JOURNAL OF THE
CHUNGCHEONG MATHEMATICAL SOCIETY
Volume 2, June 1989

A Note on Public Key Cryptosystem

S. HAHN

This is an extended version of the talk the author gave at annual meeting of Choong-chung mathematical society, 1989. The aim of this paper is to explain the so called RSA public key cryptosystem assuming minimum background. Currently, it is considered to be one of the safest cryptosystem.

First we need some concepts from complexity theory. Suppose there is an algorithm A. That algorithm will work with some initial data of length N (Think of a computer program to which a user inputs a number using binary expression.). If the total amount of necessary operations to finish the algorithm with given data is bounded from above by a certain polynomial of N, we call the algorithm A has polynomial running time. If the total amount of necessary operations is bounded from above by a polynomial of $\exp(N)$, we call the algorithm A has exponential running time. Of course there might be an algorithm which has both polynomial and exponential running time. Generally speaking, most algorithms which have polynomial running time are within the reach of the computing speed of any supercomputer of current decade. There are several problems which have algorithms with polynomial running time. Examples are linear programming, factorization of polynomials with integer coefficients over the integer ring. On the contrary, it is an open question whether there exists an algorithm which will factorize integers in polynomial time. About this very deep question, most opinions are that there is no such algorithm.

The RSA system-invented by Rivest, Shamir, and Adleman-is based on the conjecture of the non-existence of integer factorization algorithm with polynomial running time. Let's see the idea behind the RSA system;

Received by the editors on 29 June 1989.
1980 *Mathematics subject classifications*: Primary 94A60.

Choose two large prime integers p and q. Generating two hundred decimal digits prime will be done within a single day by methods of Rumely or Atkin. Moreover for practical purposes it looks like that it is not necessary to prove that the chosen integers are prime if those numbers passe several pseudo-prime tests.

Let $r = pq$. The security of RSA system depends on the fact that r is very hard to factorize without knowing p or q. Now each user i does the same thing, i.e. choose their own p_i, q_i, and r_i. User i knows $\phi(r_i) = (p_i - 1)(q_i - 1)$ because he constructed r_i, where ϕ is Euler's ϕ function, but other than user i, nobody knows $\phi(r_i)$. It is proved that knowing $\phi(r_i)$ is equivalent to knowing the factorization of r. User i selects a random integer s so that $\gcd(r_i, s_i) = 1$. User i computes t_i so that $s_i t_i = 1 \bmod \phi(r_i)$. This can be done by Euclid's division algorithm very easily. Now user i publishes r_i and s_i only. This is same as publishing the encryption algorithm which we are going to explain.

Encryption algorithm is as follows: Suppose user j wants to send a message to user i in secret. User j changes his message into binary string. Cut his string into pieces so that each piece is smaller than r_i as a number. Let M be one of those pieces. User j changes his M into E by $E = M^{s_i} \bmod r_i$ and send E, instead of M, to user i. User i recovers M by computing $E^{t_i} = M^{s_i t_i} = M \bmod r_i$ by Euler's theorem. Check that even if $\gcd(M, r_i)$ is not one, still user i recovers M from E. All this modulo calculations will be done very fast in a computer. For example try $2^{2^{10}+1} \bmod 19$.

Now let's look at the RSA system more closely. When there is an intruder, he will certainly try to factorize published r_i. So we have to consider how large the integer r_i has to be. The author think that National Security Agency of USA is capable of factorizing one hundred decimal digits integer within a day. However as the number of digits increase the expected running grows fast, and choosing p_i and q_i to have one hundred decimal digits each seems to keep the secret for considerably long time. There is another famous attack which does not require the factorization of r_i. Suppose that the order of s_i modulo $\phi(r_i)$ is quite small. Then an intruder can recover M from E by successively computing E^{s_i}, $E^{s_i^2}$ and so on. This is because $t_i = s_i^a \bmod \phi(r_i)$ for some small integer a. So each user of the system

has to avoid such situation. The order of s_i modulo $\phi(r_i)$ is a divisor of $\phi(\phi(r_i))$. So choose p_i and q_i so that $p_i - 1 = 2P_1P_2$ and $q_i - 1 = 2Q_1Q_2$ where P_1, P_2, Q_1, and Q_2 are primes of size about $r_i^{1/4}$. Then $s_i \bmod \phi(r_i)$ will have order two or order not smaller that 10^{40} for our choice of hundred decimal digits p_i and q_i.

Note that there are some un-concealable messages. Trivial example are $M = 0$, $M = 1$, and $M = r_i - 1$. Suppose that $M^{s_i} = M \bmod r_i$. then $M^{s_i - 1} = 1 \bmod r_i$. Recall that s_i is trivially odd because $\gcd(s_i, \phi(r_i)) = 1$. Since the unit group of the ring $\mathbf{Z}/(r_i)$ is isomorphic to the group $\mathbf{Z}/(p_i - 1) \oplus \mathbf{Z}/(q_i - 1)$, we see that there are at least nine un-concealable messages. It is easily seen that if someone knows any of the remaining six un-concealable messages, then he can break the system of user i. So user i has to make sure that those six messages do not appear in plain ordinary language.

Finally there is another problem an intruder may cause. Since the encryption algorithm is published, anyone can send false message to user i to confuse him. Even if the message from user j is legitimate, user i may reject the message suspecting the message is a false one or user i may reject the message if he does not like the content of the message and justify his decision by saying it could be a false one. Also user j may claim that he did not sent a certain message even thouth actually he sent it. The theory of digital signatures were invented to prevent such situation. When sending the secret message E to user i, user j sends his name J (or his address or whatever identification mark of user j) after transforming into $S = J^{t_j} \bmod r_j$ using his secret key t_j. After receiving E and S, user i can check whether $S^{s_j} = J^{t_j s_j} = J \bmod r_j$ using published s_j. Note that forging digital signature S is same as computing s_j-th root of $J \bmod r_j$ because $S^{s_j} = J \bmod r_j$. And it is also a hard problem believed at the same level of difficulty with integer factorization problem. So digital signature S will certify the legitimacy of the message E.

There are several variations of RSA system using, for example, matrix codes, algebraic number fields, or elliptic curves. But their theoretical security has not been studied yet. Also from practical point of view, interactive proof may be more suitable in many situations because it is easier to generate and easier to change from time to time

though it takes more time in sending and receiving measages.

REFERENCES

1. S. Hahn, *A Survery of Integer Factorization Algorithms*, (to appear).
2. N. Koblitz, *Elliptic Curve Cryptosystems*, Mathematics of Computation Vol. 48, No. 177, January (1987), 203–309.
3. S. Lakshmivarahan, *Algorithms for Public Key Cryptosystems*, in "Theory and Application, Advances in Computer Sciences," Academic Press.
4. A. Lenstra and M. Manasse, *Factoring by Electronic Mail*, extended abstract.
5. _____, *Algorithms in Number Theory*, (to appear).
6. H. Lenstra, *Primality Testing Algorithms*, Seminaire Bourbaki 33e annee (1980/1981). no. 576
7. S. Qi, *Some Results in the Application of the Number Theory to Digital Signal Processing and Public-Key Systems*, in "vol. 77 of Contemporary Mathematics of AMS," pp. 107–112.

Department of Mathematics
Korea Institute of Technology
Taejeon, 305-701, Korea