Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

Internet is a network of **connected computing devices**, infrastructure that connects & allows hosts/end-systems to communicate with each other. Networks organised into Autonomous Systems, each owned by an organisation
- connecting each access ISP to each other directly doesn't scale: O(N^2) max connections.
- IP addr, Internet Naming under Network Info Centre (NIC)
- Internet Society (ISOC) provides leadership in Internet related standards, education, and policy
- Internet Architecture Board (IAB) - authority issue, update technical standards regarding Internet protocols
- Internet Engineering Task Force (IETF) is protocol engineering, dev and standardization arm of the IAB. Internet standards published as RFCs (Request For Comments)

Network edge - End hosts, servers, access networks, links, etc. Hosts access the internet through access networks.
- wireless: connect hosts to router via base station
- physical media (cable)

Guided: solid media, e.g. fiber. Unguided media: signals propagate freely, e.g., Wi-Fi, cellular

Network core - ISPs, Routers, etc. A mesh of interconnected routers. Data transmitted through CS, PS

Circuit switching - Setup/teardown required
- End-to-end resources allocated to and reserved for "call"

1. call setup required
2. circuit-like (guaranteed) service, performance, throughput
3. circuit segment idle if not used by call (no sharing)
4. divide link bandwidth in "pieces" - freq or time division

Packet switching (e.g. Internet) - No setup/teardown required, Resources shared on demand, Best effort service
- Host sending function:
  - breaks application message into smaller chunks, known as packets, of length L bits
  - transmits packets onto the link at transmission rate R
- Packets are passed from one router to the next, across links on path from source to destination.
- Store and forward: entire packet must arrive at a router before it transmits on the next link.
- Sending a packet

1. Sender transmit a packet onto link as a sequence of bits
2. Bits propagated to next node (e.g. a router) on the link
3. Router stores, processes, forwards the packet to next link
4. Steps 2 & 3 repeat till the packet arrives at the receiver.

Packet delay (4 sources together make up end-to-end. Traceroute program displays path from src to dest, measures delay from src to each router)

d_proc: **processing** delay
- check bit errors, determine output link, usually < msec

d_queue: **queueing** delay
- depends on router congestion level
- Generally, if all packets have length L (bits), transmission rate is R, x bits of the currently-being-transmitted packet have been transmitted, and n packets already in the queue,

d_queue = $\frac{nL+(L-x)}{R}$

d_trans: **transmission** delay = $\frac{L}{R} = \frac{pkt\ length\ (bits)}{link\ bandwidth\ (bits/sec)}$

d_prop: **propagation** delay = $\frac{d}{s} = \frac{length\ of\ physical\ link}{propagation\ speed\ in\ medium}$

s ~2x10^8 m/sec, if not given

Throughput – measure for **end-end**, of how many bits can be transmitted per unit time

Bandwidth (link capacity) – trans rate for a **specific link**.

Routing - Routers determine source-destination route taken by packets using routing algorithms.

Addressing - each packet needs to carry src and dest info

Modularisation - eases maintenance & updating of system e.g. chg one layer's service transparent to rest of system

Lect 2 Application Layer - used by every internet app.
- network apps run on hosts, contain communicating processes. Server process waits to be contacted, and client process initiates the connection.

network apps may be:

1. client-server architecture - server waits for incoming reqs, provides service to client - client initiate contact w server. web client usually in browser

2. peer-to-peer (p2p) architecture - no always-on server, arbitrary end systems directly communicate. more scalable but hard to manage.

3. hybrid e.g. instant messaging
- chatting p2p but presence detection centralised (user register IP address with central server when come online, contact central server to find IP addresses of buddies)

App transport service requires Data integrity (reliability), Timing (delay), Throughput, Security

App layer protocols define
- format, order, type of msg (e.g. request, response)
- actions taken, rules (when & how apps send & respond to msgs)
- msg syntax (what fields, how fields are delineated) and
- semantics (info in fields meaning)

Open protocols defined in RFCs. They allow for interoperability. e.g. HTTP, SMTP, DNS. Proprietary protocols are privately owned, may need license to use e.g. skype

Identifying network process

1. IP address (globally unique address): Identifies the host
- IPv4: 32-bit number, dotted decimal (192.168.0.1)
- IPv6: 128-bit, Hexadecimal (2001:0db8:85a3:0000:0000:8a2e:0370:7334)

2. Port Number (Locally Unique Name): Identifies the process - assigned by IANA, 16-bit number (1 to 65535), 1 to 1023 are reserved

TCP (Transmission Control Protocol) - stream abstraction (vs **UDP** datagram abstraction, connectionless, unreliable)

1. **connection oriented** - reliable conn must be established between 2 devices before exchange data (3-way handshake)
2. **flow controlled** - prevent sender from flooding receiver
3. **congestion controlled** – throttle sender when network overloaded
4. **reliable** – guarantees data deliver (not throughput/delay)
5. does not provide timing, min throughput, security.

- Web page usually is base HTML file with several other referenced objects. Objects can be HTML file, JPEG, etc.
- Web objects addressable by Uniform Resource Locator

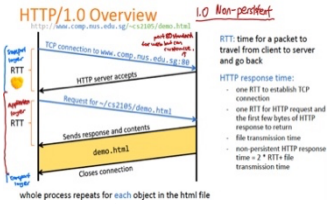
www.comp.nus.edu.sg/~cs2105/img/doge.jpg

host name | path name

(URL) e.g.
- Web resources request n receive using HyperText Transfer Protocol. http 1.0: RFC 1945, 1.1: RFC 2616

HTTP (transport service over TCP)
- IP address is NOT in http req msg
- stateless (server maintain no info about past client requests), cookies maintain state

1. cookie header field of HTTP req / response messages
2. cookie file kept on user's host, managed by user's browser
3. back-end database at Web site



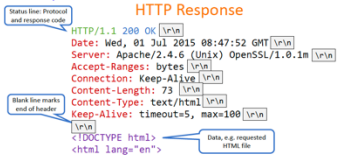
HTTP 1.0: non-persistent
- At most one object is sent over one TCP connection.
- connection is then closed.
- Download multiple objects need multiple conns.
- TCP connections may be launched in parallel

HTTP 1.1: persistent
- Server leaves conn open after sending a Web object.
- Multiple objects can be sent over a single TCP conn.
- Reqs may be sent in parallel
- Pipelining
- Head of line blocking (responses must be returned in order)

HTTP 2 has multiplexing

HTTP request format

reqtype/method sp path sp version \r \n (this is request line)
header field : sp value \r \n (req headers, can have multiple)
\r \n body



Date = Server response time Content-Length = No. of bytes in (Content-type) being returned

telnet x.com 80
- connect to x.com on OWN port 80 (http), test if port open
- DNS lookup for x.com IP
- TCP SYN packet sent to x.com

curl https://example.com
- fetch example.com content (HTTP GET)
- API request example: curl -X GET https://api.example.com/data -H "Authorization: Bearer token"

HTTP Status Code (the number)

200 Ok – req gd reqed obj follows
301 Moved Permanently - New location to follow
304 Not Modified - Object no change since date/time
403 Forbidden - Server refuse show webpage
404 Not Found
500 Internet Server Error - Unspecified error

DNS (Domain Name System)
- protocol that runs over UDP/53
- host -> ip (forward DNS lookup)
- ip -> host (reverse DNS lookup)
- translates between host name (e.g. www.comp.nus.edu.sg) and IP address. Client must carry out a DNS query to determine IP address corresponding to the server/host name prior to the connection.

Mapping between host names and IP addresses (and others) are stored as Resouce Records (RR)

RR Format: <name, value, type, TTL>

Type	Name	Value
A (adress)	Hostname	IP Address
NS (name server)	Domain, e.g nus.edu.sg	Hostname of authoritative name server for domain
CNAME (canonical name)	Alias for real name, e.g. www.comp.nus.edu.sg	The real name, e.g. www0.comp.nus.edu.sg
MX (mail exchange)	Domain of email address, e.g. xyz@gmail.com	Name of mail server managing the domain

nslookup/dig www.site.com any
- nslookup finds DNS mapping between hostname and IP address. dig similar, more in-depth

RR stored in distributed databases implemented in a hierarchy of many name servers. 13 root name servers worldwide. **root (top) - .edu (middle) - .mit, nyu**

Top-level domain (TLD) servers - responsible for .com, .org, .net, .edu, and all top-level country domains

Authoritative servers - organization's own DNS server(s)
- provides authoritative hostname to IP mappings for organization's named hosts (e.g. Web, mail)
- can be maintained by org or service provider

Local DNS Server - not strictly belong to hierarchy
- Each ISP (residential ISP, company, university) has one, also called "default name server"
- host makes DNS query -> query sent to local DNS server. name-to-address translation retrieved from local cache
- Local DNS server acts as proxy and forwards query into hierarchy if answer is not found locally.

DNS queries

Recursive: requesting host (on nus network) -> local DNS server (ns1.nus.edu.sg) -> root DNS server -> TLD DNS server -> authoritative DNS server (a.ns.facebook.com) -> TLD DNS server -> root DNS server -> local DNS server -> requesting host

Iterative: requesting host -> local DNS server -> root DNS server -> local DNS server -> authoritative DNS server -> local DNS server -> requesting host

DNS caching (occurs when name server learns a mapping). Entries, may be out-of-date (best effort), expire after some time (TTL) - RFC 2136, if host changes IP address, this change might not propagate globally until all caches expire

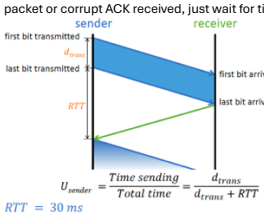
Lect 4 Reliable Protocols

Transport layer - resides on end hosts, process-to-process communication. Transport layer protocols run in hosts.
- Sender side: breaks app message into segments (as needed), passes them to network layer (aka IP layer).
- Receiver side: reassembles segments into message, passes it to app layer.
- Packet switches (routers) in between: only check destination IP address to decide routing.

Network layer - host2host, best effort, unreliable.
- Underlying network may corrupt, drop, re-order, or deliver packets after arbitrarily long delay
- End-to-end reliable transport service should guarantee packets delivery and correctness and deliver packets (to application) in the same order they are sent

Reliable data transfer
- **rdt 1.0**: perfectly reliable
- **rdt 2.0**: may corrupt (flip bits in) packets
  Stop-and-wait protocol; receiver sends ACK or NAK back
  Fatal flaw if ACK corrupted as sender will resend packet and receiver will treat as new packet
- **rdt 2.1**: To fix rdt 2.0, add 1-bit sequence number to each packet; receiver can now detect and discard duplicate packet (but must still send ACK for the duplicate packet)
- **rdt 2.2**: Same functionality as rdt 2.1, but is NAK-free; receiver ACKs sequence number of last received packet
- **rdt 3.0**: May corrupt packets, may lose packets, may incur arbitrary long packet delay. Sender waits "reasonable" amount of time for ACK, and retransmits if ACK is not received before timeout; sequence number included in both data and ACK just like rdt 2.2.

Both rdt 2.2 and 3.0: no action taken immediately if corrupt packet or corrupt ACK received, just wait for timeout



$U_{sender} = \frac{Time\ sending}{Total\ time} = \frac{d_{trans}}{d_{trans}+RTT}$

$RTT = 30\ ms$
$R = 1\ Gb/s$
$L = 8000\ bits$

$d_{trans} = {}^{L}/_{R} = 0.008\ ms$

throughput = $\frac{L}{RTT+d_{trans}} = \frac{8000}{30.008} = 267kbps$

$U_{sender} = \frac{d_{trans}}{RTT+d_{trans}} = \frac{0.008}{30+0.008} = 0.00027$

Transport layer multiplexing: data from multiple sockets sent using only one transmission channel

Multiplexing: When receiving packet from network layer, TCP/UDP must read transport header to decide which socket to deliver the message to (de-multiplexing); when sending messages from application layer, TCP/UDP must combine packets from different messages into the same network interface (multiplexing)

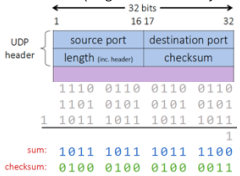
UDP connectionless de-multiplexing: decide using destination port only

TCP connection-oriented de-multiplexing: decide using (src IP addr, src port, dest IP, dest port)

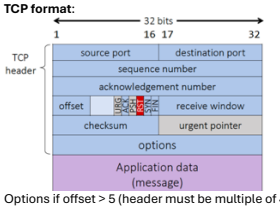
Lect 5 TCP, UDP - rdt is stop-and-wait protocol.

TCP, SR and Go-Back-N are pipelining protocols (sender send multiple pkt without waiting for acknowledgement)

UDP Format: (length field is no. of bytes of entire datagram)



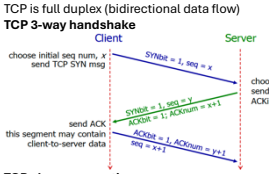
TCP format:



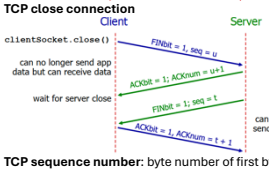
Options if offset > 5 (header must be multiple of 4 bytes or 32 bits, pad with 0s if needed). Offset min 5 (default header)

TCP is full duplex (bidirectional data flow)

TCP 3-way handshake



TCP close connection



TCP sequence number: byte number of first byte of data in a segment

Maximum segment size: maximum number of data bytes

Maximum packet size: includes header bytes

Event at TCP receiver	TCP receiver action	
Arrival of in-order segment with expected seq #. All data up to expected seq # already ACKed	Delayed ACK: wait up to 500ms for next segment. If no next segment, send ACK	
Arrival of in-order segment with expected seq #. One other segment has ACK pending	Immediately send single cumulative ACK, ACKing both in-order segments	
Arrival of out-of-order segment higher-than-expect seq. # (gap detected)	Immediately send *duplicate ACK*, indicating seq. # of next expected byte	
Arrival of segment that partially or completely fills gap	Immediately send ACK, provided that segment starts at lower end of gap	

Dynamic TCP timeout:

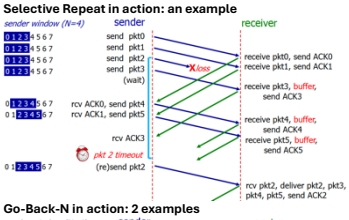
$SampleRTT :=$ RTT of new packet
$EstRTT \leftarrow (1-\alpha) \times EstRTT + \alpha \times SampleRTT$
(typically $\alpha = 0.125$)
$DevRTT \leftarrow (1-\beta) \times DevRTT + \beta \times |SampleRTT - EstRTT|$
(typically $\beta = 0.25$)
$TimeoutInterval \leftarrow EstRTT + 4 \times DevRTT$

TCP fast retransmission: (RFC 2001) If 3 duplicate ACKs (i.e. 4 in total) are received, next segment is treated as lost and thus retransmitted immediately.

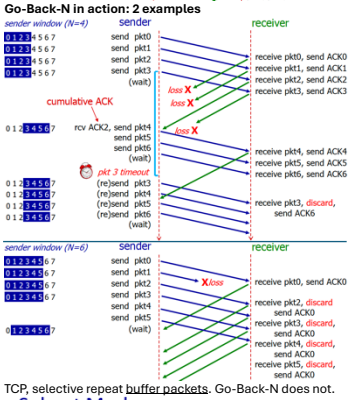
Maintains single timer and resends oldest unACKed packet on timeout; timer started only when prev. ACK is received

TCP flow control: receive window = 0 in ACK, sender wait first, then send 0-window probe (header only) for new ACK

Selective Repeat in action: an example



Go-Back-N in action: 2 examples



TCP, selective repeat buffer packets. Go-Back-N does not.

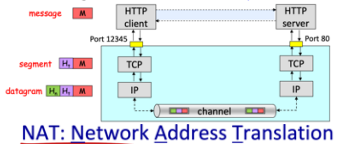
Subnet Mask

- IP address 192.168.0.102/24

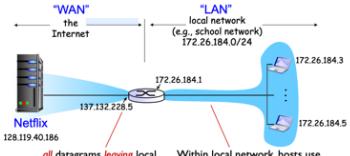
Network Address 11000000 10101000 00000000 00000000

this subnet contains 2^8 IP addresses

- Each IP datagram contains source and dest IP addresses.
  - Receiving host is identified by dest IP address.
  - Each IP datagram carries one transport-layer segment.
  - Each segment contains source and dest port numbers.



NAT: Network Address Translation



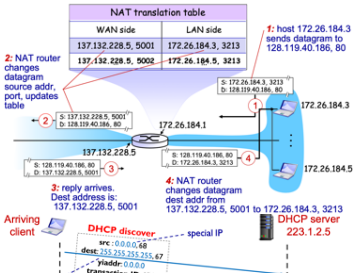
all datagrams leaving local network have the same source NAT IP address: 137.132.228.5

Within local network, hosts use private IP addresses 172.26.184.* for communication

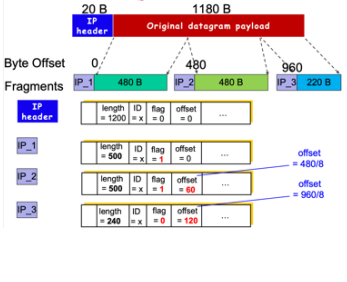
Dst IP	Dst Port	Src IP	Src Port
128.119.40.186	80	137.132.228.5	Src Port

We are only free to play with this

NAT: Illustration



IP Fragmentation Illustration



Lect 3 Socket Programming
- Host can run several processes (Top-level execution container, independent and isolated memory space)
• Threads run in a process and share same memory
• Socket is the abstraction interface between processes (application layer) and transport layer protocols.
- Process sends/receives messages to/from its socket.
-Conceptually: IP address + port no.
-Programming-wise: set of API calls
Stream socket uses TCP as **transport layer protocol**
•process establishes a connection to another process, while connection in place, data flow in continuous stream
- Server uses (client IP + port #) to distinguish clients
•TCP Server
-creates welcome/listening socket
-when contacted by client, forks a new socket for server process to communicate with client.
•TCP Client
-creates a socket to establish a connection with server
-multiple sockets for multiple connections
Every connection has its own socket instance
- two processes communicate as if there is a pipe between them. The pipe remains in place until one of the two processes closes it.
-When one of the processes wants to send to other process, it simply writes data to that pipe.
- sending process doesn't need to attach destination IP address and port number to bytes in each attempt as logical pipe has been established (which is also reliable).
Datagram socket uses UDP
•Only one socket is needed (but can create more)
•Application creates each packet
-Specifies recipient (destination IP address + port)
-OS will attach return info (source IP + port)
•Receiver identifies sender
-extracting the source IP + port from received packet
- need to form packets explicitly and attach destination IP address / port number to every packet.
- No connection established before sending data
- Transmitted data may be lost or received out-of-order

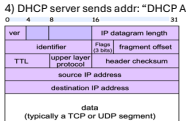
Web page = base HTML file + objects referenced by HTML file (e.g. img, css). HTTP runs on TCP (reliable delivery) runs on IP (routing across networks)
Lect 6 Network layer - between hosts
Packets travel through layers, links before dest. Router is device that forward packets between networks. Routing on the Internet is done hierarchically (Inet big, decentralised)
Aim: comms btw any 2 hosts on internet
Sub-problems: Path between all pairs of host needs to be determined. Need to define a protocol / service Guarantee. Every router on the path should cooperate. Every node/host on the internet should use the same protocol. Each host needs to be addressed. (IP address globally unique, 32-bit, assoc with every interface of host or router. Interface is conn btwn host/router and physical link. Usually router have multiple interfaces, host have 2 interfaces e.g. wired Ethernet, wireless 802.11) Systematic IP address allocation can reduce forwarding table size via address aggregation. Hosts within same subnet have same network prefix of IP address, interconnected w/o router, connect to external via router
IP: internet protocol (thin waist, only one in nw layer) ISP: internet service provider
Organisation get block of IP addresses by buying from registry or renting from ISP's address space. ISP get block from ICANN: Internet Corporation for Assigned Names and Numbers, which Allocates addresses, Manages DNS, Assigns domain names, resolves disputes. NUS bought 137.132.0.0/16

Network	IP Subnets
Research	COM1 : 137.132.82.0/24, 172.26.186.0/23 COM2 : 137.132.83.0/24, 172.26.190.0/23 COM4 : 137.132.92.0/24, 172.28.176.0/23
Incubator	137.132.80.128/25
Wireless	NUSSTU (students): 172.25.96.0/20 NUS (staff): 172.25.120.0/21
Student Lab	COM1 : 172.18.181.0/24 COM2 : 192.168.106.0/24
Student VM	172.25.64.0/20

Special Addresses	Present Use
0.0.0.0/8	Non-routable meta-address for special use
127.0.0.0/8	Loopback address. A datagram sent to an address within this block loops back inside the host.
10.0.0.0/8 172.16.0.0/12 192.168.0.0/16	Private addresses, can be used without any coordination with an Internet registry.
255.255.255.255/32	Broadcast address. All hosts on the same subnet receive a datagram with such a destination address.

All special addresses not routable on backbone internet + not globally unique. Private addresses are routable within an organisation. Loopback: only routable inside the host itself. 0.0.0.0/8 is special use (e.g., "any" address); not routable anywhere. 255.255.255.255/32 for broadcast, not routing between subnets

Limited Broadcast
Address: 255.255.255.255
- Reach all hosts on same local subnet
- Used when the sender doesn't yet know the network's structure
- Example: DHCP Discover (when a device doesn't yet have an IP)
Directed Broadcast
- Address: last one in the subnet (all host bits = 1)
- Example: If subnet = 192.168.1.0/24, dir broadcast address = 192.168.1.255
- Announcement to all devices within that specific subnet. Router can be configured to allow from outside subnet.
NAT routers must:
- Replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #).
- Remember (in stateful NAT translation table) mapping from (source IP address, port #) to (NAT IP address, new port #).
- Replace (NAT IP address, new port #) in destination fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT translation table.
NAT: Motivation and Benefits
- Just one public IP for the NAT router
- All hosts use private IP addresses. Can change addresses of hosts in the local network without notifying outside world.
- Can change ISP without changing addresses of hosts in local network.
- Hosts inside local network are not explicitly addressable and visible by outside world (a security plus).
16-bit port-number field can support up to 65535 simultaneous connections with 1 WAN-side address (0 is reserved)
eth0 shows private IP address, public will not be shown. Can find out public IP by searching on google 'what is my IP'
How host get an IP address?
- manually configured by system administrator
- Or, automatically assigned by a DHCP (Dynamic Host Configuration Protocol) server. DHCP runs on UDP. DHCP server port no 67, DHCP client port 68. msgs:
1) Host broadcasts "DHCP discover"
2) DHCP server responds "DHCP offer"
3) Host reqs IP address: "DHCP request"
4) DHCP server sends addr: "DHCP ACK"



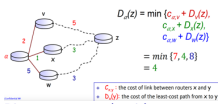
Fragment offset in IPv4 header indicates the starting byte of data within original (unfragmented) datagram. measured in units of 8 bytes. (e.g. offset 100 -> 800) It has nothing to do with payload length.
MTU = headers + payload
MSS = MTU – IP header – TCP header (MSS is TCP specific, UDP don't have)
Lect 7 Network Layer (Layer 3)
Abstract view intra-Autonomous System (AS) routing: graph; vertices are routers, edges are physical links btwn routers. Routing: finding a least cost path between two vertices/nodes in a graph. Constraint: each node only has info about immediate neighbour at t=0, get 1 hop more of info (receive DV from neighbours t hops away) with each t (State information diffusion)

Bellman-Ford Equation

N: 1-hop Neighborhood

$$D_a(z) = min_{a \in N}\{c(a,a) + D_a(z)\}$$

where min is taken over all direct neighbors a of a



Distance vector algorithm:

each node:
wait for (change in local link cost or msg from neighbor)
recompute DV estimates using DV received from neighbor
if DV to any destination has changed, notify neighbors

iterative, asynchronous: each local iteration caused by:
- local link cost change
- DV update message from neighbor

distributed, self-stopping:
each node notifies neighbors only when its DV changes
- neighbors then notify their neighbors – only if necessary
- no notification received, no actions taken!

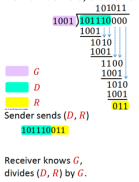
RIP (Routing Information Protocol) implements the DV algorithm.
- It uses hop count as the cost metric (i.e., insensitive to network congestion).
- Exchange routing table every 30 seconds over UDP port 520.
- "Self-repair": if no update from a neighbouring router for 3 minutes, assume neighbour has failed.
- Distributed: Each node receives information from one or more of its directly attached neighbors, performs calculation and then distributes the result back to its neighbors.
- Iterative: The process continues until no more information is available to be exchanged between neighbors.
- Asynchronous: It does not require that all of its nodes operate in the lock step with each other.

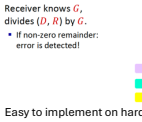
Link-State Algorithms e.g. OSPF (Open Shortest Path First)
- Routers broadcast link costs to all other routers in the AS.
- Each router receives all link-state advertisements (LSAs), builds the same global view of the network graph.
- All routers independently compute their own forwarding tables from the same information.
- Each router uses Dijkstra algorithm to compute least cost path from itself
Intra-AS routing e.g. RIP, OSPF
- Finds a good path between two routers within an AS.
- Single admin, so no policy decisions are needed.
- Routing mostly focus on performance.
ICMP (Internet Control Message Protocol) is network-layer protocol used by hosts & routers to communicate network-level information.
ICMP does not carry application data.
- Error reporting: unreachable host / network / port / protocol
- Echo request/reply (used by ping)
- ICMP messages are carried in IP datagrams. ICMP header starts after IP header.
ICMP header: Type + Code(Sub type) + Checksum + others.

type	code	descript	meaning
8	0	Echo request	Ping test reachability
0	0	Echo reply	
3	1	Dest host unreachable	Route can't deliver packet to host
3	3	Dest port unreachable	Host reachable but no service listening on that port (e.g. closed UDP port)
11	0	TTL expired	Packet discarded because TTL hit 0 (used by traceroute)
12	0	Bad IP header	IP header error e.g. incorrect checksum, version mismatch

Traceroute: last number is no. of hops
Traceroute sends 3 packets per hop
Multiple IP in one hop: load balancing (multiple paths), single router with multiple interfaces, router aliasing
Data plane: local, per-router function that determines how datagram arriving on router input port is forwarded to router output port
Control plane: (external) network-wide logic that determines how datagram is routed among routers along end-end path from src host to dest host. Two approaches: traditional routing algos in routers, external software-defined networking (SDN) in remote servers.
TCAMs (ternary content addressable memories) is used to perform longest prefix matching. It returns location of data matching a pattern. Can always retrieve address in 1 clock cycle.
Router input ports -> high-speed switching fabric (forwarding data plane (hardware), operates in ns), controlled by routing processor (management control plane (software), operates in ms) -> router output ports
replicated, application-layer distributed services (data centers, content distribution networks) connecting close to clients' networks, allow service from multiple locations.
DHCP, BGP (Border Gateway Protocol. TCP port 179), RIP: all application layer
Lect 8 Link Layer (Layer 2)
Link: communication channels that connect adjacent nodes. Aim: send data between n nodes via cable. Solution 1: interconnect the n nodes and send data. Each link needs to be addressed.
Drawback: does not scale, n-1 links needed Solution 2: Connect n nodes via a broadcast (shared) link. Each link needs to be addressed, Need to define a protocol, Need to handle errors
Link layer sends datagram between adjacent nodes (hosts/routers) over a single link.
- IP datagrams are encapsulated in link-layer frames for transmission.
- Different link-layer protocols may be used on different links.
- data-link layer has responsibility of transferring datagram from one node to physically adjacent node over a link
Possible Link Layer Services
Framing: encapsulate datagram into frame (+header, payload, +trailer)
Link access control: multiple nodes share single link, need coordinate which nodes can send frames at a point in time
Error detection: usually caused by signal attenuation/noise. Receiver detects, may signal sender to retransmit or drop
Error correction: Receiver identifies and corrects bit error(s) (no retransmission)
Reliable delivery: Seldom used on low bit-error link (e.g., fiber) but often used on error-prone links (e.g., wireless link)
Link layer implementation - In adapter (NIC) (e.g. ethernet card, wi-fi adapter) or chip. Adapters semi-autonomous (implement link + physical layers)
Point-to-point Link network link
- Sender and receiver/host and router connected by a dedicated link. No need for multiple access control.
- E.g. Point-to-Point Protocol (PPP), Serial Line Internet Protocol (SLIP)
Broadcast Link network link
- Multiple nodes connected to a shared broadcast channel. A node transmits a frame, channel broadcasts the frame, and every other node receives a copy
Multiple Access Protocols
- in broadcast channel, Collision if node receives >1 signal at the same time
- random access most complex, channel partitioning least complex
- Random Access: channel not divided, collisions possible (need "recover")
- Taking Turns: for each node to transmit
- Channel Partitioning: divide channel into fixed, smaller pieces (e.g. timeslot or frequency), allocate piece to node for exclusive use
Ideal Multiple Access Protocol - Given broadcast channel of R bits per second:
1) Collision free
2) Efficient (when only one node wants to transmit, can send at rate R)
3) Fairness (when M nodes want to transmit, can send at average rate R / M)
4) fully decentralised (no coordinator)
NO out-of band channel signalling
Random Access Protocols
- how to detect, recover from collisions
Slotted ALOHA: no coordination, if have data, transmit at full channel rate R. All frames of equal size, L bits. Time divided into equal slots of the time needed to transmit 1 frame, L/R. Transmission can only begin at the start of a slot. Fail if collision. Retry in each subsequent slot with probability p until success.
NOT collision free, efficient when only 1 node (throughput of R), NOT efficient when many active nodes (max 37% as slots wasted due to collision and empty), perfectly fair, decentralised
Pure (unslotted) ALOHA: no time slots. When node has fresh frame, transmit entire frame immediately. If collision, wait for 1 frame, then retry with probability p until success. Chance of collision increases as frame sent at t0 will collide with any other frame sent in t0 – 1, t0 + 1. NOT collision free, NOT efficient when many active nodes (max 18%). Everything else same as slotted. Both vers ALOHA – downside: probability of collision in all subsequent timeslots remains the same, can even increase if a new node starts transmitting
Example: wireless pkt switched network

CSMA: if channel sensed idle, transmit entire frame. If channel sensed busy, defer transmission (wait for it to idle). Still can hv collision (propagation delay).
CSMA/CD: If collision detected, abort transmission and retry after a random delay. Binary exponential backoff: Retransmission attempts to estimate current load. More collisions implies heavier load, so longer backoff interval. After 1st collision: choose K at random from {0,1}. p = 1/2
After 2nd collision: choose K at random from {0,1,2,3}. p = 1/4
After mth collision: choose K at random from {0,1, …, 2^m – 1}. p = 1/(2^m)
Must wait K time units before retrying. In Ethernet, 1 time unit = 512 bit transmission times. K * 512 / 10^7 ms delay if on 10Mbps Ethernet.
CSMA and CSMA/CD are NOT collision free, are efficient, fair, decentralised
Minimum frame size (imposed by CSMA and CSMA/CD): ensure collisions are detected. **2 max(dprop) <= dtrans**
dprop directly proportional to diameter (max dist between any 2 nodes in the network). dtrans directly proportional to frame size and inversely proportional to link transmission rate.
Ethernet: min frame size is 64 bytes.
Taking Turns Protocols
Polling: one node is master. Master polls nodes in round-robin.
Collision free, higher efficiency but with polling overhead, perfectly fair, NOT decentralised (master is failure point)
Example: Bluetooth
Token passing: special frame (token). Collision free, higher efficiency but with token passing overhead, perfectly fair, decentralised. Downside: token loss disruptive (data loss and system bugs), node failure can break the ring
Example: FDDI, token ring
Channel Partitioning Protocols
Time Division Multiple Access (TDMA): Each node gets fixed length time slots in each round. Length of time slot = data frame transmission time. If N nodes, N timeslots together is the time frame. Collision free, inefficient as unused slots idle (max throughput is R/N), perfectly fair, decentralised. Example: GSM
Freq Division Multiple Access (FDMA): Channel spectrum divided into frequency band, each node given one. Collision free, inefficient (same as TDMA), perfectly fair, decentralised
Example: Radio, satellite systems
Error detection
Parity checking: Single bit
Suppose data D has d bits. Even parity: sender includes one additional bit. Value of this bit chosen such that total 1s in d+1 bits is even. Can detect odd number of single bit errors. Good if error probability independent (unlikely multiple errors). Limitation: Bursts (multiple bits corrupted together) - even numbers of errors more common. Probability of undetected errors (from even no. flipped bits) can approach 50%.
Parity checking: 2-D
The d bits are divided into i rows and j columns. Parity value is computed for each row and each column. Resulting i + j + 1 parity bits comprise link-layer frame's error detection bits. Can detect and correct single bit errors in data. Can detect any two-bit error in data.
Cyclic Redundancy Check (CRC)
For D (non-binary number) R is r digit error detection code. Need to generate R such that sender can easily compute R, receiver can verify integrity of D easily. r digit number G is called Generator.
Example: D = 21027845, r = 3, G = 401
1. Create new number X by appending r 9s to D. X = 21027845999 (Note that X = D x 10^r + (10^r – 1)
2. y = X % G = 281
3. Message M to transmit, M = X – y
M = 21027845999 – 281 = 21027845718
(M is divisible by G i.e. M mod G = 0)
CRC for D data bits (binary number): G is r +1 bits, R is r bit CRC. Calculations are modulo 2, no carries or borrows, XOR. Division: append r 0s to D, remainder gives R. Sender send (D, R). Receiver knows G, divides (D, R) by G. Remainder is 0, else error detected





Easy to implement on hardware. Powerful error-detection coding widely used in practice (e.g., Ethernet, Wi-Fi). Can detect all odd number of single bit errors. CRC r bits can detect all burst errors of up to r bits, burst errors greater than r bits with probability 1 – 0.5^r
CRC is also known as Polynomial code. A k-bit frame is regarded as the coefficient list for a polynomial with k terms, from x^(k–1) to x^ 0.
Lect 9 Link Layer (Layer 2)
Switches in interconnecting subnets in a LAN (Local Area Network - interconnects computers within a geographical area) LAN technologies include IBM Token Ring (IEEE 802.5), Ethernet (IEEE 802.3), Wi-Fi (IEEE 802.11), Asynchronous Transfer Mode(ATM), and others.
- Ethernet simpler, cheaper vs TR, ATM
- Across Ethernet standards:
Can have diff speeds, diff physical layer media (e.g. cable, fiber optics) BUT link layer - Media Access Control (MAC) protocol and frame format - unchanged.
8 bytes preamble: 7 bytes of 10101010, then start of frame byte 10101011. Synchronises receiver, sender clock rate. Square wave tells receiver bit width
6 bytes destination MAC addr, 6 bytes source MAC addr, 2 bytes type (payload protocol eg IPv4 is 0x0800. This field allows Ethernet to multiplex network-layer protocols as hosts may not use IP, and is analogous to protocol field in network-layer datagram and port-number fields in transport-layer segment), 46 to1500 bytes data payload, 4 bytes CRC
Sending an IP datagram from one host to another on the same Ethernet LAN: the sending NIC adapter encapsulates IP datagram in Ethernet frame.
If NIC receives a frame with matching MAC address (dest or broadcast i.e. FF:FF:FF:FF:FF:FF), it passes data in the frame to network layer protocol. Otherwise, NIC discards frame.
Ethernet Data Delivery Service
Unreliable (receiving NIC does not send ACK or NAK to sending NIC. Data in dropped frames only recovered if initial sender uses higher layer rdt e.g. TCP). Ethernet use CSMA/CD + expo backoff.
Bus topology is broadcast LAN
Bus: backbone cable, Star: modular
Star topology Hub nodes directly connected to hub - physical-layer device that acts on indiv bits rather than frames. When a bit arrives from one interface, hub re-creates the bit, boosts its energy strength, and transmits the bit onto all the other interfaces.
Star topology Switch nodes directly connected to switch – layer-2 device that works on frames rather than indiv bits. Bona-fide store-and-forward (MAC) packet switch. No collisions here! Transparent (hosts unaware of switch), Plug-and play (self-learn, no need configure), switches buffer packets.
Switch table format: < MAC addr of host, interface to reach host, TTL >
When frame received at switch:
1) record MAC addr, interface of sender
2) index the table using MAC dest addr
3) if entry found for dest - if dest on segment from which frame arrived, drop frame. Else forward frame to dest
4) No entry found: flood (broadcast – forward to all interfaces except arriving)
MAC address allocation done by IEEE. First 3 bytes identifies adapter vendor. MAC address: permanent, link-layer, single-link. IP address: dynamically assigned, hierarchical, network-layer.
ARP (Address Resolution Protocol, RFC 826) - host can discover MAC addresses of other nodes in the same subnet. ARP table: < IP addr, MAC addr, TTL (then address mapping forgotten, usu mins) >.
Sending data to a new IP (e.g. using ping) triggers an ARP request, and the result is cached for subsequent use. ARP is plug-and-play (nodes create ARP tables without network admin intervention)
Lect 10 Network Security goals:
1. Confidentiality (Prevent unauthorized access to information / eavesdropping)
2. Integrity (Prevent undetected modification of data) 3. Authentication (Ensure the sender is who they claim to be, prevent impersonation / repudiation)
Plaintext "This" -> Caesar cipher shift 4 -> Ciphertext "Xlmw"
Monoalphabetic and Caesar's are sub
Polyalphabetic cycle through n sub
Block cipher: K-bit blocks. 2^K possible input blocks. Each block encrypted independently with 1-1 mapping, possible mappings = possible keys = (2^K) ! Note: AES only 128, 192, or 256 bit keys (i.e. AES-128 use 128-bit key which restricts us to 2^128 keys, …)
DES 56-bit symmetric key, 64-bit block. Not considered secure, easy brute force. More secure is 3DES (encrypt 3 times with 3 diff keys)
Public-key cryptography - enables effective encryption in the SSL (secure socket layer) of the HTTPS protocol
- first proposed by Diffie and Hellman
- If you want to **protect confidentiality** (only the target can read it): → Encrypt with the receiver's public key. Receiver will decrypt it using his own private key.
- If you want to **prove authenticity** (prove *you* sent it): → Sign/encrypt with your own private key. Anyone can verify using your public key.
Modulo arithmetic
$[(a \bmod n) + (b \bmod n)] \bmod n = (a+b \bmod n)$
$[(a \bmod n) - (b \bmod n)] \bmod n = (a-b \bmod n)$
$[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b \bmod n)$
$(a \bmod n)^d \bmod n = a^d \bmod n$
RSA (Rivest, Shamir, Adleman algorithm)
1. choose two large prime numbers p, q.
2. compute $n = pq$, $z = (p-1)(q-1)$
3. choose e (with $e < n$) that has no common factors with z (i.e., e and z are "relatively prime").
4. choose d such that $ed-1$ is exactly divisible by z. (in other words: $ed \bmod z = 1$).
5. public key is (n, e). private key is (n, d).
- to encrypt message m (Note: $m < n$), compute $c = m^e \bmod n$
- to decrypt received bit pattern, c, compute $c^d \bmod n$ (get back m)
RSA in practice: session keys
- RSA is computationally intensive
- DES is at least 100 times faster than RSA, but needs prior knowledge of K_S
- To combine: 1. Select a Key K_S 2. Use RSA to transfer K_S 3. Use K_S as the symmetric key in DES for encrypting data for this session
K_S = symmetric key = session key
Lect 11 Network Security II
Hash function take input, produce fixed size msg digest (fingerprint). Crypto-graphic: resistant to msg substitution.
MD5 compute 128-bit, SHA-1 compute 160-bit message digests respectively. Both are cryptographically broken (bad)
- Hash same input twice -> same output
- MAC - Msg Auth Code, refers to H(m+s)
- Secret should always be mixed into the input of hash, not added after hashing
CA (Certification Authority)
- signs certificates with CA's private key
- maintains dir of websites' public keys
- CA's public key should be securely distributed to all involved entities. Pre-installed in browser or OS (root certs)
Browser/Client verifies certs with CA's public key
Certificate: Digital document, contains minimally:
1. owner (e.g. bob@nus.edu, google.com) 's identity
2. owner's public key
3. time window that cert is valid
4. signature of CA
Optionally: additional info like intended purpose of cert e.g. client authentication/secure email