Information Systems Security (SOEN321)

Other Public Key Encryption and Digital Signature Schemes

Dr. Amr Youssef

Concordia Institute for Information Systems Engineering (CIISE)
Concordia University
Montreal, Canada

youssef@ciise.concordia.ca

Famous Number Theory Problems

FACTORING	Given n, find a factor of n
RSAP	find m such that $m^e = c \bmod n$
QRP	if , decide whether a is a QR or not.
SQROOT	find x such that $x^2 = a \bmod n$
DLP	find x such that $g^x = y \bmod p$
GDLP	DLP on a finite cyclic group G
DHP	given $g^a \bmod p$, $g^b \bmod p$, find $g^{ab} \bmod p$
GDHP	DHP on a finite cyclic group G
SUBSETSUM	given $\{a_1, \ldots, a_n\}$ and s, find subset of a_j that sums to s

Discrete Logarithm Problem (DLP)

- Given a multiplicative group (G, *), an element g in G having order n and an element y in the group generated by g, denoted <g>
- Find the unique integer x such that $g^x \bmod n = y$
- x is the discrete logarithm

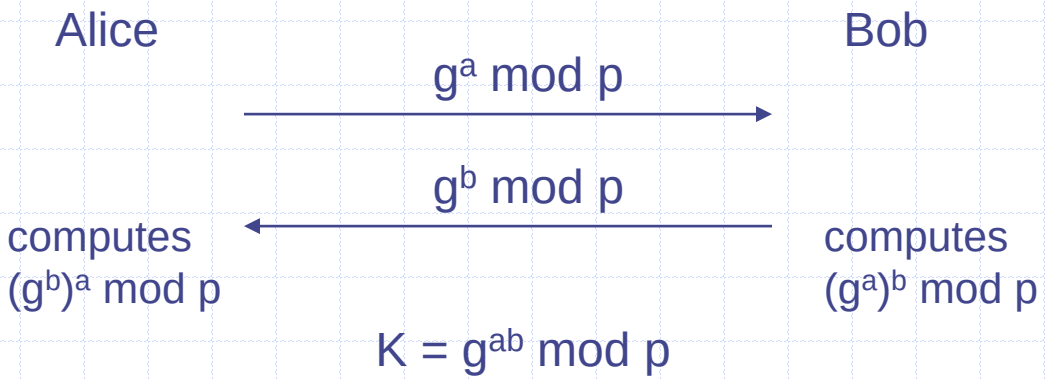
Diffie-Hellman Key Exchange

Public parameters:

p: A large prime

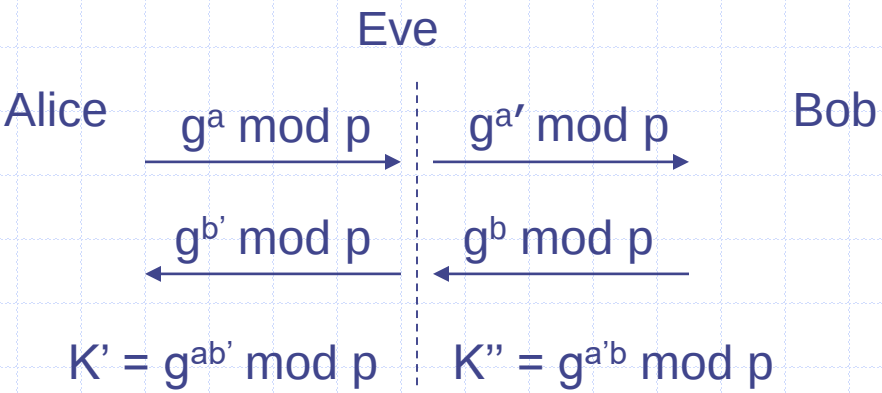
g: A generator of Z_p^*. ie., $\{g^i \mid 0 \le i \le p-2\} = \{1, 2,\ldots,p-1\}$.

a, b $\in$ {0, 1, 2,…,p-2} are secret.



Man in the middle attack

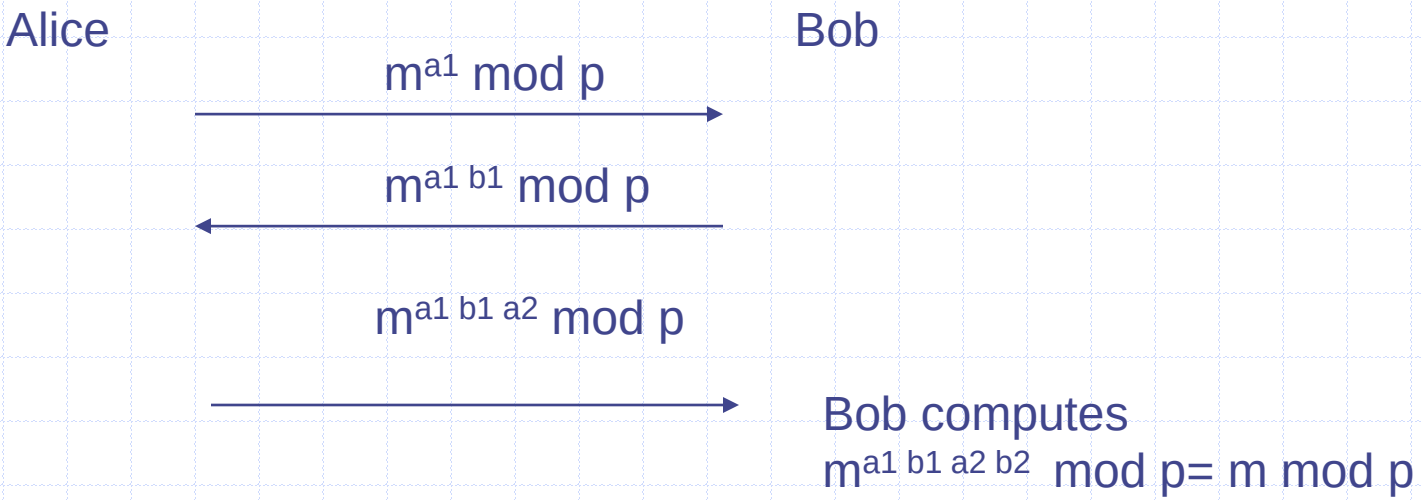
- Attacker can intercept, modify, insert, delete messages on the network.
- E.g., Man-in-the-Middle attack against DH:



- Eve can translate messages between Alice & Bob without being noticed
- Similar attacks possible on RSA & other PKC protocols.

DH Encryption

- Three pass protocol
- Alice chooses a_1, a_2 such that $a_1\ a_2 = 1 \bmod (p-1)$
- Bob chooses b_1, b_2 such that $b_1\ b_2 = 1 \bmod (p-1)$



Security of DH

- Discrete Logarithm Problem: Given p, g, g^a mod p, what is a? (easy in Z, hard in Z_p.)
- DH Problem: Given p, g, g^a mod p, g^b mod p, what is g^{ab} mod p?
- Conjecture: DHP is as hard as DLP.

  (note: Neither is proven to be NP-complete.)
- "Strong prime": If (p-1)/2 is also a prime.
- Best known method for DLP: "Number Field Sieve" with running time $e^{(1.923 + O(1)) ((\ln p)^{(1/3)}) ((\ln \ln p)^{(2/3)})}$.

Square and Multiply (Review)

How to compute (g^a mod p) for large p, g, a?

$$x^n = \begin{cases} (x^k)^2 & \text{if } n = 2k \\ (x^k)^2 x & \text{if } n = 2k + 1 \end{cases}$$

"Repeated squaring": Start with the most significant bit of the exponent.

E.g. Computing 3^{25} mod 20. $25 = (11001)_2$

$y_0 = 3^{(1)} \bmod 20 = 3$

$y_1 = 3^{(11)} \bmod 20 = 3^2\, 3 \bmod 20 = 7$

$y_2 = 3^{(110)} \bmod 20 = 7^2 \bmod 20 = 9$

$y_3 = 3^{(1100)} \bmod 20 = 9^2 \bmod 20 = 1$

$y_4 = 3^{(11001)} \bmod 20 = 1^2\, 3 \bmod 20 = 3$

Rabin Public Key Encryption

- The first example of a provably-secure public key encryption scheme.
- Recovering the plaintext is computationally equivalent to factoring.

Key Generation

select p and q primes s.t. p=3 mod 4 and q=3 mod 4

n = pq

Public key: n

Private key: (p, q)

Rabin Public Key Encryption (Cont.)

- **Encryption**

$$C = m^2 \bmod n$$

- **Decryption**

$$m = \sqrt{C} \bmod n$$

- Decryption is not an injection:
- Find the four square roots
- m1 , m2 , m3, and m4 of c mod n
- The message sent was either m1 , m2 , m3, or m4

Finding square root mod n

- Given c, finds out x such that $x^2 = c \bmod n$
- • $(c^{(p+1)/4})^2 \equiv c^{(p+1)/2} \equiv c^{(p-1)/2} \cdot c \equiv c \pmod p$
- – x mod p should be $c^{(p+1)/4}$ or $-c^{(p+1)/4}$
- Similarly, x mod q should be $c^{(q+1)/4}$ or $-c^{(q+1)/4}$
- Use Chinese Remainder Theorem to solve it
  - Find integers a, b such that $ap + bq = 1$.
  - Compute $r = c^{(p+1)/4} \bmod p$ and $s = c^{(q+1)/4} \bmod q$.
  - Compute $m = (aps + bqr) \bmod n$.
  - Compute t = $(aps - bqr) \bmod n$.
- – The four square roots of c modulo n are m, $-m \bmod n$, t, and $-t \bmod n$.
- For p=1 (mod 4), there is no known PTIME deterministic algorithm to compute square roots modulo p

Security of Rabin

- Provably secure against passive adversary
- relying on the difficulty of factoring large composites
- Obtaining plaintext from the ciphertext is equivalent to the modulo square root problem
- Modulo square root problem is equivalent to prime factoring
- Susceptible to chosen ciphertext attack similar to RSA
- Many RSA attacks can be also applicable to Rabin

ElGamal Encryption

- Published in 1985 by ElGamal
- Its security based on the intractability of the discrete logarithm problem
- Message expansion: the ciphertext is twice as big as the original message
- Uses randomization, each message has p-1 possible different encryptions

ElGamal Encryption (Cont.)

- Key Generation
  - prime p (system-wide parameter) and a generator g of Z_p^*
  - A's public key is $y=g^x$, A's private key is x
- Encryption
  - generate random integer k and compute $r = g^k \bmod p$
  - compute $c = m\, y^k \bmod p$
  - Ciphertext (r, c)
- Decryption
  - $m = c\, r^{-a} \bmod p$

McEliece PKE

- Based on error-correcting codes
- Select a particular code for which an efficient decoding algorithm is known, then to disguise it as a general linear code
- The problem of decoding an arbitrary linear code is NP-hard
  - description of the original code can serve as the private key
  - while a description of the transformed code serves as the public key
- Very inefficient but the key size is very large

Knapsack

- Based on Subset sum problem = NP-complete
  - select a subset sum problem that is easy to solve (super increasing sequence)
  - then disguise it as an instance of the general subset sum problem which is hopefully difficult to solve
- Key
  - The original knapsack set can serve as the private key
  - while the transformed knapsack set serves as the public key
- Most Knapsack proposals are broken

More Details

Knapsack Problem: Let $I = \{0, 1, \ldots, n-1\}$. Given the integer vector $A = \{a_0, a_1, \ldots, a_{n-1}\}$ and another integer X, is there a $J \subseteq I$ such that

$$\sum_{i \in J} a_i = X$$

Easy Knapsack Problem: If the numbers a_i have the **superincreasing property**, i.e.,

$$\sum_{i=0}^{j-1} a_i < a_j$$

then the knapsack problem is easy

Hard Knapsack Problem: Without the superincreasing property the knapsack problem (in general) is hard

Super increasing Sequences

Hard Example:
$A = \{3, 4, 5, 12, 13\}$; Nonsuperincreasing

Let $X = 19$. We need to try all subsets of A to find out which one of these sums to 19

$3+4=7$ $3+5=8$ $3+12=15$
$3+13=16$ $4+5=9$ $4+12=16$
$4+13=17$ $5+12=17$ $5+13=18$
$12+13=25$

$3+4+5=12$ $3+4+12=19$
$3+4+13=20$ $4+5+12=21$
$4+5+13=22$ $5+12+13=30$

Easy Example:
$A = \{1, 2, 4, 8, 16\}$; Superincreasing

$1 < 2$; $1+2 < 4$; $1+2+4 < 8$; $1+2+4+8 < 16$

Let $X = 23$. Solution is found by computing the binary expansion of $X = 23 = (10111)_2$, thus $1 + 2 + 4 + 16 = 23$

How to design a trapdoor Knapsack

Take an easy knapsack and disguise it

Example: $A = \{1, 2, 4, 8, 16\}$ Select a prime p larger than the sum 31, for example $p = 37$ Select t and compute $t^{-1} \bmod p$, for example, $t = 17$ and $t^{-1} = 24$

Produce a new knapsack vector B from A such that

$$b_i \equiv a_i t \bmod p$$

This gives $B = \{17, 34, 31, 25, 13\}$

This knapsack problem is nonsuperincreasing

However, with the special trapdoor information $t = 17$, $t^{-1} = 24$, and $p = 37$, we can convert this problem to the easy version

Example: Given B and $X = 72$, is there a subset of B summing to X ?

Solve the easy knapsack with X'

$$\begin{aligned} X' &\equiv X t^{-1} \bmod 37 \\ &= 26 \end{aligned}$$

$A = \{1, \underline{2}, 4, \underline{8}, \underline{16}\}$ and
$X' = 2 + 8 + 16 = 26$

This gives the solution for the hard knapsack:

$B = \{17, \underline{34}, 31, \underline{25}, \underline{13}\}$ and
$X = 34 + 25 + 13 = 72$

Example for a Knapsack Cryptosystem

User R: Publishes $B = \{17, 34, 31, 25, 13\}$

Keeps $A = \{1, 2, 4, 8, 16\}$, $t = 17$, $t^{-1} = 24$, and $p = 37$ secret

User S: wants to send the message $M = 12$ to User R

User S: Takes $M = 12 = (01100)_2$

Computes

$$C := 0 \cdot 17 + 1 \cdot 34 + 1 \cdot 31 + 0 \cdot 25 + 0 \cdot 13$$

which gives $C = 65$

Send $C = 65$ to User R

User R:

Receives $C = 65$

Computes $C' = 65t^{-1} = 65 \cdot 24 \equiv 6 \bmod 37$

Solves the easy knapsack problem:

$$6 = \underline{0} \cdot 1 + \underline{1} \cdot 2 + \underline{1} \cdot 4 + \underline{0} \cdot 8 + \underline{0} \cdot 16$$

This gives the message as $(01100)_2 = 12$

Digital Signature

Digital Signature Schemes

- Digital Signature: a data string which associates a message with some originating entity.
- Digital Signature Scheme:
  - secret signing key and a public verification key
- Services provided:
  - Authentication
  - Data integrity
  - Non-Repudiation (MAC does not provide this.)

Attack Models for Digital Signatures

- **Key-only attack**: Adversary knows only the verification function (which is supposed to be public).
- **Known message attack**: Adversary knows a list of messages previously signed by Alice.
- **Chosen message attack**: Adversary can choose what messages wants Alice to sign, and he knows both the messages and the corresponding signatures.

Adversarial Goals

- **Total break**: adversary is able to find the secret for signing, so he can forge then any signature on any message.
- **Selective forgery**: adversary is able to create valid signatures on a message chosen by someone else, with a significant probability.
- **Existential forgery**: adversary can create a pair (message, signature), s.t. the signature of the message is valid.
- A signature scheme cannot be perfectly secure; it can only be computationally secure.
- Given enough time and adversary can always forge Alice's signature on any message.

Digital Signatures and Hash

- Very often digital signatures are used with hash functions, hash of a message is signed, instead of the message.
- Hash function must be:
  - Pre-image resistant
  - Weak collision resistant
  - Strong collision resistant

RSA Signature

- Key generation (as in RSA encryption):
  - Select 2 large prime numbers of about the same size, p and q
  - Compute $n=p\,q$ and $\phi(n)=(p-1)(q-1)$
  - Select random integer e, $1 < e < \phi(n)$ s.t. $\gcd(e, \phi(n)) = 1$
  - Compute $d = e^{-1} \bmod \phi(n)$
- Public key: (e, n)
- Secret key: d, p and q must also remain secret

RSA Signature (cont.)

- **Signing message M**
- M must verify $0 < M < n$
- Use private key (d)
- compute $S = M^d \bmod n$
- **Verifying signature S**
- Use public key (e, n)
- Compute $Se \bmod n = (M^d \bmod n)^e \bmod n = M$
- Note: in practice, a hash of the message is signed and not the message itself.

Example of forging

- Attack based on the multiplicative property of property of RSA.
- $y1 = sig_K(x_1)$
- $y2 = sig_K(x_2)$, then
- $ver_K(x_1\ x_2 \bmod n, y_1\ y_2 \bmod n) = true$
- So adversary can create the valid signature $y_1\ y_2 \bmod n$ on the message $x_1\ x_2 \bmod n$
- This is an existential forgery using a known message attack.

ElGamal Signature Scheme

- **Key Generation (as in ElGamal encryption)**
- Generate a large random prime *p* such that DLP is infeasible in Zp and a generator g of the multiplicative group Z*p* of the integers modulo *p*
- Select a random integer *a*, 1 ≤a ≤*p*-2, and compute y= g^a mod *p*
- Public key is (p, g, y)
- Private key is *a*.
- Recommended sizes: 1024 bits for p and 160 bits for a.

ElGamal Signature (Signing and verification)

- Signing message M
  - Select random k, $1 \leq k \leq p-1$, , $k \in Z^*_{p-1}$
  - Compute
    - **$r = g^k \bmod p$**
    - **$s = k^{-1}(M - ar) \bmod (p-1)$**
- Signature is: (r,s)
- Verification
  - Verify that r is in Z_{p-1}
  - Compute $v1 = y^r r^s \bmod p$
  - Compute $v2 = g^M \bmod p$
  - Accept iff v1=v2
- Size of signature is double size of p
- Hash function must be used, otherwise easy for an existential forgery attack)

Digital Signature Algorithm (DSA)

- Variant of El Gamal
- Use a subgroup of Z_p^*