CS2105 cheatsheet

Commands

Ipconfig/4 – check network configuration of your local machine – DHCP enabled, private ip address, subnet mask

Ping – test network by sending dummy packets and viewing packet response (no. of bytes in packet, RTT (under time field), TTL

Dig – used to query DNS servers. Can be used to find out query time etc.

telnet – connect to a remote computer over network

nslookup/host – name server lookup. Similar to dig

General Formula to send P packets of length L across N links at R transmission rate:

(P + N – 1)(L/R)

The Network Core

- Mesh of interconnected routers
- Data transmitted through

→ Circuit switching

→ Packet switching

Circuit Switching

End-end resources allocated to and reserved for 'call' between source & dest:

- call setup required
- circuit-like (guaranteed) performance
- circuit segment idle if not used by call

Packet Switching

Host sending function:

- breaks app message into smaller chunks (packets), of length L bits
- transmits packets at transmission rate R → link *bandwidth*
- *Store and forward* – packets are passed from one router to next. Entire packet must arrive at router before it can be transmitted onto next link.

→ Internet is a packet switching network

Packet Delays

- Processing Delay, queueing delay, transmission delay, propagation delay.

Application Layer

- Client-Server Architecture

→ Server waits for incoming requests, provides requested service to clients

→ Clients initiates contact with server, requests service from server

- P2P architecture

→ Arbitrary end systems directly communicate

→Requests service from peers and in turn provides service to other peers

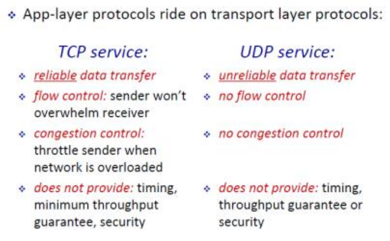
Transport service Application layer might need:

- Data Integrity, Throughput, Timing, Security

What is defined in App Layer Protocol?

- Types of message exchanged
- Message syntax
- Message semantics
- rules for when and how apps send and respond to messages

UDP vs TCP



❖ App-layer protocols ride on transport layer protocols:

TCP service:	UDP service:
❖ *reliable data transfer*	❖ *unreliable data transfer*
❖ *flow control:* sender won't overwhelm receiver	❖ *no flow control*
❖ *congestion control:* throttle sender when network is overloaded	❖ *no congestion control*
❖ *does not provide:* timing, minimum throughput guarantee, security	❖ *does not provide:* timing, throughput guarantee or security

Http: Run over TCP.

Non-persistent vs Persistent HTTP: one object send over one TCP connection vs multiple objects sent over one TCP connection

→ Persistent HTTP + Pipelining

DNS

A host is identified by:

- Hostname

- IP Addr
DNS translates between the two.

RR – Resource Records
- Mapping between host names and IP addresses are stored as RRs.
RR Format: (name, value, type, ttl)
- Type = A: name: hostname, value: IP Addr
- Type = CNAME: name: alias name for some 'canonical' (real) name, value: canonical name
- Type = NS: name: domain, value: hostname of authoritative name server for domain
- Type = MX, name: domain, value: name of mail server

Distributed, Hierarchical Database
-DNS stored RR in distributed DBs implemented in hierarchy of many name servers.
-DNS server listen to UDP port 53
Root DNS → Top Level Domain (TLD) Servers → Authoritative Servers

Local DNS Server
Each ISP has one local DNS Server, or default name server
- Caches DNS mapping, expires after some time (TTL)
- Runs over UDP
- iterative vs recursive query

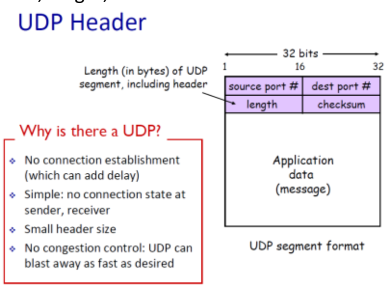
Multiplexing: Assembling segments from processes and adding headers and passing to network layer
De-Multiplexing: Delivering segments to correct socket

Connection-Oriented vs *Connectionless* Multiplexing: TCP vs UDP

Network Congestion
- Affects queueing delay at each router node.

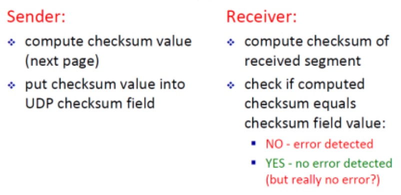
__Transport Layer__
UDP Services:
- Multiplexing at sender: UDP gathers data from processes, forms packets and passes them to IP
- De-multiplexing at receiver: UDP receives packets from lower layer and dispatches them to right processes
- Calculate checksum of packet's body
- Unreliable transmission
De-multiplexing:
- Checks destination port # in segment
- Directs UDP segment to the socket with that port #
- Datagrams with same port # will be directed to same UDP socket at destination

Header
Size – 64 bits
Port, length, checksum – 16 bits



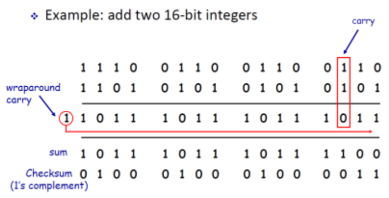
Why is there a UDP?
- No connection establishment (which can add delay)
- Simple: no connection state at sender, receiver
- Small header size
- No congestion control: UDP can blast away as fast as desired

Checksum
- detect 'errors' (i.e. flipped bits) in transmitted segment


Sender:
- compute checksum value (next page)
- put checksum value into UDP checksum field

Receiver:
- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - NO - error detected
  - YES - no error detected (but really no error?)

Calculating Checksum


Example: add two 16-bit integers
carry
1 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0
1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
wraparound carry
1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1
sum 1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 0
Checksum (1's complement) 0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1

If 3 or more 16-bit integers:

Apply binary addition for every integer first, before adding MSB to result.

Reliable Transfer over Unreliable Channel

Underlying network may:

Corrupt, drop and reorder packets, or deliver packets after arbitrarily long delay

rdt 1.0

Assume underlying channel is perfectly reliable.

Sender sends data into perfect channel

Recevr reads data from perfect channel

rdt 2.0

Channel with Bit Errors.

Underlying channel may flip bits.

- **Detect** bit errors using checksum.
- **Recover** from bit errors by using *ACKs* and *NAKs*.

Fatal Flaw:

If ACK/NAK is corrupted, sender does not know what happened at receiver

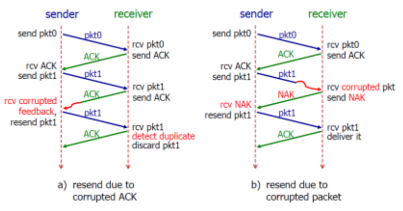
If sender simply resends packet if ACK/NAK is garbled, what if packet was originally successfully received by receiver? How does recver know that it's a duplicate packet?

rdt 2.1: rdt 2.0 + packet sequence

Handle duplicates:

Sender retransmits current packet if ACK/NAK is garbled, add sequence number to each packet, receiver discards duplicate packet.

rdt 2.1 In Action



rdt 2.2: NAK free protocol

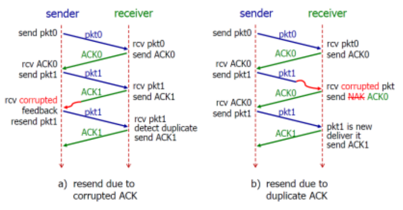
Use ACKs only. Instead of sending NAK, receiver sends ACK for last packet received OK.

→ Receiver explicitly includes seq # of packet being ACKed.

Duplicate ACKs at sender → retransmit current pkt

→ So basically sender resends packet if *it receives garbled ACK,* OR *it receives duplicate ACK number.*

rdt 2.2 In Action



rdt 3.0: Channel with Errors and Loss

Assumption: underlying channel

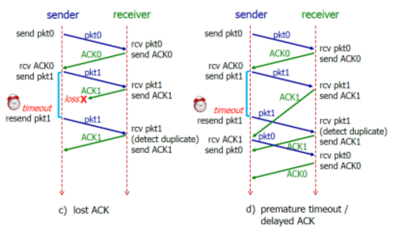
- May flip bits in packets
- May lose packets
- May incur arbitrarily long packet delay
- Won't reorder packets

To handle packet loss:

Sender waits 'reasonable' amount of time for ACK. Resends packet if no ACK received till timeout.

For corrupted packets, sender does nothing. Only uses timeout to resend packet.

rdt 3.0 In Action



However, based on this model, performance is extremely bad.

→ Pipelined Protocols!

GBN – Go-Back-N

Key Features

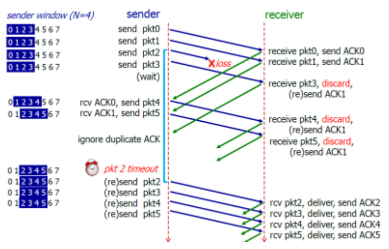
Sender

- up to N unACKed packets in the pipeline.
- seq # in packet header
- ‘sliding window’ to keep track of unACKed packets
- Timer for oldest unACKed packet.
- timeout(n): retransmit packet n and all subsequent packets in the window

Receiver

- Only ACK packets that arrive in order.
- discard out of order packets and ACK the last in-order packet.

Go-back-N In Action



Selective Repeat

Key Features

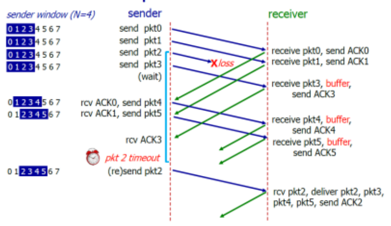
- Individually acknowledges all correctly received packets.

→ *Buffers* out of order packets, for eventual in-order delivery to upper layer

- Sender maintains timer for **each** unACKed packet

→ Only retransmit that unACKed packet when timer expires.

Selective Repeat In Action



<u>TCP</u>

- Welcome socket (port 80)
- Point to point (one sender, one receiver)
- Sequence No of packet = sequence number of first byte.
- Connection-oriented

→ *handshaking* (exchange of control messages) before sending app data

- Full Duplex service (bi-directional data flow in the same connection)
- Reliable, in-order byte stream (seq # is label bytes)

→ header size minimum 20 bytes or 160 bits, to 60 bytes or 480 bits.

srcIPAdd, srcPort, destIPAdd, destPort used to direct a segment to an appropriate socket.

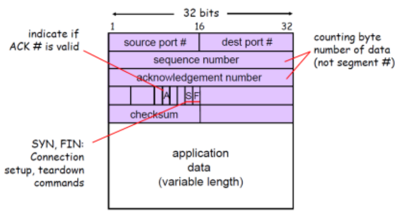
- Port number is 16 bits, or 2 bytes
- Can work correctly in a network that may reorder packets
- Bi-directional, hosts individually choose their own ISNs

- TCP send and receive buffers.

→ 2 buffers created after handshaking at any side

- App layer data carried – Maximum segment size (MSS), usually 1460 **bytes**. **Does not include header size.**
- app passes data to TCP and TCP forms packets in view of MSS.

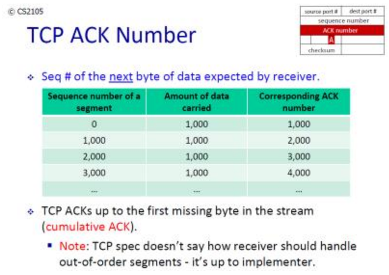
TCP Header



ACK: packet to indicate that this is in reply to another packet we received, and that contained data. 1 or 0

Feedback packet (packet with ACK 1) may have data from receiver to sender. This is known as *piggybacking*, i.e. combine ACK packet + data packet into 1 packet.

TCP ACK number



© CS2105

TCP ACK Number

- Seq # of the next byte of data expected by receiver.

Sequence number of a segment	Amount of data carried	Corresponding ACK number
0	1,000	1,000
1,000	1,000	2,000
2,000	1,000	3,000
3,000	1,000	4,000
…	…	…

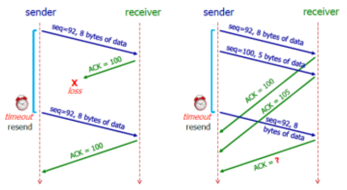
- TCP ACKs up to the first missing byte in the stream (cumulative ACK).
  - Note: TCP spec doesn't say how receiver should handle out-of-order segments - it's up to implementer.

TCP ACK Generation

- If in-order segment with expected seq # received, receiver will wait up to 500ms for next segment. If no segment received, send ACK.
- If 2nd segment arrives, send one cumulative ACK to ACK both segments
- If out-of-order segment is received, receiver immediately sends duplicate ACK.
- If segment that fills gap arrives, receiver will send updated ACK if it fills
lower gap.

If last segment received by receiver has sequence number 10, then the ACK sent is 11.

TCP Timeout/Retransmission



a) Lost ACK b) premature timeout

- TCP only has 1 timer: timer on the **oldest unACKed packet. Will only retransmit one packet.**

TCP Timeout Value

- Too short timeout: premature timeout, unnecessary retransmissions.
- Too long timeout: slow reaction to segment loss.

→ Timeout value must be higher than RTT. However, RTT always changes.

→ TCP computes and constantly updates timeout interval based on estimated RTT.

$$\texttt{EstimatedRTT} = (1-\alpha)*\texttt{EstimatedRTT} + \alpha*\texttt{SampleRTT}$$

(typical value of α : 0.125)

$$\texttt{DevRTT} = (1-\beta)*\texttt{DevRTT} + \beta*|\texttt{SampleRTT-EstimatedRTT}|$$

(typical value of β : 0.25)

$$\texttt{TimeoutInterval} = \texttt{EstimatedRTT} + \underbrace{4*\texttt{DevRTT}}_{\text{"safety margin"}}$$

Fast Retransmission

If sender receives 4 ACKs for the same segment, it assumes that segment is lost and resends the segment immediately.

Establishing and Closing connection

Establishing connection – handshaking.

→ 3-way handshake. In the 3rd packet, sender can send app data. In the 2nd packet, receiver sends initial seq number.

→ Client sends SYNbit = 1 to indicate control packet. Server replies with SYNbit = 1 as well when ACKing SYN

→ 3rd packet onwards, i.e. client to server, client sends ACK packet for previous control packets + data packet piggybacking

- Initial Seq Number (ISN) – unique 32-bit sequence number assigned to each new conncetion on TCP
- Designed to randomly select a seq number for the first byte of data transmitted in a new TCP connection

Close connection - FINbit

→sender sends FINbit = 1. Server sends ACK. After closing connection, server sends FINbit = 1 to client. Client sends ACK.

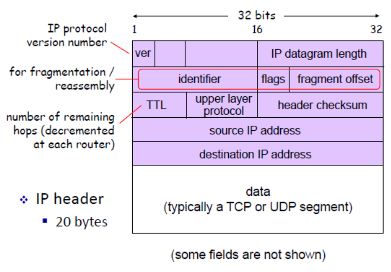
Other features of TCP
- Flow control
→ Sender wont overflow receiver's buffer by sending too much or too fast
→ receiver feedbacks to sender how many more bytes it is wiling to accept
- Congestion control
→ Send less if network is congested

TCP and Web Servers
For each persistent connection, web server creates a separate 'connection socket', identified by the four-tuple:(src IP, src port, dest IP, dest port)
Requests from different hosts come in from different sockets. They all have 80 as destination port. However, the identifiers for these sockets have different IP addresses to differentiate.

Network Layer
IP Address
- 32 bit integer expressed in binary/decimal

IP Header:

IPv4 Datagram Format


32 bits
1
16
32
IP protocol version number
for fragmentation / reassembly
number of remaining hops (decremented at each router)
ver
IP datagram length
identifier
flags
fragment offset
TTL
upper layer protocol
header checksum
source IP address
destination IP address
data (typically a TCP or UDP segment)
IP header
20 bytes
(some fields are not shown)

Checksum calculates only **datagram's header.**
20bytes in size.
Length in IP datagram includes header length.
→ MTU (Maximum transfer unit) includes header as well.

IP is either:
- Manually configured by sysadmin
- Automatically assigned by DHCP (Dynamic Host Configuration Protocol) server.

DHCP allows host to dynamically obtain its IP address from DCHP server when it joins the network.
- IP is renewable
- Addresses can be reused (host only holds address when connected)
- support mobile users who want to join network.
s
4-step process:
1. Host broadcasts 'DHCP Discover' message
2. DHCP server replies with 'DHCP offer' message
3. Host requests IP Address: 'DHCP request' message
4. DHCP server sends IP Address with 'DHCP ACK' message

Other features of DHCP
DHCP provides a host additional network information:
- IP Address of first-hop router
- IP Address of local DNS Server
- Network Mask (Network prefix vs host ID of an IP Addr)

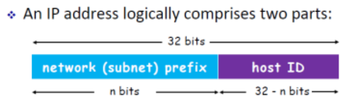
DHCP **runs over UDP** ← UDP is fast
DHCP server port number:67

DHCP client port number: 68

Special IP Addresses
0.0.0.0/8 – special IP that is used temporarily when host just joins network until it is assigned IP from DHCP server
127.0.0.0/8 – Loopback address. **Any** address with this network prefix are loopback addresses, or ‘localhost’
10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 – Private IP Addresses. Any IP with these network prefix are used for private networks, like home/school networks etc. Not globally unique
255.255.255.255/32 – Broadcast Address. All hosts on the this subnet will receive a packet if this destination address is specified.

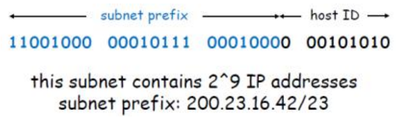
- An IP Address is associated with a network interface.
- A *host* usually has one or two network interfaces (e.g. wired ethernet and wifi)
- A *router* has multiple network interfaces

IP Address and Subnet

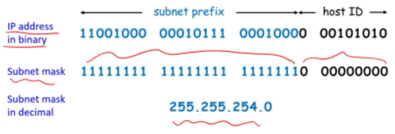

❖ An IP address logically comprises two parts:

- A *subnet* is a network formed by a group of ‘directly’ interconnected hosts.
→Hosts in the same subnet have the same network prefix of IP addresses.
→Hosts in the same subnet can physically reach each other without intervening router (first hop router)
→ Connect to the outside world through a router.

CIDR
Internet’s IP address assignment strategy is known as Classless Inter-Domain Routing (CIDR).
- Facilitates routing.
- Subnet prefix of IP is of arbitrary length.
- Address format: a.b.c.d/x, where x is the number of bits in subnet prefix


subnet prefix
host ID
11001000 00010111 00010000 00101010
this subnet contains 2^9 IP addresses
subnet prefix: 200.23.16.42/23

Subnet Mask
- Used to determine which subnet an IP belongs to



Hierarchical Addressing
- IP addresses allocated in blocks
→ facilitates routing
→ ISP can take all packets that have x number of subnet prefix, and from there, disseminate packets based on x + y number of subnet prefix

If an organisation now switches to another ISP, but don’t want to renumber all of its routers and hosts, just add instruction “Send me any packets with addresses beginning with w.x.y.z/16 or a.b.c.d/23”.
→ Works *most* of the time, by checking its *forwarding table* to decide where is the next hop, but sometimes subnet prefixes might be very similar
→ Longest Prefix Match – whichever IP has a longer prefix match, the next hop of the packet will be of that router as referenced from forwarding table.
Forwarding – happens inside a router. Based on longest prefix match, router will be able to identify next-hop router.

Routing Algorithms
Intra-AS (Autonomous System) routing
- Finds path between routers within an AS
→ Commonly used protocols: RIP, OSPF
→ Mostly focused on performance
→ Use of shortest path algorithms such as Djikstra’s, Bellman-Ford
→ Routing Information Protocol (RIP) implements Distance Vector (DV) algo, uses hop count as cost metric (ignores network congestion)
→ Entries in routing table are aggregated subnet masks
→ Exchange routing table every 30 secs over UDP port 520
→ “Self-repair”: If no update from neighbour for 3 minutes, assume neighbour has failed

Network Address Translation (NAT)
→ IP address is split between private IP within a LAN, and public IP. Different machines in different LANs can have the same private IP.
→ Packets go through a NAT router to assign the packet a public IP and port number so that servers in the internet can differentiate between packets that can come from different LANs, but yet have same private IP.

NAT routers
- Replace (source IP address, port #) of every outgoing datagram to (NAT IP addr, new port #).
- Remember (in NAT translation table) the mapping from (src IP addr, port #) to (NAT IP addr, new port #).
- Replace (NAT IP addr, new port #) in destination fields of every incoming datagram with corresponding (src IP addr, port #) stored in NAT translation table.

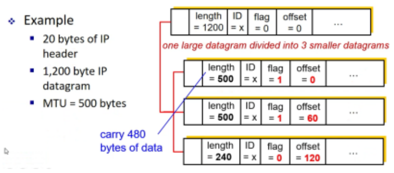
Benefits of NAT
- No need to rent more public IP addr from ISP: 1 public IP for NAT router
- All host use private IP addr, can change addr of host in LAN without notifying outside world
- Can change ISP without changing addr of hosts in LAN
- Host inside LAN are not explicitly addressable and visible from outside (better security)

Internet Protocol (IPv4)
- IP header – 20 bytes (80 bits)
- checksum only for IP header
- upper layer protocol – TCP/UDP
- TTL – number of remaining hops (decremented at each router, ensures packet doesn't circulate in network forever)
- IP fragmentation

Different links have different MTU (Max transfer unit) – maximum data a link layer frame can carry
→ IP datagrams that are too large will be fragmented by routers

Host reassembles packets, need to know number & order of fragments
→ Given in IP Header

IP Fragmentation
- Uses length, identifier, flags, offset fields in IP header
- length INCLUDES IP header's 20 bytes
- MTU includes IP header as well
- ID specified by sender
- flag = 1 → there is a next fragment, 0 → last fragment
- offset expressed in unit of **8 bytes**

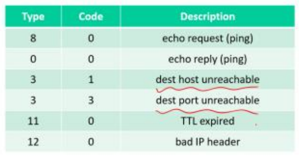


- Example
  - 20 bytes of IP header
  - 1,200 byte IP datagram
  - MTU = 500 bytes

ICMP – Internet Control Message Protocol
- Commonly used for network troubleshooting
- ICMP messages carried in IP datagrams, after IP header

IP header
ICMP header

- ICMP header: Type + Code + Checksum + others.



Type	Code	Description
8	0	echo request (ping)
0	0	echo reply (ping)
3	1	dest host unreachable
3	3	dest port unreachable
11	0	TTL expired
12	0	bad IP header

Ping command
Traceroute – shows all routers in between sender and remote host

<u>Link Layer</u>
Services provided:
- Framing → encapsulate datagram with LL frame, consist of header + datagram
- Link access → MAC protocol – coordinates frame transmission when multiple nodes on both ends share 1 link. Sender sends frame when link is idle if both ends only have 1 node
- Reliable Delivery – similar to TCP
- Error Detection **& correction** – More sophisticated than checksum in IP/transport layer

6.2 Error Detection & Correction
- Allows receiver to *sometimes* detect bit errors that have occurred
→ Parity Checks, checksum, cyclic redundancy

Parity Checking
- Single Parity Bit (PB)
→ Even vs odd parity schemes
- Ensure number of 1s (incl PB) is even
Vs number of 1s is odd (odd scheme)
→ **Only detects** errors

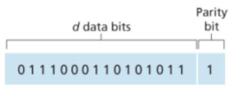


Figure 6.4 One-bit even parity

2-Demensional Parity Scheme
- bits divided into i rows and j cols
- parity value computer for each row and col, if err occurs, parity of that row and col with the bit err will be wrong & we can identify which bit was corrupt
- Can **correct** the error
- **Forward Error Correction (FEC)** → Ability to detect & correct errors
→ Decrease sender retransmission, avoid waiting for round-trip delays
- can identify but not correct 2 bit err

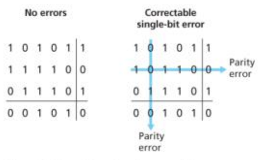
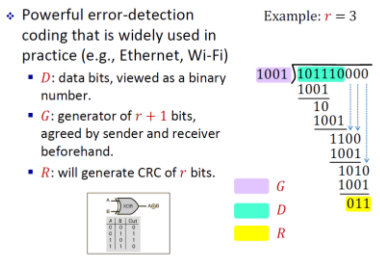


Figure 6.5 Two-dimensional even parity

Checksum
- Not really used in link layer, use CRC instead

Cyclic Redundancy Check (CRC)



- Powerful error-detection coding that is widely used in practice (e.g., Ethernet, Wi-Fi)
  - D: data bits, viewed as a binary number.
  - G: generator of $r+1$ bits, agreed by sender and receiver beforehand.
  - R: will generate CRC of r bits.

Example: $r = 3$

```
1001 )101110000
      1001
        10
        1001
          1100
          1001
           1010
           1001
            011
```

G D R

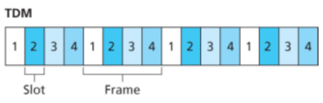
- Bitwise XOR without carry or borrow
- Sender sends (D,R)
- Receiver knows G, divides (D,R) by G
→ Non-zero remainder = there is error

6.3 Multiple Access Links & Protocols
Network Links
- Point to point link – sender and receiver connected by dedicated link
- Broadcast link – multiple nodes connected to shared broadcast channel
→ Channel broadcasts a frame transmitted and all nodes receive copy

- Coordination of the access of multiple sending and receiving nodes to a shared broadcast channel – multiple access problem

Multiple Access (MA) Protocols
- Channel Partitioning Protocols, Random Access Protocols, taking-Turns Protocols
- Characteristics of a MA protocol:
  - When only 1 node has data to send, that node has throughput of R bits
  - When M nodes have data to send, each node has average R/M bps throughput over an interval of time
  - Protocol is decentralized; there is no master node that represents single point of failure for network
  - Protocol is inexpensive to implement

Channel Partitioning Protocols
TDMA (Time Division Multiple Access)
- Partition bandwidth among all nodes into time frames, which is further divided into time slots.
- Each time slot is assigned to one of the N nodes, node transmit packet during assigned time slot



+ Eliminates collision, perfectly fair
- Node is limited to average rate of R/N bps even if it is the only node with packets to send
- Node must always wait for its turn to transmit packet

FDMA (Freq Division Multiple Access)
- Divides R bps channel into diff freq.
- Create N smaller channels of R/N bps
- Similar advantages and drawbacks of TDMA

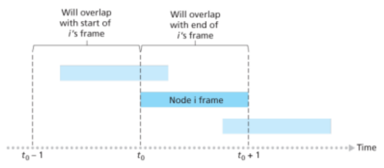
Random Access Protocols
- Transmitting node always transmit at full rate of channel, R bps
- When collision occurs, wait a random delay before retransmitting frame. Each node involved chooses independent random delay

Slotted ALOHA Protocol
Assumptions
- All frames have exactly L bits (equal)
- Time is divided into slots L/R (i.e. a slot equals time to transmit 1 frame)
- Nodes start to transmit frames at beginning of slots
- All nodes detect collision event before slot ends if >= 2 frames collide

Operation:
1. When node has new frame to send, wait until beginning of next slot and transmit the frame in that slot
2. If no collision, end
3. If collision, node detects collision before end of slot, retransmit its frame in each subsequent slot with probability *p* until frame is transmitted w/o collision

Advantages:
- Allows node to transmit at full rate R if it is the only node with frames to send (active)
- Highly decentralized; each node detects collisions and independently decides when to retransmit
- Very simple protocol
Disadvantage:
- Not very efficient when there are many active nodes, probability of collision will be very high

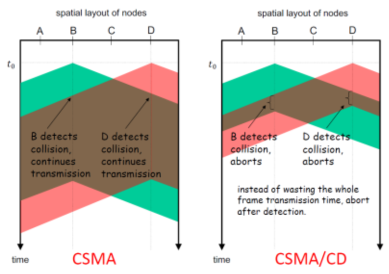
ALOHA Protocol
- Same assumptions as slotted ALOHA
- No slot, no synchronisation, chance of collision increases
→ Efficiency of pure ALOHA is only half of slotted ALOHA


Will overlap with start of i's frame
Will overlap with end of i's frame
Node i frame
Time
$t_0 - 1$
t_0
$t_0 + 1$
Figure 6.11 Interfering transmissions in pure ALOHA

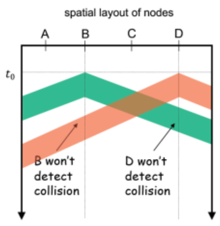
CSMA (Carrier Sense Multiple Access)
Carrier Sensing – Listen to channel before transmitting
Collision Detection – Transmitting node listens while transmitting, stop transmitting and waits random time before retransmitting
→ CSMA and CSMA/CD protocols

Why is there collisions if there is CS?
- Assume network of nodes A, B, C, D, node B senses channel is idle, begins transmitting. Propagation of B's bits is non-zero. After, node D has a frame to send, thinks channel is idle because bits from B have yet to reach D, sends frame, and after short period of time, B's transmission interferes with D's transmission
- In CSMA, nodes do not perform collision detection, continues to transmit their frames even if collision occurs

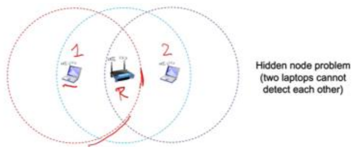
CSMA/CD
Nodes abort transmission a short time after detecting a collision


spatial layout of nodes
A
B
C
D
t_0
B detects collision, continues transmission
D detects collision, continues transmission
time
CSMA
spatial layout of nodes
A
B
C
D
t_0
B detects collision, aborts
D detects collision, aborts
instead of wasting the whole frame transmission time, abort after detection.
time
CSMA/CD
- Both nodes will abort transmission
- Both nodes waits a random amount of time to retransmit frame
- Need to choose random time where interval is short when no. of colliding nodes are small and large when no. of colliding nodes are large to maximise efficiency

Minimum Frame Size
- Frame cannot be too small, as collision might happen but may not be detected.
- Ethernet req. min size of 64 bytes



CSMA/CA Protocol
Collision Detection difficult to detect in wireless LANs
- Hidden Node problem
→ The transmission range of 1 node in the LAN cannot reach the other in the same LAN, so they do not know of each other's existence
→ CSMA/CD protocol is not appropriate, use CSMA/CA instead, where receiver (router) need to return ACK is frame is ok



Taking Turns Protocols
Polling protocol
- Master node polls each node is a round-robin fashion, tells each node the max number of frames it can transmit
+ Eliminates collisions and empty slots, much higher efficiency
- Polling Overhead – amt of time req to notify node that it can transmit
- Single point of failure, if master node fails, channel is inoperative
→ Bluetooth is a polling protocol

Token-passing protocol
- No master node
- Special-purpose frame called token exchanged among nodes in a fixed order, nodes holds onto token only if it has frames to transmit
+ Decentralized and highly efficient
- Failure of one node can crash entire channel
- Recovery procedures must be invoked to get token back in circulation if node accidentally neglects to release token

Switched Local Area Network (LAN)
- Use link-layer addresses to forward link-layer frames through network of switches
- Link-Layer Addressing & ARP, Ethernet protocol, how link-layer switches operate

Link-Layer Addressing & ARP
- Ipconfig/ifconfig – find out MAC addresses of your machine
- arp – show mappings in arp module
- MAC Address – adapters of hosts/routers have LL addresses
- Note: Switches have no LL address
- MAC address: 6 bytes (48 bits) long, fixed, typically in hexadecimal notation
- Adapter will check if dest MAC address of frame matches own MAC, if yes, pass to protocol stack, else discard
- Receiving adaptors will always process FF-FF-FF-FF-FF-FF, the broadcast frame

- Why have MAC?
+ Support other network-layer protocols (IPX, DECnet)
+ If no MAC, network protocol must check more matching addresses
+ Abstract each layer, make them less dependent on each other

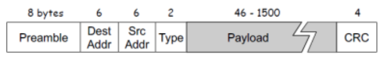
Address Resolution Protocol (ARP)
- Translate between network-layer (IP) and link-layer addresses (MAC)
- ARP module is in sending host, you pass it dest IP, it gives you dest MAC
- ONLY resolves host in same subnet
- Basically mapping table of IP to MAC
- Contains TTL value, typically 20 mins
- So what if ARP table doesn't have entry for destination?

ARP Protocol
- Sender constructs ARP packet, contains sending and rcving IP/MAC addresses, same for query & resp pkts
- Sender uses broadcast MAC add, dest ARP modules check IP, return with resp pk, querying host updates ARP table with cached data
- Query sent in broadcast frame, resp sent in standard frame

Sending Data to Another Subnet
- Cannot use MAC address of adaptor of host in another subnet as adapters wont even send it up to its network layer
- Indicate MAC address of router interface by referring to ARP table

- Router refers to its forwarding table to determine correct interface for datagram to be forwarded, pass datagram to that adapter, sends frame to other subnet, use ARP to resolve dest MAC and successfully sends

Ethernet
- Most prevalent wired LAN technology
- Used bus topologies in early years, now typically star topologies are used, with a switch in the centre
- No handshaking between sending/recving adapters (unlike TCP)
- Does not ensure reliable transport, if frame fails CRC, recv discards frame but sender does not know. Relies on transport layer (TCP) to resend packet. If UDP, then application will see gaps in data
- Ethernet uses CSMA/CD Protocol in bus topology, similar to that in LL.
- CSMA/CD is not needed in current switch-based Ethernet as modern switches are full-duplex (Switches and nodes can send frames to each other w/o interference)
→ Listen to channel before transmission, if sensed busy, wait until idle then transmit. If no collision, success. Else, when adapter detects transmission while transmitting, stop immediately, send a JAM signal to notify all NICs.
→ After aborting, enter binary backoff: After m collisions, choose random number K at random from {0,1,...,2^m-1}. NIC waits K * 512 bit times, retry send.

Ethernet Frame Structure


- Payload/Data Field: (46 to 1,500 bytes): Carries IP Datagram. MTU of Ethernet is 1,500 bytes, if >1,500, fragment datagram. If <46, add "stuffing" to fill out to 46 bytes, network layer uses length field to remove stuffing
- Destination Address:
6 bytes, contains MAC address of destination adapter
- Src Address:
6 bytes, contains MAC address of source adapter
- Type field:
2 bytes, permits Ethernet to multiplex network-layer protocols besides IP (Novell IPX, AppleTalk), with their own standardized type number, ARP also has own type number
- CRC (4 bytes):
Allow receiving adapter to detect bit errors in the frame, corrupted frame dropped
- Preamble (8 bytes)
Begins with preamble, first 7 bytes of preamble is 10101010, last byte 10101011. Used to sync clocks to sender's clock.
Provides square wave pattern to tell recver sender's clock rate; tells the recver width of a bit

Link-Layer Switches
- Receive incoming LL frames and forward to outgoing links
- Transparent to other hosts
- Switches have buffers to accommodate interfaces of switches exceeding link capacity

Forwarding/Filtering
- Filter: Determines whether a frame should be fwded/dropped by some interface
- Fwding: Determines the interface to which a frame should be directed
→ Both done with a switch table

Switch Table
- Contains entries of hosts and routers in the LAN (some, not always all)
- Each entry: MAC address, interface that lead towards that MAC, time entry was placed in table.
- Switches fwds packets based on MAC rather IP addresses

How it works:
- A frame arrives at switch on interface x:
  - If not entry in table for dest MAC address, switch broadcasts the frame
  - If there is entry in table, two cases: If dest MAC address comes from x, switch filters the packet and discards it.
If dest MAC address of entry comes from y =/= x, forward frame to LAN attached to y, performs fwding by placing frame in output buffer preceding interface y

Self-Learning
- Tables are self-learning
- Switch tables initially empty
- For each incoming frame, switch stores new entry in its table.
- By TTL, if no frames received with that address as src add, switch deletes entry from table
→ Switches are plug-and-play devices, requiring no intervention from admin

Properties of LL-Switching:
→ Elimination of collisions, no wasted bandwidth

Switches vs Router

- Both store-and-forward, but switches forward using MAC, router use IP
- Switches layer-2 packet switch, router layer-3 packet switch

Switches – Plug & Play, process up to layer 2, <u>BUT</u> large LANs require large ARP tables & generate substantial ARP traffic, susceptible to broadcast storms (if 1 host goes haywire and transmits endless stream of ethernet broadcast frames, switches fwds all frames and network collapses)

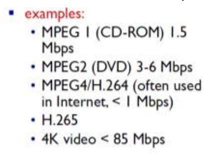
Routers – Network addressing hierarchical, pkts wont cycle thru router, calculates best path between src and dest, provide firewall protection against layer-2 broadcast storms, <u>BUT</u> not plug-and-play, larger processing time.

<u>Multimedia Networking</u>
- Video is extremely prominent in the internet
- Split into audio and video multimedia

Video
- OTT (over the top): svc offered directly to users via internet, bypassing cable, broadcast, satellite tv platforms

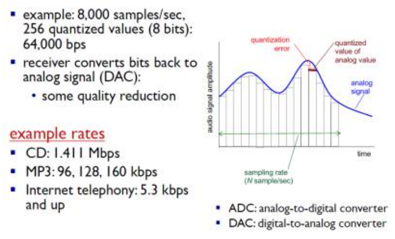
-Spatial Redundancy
→ Redundancy within a given image – image consisting mostly of white space has a high degree of redundancy and can be compressed
- Temporal Redundancy
→ Repetition from image to subsequent image – if image and subsequent images are the same, no reason to re-encode image, more efficient to indicate subsequent image is same during encoding

→ Compress video to desired bit-rate, higher bit rate means better img qlity
→ Use compression to create multiple versions of a video
→ Video-conferencing applications can compress on-the-fly to provide best quality given available end-to-end bandwidth
→ Constant bit rate (CBR) – video encoding rate fixed, vs Variable bit rate(VBR) – rate changes as amount of spatial/temporal coding changes


- examples:
  - MPEG 1 (CD-ROM) 1.5 Mbps
  - MPEG2 (DVD) 3-6 Mbps
  - MPEG4/H.264 (often used in Internet, < 1 Mbps)
  - H.265
  - 4K video < 85 Mbps

Audio
Telephone/ VoIP: 8,000 samples/s
CD music 44,100 samples/s, 2^{16}quantized values = 16 bits/sample
- Significantly lower bandwidth requirements than video

How it works
- Analog sample will be sampled at a fixed rate, for example, 8000 samples per second. Value of each sample is some real number.
- Each value is rounded to a quantization value, and each sample will be represented by it. Bit representations of the sample are concatenated to form digital representation of signal.
→ Digital representation is simply a approximation of the original signal, increasing sampling rate and number of quant values will give better sound quality.
→ Technique above is known as pulse code modulation (PCM).

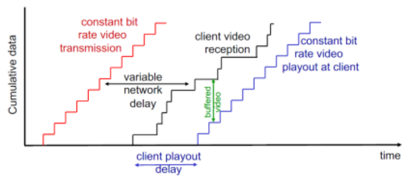

- example: 8,000 samples/sec, 256 quantized values (8 bits): 64,000 bps
- receiver converts bits back to analog signal (DAC):
  - some quality reduction

example rates
- CD: 1.411 Mbps
- MP3: 96, 128, 160 kbps
- Internet telephony: 5.3 kbps and up

- ADC: analog-to-digital converter
- DAC: digital-to-analog converter

Streaming stored video
- User send requests to these servers to view videos on demand
- Interact by watching from start to end, stop watching before vid ends, or pausing/repositioning to a future/past scene.
- UDP, HTTP, adaptive HTTP streaming
- When video arrives at client, client builds up reserve of video in app buffer

→ Absorbs variations in server-to-client delay
→ If server-to-client bandwidth drops below video consumption rate, user continues enjoying continuous playback so long app buffer isn't drained



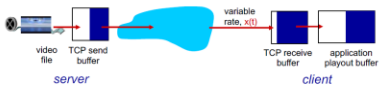
- *client-side buffering and playout delay:* compensate for network-added delay, delay jitter

UDP Streaming

- Server transmits video at rate that matches client's video consumption rates using UDP
- UDP has no congestion-control mechanism, so server can push packets into network without rate-control.
- Small client-side buffer, <5s of video
- push-based streaming (server push)
- May not go through firewalls
- Server encapsulates vid chunks with transport packets using Real-Time Transport Protocol (RTP).

HTTP Streaming
- Video is stored in HTTP server as file with specific URL
- When client wants file, it will retrieve via HTTP GET (pull-based streaming)
- Server sends as quickly as TCP congestion control and flow control allows.
- Bytes are collected in client app buffer on client side

When transmitting a file over TCP, server-to-client trans rate can vary a lot due to TCP's congestion control. Packets can also be significantly delayed due to TCP's retransmission mechanism



+ traverse firewalls, NATs
+ don't need media control server such as RTSP server, reduce cost of large-cost deployment

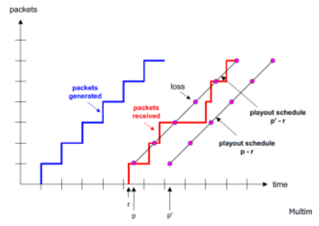
<u>VoIP (Voice over IP)</u>
- Transmitted via UDP
-Timing considerations important as conversational apps are highly delay-sensitive.
→ <150ms not perceived, 150ms<d<400ms acceptable, >400ms is bad
→ Segment sent into socket every 20ms during talk spurt

Limitations of IP protocol for VoIP
- IP provides best-effort svc, lack of guarantee is bad for real-time conversational apps
→ Enhance VoIP's performance!

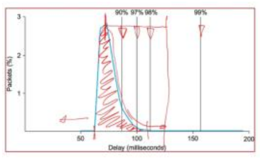
- Packet Loss
→ Datagram may be discarded when buffer in path from sender to receiver is full
→ Loss tolerance: loss rates between 1 and 10% can be tolerated
→ If use TCP? End- to-end delay may increase, and due to congestion control, sender's transmission may decrease to a rate lower than receiver's buffer drain rate → buffer starvation (delay loss)

- Packet Jitter
→ Time from which packet is generated to received can vary (esp due to queuing delay). Phenomenon is called packet jitter
→ If receiver ignores presence of jitter and plays chunks as soon as they arrive, resulting audio quality can be unintelligible
→ Use of playout delay / timestamps / sequence numbers to remove jitter

Fixed playout delay
-Rcvr plays chunk exactly q ms after chunk is generated
- If chunk has timestamp t, receiver plays out chunk at t + q. If packet arrive after scheduled playout time, its discarded
→ Smaller q -> more satisfying convo
→ Big q if large variations in e2e delay
→ Small q < 150ms used if small variations in e2e delay



Adaptive Playout Delay
- Challenge of low playout + low loss seen in fixed delay
- Solution: Estimate network delay, adjust playout delay at beginning of each talk spurt
- Idea of algorithm – Estimate delay of future packet by taking a weighted ratio of the delay of previous packet and current packet
- If estimate is wrong, we can compress silence periods and play talk spurt earlier
- Select decent playout time and buffer a bit of data if possible in case of jitter by looking at standard dev of delay
- Set delay at a point where most packets are alrd recved



Loss Anticipation: Forward Error Correction (FEC)
Idea: Add info to original packet stream, this info can be used to reconstruct approximations of lost packets
Simple FEC
- For every group of n chunks, create redundant chunk by XOR-ing the n chunks
- send n+1 chunks, which increases transmission rate by factor of 1/n
- If 1 packet lost in the n, redundant packet reconstructs lost packet (but cannot reconstruct 2 or more)
- Increases playout delay since recvr must wait for whole group of packets before it begins playout

Second FEC Scheme
- Send lower-res audio stream as redundant info
- Sender constructs nth packet by taking nth chunk + (n-1)th chunk from redundant stream (piggy backing)
- With non-consec packet loss, recvr conceal loss by playing low-bit rate encoded chunk that arr w subsq pkt
→ Only has to recv 2 packets before playback – small increase in playout delay
→ Marginal increase in transmission rate if low-bit rate encoding is much less than nominal encoding

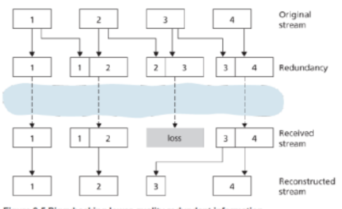
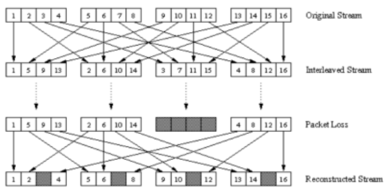


Figure 9.5 Piggybacking lower-quality redundant information

Interleaving
- Resequence units of audio data before transmission
- Mitigates effect of packet loss
- Loss of one packet simply results in small gaps in reconstructed stream, as opposed to 1 large gap
+ Improve perceived audio quality, low overhead
+ dont increase bndwdth req of stream
- Increase latency, not good for conversational apps like VoIP



Protocols for real-time conversational apps: Real Time Protocol (RTP), SIP

RTP Suite: Consists of RTP, RT Control P (RTCP), RT Sender P
- RTP – actual media stream
- RTCP – lightweight control protocol, one packet every few secs, sent by both sender and recv, provide stats about loss rates, congestion controls, over UDP
-RTSP – over TCP, commands such as play, pause etc are sent over RTSP

- Most multimedia networking apps can use info like sequence numbers and timestamps, good to have standardized packet structure
- RTP – can transport formats such as PCM, ACC, MP3 for sound, MPEG etc for video
- Runs on top of UDP
- Sender encapsulates media chunk within RTP packet, then in a UDP segment and hand over to IP
- RTP header is 12 bytes
- Contains type of audio encoding, seq number, timestamp
- RTP is purely for end-systems – no ensuring of reliability of data xfer etc
- Since it forces standardization, VoIP apps developed by different companies might be able to interoperate

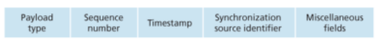


Figure 9.8 RTP header fields

Payload header – 7 bits long
- Indicate type of audio encoding
- Sender may change encoding middle of a session to increase audio quality or decrease RTP stream bit rate
- Also can indicate type of vid encoding
- Can also change in mid of session

Seq number field – 16 bits long
- Increments by 1 for each RTP pkt sent, used to detect packet loss etc

Timestamp field – 32 bits long
- timestamp that first byte in RTP packet was sampled, used to remove packet jitter and provide synchronous playout

SSRC – 32 bits

- Synchronisation source identifier
- Identifies source of RTP stream
- Each stream in an RTP session has distinct SSRC
- A number src assigns randomly, senders pick new SSRC value if clash

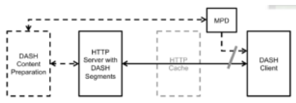
WebRTC
- Not really used anymore, but used for low-latency 1/2-way communication
- RTP/RTCP protocols are implemented in browsers, can be called using JS APIs

SIPs: Session Initiation Protocol
Open and lightweight protocol that:
- Provides mechanisms for establishing calling: caller notify callee, allows participants to agree on media encodings, end calls etc
- Allows caller to determine curr IP of callee
- Allows for call management, such as adding new media streams, changing encoding, inviting new participants, call xfer, call holding all during call

DASH
- Streaming challenges – need special purpose server for fine-grained packet scheduling, keep state
- TCP/UDP both used (firewall blocks)
- Difficult to cache data
+ Short e2e latency (<100-500ms)

HTTP Streaming
- Increasingly being used
- Dynamic Adaptive Streaming over HTTP (DASH)
- Main Idea: Use HTTP to stream media, divide media into streamlets
+ Simple web server needed
+ No firewall issues
+Standard web caching works
- Based on media segment transmissions (2-10s in length), does not provide low latency for interactive, 2-way applications (vid conferencing)

- Encode media in multiple different streamlet files (low/medium/high)



MPD: Media Presentation Description

How it works:
- Web server provides playlist, in a format that lists all available qualities & and all streamlets for each quality
- .m3u8/.mpd extension
→ Media file split into streamlets, streamlets transcoded into different qualities
- ISO/IEC Standard
- Client downloads MPD, which describes available videos and qualities
- Client/player executes an adaptive bitrate algo (ABR) to determine which segment to download next
- Has replaced almost all VoD streaming protocols: youtube, FB, Netflix
- Recent focus is on lower latency

Summary:
- VoD, live-stream: Use DASH
- VoIP, teleconferencing – use RTP

<u>**Network Security**</u>
- **Confidentiality** – Only sender and intended recvr can understand contents of transmitted message. **Encrypt** message so intercepted msg cant be understood by interceptor
- **Message Integrity** – Ensure contents of messages is not altered (maliciously or by accident) in transit – use of checksums + more
- **End-point authentication** – Sender and recvr should be able to confirm identity of other party involved
- **Access & Availability** – Svcs must be accessible and available to users
- **Operational Security** – Firewalls & intrusion detection systems to control packet access and do deep packet inspections

- Intruder can eavesdrop (sniff and record control/data msgs on channel), or modify, insert, dlt msgs.
- Can spoof source address in packet (impersonation)
- Hijack connection, prevent svc from being used by others
- Online transactions, DNS servers, routers exchanging routing table updates..

8.2 Principles of Cryptography
- Allow sender to disguise data so an intruder cannot gain info from intercepted data

- Symmetric Key & public key systems
=> KB(KA(m)) = m

3 different scenarios based on info intruder has:
1. Ciphertext-only attack – access to only ciphertext with no certain info of contents of plaintext msg
2. Known-plaintext attk – Intruder knows some of the plaintext-ciphertext pairings
3. Chosen-plaintext attk – Intruder has a certain plaintext's ciphertext message, when that msg is sent, Trudy can break encryption scheme

Block Cipher
- Message to be encrypted processed in blocks of k bits. Cipher uses a 1-to-1 mapping to map k-bit blocks of cleartext to k-bit blocks of ciphertext

Symmetric Key Cryptography
- Bob & Alice share same (symmetric) key: Ks
- Simplest cipher: Caesar's Cipher, encryption key = shift number
- monoalphabetic/polyalphabetic ciphers: Provide mappings, switch in a specific order encoding using different mapping tables
- Block ciphers and stream ciphers
- Block ciphers used in many internet protocols: PGP (email), SSL (TCP conn), IPsec (network-layer transport)

- Popular block ciphers include DES, 3DES, AES
- Use functions to encrypt plaintext
- DES uses 56-bit keys w 64-bit blocks
- DES not very secure, can be decrypted with brute-force in 1 day

AES (Advanced Encryption Standard)
- Replaced DES, process data in 128 bit blocks, 128/192/256 bit keys

Public Key Encryption
Goal: 2 parties communicate with encryption without having a shared secret key that is known in advance
- Concept of Bob having both a public key and private key, denoted by KB+ and KB- respectively.
- Alice encrypts msg m using KB+, computing KB+(m). Bob uses KB- to decrypt m, i.e. KB-(KB+(m)) to get m.
- This way, no secret key had to be exchanged earlier!
- Problem: Trudy can mount any plaintext attack using Bob's public key, encode any msg she chooses
→ Key selection, encryptn/decryptn must be done in way that it is impossible for Trudy to determine Bob's priv key/decrypt/guess Alice's m
- Problem: Trudy can impersonate Alice and send encrypted msg to Bob
→ Digital Signatures

RSA
- Extensive use of modulo-n arthmtic
- facts:
  $[(a \bmod n) + (b \bmod n)] \bmod n = (a+b) \bmod n$
  $[(a \bmod n) - (b \bmod n)] \bmod n = (a-b) \bmod n$
  $[(a \bmod n) * (b \bmod n)] \bmod n = (a*b) \bmod n$
- thus
  $(a \bmod n)^d \bmod n = a^d \bmod n$
- Every msg is just a bit pattern which can be represented uniquely by integer
→ Encrypting msg == encrypting unique integer representing msg
- KB-(KB+(m)) = m = KB+(KB-(m))
- Secure because need to find factors of n without knowing p and q, and factoring number is hard

Generating public/priv RSA keys
1. Choose 2 large primes p, q. Larger == harder to break, but longer to perform encoding/decoding
2. Compute n=pq, z = (p-1)(q-1)
3. Choose a number e < n, that is relatively prime to z. e for encryption
4. Find d such that ed-1 is exactly divisible by z. d for decryption.
4a. In essence, ed mod z = 1.
5. KB+ = (n, e); KB- = (n, d)

- Encode: c (encrypted value)=m^emod n
- Decode: m (msg) = c^dmod n

Session Keys
- Exponentiation in RSA computationally expensive, DES abt 100 times faster than RSA
- Solution? If Alice wants to send Bob large amt of encrypted data, session key used to decrypt message using DES/AES encrypted with Bob's public key and sent to Bob. Bob recvs session key, decrypts it to obtain Ks. Bob now use Ks to decrypt actual data.

8.3 Message Integrity/Digital Signatures
- Providing message integrity → Digital signatures & end-point authentication
- Recvr needs to verify that msg originated from sender & not tampered with on its way to recvr
- Verifiable, nonforgeable

Digital Signatures
- Cryptographically 'sign' a document
- Using public-key cryptography

- Use of RSA again. Bob simply uses his private key to encrypt his own message, m, producing KB-(m).
- Alice applies KB+ to get KB+(KB-(m)).
This will produce m again (*verifiable)*
→ The only person who knows KB- is Bob: *non-forgeable*
→ Also provides msg-integrity: if message ever modified, KB+(KB-(m)) will not produce m'

Problem: Encryption/Decryption computationally expensive → Use message digest
- Auth key needed to ensure msg integrity with a secret key, s:
  - Alice creates msg m, concatenate s with m to create m + s, calculate hash = H(m+s) = MAC (message auth code).
  - Alice appends MAC to m (m, H(m+s)), sends extended msg to Bob
  - Bob recvs extended (m, h), and knowing s, calculates H(m+s) = h.
- Also known as **message digest**
- Fixed length, easy to compute digital fingerprint. Sign this instead!

Hash Functions
- Checksum is a poor internet hash function, can easily find 2 msgs with same H(m).
→ MD5: computes a 128-bit message digest, one small change in input results large change in hash output! SHA-I: 160-bit message digest
→ Password hashing – passwords not stored in plaintext, but the hashed value is stored. When password is input, hash it and check that it matches stored hash

Public Key Certification
- Certifies that public key belongs to a special entity: IPsec, SSL
- Without this, Trudy can send message saying she's Bob, use Trudy's own public and private key to digitally sign document, Alice applies Trudy's public key thinking its Bob and verifies incorrectly
→ A Certification Authority (CA) validates identities and issues certs
- Certs are binded to public key of entity, cert contains public key and globally unique identifying info, digitally signed by CA

8.7. VPNs
- Encrypt traffic before sent to public internet
- Still communicate using IPv4 within private network
- VPN converts vanilla IPv4 datagram into IPsec datagram and forwards to internet. Routers on internet interpret this as a normal packet
- When it reaches recvr, OS in laptop decrypts the payload and passes it to upper-layer transport protocol

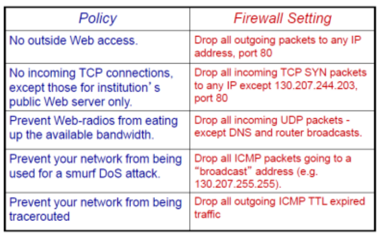
Firewalls
- Isolates an organization's internal network from Internet at large
- Allows network admin to control access between outside world and resrces by manage traffic flow to and from these resrces

Goals:
- All traffic from outside and inside passes through firewall
- Only authorized traffic defined by local security policy, can pass
- Firewall itself immune to penetration

- Prevent denial of svc attks:
  - SYN flooding: attacker establishes bogus TCP connections, no resrces left for 'real' connections
- Prevent illegal modification/access of internal data
- 3 types: stateless packet filters, stateful packet filters, application gateways

Stateless Packet Filters
- Packet filter examines each datagram in isolation, determining whether datagram should pass or be dropped in an access control list based on:
  - IP src and dest addr
  - Protocol type: TCP, UDP, OSPF, ICMP
  - TCP/UDP src and dest port
TCP Flag bits: SYN, ACK

Configure firewall based on policy of organisation, can take user productivity and bandwidth usage, security concerns into consideration



Policy	*Firewall Setting*
No outside Web access.	Drop all outgoing packets to any IP address, port 80
No incoming TCP connections, except those for institution's public Web server only.	Drop all incoming TCP SYN packets to any IP except 130.207.244.203, port 80
Prevent Web-radios from eating up the available bandwidth.	Drop all incoming UDP packets - except DNS and router broadcasts.
Prevent your network from being used for a smurf DoS attack.	Drop all ICMP packets going to a "broadcast" address (e.g. 130.207.255.255).
Prevent your network from being tracerouted	Drop all outgoing ICMP TTL expired traffic

- Example settings

Stateful Packet Filtering

- Stateless packet filtering looks at each packet in isolation, stateful packet filtering can track TCP connections and make filtering decisions
- Track beginning of new connection by observing 3 way handshake/setup (SYN, SYNACK, ACK); observe end of connection/teardown when it sees FIN pkt
- Conservatively assume connection is over if no activity in for say, 60 secs
- 'Check connection' column in access control list to track TCP connections
- Able to set such that it allows users to surf web, but not allow incoming initiation of connections

Application Gateway

- Provides telnet svc to restricted set of internal users, allow such users to auth themselves first before being allowed to create telnet sessions, app layer data that is not included in IP/TCP/UDP headers

→ Combine packet filters with app gateways for finer-level security, make decisions based on app data

- Multiple app gateways can run on 1 host
- Internal users must set up telnet sesh with app gateway → gateway prompts for user and password & check if user has permission to telnet to outside world → If not, terminated by gateway, if yes, prompts user for connection details → sets up telnet sesh between gateway and external host → relays data from user to extrnl host
- App gateway acts as both telnet client and server
- Intrnl networks have multiple app gateways for telnet, Http, email, ftp etc

Limitations of Firewalls/Gateways

- IP Spoofing: router can't know if data really comes from claimed src
- Filters often use all or nth policy for UDP
- Many highly protected sites still suffer from attacks
- Different app. gateway needed for each app, performance penalty, client software must know how to contact gateway and how to tell app gateway what external server to connect to

<u>Physical Layer</u>

- Moves data using electromagnetic signals

Use digital transmission / analog transmission to represent 0/1

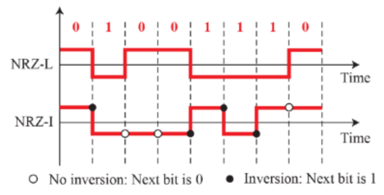
- Wi-Fi transmits analog, Ethernet digital
- Ethernet, RFID, NFC use Manchester
- USB uses NRZ-I
- Wi-Fi uses PSK, QPSK, 16-QAM, 64-QAM

Digital Transmission

- Encode 0s and 1s with voltages

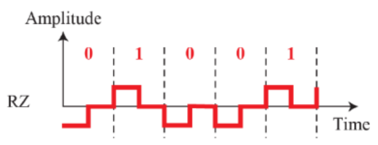
NRZ (Non-Return-to-Zero)

- NRZ-L (value of bit = abs voltage level), NRZ-I (invert voltage if bit 1 is encountered)



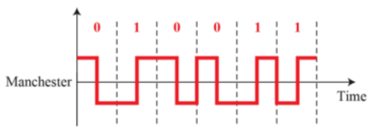
RZ (Return-to-Zero)

- 3 voltage levels, always return voltage to zero halfway through bit interval
- Detect transitions to zero, extract clock from signal: self-clocking



Manchester

- Inverts signal in middle of bit
- -ve to +ve transition represents 1, +ve to -ve transition represents 0



Analog Signal

- Most basic: Sin wave

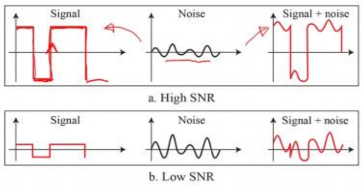
$$\underbrace{A}_{\text{Peak amplitude}}\sin\,(2\pi \underbrace{f}_{\text{frequency}} t + \underbrace{\phi}_{\text{phase}})$$

Channel Bandwidth

- Transmission channel only allows signals in a certain frequency to pass through
- Difference between highest and lowest freq that can pass through a channel is known as **bandwidth**.

Signal to Noise Ratio (SNR)

- Noise that distorts the signal, SNR measures the strength of signal over noise

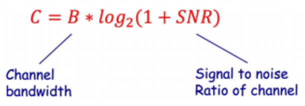


- Receiver comparator – a circuit that takes in reference and actual voltage, compare signals, see actual voltage fall under high/low depending on references' thresholds, above = 1, below = 0

→ Should never be in between the thresholds

Shannon Channel Capacity

- Theoretical max bit rate of noisy channel


$C = B * log_2(1 + SNR)$

Analog Transmission

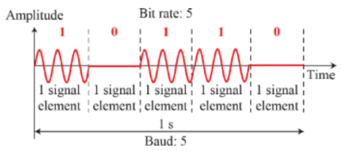
Modem = modulator + demodulator

→ Converts digital signal to analog signal (mod) and other way round (dem)

- Analog Encoding: Amplitude Shift Key (ASK), Frequency Shift Key (FSK), Phase Shift Key (PSK) to encode 0s and 1s

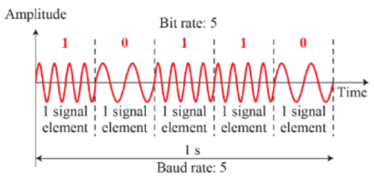
ASK

-Different amplitude levels to encode 0/1s, susceptible to noise



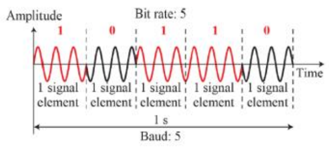
FSK

- Amplitude and phase constant, FSK is limited by bandwidth of channel



PSK

- Phase 0 degree and 180 degree

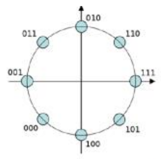


QPSK Constellation Diagram

- Send more than 1 bit during a single time period?

→ For example, split phase into 4 phases. 4 different phase = 2 bits

- Use constellation diagram to represent

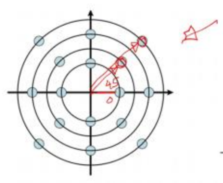


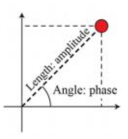
Now every signal tells the receiver 3 bits of data!

- 8-PSK Constellation Diagram

QAM

- Quadrature Amplitude Modulation – Combines ASK and PSK
- Baud rate – number of signal units/s
- Bit rate – number of bits received/s
- 32-QAM means that there are 32 different signal elements, each representing 5 bits





- Example of 16 bit QAM