL method: uses GET method, input is uploaded in URL

HEAD - asks server to leave requested object out of response PUT - uploads file in entity body to path specified in URL field |DELETE - deletes file specified in the URL field. Cookie: a method of keeping user/server state. Cookies used for: authorization, shopping carts, state (Web e-mail) , etc - Server stores cookie on database, client saves cookie file

Web cache - put data on closer server via cashing. Typically by isp or institution

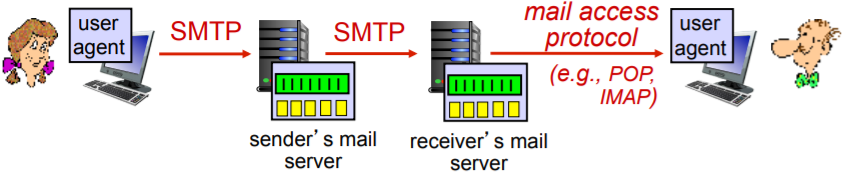
why Web caching?: reduce response time, Reduce traffic on access link, enables “poor” providers to effectively deliver content, Cheap.

Understand the example: local cache has smaller delay than vs widening access while being cheaper

Conditional GET - Goal: don’t send object if cache has up-to-date cached version Use the Conditional GET message to verify up to date. GET method: If-modified-since: <date>  **2.3 email**  major components: user agents, mail servers. simple mail transfer protocol: SMTP. SMTP uses TCP to reliably transfer, direct transfer: sending server to receiving server.

Mail message format: SMTP, RFC 822 & body: RFC 822 is standard for message header lines, e.x) To:, From, Subject: Different from SMTP MAIL FROM, RCPT TO: Body: the message in ASCII characters only

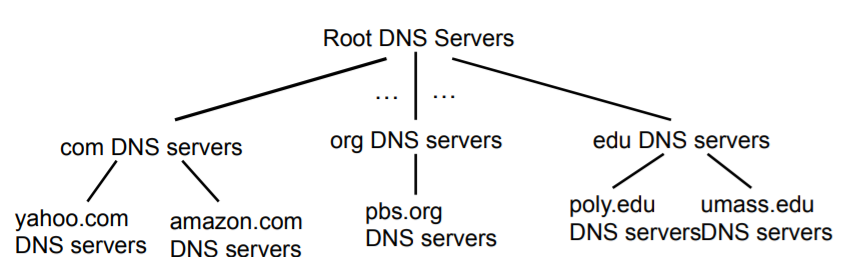
Mail access protocols: POP3: Post Office Protocol authorization, Download IMAP: Internet Mail Access Protocol more features,uses a server. HTTP: online email system. Differences: POP3 is stateless uses download & Delete, IMAP keeps all messages on server while keeping user state between sessions.

For web, link between user and their server is HTTP

**2.4 DNS** DNS: Domain Name System

Services: Hostname to IP Address Translation (aka url), host aliasing, mail server aliasing, load distribution

Hierarchical structure:



Root: contacted by local if cannot resolve name. contacts authoritative name server if name mapping not known, gets mapping, returns mapping to local

TLD (top-level domain) servers: responsible for common domain like com, org, and for country lvl

authoritative DNS servers:organization’s own DNS server(s), providing authoritative hostname to IP mappings for organization’s named hosts

Local DNS: does not strictly belong to hierarchy

each ISP has one - when host makes DNS query, query is sent to its local DNS server, has local cache

- acts as proxy, forwards query into hierarchy

iterated query: contacted server replies with name of server to contact. recursive query: puts burden of name resolution on contacted name server

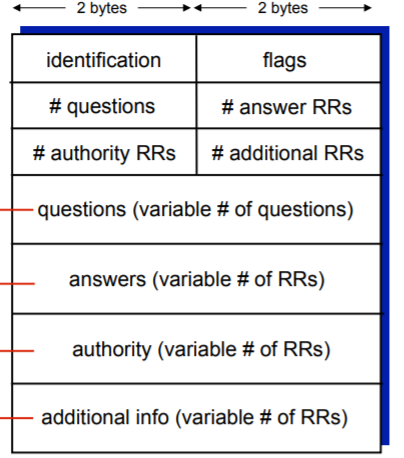
DNS caching: once (any) name server learns mapping, it caches/temp saves mapping.

DNS records: distributed database storing resource records (RR): -(name, value, type, ttl)

type=A **name** is hostname § **value** is IP address

type=NS n**ame** is domain (e.g.,foo.com)

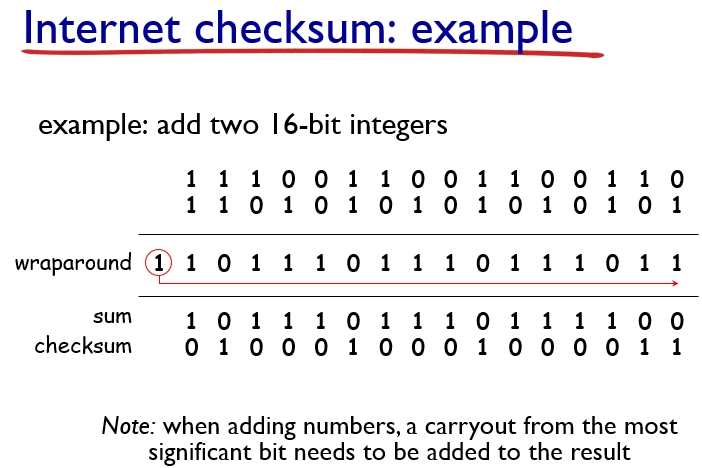
**value** is hostname of authoritative name

server for this domain

type=CNAME **name** is alias name for real name  **value** is canonical name. type=MX **value** is name of mailserver associated with **name**

Image is DNS message Format

DNS registrar inserts two RRs into .com TLD server: type A record for [www.\_url](about:blank), Type MX for \_url

Attacking DNS - DDoS Attacks bombard root or TLD server with traffic, redirect attacks intercept queries. **2.5 P2P** Client-Server Architecture: Traditional network infrastructure with servers. Server: always-on host | permanent IP address | Clients: communicate with server | intermittently connected | dynamic IP | do not communicate directly with other clients.

P2P (Peer 2 Peer) architecture: no servers - peers host data & directly communicate | peers request & provide service from other peers| more scalable &self scalable – new peers bring new capacity| Drawback: - complex management bc peer is not a dedicated server

features of P2P - file distribution, streaming, VoIP

D = time, d = download capacity, u = upload capacity, F= file size, N= Peers.

BitTorrent: P2P file distribution - gets chunks over time, registers with trackers to get peers understand the mechanism. Tit-for-tat: While sending looks for top 4 peers, evaluates peer every 30s. **2.6 video streaming and CDN**

DASH(Dynamic, Adaptive Streaming over HTTP) server: divides video file into multiple chunks, each chunk stored, encoded at different rates. manifest file: provides URLs for different chunks. client: periodically measures bandwidth •chooses maximum coding rate sustainable given current bandwidth. Content distribution Networks - store/serve multiple copies of videos at different places ex) Netflix cx wants madman

CDN: stores copies of content at CDN nodes

When getting file from CDN - directed to nearby copy, retrieves content, may choose different copy if network path congested. Youtube - does not use DASH, uses DNS Redirect.Kankan - Hybrid CDN-P2P - CDN when P2P bad. **2.7 socket programming**

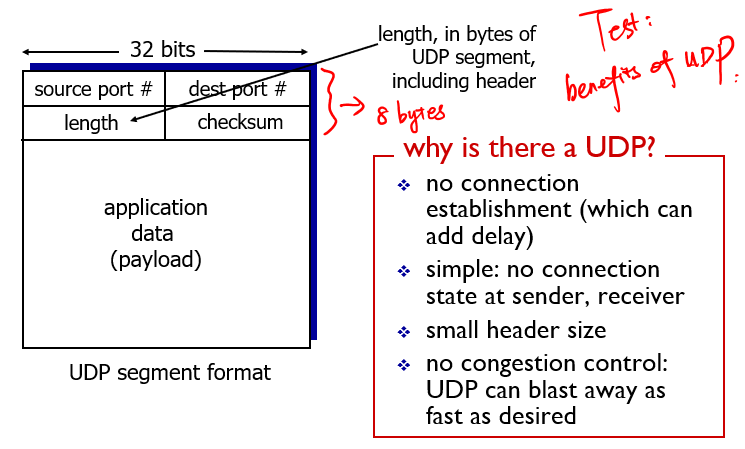
UDP: no “connection” between client and server.

TCP: Client must contact server, doing so by creating tcp socket, specifying ip address, and port number of server process. Server creates new socket which allows talk with multiple clients.

**Chapter 3 Transport Layer 3.1 Transport layer services** Transport Layer provides logical communication between app processes running on different hosts - Network is host to host, transport is process to process. Transport is more precise than network layer because delivers to specific processes. Transport protocols run in end systems: send side: breaks app messages, passes to netwrk layer. Rcv side: reassembles segments, passes to app layer

TCP VS UDP - TCP is reliable in order, has congestion control, flow control, connection setup, UDP is unreliable, unordered. “no frills extension of ”best effort” service”. TCP is more reliable but UDP is faster and smaller. Both have no security, delay or bandwidth guarantees. **3.2 Multiplexing and Demultiplexing**

Multiplexing from app to transport, handle data from multiple sockets, adds header. Demultiplexing - from transport to app. Host receives datagrams, each has source and destination IP addresses. Host use IP addresses and port# to direct segment to the appropriate socket. Connectionless - when host receives, checks destination port and directs segment to socket with port number. Connection oriented - TCP socket identified by 4-tuple: source IP address, source port number,dest IP address, dest port number, receiver uses all 4 values to direct segment to socket.

**3.3 Connectionless transport: UDP**

UDP (User Datagram Protocol)- Unreliable, unordered delivery.

connectionless protocol- no handshaking w/ send & Rcv

“no frills extension of ”best effort” service” - stuff can be lost or out of order.

UDP Checksum: a method to detect errors Sender - treats contents as a sequence of 16 bit integers, adds up segment contents and puts checksum value into UDP field.Receiver calculates checksum of segment contents, if values do not match then there is an error (not perfect, if 2 bits flip, shows ok).

**3.4 Principles of reliable data transfer** Application layer assumes a reliable channel but infrastructure is inherently unreliable. Need to enforce a reliable protocol using an unreliable channel.

Reliable Data Transfer protocol (RDT)

RDT 1.0 - Assumes that Channel is reliable, does nothin

RDT 2.0 - Introduces error detection(checksum) - assumes underlying channel may flip bits, recover from errors via ACK and NAK. ACK - Acknowledge that packet is OK NAK - Negative Acknowledgement

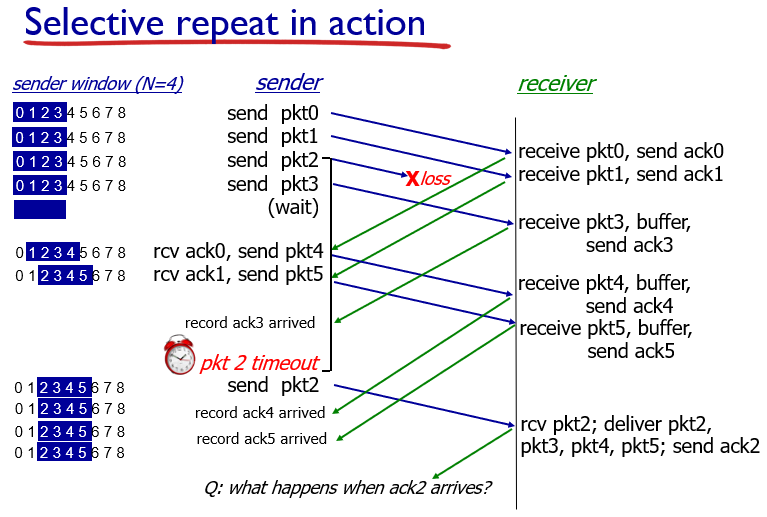
RDT 2.0 Fatal Flaw - what if ACK or NAK get corrupted?

RDT 2.1 - Solves Fatal flaw - avoids duplicate packets with sequence numbers- must check of ACK/NAK is OK

RDT 2.2 - NAK Is redundant - if you repeat last ACK, we know that current ACK is bad.

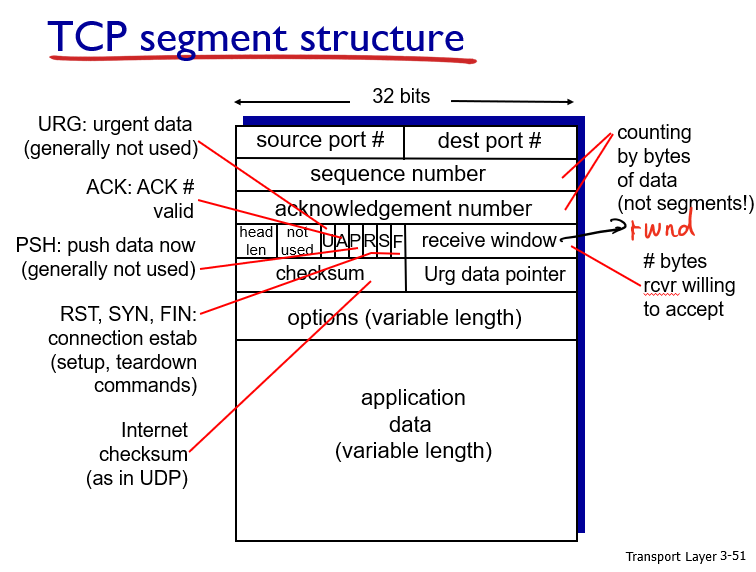
RDT 3.0 - New assumption - underlying channel can lose data, previous RDT only looks at error. Issue with 3: Performance sucks. Pipelined Protocols - pipelining: sender allows multiple, “in-flight”, yet to be acknowledged pkts. Two types of pipelined protocols: go-Back-N, Selective Repeat. We have pipelining because it significantly increases utilization. go-Back-N has a timer, within the timespan - sender can have up to N un-acked packets in pipeline. receiver only sends cumulative ack. doesn’t ack packet if there’s a gap - when timer expires, retransmit all unacked pkts.

Selective Repeat: sender has up to N unACK’ed packets in pipeline, rcvr sends ack for each packet. Sender maintains individual timer, when expires, retransmits 1.

Window Size - window of up to N consecutive un-ack’d packets are allowed .

**3.5 Connection-oriented transport: TCP**

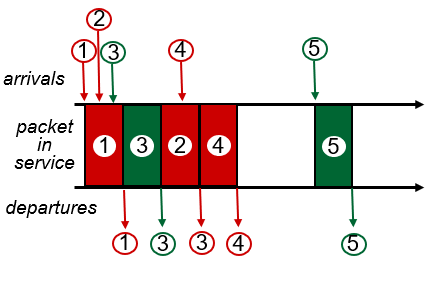
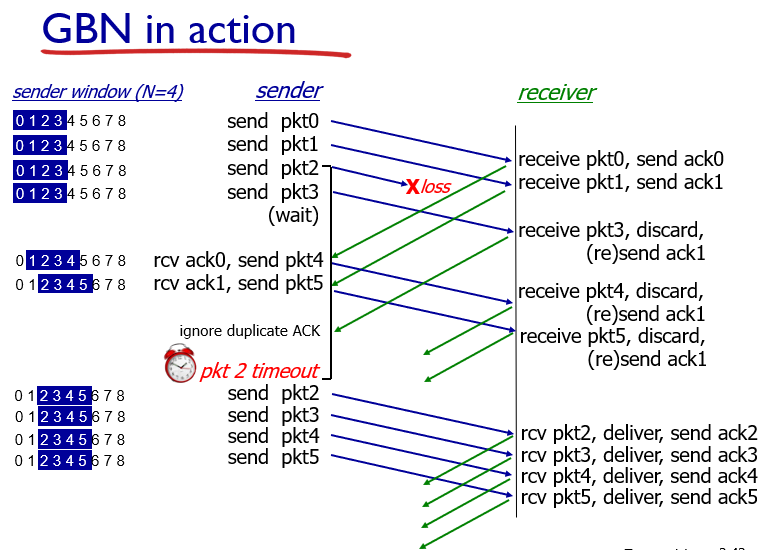
TCP (Transmission Control Protocol) - reliable, in order delivery, does not provide timing, security

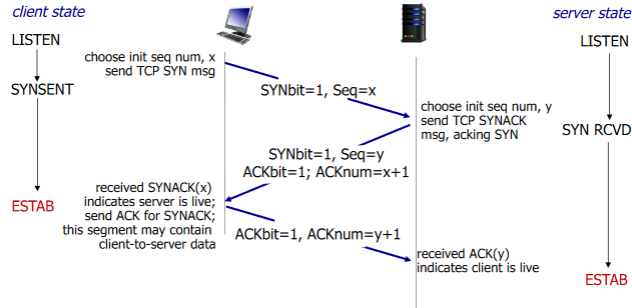
features of TCP:point-to-point:one sender, one receiver 

reliable, in-order byte stream:no “message boundaries” Pipelined: TCP congestion & flow control set window size. full duplex data: bi-directional data flow in same connection. connection-oriented: handshaking (exchange of control msgs) inits sender, receiver state before data exchange. Makes more reliable. Flow Control: sender won’t overwhelm receiver. Congestion Control: throttle sender when network is overloaded. Sequence numbers: Byte stream “number” of first byte in segment’s data. acknowledgements: seq # of next byte expected from other side Ex) Host A types C, seq 42,ACK 79,data ‘C’, host B ACKs C and echoes back seq 79, ACK 43,data ‘C’, host A ACKs receipt of echoed C Seq43, ACK 80. How long to configure timer for TCP? Slightly longer than Round Trip Time (RTT)

TCP Scenarios: Lost ACK ACK is lost and sequence is retransmitted even though data has already been received. Premature timeout ACK return is slow and ACK gets back to sender after resubmits information

Cumulative ACK - host A sends to host B data cumulatively, sends cumulative ACK back. Early ACK lost but later received, continues to send information from latter. TCP Fast Retransmit: when long timeout window, receiver send 3+ duplicate ACK and sender will resubmit early because segment lost.

TCP Flow Control: TCP limits flow to free buffer space - receiver sends back rwnd value and sender limits data to rwnd value. TCP 3 Way Handshake



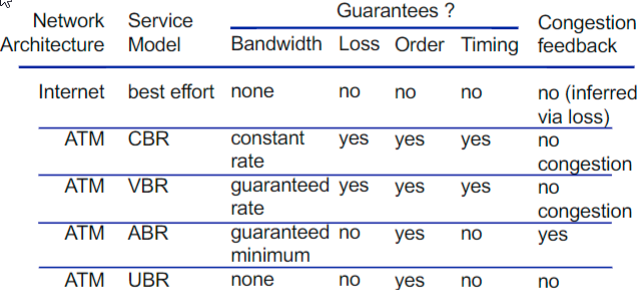
**3.6 Principles of Congestion control:**

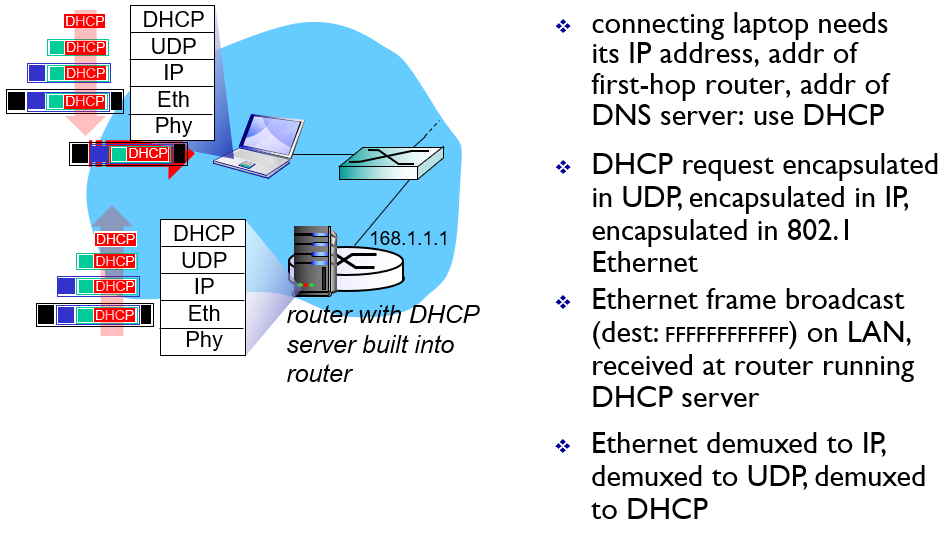
Main Causes of Congestion - lost packets (buffer overflow at routers)&long delays (queueing in router buffers). **3.7 TCP congestion control**

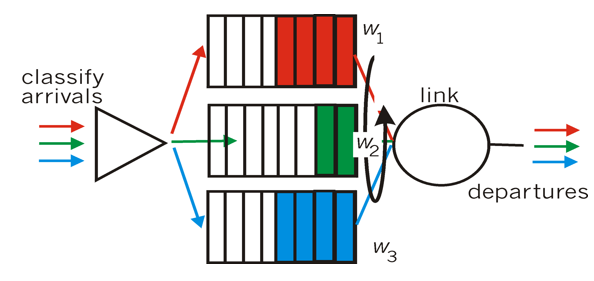
CWND - congestion window. Two ways to determine:

AIMD(Additive Increase Multiplicative decrease) - “Sawtooth”, increments the transmission rate/window size until packet loss occurs then cuts in half TCP Slow Start - exponentially increase until loss occurs, initially cwnd = 1 MSS (Max Segment Size), double cwnd every RTT(Round Trip time), done by incrementing cwnd for every ACK. TCP Reno is AIMD, TCP Tahoe is slow start

Why TCP is fair? -1) additive increase gives slope of 1, as throughput increases 2) multiplicative decrease decreases throughput proportionally, eventually will even out to equal bandwidth.

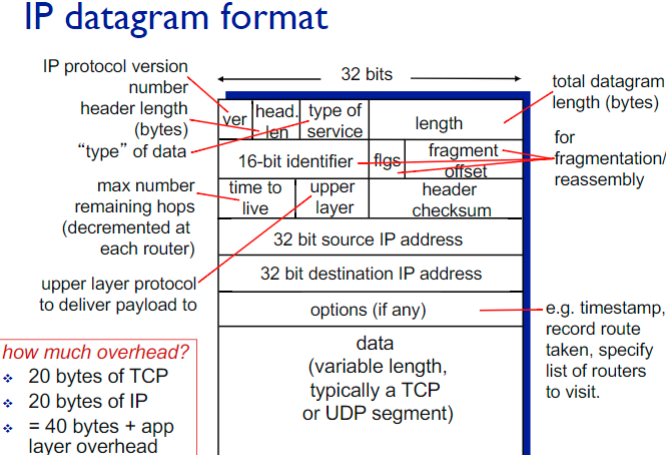
**Chapter 4 Network Layer 4.1 Introduction** Network Layer - transport segment from sending to receiving host. Two network-layer functions - Forwarding and Routing: Forwarding: move packets from router input to output. Routing: determine route by packets from source to dest. Routing algorithm determines end-end-path through network forwarding table determines local forwarding at specific router. **4.2 Virtual circuit and datagram networks**

Datagram forwarding table: Bc billions of IP, rather than list individual destination address, list range of addresses, and use longest prefix matching to find specific IP. Longest prefix matching: When looking for the forwarding table entry for given destination address, use the longest address prefix that matches the destination address, aka the one that is the closest.

**4.3 What’s inside a router** two key router functions: run routing algorithms/protocol (OSPF, BGP) & forwarding datagrams from incoming to outgoing link. The architecture of a router: 4 main parts: Input & output ports, routing processor, high speed switching fabric. Three types of switching fabrics: memory, bus, crossbar. scheduling: choose next packet to send on link - FIFO (first in first out) scheduling: send in order of arrival to queue. priority scheduling: high priority & low priority queues, if 2 come same time, send highest priority queued packet first. Round Robin (RR) scheduling: multiple classes, cyclically scan class queues, sending one complete packet from each class.

Weighted Fair Queuing (WFQ): -> generalized Round Robin - each class gets weighted amount of service in each cycle - like a mix of RR & priority.

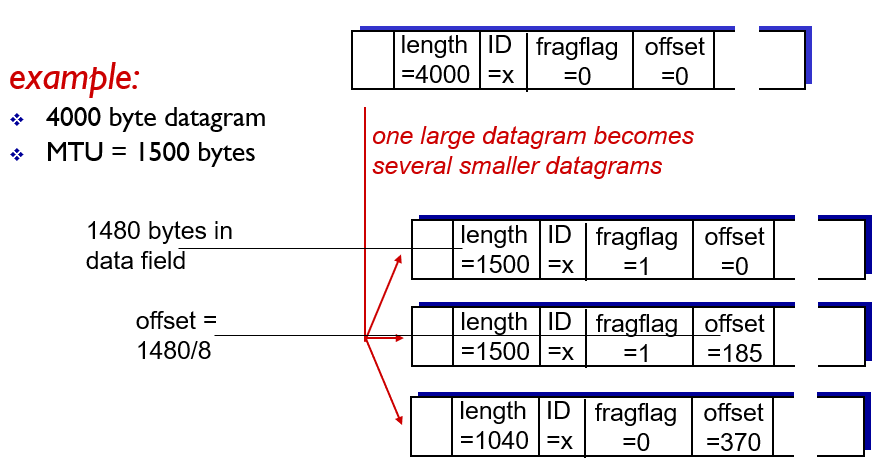
**4.4 Internet protocol:** IP datagram format

IP fragmentation: network links have MTU (max.transfer size). To get datagrams to fit, need to fragmented large IP datagrams into several datagrams. “reassembled” only at final destination. IP header bits used to identify, order related fragments. **Orig Data** =*Dtgm – hdr* = 4000 – 20 =3980. 

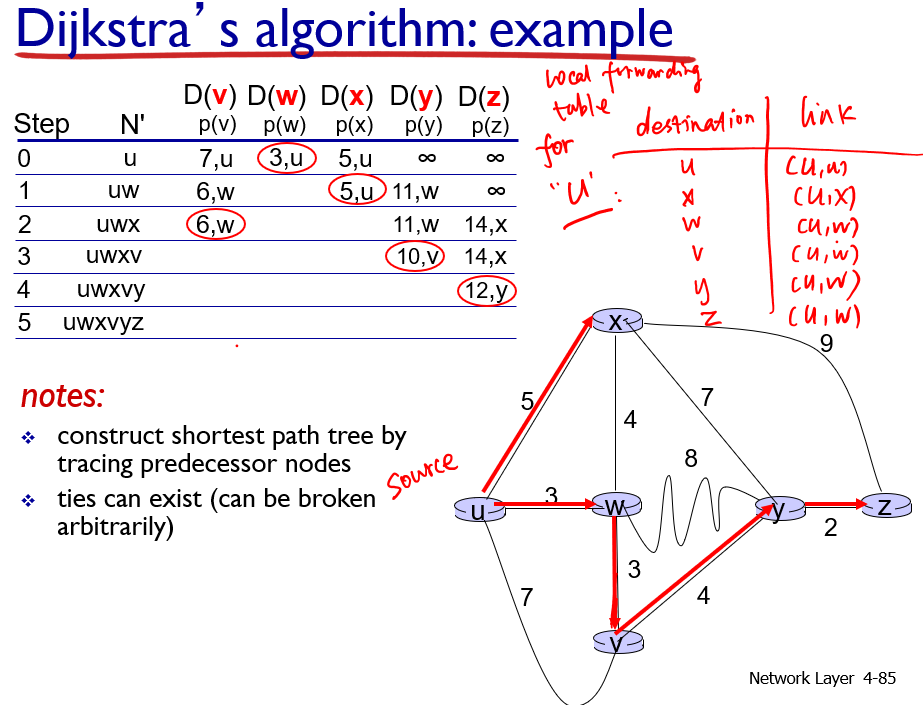
***Frg data*** = MTU – hdr = 1500 – 20= 1480.

***# of fragments*** = Orig Data/Frg Data= 3980/1480 = 3.

**Data in last frag(Leftover)** = 3980 -2x1480 = 1020

Offset has 8B bound. Thus **Offset #** = 1480/8 = 185. 

fragflag = 1, last fragflag = 0.

IPv4 addressing - IP address: 32-bit identifier for host, router interface, IP addresses associated with each interface. (usually broken into 4 8 bit parts, separated by a decimal, ex)223.1.1.1 ) Subnet - Devices that interface with same subnet part of IP address(Can physically reach each other without intervening router.) determine subnets by detaching each interface from host or router creating island of isolated networks. CIDR: Classless Inter Domain Routing - address format: a.b.c.d/x, where x is # bits in subnet portion of address. Total # of bits is 32, find host by subtracting 32 - x. ex) given a subnet such as 138.65.89.0/22: 22 bits used to denote the subnet part, 10 bits to denote host. Up to 2^10 IP addresses.

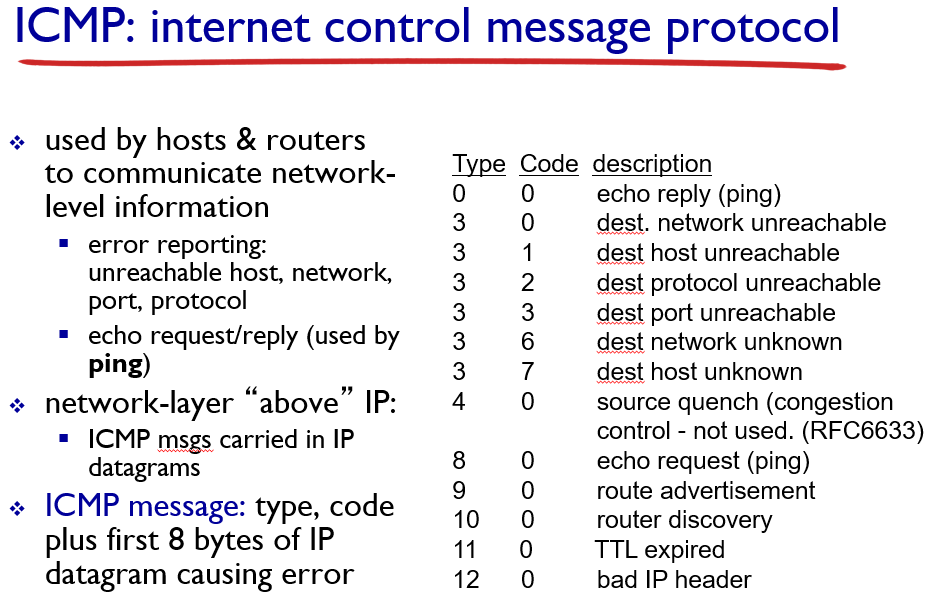
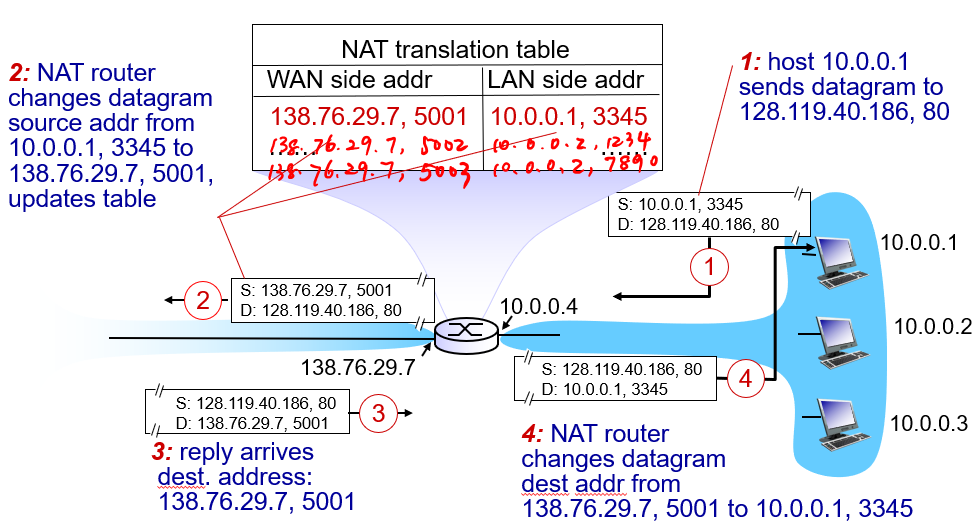
DHCP:Dynamic Host Configuration Protocol - allow host to dynamically obtain its IP address from network server when it joins network. Can ask for more than IP, could include address of first-hop router and IP address of DNS server, network mask. DHCP Steps to get IP:

1)host broadcasts “DHCP discover” msg [optional]

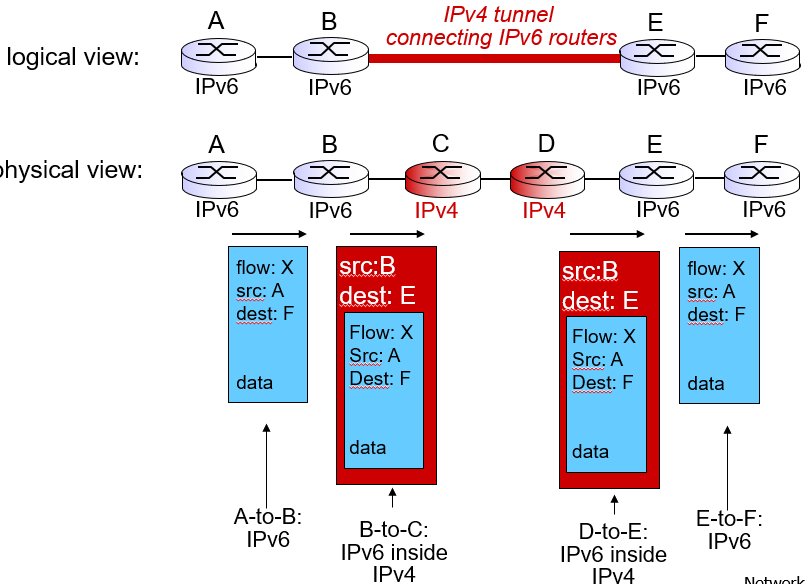
2) DHCP server responds w“DHCP offer”msg [optional]

3) host requests IP address: “DHCP request” msg 4) DHCP server sends address: “DHCP ack” msg

Hierarchical addressing - Allows for more efficient advertisement of routing. example) from internet -> ip -> organization (“send me Send me anything with addresses beginning w/ …. or …..)

NAT: network address translation all datagrams leaving local network have same single source NAT IP address ex)138.76.29.7, different source port numbers. Motivation: only need 1 IP for all devices, can change in side without notifying outside world, more secure. ICMP IPv6: allows for longer IP addresses because 32 bit address space completely allocated soon.  
 IPv6 datagram format: Fixed-length 40 byte header

no fragmentation allowed. tunneling: IPv6 datagram carried as payload in IPv4 datagram among IPv4 routers

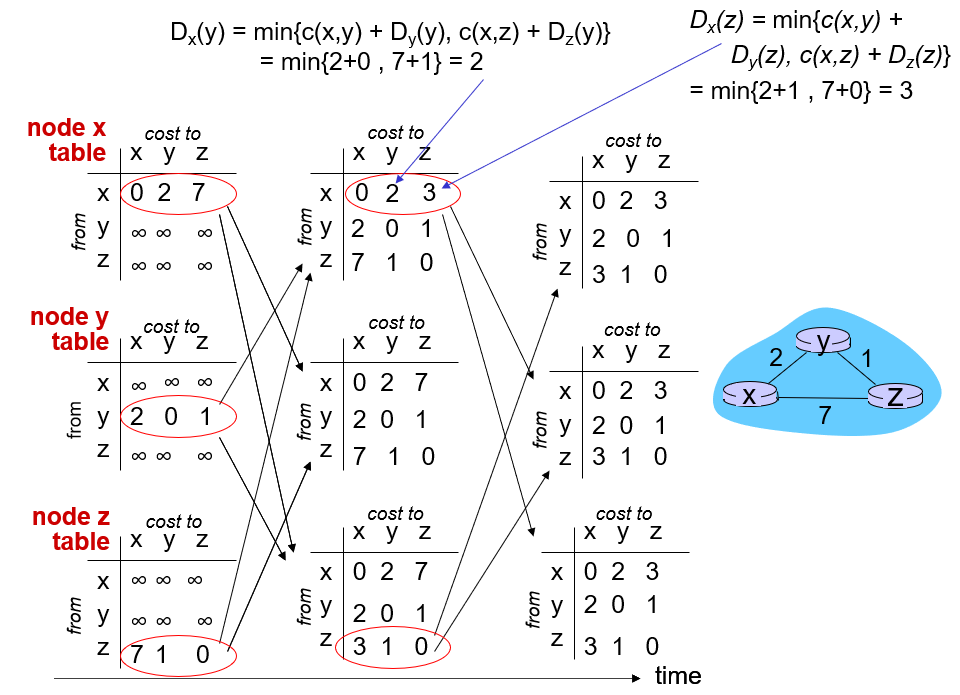
**4.5 Routing algorithm**  link state algorithm/Dijkstra

c(x,y):link cost from node x to y; = ∞ if not directly adjacent

D(v): current value of cost of path from source to dest. v

p(v): predecessor node along path from source to v

N': set of nodes whose least cost path definitively known. Start with initial node in n’, DV chart at ∞. Replace DV with path if smaller than current path, include last node where came from. After you filled in all the DVs you can at that node, add shortest path to N’ and repeat w/ new node until all elements are in N’

Distance Vector Algorithm/ Bellman-Ford:

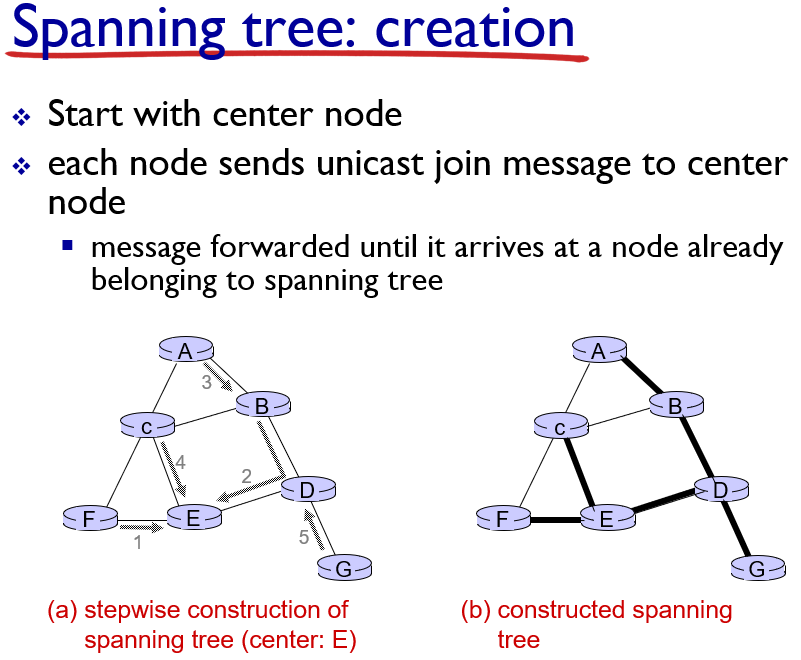
dx(y) := cost of least-cost path from x to y = dx(y) = minv {c(x,v) + dv(y) }. minv is the total taken over all neighbors v of x, c(x,v) = cost to neighbor v.

dv(y) = cost from neighbor v to destination (y)

Hierarchical routing - aggregates routers into regions, “autonomous systems” (AS)Intra-as routing - Tasks and protocol within an AS, can be different to another AS Inter-as routing - Tasks and protocols outside of AS

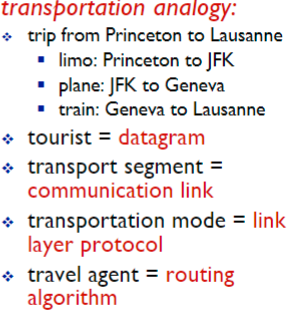
**4.6 routing in the internet** OSPF(Open shortest path first) - opens publicly available paths and floods advertisement to entire AS.uses link state algorithm, Intra-as routing. BGP(Border Gateway Protocol) - the de facto inter-domain routing protocol,allows subnet to advertise its existence to rest of Internet, inter-as routing. Two types of BGP - eBGP, used to obtain subnet reachability information from neighboring ASs, iBGP used to propagate reachability information to all AS-internal routers. BGP route selection rules - The router may learn about more than one route to destination AS, selects route based on 1. Local preference value attribute, policy decision. 2. Shortest AS-path 3. Closest next-HOP router: hot potato routing. 4. Additional criteria.

**4.7 Broadcast:** Spanning tree is a method of how to prevent redundant packets received by node

**Chapter 5 Link Layer: 5.1 intro services**

Host and routers = nodes, Communication channels that connect adjacent nodes = links (wired/wireless)

Layer-2 packet = frame (encapsulates datagram)

Data-link layer responsible for transferring datagram from node to physically adjacent node over link.

Link Layer Services: Framing: Encapsulate datagram, adds header and trailer. Provides channel access if on shared medium. MAC address in frame header to identify source, dest. Reliable delivery btwn adjacent nodes: used in wireless links bc high error rates, not for low bit-error links. (fiber). Flow control: pacing btwn adjacent nodes. Error Detection: caused by signal attenuation, noise. When receiver detects error it signals sender for retransmission or drops frame.

Error Correction: Receiver find and corrects bit errors without retransmission. Half-duplex: both ends can’t transmit at the same time. Full-duplex: Can both do it.

**5.2 Error Detection, correction**

Parity checking: count number of 1’s, if different to parity bit then there is an error.

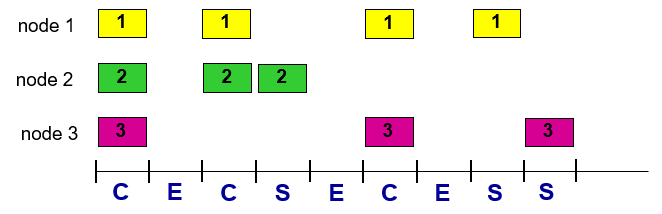
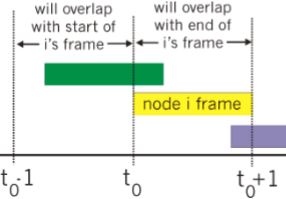
Even parity: add extra parity bit to data such that total number of 1’s is even.

Odd parity: add parity such that total bits is odd.

single bit parity: detect odd number of bit errors

two-dimensional bit parity: can correct single bit errors and detect multiple bit errors. Cyclic Redundancy Check: Why? Bc its a more powerful error-detection method. Can detect all burst errors less than r+1 bits - message represented by some polynomial D, and a generating polynomial G that is r+ 1 bits - XOR Divide G by D\*2^r to get remainder R If D\*2r XOR R = G, then data good.  **5.3 Multiple Access Protocols**

**2 types of links**: point-to-point and broadcast.

MAC Protocols: Channel partitioning: divide channel to pieces. (time slots, frequency, code) Allocate piece to node for exclusive use. 2 methods, (1) Time division multiple access (TDMA): Access in rounds, each station gets fixed length slot per round. Unused slot is idle. (2) Frequency division multiple access (FDMA): Channel spectrum divided into frequency bands. Each station assigned fixed freq. Band. Unused trans. time in freq. band goes idle. Random Access: Not divided channel, allows collisions and recovery. When node needs to send pkt it transmits at full channel data rate R. No coordination among nodes, 2 or more transmitting is a collision. Will have a method to detect and recover from collisions. (Ex of random access MAC protocols: The 4 protocols after this line.) Slotted ALOHA: Transmits data on a clock cycle in slots as long as there are no collisions. Pros: can transmit full rate of channel, decentralized, simple Cons: collisions and idle slots, which waste time and space. Nodes must detect collisions Slotted ALOHA max efficiency = 1/e = .37 Pure ALOHA: Transmits data when frame arrives immediately as long as there are no collisions, but has collision issues = poor efficiency (Max efficiency = .18) 

CSMA (carrier sense multiple access) - listens before transmit: if channel sensed idle: transmit entire frame. if channel sensed busy, defer transmission (less collision than in pure, but still has some) CSMA/CD (collision detection) - adds collision detection to CSMA, better efficiency than ALOHA. - collisions detected within short time. - collision transmission aborted, reducing channel waste. “Taking Turns”: nodes takes turns, longer msg = longer turns. Polling: master node lets slave nodes transfer in turn, slaves are usually simple “dumb” devices Concerns - polling overhead, latency, single point of failure Token passing: data gets passed around sequentially Concerns - overhead, latency, single point of failure.

**5.4 LANs:** 32-bit IP address: Network-layer address used for forwarding. MAC address (48 bit): Used to get frame from interface to another physically-connected interface. (same Ntwrk, in IP addressing sense). MAC address allocation done by IEEE. Manufacturer buys part of MAC address space. (to assure uniqueness)

MAC vs. IP: MAC=SSN, IP=postal address. MAC is portable. (move LAN card) IP address is hierarchical, not portable. (depends on IP subnet to attached node)

Address resolution protocol (ARP): Used to determine MAC, knowing its IP address. Each IP node (host router) on LAN has an ARP table to map IP -MAC. TTL= time till address mapping is forgot.(usual 20min)<IP; MAC; TTL>

ARP Same LAN: **A** wants to send datagram to **B**, but doesn’t have **B**’s MAC in ARP table. **A** broadcasts ARP query pkt to **B**’s IP. **B** replies to **A** with MAC. **A** caches IP-MAC pair to ARP table. ARP Diff LAN: **A** makes IP datagram to **B**. (must go thru router **R**) **A** makes link-layer frame with **R**’s MAC as dest, frame contains IP datagram for **B**. **R** makes link-layer frame of **A-B** datagram with **B**’s MAC as dest.

Ethernet frame structure: Ethernet’s MAC protocol - unslotted CSMA/CD with binary backoff. Connectionless (no handshake) btwn NICs, unreliable bc does not send ACK or NACK. Recovery from higher layer rdt. (e.g., TCP)

Preamble: 7 bytes 10101010 followed by 1 byte 10101011. Used to sync receiver, sender clock rates.

Addresses: 6 byte source, dest MAC. If adapter receives frame matching dest or w broadcast address, it passes data to network layer protocol, otherwise discards.

Type: indicates higher layer protocol. (mostly IP)

CRC: Error detection check at receiver. (5.2)

Ethernet Switch: Store, forward frames. Checks MAC, selectively forwards, uses CSMA/CD to access segs.

Self-learning: learns what hosts can be reached, records sender/location pair in switch table.

Switches vs. Routers: Both store-forward, **R**-network layer, **S**-link layer. Both have forwarding tables, R-compute tables using routing algorithm (IP), S-flooding learning to make tables (MAC).

Virtual Local Area Network (VLANS): Multi LANS in single physical LAN, helps security/privacy, efficiency issues. Port-based VLAN: Use ports to separate groups, isolate traffic,dynamic membership,frwrd btwn VLANS.

Trunk port: carries frames btwn VLANS over switches.

**5.6 Data Center:** 100’s thousands of hosts in proximity. (Amazon, Youtube, Google) Load balancer: app-layer routing. Avoid processing, networking, data bottlenecks, interconnected switches (racks) to help throughput. This increases reliability via redundancy.

**5.7 a day in the life of web request**

protocols are used at each layer: application: FTP, SMTP, HTTP transport: \*TCP, UDP network: IP, routing protocols, Link Ethernet, 802.11 (WiFi), PPP.

Understand the entire process for the given scenario: student attaches laptop to campus network, requests/receives www.google.com

A/1) Host needs to get IP addr, addr of first hop router, addr of DNS Server - use DHCP.

Why broadcast in DHCP: to get the info DHCP is for IP

2) DHCP encapsulated in UDP, in IP, in 802.3 Ethernet

Why Using UDP for DHCP: because DHCP is to get IP, TCP needs to be connected together first

3) Ethernet frame broadcasts (dest: FFFFFFFFFFFF) on LAN, which is received at router running DHCP server,

How to do broadcasting at link layer? Self learning

ethernet signal demuxed to IP demuxed to UDP, demuxed to DHCP to connect to DHCP server

4) DHCP server formulates DHCP ACK containing client’s IP address, IP address of first-hop router for client, name & IP address of DNS server

5) DHCP client receives DHCP ACK reply, information returned in DHCP response message: Client now has IP address, knows name & address of DNS server, IP address of its first-hop router. Resolution of google.com, DNS: B/6) Now the host needs IP address of [www.google.com](http://www.google.com), gotten through DNS - DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth (ARP). How could the frame be sent to the first-hop router at link layer: need to get MAC address of first-hop router, received through ARP. 7)ARP query broadcast, received by router, which replies with ARP reply giving MAC address of router interface. Client now knows MAC address of first hop router, so can now send frame containing DNS query . 8) IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router. 9) IP datagram forwarded from campus network into comcast network, routed (tables created by RIP, OSPF, IS-IS and/or BGP routing protocols) to DNS server 10) DNS server replies to client with IP address of [www.google.com](http://www.google.com) 11) to send HTTP request, client opens TCP socket to web server and does the 3 way handshake.11-1) SYN segment routed to server SYN= 1, ACK = 0. 11-2) web server responds with TCP SYNACK ,SYN = 1 ACK = 1. 11-3) Send ACK of SYNACK, TCP connection established! SYN = 0 ACK = 1. C/12) HTTP request sent into TCP socket. 13) IP datagram containing HTTP request routed to [www.google.com](http://www.google.com). 14) web server responds with HTTP reply (containing web page). 15) IP datagram containing HTTP reply routed back to client.