Internets

Packets:

- Packet = data + metadata
- Can store and forward
- Hard to control delay
- Need switch buffer memory
- Flow convergence can create congestion

End to end principle:

- Do as much as possible at the edge: dumb network
- Lower layer protocols only justified as optimization

Decentralization:

- Allows scaling but precludes reliability!
- No uniform solution for accounting and billing
- In practice routing based on local not global optima

ATM

Virtual circuits:

- Overcome fixed bandwidth allocation of synchronous transmission mode (STM) with packets
- Carry a local virtual circuit identifier which is translated into new link-specific ids by retransmitters: fast hardware lookup
- Can preestablish virtual paths (bundles of routed circuits)
- No flow or error control by themselves, but maintains ordering
- Necessary setup can delay short messages: mitigate with a virtual path, sending data with setup packet or VCI for datagrams

Fixed size packets:

- Simple buffer hardware and scheduling
- Internal fragmentation for small pieces of data
- Segmentation and reassembly cost
- Tradeoff between packetization delay and header overhead

Provides integrated services (useful for telcos)

ATM Adaptation Layers:

- AAL1: provides timestamps, clocking, sequencing, FEC for synchronous apps
- AAL3/4:
  - Provides segmentation, reassembly and error detection for data traffic
  - Has header and trailer per Encapsulated Protocol Data Unit and a per cell header
- AAL5: violates layering to reduce overheads of AAL3/4 bits in cell headers mark the end of a frame

IP-over-ATM:

- Treats ATM as a link level technology, but must work around point-to-point connection oriented nature
- Place entire IP packet within AAL5 frame
- Designate one host as an ARP server which all clients stay connected to
- The ATM network is partitioned into logical IP subnets to reduce the cost of broadcasts which are expensive on ATM, but this increases path length
- However, next-hop routing stores IP-ATM translations independent of subnet boundaries and reduces it again!
- A Multicast Address Resolution Server provides and updates a translation from Class D IP addresses to endpoints
- Holding time problem: when to close an ATM connection to an "IP" after opening it? Depends on traffic distribution

Can also have ATM-over-IP to reuse inexpensive Ethernet adapters, but it does require an attachment device to decant these cells into a real ATM network.

Protocols

Set of rules and formats that governs communication between communication peers. Defines syntax, semantics and actions. Typically layerable to form a stack, which decouples changes and hides information. This allows composition and reuse but can reduce performance.

Service access point: interface between an upper layer and a lower one.

Service data unit: packet handed to a layer by an upper layer.

Protocol data unit: packet exchanged between peer entities, typically = SDU + header or trailer.

Openness requires:

- Public protocol details
- Changes managed by an organization whose membership is open to the public

ISO OSI

A reference model and service/protocol architecture

Physical layer: moves bits between physically connected systems (coding schemes, connector shapes etc)

Datalink layer: introduces frame (group of associated bits), may require an addressing scheme

Network layer: concatenates a set of links to form an end-to-end one, introduces network addresses

- Datagram: does routing and forwarding
- Connection oriented: data plane forwards and schedules, control plane does routing, setup/teardown

Transport layer: adds multiplexing, error and flow control

Session layer: provides full duplex service, expedited delivery and synchronization with rollback (uncommon)

Presentation layer: hides data representation differences (e.g. endianness, encryption)

Application layer: useful application protocol (e.g. HTTP)

System Design

Bottleneck: most constrained element in a system. If every component is bottlenecked the system is *balanced*.

Resources impose constraints on design. We try and trade them off against each other to alleviate the bottleneck. Example resources:

- Time: deadline, time between failures, response time, throughput, parallelism degree
- Space: limit to memory, capacity
- Computation: processing done in unit time
- Money: constraints components, number of engineers available
- Labor: human effort
- Social: standards, market requirements (e.g. backwards compatibility, OS support)
- Scaling: design constraint

Multiplexing:

- Sharing, use a schedule to resolve contention
- Statistical multiplexing (tradeoff capacity for delay if peak rate is not average rate)
  - Spatial: only a fraction of all tasks are simultaneously active
  - Temporal: tasks are active only part of the time

Pipelining:

- Break task into independent subtasks so you get hardware reuse
- If the slowest stage takes time S throughput is $\frac{1}{S}$, say response time is R so parallelism is $\frac{R}{S}$

Locality: spatial and temporal

Binding: translation from an abstraction to an instance

Virtualization: combines indirection and multiplexing, virtual resources matched to instances

Randomization: lets us break ties fairly

Soft state: deleted on a timed basis unless it is refreshed

Hysteresis: use a state-dependent threshold to deal with fluctuations near it

Naming And Addressing

Resolution: determine an address from a name

Indirection: can have a many-to-many map between addresses and names, move machines transparently

Name resolution:

- Centralized
  - Consistent but a single point of load and failure
- Hierarchical
  - Based on recursive decomposition
  - Scale well and guarantee name uniqueness by prefix property
  - Used by DNS
    - Name server is authoritative for a set of domains
    - Delegate responsibility to a child name server
    - Root servers replicated
    - Replies cached and timed out
    - Can request server to perform recursion: this builds up server cache
  - Used by IANA

        - Original host/network hierarchy replaced with one based on CIDR with locally visible masks and subnetting (clustering addresses within a network)
        - Hierarchy enables routing table route aggregation
    - Used by ATM for network service access points (variable length!)
- Distributed
    - Used by Ethernet for MAC discover (i.e. ARP), however susceptible to spoofing due to untrusted caching

Other techniques:

- DHCP: computer broadcasts to obtain offers, requests one, releases the lease when done
- NAT: aggregates addresses using source port mapping (multiplex network level at transport level)

Routing

The process of finding a path from a source to every destination in the network. The key problem is that to make correct local decisions each router must know something about the global state, which is inherently large, dynamic and difficult to collect. We want a robust algorithm that avoids black holes, loops and oscillations, and which uses optimal paths.

- Centralized vs. distributed (simplicity vs. failure and congestion)
- Source based vs. hop-by-hop (intermediate is loose-source routing)
- Stochastic vs. deterministic (load spreading vs. misordering)
- Single vs. multiple paths (primary and alternative?)
- Network state dependence of protocol

Telephone networks:

- Three level hierarchy with a fully connected core
- Stable load, switch reliability and single point of control means we can choose optimal routes in advance
- Costly due to reliability and full connectedness, attempted to be fixed in ATM
- Dynamic Non-Hierarchical Routing:
    - Divide day into periods, in each period assign each switch a primary one-hop path and some alternatives to overflow to if needed
    - If all alternate paths are busy crankback to previous level
    - Doesn't deal with with unexpected traffic patterns
- Trunk Status Map Routing:
    - Like DNHR but updates traffic measurements one per hour rather than per week if necessary
    - Metastability prevented by preventing spilled traffic taking over direct path
- Real-Time Network Routing:
    - No centralized control: each toll switch maintains a list of lightly loaded links at each node
    - Intersection of source and destination lists given sets of lightly loaded paths

Packet networks:

- Distance vector routing
    - Node tells its neighbors it's best idea of distance to every other network node
    - Node updates its notion of best path to any destination as it receives distance estimates from all other nodes
    - Distributed and adaptive, if network is static then it converges with iteration due to Bellman-Ford property
    - Count to infinity problem due to link failure
        - Split horizon: don't tell neighbor cost to X if that neighbor is the next hop to X (doesn't work in general)
        - Triggered updates: send updates on change rather than timer
        - Reduce the size of infinity
    - Used in the Internet as RIP and EGP
- Path vector routing
    - As distance vector, but annotates shortest paths with entire path
    - Loop-free
    - Used in the Internet as BGP
- Link state routing
    - Each node knows the entire network topology and finds shortest path itself
    - Use controlled flooding to distribute the local link state to others
    - Sequence number schemes

        - Aging: must wait for old LSPs to die after reboot
        - Lollipop sequence numbers:
            - Start at $-\frac{N}{2}$, progress to 0 and then wrap around to 0 at $\frac{N}{2} - 1$
            - a is older than b if $(a < 0 \wedge a < b) \vee (a > 0 \wedge a < b \wedge b - a < \frac{N}{4}) \vee (a > 0 \wedge b > 0 \wedge a > b \wedge a - b > \frac{N}{4})$
            - Since routers who get older LSPs tell the sender about the newer one,$-\frac{N}{2}$ can act as a trigger to evoke a response from community memory
    - Heartbeat to detect failure but must age LSPs anyway in case of dead router traffic
    - Used in the Internet as OSPF and PNNI

- Hierarchy
    - Divide network into domains connected by gateways which only talk to other gateways
        - Exterior and interior gateway protocols reflect different trust levels
    - In the Internet, core only advertises routes to networks

Link costs:

- Static: weight proportional to capacity or constant
- Dynamic
    - Router queue length
    - Must correct for wide dynamic range and transient spikes
    - Negative reinforcement leads to oscillation
- Multiple costs may be setup per link
    - Remote routers must use the same rule in constructing paths or we risk forming loops

Crankback: if there is no next hop with sufficient QoS return the packet to the previous level

Mobile routing:

- Can be done with a home station that forwards packets to the current base station and sends packets on your behalf
- Old care-of agent may forward incoming packets to the new care-of agent until the home learns of the change, but this may lead to loop formation.

Multicast Routing

Multicast groups associate a set of senders and receivers with each other, where the senders do not need to know the receivers identities. They are created on demand.

We can use multicast with an increasing TTL to perform expanding ring search.

We want to send one multicast packet per link, which means we must form a multicast tree rooted at the sender: ideally this will be the shortest-path tree.

IP includes a range of multicast addresses encoded by setting the multicast bit and a unique identifier in the lower bits. This identifier is copied into a MAC address which is broadcast on by Ethernet.

IGMP: router periodically broadcasts a query message and hosts reply with the list of groups they are interested in: allows us to prune shortest path tree.

Flooding:

- Forward packet if a router has not seen it before regardless of group membership
- Simple and always works, but requires router storage and eats bandwidth
- Reverse Path Forwarding
    - Forward a packet from S to all interfaces iff the packet arrived on the interface corresponding to the shortest path to S
    - You don't need to send a packet downstream if you are not on the strictly shortest path from the downstream router to the source
- Pruning: routers explicitly notify their parent that they do not (or, later, do) wish to receive traffic for particular groups
- Tunneling
    - IP-in-IP for getting around multicast unaware hosts that discard multicast addresses
    - For the purposes of RPF we must consider shortest paths only taking multicast capable routers into account

Core Based Trees:

- Nominate a core router which receives all join requests
- Routers along join request path mark the incoming interface for forwarding

- Routers which are not part of a group don't have to prune, and membership changes are fast due to explicitness. However, all traffic travels via the core which acts as a bottleneck and ensures paths are rarely optimal

Protocol Independent Multicast:

- With a dense multicast tree, use flood and prune since only a few prunes are required
- With a sparse tree use modified core based trees
  - Multicast sources attempt to discover a shorter path to receivers by consulting routing tables, only going via the core if necessary
  - Leaf routers set a timer upon receiving a heartbeat message from the core (aka rendezvous point): if it expires then send a join request to alternate rendezvous point

Error-Control

Packet errors

- Loss, duplication, insertion, reordering
- Detect with sequence numbers and timeouts, correct with retransmission
- Sequence space should be at least as large as (transmission rate) * (2 * maximum packet lifetime + timeout + receiver delay time) or without ACKs depends on the reordering and burst loss spans

Packet insertion detection is subtle:

- Per-connection incarnation number (takes up header space, needs persistent storage)
- Only reassign port numbers after an MPL (needs persistent storage)
- Assign initial sequence numbers serially (needs persistent storage)
- Wait an MPL upon bootup to flush old packets from the system!
- Standard solution is the 3-way handshake: communicating ends tell each other an initial sequence number and protect against delayed SYN
  - Client → Server: SYN(c)
  - Server → Client: SYN(s), ACK(c+1)
  - Client → Server: ACK(s+1)

Loss detection:

- With NACK or timeout with cumulative ACKs (needed anyway since NACK may be lost!)
- The timeout should approximate the RTT, but we can obtain it statically or dynamically
- Using timeout conflates delay and loss, may be subject to high RTT variability and leads to self-blame in flow control
- Original TCP scheme:
  - RTTs are measured periodically and updated as $rtt_{estimate} = a * rtt_{estimate} + (1 - a) * rtt_{measured}$
  - Timeout is a constant multiple of the RTT estimate
- Jacobson TCP scheme:
  - Introduce $m_{measured} = |rtt_{estimate} - rtt_{measured}|$, $m_{estimate} = a * m_{estimate} + (1 - a) * m_{measured}$
  - Now timeout is just $rtt_{estimate} + bm_{estimate}$ for constant b

Retransmission:

- Typically window based
- Go-back-N: upon timeout retransmit the entire window, the client need only accept in-order packets
- Fast retransmit: if the sender sees repeated cumulative ACKs, a packet is most likely lost so send the lost one
- Selective retransmission can be done a header bitmap either sent back in every packet or asked for by sender
- SMART: ACK carries the cumulative sequence number and packet number prompting ACK, so sender can create the bitmap

Flow Control

Open loop:

- Source describes the desired flow control rate
- Network admits the call and the source sends at its rate, being policed by the network

Closed loop:

- Source monitors the available service rate and sends at that rate
- Can determine the service rate explicitly or implicitly

- First generation: only matches the receiver, typically hop-by-hop
- Second generation: responsive to state, typically end-to-end

Traffic descriptions:

- Representativity: adequately describe flow
- Verifiability: so that the network can police it
- Preservability: doesn't change inside the network
- Usability: humans are involved too!
- Typical examples constraint worst case behavior:
  - Peak rate
  - Average rate over a jumping or moving window
  - Linear Bounded Arrival Process
    - The number of bits transmitted in any interval of length t is less than $rt + s$ for burst limit s
    - Can be regulated with a leaky bucket: data fills up a bucket of size s which "leaks" at a rate r
    - Token buckets have tokens which arrive at rate r into a bucket of size b: tokens may be consumed to transmit at a maximum rate of p
    - There is a tradeoff between r and s

Typically we abuse the error control window size to perform flow control because if the window is exhausted the sender must stop. However, this can be coarse grained and causes coupling between the two concerns.

Two special cases of this are Stop-and-Wait and Static Window with window size at least bR for round trip time R and bottleneck packet rate b.

X-On, X-Off: bursty and suffers if OFF is lost, but fine if RTT is small

DECbit:

- Every packet has a bit in its header, which intermediate routers set if a queue has built up
- Bit is copied to the ACK by the sink, which the source can use to reduce window size
- Routers will set the bit if a source is making more than the mean capacity demand or it is becoming overloaded

TCP Flow Control:

- Window size controlled by ACK timeouts (and fast retransmits), which approximate losses
- "Slow start" phase begins by doubling window size every RTT as long as there are no timeouts
- "Congestion avoidance" phase increments the window size ever RTT as long as there are no timeouts
- On loss reset window size and switch to "slow start" phase
- Switch phases upon a preset threshold being reached
- TCP Vegas instead measures actual throughput and compares it to an expected throughput (window size / propagation delay) to decide if to change window size

NETwork Block Transfer:

- Application data is sent as a series of buffers, each with a given rate
- If the received rate is not as high as expected we multiplicatively reduce the rate

Packet Pair: assumes bottlenecks serve packets in round robin order, so the spacing between packets at the receiver $= \frac{1}{\text{rate of slowest server}}$

End-to-End Rate-based Flow Control:

- Similar to DECbit, but routers can add an entire cells data instead of a single bit
- Sources periodically send a Resource Management frame with a rate request, which is filled in with the current share by the each router

Multiple Access

Circuit-mode vs. packet mode: do we need to deal with bursts or streams of packets?

Parameter a: the number of packets sent by a source before the farthest station receives the first bit

Performance metrics:

- Normalized throughput: fraction of link capacity used to carry non-retransmitted packets
- Mean delay: amount of time a station has to wait before it successfully transmits
- Stability: under heavy load, is all time spent resolving contentions?
- Fairness: usually means no starvation

Plesiochronous hierarchy: streams being multiplexed are allowed to run slightly faster or slower than some synchronous limit. Overhead in the output stream is used to pad and identify bits from a scheme which is transmitting slightly faster than the limit. This makes routing and unframing difficult.

Technologies:

- Multiplexing
  - FDMA: simple, but number of frequencies limited
  - TDMA: needs time synchronization with the associated overhead, problem of multipath interference on wireless which changes phases
  - CDMA: no hard limit on capacity, but complex to implement and requires a large contiguous frequency band
- Duplexing (for wireless channels)
  - Frequency Division Duplex: downlink and uplink use different frequencies
  - Time Division Duplex: downlink and uplink use different timeslots

Schemes:

- Centralized:
  - Only transmit when master allows
  - This is simple but a single point of failure and a delay source
  - Polling: stations are asked to see if they want the slot (if a is high, you may just give it to them)
  - Probing: master uses binary search to find a station that wants to transmit
  - Reservation based schemes may be useful if a is high, but some will incur overhead from minislot transmission and contention
- Distributed:
  - More reliable, have lower delays and allow higher utilization!
  - Polling: each station is assigned a slot that it may use to transmit, but setting up slot and timebase is tricky
  - Probing: All stations in left subtree of root place packet on medium - if there is a collision then iterate with root being the left son of the current root, else everyone in the right subtree places a packet on the medium etc
- Carrier Sense Multiple Access
  - Check whether the medium is active before sending: only works if a is small
  - A scheme is *persistent* if it waits for an idle line after finding it is busy, but this doesn't work well for contention resolution
  - Instead *p-persistence* can be used, where you transmit upon idle with probability p
  - Alternatively, set a timer after detecting busyness with a period that grows exponentially: *exponential backoff*
  - Ethernet refines this by requiring packets are long enough that collision can be detected before transmission completes, and it jams the bus upon collision to ensure it is detected everywhere
- CSMA/CA
  - In wireless LANs you cannot detect collisions because the transmitter overwhelms colocated receivers, so explicit acknowledgments are required
  - Upon the medium becoming idle, wait for up to the contention window, transmit the message and wait for an acknowledgment: if it is not acknowledged then try again
  - BTMA: to deal with hidden terminals a separate busy tone channel is transmitted by the base station when it is receiving: this allows all connected devices to detect that their message would be lost if they transmitted
  - MACA: uses messages in-band rather than a separate channel to indicate contention since split frequencies may have different propagation characteristics!
- Token Passing
  - Allows us to quickly skip past idle stations
  - Receiver sets the ACK flag in the packet so it can be deleted by the source (ensures fairness)
  - Can use a physical double ring so that wrap mode can be used if a link fails
  - Extremely simple and fair, but the token is a source of failure and stations must cooperate
- ALOHA
  - Just send the data and wait for an ACK: if it doesn't arrive, try again after a random time
  - Useful when a is large as carrier sensing won't help, but goodput is at most .18 at high loads and potentially unstable
- Slotted ALOHA: make sure that transmissions start on a slot boundary to halve the window of vulnerability
- Reservation ALOHA
  - Contend for reservation minislots using slotted ALOHA

  - The reservations are inspected independently to decide upon a consistent schedule
  - This supports circuit and packet mode transfer and works for large a, but cannot preempt hogs and arriving packets must wait for the entire frame

Switching

Try and minimize packet loss and maximize throughput while attempting not to reorder packets. Should also strive for packet or byte fairness.

Circuit switching:

- Blocking
  - Internal blocking: a slot in the output frame exists but no path
  - Output blocking: no slot in the output frame is available
- Time division switching
  - When demultiplexing, the position in the frame determine the output trunk
  - Sample position within a frame is changed by a Time Slot Interchange which is typically a memory
- Space division switching
  - Each sample takes a different path through the switch depending on its destination
  - Crossbar: internally nonblocking but $O(n^2)$
  - Multistage crossbar:
    - Save crosspoints by attaching them to more than one input line and reconfiguring the crossbar every switching time
    - Could have e.g. two 10×3 crossbars, then three 2×2 ones and finally two 3×10 crossbars
    - Can suffer from internal blocking unless there are sufficient second level stages
- Time-space division switching
  - Precede input trunks in a crossbar with a TSI
  - Delay samples so that they arrive at the right time for the space division switch's schedule
- Time-space-time division switching: flip samples with a TSI on both input and output trunk for even more flexibility

Packet switching:

- Multiple levels
  - Repeaters: physical level (e.g. amplifier)
  - Bridges: datalink level (e.g. Ethernet bridge)
  - Routers: network level (e.g. IP router)
  - Application level gateways: application level (e.g. HTTP proxy)
- Generations
  - First generation: CPU and main memory with lots of line cards attached
  - Second generation: CPU with line cards with memories which contain the port mapping intelligence
  - Third generation: Self routing fabric between input and output ports, coordinated by a processor
- Need to look up the destination port on the fly but otherwise similar
  - Can use a trie for this, with recent accesses cached in a CAM
- Cannot predict if the switch will block, so overprovision or add buffers
- Variable sized packets must be fragmented, routed as chunks and then reassembled
- Crossbar: compute schedule in advance
- Buffered crossbar: buffer the crosspoints of the crossbar to ensure no loss
- Broadcast: packets are tagged with a port number and the output matches against a flow of packets on a bus
- Switch element based fabrics:
  - A switch element examines a bit in front of a packet: if 0 it goes to the upper output else the lower output. If both packets want to go to the same output, buffer or drop
  - This can be used to push the configuration into the network with the packets themselves, just by tagging them correctly
  - A Banyan switch is based on a regular grid of switch elements based on binary digit encoding at the start of the packet encoding which should be the output port
  - You can check if a path is available before sending with a 3-phase request/inform winner/send scheme, and even have one Banyan doing that while another is making use of the output ports
  - Alternatively, you can have multiple Banyan fabrics in parallel, each of which handles the collided packets from the previous round, but this gets expensive

    - Sorting: present packets at inputs sorted by output with duplicates and gaps removed - the actual sorting can be done by recursive pairwise sorting and merging networks
    - Batcher Banyans: these sort, trap duplicates, shuffle and then route packets, with duplicates being recycled

Buffering:

- Input buffers
    - Experience head of line blocking (so utilization is at most 58.6%)
    - This can be solved with per-output queues at inputs
    - Possible arbiter is one based on parallel iterated matching: output selects one interested input, and if some input has more than one grant the input picks one at random - losers try and match again
- Output buffers
    - May need to run much faster than trunk speed
    - Knockout principle: it's unlikely that all N inputs will have packets for the same output, so drop extra packets fairly
- Shared memory: just route the header to output and index into memory with that, but doesn't scale well
- Buffered fabric (as above): backpressure can reduce buffer requirements but costly

Integrated Services

Requirements:

- QoS with traffic descriptions and policing
- Service interface
- Admission control
- Scheduling

Challenges:

- Differing traffic patterns of applications
- Router friendly large packet sizes slow real time traffic
- Small packets have less delay and are more loss tolerant but less efficient and not router friendly
- Delay dependent on buffering, processing, transmission, propagation, unknown paths, localized traffic..
- Jitter can be significant due to flow-unaware FIFO queuing, router load and dynamic path changes
- Loss due to retransmission, failure, traffic violations and reordering
- Data rate changes due to path changes, congestion and traffic aggregation

Elastic applications are broadly those that do not need to make use of QoS: they include interactive (telnet), interactive bulk (HTTP) and asynchronous (e-mail) traffic classes

Inelastic applications are real time and either intolerant or tolerant. If tolerant, they can be adaptive or non-adaptive to rate and delay changes.

Integrated Services and RSVP:

- Packets marked by the application by its participation in RSVP, applied per flow
- Flow specifications are sent through the network
- Receivers confirm unidirectional reservation which is stored in soft state

Differentiated Services:

- Packets are marked by the network to provide expedited (low delay) or assured (high data rate and low loss) connectivity
- Fundamentally applied per customer/user-group
- Traffic conditioners operate to impose policy, marking traffic as in or out of profile and shaping or dropping the packets
- Problems include the fact that there is no standard for what the DS codepoints mean between providers, so guarantees are few, and the QoS is asymmetric

RTP/RTCP

- Carried in UDP packets without reliability necessarily
- RTCP provides feedback to senders and receivers, including information on loss, delay and jitter so that this functionality does not have to be duplicated

Pricing: try to create incentives not to always ask for the best QoS through billing schemes - do we want to price based on congestion?