# Announcements

- Project #5 extended until Dec. 10
- Reading: 7.6
- No Class or office hours on Tuesday
- Project Demos are on Wed
- Extra Office hours next week:
  - Th: 10-11
  - F: 11-12

# Design Issues In Layers

- Rules for data transmission (Protocol)
  - full vs. half duplex
  - error control (detection, correction, etc.)
  - flow control (rate matching, overuse of shared resources)
  - message order (do things arrive in the same order as sent?)
- Abstractions for communications
  - end points for communication
    - switches, nodes, processes, threads in a process
    - how are these end points named (addresses)?
  - service providers and service users
- Service Primitives
  - operations performed by a layer
    - events and their actions
  - request, indication, response, confirm

# Protocols are divided into layers

- ISO - seven layer reference model
  – Application
  – Presentation
  – Session
  – Transport
  – Network
  – Link
  – Physical
- TCP/IP - four layer model
  – link
  – network
  – transport/session/presentation
  – application
- Old Saying: If you know what you are doing, four layers is enough; if you don’t seven won’t help.

# Error Correcting Codes

- Idea: add redundant information to permit recovery
  - this is the dual of data compression (remove redundancy)
- Hamming distance (n)
  - number of bit positions that differ in two words
  - key idea: need n single bit errors to go from one word to the other
  - to detect d errors, need a hamming distance of d+1 from **any other valid word.**
  - to recover d errors, need a hamming distance of 2d + 1
    - any error of d bits is still closer to correct word
- Parity bit
  - ensure that every packet has an odd (or even) # of 1's
  - permits detection of one 1 bit error

# Error Codes (cont.)

- Error Recovery
  - Given m bits of data and r bits of error code
  - Want to correct any one bit error
  - There are n words one bit from each valid message
    - so need n+1 words for each valid message
    - thus $(n + 1)\, 2^m <= 2^n$
    - but $n = m + r$ so $(m + r + 1) <= 2^r$
- Hamming Code
  - recovers from any one bit error
  - number bits from left (starting at 1)
    - power of two bits are parity
    - rest contain data
  - bit is checked by all parity bits in its sum of power expansion
    - bit 11 is used to compute parity bits 1, 2, and 8

# CRC's

- several G's are standardized
  - CRC-12 = $x^{12} + x^{11} + x^3 + x^2 + x + 1$
  - CRC-16 = $x^{16} + x^{15} + x^2 + 1$
  - CRC-CCITT = $x^{16} + x^{12} + x^5 + 1$
- 16 bit CRC will catch
  - all single and double bit errors
  - all errors with an odd number of bits
  - all burst errors of length less than 16

# Sliding Window Protocol

- Need to
  - have multiple outstanding packets
  - limit total number of outstanding packets
  - permit re-transmissions to occur
- Sliding Window
  - permit at most N outstanding packets
  - when packet is ACK'd advance window to first non-ACK'd pkt
- Retransmission
  - Go-back N
    - when a packet is lost, restart from that packet
    - provides in-order delivery, but wastes bandwidth
  - Selective Retransmission
    - use timeout to re-sent lost packet
    - use NACK as a **hint** that something was lost

# Connection vs. Connectionless

- Two possible designs for network layer
  - connection oriented service (ATM)
    - based on experience of telcos
  - connectionless service (IP)
    - based on packet switching (ARPANET)
- Connectionless
  - transport datagrams from source to destination
    - end-point addresses in every datagram
  - less complex network layer, more complex transport
- Connection oriented
  - also called virtual circuits
  - establish an end-to-end connection with network state
    - can use VCI (global or next hop) in each packet

# Routing: Goals

- Correctness
  - packets get where they are supposed
- Simplicity
  - easy to implement correctly
  - possible to make routing choices fast (or updates easy)
- Robustness
  - failures in the network still permit communication
- Stability
  - small changes in link availability results in a small change in the routing information
- Fairness
  - each host, VC, or datagram has the same chance
- Optimality
  - best possible route
  - best utilization of bandwidth

# Distance Vector Routing

- Also known as Bellman-Ford or Ford-Fulkerson
  - original ARPANET routing algorithm
  - early versions of IPX and DECnet used it too
- Each router keeps a table of tuples about all other routers
  - outbound link to use to that router
  - metric (hops, etc.) to that router
  - routers also must know “distance” to each neighbor
- Every T sec., each router sends it table to its neighbors
  - each router then updates its table based on the new info
- Problems:
  - fast response to good news
  - slow response to bad news
    - takes max hops rounds to learn of a downed host
    - known as count-to-infinity problem

# Link State Routing

- Used on the ARPANET after 1979
- Each Router:
  - computes metric to neighbors and sends to **every** other router
  - each router computes the shortest path based on received data
- Needs to estimate time to neighbor
  - best approach is send an **ECHO** packet and time response
- Distributing Info to other routers
  - each router may have a different view of the topology
  - simple idea: use flooding
  - refinements
    - use age sequence number to damp old packets
    - use acks to permit reliable delivery of routing info

# Congestion

- Too much traffic can destroy performance
  - goal is to permit the network to operate near link capacity
  - can reach a knee in the packets sent vs. delivered curve
- Sources
  - all traffic is destined for a single out link
    - backup in traffic consumes buffers
    - other (cross traffic) will not get through due to lack of buffers
  - slow router CPU
    - can't service all requests at link speed
      - links still backup
- Often feeds on itself
  - queuing delays can cause packets to timeout
    - introduces more traffic due to re-transmissions

# Congestion Control

- Two possible approaches
  - open loop: prevent congestion from every happening
    - tends to be conservative and result in under utilizaion
  - closed loop: detect and correct
    - some congestion will still occur until it is corrected
- Open loop
  - request resources before using them
  - global (or regional) resource allocation
    - responds yes or no to each request for service
- Closed loop
  - monitor network to detect congestion
  - pass information back to location where action can be taken
  - adjust system operation to correct the problem

# Responding to Congestion

- Add more resources
  - dialup network: start making additional connections
  - SMDS: request additional bandwidth from provider
  - split traffic: use all routes not just optimal
- Decrease load
  - deny service to some users: based on priorities
  - degrade service to some or all users
  - require users to schedule their traffic

# Internetworking

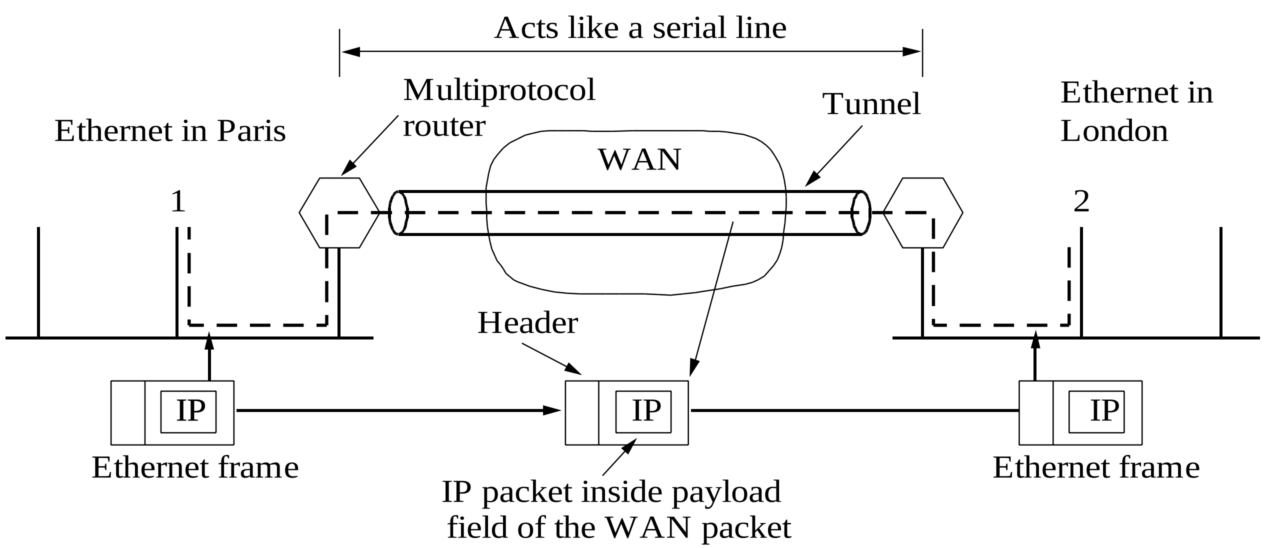
- Goal: seamless operation over multiple subnets
  - could be two similar LANs
  - link WANs to LANS
  - link two different LANs together
- Issues:
  - packet size limits (different networks may have different limits)
  - quality of service (is it provided, how is it defined)
  - congestion control
  - connection vs. connectionless networks
- Possible at many levels
  - physical layer: repeaters
  - link layer: bridges - regenerate traffic, some filtering
  - network: routers - route packets between networks
  - transport: gateway byte streams
  - application: gateway email between two different systems

# Firewalls

- A way to limit information flow
  - selective forwarding of information based on **policy**
  - policy: rules about what should be permitted
  - mechanism: way to enforce policy
- Can be implemented at many levels
  - at higher layers have more information
  - at lower layers can share filtering between multiple higher level entities
- Possible Layers
  - link layer: filter based on MAC address
  - network layer: filter based on source/destination, transport
  - transport: filter based on service (e.g. port number)
  - application: filter based on user name in email, based on content

# Tunneling

- Problem
  - Source and Destination are compatible
  - something in the middle is not compatible
- Solution: Tunnel though the middle
  - only multi-protocol routers need to understand conversion
  - possible to tunnel through almost anything
    - can tunnel IP through IP (for mobile computing perhaps)

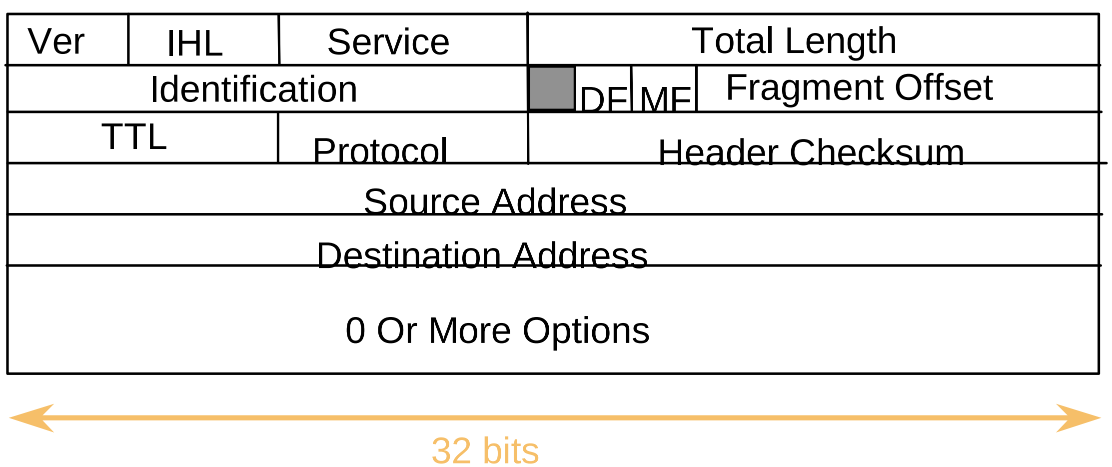


# Fragmentation

- Sometimes need to split packets into smaller units
  - limits of the hardware being used
  - operating system buffer constraints
  - protocol limits (max permitted packet is x bytes)
  - reduce channel occupancy (head of link blocking)
- Fragmentation
  - where to split it into smaller packets
    - source (requires end-to-end information on max size)
    - when it reaches boundary
  - how to represent split packets
    - need to encode fragment offset
- Reassembly
  - where to re-combine packets
    - destination (may result in poor performance)
    - at the gateway to the subnet that supports the full size

# The IP Protocol

- IP Header
  - source, destination address, total length
  - version, ihl (header length in 32-bit words), ttl, protocol
  - fragmentation support: identification, df, mf, frag. offset
- Options
  - variable length
  - defined options
    - loose source routing
    - timestamp
    - record path



# Transport Layer

- Goal: provide error free end-to-end delivery of data
  - provide in-order delivery over unreliable network layer
- Issues:
  - checking packet integrity
  - re-transmission of lost of corrupt packets
  - connection establishment and management
  - addresses
    - need to define a host plus process
    - typical abstraction is <host, port>
  - byte vs. packet transport service
    - byte service
      - bytes are in order, but packet boundaries are lost
      - used by TCP
    - packet service
      - preserve packet boundaries

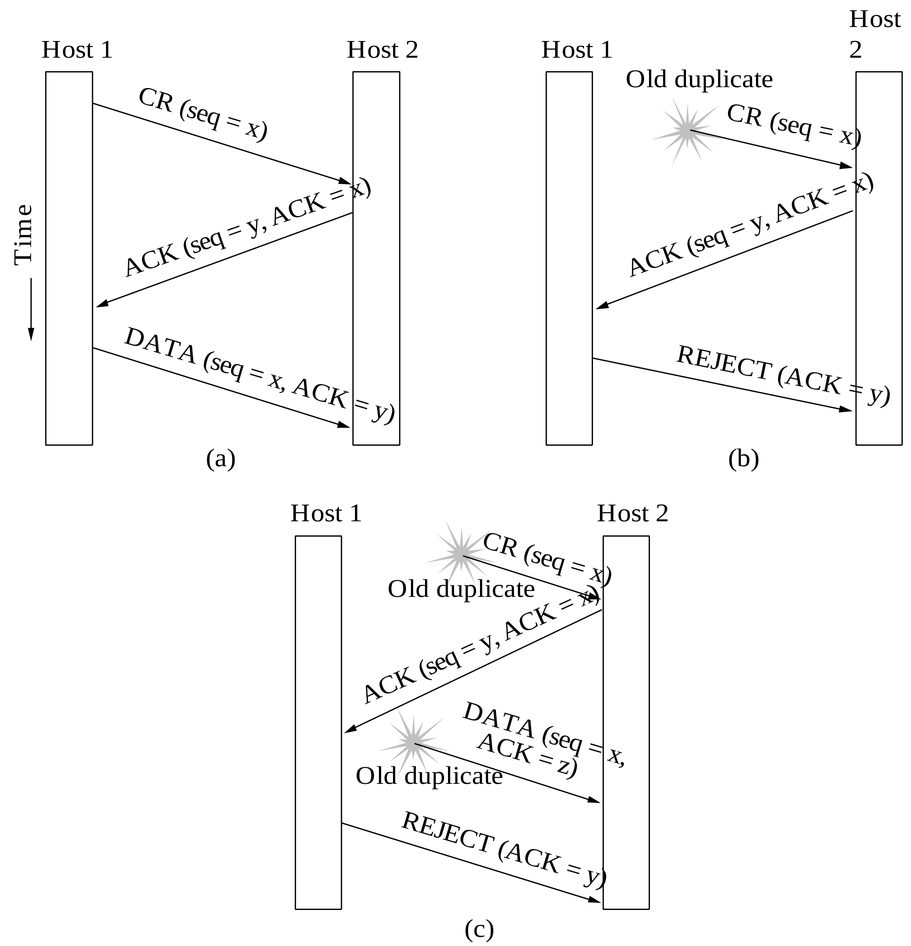
# Duplicate Packets

- Issue: packets can be lost or duplicated
  - need to detect duplicates
  - need to re-send lost packets
    - but how do we know they are not just delayed?
- Solution 1
  - use a sequence number
    - each new packet uses a new sequence number
    - can detect arrival of stale packets
  - problem: when node crashes, sequence number resets
- Solution 2
  - use a clock for the sequence number
    - clocks don’t reset on reboot, so we never lose sequence #
  - use a max lifetime for a packet
    - permits clocks to roll over
  - can get into **forbidden** region

# Three-way Handshake

- Use different sequence number spaces for each direction
- Three messages used
  - Connection Request
    - send initial sequence number from caller to callee
  - Connection Request Acknowledgment
    - send ACK of initial sequence number from caller to callee
    - send initial sequence number from callee to caller
  - First Data TPDU
    - send ACK of initial sequence number from callee to caller
- Each Side Selects an initial number
  - it knows that the number is not currently valid
    - uses time of day
    - limits number of connects per unit time, but not data!

# Example of Three-way Handshake




From: *Computer Networks*, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.

# Closing a Connection

- To prevent data loss,
  - both sides must agree they are done
- Problem: how to agree
  - possible that “I am done” messages will get lost
  - possible that “I ACK you are done” messages will get lost
- Solution:
  - initiator sends Disconnect Request, start DR timer
  - when initiated party receives DR, send DR and start DR timer
  - when initiator gets DR back, send ACK and release connection
  - when initiated gets ACK, release connection
  - if initiator times out, send new DR
  - if initiated times out, release connection

# TCP Protocol

- TSAPs
  - Use <host, port> combination
  - Well known ports provide services
    - first 256 ports
    - SMTP 25, Telnet 23, Ftp 21, HTTP 80
- Provides a **byte** stream
  - this is **not** a message stream
  - a message (single call to send) may be split, merged, etc.
- Urgent Data field
  - provides cut through delvery *within* a trasport connection
  - used to send breaks or other high priority info

# TCP Connection Management

- Three-way Handshake
- Initial Sequence Numbers
  - Use a 4 micro-second clock
  - hosts must wait T (120 seconds) before a reboot
- Connection Closure
  - Each side uses a FIN and FIN_ACK message
  - A FIN times out after 2 T (240 seconds)
  - Keep alives used to timeout half dead connections

# TCP Flow Control

- Use Variable Sized Sliding Window
  - ACK indicates start of window
  - Window size indicates current size of window
- Receiver can send a window of 0
  - indicates that it want to pause connection
  - urgent data need not follow this request
- Window size of 16 bits is too small
  - 64K Bytes
  - only a small fraction of the in-flight bytes when
    - bandwidth is high
    - delay is high
  - solution: window shift option:
    - bit shift window up to 16 bits
    - permits up to $2^{32}$ byte windows
    - reduces window granularity

# TCP Congestion Control

- Detecting Congestion
  - In general it is difficult
  - But, consider why a packet might be dropped
    - link error - but links are very reliable now
    - buffer overflow --> congestion
  - Use re-transmission timeouts as an estimate of congestion
- Dealing with Congestion
  - add a second window (congestion window)
    - limit transmissions to min(recv window, congestion window)
  - start with congestion window = max segment window
    - initial max segment is one kilo-byte
    - on a ACK without a timeout

      if window < threshold, increment by one max segment

      otherwise increment by initial max segment
  - on timeout
    - cut threshold in half
    - set window size to initial max segment

# Max Data Rates Over A Channel

- Shannon/Nyquist limit
  - max data rate is $2H\log_2 V$ bits/sec
    - H - bandwith of the channel
    - V - number of levels used to encode data
  - for example, a noiseless 3khz channel can carry
    - 6,000 bps for binary traffic but
    - 12,000 pbs for quadary (4 level) traffic
- What about noise?
  - noise is measured as the ratio of signal to noise power
  - normally measured in db or $10 \log_{10}(S/N)$
  - Shannon limit:
    - max bits/sec = $H \log_2(1+S/N)$
    - 3khz, 30dB channel limited to 30,0000 bps

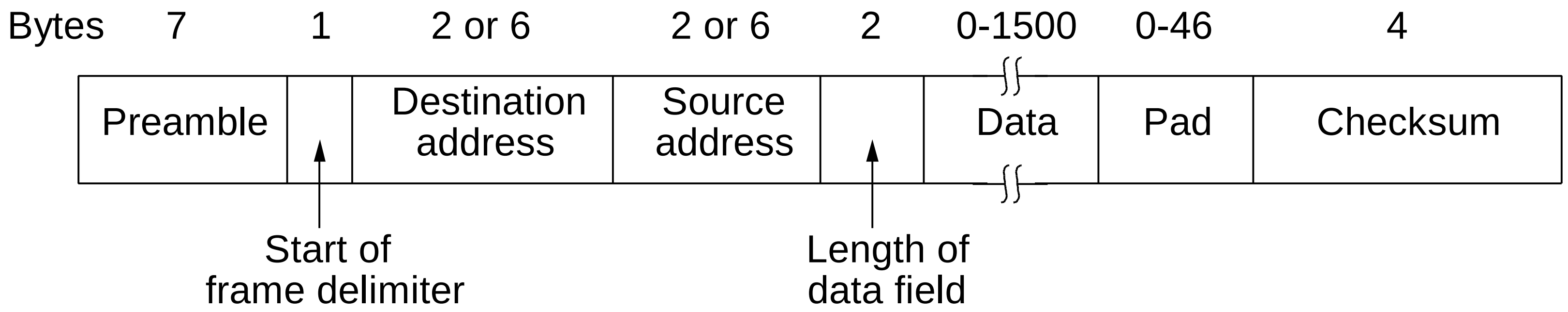
# Carrier Sense Multiple Access

- look before you leap!
  - don’t send if someone else is sending
- collisions are still possible
  - propagation delay induces uncertainty into sensing
  - possible two hosts both start sending at the same time
- persistence: when to send after detecting channel in use
  - 1-persistent
    - as soon as the channel is free, starting sending
  - nonpersistent CSMA
    - if channel is sensed busy, wait a random time and try again
  - p-persistent CSMA
    - if slot is idle send with probability p, else wait for next idle slot

# Collision Detection

- If a sender senses a collision
  - stop sending at once
  - apply random backoff
- “contention” period
  - after contention period, there will be no collision
  - send for for $2\tau$ (max propagation delay)
    - need $2\tau$ since might be a collision at far end at $\tau - \varepsilon$

# Ethernet Frame Format




From: *Computer Networks*, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.

- Preamble used to sync clock
- Addresses
  - 48 bits
  - if it starts with a 0 it is globally unique (assigned by IEEE)
  - if it starts with a 1 it is locally unique
- Length
  - 0 to 1500 bytes
  - **min** length is 46 bytes
    - ensures frame reaches end of cable before end of frame is sent
- Checksum
  - 32 bit CRC to detect garbled data at link level

# Where to Provide Security?

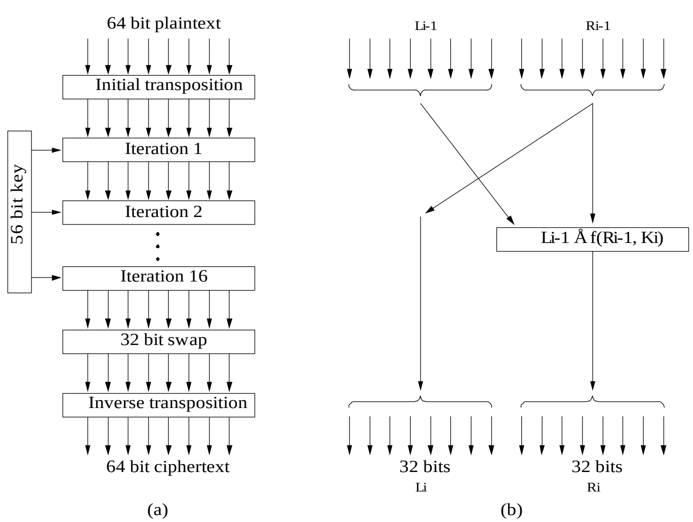
- Short Answers: at all levels
- physical:
  - wrap gas or tripwires around cable
- link:
  - encryption protects the wire but not the router
- network:
  - firewalls filter packets
  - end-to-end encryption
- session/presentation:
  - “secure” socket layer
- application:
  - PGP signed messages
  - application specific authentication

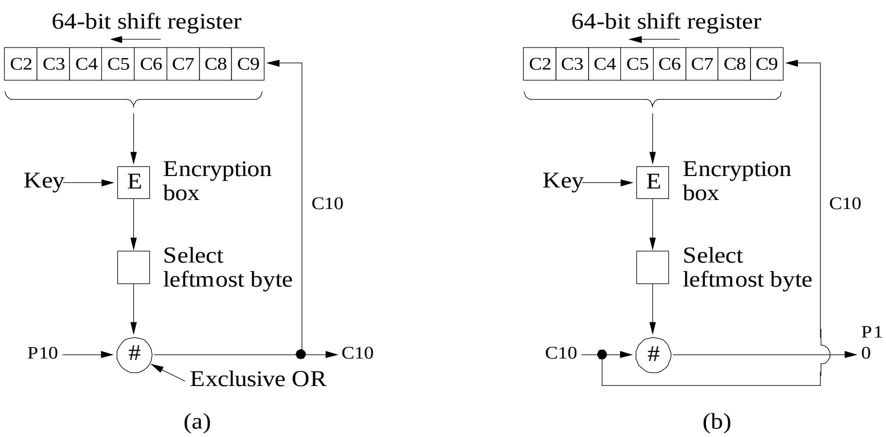
# One Time Pad

- Key Idea: randomness in key
- Create a random string as long as the message
  - each party has the pad
  - xor each bit of the message with the a bit of the key
- Almost impossible to break
- Some practical problems
  - need to ensure key is not captured
  - a one bit drop will corrupt the rest of the message
- Pseudo-random is not good enough
  - Japanese JN-25 during WWII was pseudo random onetime pad

# DES

- Block cipher: uses 56 bit keys, 64 bits of data
- Uses 16 stages of substitution
- Variations
  - cipher block chaining: xor output of block n with into block n+1
  - cipher feedback mode: use 64bit shift register
    - can produce one byte at a time




From: *Computer Networks*, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.

# Public Key Encryption

- Split into public and private keys
  - public key used to encrypt messages
    - publish this key widely
  - private key used to decrypt messages
    - keep this key a secret
- RSA
  - algorithm for computing public/private key pairs
  - based on problems involved in factoring large primes
  - for an n bit message P, $C = (P^e \bmod n)$, and $P = (C^d \bmod n)$
- Other Public Key Algorithms
  - knapsack
    - given a large collection of objects with different weights
    - public key is the total weight of a subset of the objects
    - private key is the list of objects

# Authentication

- Identify the parties that wish to communicate
- Create a session key
  - a random string
  - used only for one session
- Authentication based on Shared Keys
  - each party already shares a private key
    - exchanged via an out of band transmission
  - challenge-response
    - send a random string
    - response is the encryption of the random string with the shared key

# Message Digests

- Goal: Send Signed Plain text
  - can use slow cryptography on signature since its short
- Need:
  - Given P, easy to compute MD(P)
  - Given MD(P), impossible to find P
  - no P and P’ exist such that MD(P) = MD(P’)
    - use hash functions that produce >= 128 bit digest
- Operation
  - A sends P, $D_a$(MD(P))
- Digest Functions
  - MD5
    - produces 128 bit digest
  - SHS
    - NSA/NIST effort
    - produces 160 bit output

# Naming Hosts In the Internet

- Originally used a single file
  - all hosts had line line with name and IP Address
- Domain Naming System (DNS)
  - introduced in 1986
  - tree based structure to names
  - Names
    - full name must be less than 256 characters
    - each part can be up to 64 characters
    - are case insensitive
  - administration of subtrees can be deligated
    - each administrative region is called a zone

# Email

- Dominate Email is RFC821/822
  - X.400 and Lotus notes are also rans for standards
- Basic components
  - message: the actual thing sent
  - mailbox: place where email is stored (may be a file or a directory)
    - identified by a unique name
    - user@dnhost is the standard format
  - transfer agent: something that sends email
    - usually speaks SMTP
    - under UNIX is a program called sendmail
  - user agent
    - program for reading and generating mail
    - can be remote:  use POP, IMAP, or DMSP to talk to mailbox
  - alias
    - a virtual mailbox that maps to one or more real mailboxes
      - may also be a program to handle the inbound mail

# Message Envelop Format

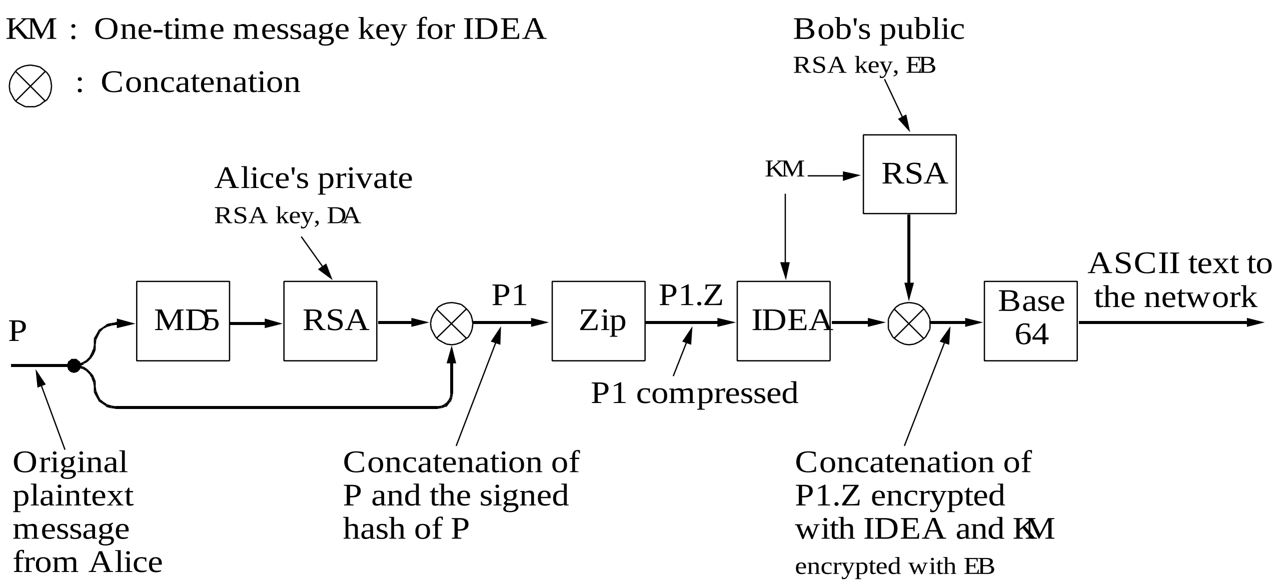
- Information associated with mail delivery
- Destination:
  - To: email address of primary recipient
  - Cc: email address of secondary recipients
  - Bcc: address for blind carbon copies
- Origination
  - From: person who created message
  - Sender: email address of actual sender
- In transit
  - Received: added by each MTA along the way
  - Return-Path: added by destination
- Misc Fields
  - Info: Date, Subject, Keyword
  - Handling: Message-id, Reply-To In-Reply-To, References

# Message Body

- Under RFC822
  - raw ascii text with no semantic meaning
- MIME: Multipurpose Internet Mail Extension
  - provides an interface to send non-ascii text in mail
    - envelop not changed, so only user agents need to be modified
  - supports multiple languages
  - supports multi-media and file attachments
  - headers:
    - MIME-Version
    - Content-Description: human readable description
    - Content-Id: unique id for this part of the message
    - Content-Transfer-Encoding:
      - text: ascii, and 8bit characters
      - binary: may not get there since it is a non-conforming body
      - base64: 26 binary bits-> 4 ascii characters
      - quoted printable: only use base64 for “special” characters
    - Content-Type: what is this

# Pretty Good Privacy: PGP

- Developed by a single person
  - uses RSA, IDEA, and MD5
- Provides: privacy, compression, and digital signatures
- Has a collection of key servers for public key registration
- Uses three different key lengths (384, 512, and 1024 bits)




From: *Computer Networks*, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.

# News

- Large Collection of newsgroups
  - currently a hierarchalnamespace (used to be rather flat)
  - can be moderated: must be approved before being posted
- Messages
  - have a unique id
  - are associated with one or more newsgroups
  - contain a superset of RFC822 fields
- Transport of news
  - a site a list of one or more sites it gets is newsfeed from
    - a site periodically polls its newsfeeds for news
    - newsfeeds can also push new news out
  - UUCP: Unix-to-Unix CoPy
    - historical path using dialup modems
  - NNTP: Net News Transfer Protocol (TCPport 119)

# WWW (cont.)

- HyperText Markup Language
  - based on SGML
    - font changes, text placement
    - includes support for images
  - supports references to other document (links)
  - supports alternatives to display if browsers can't support a format
- HyperText Transport Protocol
  - used to move HTML from server to client
  - Basic protocol
    - GET: get a page
    - PUT: store a page
    - POST: append to a page

# Interactive Web Pages

- Forms
  - HTML can describe fields which permit users to enter data
    - textboxs, checkboxes, lists, etc.
  - contain an action
    - a URL to POST the completed form
- Common Gateway Interface (CGI)
  - Servers can be told that some pages are really programs
    - could be executable binaries, perl programs, etc.
  - An attempt to POST to a CGI script runs it
    - the form data is taken as input
    - CGI script returns an HTML page as output
      - output can be a function of the input
  - common examples:
    - perl scripts
    - interfaces to database systems