COM 5335
Lecture 7
Other Public-Key Cryptosystems

Scott CH Huang

Outline

- Rabin public-key encryption algorithm
- ElGamal public-key encryption algorithm
- Diffie-Hellman key exchange protocol

Contemporary Public-Key Cryptosystems

- Based on the Factorization Problem:
  - RSA, Rabin
- Based on the Discrete Logarithm Problem:
  - ElGamal, Elliptic Curve, DSA (signature scheme only), Diffie-Hellman (key exchange & encryption)

Rabin Public-Key Cryptosystem

- Rabin encryption is an extremely fast operation as it only involves a single modular squaring. By comparison with RSA.
- Rabin decryption is slower than encryption but is comparable in speed to RSA decryption

Rabin Key Generation

- Generate 2 large random numbers primes p an q, each with the same size
- Compute $N=pq$
- The public key is N and the private key is p and q

Rabin Encryption

- Rabin Encryption is nothing more than doing a SQUARE operation as follows.
  - Represent the message as an integer m in the range $\{0,1,\ldots.,N-1\}$
  - Ciphertext is $c \equiv m^2 \bmod N$

Rabin Decryption

- **Rabin Decryption is a SQROOT operation**
  - Find the square roots *m1,m2,m3* and *m4* of *c* mod *N*
  - The message sent was either *m1,m2,m3* or *m4.*

The Legendre Symbol

- The Legendre symbol is a useful tool for keeping track of whether or not an integer has a sqrt mod a prime number *p*.
- Let *p* be an odd prime and *a* an integer. The Legendre symbol (*a*/*p*) is defined as follows.

$$(a/p) = \begin{cases} 0 & \text{, if } p|a \\ 1 & \text{, if } a \text{ has square root(s).} \\ -1 & \text{, otherwise.} \end{cases}$$

Facts of the Legendre Symbol

- $(a/p) \equiv a^{(p-1)/2} \bmod p$
- $(ab/p)=(a/p)(b/p)$
- (Law of quadratic reciprocity) If q is an odd prime distinct from p, then

$$(p/q)=(q/p)(-1)^{(p-1)(q-1)/4}$$

Find SQROOT in Z_p

- INPUT: an odd prime *p* & an odd integer *a* s.t. $0<a<p$
- OUTPUT: two square roots of *a* mod *p*

1. If $(a/p)=-1$, stop & return
2. Select *b* ($0<b<p$) with $(b/p)=-1$. Represent $p-1=2^s t$ where *t* is odd.
3. Compute a^{-1} mod *p*
4. $c \leftarrow b^t$ mod *p*, $r \leftarrow a^{(t+1)/2}$ mod *p*
5. For *i* from *1* to *s-1* do
    1. Compute $d \equiv (r^2 a^{-1})^{2^{s-i-1}} \bmod p$
    2. If $d \equiv -1$ mod *p*, set $r \leftarrow rc$ mod *p*
    3. $c \leftarrow c^2$ mod *p*
6. Return $(r,-r)$

Find SQROOT in Z_p where $p \equiv 3 \bmod 4$

- INPUT: an odd prime p where $p \equiv 3 \bmod 4$, and square a s.t. $0 < a < p$
- OUTPUT: two square roots of a mod p

1. Compute $r \equiv a^{(p+1)/4} \bmod p$
2. Return $(r, -r)$

Find SQROOT in Z_p where $p \equiv 5 \bmod 8$

- INPUT: an odd prime p where $p \equiv 5 \bmod 8$, and square a s.t. $0<a<p$
- OUTPUT: two square roots of $a \bmod p$

1. Compute $d \equiv a^{(p-1)/4} \bmod p$
2. If $d \equiv 1 \bmod p$ then compute $r \equiv a^{(p+3)/8} \bmod p$
3. If $d \equiv -1 \bmod p$ then compute $r \equiv 2a(4a)^{(p-5)/8} \bmod p$
4. Return $(r,-r)$

Find SQROOT in Z_n where n=pq (p,q primes)

- INPUT: $n=pq$ & an integer a s.t. $0<a<n$, a has SQROOT(s)
- OUTPUT: four sqrts of a mod p

1. Find the two sqrts (r,$-r$) of a mod p
2. Find the two sqrts (s,$-s$) of a mod q
3. Use extended Euclid's algorithm to find integers c,d s.t. $cp+dq=1$
4. Set $x \equiv rdq+scp$ mod n and $y \equiv rdq-scp$ mod n
5. Return (x,$-x$,y,$-y$)

A Problem Regarding Rabin's Encryption Scheme

- To decrypt a ciphertext, we need to compute the sqrt. However, there are 4 sqrts, how to decide which one is the plaintext???
- **Appropriate coding** is needed to decide which one is the plaintext.
- In practice, we usually take part of the plaintext and append it to the end.

Rabin – An Example

- **Key generation:** Alice chooses the primes p=277, q=331, and computes N=pq=91687. Alice's public key is N=91687 and private key is p=277 and q=331
- **Encryption:** Suppose that the last six bits of the original messages are required to be appended prior to encryption. In order to encrypt the 10-bits message m=1001111001, Bob appends the last six bits of m to obtain 16-bits message.

  m=1001111001111001 which in decimal notation is m=40569, the ciphertext is:

$$C \equiv m^2 \bmod N \equiv 40569^2 \bmod 91687 \equiv 62111$$

Rabin (cont'd)

- **<u>Decryption:</u>** to decrypt C, Alice computes the four sqrts of C mod N
- m1=69954,m2=22033,m3=40569,m45118
  - m1=1000100000010110,
  - m2=101011000010001,
  - **m3=1001111001111001**,
  - m4=110001111010110
- Therefore, m3 is the plaintext.

SQROOT Problem

- SQROOT Problem
  - If $x^2 \equiv a \bmod N$ has a solution for a given composite integer N=pq (p,q primes), find a sqrt of a mod N.
- FACTOR =>? SQROOT
  - Use previous algorithm, we can find sqrt mod p and sqrt mod q
  - Then we use extended Euclid's algorithm to find sqrt mod N

SQROOT Problem

- SQROOT=>? FACTOR
  - Suppose A is an algorithm that solves SQROOT
  - Then we generate x randomly and compute $a \equiv x^2 \bmod N$
  - Apply A to find sqrt y
  - If y=x or –x, try another x and repeat
  - If not, we are done! (why?)

Security of Rabin

- Rabin=SQROOT=Factor
- Provably secure against passive adversary (cf. RSA)
- Susceptible to chosen ciphertext attack similar to RSA
- Many RSA attacks can be applied to Rabin

Finite Cyclic Groups and the Discrete Logarithm Problem

- A finite group G is *cyclic* if it can be represented as powers of some element g in G as follows.
  - $G=\{e,g,g^2,g^3,\ldots g^{n-1}\}$
  - g is called a *generator* of G, and n is called the *order* of G.
- Example: Let p=97. Then Z_{97}^* is a cyclic group of order n=96. A generator of Z_{97}^* is g=5. Since $5^{32} \equiv 35 \pmod{97}$, $\log_5 35 = 32$ in Z_{97}^*.
- Let G be a finite cyclic group of order n. Let g be a generator of G, and let $y \in G$. The discrete logarithm of y to the base g, denoted $\log_g y$, is the unique integer x, $0 \le x \le n-1$, such that $y = g^x$.

Discrete Logarithm Problem

- **DLP in Z_p*** : Given a prime p, a generator g of Z_p^*, and an element $y \in Z_p^*$, find the integer x, $0 \le x \le p-2$, such that $g^x \equiv y \pmod p$.
- The security of many cryptographic techniques depends on the intractability of the discrete logarithm problem.
- Both ElGamal encryption scheme and Diffie-Hellman key exchange are based on DLP in **Z_p***.The Elliptic curve Cryptosystem is based on DLP in general cyclic groups.

ElGamal Encryption Scheme

- ElGamal encryption scheme is an asymmetric key encryption algorithm
- ElGamal encryption is *non-deterministic*, meaning that a single plaintext can be encrypted to many possible ciphertexts

ElGamal Key Generation

- Each entity randomly choose a large prime p and picks a generator $g \in Z_p^*$
- Each entity randomly chooses an exponent x ($x<p$), and computes $y \equiv g^x \pmod{p}$.
- Public key = (p,g,y)
- Private key= x

ElGamal Encryption

- Suppose Bob wants to encrypt a message M (M<p) and send to Alice

1. Bob obtains Alice's public key (p,g,y) and randomly picks an integer r (r<p)
2. Bob computes
   - $A \equiv g^r \bmod p$
   - $B \equiv My^r \bmod p$
3. Ciphertext C = (A, B).

ElGamal Decryption

Alice does the followings

- Computes $K \equiv A^x \bmod p$,
- $M \equiv BK^{-1} \bmod p$

ElGamal - An Example

- Key Generation:
  - p =2357
  - g = 2
  - x = 1751
  - $y \equiv g^x \equiv 2^{1751} \equiv 1185 \pmod{2357}$
- Public key: (p,g,y) = (2357, 2, 1185)
- Private key: x = 1751

ElGamal

- Encryption:
  - say $M = 2035$
  1. Pick a random number $r = 1520$
  2. Computes

     $A = g^r \equiv 2^{1520} \equiv 1430 \pmod{2357}$

     $B = My^r \equiv 2035 * 1185^{1520} \equiv 697 \pmod{2357}$
  - The ciphertext $C = (A, B) = (1430, 697)$
- Decryption:
  1. Computes $K \equiv A^x \equiv 1430^{1751} \equiv 2084 \pmod{2357}$
  2. $M \equiv B K^{-1} \equiv 697 * 2084^{-1} \equiv 2035 \pmod{2357}$

Remarks on ElGamal Encryption Scheme

- ElGamal encryption scheme is non-deterministic
- Randomization is introduced to
  - increase the effective size of the plaintext space

    i.e. one plaintext can map to a large set of possible ciphertexts
  - decrease the effectiveness of chosen-plaintext attack by means of a one-to-many mapping in the encryption process
- Efficiency:
  - encryption requires two exponentiation operations
  - exponentiation operations may be very expensive when implemented on some low-power devices. e.g. low-end PalmPilots, smart cards and sensors.
  - message expansion by two-fold
- Security:
  - depends on the difficulty of solving DLP (more precisely, Computational Diffie-Hellman Problem).

Diffie-Hellman Key Exchange

- A Key Exchange Protocol:
  - provide a secure way for two communicating party to share a symmetric key (so called a session key)
  - This session key is then used to provide privacy and authentication for subsequent message flow.
  - History: problem first posed by Merkle at UC Berkeley, Diffie and Hellman came up with the protocol:

Alice — Bob

$a<p$ — $b<p$

$$\text{Alice} \xrightarrow{g^a \bmod p} \text{Bob}$$

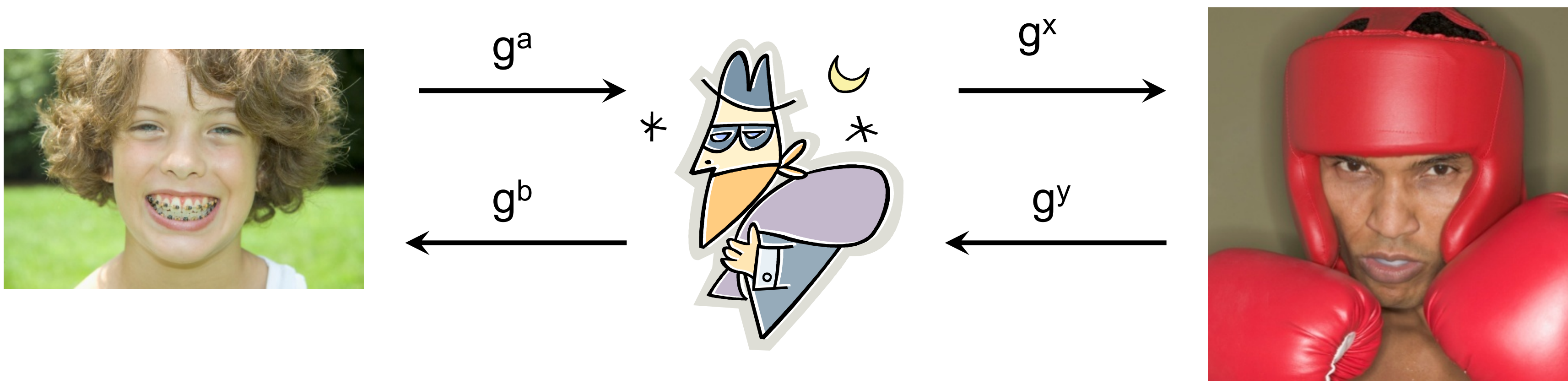
$$\text{Alice} \xleftarrow{g^b \bmod p} \text{Bob}$$

Shared Session Key = $g^{ab} \bmod p$

- W. Diffie, M. E. Hellman, "New directions in Cryptography", IEEE Trans. Information Theory, IT-22, pp. 64-654, Nov 1976.

Man-in-the-Middle Attack

Diffie-Hellman key exchange



Alice computes g^{ab}

Bob computes g^{xy}

Key Management Using Other PKC

- Public-key encryption helps address key distribution problems in two aspects:
  - distribution of public keys
  - use of public-key encryption to distribute secret keys

Distribution of Public Keys

- Can use the following approaches:
  - 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
    - can masquerade as claimed user until forgery is discovered

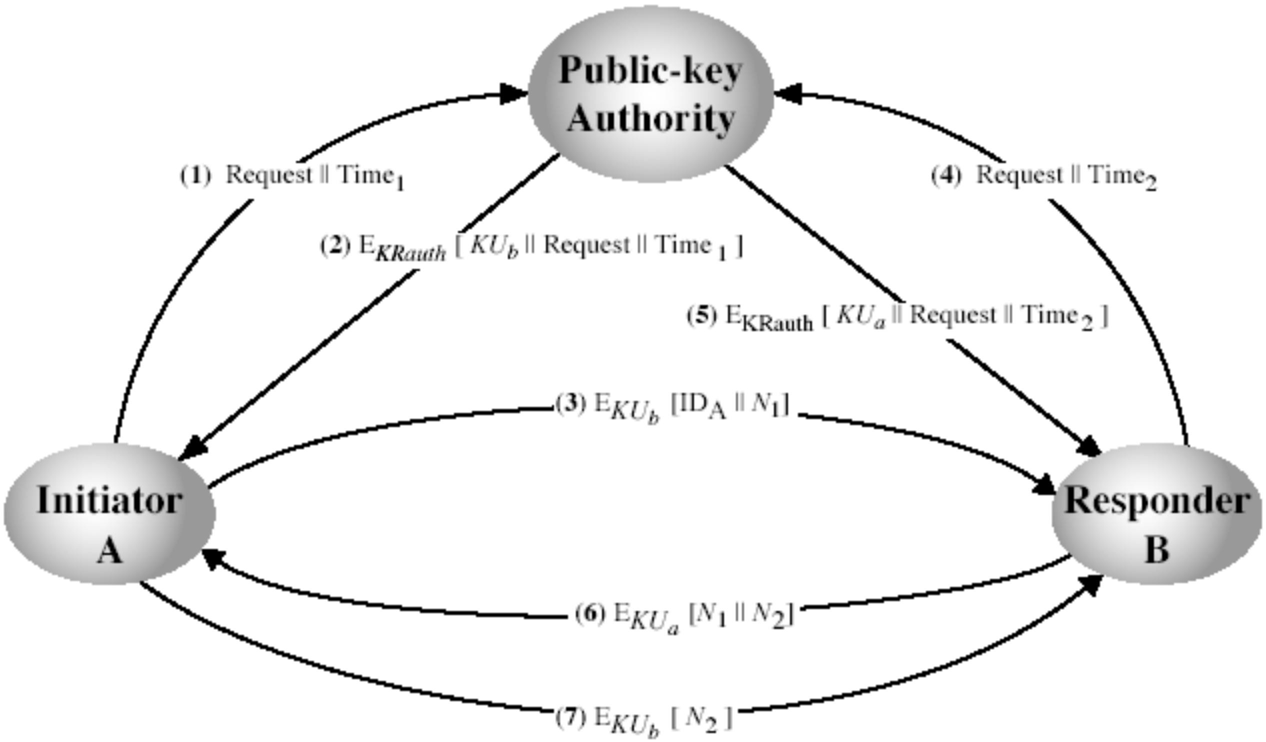
Publicly Available Directory

- Achieve 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

- Further improve security by tightening control over distribution of keys from directory
- Keeps all the properties of directory
- Requires users to know the public key for the directory
- Users interact with directory to obtain any desired public key securely
  - does require real-time access to directory when keys are needed

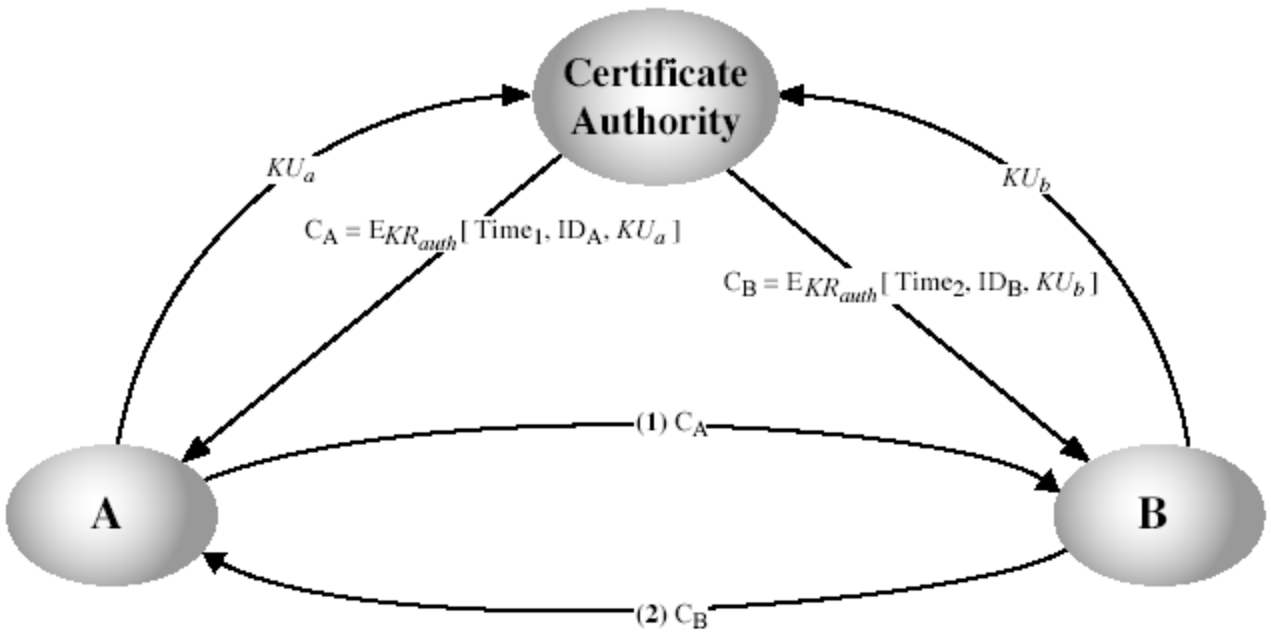
Public-Key Authority



Public-Key Certificates

- Certificates allow key exchange without real-time access to public-key authority
- a certificate binds **identity** to a **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

Public-Key Certificates



Distribution of Secret Keys using Public-Key

- public-key cryptography can be used for secrecy or authentication
  - but public-key algorithms are slow
  - so usually we want to use private-key encryption to protect message contents, such as using a session key
- There are several alternatives for negotiating a suitable session key

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

- if A and B have securely exchanged public-keys: