TCP IP networking

Pierre Colson

Thursday 06 January 2022

Contents

Markdown version on *github*
Compiled using *pandoc* and *`gpdf` script*
More fiches *here*

General

- **Application layer** helps people and machines communicates
- **Transport layer** helps Application layer
  - Provides programming interface to application layer
    - **UDP**
    - **TCP**
  - **Port numbers** allow to differentiate source/destination processes on one machine
    - Source and destination port number are carried in UPD/TCP header
- **Network layer** provides full connectivity
  - **IPv4** (32 bits)
  - **IPv6** (128 bits)
- **IPv6**
  - The 3 first blocks or more are the routing prefix
  - The 4th block or fewer is the subnet id
  - The 4 first blocks are the network prefix
- Data is broken into chunks called **IP packets** of size ≤ 1500 bytes
- **Names** are human readable synonyms for IPv4 and IPv6 address
  - Mapped to address by **DNS** servers
- **Link layer** = MAC layer
  - Interconnects a small number of devices without any configuration
  - **MAC address** are hardware address (48 bits, set by manufacturer)
- **Local Area network** : A set of devices that are connected at the MAC layer
- LANs can be interconnected by **routers** : devices that forward packets based on IP address
- **Bridges** or **Switch** : A system that forwards packets based on MAC addresses
- Every machine must know the IP address of the next router (**default gateway**)
- The IP address of all machines in one subnetword must have same **subnet prefix**
- The size of IP subnet prefix is often specified using a **network mask**
- **MAC frame**
  - MAC header (detination MAC address + other things)
  - MAC payload
- **IP packet** is included in MAC payload
  - IP header (IP destination address + other things)
  - IP payload
- **TCP segment** is included in IP paylaod
  - TCP header (source and dest port nb + other things)
  - TCP payload
- TCP payload can include encryption header + encrypted bytes of an HTML file
- The **bit rate** of a channel is the numbber of bits per seconds.
- The **bandwidth** is the width of the frequency range that can be used for transmission over the channel
- In computer science, many people use *bandwidth* instead of *bit rate*
- **Throughput** is the number of useful data bits / time unit
- **Stop and Go Protocol**
  - Wait acknoledgement before sending a new packet

The network layer IPv4 and Ipv6

- Every system (host = end-system, router = intermediate system) has a forwarding table (= routing table) and performs **Longest prefix match** on destination address
- Between subnets (= LANs) use routers, inside subnet don't
- Host in same subnet must have same *subnet router* and same *subnet prefix*
- **NAT** (Network Address Translation) box
  - Allow $n > 1$ devices to use one single IP address
  - Translates *internal IP address* and *internal port* number into *NAT IP address* and *NAT port*

number
  - Forwarding is based on exact matching in NAT table
  - Internal addresses are typically private addresses
  - A server port can be accessed only if explicitely configured
  - With IPv6, home routers often do not use NAT because provider typically allocates a block if IPv6 addresses, not just one as with IPv4
- With IPv6 the home router provides protection by acting as a **filtering router** : allows communication from outside only if initiated from inside or manually configured
- **Link local address** : can be used only between systems on same LAN
- Some system still use NAT with IPv6 : Use NAT with link local IPv6 address on internal network
- **NDP** : Mac address resolution with IPv6
  - Done via **NS** (Neighbour Solicitation) packet
- **ARP** : Mac address resolution with IPv4
  - ARP request is broadcasted to all nodes in LAN
- Host configuration
  - IP address of this interface
  - Mask of this interface
  - IP address of default router
  - IP address of DNS server
  - Can be done manually or automatically with
    * IPv4 : DHCP (Dynamic Hsot Configuration Protocol)
    * IPv6 : DHCP stateful, SLAAC (stateless), DHCP stateless
- **DHCP** : Configuration info is kept in central DHCP server, contacted by host when it needs an IP address
  - Router implements a **DHCP Relay** because host cannot contact DHCP servers since they do not have an IP address yet

The MAC layer

- **MAC** Medium Access Control
- **Aloha** is the basis of all non deterministic acces methods
  - It assumes one shared channel from remote to center, and a separate (non interfering) channel from center to remotes
  - Collisions occur when two packets transmissions overlap, and if a packet is lost, then source has to retransmit
  - There is no feedback to the source in case of collision
- **CSMA** (carrier sense multiple access) improves on Aloha by requiring that stations listen before transmitting
  - Many collisions can be avoided, but not all. This is due to propagation delays
  - Call T the maximum propagation time from station A to any other stations. If no collision occurs during a time interbal of duration T after A started transmitting then A has seized the channel (no other can send)
  - CSMA works well only if the transmission time is much larger than propagation namely bandwidth-delay product « frame time
  - In order to avoid repeated collisions, it is required to wait for a random dalay before re-transmitting
- **CSMA/CD** (collision detection) improves on CSMA by requiring that stations detect collisions and stop transmitting
  - It has better perfromance than Aloha or CSMA
  - After a collision is detected, stations will re-attempt to transmit after a random time
  - Acknowledgments are not necessary because absence of collision means that the frame could be transmitted
  - A minimum frame size is necessary to guarantee Collision Detection
  - This is (old style) Ethernet
- Wifi : **CSMA/CA** (Collision Avoidance) : Variant of CSMA/CD where collision detection is replaced

by **RTS/CTS** (Request to Send / Clear to send)

- Ethernet LANs use **switches (=bridges)**
  - A bridge is an intermediate system of the MAC layer, it forwards packet based on MAC addresses
  - A bridge is a queuing system
  - The design of bridges requires them to be *transparent*
  - Bridges use a forwarding table with **exact match**
    - Forwarding tables contains list of destination MAC addresses and interfaces (also called *ports*)
    - If destination address is not in the table, then the message is broadcasted to all interfaces
    - Table is built by learning from source address field in MAC frame
    - Learnt addresses time out if not re-learnt
- Bridges run a protocol between neighboring bridges, called **spanning tree protocol** to force the active topology to be loop-free
  - De-activate some ports such that the remaining topology is a tree that spans all bridges (adapt to failures and additions)
  1. Bridges elect on **root** bridge (bridge with smaller cofigurable bridge label)
  2. Only links that go to root bridge along the shortest path are active (Spanning tree = set of **shortest paths** to root)
  3. All bridges monitor that the root is reacheable and if not, trigger re-computation of a new spanning tree
  - Each LAN between bridges has a configurable **cost**, by default, decreasing function of bit rate
- **Ethernet Frame** = Ethernet Protocol Data Unit (PDU)
  - An Ethernet frame typically transports an IP packet, sometimes also other
  - The *preamble* is used for receiver to synchronize
  - *SFD* (start frame delimiter) is used to validate the beginning of a frame
  - *Destination length* is used to indicate the total length before padding. Padding is require is the minimum frame size of 521 bits = 64 bytes is not reached
  - *FCS* (Frame check Sequence) is an error detection mechanism
    - This detects the frames that have an error
- Ethernet addresses are known as **MAC addresses**. Every Ethernet interface has its own MAC address, which is in fact the serial number of the adapter, put by the manufacturer
  - They are 48 bits long.
    - The first address bit is the individual/group bit, used to differentiate normal addresses from group addresses
    - The second bit indicates whether the address is globally administered or loacally administered
- `ff:ff:ff:ff:ff:ff` is the **broadcast address**
  - On shared medium LAN all machines receive packet and do not discard it
  - With bridges, broadcast frames are sent on all nodes and ports on the spanning tree
- MAC addresses with bit 8 equal to 1 are **multicast** addresses
  - Ethernet adapter discards such packets exept if host subscribes to address
  - Non smart bridges broadcast such frames
  - Smart bridges send only to nodes that subscribes to multicast address
- **Virtual LAN** are handled as if they were physically seperated : different forwarding tables, different spannign tree
  - Configure (by network managment) which switch port belongs to wich VLAN
  - Switch handles ports of different VLANs as seperated communicating VLAN
  - VLAN label in Ethernet header on links between switches allows switch to allocate packets to VLAN
- **Wifi access points** are usually a combination of
  - A Wifi base station
  - A bridges with an ethernet backend (called the *Distribution System*)
- Security aspects
  - MAC addresses are sent in the clear, attacks :
    - **Eavesdropping** : hearing someone else's data
    - **Free riding** : Inserting a device into a network without authorization

        * **Inpersonation** : send data with someone else's MAC address, replay attacks
    - Solutions :
        * **Access Control** require user to show credentials before allowing a given MAC address into the network
        * **Authentication** every MAC frame is signed using cryptography and numbered
        * **Encryption** MAC frame payload is encrypted
- The **MAC layer for wireless medium** (WiFi) takes care of sharing the radio waves in a local environment
- The **MAC layer for wired medium** (Ethernet) was originally designed for sharign a cable; in this form it uses a protocol called CSMA/CD very similar to WiFi
- The **MAC layer for wired meidum** (Ethernet) has now got rid of any protocol using the interconnected bridges. It thus forms an interconnection layer of local scope
- **Bridges** use routing tables that are nor structured. A bridges must search the entire tabel for every packet
    - They are independant of whether IPv4 or IPv6 is used. They are plug and play and are independant of IP numbering plans

The transport layer TCP and UDP

- **UDP** (User Datagram protocol)
    - Uses **ports number**
    - One message, up to 65 535 bytes
    - Destination address, destination port, source address, source port
    - Destination address can be unicast or multicast
    - Delivers exaclty the message or nothing
    - Consecutive messages may arrive in disorder
    - Message may be lost – applicaton must handle
    - Used via socket library
- **TCP** (Transmission Control Protocol) : error recovery + flow control
    - Guarantees that all data is delivered in sequence and without any loss, unless the connection is broken
    - Data in numbered (per byte sequence number)
    - A connection (synchronization of sequence numbers) is opened between sender and receiver
    - TCP waits for acknoledgements, if missing data is detected, TCP retransmits
- UPD and TCP are not affected by the choice of IPv4 and IPv6
    - However there are UPDv4 sockets and UDPv6 sockets
    - An IPv6 socket can be dual stack : the socket can be found ot both IPv6 and IPv4 addresses of the local host
        * IPv4 addresses of correspondants are mapped to IPv6 addresses using the **IPv4 mapped IPv6 address** format (i.e. an address in the block `::ffff:0:0/96`
- On the sending side : Operating System sends the UDP datagram as soon as possible
- On the receiving side : Operating System re-assembles IP fragments of UDP datagram (if required) and keeps it in buffer ready to be read. Packet is removed from the buffer when applications reads
- In addtion to the **ACK** field, most TCP implementation also use the **SACK** field (Selective acknowledgment)
- TCP receiver uses a **receive buffer** = re sequencing buffer to store incomming packets before delivering them to application
- TCP uses a **sliding windows**
    - Lower edge = smaller non acknowledged sequence number
    - Upper edge = lower edge + window size
    - Only sequence numbers that are in the window may be sent
    - TCP constantly adpats the size of the window be sending "window" advertisements back to the source
        * This is called **flow control** = adapt sending rate to source to speed of receiver $\neq$ congestion

control, which adapts rate of source to state of network

- TCP requires that a connection is opened before transmitting data
  - Used to agree on sequence numbers and make sure buffers and window are initially empty
- TCP is used by means of sockets like UPD. However, TCP sockets are more complicated because of the need to open/close a connection
  - `socket(AF_INET, ...)` creates an IPv4 socket
  - `socket(AF_INET6, ...)` creates an IPv6 socket
  - `bind(port)` associates the local port number with the socket. The server must bind, the client not bind, a temporary port number is allocated by the OS
  - `connect()` associates the remote IP address and port number with the socket and sends a SYN packet
  - `send()` sends a block of data to the remote destination
  - `listen()` declares the size of the buffer used for storing incoming SYN packets
  - `accept()` blocks until a SYN packet is received for this local port number
  - `recv()` blocks until one block of data is ready to be consumed on this port number. You must tell in argument how many bytes at most you want to read
- **Fast retransmit** when n duplicate ACKs are received, declare a loss
- TCP also uses a **retransmit timer** set for every packet transmission
- With SYN cookies, TPC server does not state information after receiving a SYN packet. State is encoded in the Seq Number field using a cryptographic funcion and returned to client
- **TLS** adds to TCP :
  - **Confidentiality** : data is encrypted with symmetric encryption, using secret key created on the fly for this session
  - **Authentication** : data is protected against forgery + identity of end system is authentified

IP multicast

- **multicast** : send to a group of destinations
  - **Any Source Multicast** (ASM) : the group is identified by the multicast address. Any source can send to this group
  - **Source Specific Multicast** (SSM) : the group is identified by (s, m) where m is a multicast address and s is a (unicast) source address. Only s can send to this group
- Source simply sends one single packet for n destination
- Destination subscribe via **IGMP** (Internet Group Managment Protocol, IPv4) or **MLD** (Multicast Listener Discovery, IPv6)
- Routers either build distribution tree via a **multicast routing protocol** (PIM-SM) or use tunnels (BIER)
- **Packet multiplication** is done by routers
- **Protocol Independend Multicast** (PIM) supports ASM and SSM and exists in two versions :
  - PIM-DM (Dense mode) : Makes eavy use of broadcast and can be used only in small tighly controlled network
  - PIM-SM (Sparse mode) is more reasonable and is used e.g. for TV distribution When used with SSM, PIM-SM uses **reverse path forwarding**
- Multicast routing protocol requires to keep per flow state. **BIER** (Bit index Explicit Replication) is an alternative that avoids this
  - A multicast packets receives a BIEA header that contains the list of destination BIER routers (Bit forwarding Egree Routers, BFERs)
    - The set of destination (BFERs) is encoded using a bitstring
  - BIER router use unicast routing in order to deliver packets to the set of BFERs indicated in the BIER header
  - A BIER router pre computes a **forwarding bit mask** that indicates, for every destination BFER, the set of destination BFERs than can be reached with a single packet
- There is ARP for multicast. IP multicast address is algorithmically mapped to a multicast MAC address
- **MAC multicast**

  - Some (non smart) switches simply treat multicast frames as broadcast
  - Some smarter switches simply listen to IGMP/MLD and overhear who listens, deliver only to intended recipients (IGMP or MLD snooping), but does not distinguish SSM from ASM
- **Security of IP multicast**
  - IP multicast with or without BIER makes life easier for attackers
  - SSM is safer as router and destination can reject unwanted sources
  - IGMP/MLD is not secured and has the same problems as ARP/NDP
  - Multicast capable networks must deploy exhaustive filtering and monitoring tolls to limit potential damage
- Multicast is good for **sources** : One packet sent for n destination
- Multicast works only with **UDP**
- Multicast routing requires per flow state and in non scalable. tneds to be replaced by BIER. BIER uses a different forwarding principle, based on bitstrings (which represents sets of destinations)

IP state routing

- **Routing** is often taken as synonym for "IP packet forwarding"
- IP packet forwarding uses a routing tables
- A **routing protocol** is a means to automatically compute the routing tabless in a number of routers
- **Link state** : all routers in one domain know a map of entire domain, obtained by gossiping with other routers
  - Every link on the map has a cost
  - Router compute next hop to destination by computing shortest paths based on their maps
- **Distance vector** : all router in one domain know only their neighbours + distance to all destination
  - Routers inform their neighbours of their own list of distances to all destination (the vector of all distances)
- **Path vector** : Every router knows only its neighbours + explicit paths to all destinations used with BGP
- **Source routing** : Path are computed by the source host and put into packets headers. With IPv6 routing header is an extension header (Contains intermediate hops and ultimate destination)
- **Loose source routing** : force some intermediate hops, which need to be on-link
- **Segment routing** : generalizes loose source routing by allowing the RH to contain indications for processing by intermediate hop
- **OSPF** (Open shortest Path First) is a link state routing protocol, In single area
  - Every router has an *interface database* (describing its physical connections, learnt by configuration), an *adjacency database* (describing neighbour state, learnt by the `hello` protocol) and a *link state database* (the network map, learnt by flooding)
  - `Hello` protocol is used to discover neighbouring routers and to detect failures
  - When two routers become new neighbour the first **synchronize** their link state databases. Typically one router is new and copies what the other already know
  - **Link state database** and **LSAs**
    * One synchronized, a router sends and accecpts **link state advertisements** (LSAs)
    * Every router sends one LSA describing its attached networks and neighbouring routers
    * LSAs are flooded to the entire area and stored by all routers in their link state
    * LSAs cntains a sequence number and age; only messages with new sequence number are accpeted and re-flooded to all neighbours. Sequence number prevents loops. Age field is used to periodically resend LSA (eg every 30 mn) and to flush invalid LSAs
- Ethernet LANs are treated in a special way, the routers elect one designated router per LAN (and one backup designated router)
  - Every router that is connected to an Ethenet LAN floods a "router LSA" idicating its connection to this LAN. The designated router "speaks for the switch" and sends a "network LSA" which gives the list of all routers connected to the LAN
- OSPF packet are sent directly over IP
  - Reliable transmissions is managed by OSPF with OSPF acks and timers

  - Routers are identified by a 32 bit number
- Path conputation uses **Dijikstra's Algorithm**
  - Performed at every router, based on link state database
  - Router computs one or several shortest paths to every desination from self
  - Worst case complexity of this $\mathcal{O}^2$
- OSPF uses Dijikstra's shortest path : the best known algorithm for centralized operation
- Path are computed independently at every node
  - Link state database is same at all routers, but every router performs a different computation as it computes path from self
  - Synchronisation of database guarantees absence of persistent loops
- OSPF supports **multiple shortest paths**, IP allows to ahev multiple next hop to the same destination in the routing table
- **per-flow load balancing** requires that packets of the same *flow* are setnt to the same next hop. The definition of the flow depends on the system
  - A flow is identified by the source and destination addresses
  - A flown can also by identified by the next hop
- Link or routers **failure** are detected by OSPF's hello protocol or by the BFD protocol at a lower layer
  - When a router sees a change in the state of a link or a neighbouring router, its send a new LSA to all its neighbour
  - Changes to link state database trigger re computation of shortest paths and routing tables
- **Security of OSPF**
  - Attacks against routing protocols
    1) Send invalid routing information $\Rightarrow$ disrupt network operations
    2) Send forged routing information $\Rightarrow$ change network paths
    3) Denial of service attacks
  - **OSPF** security protects against 1. and 2. using **authentication**
- **OSPF Type 3** Authentication uses secret, shared key
  - Digest and Crypto Sequence number are appended after OSPF message and sent in IP packet in cleartext
- OSPF with **multiple areas**
  - Link state floods all information to all routers, therefore does not scale to very large network
  - Inside one area, link state is used. One link state database per area
  - Area **broder routers** belong to both areas
  - An area border router injects **aggregated distance information** learnt from one area into another area
- Link state routing provides a complete view of area to every node. This can be used to provide advanced functions
  - **Multi-class routing** : compute different routes for different types of services
  - **Explicit routes** : an edge router computes the entire path, not just the next hop, and writes it in the packet header. Avoids transients loops / supports fast re route after failure. Used in deterministic networks
- **OSPF** (and routing protocol) automatically build connectivity and repair failures. With routing :
  - All routers compute their own link state database, replicated in all routers
  - All routers computer their own routing tables using Dijikstra and the link state database
  - Converging after failure is fast (if detection is fast)
  - Non standard cost definitions are possible, can be used for routing specific flows in different direction
  - Large domains must be split into areas
- More control can be obtained by an outside application (SDN). SDN is used today primarly with switches, but also starts to be used with routers

Congestion control

- A network should be organized so as to avoid inefficiency
- **Pareto efficiency** : A feasible allocation of rates x is called **Pareto efficient** iff increasing one source

must be at the expense of decreasing some other source
  - Pareto efficient iff every source has a bottleneck link and cannot be increased unilaterally
- We say that a feasible allocation x is **max-min fair** iff for every source i, increasing its rate must force the rate of some other (not richer) source j to decrease
  - If it exists, the min-max fait allocation is unique
  - For the set of feasible rates, the unique max min fair allocation is obtained by **water-filling**
- We say that a feasible allocation x is **proportionally fair** if all allocation is proportionally fair if for any other allocation, the total *rate of change* is ≤ 0
- **Utility fairness** : One can interpret proportional fairness as the allocation that maximizes a global utility
- **Congestion control implementation**
  - **Explicit** (rate based) : tell every host how fast it can send MPLS networks (smart grid)
  - **Hop by hop** (backpressure) : STOP/GO signals sent upstream
  - **Fair queuing per flow** : One queue per flow/per user, served round robin
  - **End to end** : host taste the water and increase or decrease their sending rate using a host congestion control algorithm (the solution in the internet)
- Among the linear controls, only the **additive increase - multiplicative decrease (AIMD)** tends to bring the allocation towards fairness and efficiency
- AIMD convergence can be accelerated when initial conditions are very different. **Slow start** is an additional method added to AIMD, used at the beginning of the connection and at losses detected by timeout
  - Increase the rate multiplicatively
- **TCP** is used to avoid congestion control in the internet
  - A TCP source adjusts its window to the congestion status of the internet
  - This avoids congestion collapse and ensures some fairness
  - TCP sources interpret losses as a **negative feedback**
- **TCP Reno** congestion control, uses AIMD and slow start
  - TCP adjusts the windows size based on the approximation rate $\sim \frac{W}{RTT}$
    - W = min(cwnd, offeredWindow)
    - OfferedWindow = window obtained by TCP's window field
    - cwnd = controlled by TCP congestion control
  - Negative feedback = loss
    - Multiplicative decrease ($u_1 = 0.5$)
  - Positive feedback = ACK received
    - Increase in additive ($\sim +1$ MSS per RTT)
  - Slow start with increase factor ($w_0 = 2$) per round trip time
  - Loss detected by timeout $\rightarrow$ slow start
  - Loss detected by fast retransmit $\rightarrow$ **fast recovery**
  - **Multiplicative decrease** : $ssthresh = 0.5 \times cwnd$
  - **Additive increase** : for every ack received $cwnd = cwnd + MSS \times MSS/cmd$ (in we count in packet this would be : $cwnd+ = 1/cwnd$
  - **Multiplicative increase** (slow start) : no duplicate ack received during slow start
    - $cwnd+ = MSS$ (in bytes) (in packets : $cwnd+ = 1$)
    - If $cwnd = ssthresh$ the go to the congestion avoidance
  - Target window of slow start is called ssthresh (slow start threshold)
  - There is a slow start phase initially and after every packet loss detected by timeout
  - In all aother packet loss detection events, slow start in not used, but **fast recovery** is used instead
    - Target window is halved
    - But congestion window is allowed to increase beyong the target window until the loss is repaired
  - Two competing sources with different RTTs are not treated equally, source with higher RTT obtain less (**RTT bias**)
    - Cause : additive increase is one packet per RTT (instead of one packet per constant time interval)
  - TCP Reno shrortcomings

        * RTT bias
        * Periodic losses must occur, not nice for application
        * TCP controls the window, not the rate. Large *bursts* typically occur when packets are released by host following, not nice for queues in the internet, makes non smooth behaviour
        * **Self inflicted delay** : if network buffers are large, TCP first fills buffer before adapting the rate. The RTT is incresed unnecessarily. Buffer are constantly full, which reduces their usefulness (bufferbload) and increases delay for all users. Interactive, short flows see large latency when buffers are large and full
- **TCP cubic** modifies congestion control. TCP cubic keeps the same slow start, congestion avoidance, fast recovery phases as TCP reno but :
  - Multiplicative decreaste is $\times 0.7$ (instead of $\times 0.5$)
  - During congestion avoidance, the increase is not additive but **cubic**
    * Cubic increases window in concave way until reaches W_{max} then increases in a convex way
    * Cubics's window function is independant of RTT
    * Is smaller than Reno when RTT is small, and larger when RTT is large
  - Cubic's throughput $\geq$ Reno's throughput with equality when RTT or bandwidth delay product is small
- **ECN (Explicit Congestion Notification)** aims to avoid bufferbloat
  - Router marks packet instead of dropping
  - TCP destinations echoes the mark back to source
  - At the source, TCP interprets a marked packet as if there would be a loss detected by fast retransmit
  - ECN flags in IP and TCP header
- **RED (Random Early Detection)**
  - Queue estimates its average queue length
  - Incomming packet is marked with probability given by RED curve
  - RED is an example of Active Queue Managment (AQM)
- **Active Queue Managment** (AQM) can also be applied even if ECN is not supported, In such case, e.g., with RED, a packet is dropped with proba q computed by the RED curve
  - Packet may be discarded even if there is some space available
  - Benefits :
    * Avoid bufferbloat - reduce latency
    * Avoid irregular drop patterns
- **Data Center TCP**
  - Improve performance for jumbo flow when ECN is used. Same a Reno except the brutal multiplicative decrease by 50%
  - TCP source estimates proba of congestion p
  - ECN echo is modified so that the propagation of CE marked ACks $\sim$ the proprobability of congestion
- Buffer **Drain time** = buffer capacity / link rate
  - To keep buffer drain time constant, the product (memory speed $\times$ memory size) should scale faster than link rate, which is technologically not feasible
- In general, all flows compete in the internet using the congestion control method of TCP. It is possible to modify the competition and separate flows using **per-class flow**
  - Each routers classify packets
    * Each class is guaranteed a rate
    * Classes may exceed the guaranteed rate by borrowing form other classes if ther is spare capacity

External routing BGP

- **Routing protocol** are said
  - *Interior* : Inside ASs : RIP, OSPF (standard), IGRP (Cisco)
  - *Exterior* : Between ASs : BGP
- **ARD** : Autonomous Routing Domain : Routing domain under one single administration
- **AS** : Autonomous System = ARD with a number ("AS number")

- **IGP** is used inside a domain, **BGP** is used between domains
- **BGP** is a routing protocol between ARDs. It is used to compute paths from one router in one ARD to any network prefix anywhere in the world
  - It can handle both IPv4 and IPv6 addresses in a single process
- **Path vector routing** : Find best routes in a sense that can be decided by every ARD using their own criteria
  - A route between neighbours is where path is a sequence of AS numbers and dest is an IP prefix
  - Every AS appends its number to the path it exports
  - Every AS uses its own rules for deciding which path is better
- A router that runs BGP is called a **BGP speaker**
- At the **boundary** between 2 ARDs there are 2 BGP speakers, one in each domain
- Inside one ARD there are usually several BGP speakers
  - They all talk to each other, in order to exchange what they have learnt, Using **Internal BGP**
  - **I-BGP** is the same as **E-BGP** except
    - Routers learned form I-BGP are not repeated to I-BGP
    - Router does not prepend own AS number over I-BGP
    - NEXT-HOP is not modified
- **Policy routing** Interconnection of ASs is self organized
  - Point to point links between netwoks
  - Interconnection points : All participants run a BGP router in the same LAN
  - Mainly two types of relations
    - Customer-provider : EPFL is customer of Switch. EPFL pays Switch
    - Shared Cost peer : Swisscom and Switch are peers. They collaborate to serve their customers
- **BGP** How it works :
  - BGP routers talk to each other over TCP connections
  - BGP messages : OPEN, NOTIFICATION, (=RESET), KEEPALIVE, UPDATE
    - UPDATE messages contains modifications
  - A BGP router transmits only modifications
- A **BGP router**
  - Receives and stores candidates routers from its BGP peers and from itself
  - Applies the decision process to select at most one route per destination prefix
  - Exports the selected routes to BGP neighbours, after applying export policy rules and possibly aggregation
  - Stores result in Adj-RIB-out (one per BGP peer) and sends updates when Adj-RIB-out changes (additions or deletion)
  - Only routes learnt from E-BGP are sent to an I-BGP neighbor
- The records sent in BGP messages are called *Routes*. Routes + their attribute are stored in the Adj-RIB-in, Loc-RIB, Adj-RIB-out
- A **route** is made of :
  - Destination (subnetwork prefix)
  - Path of destination
  - NEXT HOP (set by E-BGP, left by I-BGP)
  - Origin : route learnt from IGP, BGP, static
  - Other attributes
- In addition, like any IP host or router, a BGP router also has a **Routing Table** = IP forwarding table
- **The decision process** decides which route is selected
  - At most one best route to exaclty the same prefix is chosen
  - A route can be selected only if its next hop is reachable
  - Route are compared against each other using a sequence of criteria, until only one route remains. A common sequence is :
    1) Highest weight (Cisco proprietary)
    2) Highest LOCAL-PREF
    3) Shortest AS-PATH
    4) Lowest MED, if taken seriously by this network

5) E-BGP > I-BGP
6) Shortest path to NEXT HOP, according to IGP
7) Lowest BGP identifier

- Some BGP routers must **originate the routes** that are in their own domains. Several methods for the source of a route
  - Static configuration : tell this BGP router which are the prefix to originated
  - **Redistribute connected** : tell this BGP router to originata all prefixes that are no-link with this router
  - **Redistribute from IGP** : tell this router to originate all prefixes that IGP has learnt
- Domain that do not have a default route must know all routes in the world
  - In IP routing tables unless default routes are used
  - In BGP announcement
  - **Aggregation** is a way to reduce the number of route
  - Aggregation is expected to be very frequent with IPv6, less with IPv4
- Forwarding Entries learnt by GB¨are written into **routing tables**
  - **Redistribution** of BGP into IGP : routes learnt by BGP are passed to IGP
    - Only routes learnt by E-BGP are redistributed into IGP
    - IGP propagates the toues to all routers in domain
    - Works with OSPF
  - **Injection** of BGP routes into forwarding tables of this router
    - Route do not propagate, this helps only this router
- Avoiding RE distribution of BGP into IGP : Many operators avoid re distribution of BGP into IGP
  - Large number of routing entries in IGP
  - Convergence time after failures is large if IGP has many routing table entries
  - OSPF is able to handle large routing tables
  - If redistribution is avoided, only injection is used
- **Injection Conflicts** : a route may be injected into the forwarding table by both OSPF and BGP
  - To solve this conflict, every route in the forwarding table receives an attribute called the administrative distance which depends on which process wrote the route
  - Only the route with the smallest administrative distance is selected to forward IP packet
  - The decision process selects a BGP route only if there is no route to same destination prefix with smaller administrative distance in the forwarding table
- **Weight** is a route attribute given by Cisco or similar router
  - It remains local to this router
  - Never propagated to other routers, even in the same AS
  - Therefore there is no weight attribute in router announcments
- **MED** is used between ASs
  - Is used to tell one provider AS which which entry link to prefer
- **LOCAL-PREF** is used inside one AS (over I-BGP)
  - Is used inside one AS to propagate AS path preferences, this allow this AS to tell the rest of the world which AS path it wants to use, by not announcing the other paths
- With BGP routes are explicitly xithdrawn (and updated)
- **Route flap** : a route is sucessively withdrawn, updated. The flap propagates to th AS and to others ASs. Causes CPU congestion on BGP routers
- **Route flap damping** (also called dampening) mitigates route flap
- Malicious or simply buggy BGP updates may cause damage to global internet
- **Routing registry** manages address allocations / delegated to 5 Regional Internet Registries, IRRs
  - RIPE maintains a public Routing Registry, database of address blocks + some policy information
- Owner of an address block creates a (cryptographically signed) **Route Origin Authorization** (ROA) that contains AS number and IP address block, this validates origination, prevents bogus origination. More secure than IRR
  - Uses RPKI (resource public key infrastructure) rooted at IANA/ICANN and deployed in RIRs

Application layer

- The application layer of TCP/IP consists of
  - The distributed applications themselves, it is the topic of courses such as information
  - Systems, distributed system, industrial informatics
  - A number a generic intermediate "layers"
- **QUIC**
  - Avoid latency and head-of-the line blocking TLS avec TCP
  - Modifications to TCP are often impractical because of filtering router that discard TCP options
  - QUIC opens one TLS 1.3 session first
  - QUIC runs over UDP
  - Data is always secured
- One QUIC session (also called "connection") has multiple **streams**
  - Every stream is reliable, like TCP, and data of this stream is delivered in sequence. But no sequence between different streams
  - Streams can be created and deleted on the fly
  - Stream 0 is for TLS1.3 and connection management
- One QUIC packet has unique (increasing) packet number (64 bits), never wraps. A retransmitted packet has a different number. A packet contains frames for streams data and other data
- QUIC obtains **reliability similar to TCP** : missing packets are detected and retransmitted
- **Flow control** is both per stream and per connection, using an explicit offset (instead of window)
- **Congestion control** is similar to TCP Reno or Cubic, per connection (not per stream)
- **Unreliable streams** are also foreseen, they do not have packet retransmission (in principle) but are subject to flow control and to congestion control
- The **Domain Name System (DNS)**
  - Support user friendly naming of resources
  - Hide IP address changes on servers
  - Map DNS names to IP addresses
- **Reverse DSN lookup** : Reverse Query : find name given some IP address
- DNS **authority** is hierarchical
  - Top level authority allocates top level domains
  - The top level is currently run by PTI
- **National registry** allocates `.ch` domain names
- **Registar** is a commercial organization accredited by registry to sell names
- DNS server are run by ISPs or other organization
- **namecoin** is an attempt to provide perr to peer DNS names for the domain `.bit` It uses the blockchain technology
- DNS alone in unsecure even if server are trusted, anyone can send (incorrect) answers to DNS queries, **DNSSEC** solves this problem :
  - Records are signed, using public key cryptography
  - Chain of trust is initialized with root servers
- Top level doain `.local` is reserved for names visible in subnet
- An **Application layer gateway** is an application layer intermediate system. It terminates the TCP connections and does *store and forward* for the application layer data, it is another example of *middle box*
- **Web proxy** can be deployed to reduce traffic
  - They can keep frequently asked documents close to user
  - Requested file are kept in the cache
  - Opens a TCP connection to remote server on behalf of the client, this is called an **HTTP Tunnel**
    - There are two TCP connections, one client-proxy and one proxy-remote server
- The **End-to-end** principle of the internet says that any layer above the nerwork layer should avoid intermediate systems
  - Thus : Application Layer Gateways should be avoided
  - Simplify the nerwork

  - The network is independant of applications and can be run more safely
  - Performance is better, no store and forward
  - Application layer are still desirable in some cases
    * Mobility
    * Security / Access control
    * Interworking
    * Performance
- **Server side load balancers** are application gateways : they terminate TCP or QUIC connecitons and re-direct user request to on of the applications servers. Also called **reverse proxy**
- **Firewall** is a systm that separates Internet from intranet : all traffic must go through firewall, only authorized traffic may go through firewall itself cannot be penetrated
- **QUIC** can be used as a replacement for TCP. Main difference are : TLS is integrated + multiple streams
- DNS is used by apps on dual stack machines to know whether to use IPv4 or IPv6
- ALG46s (Dual stack Application layer gateways) (web prowies and HTTP tunnels) can be used to solve the h4 to h6 interworking problem

The network layer (again)

- Link layer network have different maximum frame lengths
- **MTU** (maximum transmission unit) = maximum frame usable for an IP packet
  - MAC layer options and tunnels reduce MTU
- **Fragmentation** : Hots or router may have IP datagrams larger the MTU
  - Fragmentation is performed at source or at routers when IP datagram is too large
  - New application should *avoid* fragmentation
  - IPv6 routers never fragment (drop packet if too large), but IPv4 router may
  - All fragments are self contained IP packet
  - IP *datagram* = orginal, IP *packet* = fragment or complete datagram
- If fragmentation occurs, only the first fragment contains the port number
  - This is a problem for NATs. Typically sees a firt fragment, it caches (IP address, identification, ports numbers). When a subsequent fragment arrives, the port number can be recovered from the cache. If the first received fragment is not the first of the original datagram, this does not work and the fragment is discarded
  - This is also a problem for firewalls / instusion detection systems : Hard to do deep packet inspection
- Applicatons (UDP or TCP) should avoid fragmentation by *estimating the path MTU*, i.e. the mac MTU of all links betwenn source and destination
  - One (suboptimal) method is to use the default MTU
    * All IPv6 implementations must have $MTU \geq 1280$ Bytes
    * All IPv4 implementations must gave $MTU \geq 68$ Bytes but most should have $MTU \geq 576$ Bytes
  - A better method : **Packetization Layer Path MTU Discovery** (PLPMTUD)
    * Observer the largest possible PATH-MTU that works by observing packets whose reception was confirmed. Kepp this information in the IP layer (correspondent table)
    * Start from default MTU, from time to time, try a larger MTU
    * If destination in onlink, path MTU should be equal to the interface MTU
    * TCP conenctions should negociate an MSS (Maximum segment size) such that

$$MSS \leq PathMTU - IPheadersize - TCPheadersize$$

- A **Tunnel**, also called **encapsulation** occurs whenever a communication layer carries packets of a layer that is not the one above (e.g. Ip packet in UDP etc)
- **6 to 6 over 4** : **Tunnel Broker** uses IPv6 in IPv4 encapsulation
- **4 to 4 over 6** : **464XLAT** uses NATs (XLAT = translation, not tunnels)
- **MPLS** Multi protocol Label Switching complements IP, it uses the priciple of *connection oriented* network layer as opposed to connecitonless as IP does, it can be used as replacment to tunnels

  - Before transmitting data betwenn two end points one must establish a connection using a *signaling protocol*
  - Every packet header contains a label which identifies the connection on this link
  - The label is local to the link or to the node and can be modified by intermediate system
  - Intermediate system called switches or routers performs *exact lookup* in the label switching table
  - The value in the lable switching table are managed by the control plane or by a centralized SDN application
  - The conenction is called Label Switched Path (LSP) with MPLS
- **MPLS** header comes after the MAC mayer header
  - The MPLS header contains mainly the label, plus 3 bits for quality of service and a TTL field
  - There can be several MPLS headers on top of one anotheer. The innermost is recognized by the flag *Bottom of Stack* (bos)
- **Proxy ARP / ND Proxy** is a trick used to solve the problem caused buy a subnet present a different locations
- **Fragmentation** is due to different MAC layers having different packet sizes.
  - Fragmentation occurs only at IPv6 hosts, IPv4 hosts or IPv4 routers. Re-assembly is never done by routers
  - Fragmentation may cause problem and should be avoided is possible
- **Tunnels** are used e.g. to create virtual private networks
  - Transition to IPv6 creates many problems that can be solved with various methods involving automatic tunnels, header translation, and DNS manipulation
- In operator networks, MPLS is an alternative to tunnels, it builds path betwenn adge routers that by-pass the usual IP forwarding

Conclusion slides

The network layer IPv4 and IPv6

- IP is built on two principles:
  - One IP address per interface and longest prefix match, thsi allows to compress routing tables by aggregation
  - Inside subnet, don't use routers
- IPv4 and IPv6 are not compatible; interworking requires tricks
- NATs came as an after-thought and are widely deployed, primarly with IPv4, but sometimes also with IPv6
- ARP/NDP finds the MAC address corresponding to an IP address
- DHCP is used to allocates IP address, network mask and DNS server's IP address to a host; SLAAC automatically allocates IPv6 addresses withour DHCP
- TTL/HL limits the number of hops of an IP packet

The MAC layer

- The MAC layer for wireless medium (WiFi) takes care of sharing the radio waves in a local environment
- The MAC layer for wired medium (Ethernet) was originally designed for sharing a cable; in this form it uses a protocol, called CSMA/CD very similar to WiFi
- The MAC layer for wired medium (Ethernet) was now got rid of any protocol ("full duplex Ethernet"), using interconnected bridges. It this forms an interninterconnection layer of local scope
- We have seen two mechanisms to interconect devices : the network layer (with IP routers) and the MAC layer (with bridges)
- Bridges use routing tables that are not structured. A bridge must search the entore table for every packet. The table size and lookup time are prohibitive if the network becomes very large. We prefer routers for a large networks
- Bridges are independant of whether IPv4 or IPv6 is used. They are plug-and-play and are independant of IP numbering plans

- Historically, routers were in software and bridges in hardware, Nowadays every combination exists

The transport layer TCP and UDP

- The tranport layer in TCP/IP exists in two flavours
  - Reliable and stream oriented : TCP
  - Reliable and message based : UDP
- TCP uses : slidig window and selective repeat; window flow control; congestion control
- TCP offers a strict streaming service and requires 3 ways handshake
- Other transport layer protocol exist bur their use is marginal : e.g. SCTP (reliable + message based)
- Some application layer frameworks are subsititute to TCP for some application : QUIC (reliable and message based), websockets

IP multicast

- IP multicast came as an after-thought and uses a different principle than IP unicast (exact match versus longest prefic match); deployed only in specific network
- IP multicast addresse cannor be aggregated
- Multicast routing requires per-flow state and is non scalable. Tends to be replaced by BIER
- BIER uses a different forwarding principle, based on bitstring (which represent set of destinations)

IP state routing

- OSPF (and routing protocols) automatically build connectivity and repairs failures. With link state routing :
  - All routers compute their own link state database, replicated in all routers
  - All routers compute their own routing tables using Dijikstra and link state database
  - Convergence after failure is fast (if detection is fast)
  - Non standard cost definitions are possible; can be used for routing specific flows in different ways
  - Large domains must be split inta areas
- More control can be obtained by an outside application (SDN). SDN is used today primarly with switchesn but also starts to be used with routers

Congestion control

- Congestion control is necessary to avoid inefficiencies and callapses
- A congestion control scheme aims at allocating rates according to some form of fairness
- In the internet, we use end-to-end congestion control with :
  - AIMD
  - Slow start
  - And other refinements
- Congestion control is in TCP or in a TCP-friendly UDP application
- Traditional TCP :
  - Uses the window to control the amount of traffic : additive increase or cubic (as long as no loss); multiplicative decrease (following a loss)
  - Uses a loss congestion signal
- Too much buffer is as bad as too little buffer; bufferbloat provokes large latency for interactive flows
  - ECN can be used to avoid bufferbloat; it replaces loss by an explicit congestion signal; partly deployed in the internet; part of Data Centet TCP
  - TCP-BBR aims at avoiding the problem by probing available bandwidth
- Class based queuing is used to separate flows in enterprise networks or classes of flows in provider network

External routing BGP

- BGP integrates different ASs
- Interface BGP-IGP is complex and has many subtleties
- Security ofBGP is an active area of research and development
- BGP - MPLS - VPN
- Beyond BGP : SCION is an alternative to BGP (and to IP) that uses source routing and systematic encryption. Aims to provide more security

  not very useful this conclusion ...

Application layer

- QUIC can be used as a replacment for TCP. Main differences are :
  - TCL is integrated
  - Multiple streams
- The end-to-end principle states that Applications layer gateways (ALGs) should be avoided whenever possible. AGLs are still deployed for security / performance
- DNS is used by apps on dual stack machines to know whether to use IPv4 or IPv6
- ALG46s (web proxies, HTTP tunnels) can be used to solve the h4 to h6 interworking problem

The network layer (again)

- Proxy ARP / ND Proxy is a trick used to solve the problems caused buy a subnet present at different locations
- **Fragmentation** is due to different MAC mayers having different packet sizes
  - Fragmentation occurs only at IPv6 hosts, IPv4 hosts and IPv4 routers
  - Re-assembly is never done by routers
  - Fragmentation may cause problems and should be avoided if possible
- Tunnels are used e.g. to create virtual private networks
  - Transition to IPv6 creates many problems that can be solved with various methods involvong automatic tunnels, hedaer translation (CLAT, NAT64 = PLAT) and DNS manipulation (DNS64)
- In operator networks MPLS is an alternative to tunnels; it builds paths between edge routers that by-pass the usual IP forwarding