Announcements

- Project #4 Due this week
- Midterm #2 is Tuesday
- No Office hours next week
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 $P$, $C = (P^e \mod n)$, and $P = (C^d \mod n)$

- **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
Authentication Example

1. A
2. RB
3. KAB (RB)
4. RA
5. KAB (RA)

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

- Only requires three messages
- But it is subject to a “man in the middle attack”
Attacking the Simplified Protocol

- T can get B to respond to its own challenge
- T opens a second session with B
  - it issues B’s session 1 challenge back to B in session 2

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

- **Problem with Private Key Authentication**
  - Need to establish key
  - for n people need $n^2$ keys
  - keys must be established via *out-of-band* communication

- **Solution: Key Distribution Center (KDC)**
  - trusted party used to assist in authentication
  - each party establishes a private key with the center

- **have KDC trans-code a message with a session key**
  - A sends to KDC $<A, K_A(B, K_s)>$
  - KDC sends to B $<K_b(A, K_s)>$
  - open to replay attack
    - T logs KDC to B message and all traffic using $K_s$
Needham-Schroeder Authentication

- $R_A$, $R_{A2}$ and $R_B$ random strings
  - used to prevent replay attacks
- If $T$ ever gets $K_S$ can establish contact with $B$
  - can prevent this with a slight variation of the algorithm
- Used in Kerberos Authentication System
Authentication using Public Keys

- Assume each party knows the other’s public key
  \[ E_b(A, R_a) \]
  \[ E_a(R_a, R_b, K_s) \]
  \[ K_s(R_b) \]

- How To learn others Public Key?
  - use a public key server
    - but how do we trust the public key server?
    - have a public key for the public key server
    - possible to have man-in-the-middle attacks
  - interlock protocol
    - only send half the message (odd bits) at a time
    - prevents man-in-the-middle attacks
    - still possible to spoof service
Digital Signatures

- **Want to “sign” a message such that:**
  - receiver can verify the identity of the sender
  - sender cannot repudiate the contents of the message
  - receiver cannot forge a message
- **Central authority (BB)**
  - A sends BB A, KA(B, RA, t, P)
  - BB sends B KB(A, RA, t, P, Kbb(A, t, P))
  - everyone trusts BB
    - BB can be called on to decrypt messages to verify them
    - BB need not store all message that it validates
  - t - timestamp used to prevent replay attacks
- **Public Key**
  - need E(D(P)) = P and D(E(P)) = P
  - A sends B EB(Da(P))
    - B keeps Da(P) and third party can use Ea to verify it’s from A

- **Used to prevent replay attacks when t has not changed yet (i.e. slow clock)**
Digital Signatures (cont.)

- **Problems**
  - Repudiation
    - inform police that the key was stolen
    - claim the “bad guy” sent the message
  - Key Changes
    - need to keep records of when keys were in use

- **Standards**
  - RSA Algorithm
    - popular with many commercial systems
  - El Gamal
    - NSA/NIST Standard
    - too new, and private to have trust
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 \( \geq 128 \) 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