Modern Cryptography

Chung Chan

November 30, 2005

1 Introduction

The research problems of cryptography and other related fields are based on the following general communication model:

An entity U need to communicate a message M with another entity I 'safely' and 'easily' through a channel that 'leaks information' to the public including their enemy Q, who want to utilize it to 'harm' others.

It is important to point out that Q can sometimes play the role of U or I to learn more from the communication so that he can do more harms to other entities. The terms 'safely', 'easily', 'harm' and 'leaks information' are not absolute concept as they can be further specified in different aspects with different levels under different scenarios.

In the extreme case, 'safe' may mean not revealing the existence of the communication at all. The studies of how U can completely hide the communication is called *steganography*, while Q's attempt to detect it is called *steganalysis*. We may step away from this extreme by assuming the existence of the communication is publicly known. The studies of how U and I can communicate safely with this assumption is usually referred to as *cryptography*, while Q's attack is called *cryptanalysis*. *Cryptology* is the union of both. The strategy to achieve secure communication is called *cryptosystem*.

Cryptology intersects with a wide field of studies. The most important ones are:

1. information theory
2. computational complexity theory
3. number theory, group theory, algebraic geometry, ...

The *information theoretical* side concerns with whether the information gathered by Q is statistically sufficient for an attack, such as recovering M, based on a probabilistic model of the information sources from U and I. Without sufficient information, Q's attack is futile even with unlimited computational resources. In practice, not leaking sufficient information (referred to as *unconditionally secure*) is a rather expensive goal, though not impossible. We may step away from this extreme again by assuming that Q may obtain sufficient

information. The only way to thwart Q's attack is to make it computationally infeasible to succeed with any computational resources available. This type of security, referred to as *computationally secure*, therefore requires the studies of *complexity theory. Number theory, group theory,*etc. come into play because the message M can be represented by finite set of integers, which can be manipulated easily with mathematical transformations. The studies of good transformations that provide security is the core of cryptography.

2 Security Goals

The purpose of a cryptosystem can be summarized as follows,

1. confidentiality/privacy (against extraction)
2. message integrity (against injection)
3. anonymity assurances (against identity theft)
4. efficient handling of secrets (reduce overhead)

These design goals define in some aspects what a 'safe' and 'easy' communication should be and what are considered as 'harms'. The enemy Q may attempt to read the message from U (*extraction*), masquerade U to send a fake message to I (*injection*). To ensure *privacy*, U and I often share a secret not known to Q and therefore an efficient handling of the secrets is important in reducing *overhead*. To ensure *message integrity*, U often need to provide *identification information*, which should be protected from being stolen by Q and even I.

This list of objectives may not be exhaustive. New requirements may arise as a result of new applications in communications.

3 Framework of Cryptosystem

We now introduce a common mathematical framework of Cryptosystem, using the following notations,

P	plaintext, in set $\mathscr{P}$
C	ciphertext, in set $\mathscr{C}$
E	encryption function
D	decryption function
H	[cryptographic/one-way hash]/[message digest] function H
V	verification function
K	key, in set $\mathscr{K}$
X	signature/[one-way hash]/[message digest]

With this framework, two fundamental properties of cryptosystems can be formulated as follows,

decryptability:

$$P = D(E(P))$$

verifiability:

$$V(P, X) = 1 \iff X = H(P)$$

The implication and the converse are called soundness and completeness respectively.

The plaintext P is some numerical representation (e.g. ASCII) of a message. This mapping and its inverse is publicly known. To ensure privacy, U encrypts P to another numerical representation C called the ciphertext using the encryption function E. Decryptibility is a guarantee that I can recover P with some decryption function D. D must be kept secret from Q or he can recover P as well.

The choice of (E,D) pair is called the key K, which can refer to some parameters specifying a transformation for E and D out of a publicly known set. The part of K that can be made public to everyone (e.g. the possible set of transformations) is called the public key. The secret key refers to the other part supposed to be kept secret to Q (e.g. D), or either I or U.

Depending on the design of the transformation set, U and perhaps I may take part in generating the key using a truly random source (or a pseudorandom source that is computationally infeasible to predict). I and U may even negotiate the key online in public using a public key distribution algorithm.

To ensure message integrity with anonymity assurance, U generates the signature X on P using the digest function H, the choice of which is part of the key K kept secret to everyone but U. This together with verifiability guarantees I can figure out whether U sends P by verifying X on P with the verification function V.

Above all these, there is a cryptographic protocol that specifies how one or more cryptosystems are used in the communication, and how the keys are generated and maintained.

4 Classification of Cryptosystems

According to the historical development of cryptography, we may classify cryptosystems to two groups based on their security objectives: *conventional cryptography* which concerns with privacy only; and *modern cryptography* which concerns with privacy, message integrity and other objectives needed by a variety of modern applications.

According to difficulty of inverting E without additional knowledge (referred to as trapdoor) other than E, we may classify cryptosystems to two groups based on the key: *symmetric/private-key encryption* in which D can be easily obtained from E in *polynomial time and space* without any trapdoor information; *asymmetric/public-key encryption/distribution* in which D cannot be easily obtained unless there is trapdoor information. All conventional cryptography

uses private-key encryption, while the modern cryptography uses public-key encryption/distribution or both (*dual encryption*). Since private-key encryption is often more efficient than public-key encryption while public-key encryption enables authentication and other security objectives not available from private-key encryption, the dual encryption system combine the benefits of both by negotiating a temporary *session key* using public-key encryption/distribution methods so that private-key encryption can be used throughout entire session.

According to the information theoretic standpoint introduced by Shannon, private-key cryptosystem can be classied in each of the following ways,

pure secrecy system Keys are equally probable and $D_{K_2} \circ E_{K_1}$ forms a group under composition. The opposite of pure is mixed.

perfect secrecy system C is independent of P.

ideal secrecy system The entropy of the key given N ciphertexts, $\mathcal{H}(\mathsf{K}|\mathbf{C}^N)$, do not goes to zero as N approaches infinity. For strongly ideal system, the entropy does not drop at all. i.e. $\mathcal{H}(\mathsf{K}|\mathbf{C}) = \mathcal{H}(\mathsf{K})$, which is equivalent to saying that K is independent of $\mathbf{C}$.

closed secrecy system $|\mathscr{P}| = |\mathscr{C}|$.

In addition, Shannon introduced two types of composite secrecy systems,

weighted sum With probability p_i, the i-th cryptosystem is used.

product cipher A cascade of ciphers, in which the output ciphertext is the result of applying the same crytosytem repeatedly or different cryptosystems in turn.

For public-key crytosystem in particular, the difficulty of inverting E without trapdoor information corresponds to solving specific types of hard problems, and so they can be categorized accordingly,

integer factorization problem (IFP) finding the prime factors of a large integer n.

discrete logarithm problem (DLP) finding x such that $g^x \equiv b \mod p$ where p is a large prime.

elliptic curve discrete logarithm problem (ECDLP) Find the integer n such that $nB = P$ where B and P are points on the elliptic curve over a prime field.

According to how a sequence of plaintexts are encrypted, cryptosystem can be classified into two groups:

block/static/memoryless cipher The same block of plaintext is always mapped to the same block of ciphertext.

stream cipher The same block of plaintext can be mapped to different ciphertext based on the position (state/memory) of the block within the sequence.

To avoid certain type of attack (namely the forward search attack), a block cipher can be turned into a stream cipher by various techniques such as *Cipher Block Chaining (CBC), k-bit Cipher Feedback (CFB), k-bit Output Feedback (OFB), and error-Propagating Cipher Block Chaining (PCBC)* specified by the Federal Information Processing Standards (FIPS).

Based on the different family of transformations used for E, cryptosystems may be classified into the following groups,

linear cipher like $C \equiv P + e \pmod{n}$ which uses linear diophantine equations and linear congruences.

quadratic cipher like $C \equiv P^2 \pmod{n}$ which uses quadratic congruences

matrix cipher like $\boldsymbol{C} \equiv \boldsymbol{AP} + \boldsymbol{B} \pmod{n}$ which uses systems of linear congruences with single modulus

exponential cipher like $C \equiv P^e \pmod{n}$ which uses exponential congruences

5 Classification of Cryptanalysis

Based on the different level of 'information leakage', Q can launch different types of attack against a cryptosystems. There are five major types,

ciphertext attack Q have access to C only.

known plaintext attack In addition to C, Q also have access to a database of plaintext and the corresponding ciphertexts.

chosen plaintext attack In addition to C, Q have access to the encryption machine to selectively encrypt plaintexts to ciphertexts.

chosen ciphertext attack Q have access to the decription oracle to decrypt ciphertexts either before or after knowing C. If Q have such access before knowing C, he may launch a *lunchtime/[indifferent chosen ciphertext] attack*, which is indifferent to C because C is not known at the time of access. If Q have such access after knowing C, he may decrypt any ciphertext but C (to make it non-trivial). He may launch a *mid-night/[adaptive chosen ciphertext] attack* by adaptively decrypts other ciphertexts related to C due to some structure of the encryption. E having such structure is described as *malleable*.

side channel attack In addition to C, Q have access to side information such as communication frequency, duration and cache timing which are due to specific implementations of the cryptosystem.

The attacks can be generally categorized as *practical* or *theoretical*. Theoretical attacks cannot be launched in practice due to limited resources available to Q. Cryptosystems vulnerable to such attacks are said to have *certificational weaknesses*.

6 RSA Cryptosystem

In the following, we will study in detail the widely used RSA cryptosystem proposed in 1977 by Ron Rivest, Adi Shamir and Len Adleman. We shall prove its decryptability and verifiability, relate the strength of the system to the intractibility of the integer factorization problem (IFP). We will then finish off with some cryptanalytic attacks on the system and their countermeasures.

6.1 Algorithm

Two large primes p,q are first chosen, and their product n is computed, so that,

$$
\begin{aligned}
n &= pq \\
K &= [\underbrace{n, e}_{\text{public}}, \underbrace{d}_{\text{private}}] \\
C &= E(P) \equiv P^e \pmod{n} \\
P &= D(C) \equiv C^d \pmod{n} \\
X &= H(P) = D(P) \equiv P^d \pmod{n} \\
V(P, X) &= \begin{cases} 1 & \text{if } P = E(X) \equiv X^e \pmod{n} \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

Decryptability then requires

$$
\begin{aligned}
P &= E(D(P)) \\
P &\equiv P^{ed} \pmod{n}
\end{aligned}
$$

Verifiability requires

$$
\begin{aligned}
V(P, X) = 1 &\iff X = H(P) \\
P \equiv X^e \pmod{n} &\iff X = P^d \pmod{n} \\
&\iff P \equiv P^{ed} \pmod{n}
\end{aligned}
$$

Hence, the necessary and sufficient conditions for decryptability and verifiability are the same. i.e. the choices of e and d such that $P \equiv P^{ed} \pmod{n}$.

Let us introduce the following notations,

$a\|b$	a divides b
(a,b)	denotes the greatest common factor of the integers a and b.
$[a,b]$	denotes the least common multiple of the integers a and b.
$\|a\|_n$	denotes the order of a modulo n where $(a,n)=1$. $\|a\|_n \triangleq \min\left\{k \middle\| k \in \mathbb{Z}^+, a^k \equiv 1 \pmod{n}\right\}$
$A(n)$	The maximum order $A(n) = \max\left\{\|a\|_n \middle\| \forall a : (a,n) = 1\right\}$
$\phi(n)$	the Euler totient function, which is the number of positive integers less than n and coprime with n.

To find the necessary condition on ed, consider three different cases:

In the first case, $P = ap$ where a is a positive integer less than q. Using the Fermat's little theorem (FLT),

$$\begin{aligned} (P,q) &= 1 && (a<q) \\ \forall c \in \mathbb{Z}^+,\ P^{c(q-1)} &\equiv 1 \pmod{q} && (FLT) \\ \exists k \in \mathbb{Z}^+,\ P^{k_1(q-1)} &= kq+1 \\ P^{k_1(q-1)+1} &= ka(pq)+P && (P=ap) \\ P^{k_1(q-1)+1} &\equiv P \pmod{n} && (n=pq) \end{aligned}$$

Thus, the necessary and sufficient condition is $(q-1)|(ed-1)$.

In the second case, $P = aq$ where a is a positive integer less than p. By symmetry between p and q, we have the necessary and sufficient $(p-1)|(ed-1)$.

In the final case, $(P,n) = 1$. We use the known result that $|P|_n \,|A(n)$, $A(pq) = [p-1, q-1]$, and $A(p) = \phi(p) = p-1$ for any prime p, q to derive the necessary and sufficient condition as follows,

$$\begin{aligned} A(n) &= [A(p), A(q)] = [p-1, q-1] \\ \forall c \in \mathbb{Z}^+,\ P^{cA(n)} &\equiv 1 \pmod{n} && (|P|_n \,|A(n)) \\ P^{c[p-1,q-1]+1} &\equiv P \pmod{n} \end{aligned}$$

Thus, the desired condition for this case is $[p-1, q-1]|(ed-1)$, which is indeed the overall condition since it implies both $(p-1)|(ed-1)$ and $(q-1)|(ed-1)$. Thus, the necessary and sufficient condition is,

$$ed \equiv 1 \pmod{[p-1, q-1]} \tag{1}$$

where $[p-1, q-1]$ denotes the LCM of $p-1$ and $q-1$, which be easily computed using the Euclidean algorithm. For convenience, one may use the following sufficient condition instead.

$$ed \equiv 1 \pmod{\phi(pq)} \tag{2}$$

because $[p-1, q-1] | \phi(pq) = (p-1)(q-1)$. Note that this is not a necessary condition because $p-1$ and $q-1$ are both even and so $[p-1, q-1] | \frac{(p-1)(q-1)}{2} < \phi(pq)$.

With the condition on ed, one can first randomly generate a prime number for d, and compute its multiplicative inverse e. This will then satisfy decryptability and verifiability.

6.2 Attacks on RSA

The objective of an attack here is figure out P from the public knowledge of n and e, which are part of the public key, and the intercepted ciphertext C.

Consider the possibility of a chosen plaintext attack. In a *forward search attack*, Q encrypts different plaintexts into ciphertexts, and check if they are indeed C. If it is, P is found. This is impractical if n is chosen large enough as it require a search space exponential with $\lg n$, the number of bits of n.

Consider the ways Q may try to invert E to find P from C. If Q is able to factor n into pq, he can compute $\phi(n) = (p-1)(q-1)$ easily and obtain d from e by the polynomial time *[extended Euclidean]/Bankinship's algorithm* using relation (2). However, the integer factorization problem (IFP) is believed to be hard, even though it is probably neither NP complete nor co-NP complete. Currently, the most efficient probabilistic factoring algorithm is the General number field sieve, whose complexity is subexponential $\mathcal{O}(\exp((\frac{64}{9}\lg n)^{\frac{1}{3}}(\lg\lg n)^{\frac{2}{3}}))$. The most efficient deterministic factoring algorithm is the Lenstra elliptic curve factorization, which is still exponential time in $\lg n$. Other less efficient factoring algorithms exist that exploit weaknesses of the generated primes $p-1$ and $q-1$. As argued by Rivest, the weaknesses is unlikely by choosing n large enough.

Q may avoid the IFP by computing $\phi(n)$ directly. This success of this, however, solves IFP because $n = pq$ and $\phi(n) = (p-1)(q-1)$ implies $p + q = \phi(n) - pq - 1$ and $p - q = \sqrt{(p+q) - 4n}$, from which p and q can be computed easily. Thus, computing $\phi(n)$ directly is as hard as IFP, and therefore currently impractical for large n.

Q may try to compute a decryption key, δ, where

$$e\delta \equiv 1 \pmod{k[p-1, q-1]} \tag{3}$$

for some positive integer k. δ satisfying (3) is a valid decryption key because it satisfies the necessary and sufficient condition (3). The sucess of this, however, solves IFP again because $ed-1$ is some multiple of $[p-1, q-1]$, which allow us to factor n easily with technique developed by Miller. Thus, computing δ directly is currently impractical. As a side note, if everyone uses the same modulus n as their private key, Q, knowing just one key pairs (e',d'), can figure out all the key pairs by factoring n using $e'd' - 1$. This is called the *common modulus attack*, which can be easily countered by using different moduli.

Q may try to invert E directly without computing the decryption key. This is indeed taking the e-th root modulo n. If e is small enough so that C is indeed a perfect square (NOT modulo n), computing the root is very easy with, say, the Newton's method. This can be called the *small encryption exponent attack*. The countermeasure is to ensure $e > \lg n$ so that for any $P > 2$, $P^e > n$, and thus the mod n operation does take effect. If the same P is broadcast to

different entities who have the same e but different n's, say n_i, that are pairwise relatively prime. Q can, applying the Chinese Remainder Theorem, obtain from the intercepted ciphertext a congruence of P^e module the product $\prod n_i$. i.e.

$$\because C_i \equiv P^e \pmod{n_i}$$

$$P^e \equiv \sum_k C_k \left(\prod_{j \neq k} n_j\right) \left(\prod_{j \neq k} n_j\right)^{-1}_{n_k} \pmod{\prod_i n_i}$$

where the notation $(a)_m^{-1}$ denotes the multiplicative inverse modulo m of a such that $a(a)_m^{-1} \equiv 1 \pmod{m}$. For every e, Q can have $P^e < \prod_i n_i$ if Q can gather enough C_i's. Q can then obtain P by taking the root. Rather than making e large, the countermeasure is to use different e, and to pad the ciphertext with some random bits to make P being sent to multiple entities large and different. This strategy is called *salt*. The *Optimal Asymmetric Encryption Padding (OAEP)* proposed by Bellare and Rogaway in 1994, improved by Victor Shoup in 2001, is an implementation of such strategy on RSA.

Like IFP, it is not known whether an efficient algorithm exists, and whether solving this also solves IFP. In other words, chosen plaintext attack on RSA is no harder than IFP, while it is unclear if it is as hard as IFP. There are cryptosystems, such as the Rabin cipher, the breakage of which is as hard as IFP.

Consider the possibility of a chosen ciphertext attack. Having access to a decryption oracle before knowing C only help build a database of plaintext-ciphertext pairs. Thus, the indifferent chosen ciphertext attack is no more effective than the plaintext attack. Consider the adaptive chosen ciphertext attack, in which Q has access to the decryption oracle after knowing C. Although Q is not allowed to directly decrypt C, Q can do so indirectly by decrypting $C' \equiv k^e C \pmod{n}$ where $(k, n) = 1$, which gives the result $P' \equiv kP \pmod{n}$. P can then be easily calculated as $P \equiv k^{-1}P' \pmod{n}$. In general, this type of attack exploit the property, called *malleability*, that there exist functions f and f' such that decrypting $C' = f'(C)$ leads to $P' = f(P)$. The countermeasure is to use salt (padding P with random bits) to break the malleability. This also solve another problem of the exponential function: some plaintexts are left unchanged by the encryption. Borosh and Blakley in 1979 show that there are at least 9 such plaintexts, including the trivial ones $P = 0, 1$. Adding randomness means the the same message can be mapped to different plaintexts, and thus spreads those vulnerable plaintexts over different messages or avoids them completely. Furthermore, we can restrict the number to such plaintexts to be exactly 9 by choosing e such that $(e - 1, [p - 1, q - 1]) = 2$.

Finally, consider the possibility of a side-channel attack. Q may obtain information of the nature of the conversion by observing the traffic. For example, rapid short communication may indicates negotiations between the communicating parties. This type of analysis is called the *traffic analysis*. In some cryptosystems, there exists some non-trival side-channel attack that can even recover the private key. An example is the cach-timing attack on the Rijndael

crytosystem, which is accepted by the US government as the Advanced Encryption Standard in 2002 to replace the earlier Data Encryption Standard, which is proven vulnerable to *key-exhaustion attack*, a strategy in which Q simply tries out all possible keys to decrypt the C.