ANSWERS FOR THE BOYS/GIRLS - MAY THE FORCE BE WITH YOU

https://www.google.com.au/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwiz9Ke86abXAhXHoJQKHeBZAIoQFggoMAA&url=https%3A%2F%2Fwebcms3.cse.unsw.edu.au%2Ffiles%2F081ef53557df32a2b8ee118f0938d6124b36b7512e91146129f2fff4c3d94baa%2Fattachment&usg=AOvVaw2pen6816NF-I2QRvp7aPrS

Chapter 0 - Sample exam questions

Question 2
Assume that the SendBase for a TCP Reno sender is currently 4000. The TCP sender has sent four TCP segments with sequence numbers 4000, 4500, 5500 and 7000. The sender then receives a segment with an acknowledgement number 7500 and a receive window 6000. The congestion window, CongWin, is set to 10000 bytes after this ACK is processed. Answer the questions (i)-(iii) assuming that this ACK is processed and no further ACKs are received:
(i) SendBase is set to 7500.
(ii) First segment carries 500 bytes, second segment carries 1000 bytes, third one carries 1500 bytes and the last segment carries 500 bytes. Thus, the total data in the four segments is 3500 bytes.
(iii) The window size is set to the minimum of congestion window and receive window. Hence, the window size will now be set to 6000 bytes. Since current SendBase is 7500, this implies that the last byte that the sender can send with certainty is 13499.
(iv) Since the sender receives three duplicate ACKs, the CongWin is reduced to half the current value (current value = 10000 bytes), which is 5000 bytes.
(v) Since the sender received three duplicate ACKs for 7500, it will now retransmit the segment with sequence number 7500.

Question 3 (X marks) In the lecture (and in the text) we observed how the AIMD algorithm implemented by TCP enables two TCP connections sharing a bottleneck link to achieve a fair share of the bottleneck link capacity (see Figure 3.56 in the text). Suppose that instead of a multiplicative decrease TCP decreased the window size by a constant amount. Would the

resulting AIAD algorithm converge to an equal share algorithm? Justify your answer using a diagram similar to Figure 3.56.

Question 4 (X marks)
Consider that only a single TCP Reno connection uses one 10Mbps link which does not buffer any data. Suppose that this link is the only congested link between the sending and receiving hosts. Assume that the TCP sender has a huge file to send to the receiver and the receiver's receive buffer is much larger than the congestion window. We also make the following assumptions: each TCP segment is 1,500 bytes; the two-way propagation delay of this connection is 150 msec; and this TCP connection is always in congestion avoidance phase, that is ignore slow start.
(a) What is the maximum window size (in segments) that this TCP connection can achieve?
10x10^6*0.15 = 15x10^5 bits in the RTT. 15x10^5/(8*1500) = 125 segments in this time.
(b) What is the average window size (in segments) and average throughput (in bps) of this TCP connection?
(c) How long would it take for this TCP connection to reach its maximum window again after recovering from a packet loss?

Question 5 (X marks)

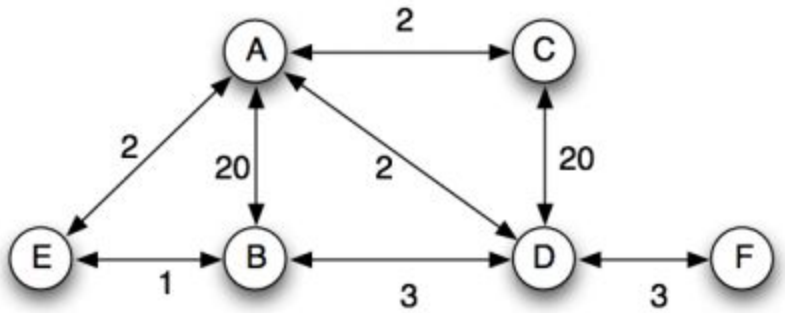


Figure 1: Figure for Question 5

a) Consider the network topology shown in the figure above. Show the operation of Dijkstra's link-state algorithm to compute routes from node A to all destinations.

Step	Searched nodes	B	C	D	E	F
0	A	20, A	2, A	2, A	2, A	inf
1	AC					
2	ACD	5, D				5, F
3	ACDE	3, B				

4	ACDEB					
5	ACDEBF					

b) Show the distance table that would be computed by the distance vector algorithm in A once the distance vector algorithm has finished executing. You do not have to run the distance vector algorithm; you should be able to compute the table by inspection. Note: make sure you have a row in the distance table for each neighbour of A.

From A to Neighbour	Min dist/cost
B	3
C	2
D	2
E	2
F	5

c) Consider three instances of link failure: (i) Link A-B fails, (ii) Link A-C fails, and (iii) Link A-E fails. In each of these three cases, all links except the one mentioned are still active. In which of the three instances of link failure will the count-to-infinity problem occur? Describe briefly how this problem occurs for the chosen instance. Name a solution to the count-to-infinity problem.
count-to-infinity problem occur
When A-C goes down, A says "I have no link to C"
D then says "I have a link to C of cost 4"
A says "I am 2 away from D, so the cost to C is 2 + 4 = 6"
D says "I am two away from A, so the cost to C is 6 + 2 = 8"
…
A says I am 20 away from D so the cost to C is 20 + 2 = 22"
D says I have a direct link to C of cost 20 so the cost to C is NOT 22, it is 20.

At this stage the system stabilises at the correct values.

One solution, split horizon rule: A node will not advertise to another node a route that includes that node. I.e. D would not advertise to A the path of cost 4, instead it would advertise 20 straightaway.

Another solution is poison reverse: **A router actively advertises routes as unreachable over the interface over which they were learned by setting the route cost to infinity. However, this significantly increases the size of routing announcements.**

Chapter 3 - The Transport Layer

Quick Notes - Feel free to add please:

Max Window size: W*MSS/RTT = Max Transfer Speed. Solve for W
SendBase value is set to most recent largest ACK from receiver assuming no errors.

With AIMD (Additive Increase Multiplicative Decrease):

- Window size varies from W to W/2. Therefore avg window size = 0.75W. Window size is halved in the event of an error or retransmission.
- Three duplicate ACKs will lead to a re-transmission of packets and halving of window size (With AIMD). The sequence number of the re-transmission will be the number of the duplicate ACK.

With a slow start System, Timeouts send Window size back to 1, but it makes use of the previously known window size and increases more quickly.

SampleRTT is not calculated on retransmitted packets in the event it receives the original ACK from the receiver, which would calculate an extremely short RTT mistaking it for the ACK for the retransmitted packet.

TCP-Tahoe

- cwnd =1 on triple dup ACK & timeout

TCP-Reno

- cwnd =1 on timeout
- cwnd = cwnd/2 on triple dup ACK

TCP-newReno

- Same as TCP-Reno above
- improved fast recovery

Chapter 4 - The Network Layer

The goal of the Network layer is to send data, packets, from a host to a receiver.

Forwarding: When a router receives a packet, it must figure out where to send the packet along so that it will end up at the host.

Routing: The network must figure out a path from the Host to the Receiver. This is calculated by a routing algorithm

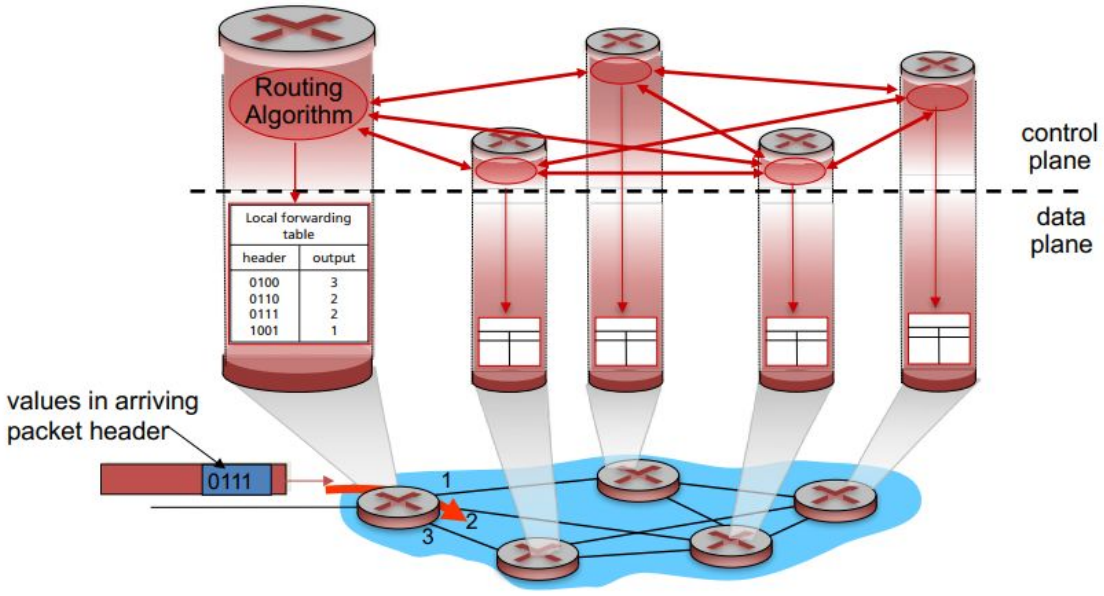
Every Router has a **forwarding table**. A field in the packet header is used to index into the forwarding table. This header value could be the destination address or an identifier that shows which connection the packet belongs to. This depends on the network layer protocol.

Routing Algorithms may be centralized or decentralized. A centralized algorithm executes on a central site and downloads routing information to each router. A decentralized algorithm is distributed and run across each router.

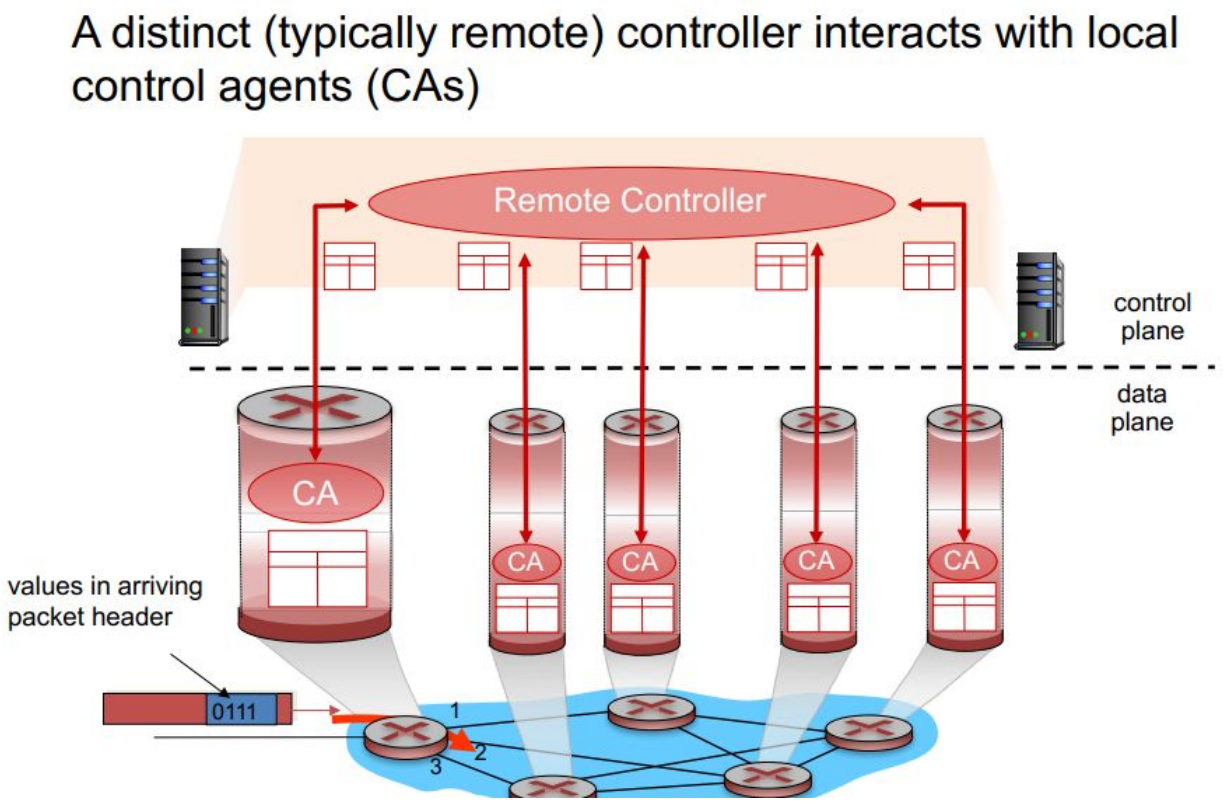
The **Data Plane** determines how datagrams arriving into the router is sent to the output port. It performs the **forwarding function.**

The Control Pane is Network wide logic, and determines how a datagram will be routed from the source to the host. It contains the **Routing Algorithm.**

A decentralized Control Pane



A Centralized Control Pane



Packet-Switch is any device takes a packet from an input link to an output link, as according to the header in the packet.
Link Layer Switches make forwarding decisions based off fields in the link layer frame.
Routers base their forwarding decisions off the value in the Network layer field. Routers are a Network layer (level 3) device but must implement layer 2 protocols.

ATM, frame relay, and MPLS require routers along a chosen path to handshake with one another to set up a state for network layer packet sending, where source to destination packets can flow. This is called **connection setup.**

The Network Service Model provides possible characteristics of end to end packet transportation.

- Guaranteed Delivery (eventually)
- Guaranteed Delivery with bounded delay
- In order packet delivery
- Guaranteed minimum bandwidth
- Guaranteed maximum jitter (the amount of time between the transmission of two successive packets at the sender is equal to the amount of time between their receipt at the destination (or that this spacing changes by no more than some specified value).

- Security Services (Such as a secret session key known only by the source and dest to encrypt and decrypt packet data)

There are infinite configurations / features you could have.

Comparisons - Table 4.1

Network Architecture	Service Model	Bandwidth Guarantee	No-Loss Guarantee	Ordering	Timing	Congestion Indication
Internet	Best Effort	None	None	Any order Possible	Not maintained	None
ATM	CBR	Guaranteed constant rate	Yes	In order	Maintained	No Congestion
ATM	ABR	Guaranteed Minimum	None	In order	Not maintained	Congestion indication provided

Best Effort == poor bloody effort.
A network that delivers no packets can still be considered best effort.

Constant Bit Rate (CBR) ATM Network:

- Provides a connection similar to a dedicated fixed bandwidth transmission.
- Any lost or delayed packets (cells in this case) will not exceed a specified value. (the CBR is agreed upon by host and dest)

Available Bit Rate (ABR) ATM Network:

- A bit better than Best Effort
- Packets cannot be recorded, though can be lost.
- A minimum transmission rate is guaranteed, but if available bandwidth is available it can exceed that.
- Can provide congestion feedback to sender.

4.2 Virtual Circuit and Datagram Networks

Connection vs Connectionless Networks, e.g TCP vs UDP.
Similarly, a connection based Network uses Handshakes, a connectionless does not.
Regardless, there are key differences to the transport layer:

- Host-to-host services by the network layer for the transport layer vs transport layer for the application layer.
- A network layer must have only one type of network. Connection based network layers are called **Virtual Circuit networks**, connectionless are called **datagram networks.**

Virtual Circuit Networks
To have a Virtual Circuit Network you need:

- A path of links and/or routers
- VC Numbers, one for each node on the path
- Entries in forwarding table in each node on the path
- Nodes must replace the VC Number in the packet it forwards from its forwarding table.

When a VC is created, a new entry is added to the forwarding table. When it's terminated the entries are removed.

It's simpler and uses less bandwidth to change the packet header via the forwarding table. VC number picking can be independent of other links in the path.

VC networks must **maintain state**.
Three phases in a VC

Setup:

- Transport layer contacts Network layer with receiver address.
- Network layer determines path
- Sets up VC Numbers in forwarding tables.
- May reserve bandwidth for the transmission of packets.

Data Transfer:

- After setup occurs, data transfer is trivial, packets flow

VC Teardown:

- Sender/Receiver notifies the network layer to terminate. The other partner receives a signal of termination and forwarding tables are updated, reserved bandwidth removed.

In a VC network layer, routers along the path between the two end systems are involved in VC setup, and each router is **fully aware** of all the VCs passing through it. In contract, a Transport layer connection, Routers within the network are oblivious, the sender and receiver are in control.

Signalling messages initiate or terminate VC connections, and are defined by **signalling protocols.**

Datagram Networks
No state.
Packets have the address of their destination.
Routers have forwarding tables for destination address, and are mapped to link interfaces.

Forwarding Tables in this network use **prefix matching** to hold limited destination addresses. If an IP matches with multiple prefixes, **longest prefix matching rule** is used.

longest prefix matching
when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

examples:

DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

It's important to figure out where to send the packet quickly. We do this with **ternary content addressable memories (TCAM).** This means Destination Addresses will be found in one clock cycle regardless of table size.

Datagram network forwarding tables are modified **slowly, but can happen at any time**. Therefore you can see **packets take different routes to their destination.**

Packet buffers should be of size roughly AvgRTT*LinkCapacity.

4.3 What's inside a Router?

Input Ports:

- Terminates incoming physical links
- Interpolates the link layer at the other side of the incoming link
- Performs the forwarding table lookup function
- Forwards control packets to the routing processor

Switching Fabric: Connects input ports to output ports

- Network within a network

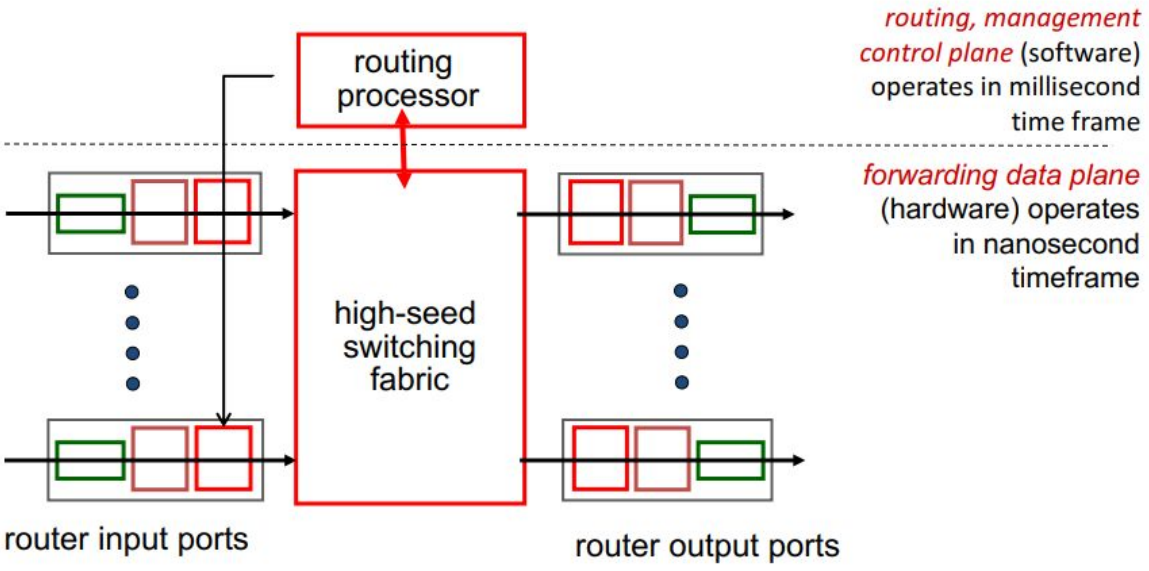
Output Ports:

- Gets packet from switching fabric and sends it.
- If bidirectional output ports can be linked with input ports.

Routing Processor:

- Executes routing protocols
- Maintains routing tables and link state info
- Computes new forwarding tables
- Network management stuff

Forwarding is done often by the hardware. Forwarding methods are referred to by the **router forwarding plane.** In contrast, software operations are performed by the **Router control plane** and operate slower at a millisecond or second time scale.

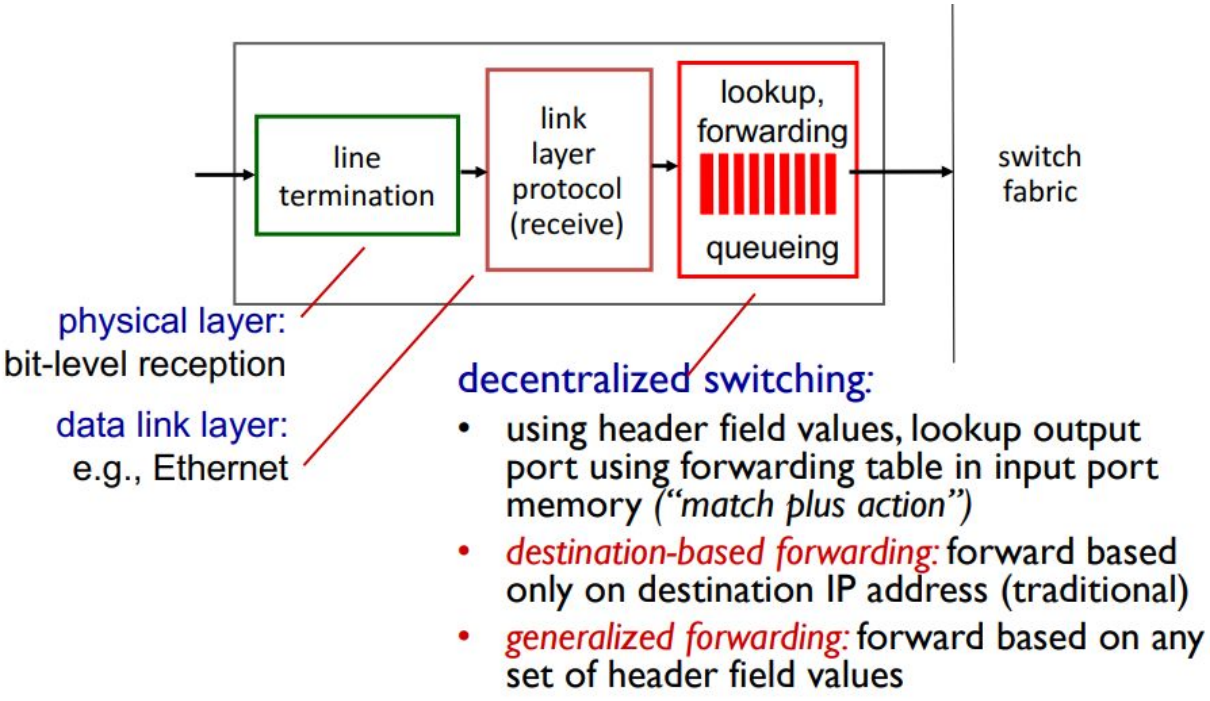


4.3.1 Input Processing

Lookups must be done quickly in order to maintain high speed internet connections.
Once a lookup is done, an output port is selected. In some cases a packet can be blocked in congestion, and will queue within the input port and scheduled by the switching fabric for a later time until it can be sent.

Some important processing steps

- physical- and link-layer processing must occur
- the packet's version number, checksum and time-to-live field must be checked and the latter two fields rewritten
- counters used for network management (such as the number of IP datagrams received) must be updated.



4.3.4 Where does Queueing occur?

Both Input and output ports can have queues.
High Traffic load -> Input queue
Slow Switcher -> Input queue
Line speed/Transmission Rate -> Output queue

Packetloss if Queue reaches maximum size.

In the event of scheduling at the output port, we need a **Packet Scheduler.**
It can be implemented as a First come first served, or weighted fair queuing which attempts to appease all connections, etc.
The implementation of the scheduler affects **Quality of Service guarantees.**

Discard Policies -

- A method which drops incoming packets over packets already in the queue is called **drop tail.**
- A method that makes an intelligent choice based a variety of factors is a **Priority Queue**.
- A method that drops random packets is a **Random** policy.

Scheduling Policies -

- **Priority Scheduling**. You could seperate packets into classes depending on header info, source dest, port numbers, etc.
- **Round Robin.** Seperate packets into classes, send one from each class each cycle.
- **Weighted Fair Queue.** Round Robin but each class has a priority/weight.

Packet Schedulers can make use of **active queue management** techniques such as:
Random Early Detection (RED)

- Weighted average maintained for length of output queue
- If average queue length is below a threshold, packets are allowed to enter the queue, else marked/dropped.
- Else, if inbetween [min, max], it is marked/dropped with a probability relating to its position between min and max.

A slow switcher can cause **Head of The Line Blocking.** An input packet queued with a free corresponding output port must wait because it is blocked by a packet at the front of the line.

The **Routing control pane** is *generally* **decentralized** and exists within the routing processor across various nodes in the network.

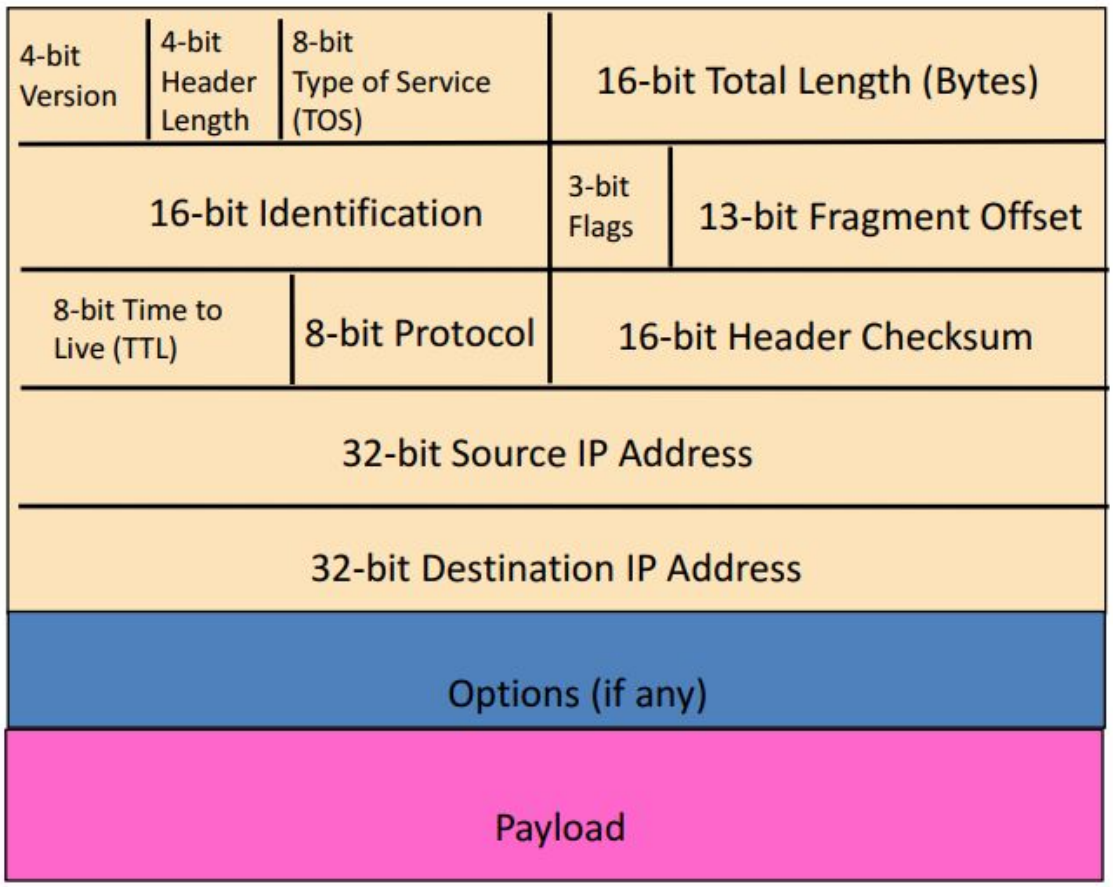
4.4 The Internet Protocol: Forwarding and Addressing the Internet :

The internet's Network layer has **three major components:**

1. IP Protocol.
2. Routing Component, finding paths
3. Reporting errors in Datagrams and responding to requests for Network layer information.

4.4.1 Datagram format

IPV4 Diagram



To read a packet correctly, you really need the **Version Number, Header Length,** and **Total Length** fields.

Version Number:

- 4 bits
- Specifies IP protocol version
- Used to figure out how to interpret the rest of the packet

Header Length:

- 4 bits. Header information can vary so need to figure out where data begins.

Type of Service:

- Different kinds of datagrams exist such as:
- Low delay, High throughput, reliability, real time, non-real time
- Tells the router to treat it differently.

Datagram Length

- Total length, header + data
- In Bytes
- Max size is 65,535 bytes, usually a lot smaller

Identifier, flags, fragmentation offset:

- These three fields deal with IP Fragmentation. IPv6 does not allow for fragmentation at routers.

TTL:

- Stops circulation forever within the network when this reaches 0. Decrements by one each time is passed by a router.

Protocol:

- Only used when Datagram reaches destination.
- Specifies Transport layer protocol (6 == TCP, 17 == UDP, etc)
- Protocol number is analogous to the Transport layer port number.
- Tells the end host how to handle the packet.

Header Checksum:

- Helps a router detect errors
- Treats each 2 bytes in the header as a number, sums these numbers using 1's complement.
- If the header checksum != the calculated checksum, an error has occurred.
- If an error is detected the datagram is usually discarded.
- TCP/IP Does error checking independently at both the Network layer and Transport layer.
- Recalculated at every Router.

Source/Dest IP Addresses:

- The creator of the datagram is the Source IP, the Destination is the destination IP address.
- Routers will use DNS' look up to determine/change(?) the destination Address.

Options:

- Allows for the extension of the IP header
- Complicates things, increasingly varies the length of IP addresses, not often used.
- Dropped in IPv6 headers.

Data:

- What we're sending yo

Overall, there are **20 bytes of TCP, 20 bytes of IP**, + any application overhead.

Potential Problems & solutions:

- **Header Corrupted** | **Checksum**
- **Endless Loop** | **TTL**
- **Packet Too large** | **Fragmentation**

IP Datagram Fragmentation

Splitting up bad boy Datagrams that are too large, with the fragmentation flag set to 1 (except for the last split up packet). Offset fields are set, and the packets are sent. Fragment pieces have the same ID number in the header.

There are lots of problems that can occur, Fragments overlapping, filled buffers, excessive fragmentation, incomplete datagrams, missing packets getting lost, etc.

IPv4 Addressing

A host typically only has one link into their Network.
Boundary between a host and physical link is called an **Interface.**
The boundary between a router and it's links is also called an Interface.
IP Addresses are associated with Interfaces.
IP Addresses are written in **dotted-decimal notation**, dot representing the start/end of a byte.
The first 8 bits are usually a network address
Last 24 bits is the host address.

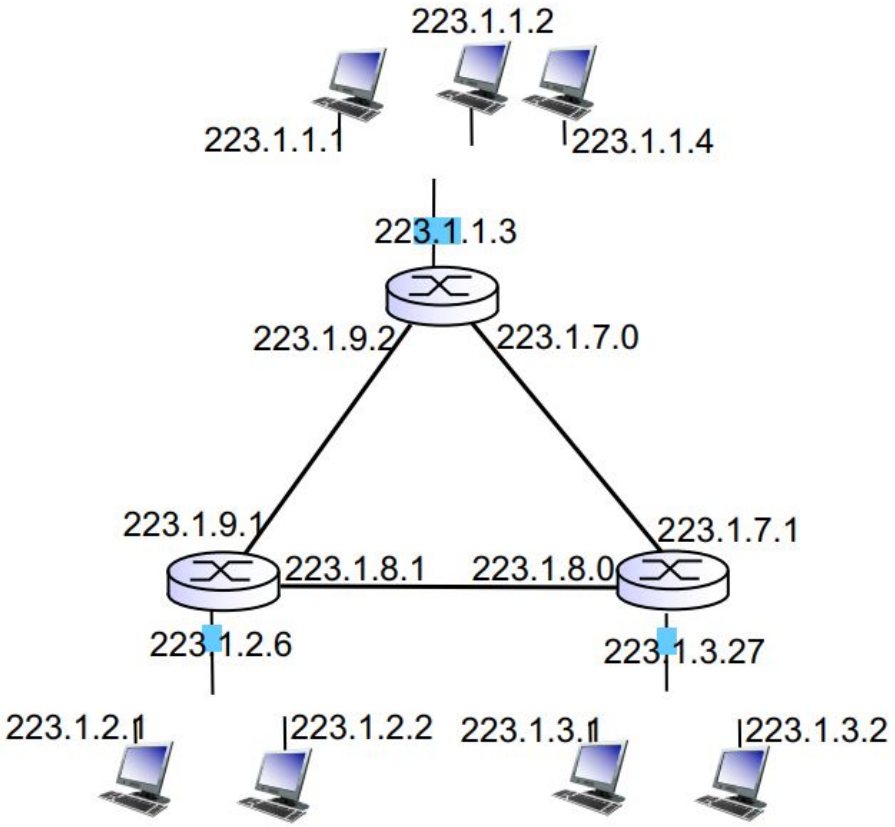
Subnets

Multiple Host's connected via **one** router interface forms a **subnet**.
Some Subnet properties include:

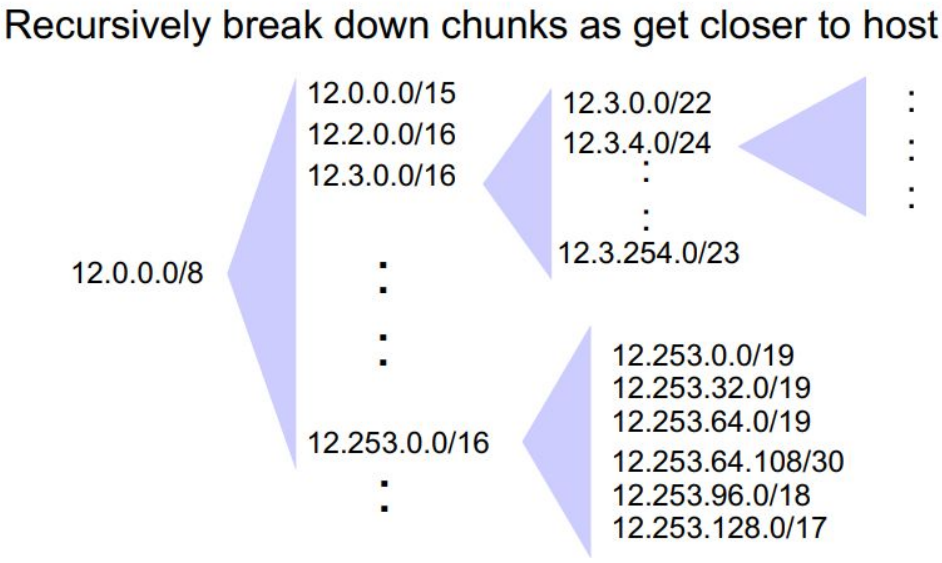
- IP Addresses in this network contain the **Subnet mask** and host portion.
- The subnet mask is a portion of the IP address that is shared by all hosts
- Subnet mask notation: 223.1.1.0/**24**. The first **24 bits** are the **subnet mask**. The last 8 bits **vary** by the **host**. E.g 223.1.1.1, 223.1.1.2, 223.1.1.3

To determine the subnets, detach each interface from its host or router, creating islands of isolated networks, with interfaces terminating the end points of the isolated networks. Each of these isolated networks is called a subnet.

There are 6 subnets in the image below. ***Links between routers are considered a subnet.***



Classless Interdomain Routing (CIDR) is used by the internet to assign addresses.
subnet portion of address of **arbitrary length**
address format: **a.b.c.d/x**, where **x** is **# bits** in subnet portion of address. Often called the network pre-fix.
IP Addresses within an organisation will share a common prefix.
Sending datagrams within an organisation is quicker as only information after the prefix is needed in the forwarding table.



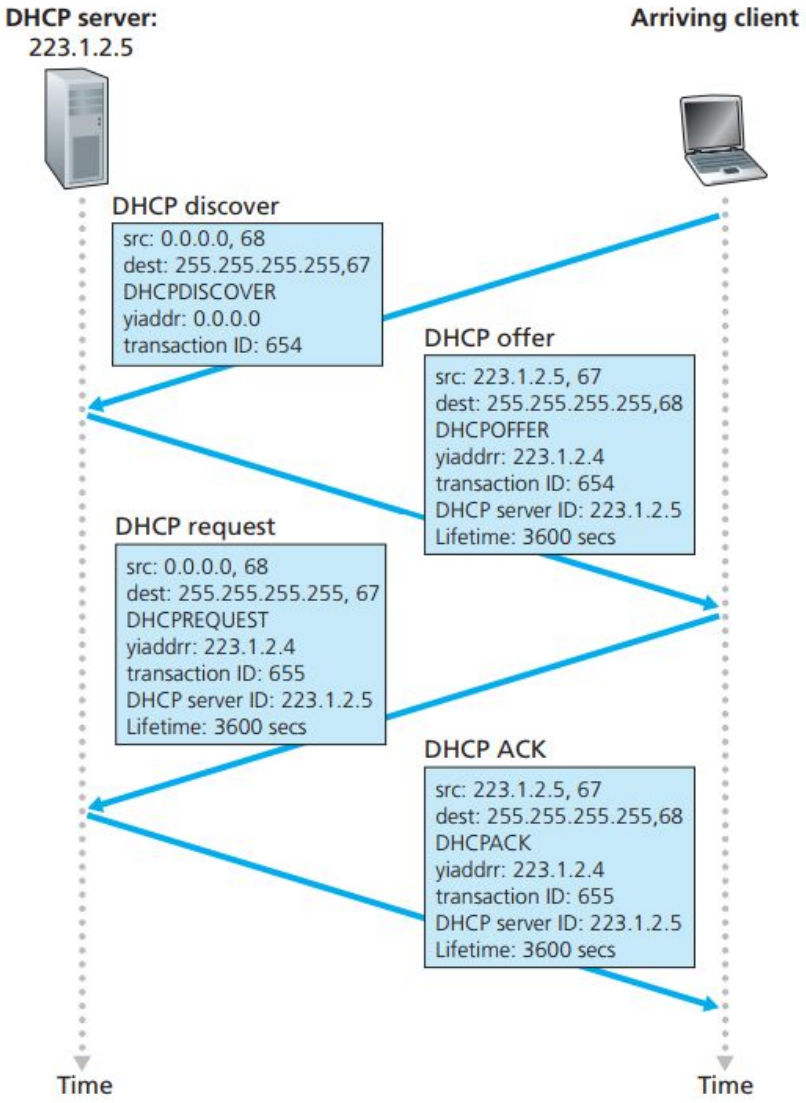
DHCP (Dynamic Host Configuration Protocol) allows the host to dynamically obtain its IP address from network server when it joins network. A host can receive the **same IP address** each time it connects to the network, or a host may be assigned a **temporary IP** address that will be different each time the host connects to the network.

Benefits:

- allows reuse of addresses (only hold address while connected/“ on ”)
- support for mobile users who want to join network (more shortly)
- Can return address of first-hop router for client
- Can return name and IP address of DNS sever
- Can return network mask (indicating network versus host portion of address

DHCP overview:

- host broadcasts “**DHCP discover**” msg (broadcast dest IP address of 255.255.255.255)
- DHCP server responds with “DHCP offer” msg to all nodes on the subnet, destination 255.255.255.255. There could be multiple DHCP servers, and this multiple offers for the client to choose from. Offers contain IP Address **lease time,** the IP, subnet mask, DNS servers and default gateway
- host requests IP address: “DHCP request” msg
- DHCP server sends address: “DHCP ack” msg

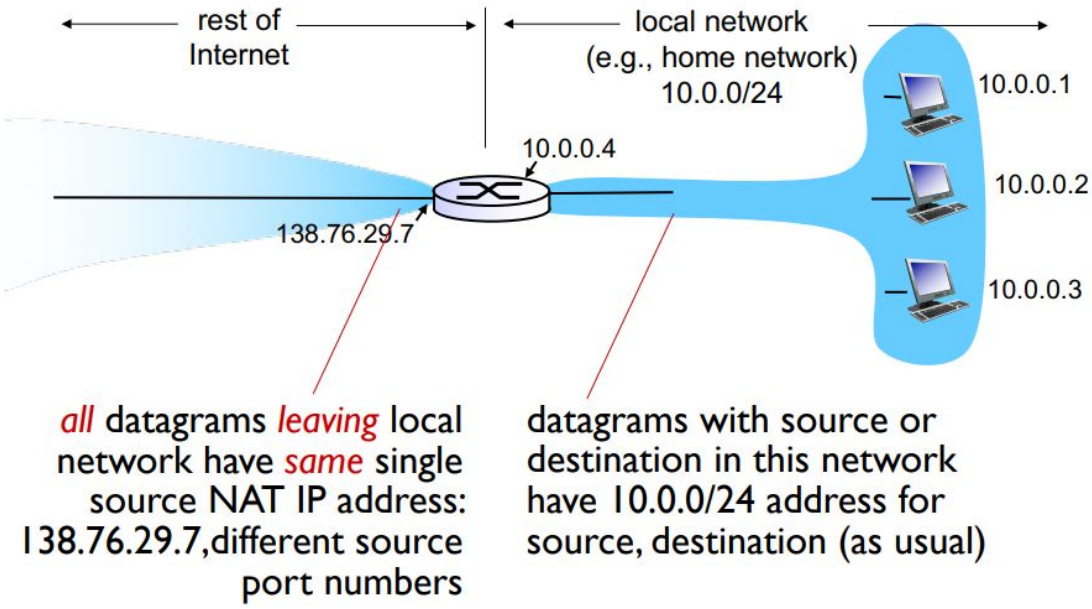


DHCP Provides mechanisms for **clients to renew the lease** on their IP Address.
DHCP uses **UDP** and port numbers 67 (server side) and 68 (client side)
Usually the **MAC address is used to identify clients**
DHCP server can be configured with a **"registered list" of acceptable MAC addresses**

Network Address Translation (NAT)

A **realm** with private addresses refers to a network whose addresses only have meaning to devices within that network

The NAT-enabled router *does not look like a router* to the outside world. Instead the NAT router behaves to the outside world **as a single device with a single IP address**



If all datagrams arriving at the NAT router from the WAN have the same destination IP address (specifically, that of the WAN-side interface of the NAT router), then how does the router know the internal host to which it should forward a given datagram? The trick is to use a NAT translation table at the NAT router, and to include port numbers as well as IP addresses in the table entries

All packets/datagrams arriving at the NAT router have the same destination IP address. To forward it to the correct host in the subnet we use a NAT translation table, and utilise port numbers.

An NAT Router must:
Replace the source IP Address and Port number of every **outgoing datagram** to the NAT IP address and new port number.Remote clients/servers will respond with these translated NAT values as the destination IP/Port Number.
Remember (in NAT translation table): every SourceIP/Portnumber to NatSourceIP/Portnumber translation pair.
Replace incoming datagrams: Replace the destination IP/Port numbers with the values stored in the NAT table.

This reduces the amount of IP's needed for each household/organisation.
Local changes can occur without notifying outside networks.
Outside changes can occur without affecting the local network.

Controversies:

- routers should only process up to layer 3
- violates end-to-end argument, certain applications are affecting by NAT, e.g peer to peer applications.
- address shortage should instead be solved by IPv6

Practical issues:

- NAT modifies port # and IP addresses, **Requiring recalculation of TCP and IP checksum**
- Some applications (DNS, FTP (PORT command), SIP, H.323) embed IP address or port numbers in their message payload (For legacy protocols, NAT must look into these packets and translate the embedded IP addresses/port numbers)
- If applications change port numbers periodically, the NAT must be aware of this

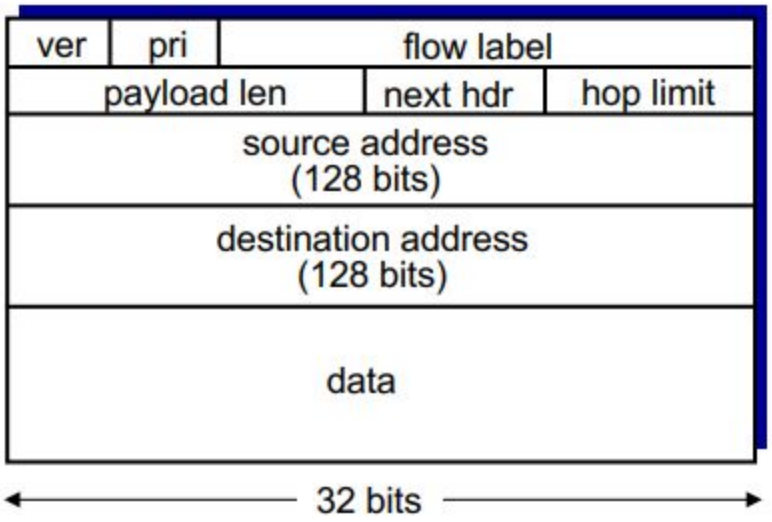
Problems and Solutions:
Problem: client wants to connect to server with address 10.0.0.1
Solution: Set up a static port to forward NAT IP's to the specific local IP.
Solution: automate static NAT port map configuration via Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. UPnP is the solution for many peer to peer tasks that involve NAT routers.
Solution (Used by Skype):

- NATed client establishes connection to relay
- external client connects to relay
- relay bridges packets between to connections

Despite the problems, NAT has been widely deployed
Most protocols can be successfully passed through a NAT, including VPN
Modern hardware can easily perform NAT functions at > 100 Mbps
IPv6 is still not widely deployed commercially, so the need for NAT is real
After years of refusing to work on NAT, the IETF has been developing "NAT control protocols" for hosts

IPv6
We are running out of 32 bit IP's. Can fix mistakes made in IPv4. (e.g disallowing fragmentation).

IPv6 is much more compact than IPv4 datagrams.
Priority: identify priority among datagrams in flow (traffic class) flow
Label: identify datagrams in same "flow." (concept of"flow" not well defined).
Next header: identify upper layer protocol for data



Changes:

- **Checksum:** removed entirely to reduce processing time at each hop
- **Options:** allowed, but outside of header, indicated by "Next Header" field
- **ICMPv6:** new version of ICMP
  Additional message types, e.g. "Packet Too Big"
  Multicast group management functions

Transitioning from Ipv4 to Ipv6:
IPv6 can be contained in the payload of IPv4. This is called **Tunneling.** A router can return the datagram to IPv6 once it's through the IPv4 routers.

Routing (Week 9)

Forwarding is considered the **Data plane**, Routing is the **Control plane.**

There are two ways to configure the Control pane:

- Per-router control (traditional, decentralized)
- Logically centralized control (software defined networking)

See earlier pages for a general look at these.

ICMP (self-study)

There are two levels of **Internet Routing**

- Each AS (Autonomous System/Domain) runs an **intra-domain** routing protocol that establishes routes within its domain
    - (AS -- region of network under a single administrative entity)
    - Link State, e.g., Open Shortest Path First (OSPF)
    - Distance Vector, e.g., Routing Information Protocol (RIP)
- ASes participate in an **inter-domain** routing protocol that establishes routes between domains
    - Path Vector, e.g., Border Gateway Protocol (BGP)

Learn some  **DIJKSTRA** 

Link State (Global)

- Routers maintain cost of each link in the network
- Connectivity/cost changes flooded to all routers
- Converges quickly (less inconsistency, looping, etc.)
- Limited network sizes

Distance Vector (Decentralised)

- Routers maintain next hop & cost of each destination.
- Connectivity/cost changes iteratively propagate form neighbour to neighbour
- Requires multiple rounds to converge
- Scales to large networks

Nodes maintain a Link State of all known links and costs.
When receiving a new Link State message, the router forwards it to all neighbours except the one that sent it the message.
This is called **LSA Flooding (Link State Advertisement)**
Routers keep a local copy so they don't forward previously seen LSA's.
Packet Loss and out of order packets cause challenges for Routing, however Acknowledgements, Retransmissions, Sequence numbers and TTL fields can guarantee success, much like TCP.

Eventually, **each node learns the entire network topology**, and can use Dijkstra's to compute the shortest paths between nodes.
This can be used to construct a forwarding table. For each node in the Topology, the router can have the shortest path pre-calculated.

Scaling problems

How many messages needed to flood link state messages?

O(N x E), where N is #nodes; E is #edges in graph

Processing complexity for Dijkstra's algorithm?

O(N2), because we check all nodes w not in S at each iteration and we have O(N) iterations There are more efficient implementations: O(N log(N) +E) using min-heap

How many entries in the LS topology database?

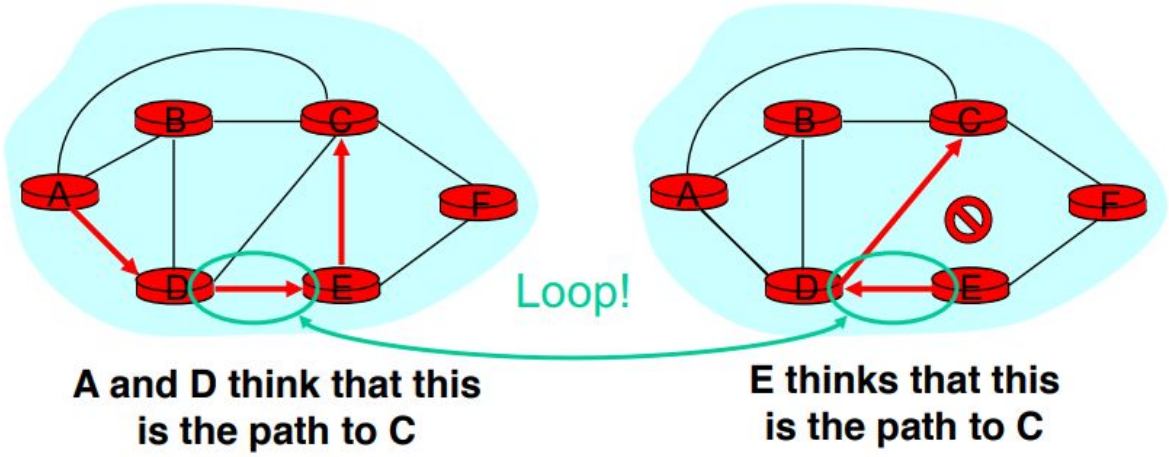
O(E)

How many entries in the forwarding table?

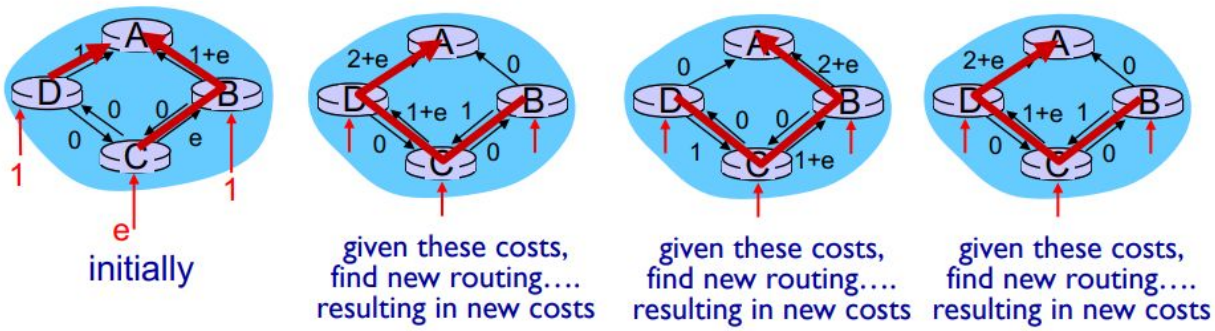
O(N)

Infinite Loop Problem

Inconsistent link-state database. Some routers know about failure before others, resulting in the shortest paths stored in routers being inconsistent. This can cause transient forwarding loops

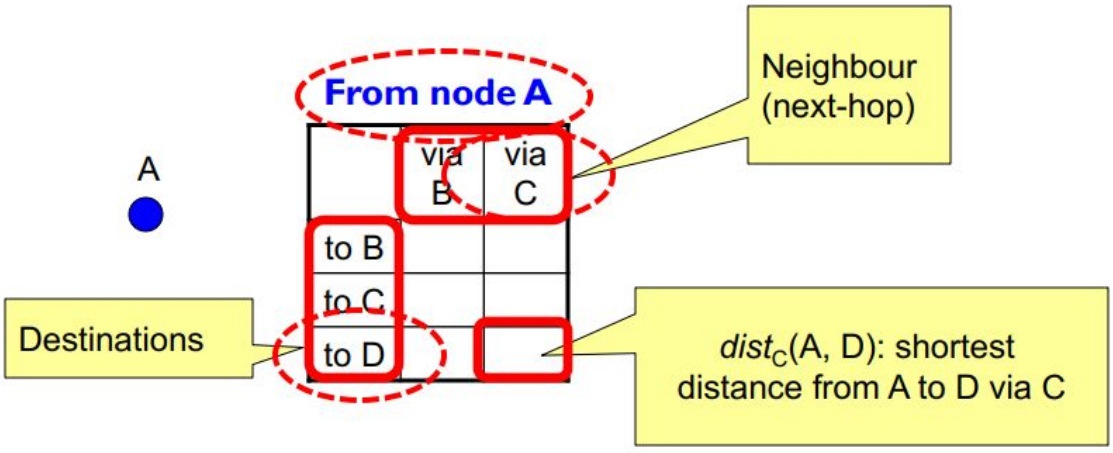


Oscillation problem (Best shown by picture)



Distance Vectors

Each router maintains its shortest distance to every destination via each of its neighbours, and
Each router computes its shortest distance to every destination via **any** of its neighbors



Each router initializes its dist() table based on its immediate neighbors and link costs
Each router sends its DV to its immediate neighbors
Routers process received DVs, And repeat…

Each router knows the links to its neighbors
Each router has provisional “shortest path” to every other router -- its distance vector (DV)
Routers exchange this DV with their neighbors
Routers look over the set of options offered by their neighbors and select the best one
Iterative process converges to set of shortest paths

iterative, asynchronous:

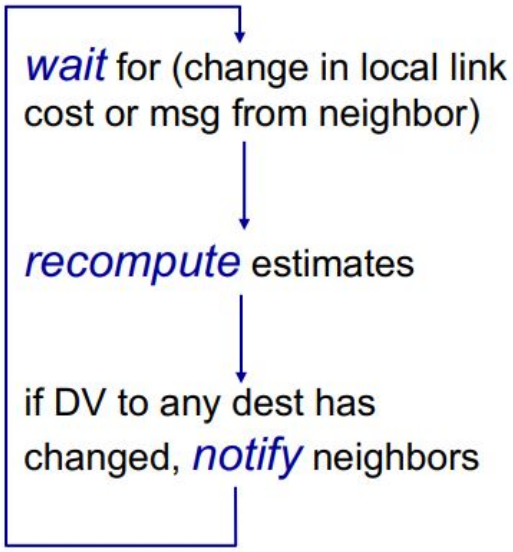
each local iteration caused by:

- local link cost change
- DV update message from neighbor

distributed:

- each node notifies neighbors *only* when its DV changes
  - neighbors then notify their neighbors if necessary

each node:



The Poisoned Reverse Rule stops counting to infinity.
Heuristic to avoid count-to-infinity
If B routes via C to get to A, B tells C it's (B's) distance to A is infinite (so C won't route to A via B)

ICMP

The most typical use of ICMP is for **error reporting.**
Lies just above IP.
Helps hosts and routers to communicate **network-layer information** to each other
ICMP messages have a **type and a code field**, and contain the header and the first 8 bytes of the IP datagram that caused the ICMP message to be generated in the first place
The Ping command sends an ICMP **type 8 code 0** message to the specified host. The destination host, seeing the echo request, sends back a **type 0 code 0** ICMP echo reply
Can be used to send "Quench" messages for congestion control, demanding a halt of transmission rate.
Trace route uses ICMP warning messages to retrieve the IP and Router name.

Chapter 5 - Link Layer

hosts and routers are called **nodes**
communication channels that connect adjacent nodes along communication path are called **links:**

- wired links
- wireless links
- LANs

layer-2 packet: **frame**, encapsulates datagram

data-link layer has responsibility of transferring datagram from one node to physically adjacent node over a link

datagram transferred by different link protocols over different links:

- e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- each link protocol provides different services, e.g., may or may not provide rdt over link

Services

Framing, link access:

- encapsulate datagram into frame, adding header, trailer
- channel access if shared medium
- "MAC" addresses used in frame headers to identify source, dest
    - different from IP address!

Reliable delivery between adjacent nodes

- we learned how to do this already (chapter 3)!
- seldom used on low bit-error link (fiber, some twisted pair)
- wireless links: high error rates

flow control:

- pacing between adjacent sending and receiving nodes

Error detection:

- Errors caused by signal attenuation, noise.
- Receiver detects presence of errors:
    - signals sender for retransmission or drops frame

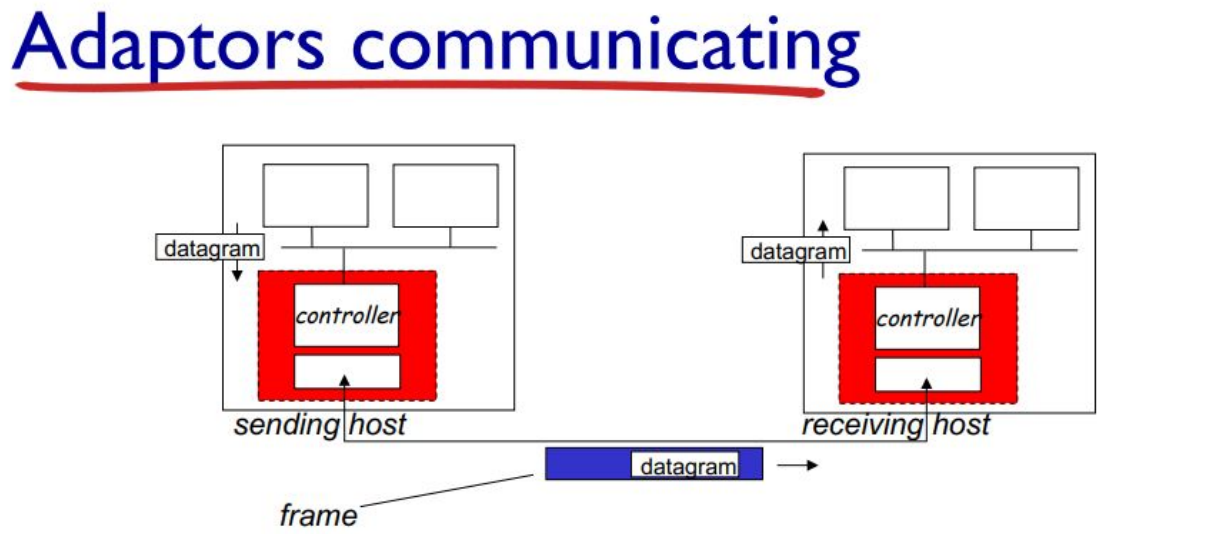
Error correction:

- receiver identifies **and corrects** bit error(s) without resorting to retransmission

Half-duplex and full-duplex

- with half duplex, nodes at both ends of link can transmit, but not at same time

The link layer is implemented in every host. It can be on a Network Interface Card or on a chip,



(Ethernet card, 802.11 card; Ethernet chipset). It's attached into the host systems busses.

EDC= Error Detection and Correction bits (redundancy)
D = Data protected by error checking, may include header fields • Error detection not 100% reliable! • protocol may miss some errors, but rarely • larger EDC field yields better detection and correction

Simple Parity
Suppose you want to send the message: – 001011011011000110010,
For every d bits (e.g., d = 7), add a **parity bit:**
– 1 if the number of one's is odd
– 0 if the number of one's is even
Note that the odd/even count is ***within the chunk defined by d***

If an odd number of bits get flipped, we'll detect it (can't do much to correct it).
Cost: One extra bit for every d

Two Dimensional Parity adds a parity **Byte.**
Can detect 1, 2, 3-bit (and some 4-bit) errors
Does a row check and column check, same rules as above.
Can correct 1 bit errors.

Cyclic Redundancy Checks
more powerful error-detection coding
view data bits, **D**, as a binary number
choose r+1 bit pattern (generator), **G**
goal: choose r CRC bits, **R**, such that

- exactly divisible by **G** (modulo 2)
- receiver knows **G**, divides by G. If **non-zero remainder: error detected!**
- can detect all burst errors less than r+1 bits

Widely used in practice (Ethernet, 802.11 WiFi, ATM)

Links
Point-to-point

- PPP for dial-up access
- point-to-point link between Ethernet switch, host

broadcast (shared wire or medium)

- old-fashioned Ethernet
- upstream HFC
- 802.11 wireless LAN

single shared broadcast channel
two or more simultaneous transmissions by nodes: interference

- collision if node receives two or more signals at the same time

multiple access protocol
distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
communication about channel sharing must use channel itself!

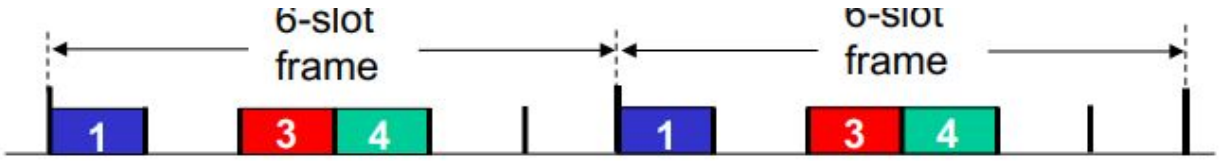
- no out-of-band channel for coordination

Ideally:
given: broadcast channel of rate R bps
desiderata:
1. when one node wants to transmit, it can send at rate R.
2. when M nodes want to transmit, each can send at average rate R/M
3. fully decentralized: • no special node to coordinate transmissions • no synchronization of clocks, slots
4. Simple

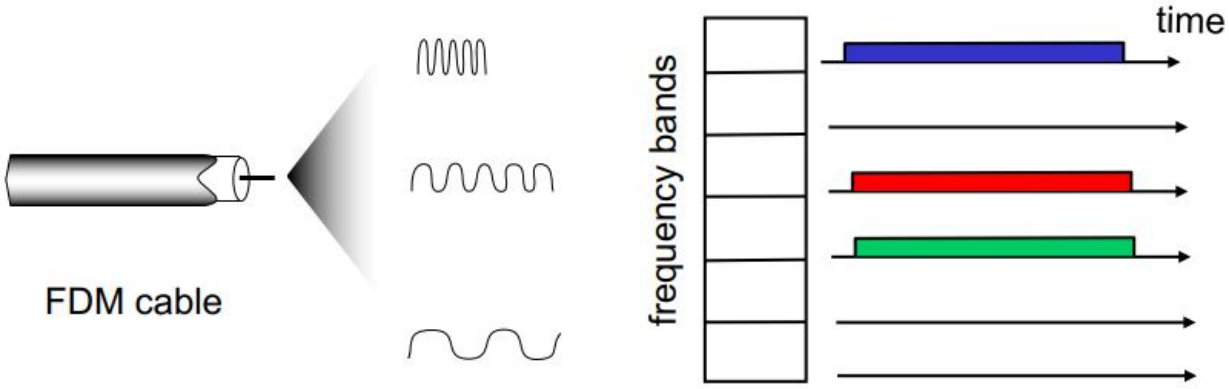
MAC Protocols
three broad classes:
Channel partitioning

- divide channel into smaller “pieces” (time slots, frequency, code)
- allocate piece to node for exclusive use

TDMA (Time Division Multiplexing Access)
access to channel in "rounds"
each station gets fixed length slot (length = pkt trans time) in each round
unused slots go idle
example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle



FDMA (Frequency Division Multiplexing Access)
channel spectrum divided into frequency bands
each station assigned fixed frequency band
unused transmission time in frequency bands go idle
example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



Random access

- channel not divided, allow collisions
- " recover " from collisions
- when node has packet to send
    - transmit at full channel data rate R.
    - no a priori coordination among nodes
- two or more transmitting nodes ➜ "collision" ,
- **random access MAC protocol specifies**:
    - how to detect collisions
    - how to recover from collisions (e.g., via delayed retransmissions)
- examples of random access MAC protocols:
    - slotted ALOHA
    - ALOHA
    - CSMA, CSMA/CD, CSMA/CA

"taking turns"

- nodes take turns, but nodes with more to send can take longer turns

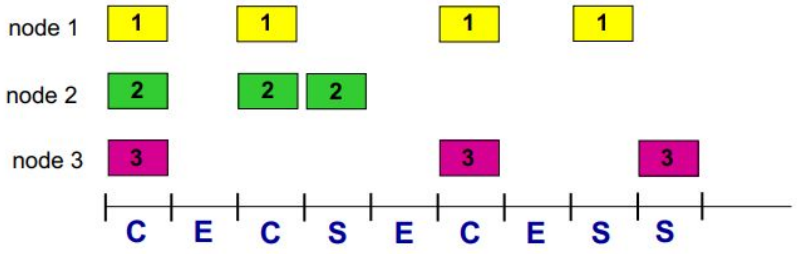
Slotted Aloha

Assumptions:

- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

Operation:

- when node obtains fresh frame, transmits in next slot
    - if no collision: node can send new frame in next slot
    - if collision: node retransmits frame in each subsequent slot with prob. p until success



Pros:

- single active node can continuously transmit at full rate of channel
- highly decentralized: only slots in nodes need to be in sync
- simple

Cons:

- collisions, wasting slots
- idle slots
- nodes may be able to detect collision in less than time to transmit packet
- clock synchronization

at best: channel used for useful transmissions 37% of time!

Pure Aloha, with no slots, and transmitting straight away has even worse efficiency.

CSMA (carrier sense multiple access)
if channel sensed idle:

- transmit entire frame

if channel sensed busy,

- defer transmission

This does not eliminate all collisions, due to nonzero propagation delay

A collision takes up a full slot and wastes time/efficiency.

CSMA/CD: CSMA ft. Collision Detection.
collisions *detected* within short time
colliding transmissions **aborted, reducing channel wastage**

collision detection

- **easy in wired LANs**: measure signal strengths, **compare transmitted, received signals**
- **difficult in wireless LANs**: received signal strength overwhelmed by local transmission strength
- For this to work, there need to be **restrictions** on **minimum frame size** and **maximum distance.**
- **Minimum frame size**: If the frame is too short and/or the distance between the nodes is too long, the transmission will undetectably collide.

CSMA Ethernet collision detection
1. NIC receives datagram from network layer, creates frame
2. If NIC senses channel idle, starts frame transmission. If NIC senses channel busy, waits until channel idle, then transmits.
3. If NIC transmits entire frame without detecting another transmission, NIC is done with frame !
4. If NIC detects another transmission while transmitting, aborts and sends jam signal
5. After aborting, NIC enters **binary (exponential) backoff**:
    - after mth collision, NIC chooses K at random from {0,1,2, …, (2^m)-1}. NIC waits K·512 bit times, returns to Step 2
    - longer backoff interval with more collisions

Taking Turns: MAC Protocols
channel partitioning MAC protocols:

- share channel efficiently and fairly at high load
- inefficient at low load: delay in channel access, 1/N bandwidth allocated even if only 1 active node!

random access MAC protocols

- efficient at low load: single node can fully utilize channel

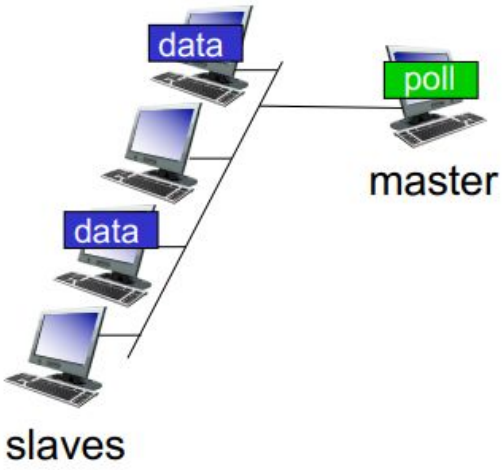
- high load: collision overhead

“taking turns” protocols look for best of both worlds!

Taking Turns: MAC

Polling:

- master node “invites” slave nodes to transmit in turn
- typically used with “dumb” slave devices
- Concerns:
    - Polling overhead
    - Latency
    - Single point of failure (master)



Token passing:

- control token passed from one node to next sequentially.
- token message
- Concerns:
    - token overhead
    - Latency
    - Single point of failure (token)

MAC addresses and ARP

32-bit IP address:

- network-layer address for interface
- used for layer 3 (network layer) forwarding

MAC (or LAN or physical or Ethernet) address:

- **function:** used ‘locally” to get frame from one interface to another physically-connected interface (same network, in IPaddressing sense)
- 48 bit MAC address (for most LANs) burned in NIC ROM, also sometimes software settable
- e.g.: 1A-2F-BB-76-09-AD

MAC address allocation administered by IEEE

manufacturer buys portion of MAC address space (to assure uniqueness)

Analogy:

- MAC address: like Social Security Number
- IP address: like postal address

MAC flat address ➜ portability

- can move LAN card from one LAN to another

IP hierarchical address not portable

- address depends on IP subnet to which node is attached

MAC addresses (used in link-layer)

- Hard-coded in read-only memory when adapter is built
- Like a social security number
- Flat name space of 48 bits (e.g., 00-0E-9B-6E-49-76)
- Portable, and can stay the same as the host moves
- Used to get packet between interfaces on same network

IP addresses

- Configured, or learned dynamically
- Like a postal mailing address
- Hierarchical name space of 32 bits (e.g., 12.178.66.9)
- Not portable, and depends on where the host is attached
- Used to get a packet to destination IP subnet

Layer	Examples	Structure	Configuration	Resolution Service
App. Layer	www.cse.unsw.edu.au	organizational hierarchy	~ manual	DNS
Network Layer	129.94.242.51	topological hierarchy	DHCP	ARP
Link layer	45-CC-4E-12-F0-97	vendor (flat)	hard-coded	

ARP Tables

A wants to send datagram to B

- B's MAC address not in A's ARP table.

A broadcasts ARP query packet, containing B's IP address

- dest MAC address = FF-FFFF-FF-FF-FF
- all nodes on LAN receive ARP query

B receives ARP packet, replies to A with its (B's) MAC address

- frame sent to A's MAC address (unicast)

A caches (saves) IP-toMAC address pair in its ARP table until information becomes old (times out)

- soft state: information that times out (goes away) unless refreshed

ARP is "plug-and-play" :

- nodes create their ARP tables without intervention from net administrator

See Lecture slides for Routing to another LAN. Key notes is that the Sending host uses the input routers MAC address as the MAC Address destination, which get's changed by the router when forwarding.

Security Issues:
Denial of Service - Hacker replies back to an ARP query for a router NIC with a fake MAC address
Man-in-the-middle attack - Hacker can insert his/her machine along the path between victim machine and gateway router
Such attacks are generally hard to launch as hacker needs physical access to the network

Solutions
Use Switched Ethernet with port security enabled (i.e. one host MAC address per switch port)
Adopt static ARP configuration for small size networks
Use ARP monitoring tools such as ARPWatch

ETHERNET

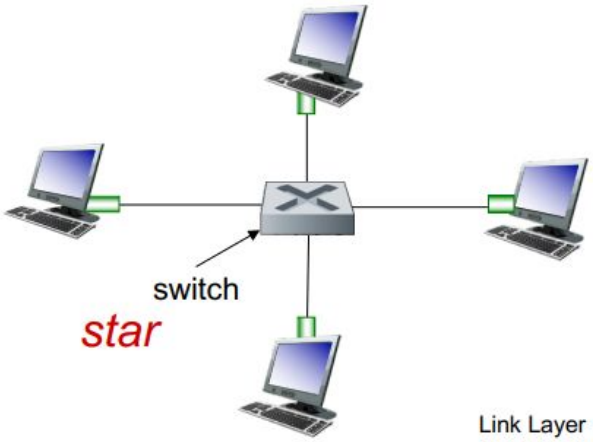
bus: popular through mid 90s

- all nodes in same collision domain (can collide with each other)
- CSMA/CD for media access control

star: prevails today

- active **switch** in center
- each “spoke” runs a (separate) Ethernet protocol (nodes do not collide with each other)
- No sharing, no CSMA/CD

bus: coaxial cable



STRUCTURE:

Preamble:

- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates

Addresses: 6 byte source, destination MAC addresses

- if adapter receives frame with matching destination address, or with broadcast address (e.g. ARP packet), it passes data in frame to network layer protocol
- otherwise, adapter discards frame

Type: indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)

CRC: cyclic redundancy check at receiver

- error detected: frame is dropped

Ethernet is **unreliable** and **connectionless**
Connectionless: no handshaking between sending and receiving NICs
Unreliable: receiving NIC doesnt send acks or nacks to sending NIC

- data in dropped frames recovered only if initial sender uses higher layer rdt (e.g., TCP), otherwise dropped data lost

Ethernet's MAC protocol: unslotted CSMA/CD with binary backoff

many different Ethernet standards:

- common MAC protocol and frame format
- **different speeds:** 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10Gbps, 40Gbps, 100Gbps,
- different physical layer media: **fiber, cable**

Ethernet SWITCH

Link-layer device: takes an ACTIVE role:

- store, forward Ethernet frames
- examine incoming frame's MAC address, **selectively** forward frame to one-or-more outgoing links when frame is to be forwarded on segment, uses CSMA/CD to access segment

Transparent: hosts are unaware of presence of switches
Plug-and-play, self-learning:switches do not need to be configured

Switches can maintain **multiple** simultaneous transmissions.
Hosts have dedicated, direct connection to switch
Switches buffer packets
Ethernet protocol used on each incoming link, but no collisions; full duplex

- each link is its own collision domain

Switching: A-to-A' and B-to-B' can transmit simultaneously, without collisions

Switch Tables:

- MAC address of host, interface to reach host, time stamp/TTL
- Looks like a routing table!

Self Learning Switches:
Switch learns which hosts can be reached through which interfaces

- when frame received, switch "learns" location of sender: incoming LAN segment
- records sender/location pair in switch table

Switch frame receive steps:
1. record incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. if entry found for destination then {
    if destination on segment from which frame arrived then:
        drop frame else forward frame on interface indicated by entry
    } else flood /* forward on all interfaces except arriving interface

Switches vs Routers

Both are store-and-forward:

- **Routers**: network-layer devices (examine networklayer headers)
- **Switches:** link-layer devices (examine link-layer headers)

Both have forwarding tables:

- **Routers**: Compute tables using routing algorithms, IP addresses
- **Switches**: Learn forwarding table using flooding, learning, MAC addresses

Security Issues

In a switched LAN once the switch table entries are established frames are not broadcast

- Sniffing frames is harder than pure broadcast LANs
- Note: attacker can still sniff broadcast frames and frames for which there are no entries (as they are broadcast)

Switch Poisoning: Attacker fills up switch table with bogus entries by sending large # of frames with bogus source MAC addresses

Since switch table is full, genuine packets frequently need to be broadcast as previous entries have been wiped out

Wireless Networks

A Base Station:

Typically connected to wired network

Relay - responsible for sending packets between wired network and wireless host(s) in its " area
e.g., cell towers, 802.11 access points

Ad-hoc mode (No infrastructure)

- nodes can only transmit to other nodes within link coverage
- nodes organize themselves into a network: route among themselves

	Single Hop	Multiple Hops
Infrastructure	host connects to base station (WiFi, WiMAX, cellular) which connects to larger Internet	host may have to relay through several wireless nodes to connect to larger Internet: mesh net
No Infra.	no base station, no connection to larger Internet (Bluetooth, ad hoc nets)	no base station, no connection to larger Internet. May have to relay to reach other a given wireless node MANET, VANET

Differences from a Wired Link
Decreased signal strength: radio signal attenuates as it propagates through matter (path loss)
Interference from other sources: standardized wireless network frequencies (e.g., 2.4 GHz) shared by other devices (e.g., phone); devices (motors) interfere as well
multipath propagation: radio signal reflects off objects ground, arriving ad destination at slightly different times

Free Space Path Loss d: distance : wavelength f: frequency c: speed of light

$$\mathrm{FSPL} = \left(\frac{4\pi d}{\lambda}\right)^2 = \left(\frac{4\pi df}{c}\right)^2$$

Signals bounce off surface and interfere (constructive or destructive) with one another v Self-interference

More Link Characteristics:
A High Signal to Noise ratio is good.
SNR versus BER tradeoffs
given physical layer:
increase power -> increase SNR- >decrease BER
given SNR:
choose physical layer that meets BER requirement, giving highest throughput
SNR may change with mobility: dynamically adapt physical layer (modulation technique, rate)
Hidden terminal problem/Signal Attenuation
B, A hear each other
B, C hear each other
A, C can not hear each other means A, C unaware of their interference at B
Carrier sense will be ineffective

Exposed Terminals:
Node B sends a packet to A; C hears this and decides not to send a packet to D (despite the fact that this will not cause interference) !! v Carrier sense would prevent a successful transmission

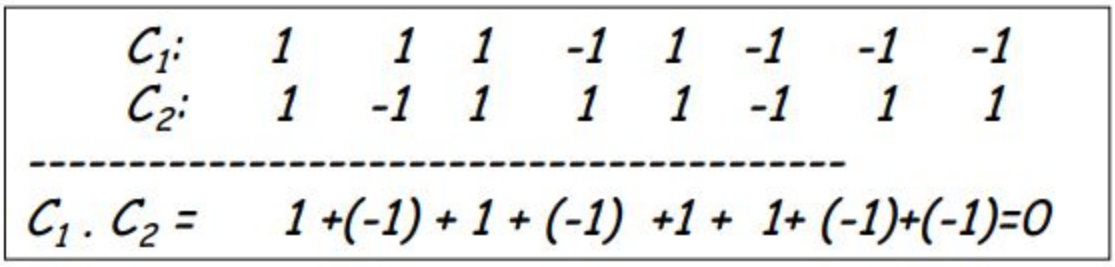
CDMA (Code Division Multiple Access)
Unique “code” assigned to each user; i.e., code set partitioning

- all users share same frequency, but each user has own “chipping” sequence (i.e., code) to encode data
- allows multiple users to “coexist” and transmit simultaneously with minimal interference (if codes are “orthogonal”)

Encoded Signal = (original data) X (chipping sequence)
Decoding: inner-product of encoded signal and chipping sequence

CDMA Encoding/Decoding
Encoded signal = (original data) modulated by (chipping sequence)
assume c(m) = 1 1 1 -1 1 -1 -1 -1
if data is 1, send **Original**
if data is -1 send **Flip the bits, 1 to -1 and -1 to 1.**

Decoding:
The product and sum of the codes should equal 0.



If there are **multiple** CDMA codes all of the codes have to be **orthogonal** to each other.
E.g: 3 codes: C1, C2 and C3. Then C1 x C2 = 0, C2 x C3 = 0 and C1 x C3 = 0

802.11 LAN Structure (Wifi)

wireless host communicates with base station
base station = access point (AP)
Basic Service Set (BSS) (aka “cell”) in infrastructure mode contains:

- wireless hosts
- access point (AP): base station
- ad hoc mode: hosts only

802.11 uses Multiple Access philosophy:
Avoids collisions: 2+ nodes transmitting at same time
802.11: CSMA - sense before transmitting

- don't collide with ongoing transmission by other node

802.11: no collision detection!

- Difficult to receive (sense collisions) when transmitting due to weak received signals (fading)
- Cannot sense all collisions in any case: hidden terminal, fading problems
- **Goal:** avoid collisions: CSMA/C(ollision)A(voidance)

Key Points:
No concept of a global collision

- Different receivers hear different signals
- Different senders reach different receivers

Collisions are at receiver, not sender

- Only care if receiver can hear the sender clearly
- It does not matter if sender can hear someone else, As long as that signal does not interfere with receiver

Goal of protocol

- Detect if receiver can hear sender
- Tell senders who might interfere with receiver to shut up

MAC Protocol: CSMA/CA
802.11 sender
1 if sense channel idle for DIFS then transmit entire frame (no CD)
2 if sense channel busy then start random backoff time timer and counts down this value when the channel is sensed idle. While the channel is sensed busy, the counter value remains frozen.
Transmit when timer expires if no ACK, increase random backoff interval, repeat 2
802.11 receiver
- if frame received OK
return ACK after SIFS (ACK needed due to hidden terminal problem)

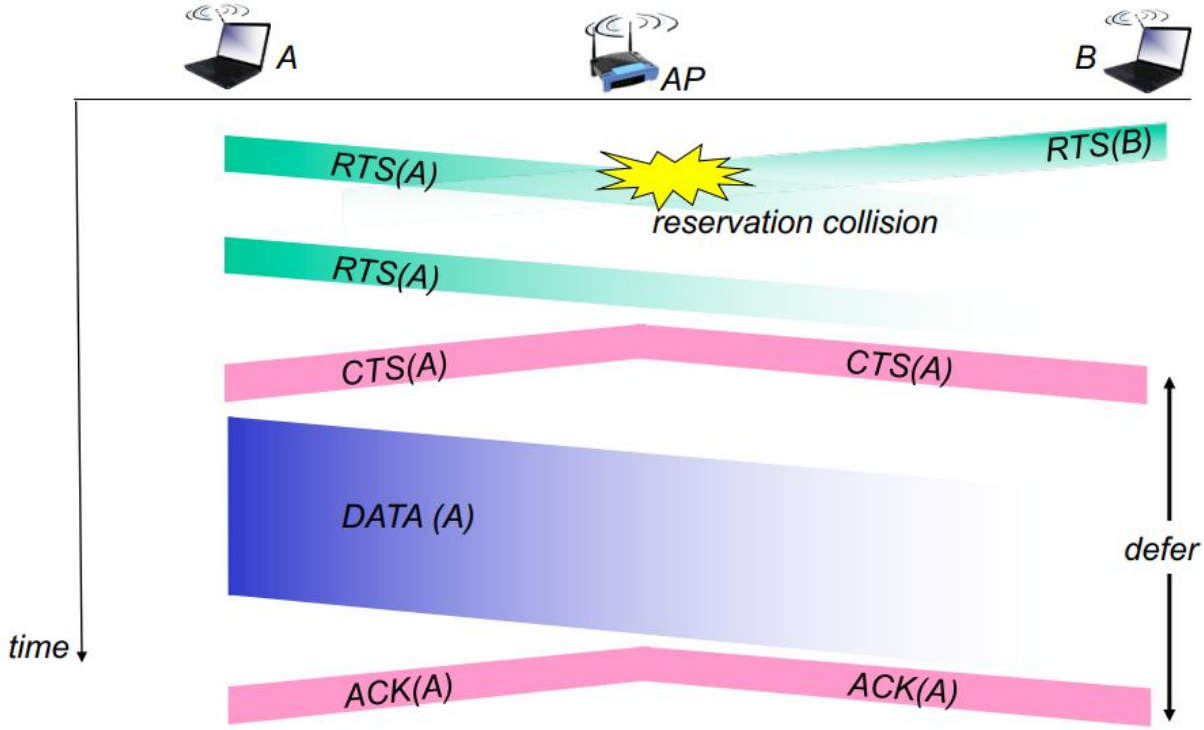
Avoiding Collisions
Allow sender to " reserve " channel rather than random access of data frames: **avoid collisions of long data frames**
Sender first transmits small request-to-send (RTS) packets to BS using CSMA

- RTSs may still collide with each other (but they're short)

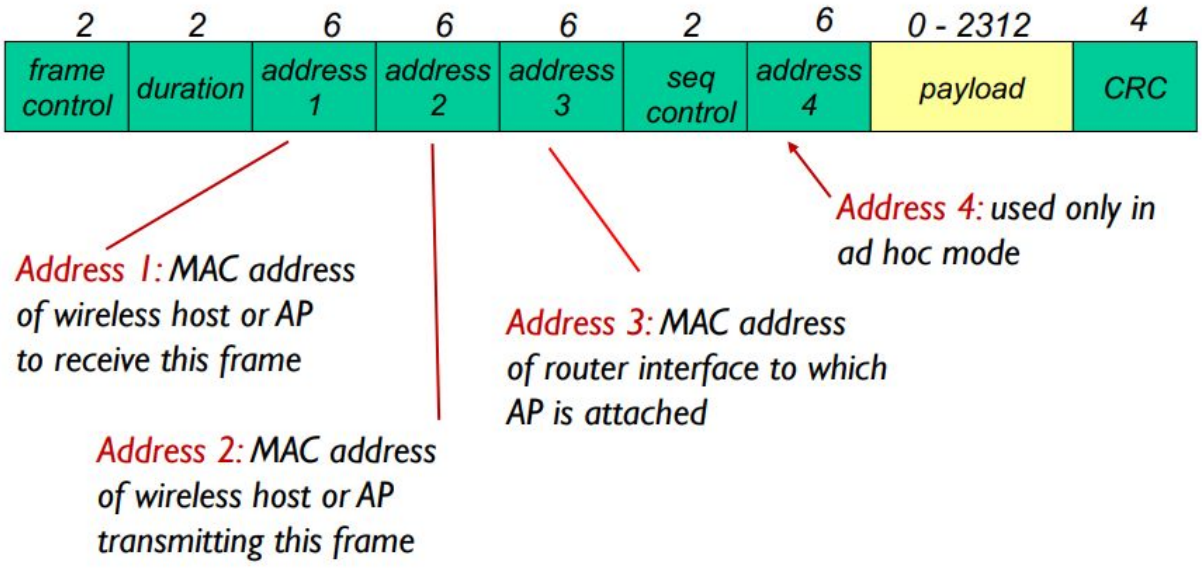
BS broadcasts clear-to-send CTS in response to RTS
CTS heard by all nodes

- sender transmits data frame
- other stations defer transmissions

The only collisions that occur are collisions of **small reservation packets.**



Frame Addressing



802.11: advanced capabilities

Rate Adaption:

- base station, mobile dynamically change transmission rate (physical layer modulation technique) as mobile moves, Signal Noise Ratio (SNR) varies

Power management

Node-To-AP: “I am going to sleep until next beacon frame”

- AP knows not to transmit frames to this node
- Node wakes up before next beacon frame

Beacon frame: contains list of mobiles with AP-to-mobile frames waiting to be sent

- node will stay awake if AP-to-mobile frames to be sent; otherwise sleep again until next beacon frame

802.15: personal area network

Less than 10 m diameter

Replacement for cables (mouse, keyboard, headphones)

Ad hoc: no infrastructure

Master/slaves:

- slaves request permission to send (to master)
- master grants requests

802.15: evolved from Bluetooth specification

- 2.4-2.5 GHz radio band
- Up to 721 kbps

Chapter 8 - Security

Confidentiality: only sender, intended receiver should “understand” message contents
Message authentication: sender, receiver want to confirm identity of each other message
Integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection
Access and Availability: services must be accessible and available to users

Security concerns deal a lot with people in the middle, who might read or intercept messages. These people could:

Eavesdrop: intercept messages
Actively insert messages into connection
Impersonation: can fake (spoof) source address in packet (or any field in packet)
Hijacking: “take over” ongoing connection by removing sender or receiver, inserting himself in place
Denial of service: prevent service from being used by others (e.g., by overloading resources)

Encryption makes data unreadable to middlemen without the Decryption key.

Symmetric Key Cryptography Means the Encryption and Decryption key are the same.

Some other simple Encryption, substitution. Characters are directly substituted by another character, the Decryption key being the alphabet remapping.

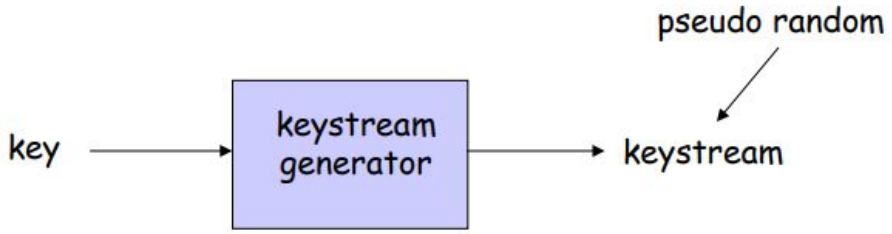
If a middleman analyses **Cipher-Text,** They can analyse it. They could Brute force attempt all keys to decrypt the data and perform statistical analysis upon it.

If the middleman retrieves plaintext that **directly corresponds to Cipher-text,** They can begin some decryption and possibly figure out the key.

Thus we need **Sophisticated Encryption.**

Stream ciphers encrypt one bit at time
Block ciphers Break plaintext message in equal-size blocks, and Encrypt each block as a unit

Stream Ciphers:



- Combine each bit of keystream with bit of plaintext to get bit of ciphertext
- m(i) = ith bit of message
- ks(i) = ith bit of keystream
- c(i) = ith bit of ciphertext
- c(i) = ks(i) ⊕ m(i) (⊕ = exclusive or)
- m(i) = ks(i) ⊕ c(i)

RC4 is a popular stream cipher

- Extensively analyzed and considered good
- Key can be from 1 to 256 bytes
- Used in WEP for 802.11
- Can be used in SSL

Block Ciphers

Ciphertext processed as k bit blocks
1-to-1 mapping is used to map k-bit block of plaintext to k-bit block of ciphertext
E.g: k=3 (see table) § 010110001111 => 101000111001
Possible permutations = 8! (40,320)
To prevent brute force attacks

- Choose large K (64, 128, etc)

Full-block ciphers not scalable

- E.g., for k = 64, a table with 264 entries required
- instead use function that simulates a randomly permuted table

Block CIphers scramble data in their second Round. More rounds can be done, but this reduces efficiency.

Symmetric key crypto: DES

DES: **Data Encryption Standard**
US encryption standard [NIST 1993]
56-bit symmetric key, 64-bit plaintext input
Block cipher with cipher block chaining
how secure is DES?

- DES Challenge: 56-bit-key-encrypted phrase decrypted (brute force) in less than a day
- no known good analytic attack

Making DES more secure: 3DES: encrypt 3 times with 3 different keys

AES, Advanced Encryption Standard, is much stronger. Would take 149 trillion years to brute force it.

In a normal Cipher block, repeated inputs are given repeated outputs. (Creating a pattern).

Cipher block chaining: XOR ith input block, m(i), with previous block of ciphertext, c(i-1). Uses the constantly being generated ciphertext in order to create the next block of ciphertext.

How do we encrypt first block?

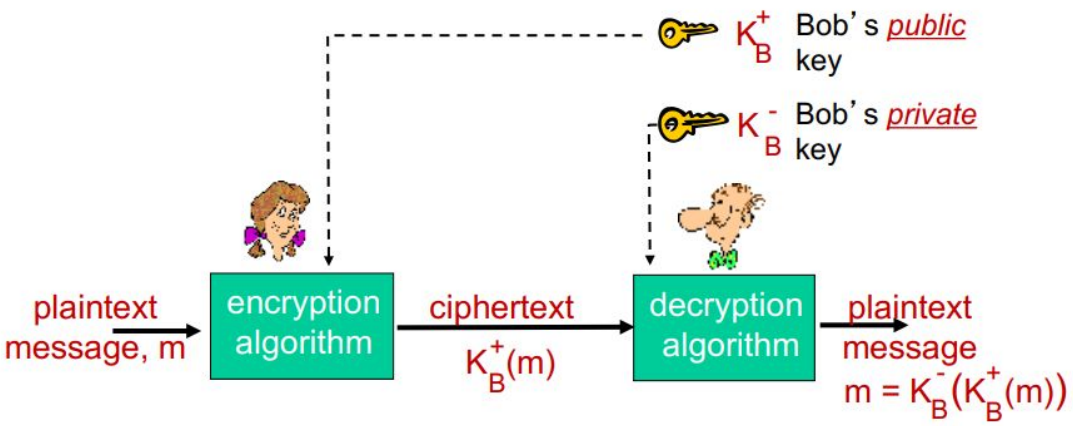
- Initialization vector (IV): random block = c(0)
- IV does not have to be secret

Change IV for each message (or session)

- Guarantees that even if the same message is sent repeatedly, the ciphertext will be completely different each time

Symmetric key Crypto requires the sender and receiver to agree on a secret key, but they may have never met before.

Public key crypto is very different but solves this problem.
Public encryption key **known to all**
Private decryption key known **only to receiver**



Requirements:

1. need $K_B^+(\cdot)$ and $K_B^-(\cdot)$ such that
   $K_B^-(K_B^+(m)) = m$
2. given public key K_B^+, it should be impossible to compute private key K_B^-

RSA: Rivest, Shamir, Adelson algorithm

RSA:

Encryption example,

Say we have m = 10010001 . This message is uniquely represented by the decimal number 145.

To encrypt m, we encrypt the corresponding number, which gives a new number (the ciphertext).

THEN **(Creating Public/Private Key)**

1. choose two large prime numbers **p, q.**
(e.g., 1024 bits each)
2. compute **n = pq, z = (p-1)(q-1)**
3. choose **e** (with **e<n**) that has no common factors
with **z** (**e, z** are “relatively prime”).
4. choose **d** such that **e*d-1** is exactly divisible by **z.**
(in other words: **e*d** mod **z** = 1).

5. *public* key is $\underbrace{(n,e)}_{K_B^+}$. *private* key is $\underbrace{(n,d)}_{K_B^-}$.

Encryption/Decryption:

0. given (n,e) and (n,d) as computed above

1. to encrypt message m $(<n)$, compute
   $c = m^e \bmod n$

2. to decrypt received bit pattern, c, compute
   $m = c^d \bmod n$

magic happens! $m = \underbrace{(m^e \bmod n)}_{c}{}^d \bmod n$

Why does it work?

Must show that $c^d \bmod n = m$ where $c = m^e \bmod n$
fact: for any x and y: $x^y \bmod n = x^{(y \bmod z)} \bmod n$

- where n= pq and z = (p-1)(q-1)

thus,

$c^d \bmod n = (m^e \bmod n)^d \bmod n$
$= m^{ed} \bmod n$
$= m^{(ed \bmod z)} \bmod n$
$= m^1 \bmod n$
$= m$

The following property will be *very* useful later:

$$\underbrace{K_B^-(K_B^+(m))}_{\text{use public key first, followed by private key}} = m = \underbrace{K_B^+(K_B^-(m))}_{\text{use private key first, followed by public key}}$$

result is the same!

RSA is 100 times more intensive to compute than DES.
RSA is often used to secure a connection, which is then used to communicate the creation of a symmetric session key.

Authentication

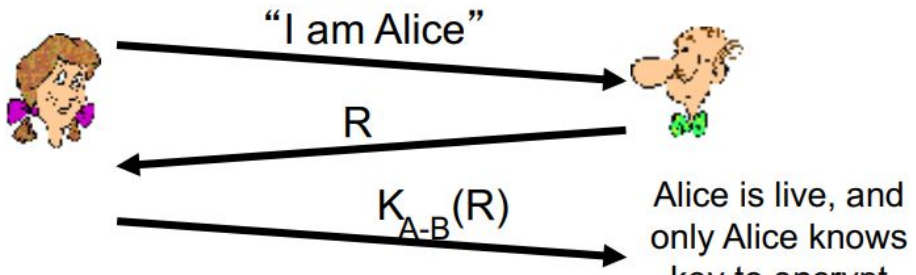
Impersonation is a big issue, anyone can claim to be anyone without authentication.

Using a password for authentication is vulnerable to **playback attack:** Trudy records Alice's packet and later plays it back to Bob. An encrypted password is still vulnerable to this.

Goal: avoid playback attack

nonce: number (R) used only *once-in-a-lifetime*

ap4.0: to prove Alice "live", Bob sends Alice *nonce*, R. Alice must return R, encrypted with shared secret key

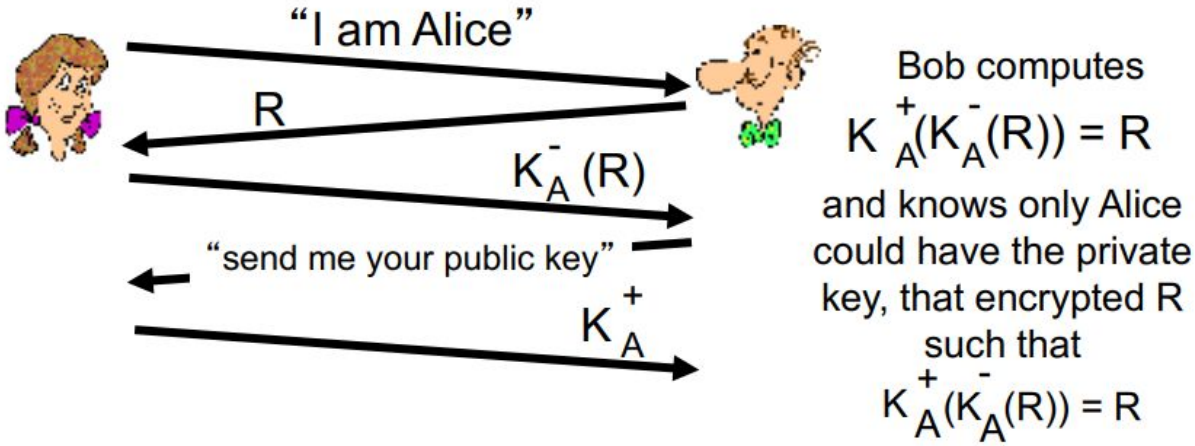


Failures, drawbacks?

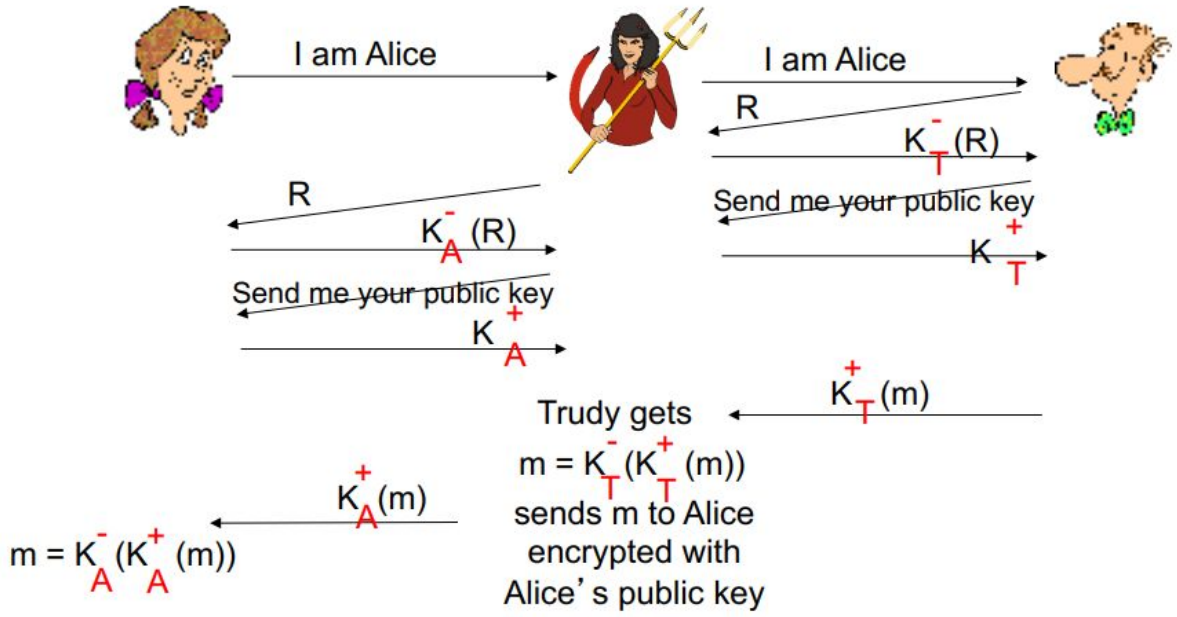
ap4.0 requires shared symmetric key

- can we authenticate using public key techniques?

ap5.0: use nonce, public key cryptography



man (or woman) in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)



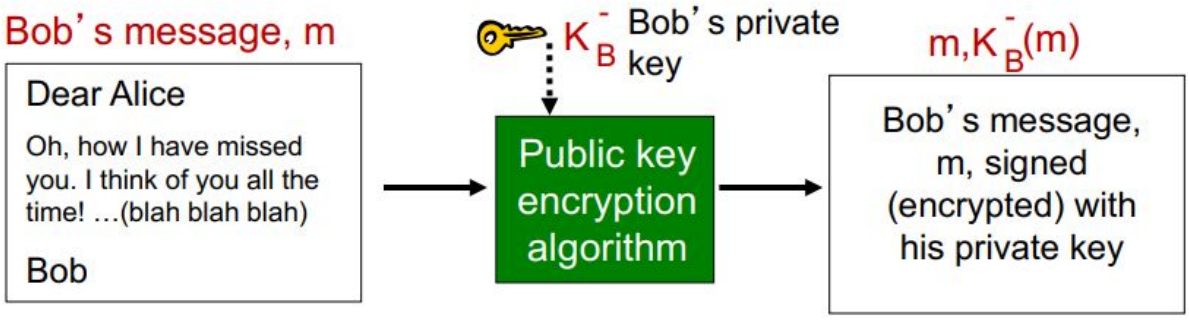
Man in the middle is difficult to detect.

Digital Signatures

Digital Signatures are **verifiable, unforgettable:** recipient (Alice) can prove to someone that Bob, and no one else (including Alice), must have signed document

simple digital signature for message m:

- Bob signs m by encrypting with his private key K_B^-, creating "signed" message, $K_B^-(m)$



- suppose Alice receives msg m, with signature: m, $K_B^-(m)$
- Alice verifies m signed by Bob by applying Bob's public key K_B^+ to $K_B^-(m)$ then checks $K_B^+(K_B^-(m)) = m$.
- If $K_B^+(K_B^-(m)) = m$, whoever signed m must have used Bob's private key.

Alice thus verifies that:

- Bob signed m
- no one else signed m
- Bob signed m and not m'

Non-Repudiation: Alice can take m, and signature $K_B^-(m)$ to court and prove that Bob signed m

Message Digests

Public key encryption of long messages is too expensive.

A digital '**Fingerprint**' is of a **fixed length.**

Hash Functions:
Many-to-1
Produces fixed-size message digest (fingerprint)
Given message digest x, computationally infeasible to find m such that x = H(m)

Internet checksums are only 16 bit, aren't big enough and have too many duplicate inputs that end up with the same hash.

MD5 Hash function is widely used, and computes **128 bit hashes.**
SHA-1 is also used, 160-bit.

Certification Authorities bind keys to entities.
A person or router can register it's key to the CA.
Bob's public key is now protected by the CA's private key.
If 'Alice' wants it, she requests it from the CA using Bob's certificate.

- Suppose Bob wants to send Alice a digital signature for the message *m*. To create the digital signature
  - A) Bob applies a hash function to *m* and encrypts the result with his private key
  - B) Bob applies a hash function to *m* and encrypts the result with Alice's public key
  - C) Bob encrypts *m* with his private key and then applies a hash function to the result
  - D) Bob applies a hash function to *m* and encrypts the result with his public key
- Suppose a CA creates Bob's certificate, which binds Bob's public key to Bob. This certificate is signed with
  - A) Bob's private key
  - B) Bob's public key
  - C) The CA's private key
  - D) The CA's public key