A Sketch of Modern Cryptology

The Art and Science of Secrecy Systems

Palash Sarkar

Palash Sarkar is with the Applied Statistics Unit of the Indian Statistical Institute, Calcutta. He received his Ph.D from the Indian Statistical Institute in 1999. His research interests are theoretical computer science and cryptology. His other interests are philosophy and literature, but he is also the holder of a first dan Taekwondo black belt.

1. Introduction

Those of us who are in the habit of reading detective stories will need no introduction to ciphers and code breaking. In the typical situation, the sleuth is presented with a cryptogram (remember the 'dancing figures' faced by Holmes) which he has to figure out. On success either a treasure trove is discovered or some vital clue is unearthed leading to the capture of the criminal. Another well-known application of secret messages is military communication. In a war, messages need to be exchanged between units of the same army in order to coordinate joint manoeuvres. Since such messages can easily fall into enemy hands, it should be ensured that none but the intended recipient can read the message. In fact, a system of exchanging secret messages was practised in the time of Julius Caesar, and the system is called 'Caesar shift' after him. The subject of cryptology has an ancient history. [1] covers cryptology from its initial use by the Egyptians some 4000 years ago, to the twentieth century where it played an important role in the outcomes of both the world wars.

However, in the present day, secure communication is no longer confined to the pages of a story book or to military communication. In the modern business world, vital information needs to be exchanged between parties for the successful completion of a transaction. Moreover, current business practices are becoming increasingly dependent on extensive use of computers and the internet. In fact, in the future, whole business transactions may perhaps be completed over the internet, giving rise to so called e-commerce. This possibility gives rise to various

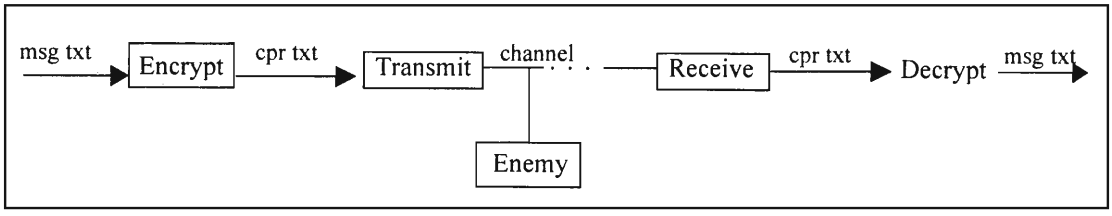


Figure 1. Basic model of a private key cryptosystem.

kinds of subtle security problems. For one thing, the exchange of information must be protected against unintentional eavesdropping. This however is only a basic requirement. Later we will briefly discuss some of the other interesting security problems of the modern business world.

2. Preliminaries

Before proceeding, let us briefly understand a few relevant terms. A *cryptosystem* is a mechanism for providing a secure means of exchanging information between two (or more) parties. A *cryptographer* is a person who designs cryptosystems to be used by others. To complete the picture we need a *cryptanalyst*, whose task is to crack cryptosystems (much like our story book detectives).

In the basic model of a cryptosystem, there is a sender and a receiver who communicate with each other (see *Figure* 1). The channel of communication is called a *public channel*, i.e. all information sent over this channel can be picked up by anybody. In particular, the opponent or the cryptanalyst can pick up any or all information from the public channel. The actual message to be sent by the sender is varyingly called *message text/plaintext/cleartext*. The process of transforming the message into secure form is called *encryption* and the process of recovering the actual message from its secure form is called *decryption*. The encrypted form of the message is called *ciphertext* or simply *cipher*. A cryptosystem is specified by the encryption and decryption procedures. These procedures involve a *secret key* which

is only known to the sender and the receiver. The secret key is exchanged between the sender and the receiver using a secret channel. To understand the difference between secret and public channels consider the following example: If two persons want to say something to each other such that no one else can hear, they can whisper in each other's ear. The whisperings correspond to communication over a secret channel. But one cannot whisper for very long. After some time they will have to speak loudly and everybody can hear them. This corresponds to communication over a public channel. In the basic model, the cryptanalyst has access to all the cipher being exchanged over the public channel. Also the encryption and decryption procedures are known to him but not the secret key. The task of the cryptanalyst is to recover a message or the key from this information.

The encryption and decryption methods must be very fast to be useful in practice. However, the cryptanalyst can use a lot of time (even days) to crack a system. It is usually assumed that the cryptanalyst has the best possible practical computing power. Also in the current era of internet, the actual work may be distributed to many computers working simultaneously to crack a system. Moreover, the target of the cryptanalyst is also modest. The ability to crack even a small part of a cryptosystem will render such a system useless. So the situation is the following. A cryptographer designs a cryptosystem and the cryptanalyst cracks it. The cryptographer modifies the system or designs a new one and again the cryptanalyst attacks it. Thus the cryptographer and the cryptanalyst play a never ending game of trying to win against one another. Of course the word *cryptology* refers to this game and more generally to the whole subject of designing and cracking secrecy systems.

At this point it might be useful to point out the difference between cryptology and coding theory, which was covered in an earlier three part article in *Resonance* [2].

The aims of the two disciplines are different. In coding theory the basic goal is to introduce redundancy into the data so that any subsequent error may be detected and corrected. This also involves a transformation of information from one form into another and back again (usually called *encoding* and *decoding*). However, the encoded message can be decoded by any person who knows the coding scheme. This is unacceptable from a security point of view. In cryptology, it is always assumed that the cryptanalyst knows the cryptosystem (the encryption and decryption methods) and yet is incapable of knowing the message without the secret key. On the other hand, it will not be possible to properly recover an encrypted message if transmission errors creep in. Thus for secure and reliable transmission one usually encrypts a message (for security) and then encodes it using coding techniques (for reliability). On the receiver side, first the received message is decoded to remove transmission errors and then decrypted to get the original message.

For our purpose, messages and ciphers will usually be integers expressed in binary notation. Let us briefly see the rationale for this assumption. Suppose we have a message written in English. This will be stored as a computer file in the form of a sequence of bits. This sequence is divided into fixed size blocks of bits, where the length of each block is (say) 1024 bits. Then each block of bits can be considered to be the binary representation of an integer in the range 0 to $2^{1024} - 1$. Thus the original message can be represented by a sequence of integers. The encryption mechanism will encrypt one integer at a time, so that for the description of the encryption procedure it is sufficient to consider the message to be a single integer.

Let us first consider the basic problem of secure information exchange. Consider the scenario where n persons want to communicate with each other and the communication between any two persons should not be intel-

ligible to the others. Such a situation may arise in the stock market, where any pair of brokers may want to exchange information without any of the other brokers knowing what is being exchanged. A naive solution to the problem is to use a model of cryptosystem as in *Figure* 1. Each person maintains a list of $n-1$ secret keys which are used for communication with the other $n-1$ persons. When person i wants to send a message to person j, he chooses from his list the secret key corresponding to j, and uses it to construct the cipher which he then sends to person j. When person j gets the message from i, he uses the key corresponding to i (which is the same key that person i has corresponding to j) to decipher the message. In this scenario, for each pair of communications, one needs a secret key and thus this gives rise to a total of $\binom{n}{2}$ keys for the whole system. Here $\binom{n}{2}$ is the binomial coefficient 'n choose 2'. So if there are 1000 brokers in a stock market each one of them will have a list of 999 secret keys and the system will have a total of $\binom{1000}{2}$ secret keys overall. Clearly maintaining and managing the secrecy of so many keys is a difficult administrative problem. Also a broker might need to communicate with some other broker very infrequently (or not at all). So it is not very sensible to maintain a secret key with such a person. Moreover, if a new broker enters the market, this person will have to establish a secret key with all the existing brokers, which is a time consuming and costly affair. A brilliant solution to this problem was first proposed by Diffie and Hellman in 1976 in a landmark paper entitled 'New Directions in Cryptography'[3], where they introduced the concept of public key cryptography.

3. Public Key Cryptography

The novel idea in public key cryptography is for each

user to have exactly two keys – an encryption key and a decryption key. The encryption key is made public, i.e. made known to everybody and the decryption key is kept secret. Going back to our stock market example, each broker has an encryption key and a decryption key. The encryption keys are published in a global (broker) directory and the decryption keys are kept secret by the respective brokers. Again suppose that broker A wants to send a message x (a positive integer as explained above) to broker B. Broker A chooses the encryption key e_B of broker B from the global directory and uses the publicly known encryption method to encrypt x to obtain a message y, i.e., $y = E(e_B, x)$, where $E(.,.)$ is the encryption function and the key e_B and x are parameters to this function. This y is transmitted to broker B. On receiving y, broker B uses the secret decryption key d_B and the publicly known decryption method to decrypt y and obtain x, i.e. $x = D(d_B, y) = D(d_B, E(e_B, x))$. A little reflection will convince the reader that such a scheme removes all the difficulties explained in the previous section. Since the encryption key is publicly known, such cryptosystems are called public key cryptosystems (PKC) and the model described in *Figure* 1 is called private (or secret) key cryptosystem. Also in a PKC the encryption and decryption keys are different and hence they are sometimes called asymmetric key systems whereas secret key cryptosystems are called symmetric key systems.

Let us now consider what are the security requirements on such a system. The functions $E(.,.)$ and $D(.,.)$, the encryption key e_B and the cipher y are known. From these it should be infeasible to obtain either the message x or the secret decryption key d_B. Viewed another way, it should be easy to obtain y from x but without the knowledge of d_B it should be difficult to obtain x from y, i.e. computation in one direction is easy, while it is hard in the reverse direction. Functions satisfying

such a criteria are called one-way functions. However, the encryption function used here is not exactly a one-way function, since knowledge of d_B makes it easy to go back. So d_B can be considered to be a sort of trapdoor which allows easy inversion. Hence the function $E(.,.)$ is actually called a trapdoor one-way function. So to implement a public key cryptosystem one has to design a trapdoor one-way function. The most popular and widely used system employing a trapdoor one-way function is the system proposed by Rivest, Shamir and Adleman and called the RSA system after them. To understand the workings of this system we need to know a bit of number theory, which is what we do next.

4. A Bit of Number Theory

Here we briefly cover some of the essential concepts of divisibility and related ideas. Let m and n be positive integers. Then it is possible to write $m = qn + r$, where $0 \leq r < n$ and q and r are unique integers (this is not difficult to prove). The integer q is called the quotient and the integer r is called the remainder. If $r = 0$, then n is said to divide m and is denoted by $n \mid m$. Given integers a, b and $n > 0$, we write $a \equiv b \bmod n$, if $n \mid (a - b)$. From these two facts it is easy to see that if we fix a positive integer n, then given any integer m, we can write $m \equiv a \bmod n$ for some $0 \leq a < n$. The set of numbers $0, \quad , n - 1$ is called a set of residues modulo n.

We now turn to the concept of greatest common divisor of two positive integers. Let m and n be two positive integers, then their greatest common divisor is denoted by $\gcd(m, n)$ and defined as the positive integer g such that $g \mid m$ and $g \mid n$ and if any y divides both m and n, then y divides g. From our school days we know of the following method to obtain $\gcd(m, n)$.

$$m = q_1 n + r_1$$

$$n = q_2 r_1 + r_2$$

$$r_1 = q_3 r_2 + r_3$$

$$\vdots$$

$$r_{k+1} = q_{k+3} r_{k+2} + r_{k+3}.$$

Here $r_{k+3} = 0$ and $\gcd(m, n) = r_{k+2}$. (Can you prove that this provides the gcd as defined above?) This procedure for finding the gcd of two integers is called the Euclidean algorithm. There is another important result on the gcd of two numbers. Let $g = \gcd(n, m)$. Then it is possible to find two integers (not both positive) λ and μ, such that, $g = \lambda m + \mu n$. (Again it is a good exercise to try and prove this result from the Euclidean algorithm.)

Of special interest are integers n and m such that $\gcd(n, m) = 1$. In such a situation n and m are said to be coprime (or relatively prime) to each other. Let n and a be such that $\gcd(n, a) = 1$. Then it is possible to show (using the last-mentioned result on gcd) that there is an integer b such that $ab \equiv 1 \bmod n$. Moreover, using a modification of the Euclidean algorithm, it is also possible to find such a number b. This number b is called the multiplicative inverse of a modulo n. Also if we restrict only to numbers in the range 0 to $n - 1$, then this number b is unique. Given a positive integer $n > 1$, the number of integers i in the range $1 \leq i \leq n - 1$, such that $\gcd(i, n) = 1$ is the value of an important number theoretic function called the Euler totient and is denoted by $\phi(n)$. In case n is a prime number, then $\phi(n) = n - 1$. Also if $\gcd(n, m) = 1$, then $\phi(nm) = \phi(n)\phi(m)$. Now we are in a position to state two fundamental number theoretic results.

Fermat's Little Theorem: Let p be a prime and a any integer coprime to p. Then,

$$a^{p-1} \equiv 1 \bmod p.$$

Euler's Generalisation: Let n be any positive integer and a be coprime to n. Then,

$$a^{\phi(n)} \equiv 1 \bmod n.$$

An elementary textbook that discusses and proves the above results (and also other interesting results) is [4]. Here we do not require any more number theory and so let us get back to cryptology.

5. The RSA System

We now present an implementation of the PKC concept as proposed by Rivest, Shamir and Adleman (the RSA system). To set up the RSA system each user chooses two large (512 bits or more) primes p and q and forms the product $N = pq$. (For the moment let us assume that such primes can be chosen. Later we will see how this can be done.) From N find $\phi(N) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$. Next two positive integers e and d (of roughly the same size) are chosen using the Euclidean algorithm such that, $1 < e, d < \phi(N)$ and

$$ed \equiv 1 \bmod \phi(N).$$

Once e and d are obtained, it is no longer required to preserve the individual values of p, q or $\phi(N)$. The public key is declared to be the pair (e, N) and the private key which is kept secret is the pair (d, N). In fact only d is kept secret.

For encryption, as explained before the binary message is divided into blocks such that each block of bits is the binary representation of an integer less than N and is encrypted independently. To encrypt an integer x one uses the public key (e, N) and forms

$$y = x^e \bmod N.$$

This y is the cipher corresponding to x and is transmitted. To decrypt, all that is required is to form,

$$z \equiv y^d \bmod N.$$

This z is equal to x and hence the original message has been recovered. ($z \equiv x^{ed} \bmod N \equiv x^{1+k\phi(N)} \bmod N \equiv x \bmod N$. Note $x^{1+k\phi(N)} \equiv x \bmod N$ if and only if $N \mid x(x^{k\phi(N)} - 1)$. Now use the fact that either $p \mid x$ or $q \mid x$ or $\gcd(N, x) = 1$.)

Thus both encryption and decryption are similar operations (modulo exponentiation). Let us briefly understand how these operations can be carried out efficiently. The first thing to observe is that $x < N$ and at no point of the computation do we allow the intermediate result to become greater than N. This is achieved by reducing the intermediate results modulo N at each step and the correctness of this procedure is guaranteed from the following identity: $st \bmod N = (s \bmod N)(t \bmod N) \bmod N$ The second question is how to perform $x^a \bmod N$ If we try to perform this by multiplying x to itself a times then we require a multiplications. Since the value of a is quite large (the binary representation will be several hundred bits), such a method will be impossible to carry out in practice. A practical method uses the following simple rule of exponentiation.

$$\begin{aligned} x^a &= (x^{\frac{a}{2}})^2 && \text{if } a \text{ is even,} \\ &= x(x^{\frac{a-1}{2}})^2 && \text{if } a \text{ is odd.} \end{aligned}$$

It is not difficult to verify that the above rule also holds when we are performing modulo exponentiation. Based on this rule it can be shown that the time to perform the computation $x^a \bmod N$ depends on $\log_2 a$ instead of a. The value of $\log_2 a$ is approximately the length of the binary representation of a and hence the operation can be carried out quite fast by a computer.

Let us now briefly try to understand the security of the system. The secret key is (d, N) which a cryptanalyst will try to recover. If from N, one can obtain the factors p and q of N then it is easy to find $\phi(N)$ and since e

is known, one can also find d using the Euclidean algorithm. It is believed that if N is a large composite number it is difficult to obtain the factors of N (However, recently it has been possible to factor an integer N which is 512 bits long.) Thus trying to break RSA by factoring N is going to be difficult. So one might try to obtain d by other ways. However, it can be shown that if one can obtain d or $\phi(N)$ from N, then one can also find p and q, i.e., factorise N It is not difficult to show that if one can obtain $\phi(N)$ then one can also obtain p and q, but showing that if one can obtain d, then one can almost certainly obtain p and q is more difficult and involves certain probabilistic arguments. Since all known attacks on RSA ultimately boil down to the problem of factoring N, it is generally believed (but not proved) that breaking the RSA system is as hard as factoring N In the next section we discuss the two problems of finding a large prime and factoring a large composite integer.

6. Primality Testing and Factoring

The problem of primality testing is to ascertain whether a given number is prime. At first glance, factoring and primality testing appear to be similar problems. But in fact there is a gulf of difference between the two and on this difference rests the strength of the RSA cryptosystem. The second problem has satisfactory practical solutions but the first problem is indeed a very difficult one. In fact it has withstood determined attacks by many brilliant researchers. Of course, the 'problems' really become problems when we are tackling large numbers – about 350 decimal digits or roughly 1024 bits long.

The naive algorithm for primality testing is very simple. To test a number n, try dividing by all positive integers greater than 1 and less than or equal to $\sqrt{n}$. If none divides n, then n is a prime (why?). However, this will require too much time when n is of the size used in cryp-

tography (see above). Instead a probabilistic algorithm is used with a very high probability of success. More precisely, if n is declared to be composite then the answer is correct, but if the number is declared to be prime then it may actually be composite but the probability of such an event is very low (typically $< \frac{1}{2^{200}}$). Such probabilistic algorithms are called Monte Carlo algorithms.

The key idea in probabilistic primality testing is to find a witness to the compositeness of n, i.e., if n is composite, we want a number b which can be used to demonstrate this. For example if $n = 525$ and $b = 50$, then since $\gcd(525, 50) > 1$, 50 is a witness to the compositeness of 525. Again let $n = 21$ and $b = 4$. Now $\gcd(4, 21) = 1$, but it is still a witness to the compositeness of 21 if we use Fermat's theorem (see Section 4). By Fermat's theorem for any prime p and integer b coprime to p, we have $b^{p-1} \equiv 1 \bmod p$. However, $4^{20} \equiv 16 \bmod 21$ (verify!), and so Fermat's theorem fails and hence 21 is composite. Motivated by this we introduce the notion of a 'witness function' $W_n(b)$, which evaluates to true if b is a witness to the compositeness of n. Note that if we fix a witness function (such as $\gcd(b, n) > 1$ or Fermat's theorem) and a composite number n, then it is possible to obtain b such that $W_n(b)$ is false. Such b's are called nonwitnesses. For example, if the witness function is $\gcd(n, b) > 1$, then for $n = 21$, the integer 4 is a nonwitness. Thus if $W_n(b)$ evaluates to false we cannot immediately conclude that n is prime, but if $W_n(b)$ is true then we know for sure that n is composite. Our aim now is to randomly pick integers b in the range 1 to $n - 1$ and evaluate $W_n(b)$. If n is composite, then we hope to obtain a b such that $W_n(b)$ is true, i.e., b is a witness to the compositeness of n. For this procedure to succeed the following condition must be satisfied. If n is composite, then there must be a large number of witnesses b such that $W_n(b)$ is true. (By a large number we mean that a constant fraction of the numbers in the

range 1 to $n-1$ must be witnesses.) If this condition is satisfied we use the following procedure.

1. Pick several distinct b's uniformly at random from the integers 1 to $n-1$ and evaluate $W_n(b)$. Typically one will choose 200 b's.
2. If for any b chosen in Step 1, $W_n(b)$ is true, then we declare n to be composite.
3. If all the evaluations in Step 1 are false, then we declare n to be prime.

If the procedure results in the answer 'n is composite', then we are sure the answer is correct. However, if the procedure results in the answer 'n is prime' then it is possible that the answer is wrong, but the probability that it is wrong is equal to the probability that n is composite and all the b's are nonwitnesses. Since there are a large number of witnesses the probability of this event is very low.

Now we come to the question of choosing a good witness function. If our witness function is $\gcd(n, b)$, then the number of witnesses with respect to this function can be very small. For example, if n is the product of two primes (the kind of composite numbers used in RSA), then the number of witnesses in the range 1 to $n-1$ is only 2. A solution may be to use Fermat's theorem as the witness function, i.e., $W_n(b)$ is true if $b^{n-1} \not\equiv 1 \bmod n$ for some b coprime to n. However, this cannot be directly used since there are composite numbers n such that for all numbers b coprime to n, $b^{n-1} \equiv 1 \bmod n$. Such composite numbers are called Carmichael numbers. For example $n = 561 = 3 \times 11 \times 17$ is the smallest Carmichael number. A way around this problem is to strengthen Fermat's theorem to build a witness function. One such witness function (called the Miller–Rabin strong pseudoprime test) is the following.

Given n and b, the function $W_n(b)$ is true if the following holds:

1. $1 \leq b < n$.
2. (a) $b^{n-1} \not\equiv 1 \bmod n$ or
   (b) There exists i, such that $2^i \mid (n-1)$ and $1 < \gcd(b^{\frac{n-1}{2^i}} - 1, n) < n$.

Condition 1 enforces b to be less than n and condition 2 enforces n to be composite. For if $2(a)$ holds then Fermat's theorem is violated and if $2(b)$ holds then n has a nontrivial divisor. It can be shown that the number of b's such that $W_n(b)$ is true is greater than $\frac{3}{4}(n-1)$. Also $W_n(b)$ can be evaluated very fast. So k distinct positive integers $b_1, \quad , b_k$ less than n are chosen and $W_n(b_i)$ is evaluated for each i. If for some i, $W_n(b_i)$ is true, then n is composite, else declare n to be a prime. Now if n is actually a prime then the result is correct since $W_n(b)$ will always evaluate to false. On the other hand the algorithm may declare a composite number to be a prime, but the probability of this is less than $\frac{1}{4^k}$. So if k is 200 then the probability of error is less than $\frac{1}{4^{200}}$, which is negligible.

The above procedure gives a method for testing whether a given number is prime. But how do we actually obtain a large prime number, which is a basic requirement to set up the RSA system? One approach is to choose a random integer and test whether it is prime. The natural question to ask is what our chances are of obtaining a prime number by this method. To get the answer we must invoke the prime number theorem, a result of fundamental importance to number theory. The prime number theorem states that the number of primes not exceeding n is approximately $\frac{n}{\ln n}$. (Here 'ln' denotes log to the natural base.) Hence if p is chosen at random, the probability that it is prime is about $\frac{1}{\ln p}$. For a 512-bit

modulus, we have $\frac{1}{\ln p} = \frac{1}{177}$. So, on average, among 177 random positive integers of appropriate size one will be prime. (If we restrict to odd integers, the probability doubles to $\frac{2}{177}$.) Thus we have a procedure to obtain large prime numbers.

Unfortunately for the factoring problem no such algorithm exists. Sophisticated techniques are being used but to date there is no 'good' algorithm for finding the factors of a large composite number. If this can be achieved then the RSA system becomes vulnerable. Currently several research groups across the world are trying very hard to factor large composite numbers. The most recent success is that of a Dutch group who could successfully factor a number which is 512 bits long. So RSA systems must employ significantly larger modulus N to assure security.

7. Other Security Problems

Here we briefly discuss several other security problems relevant to modern business world and e-commerce. As must be evident by now, these problems and their solutions are heavily dependent on the use of computers and the internet. In fact, the revolution in modern cryptology is an outgrowth of the proliferation of computer technology. In the following subsections, we outline only the problems. Except for digital signatures, we do not attempt to provide any solution to these problems. Though extremely interesting, these solutions require background material not covered in this article.

7.1 Digital Signatures

In future, complete business transactions are going to be conducted over the internet. As we all know, the heart of any business transaction is signed documents, which represent the signer's commitment to the transaction. So if transactions are going to be 'paperless' we

must be able to provide methods to sign documents digitally through computers. A conventional signature has several properties which have to be captured by digital signature schemes. One of the basic goals is to provide authentic documents, i.e., a third person can be sure that a document is genuine if it is properly signed. The actual process of this verification is called authentication. Authentication facility provides security against the following situations:

1. A 'planted' message.

2. A person denying the sending of a message.

3. A person claiming a message to have come from some other person when it actually did not.

Note that secrecy and authentication are independent concepts. A transmitted message may possess one of these without possessing the other. Next we briefly describe how the RSA system may be modified to obtain both secrecy and digital signature authentication.

Suppose Alice wants to send a message to Bob. Let (e_1, N_1) and (d_1, N_1) be the public and the private keys for Alice. Let (e_2, N_2) and (d_2, N_2) be the same for Bob. Suppose Alice wants to send a message x (an integer) to Bob. Alice performs the following steps.

1. *(Sign the message):* $s = x^{d_1} \bmod N_1$.

2. *(Encrypt the message):* $y = s^{e_2} \bmod N_2$.

3. Transmit y to Bob.

Bob performs the following steps to recover and verify the message.

1. *(Decrypt the message):* $z = y^{d_2} \bmod N_2$. If there has been no tampering, then $z = s$.

2. *(Authenticate):* $w = s^{e_1} \bmod N_1$. The w should be equal to x. Thus Bob obtains the pair (x, s) with all the properties of a signature.

7.2 Identification Schemes

Suppose a bank has branches all over the country. A person (say Alice) opens an account at one of these branches but should be able to withdraw money from any other branch. Of course we do not want a distant branch to actually verify Alice's signature with the parent branch before issuing money. This would cause unacceptable delay in payment. So the parent branch issues a secret personal identification number (PIN) to Alice. When Alice wants to withdraw money from a remote branch, she enters her PIN in a computer in the remote branch and this computer then verifies the PIN with a computer in the parent branch. This verification process is completed over a computer network. The problem here is that the PIN will be transmitted over an insecure network and the secrecy of the PIN is compromised the moment Alice reveals it. So we have to design a scheme by which the branch will be convinced that Alice is the proper person but Alice will not reveal her PIN. This is a seemingly impossible problem. In fact, one of the most interesting aspects of cryptology is to provide solutions to seemingly impossible problems. In this case, the solution is for both Alice and the verifying computer to do some computation and exchange results. From these results the verifying computer should be able to decide on Alice's identity. At no point in the protocol is the actual value of the PIN transmitted and hence it remains a secret. Since Alice is also expected to do some computation, she should also have access to a computer. In practice, Alice will be provided with a card which has a small computer chip built into it. This chip will do the computation on Alice's behalf. Such cards with built-in computer chips are called smart cards.

7.3 Secret Sharing Schemes

Those of us who have used bank lockers are aware that any such locker has two keys. One is given to the customer while the other is kept with the manager. The locker cannot be opened with any one of these keys but it can be opened if both are used. Viewed another way, if a locker has been opened then it must be the case that both the customer and the manager participated in the process. From this it is easy to imagine situations where a secret has to be shared among n persons, but at least k persons must cooperate to learn the secret. The initial distribution is done by some trusted authority who computes the 'share' of each of the n persons. The secret to be shared is some integer x. So each of the persons involved in the process gets to know something about x, but at least k persons must get together to know the value of x. One elegant solution to this problem was proposed by Adi Shamir in the late seventies. More general versions of this problem have been studied later. Discussion of the solutions requires some knowledge of linear algebra and Boolean functions.

7.4 Zero Knowledge Proofs

Suppose Alice wants to convince Bob that she knows some secret which she wants to sell to Bob. Before Bob pays for the secret, he must be convinced that Alice indeed knows the secret. So he asks Alice to prove that she knows the secret. Alice can prove this by simply revealing the secret. But then Bob does not need to pay Alice any more. So she must prove to Bob that she knows the secret without revealing the secret. The solution is for Alice and Bob to participate in a challenge-and-response protocol at the end of which Bob should be convinced that Alice knows the secret and Alice should be convinced that Bob has gained no knowledge (zero knowledge) of the secret. The study of this problem requires ideas from number theory as well as the modern

theory of computational complexity.

8. Conclusion

In this article a bare outline of the subject of cryptology has been presented. The basic purpose is to present cryptology as an exciting and rapidly developing subject. Another aim is to highlight the role of the subject in modern computer based businesses. As our society becomes more and more dependent on computers, the role of cryptology and data security will become extremely important and lead to new challenges and ingenious solutions. A good textbook on the subject is [5].

Acknowledgements

The original motivation for this article was provided by T Krishnan quite some time back. The author is grateful to him for the original motivation as well as his recent interest in the article after the lapse of a considerable amount of time.

Suggested Reading

Address for Correspondence
Palash Sarkar
Applied Statistics Unit
Indian Statistical Institute
203, B T Road
Calcutta 700035, India.
Email: palash@isical.ac.in

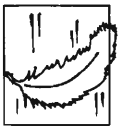
[1] D Kahn, *The Codebreakers*, Macmillan Publishing Company, New York, 1967.

[2] P Shankar, Error Correcting Codes, *Resonance*, No.10, Vol.1, 1996, Nos.1 and 3, Vol.2, 1997.

[3] W Diffie and M E Hellman, New Directions in Cryptography, *IEEE Transactions on Information Theory*, Vol. 22, pp 644-654, 1976.

[4] S Barnard and J M Child, *Higher Algebra*, Macmillan India Ltd., 1988.

[5] D R Stinson, *Cryptography: Theory and Practice*, CRC Press, Boca Raton, Florida, 1995.



"We must emphasize how amazingly close modern genetics has come to the Darwinian understanding of evolution. To the brilliant thesis of Darwin arose the brilliant antithesis of studies of mutation and to some degree Mendelism. At one time it appeared that this point of view utterly excluded Darwinism ... And now we have a synthesis so harmonious that it is necessary to declare that ever since Darwin we have only been deepening Darwinism"

Alexander Serebrovsky, 1925
(translated and quoted by M B Adams, 1980)