CO-2	179

ASAT

MILITARY TECHNICAL COLLEGE
CAIRO EGYPT

APPROACHES TO BREAK THE RSA ALGORITHM AND HOW TO COMBAT THEM

*M.N. Elsherbiny **M.N. Saleh ***A. Elosmany ***S. Elhabiby

ABSTRACT

The public key cryptosystem is one of the protection systems used to secure communication between computer terminals. Since no technique exists to prove that an encryption scheme is secure, the only test available is to see whether anyone can think of a way to break it. This paper outlines a selection of attacks that have been used and explains some of the basic tools available to the cryptanalyst. We presented survey of a collection of protocols in which the level of security is not actually attained, not because of a failure of the encryption algorithm used, but rather because of shortcomings in the design of the protocol. Guidelines will be extracted from the analysis of these protocols.

A cryptanalytic attack on the use of short RSA secret exponent is described. The attack makes use of an algorithm based on continued fractions that find the numerator and denominator of a fraction in a polynomial time when a closed enough estimate of the fraction is known. The public exponent e and the modulus n = pq can be used to create an estimate of a fraction.

1. INTRODUCTION

The protection systems of data communication between military aircraft and communication centers are applied to prevent access to the communicated data and unauthorized modifications (destroying, altering, etc). The most famous public key cryptosystem which provides security for data communication is the Rivest, Shamir, Adleman (RSA). We shall show that many approaches are used to break the RSA public key cryptosystem and also show the ways that the cryptanalyst might try to determine the secret key [1].

An encryption algorithm must be used within a set of rules or procedures known as a protocol which insures that the algorithm will actually provide the security and/or authentication required by the system. The development of a system to provide data secrecy and or integrity actually involves two areas of analysis :

- the design of strong encryption algorithm.
- the design of sound protocol.

The design of a protocol includes the specification of the characteristics of the cryptoalorithms which may be used in the protocol without degradation of security of the system.

---

* *Graduate Students.*
** *Faculty of Engineering, Ain Shams University.*
*** *Military Technical College.*

In this paper, we will consider examples of protocol failures. By this we mean instances in which the protocol fails to provide the advertised level of security or authentication. Since the examples considered do not use cryptoalgorithms which are inherently weak, we use RSA algorithm, the failure involves the protocol design [2]. We shall extract the principles of the protocol design from the analysis of the examples and from these principles we can extract the guidelines for development of future protocols which are resistant to the attacks demonstrated in this paper.

There are various attacks on RSA that require, either the public or secret exponent to be short. The attack on short secret exponent is based on continued fractions [3]. Continued fractions can be used to find a fraction involving the secret exponent using the underestimate fraction which consists of the public key and the modulus.

In section (2), we shall show cryptanalytic approaches to break RSA algorithm. In section (3), we shall show the protocol failures in RSA cryptosystem. In section (4), we shall explain how to break the RSA cryptosystem if the system uses a short secret exponent. In section (5), we shall present the resulting guidelines to design an RSA cryptosystem resistant to the demonstrated attacks. In section (6), conclusions are summarized.

2. CRYPTANALYTIC APPROACHES FOR BREAKING RSA CRYPTOSYSTEM

One approach that enables an opponent to break the RSA algorithm is to factor the modulus n. The cryptanalytic problem of factoring [4] is stated as follows : factor a composite odd number n, where n is the product of two prime factors (n = pq) and where there exists a public key (e) and a secret key (d) that satisfies the relation :

$$e.d \equiv 1 \ mod (p-1)(q-1) \qquad (1)$$

Assume that the opponent has knowledge of chosen ciphertext y_1, y_2,... and the corresponding recovered plaintext x_1, x_2,... without having knowledge of secret key (d) which satisfy the relation:

$$x_i = y_i^d \ mod \ n, \qquad i = 1,2,.... \qquad (2)$$

The problem of efficiently factoring large composite odd numbers has been of interest for centuries. In fact, several new public key cryptosystem and signature schemes, including the RSA public key cryptosystem base their security on the supposed intractability of the factoring problem. Over the last few years there has been developed remarkable six ways for factoring. The methods all have the common running time to factor the modulus (n) [5,16].

$$T(n) = exp\left[\sqrt{ln(n)\, ln\ ln(n)}\right] \qquad (3)$$

The six methods are as follows :

(i) The elliptic curve algorithm of Lenstra [6].
(ii) The class-group algorithm of Schnorr-Lenstra [7].
(iii) The linear sieve algorithm of Schroeppel [8].
(iv) The quadratic sieve algorithm of Pomerance [8,9].
(v) The residue list sieve algorithm of coppersmith, Qldlyzko and Schroeppel [10].

(vi) The continued fraction algorithm of Morrison-Brillhart [11].

It might be pointed out that none of these methods have actually been proved to have the running time T(n), but rather there are heuristic arguments that give this function as the common running time [5].

In RSA cryptosystem, each user chooses a pair of secret primes p,q which are large enough so that factorization of the modulus (n = pq) is not feasible even with the help of high speed computers and using the fastest known methods of factorization. It is therefore essential that n is large enough to make the work needed to factor n sufficiently great. Additional protection against factoring algorithms can be achieved by ensuring that the following conditions are satisfied [4] :

(i) p and q differ in length by only a few bits.

(ii) each number (p-1) and (q-1) contains a large prime factors p' and q'.

(iii) the greatest common divisor (gcd) of (p-1) and (g-1) is small.

For further protection, each number (p'-1) and (q'-1) contains a large prime factors p" and q".

Without regard for the usual methods of factoring composite odd numbers, the modulus n can be factored if either the Euler's totient function $\phi(n)$ or the secret key is available. If $\phi(n)$ is available, then n can be factored by the following steps :

(i) Compute (p+q) from the relations

$$n = pq, \qquad \text{and} \qquad \phi(n) = n - (p+q) + 1 \tag{4}$$

(ii) Compute (p-q) from the relation

$$(p+q)^2 = p^2 + 2n + q^2 = (p-q)^2 + 4n \tag{5}$$

$$(p-q) = sqrt\left[(p+q)^2 - 4n\right] \tag{6}$$

(iii) Compute the factors p or q from the relation

$$q = \frac{(p+q)-(p-q)}{2} \tag{7}$$

If the secret key (d) and the public key (e) are available, then calculate

$$d \cdot e - 1, \tag{8}$$

This value is a multiple of $\phi(n)$, so

$$\phi(n) = \frac{d \cdot e - 1}{K} \text{ for } K = 1, 2, 3, \ldots. \tag{9}$$

Using the above steps (i), (ii), (iii), we can determine the factors p and q.

Another approach of factoring the modulus (n) is that, if there is a value x not relatively prime to n, the factor p and q can be determined by calculating greatest common divisor of x and n using Eulid's algorithm

$$gcd(x, n) = p \text{ or } q \tag{10}$$

But, the probability of discovering a number x not relatively prime to n is very small for large values of p and q. This can be shown as follows :

The count of numbers which are relatively prime to n is

$$\phi(n) = (p-1)(q-1) = n-(p+q)+1 \tag{11}$$

The count of numbers which are not relatively prime to n is

$$n-\phi(n) = (p+q)-1 \tag{12}$$

The probability of discovering a number x which is not relatively prime to n is

$$\frac{n-\phi(n)}{n} = 1-\frac{\phi(n)}{n} = \frac{(p+q)-1}{pq} = \frac{1}{q}+\frac{1}{p} \tag{13}$$

3. PROTOCOL FAILURES IN RSA CRYPTOSYSTEMS

3.1. Common Modulus protocol Failure

We consider a protocol using RSA. In this system a central authority would generate two primes p and q, calculate n=pq and generate each encryption decryption key pairs $\{e_i, d_i\}$. Each user in the system would be issued a secret key and public key which consists of the common modulus n and the public key e_i. The use of the common modulus poses several problems [2,12]. First, if a message is sent to two users whose public keys (e_i, e_j), then if these public keys are relatively primes, then the message can be recovered without breaking the cryptosystem. To demonstrate this, consider the following :

$$y_i = x^{e_i} \; mod \; n \tag{14}$$

$$y_j = x^{e_j} \; mod \; n \tag{15}$$

Integers r and s can be found using the Euclidean algorithm, so that

$$re_i + se_j = 1 \tag{16}$$

From equation (16), r or s most be negative. Assume that $r < 0$, then $r = -1 \cdot |r|$.
Assume that y_i and y_j are relatively prime to n, so using Euclidean algorithm to calculate the multiplicative inverse of y_i (*mod* n), we can recover the message (x) as follows :

$$\left[y_i^{-1}\right]^{|r|} \cdot \left[y_j\right]^s = \left[x^{e_i}\right]^{-1\cdot|r|} \cdot \left[x^{e_j}\right]^s = x^{re_i+se_j} = x \; mod \; n \tag{17}$$

So, the protocol fails to protect the secrecy of the message (x) sent to two users whose public keys are relatively prime.

The second attack, in the case of common modulus between users, is as follows : The attack involves a probabilistic method for factoring the modulus [2,13]. The basic idea used to factor the modulus is to compute the square root of (1 *mod* n). By this we mean a number (b), satisfying :

(a) $b^2 = 1 \; mod \; n$

(b) $b \neq \pm 1 \; mod \; n$

(c) $1 < b < n-1$

If such a number can be found, then the modulus (n) can be factored in the following way. Since $b^2 = 1 \; mod \; n$, then :

$$b^2 - 1 = 0 \ mod\, n \quad \text{or}$$
$$(b-1)(b+1) = 0 \ mod\, n \quad \text{or} \qquad (18)$$
$$(b-1)(b+1) = sn = spq, \quad \text{for some integer } s$$

However, $1 < b < n$, so that $0 < b\text{-}1 < b+1 < n$. These inequalities make clear that p and q can not divide either (b-1) or (b+1). Hence, the greatest common divisor of (b+1) and n must be p or q.

The third type of failures [2] breaks the cryptosystem by demonstrating that a user can use his own public and private keys to generate the private key of another user. That is given a public key (e_1), the holder of an encryption/decryption pair (e_2, d_2), can find an integer (d_1) such that $e_1 d_1 = 1 \ mod\, \phi(n)$, without knowing $\phi(n)$. To find such (d_1), it is enough to find an integer which is relatively prime to e_1. If $\phi(n)$ and e_1 are relatively prime, then there are integers (r) and (s) satisfying $r\,\phi(n) + s\,e_1 = 1$, then $s\,e_1 = 1 \ mod\, \phi(n)$. Consider the procedure for finding such an integer (d_1) :

1) Using the Euclidean algorithm to find the $\gcd(e_1, e_2 d_2) = f$
2) $d_1 = (e_2 d_2 - 1)/f$

So, the above procedure then yields a decryption exponent (d_1).

3.2. The Low Exponent Protocol Failure

This protocol uses a small exponent for public key in order to make the calculations for encryption fast and inexpensive to perform [2,14]. The protocol specified that the j^{th} user should choose two large primes (p_j, q_j) and publish their product (n_j) as the modulus for an RSA algorithm. We are interested in the case when the encryption exponent (e) is chosen to be small integer. Such specification causes the protocol to fail if the exponent (e) and the same message is sent to at least (e) users. To illustrate the problem, consider the case when e = 3. Suppose that user (1) whose public exponent is 3, decide to sent a message (M) to users 2,3 and 4. The ciphertexts are :

$$C_2 = M^3 \ mod\, n_2 \qquad (19)$$
$$C_3 = M^3 \ mod\, n_3 \qquad (20)$$
$$C_4 = M^3 \ mod\, n_4 \qquad (21)$$

If n_2, n_3, n_4 are relatively prime, the chinese remainder theorem will enable to recover the message (M) from the knowledge of C_2, C_3, C_4.

To overcome this problem, a time stamp is concatenated to the message before encryption, but this solution may not overcome the weakness in the low exponent protocol.

3.3. Notary Protocol

This protocol is designed to allow a message to be signed by an entity A, in a way which allows others to verify at a later date that the message was in fact signed by A, using RSA parameters.

To sign a document (M), the notary uses the private exponent (d) to compute $(S = M^d \ mod\, n)$, and this signature is appended to the document.

Anyone can use the public information to verify that $S^e = M \ mod\, n$. It is possible to use the protocol to obtain a forged signature on the document [2,15]. To do this, the forger arbitrarily choses a value (x) and computes $y = x^e \ mod\, n$. He can use this value to modify the document on which he wants a signature by calculating $M = yp$. The forged signature (S') for (p) can be obtained by calculating $S' = S\,x^{-1}$, since :

$$S = (yp)^d = y^d p^d \ mod\, n = x\,p^d \ mod\, n \tag{22}$$

$$S' = S x^{-1} = p^d \ mod\, n \tag{23}$$

So, the ability of the protocol to produce a signature which could only have been obtained if (A) signed the document, is not attained.

4. CONTINUED FRACTION ALGORITHM APPLIED TO RSA

Continued fraction can be used to find the numerator and denominator of a fraction when a close enough estimate of the fraction is known. In case of RSA algorithm, the public exponent and the modulus will be used to construct an estimate of a fraction involving the secret exponent. The following relation between public exponent (e) and the secret exponent (d) is given in [3] :

$$ed \equiv 1 (mod\ LCM(p-1, q-1)) \tag{24}$$

$$ed = K \cdot LCM(p-1, q-1) + 1 \tag{25}$$

If we let G = gcd(p-1, q-1), since (gcd) means the greatest common divisior, and use the fact that

$$LCM(p-1, q-1) = (p-1)(q-1)/G \tag{26}$$

From equation (25), we get

$$ed = \frac{K}{G}(p-1)(q-1) + 1 \tag{27}$$

It is possible for K and G to have common factors. Let us define

$$k = \frac{K}{gcd(K,G)} \quad \text{and} \quad g = \frac{G}{gcd(K,G)} \tag{28}$$

So,
$$ed = \frac{k}{g}(p-1)(q-1) + 1 \tag{29}$$

Dividing through by (dpq) in (29) gives

$$\frac{e}{pq} = \frac{k}{dg}(1-\delta) \tag{30}$$

$$\delta = \frac{p+q-1-(g/k)}{pq} \tag{31}$$

(e/pq) is a close underestimate of K/dg.

Let $f = \frac{e}{pq}$, then the continued fraction expansion of a positive rational number (f) is formed by subtracting away the integer part of (f) and repeatedly inverting the remainder and subtracting away the integer part until the remainder is zero as follows :

$$q_o = \lfloor f \rfloor, \qquad r_o = f - q_o \qquad \text{and} \tag{32}$$

$$q_i = \left\lfloor \frac{1}{r_{i-1}} \right\rfloor, \qquad r_i = \frac{1}{r_{i-1}} - q_i \qquad \text{for } i = 1,2,\ldots,m \tag{33}$$

where $\lfloor x \rfloor$ means the integer part of x.

Then the contined fractions expression is :

$$\langle q_o, q_1, \ldots, q_m \rangle = q_o + 1/\left(q_1 + 1/\left(q_2 + 1/\left(\cdots/\left(q_{m-1} + 1/q_m\right)\cdots\right)\right)\right) \tag{34}$$

Let f be an understimate of f' :

$$f = f'(1-\delta) \qquad \text{for some } \delta \geq 0 \tag{35}$$

To construct the fraction f', let n_i and d_i, i = 0,1,....,m be a sequence of numerators and denominators defined as follows :

$$\frac{n_i}{d_i} = \langle q_o, q_1, \ldots, q_i \rangle, \qquad \gcd(n_i, d_i) = 1 \qquad \text{for } i = 0,1,\ldots,m \tag{36}$$

It can be shown that

$$n_o = q_o, \qquad d_o = 1$$

$$n_1 = q_o q_1 + 1, \quad d_1 = q_1$$

$$n_i = q_i n_{i-1} + n_{i-2}, \qquad d_i = q_i d_{i-1} + d_{i-2} \qquad \text{for } i = 2,3,\ldots,m \tag{37}$$

Then $f' = \frac{n_m}{d_m}$ and $\delta < \frac{1}{3/2\, n_m d_m}$ (38)

$f' = \frac{k}{dg}$. From equation (29), we get

$$edg = k(p-1)(q-1) + g \tag{39}$$

Dividing (edg) by (k) yields a quotient of (p-1)(q-1) and a remainder g as long as k > g. This provides a guess of (p-1)(q-1) and of q. If the guess of (p-1)(q-1) is zero, then the guess of (k) and (dg) are wrong. The guess of (p-1)(q-1) can be used to create a guess of (p+q)/2 using :

$$\frac{pq - (p-1)(q-1) + 1}{2} = \frac{p+q}{2} \tag{40}$$

If the guess of (p+q)/2 is not an integer, then the guess of k and dg are wrong. The guess of (p+q)/2 can be used to create a guess of $((p-q)/2)^2$ using :

$$\left(\frac{p+q}{2}\right)^2 - pq = \left(\frac{p-q}{2}\right)^2 \tag{41}$$

If the guess of $((p-q)/2)^2$ is a perfect square, then the guess of K and dg is correct. Then, the secret exponent (d) can be found by dividing (dg) by (g). We can also recover (p) and (q) from (p+q)/2 and (p-q)/2. To reduce the maximum size of secret exponent that can be found using the continued fraction attack on RSA, using equation (3), adding a multiple LCM(p-1, q-1) to public exponent e. From equation (31), (38), we deduce that

$$k\, dg < \frac{pq}{3/2(p+q)} \tag{42}$$

Then, if e increases, then k increases and the final result is decreasing exponent d.

5. RESULTS

We shall state the guidelines for the development of sound protocol of RSA cryptosystem. These guidelines are extracted from the analysis of the different demonstrated attacks on RSA algorithm.

We recommend that the modulus n be about 200 decimal digits long. Longer or shorter lengths can be used depending on the relative importance of encryption speed and security in the application at hand. The modulus n must be composite odd number $n = pq$, since p,q are two large prime numbers randomly generated, $\phi(n)$ is trivial to compute if n is prime. Decryption key is never printed out even for its owner, but used to decrypt messages. The decryption key must be erased if the cryptosystem is tampered.

A common modulus should not be used in a protocol using RSA in a communication network. The protocol designers should consider the following collections of ciphertexts and analyzing their effects on security.

1) A collection $E_i(M)$, where various keys are used to encrypt the same message, specially if the keys are related.
2) A collection $E(M_i)$, where the same key is used to encrypt messages M_i which satisfy if the keys are related.
3) A collection of $E_i(M_i)$, where various keys are used to encrypt various variation of the same message.

To reduce the maximum size of secret exponent that can be found using the continued fraction attack on RSA, we add a multiple of LCM(p-1, q-1), where LCM means the least common multiple, to the public key e. To avoid a forged signature using RSA algorithm, the document must possess certain predetermined structure (hash value).

6. CONCLUSIONS

RSA cryptosystem is one of the strong cryptosystems to secure data communication in a computer network since different approaches for breaking RSA algorithm are at least as difficult as factoring the modulus n.

If RSA cryptosystem resists all the demonstrated attacks for a sufficient length of time, it may be used with a reasonable amount of confidence.

In designing the RSA cryptosystem protocol, we must avoid the common modulus and low exponent cases.

As the secret key increases in size, the time required to find the secret exponent using the continued fraction attach increases exponentially.

REFERENCES

[1] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystem", Communication of ACM, Feb. 1978.

[2] Judy H. Moore, "Protocol failures in crptosystems", Proceedings of the IEEE, Vol. 76, No.5, May 1988.

[3] Micheal J. Wiener, "Cryptanalysis of short RSA secret exponent", IEEE Transaction On Information Theory, Vol.36, May 1990.

[4] Carl H. Meyer & Stephen M. Matyas, "A new dimension in Computer data security", Jon Wiely & sons, 1982.

[5] Carl Pomerance, J.W. Smith and Randy Tuler, "A pipeline arechitecture for factoring large integers with the quadratic sieve algorithm", Siam J. Comput., Vol. 17, No.2, April 1988.

[6] H.W. Lenstra, "Factoring integers with elliptic curves", Ann. of Math. 126 (1987).

[7] C.P. Schnorr and H.W. Lenstra, "A monte carlo factoring algorithm with linear storage", math. Comp., 43, (1984).

[8] C. Pomerance, "Analysis and comparison of some integer factoring algorithms", Computational Methods is Number Theory, H.W. Lenstra, and R. Tijdeman, Math. Centrum Tract, 154, 91982).

[9] C. Pomerance, "The quadratic sieve factoring algorithm", Advances in cryptology, Lectures Notes in Computer Sience, 209, 1985.

[10] D, Copper Smith, A.M. Oldyzko and R. Schroppel, "Discrete Logarithm in GF(P)", Algorithmica, (1986).

[11] M.A. Morrison and J. Brillhart, "A method of factoring and the factorization of F7", Math. Comp., 29, (1975).

[12] G.J. Simmons, "A weak privacy protocol using the RSA cryptoalgorithm", Cryptologia, Vol.7, 1983.

[13] J.M. Delaurentis, "A further weakness in the common modulus protocol for the RSA cryptoalgorithm", cryptologia, Vol.8, No.3, July 1984.

[14] J. Hasted, "On using RSA with low exponent in a public key network", in Advances in Cryptology-Proc. Crypto 85, (1985).

[15] D.E. Denning, "Digital signature with RSA and other public key cryptosystems", In Comm. of the ACM, Vol.27, April, 1984.

[16] L. Harn, "Public key crptosystem design based on factoring and discrete logarithms", IEE Proc. Comput. Digit. tech., Vol.141, May 1994.