Feasibility & Infeasibility
- Hard problems for cryptography

Lily Lidong Chen
Computer Security Division, NIST

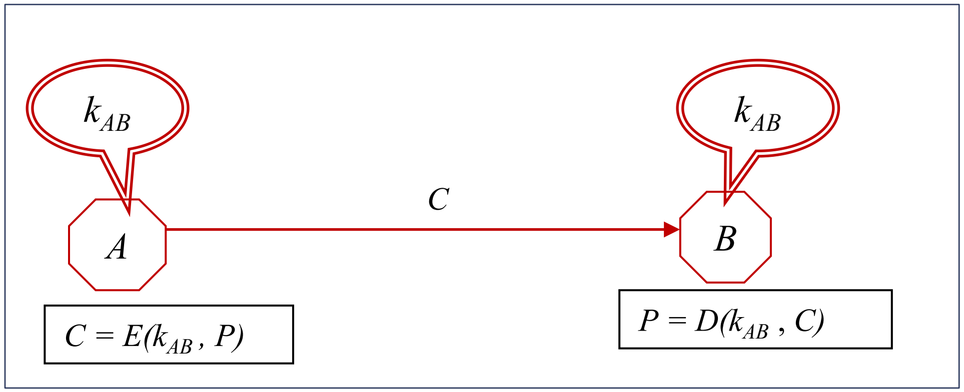
Hard Problems and Cryptography

- A problem is hard if no polynomial time algorithm is known to solve it
- The hardness is categorized by computing complexity, e.g. P and NP
- Practically, it means that it is infeasible to solve it with the currently available computing resource
- The hardness on certain problems is used as the basic assumptions for some cryptographic schemes

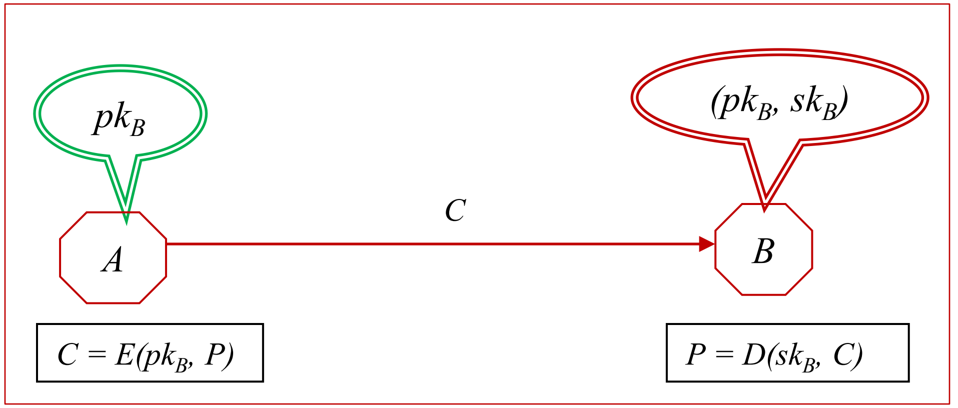
Cryptography

- Symmetric key cryptography algorithms use the same "secret" key for the sender to encrypt and for the receiver to decrypt
  - Key distribution had been a challenge
- Asymmetric key cryptography algorithms, a.k.a. public key cryptography algorithms use a pair of keys: a public key and a private key
  - The sender uses the receiver's public key to encrypt and the receiver uses the private key to decrypt
  - The public key is public, which resolved the key distribution issue

Symmetric key encryption

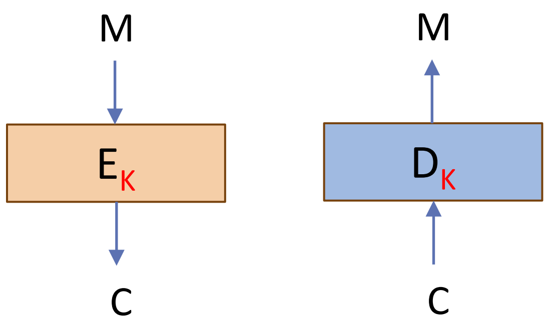


Asymmetric key encryption

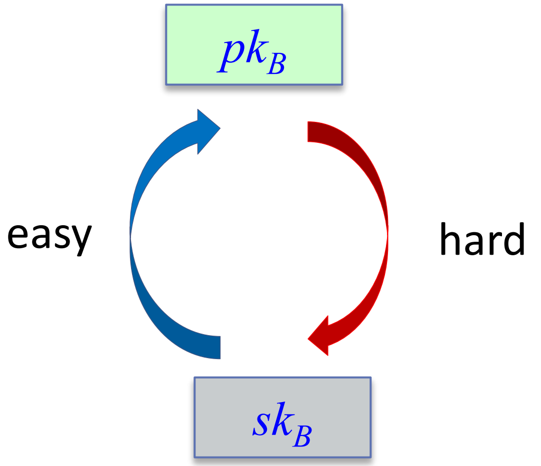


Why Asymmetric Key IS Secure?

- It is easy to understand that for symmetric key based cryptographic schemes, as long as the keys are kept secret, it shall be secure up to the strength provided by the key
  - Not necessarily, dedicated cryptanalysis may break a not well-designed cryptosystem with certain amount of plaintext and cipher-text pairs with less than the effort as made in exhaustive search



- For asymmetric key based cryptosystem, how could we know that from the public key it is infeasible to obtain the private key?
  - The security of a public key crypto scheme is based on computationally hard problems to make sure it is hard to obtain private key from public key



Integer Factorization and RSA Cryptosystem

- Given two primes p and q, it is easy to compute $n = p \cdot q$
- Given an integer n, it is *hard* to find p and q such that $p \cdot q = n$
- No algorithm has been published that can factor all integers in polynomial time
- The best algorithm of factorization is the special number field sieve (SNFS) with complexity

  $\exp((c+o(1))(\log n)^{1/3} (\log \log n)^{2/3})$
- The 2009 factorization of a 768-bit integer n took roughly a year on 2000 cores running at 2GHz.

RSA Encryption Algorithm*

Public key (n, e), where

- n, an integer, a product of two primes p and q.
- e, an integer such that $(e, \phi(n)) = 1$, where $\phi(n)$ is Euler's Totient function

Private key (n, d):

- d such that $e \cdot d \equiv 1 \bmod \phi(n)$

Encryption: For plaintext M, an integer

$$C \equiv M^e \bmod n$$

Decryption: For ciphertext C, an integer

$$M \equiv C^d \bmod n$$

Notice that $e \cdot d \equiv 1 \bmod \phi(n)$ implies

$$e \cdot d = k \cdot \phi(n) + 1$$

for some integer k. Because $M^{\phi(n)} \equiv 1 \bmod n$,

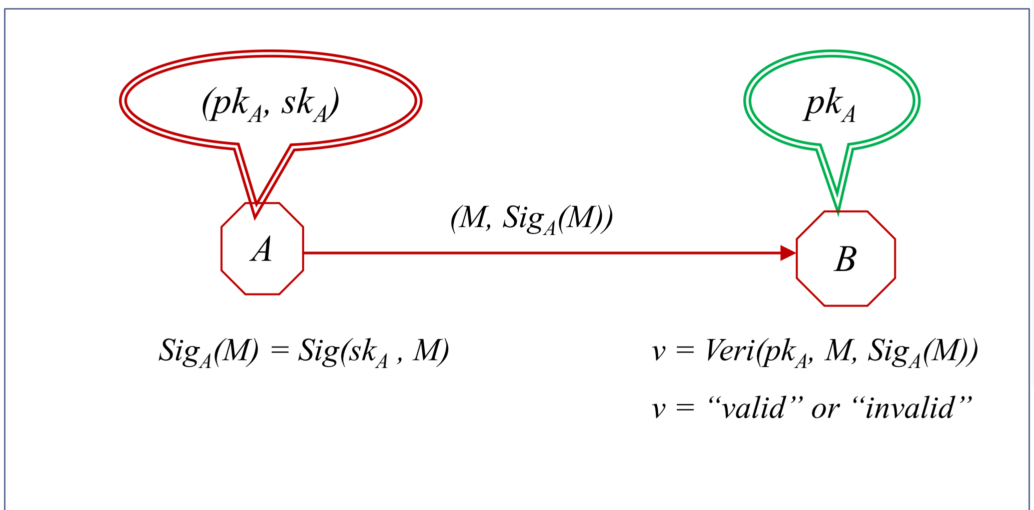
$$C^d \equiv M^{k \cdot \phi(n) + 1} \equiv (M^{\phi(n)})^k \cdot M \equiv M \bmod n.$$

*Textbook version

Digital Signatures

- Digital signatures
  - The private key is used to generate the signature
  - The public key is used to verify the signature

Digital signature: A is a signer and B a verifier



RSA Signature*

Public key (n, e), where

- n, an integer, a product of two primes p and q.
- e, an integer such that $(e, \phi(n)) = 1$

Private key (n, d):

- d such that $e \cdot d \equiv 1 \bmod \phi(n)$

Signing: For message M and a hash function H

$$Sig(M) \equiv H(M)^d \bmod n$$

Verification: For Sign(M), The verifier computes H(M) and verify whether

$$H(M) \equiv Sig(M)^e \bmod n$$

*Textbook version

Discrete Logarithm Problem

- Assume that G is a multiplicative subgroup of $GF(p)^*$ for prime p
- G is a cyclic group, $G = \{1, g, g^2, g^3, \ldots, g^{q-1}\}$, where q is a prime and g is called a generator
  - Given an integer x, $1< x <q$, it is easy to compute $g^x \equiv y \bmod p$ (in $GF(p)$)
  - Given $y \in G$, it is *hard* to find an integer x such that $g^x \equiv y \bmod p$
- Using number field sieve method, the complexity of discrete log in $GF(p)$ is estimated by

$$exp\ ((c+o(1))(\log p)^{1/3} (\log \log p)^{2/3})$$

- Since the operation is in a subgroup, it is also determined by the subgroup size q. By Pohlig-Hellman algorithm, the complexity is in the square root of q, $\sqrt{q}$
- Discrete logarithm problem over elliptic curve groups can also be used for PKC schemes
  - A curve can be over GF(p) or GF(2^q)
  - An order n subgroup of EC(GF(p)) or EC(GF(2^q)) is used, where n is a prime

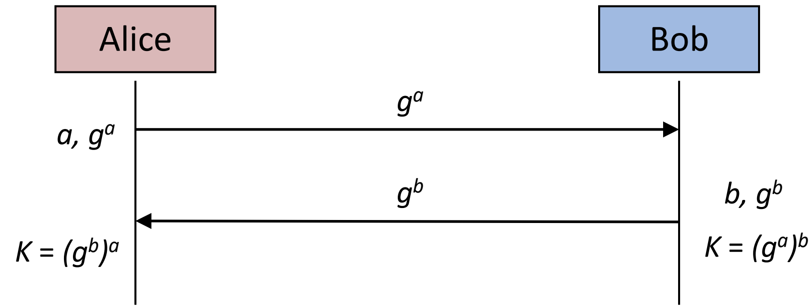
Diffie-Hellman Key Agreement

Diffie-Hellman Key Agreement

Alice and Bob can publically negotiate a set of parameters, p, q, and g, where p and q are primes and G is a q-order subgroup of GF(p)* and g is a generator of G.

1. Alice randomly selects a, 1< a < q, and computes
   $$Y_A \equiv g^a \bmod p$$
2. Bob randomly selects b, 1 < b < q and computes
   $$Y_B \equiv g^b \bmod p$$
3. Alice and Bob exchange Y_A and Y_B
4. Alice computes
   $$K_A = (Y_B)^a \equiv g^{ba} \bmod p$$
5. Bob computes
   $$K_B = (Y_A)^b \equiv g^{ab} \bmod p$$

Without secret channel, Alice and Bob agreed on a key $K = K_A = K_B$.



- Computational Diffie-Hellman problem (CDH)
  - Given $G \subset GF(p)^*$, generator g, $X \equiv g^x$, $Y \equiv g^y$, compute $Z \equiv g^{xy} \bmod p$.
- Decisional Diffie-Hellman Problem (DDH)
  - Given $G \subset GF(p)^*$, generator g, $X \equiv g^x$, $Y \equiv g^y$, and $Z \in G$, determine whether $Z \equiv g^{xy} \bmod p$.

It is not proved that CDH (or DDH) is equivalent to discrete logarithm.

PKC Applications and Standardizations

- Two classes of PKC schemes have been widely deployed
  - Discrete log based (e.g. DH, DSA, ECDH, ECDSA)
  - Integer factorization based (RSA encryption, RSA signature)
- NIST has specified digital signatures in FIPS 186-4, discrete log based key agreement like DH in SP 800-56A, and RSA key transport in SP 800-56B
  - These standards are developed for government non-classified applications
- The major schemes are standardized by many standard organizations, ISO, IEEE, IETF, ANSI, etc.
- We have been relying on PKC to protect data in transmit and in storage
  - For protect Internet traffic as in Internet Key Exchange (IKE)
  - For protect Internet applications as in Transport Layer Security (TLS)

Not Every Hard Problem Can Make a Good PKC

- Although the problem is hard to solve, which provides one-wayness, it needs a way to use some trapdoor (private information) to generate the private key
- The mathematics structure needs to be useable for a public key system to get a "invertible" mathematical transmission
- It should have a manageable size for a targeted security strength.
- To make the PKC system secure, worst case to average case reduction is needed
  - The notion of hard problems is based on worst-case analysis alone, whereas not every instance of a hard problem is necessarily hard
  - A cryptographic system based on a subclass of a hard problem will not be secure if the particular subclass turns out to be easy to solve
- It is not always straightforward to base a cryptosystem on a hard problem

112 bit security

RSA	$n \geq 2048$	
DH	$p \geq 2048$	$q \geq 224$
ECDH	G in GF(p)	$\|G\| \geq 224$

128 bit security

RSA	$n \geq 3073$	
DH	$p \geq 3072$	$q \geq 256$
ECDH	G in GF(p)	$\|G\| \geq 256$

256 bit security

RSA	$n \geq 15360$	
DH	$p \geq 15360$	$q \geq 512$
ECDH	G in GF(p)	$\|G\| \geq 512$

Hardness and Computing Power

- Moore's law
  - Over the history of computing hardware, the number of transistors in a dense integrated circuit doubles approximately every 18 months
- The techniques in computing discrete log and factorization have been continuously advanced
- As a result, in about 20 years
  - RSA public key size has increased from 512 bits to 2048 bits
  - Discrete log based system has increased the prime field modular p to be at least 2048 bits and subgroup order q at least to be 448 bits
- Advanced computing power can make infeasible be feasible with respect to certain key lengths

Quantum Computing Technology

- Quantum computing changed what we have believed infeasible
  - On a quantum computer, to factor an integer *n*, Shor's algorithm runs in polynomial time

    $$O((\log n)^2(\log \log n)(\log \log \log n))$$

  - The discrete logarithm problem can be solved in the same scale of the complexity
- With such results, all the public key cryptosystems deployed since 1980s must be replaced with the quantum resistance counterparts in the quantum computing time
- The first step is to look for proper hard problems which are computationally infeasible to be solved even by quantum computers

Quantum Computing Resistant PKC

- Some hard problems are considered as quantum computing resistant and also can be used to form public key cryptosystems, including
  - Lattice based
  - Multivariate
  - Hash based*
  - Coding based, and
  - More
- Many different schemes have been proposed in each category
  - Each of the schemes is based on a specific hard problem with respect to quantum computing (i.e. quantum computing resistant)
- Most of the quantum computing resistant PKC schemes have appeared in the past 10 years

Hard Problems – Lattice Based

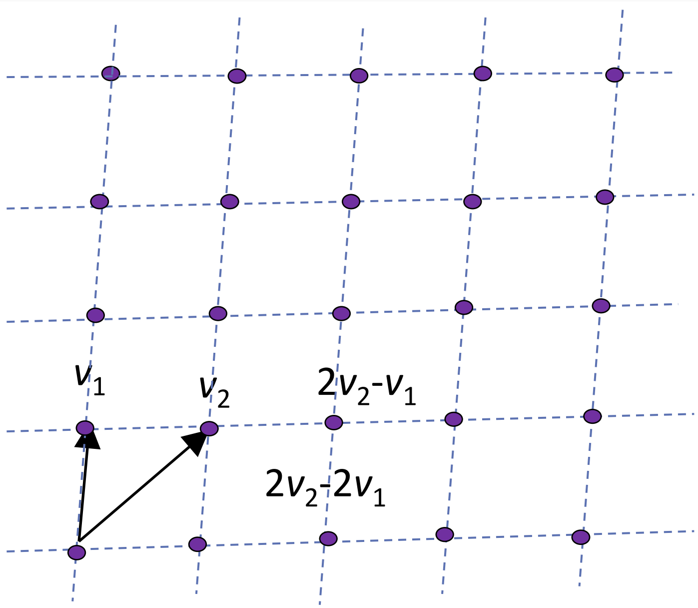
- Different hard problems in lattice have been used in constructing cryptosystems
  - Classical hard problems
    - Shortest Vector Problem (SVP)
      - Given a basis, find a shortest vector
    - Closest Vector Problem (CVP))
      - Given a basis and a target vector *t* (or a *d*-rank lattice *L*), find the closest lattice point to *t*
  - Approximation version of SVP and CVP
  - Additional
    - Decisional Shortest Vector Problem (GapSVP)
    - Bounded Distance Decoding (BDD)
    - Small Integer Solutions (SIS)
    - Shortest Independent Vector Problem (SIVP)
    - Learning With Errors (LWE)
- These problems are variants or special cases of other lattice problems
- There exists reduction relations between some of the problems
- Different lattice based crypto systems have been built based on different hard problems

Lattice: Given a basis $v_1, v_2, \ldots, v_n$ in R^n, the lattice L = L $(v_1, v_2, \ldots, v_n)$ is

$$\{ v \mid v = a_1v_1 + a_2v_2 + \ldots + a_nv_n, a_i \text{ in } Z, i = 1, 2, \ldots, n\}$$

The length of *v* is defined as

$$||v|| = (a_1^2 + a_2^2 + \ldots + a_n^2)^{1/2}$$



Early Lattice-based Crypto Schemes

- Ajtai and Dwork (1995) described a lattice-based public key cryptosystem
  - The security proof showed that every instance of the unique shortest vector problem could be transformed into a random instance of their cryptosystem with high probability
  - It encrypts one bit for each operation, not practical
- Goldreich, Goldwasser, and Halevi (1996) proposed a more practical lattice-based cryptosystem (GGH)
  - GGH is fast, but requires megabyte-size public keys to be secure
- NTRU was presented in 1996 by Hoffstein, Pipher and Silverman, that only requires RSA-sized keys, NTRU stands for
  - “N-th degree truncated polynomial ring” or
  - Number Theorists “R” Us

NTRUEncrypt

- It is typically described using the ring of convolution polynomials
  - Polynomial ring $R = Z[X]/X^n\text{-}1$, where each element is an n-1 degree polynomial over Z
- Convolution products of polynomials can also be expressed as the multiplication with a circulate matrix
  - It is possible to describe NTRU using lattices
  - Its security is related to the hardness of lattice problems in a very special class of lattices

Public parameters n, p, q, where $q > p$

1. Randomly generate two polynomials f and g in R
2. Computer f_q^{-1} such that $f^{q-1} \bullet f = 1 \bmod q$ and f_p^{-1} such that $f_p^{-1} \bullet f = 1 \bmod p$.

Public key: $h = pf_q^{-1} \bullet g \bmod q$
Private key: f

Message $m = m_0 + m_1 x + \ldots + m_{n-1} x^{n-1}$
$(-p/2 < m_i < p/2,\ i = 0, 1, \ldots, n\text{-}1)$

Encryption: select r in R at random, the ciphertext is

$$e \equiv r \bullet h + m \bmod q$$

Decryption: For ciphertext e, compute

$$a \equiv f \bullet e \bmod q$$
$$b \equiv a \bmod p$$
$$m \equiv f^{-1}_p \bullet b \bmod p$$

More on NTRU

- Its security is related to SVP in NTRU lattice
  - It is not provably secure, that is, it cannot be proved that "breaking" NTRUencrypt is equivalent to solving SVP in NTRU lattice
  - Its provably secure version is less efficient
- Computationally, NTRUencrypt is a pretty efficient scheme
- NTRU version of signature has had multiple versions, after one version is broken and then another
  - Converting NTRUencrypt to a digital signature is not as straight forward as RSA

NTRU Lattice

Public key $h = h_0 + h_1 x + \ldots + h_{n-1} x^{n-1}$
Let n x n matrix H is a cyclic matric with $C^i(h_0, h_1, \ldots, h_{n-1})^T$ as the column 1, 2, …, n, where C is cyclic shift operation

NTRU lattice is spanned by the columns of $2n$ x $2n$ matrix

$$\begin{pmatrix} I_n & O_n \\ H & qI_n \end{pmatrix}$$

where I_n is the $n \times n$ identity matrix and O_n is the $n \times n$ all-zero matrix.

Security level	n	p	q
112 bits	347	3	128
126 bits	503	3	256

The Major Challenges

- Security analysis against traditional computers
  - Is it secure to against cryptanalysis?
- Security analysis against quantum computers
  - Will a new quantum algorithm solve the underlying problem?
- Performance assessment and improvement for practical usage
  - Proper key size, ciphertext size, and signature size
- Smooth migration to quantum resistant PKC schemes in the existing applications
  - Pursue drop-in replacement and interoperability

NIST PQC Research

- Security analysis against attacks
  - Engage with crypto research community
  - Focus on security of existing schemes
  - Understand practical implications of various analysis results
- Prepare for quantum time cyber security
  - Contribute to standard activities
    - e.g. European Telecommunications Standards Institute (ETSI) white paper "Quantum Safe Cryptography and Security - An introduction, benefits, enablers and challenges"
  - Hosted "Workshop on Cybersecurity in a Quantum World" April 2-3, 2015 in NIST Gaithersburg, Maryland

Conclusion

- Finding proper hard problems for cryptographic usage is a hard problem
- The major challenge is to look for problems infeasible to solve but feasible to be used to form a cryptosystem
- As mathematicians, this is a fun area to explore