CRYPTOGRAPHY AND LARGE PRIMES*

B. Hartley

University of Manchester, England, and
National University of Singapore

The word "cryptography" derives from Greek and means "secret writing". Since ancient times, cryptographic methods have been in use in diplomatic and military contexts for the transfer of secret information. A quite simple cryptographic system is named after the Roman general Julius Caesar, by whom it was used. Nowadays cryptography has become important in commercial applications, such as electronic transfer of cash and computer files with limited access. Formerly cryptography could be considered as an art rather than a science, but more recently, mathematics and mathematicians have become increasingly involved in it. It appears that one of the most spectacular successes of the mathematical approach was the breaking of the ciphers used by the German High Command during the Second World War. Though the full story has not yet been revealed, this was apparently done by a group of mathematicians in England, among whom the name of Alan Turing stands out, using a specially built electronic machine which was one of the precursors of the modern computer. Turing is nowadays regarded as one of the founders of the abstract theory of computation.

---

*Text of lecture given at the prize-giving ceremony of the Interschool Mathematical Competition 1985 on 12 September 1985.

Several branches of mathematics, such as probability, number theory and combinatorics, play a part in modern day cryptography but in this article we will describe only some very beautiful but simple applications of number theory.

Modular arithmetic

In this kind of arithmetic we start with a fixed integer (whole number) $n > 0$, the "modulus". We work with the numbers $0,1,2,\ldots,n-1$. We introduce a type of addition on these numbers, consisting of ordinary addition followed by "reduction mod n", that is, taking the remainder left after dividing by n. This is called addition modulo n, or addition mod n for short. Thus if $n = 11$,

$$5 + 2 = 7 \pmod{11}; \quad 7 + 6 = 2 \pmod{11}; \quad 9 + 2 = 0 \pmod{11}$$

Multiplication is similar; it consists of ordinary multiplication, followed by "reduction mod n". Thus

$$3 \times 6 = 7 \pmod{11}; \quad 7 \times 7 = 5 \pmod{11}; \quad 5 \times 3 = 0 \pmod{15}.$$

Thus in this kind of arithmetic the product of two non-zero numbers can be zero; however it is not hard to see that this cannot happen if the modulus n is prime. The usual laws of arithmetic, such as the associative and commutative laws of addition and multiplication, are satisfied here, and in fact, for the cognoscenti we can say that we have a commutative ring. (For a slightly different approach see C. T. Chong's article [1].)

Most important for us is exponentiation; if r is a positive integer, we define $a^r \pmod n$ to mean $aa\ldots a \pmod n$ with r

factors a. Thus, for example, $2^4 = 5 \pmod{11}$. We have suppressed some mathematical details here, but it will suffice to say that in working out say $2^{23} \bmod 11$, we can do it in any way that seems reasonable, for example by working out 2^{23} and reducing mod 11, or, better, as

$$2^{23} = (2^4)^5.2^3 = 5^5.8 = 25.25.40 = 3.3.7 = 63 = 8 \pmod{11}.$$

Thus we see one virtue of modular arithmetic – the size of the numbers does not get out of hand.

A basic result is

Fermat's Little Theorem (P. de Fermat, 1632–1690). *If p is a prime and* $1 \le a \le p-1$, *then* $a^{p-1} = 1 \pmod{p}$.

Public key cryptographic systems

A public key cryptosystem is one which the encryption key or system is public knowledge, while decryption requires some special piece of information only possessed by the receiver for whom the message is intended. This can enable a number of individuals to communicate conveniently and in mutual secrecy with a single receiver, for example. A good analogy is the following. Suppose everyone possesses an English–Hungarian dictionary but only one person A possesses a Hungarian–English dictionary. Then English messages can be sent to A by translating them into Hungarian. Although A can read these without too much difficulty, people would find it difficult or impossible to read each other's messages. It is not that there is any difficulty in principle in this, as the English–Hungarian

dictionary could be used in reverse. Rather, it is the time necessary which is prohibitive. (Hungarian speakers should modify this example appropriately).

For our purpose we shall assume the messages to be sent are in numerical form, consisting of strings of integers between 0 and 9. This can always be achieved by a simple rule such as A = 01, B = 02,... . Consider the following public key cryptosystem. Each participating institution (for brevity we refer to these as "banks") has assigned to it a "coding modulus" which will be a large prime p, and a "coding exponent", which will be an integer c with $1 < c < p-1$ and such that c and $p-1$ have no common factor except 1. These will be public knowledge and available in some kind of directory. To send a message x to a given bank we first break up x (which is a string of digits) into blocks $x_1, x_2, \ldots$ in some preassigned way, so that for instance each block is less than p, and then send $x_1^c, x_2^c, \ldots$ (mod p), where c and p correspond to the bank in question. Thus if p = 11, c = 3 then instead of 3 5 5 2 7 we would send 5 4 4 8 2.

To decrypt or read this message, the bank must find the "cube root" of each digit mod 11. This is easily done by making a table of cubes and using it backwards. However if p were very large, say with about 100 digits, such a table would be too big to store, let alone use. Thus this system appears very secure, in as much as not even the intended receiver can read it. However it is possible to find a "decoding exponent" d which the bank may use, as follows. Since c and $p-1$ have no common factors other than 1, there exist integers d and x such that

$$cd + (p-1)x = 1. \qquad (*)$$

Then mod p we have, for $1 \le y \le p-1$,

$$y = y^1 = y^{cd+(p-1)x} = (y^c)^d\ (y^{p-1})^x = (y^c)^d$$

by Fermat's Little Theorem. Thus the message can be decrypted by raising its terms to the d-th power (mod p). For example when p = 11, c = 3, we have 3.7 − 10.2 = 1, so d = 7. We might thus supply the receiving bank with the appropriate value of d and hope to have a good cryptosystem. This hope is without foundation, however, as the integers d and x in (*) can be calculated from c and p, which are known, quite quickly using the Euclidean Algorithm, even if p is very large. Thus an interloper in possession of a suitable computer could break this system without difficulty.

The beautiful idea put forward by Rivest, Shamir and Adleman a few years ago was similar to the above, except that the coding modulus p is replaced by a coding modulus m of the form m = pq, where p,q are different large primes. "Fermat's Little Theorem" now has the form

If p,q are different primes, $1 \le a < pq$, and a is not equal to p or q, then $a^{(p-1)(q-1)} = 1 \pmod{pq}$.

This time the coding exponent c should satisfy $1 < c < (p-1)(q-1)$, and should have no factors other than 1 in common with (p−1)(q−1). For example we might have m = 11,388,301,907 and c = 257, though in practice the primes p,q should have about 100 digits.

Again coding or encryption consists of raising to the c-th power mod m. This can be carried out relatively quickly on a not particularly large computer. The decoding exponent d is given by

finding integers d,x such that

$$cd + (p-1)(q-1)x = 1 \qquad (**)$$

The point now is that although m and c are publicly known, *the primes* p,q *themselves are known only to the bank*. The latter can thus readily solve (**), finding d, and decrypt messages sent to it without difficulty. The potential interloper wishing to solve (**) needs to know (p−1)(q−1) and thus must first factorize m into its prime factors. The reader considering the m given above will begin to suspect that this is a very serious obstacle. Indeed, using the most sophisticated factorization techniques and the largest computers available, finding the prime factors of a number obtained by multiplying together two large primes (with about 100 digits each, say) would take much longer than the age of the universe.

Again, there is no difficulty in principle in factorizing a number n, as trial division by 2,3,... will eventually work. The point is that there is not enough time to do it that way, and other factorization methods are not significantly better. Thus the Rivest–Shamir–Adleman system is at present completely secure.

A few remarks are in order. Firstly, it is known that decrypting the RSA system is *equivalent* to factorizing m. In other words, there is no trick which will obviate the factorization problem. Secondly, no quick factorization method is known, and it seems to be widely believed that factorization is intrinsically complex, in that no rapid factorization method can be devised. However this has not been proved. It is entirely possible that someone will devise a new method which will render the RSA system useless. If this comes about,

however, it will have to result from a new mathematical insight. Easy estimates show that present methods will never be adequate, however much computing power becomes available, since the latter is limited by physical constraints such as the number of atoms in the universe, the speed of light, etc.

It is even possible that the above mathematical breakthrough has taken place! Research in the U.S.A. on prime numbers and factorizing, which until recently was regarded as one of the most pure and esoteric branches of mathematics, is now classified and subject to the scrutiny of the U.S. National Security Agency.

Obviously, in order for the above system to be in widespread use, a good supply of large primes is needed. For some information on prime testing and finding large primes in spite of the complexity of the factorization problems, see [1]. I understand that the time taken to encrypt messages remains a practical obstacle to the widespread use of the RSA system, as 30 seconds is considered a long time in some contexts, but work is proceeding on the development of special purpose chips.

Key word interchange

Many cryptographic systems have the feature that a secret key word must be in possession of both the sender and the receiver in order for messages to be transmitted back and forth. Acquisition of the key word by an interloper allows him to break the system and read the messages. Thus these are not public key systems. The hidden snag is the following. How can the keyword be transmitted from one to the other in the first place?

Obviously this itself needs some encryption system. Once again, modular arithmetic can play a role.

Now it is known that if p is a prime, then there exists a number a with $1 \leq a \leq p-1$, such that as k goes from 0 to p–1, the numbers $a^0 = 1, a^1, a^2, \ldots, a^{p-2}$ run over the numbers $1,2,\ldots,p-1$, each appearing once. Such a number a is called a "primitive root mod p". From the following table we see that 2, 6, 7, 8 are primitive roots mod 11, while 3, 4, 5, 9, 10 are not. The numbers displayed on each row are the powers (mod 11) of the numbers in the left hand column.

Exponent / Number	0	1	2	3	4	5	6	7	8	9	10
2	1	2	4	8	5	10	9	7	3	6	1
3	1	3	9	5	4	1	3	9	5	4	1
4	1	4	5	9	3	1	4	5	9	3	1
5	1	5	3	4	9	1	5	3	4	9	1
6	1	6	3	7	9	10	5	8	4	2	1
7	1	7	5	2	3	10	4	6	9	8	1
8	1	8	9	6	4	10	3	2	5	7	1
9	1	9	4	3	5	1	9	4	3	5	1
10	1	10	1	10	1	10	1	10	1	10	1

The distribution of primitive roots appears to be not very well understood. It is known that one can always find a primitive root mod p that is less than $2^{m+1}\log p$, where m is the number of distinct prime divisors of p–1, but experience suggests there is usually one much smaller than that.

Now fix a prime p and a primitive root a mod p. Thus, as x runs from 0 to p–2, the numbers $a^x \pmod p$ run from 1 to p–1, in some order. That is, if $1 \le y \le p-1$, the equation

$$a^x = y \pmod p$$

has a unique solution for x in the range $0 \le x \le p-2$. Finding x is reminiscent of taking logarithms to the base a, and so this is called the "mod p logarithm problem". In principle this poses no difficulty. To solve $6^x = 5 \pmod{11}$, we can simply search through the above table and find x = 6. However if p is large the scale of the problem renders this approach infeasible. There is a more ingenious and faster method due to Adleman (see [2], but even this is too slow for practical implementation when p is large (say around 200 digits). Like the factorization problem, the mod p logarithm problem is thought to be intrinsically complex, in that no rapid method exists which will solve it in a reasonable amount of time. But again, this has never been proved.

Now we return to the problem of interchanging a key word, or more precisely, of putting a common keyword into the possession of two potential communicators, traditionally known as Bob and Alice. We may imagine they are in a room with several other people, and may only communicate by writing on a large blackboard. First they choose a large prime p and a primitive root a mod p, and write them on the blackboard. Then Bob thinks of an integer B between 0 and p–2, and likewise, Alice thinks of A. Bob works out $a^B \pmod p$ and writes it on the blackboard. Alice works out $(a^B)^A = C \pmod p$ and makes a note of it. Next Alice works out a^A, writes it on the blackboard, and Bob works out $(a^A)^B = (a^B)^A = C \pmod p$. Now both are in possession of

C. The spectators know a, a^B and a^A, but to find A or B, and hence C, they must solve a mod p logarithm problem.

References

[1] C. T. Chong, *Number theory and the design of fast computer algorithms*, Mathematical Medley 12 (1984), 52–56.

[2] Alan C. Konheim, *Cryptography, a primer* (Wiley, Interscience, 1981).