Key Management

CSCI 454/554

- Diffie-Hellman Key Exchange Protocol
- Public-key encryption to distribute secret keys
- Distribution of public keys
Diffie-Hellman Key Exchange

- first public-key type scheme proposed
- by Diffie & Hellman in 1976 along with the exposition of public key concepts
  - note: now know that James Ellis (UK CESG) secretly proposed the concept in 1970
- is a practical method for public exchange of a secret key
- used in a number of commercial products

Diffie-Hellman Key Exchange

- a public-key distribution scheme
  - cannot be used to exchange an arbitrary message
  - rather it can establish a common key
  - known only to the two participants
- based on exponentiation in a finite (Galois) field (modulo a prime or a polynomial) - easy
- security relies on the difficulty of computing discrete logarithms (similar to factoring) – hard
Diffie-Hellman Setup

- all users agree on global parameters:
  - large prime integer $q$
  - $\alpha$ a primitive root mod $q$
- each user generates their number pair
  - chooses a private number: $x_A < q$
  - compute their public number: $y_A = \alpha^{x_A} \mod q$
- each user makes public that number $y_A$

Diffie-Hellman Key Exchange

- Shared session key for users A & B is $K_{AB}$:
  \[ K_{AB} = \alpha^{x_A \cdot x_B} \mod q \]
  \[ = y_A^{x_B} \mod q \] (which B can compute)
  \[ = y_B^{x_A} \mod q \] (which A can compute)
- $K_{AB}$ is used as session key in symmetric-key encryption scheme between Alice and Bob
- Attacker must solve discrete log
Diffie-Hellman Example

- users Alice & Bob who wish to swap keys:
- agree on prime $q=353$ and $\alpha=3$
- select random private numbers:
  - A chooses $x_A=97$, B chooses $x_B=233$
- compute public numbers:
  - $y_A = \alpha^{x_A} \mod q = 3^{97} \mod 353 = 40$ (Alice)
  - $y_B = \alpha^{x_B} \mod q = 3^{233} \mod 353 = 248$ (Bob)
- compute shared session key as:
  - $K_{AB} = y_B^{x_A} \mod q = 248^{97} \mod 353 = 160$ (Alice)
  - $K_{AB} = y_A^{x_B} \mod q = 40^{233} \mod 353 = 160$ (Bob)

Public-Key Distribution of Secret Keys

- can use for secrecy or authentication
- but public-key algorithms are slow
- so usually want to use secret-key encryption to protect message contents
- hence need a session key
- have several alternatives for negotiating a suitable session
Simple Secret Key Distribution

- proposed by Merkle in 1979
  - A generates a new temporary public key pair
  - A sends B the public key and their identity
  - B generates a session key K sends it to A encrypted using the supplied public key
  - A decrypts the session key and both use
- problem is that an opponent can intercept and impersonate both halves of protocol

Public-Key Distribution of Secret Keys

1. $E_{X_A}[Y_A || ID_A]$
2. $E_{X_B}[Y_A || Y_B]$
3. $E_{X_B}[Y_B]$
4. $E_{X_B}[E_{X_A}[K]]$
Distribution of Public Keys

- can be considered as using one of:
  - Public announcement
  - Publicly available directory
  - Public-key authority
  - Public-key certificates

Public Announcement

- users distribute public keys to recipients or broadcast to community at large
  - eg. append PGP keys to email messages or post to news groups or email list
- major weakness is forgery
  - anyone can create a key claiming to be someone else and broadcast it
  - until forgery is discovered can masquerade as claimed user
Publicly Available Directory

- can obtain greater security by registering keys with a public directory
- directory must be trusted with properties:
  - contains \{name,public-key\} entries
  - participants register securely with directory
  - participants can replace key at any time
  - directory is periodically published
  - directory can be accessed electronically
- still vulnerable to tampering or forgery

Public-Key Authority

- improve security by tightening control over distribution of keys from directory
- has properties of directory
- and requires users to know public key for the directory
- then users interact with directory to obtain any desired public key securely
  - does require on-line access to directory when keys are needed
Public-Key Certificates

- certificates allow key exchange without real-time access to public-key authority
- a certificate binds **identity** to **public key**
  - usually with other info such as period of validity, rights of use etc
- with all contents **signed** by a trusted Public-Key or Certificate Authority (CA)
- can be verified by anyone who knows the public-key authorities public-key

[Diagram of public-key certificate flow]
Summary

- have considered:
  - Diffie-Hellman key exchange
  - public-key distribution of secret keys
  - distribution of public keys