Int. J. Advanced Networking and Applications
Volume: 03, Issue: 05, Pages: 1362-1366 (2012)

An effective Method for Attack RSA Strategy

Vibhor Mehrotra
Assistant Professor Department of Computer Science, SSVIT, BareillyUP-243001
Email: vibhorms@gmail.com
Prakash Singh Rana
Lecturer, Department of Computer Science, SIMT Kashipur- UK
Email:-prakashranas@gmail.com

----------ABSTRACT----------
The protection on many public key encoding schemes depended on the intractability of detecting the integer factoring problem such as RSA scheme. However, there are great deals of researches regarding the RSA factoring modulus compared with the other type of attack the RSA scheme. So the need for more methods of attacks other than RSA factoring modulus to find an effective and quicker algorithm to solve this problem is still crucial.
This paper introduces a new algorithmic program which approaches the RSA scheme. The suggested algorithm aims to find the private key of the RSA scheme and then factoring the modulus based on the public key of the RSA scheme. The new idea exacted to be more efficient than the already existed algorithms particularly when the public key is small, since most of public key encryption schemes select a small public encryption key e in order to improve the efficiency of encryption. Also, the suggested algorithmic program is more effective since it is faster and takes less running time.
Keywords: **Public key cryptography, RSA scheme, factoring problem, RSA attack scheme**

Date of Submission: October 11, 2011 Revised: December 20, 2011 Date of Acceptance: December 26, 2011

1. Introduction

Public key cryptography is one of the numerical application program that are valuable in sending selective information via insecure channels, which is considered as the worse case used in the e-commerce and internet today. However, there are some algebraic assumptions which are regarded to be an important key in this issue such as prime numbers and integer factoring problem.

Factoring an integer modulus n means find its prime numbers p and q. However, factoring the modulus is in fact a hard problem and most of the democratic public key cryptography schemes are relied on [1], but surly not impossible because the RSA-120 is factored using quadratic sieve by Thomsan, Bruce, Arjen and Mark [2]. Also, the RSA-140 is factored applying number field sieve by Cavallar, Dodson, Lenstra, Leyland, Lioen, Montgomery, Murphy and Zimmermann [3]. While RSA-155 is factored in 1999, also, the RSA-160 is factored in April 2003, and the RSA-576 is factored in December 2003 by Eric [4]. The RSA-200 is factored in 2004; the RSA-640 is factored in November 2, 2005 by Bahr, Boehm, Franke and Kleinjung [5] and verified by RSA Laboratories. The relation among factoring and the public key encryption schemes is one of the main reasons that researchers are interested in factoring algorithms [6].

In 1976 Diffie-Hellmam [7] creates the first radical research in public key cryptography via presented a new idea in cryptography and to challenge experts to generate cryptography algorithms that faced the requirements for public key cryptosystems. However, the first reaction to the challenge is introduced in 1978 by RSA [8]. The RSA scheme is a block cipher in which the original message and cipher message are integer values in the interval $[0..n-1]$ where n a composite modulus.

In this paper we accept the public key (e,n) only to disclose the original modulus from the RSA scheme. However, the message in RSA scheme is encrypted in blocks after separate it to blocks, every block must convert to a value smaller than the modulus n. The intractability of the RSA assumption forms its security. The RSA assumption is the difficulty of solving the integer modulus n which is a product of two distinct odd large primes p and q with an assistance of another public key e and an integer cipher text c [9].

In other words, the RSA difficulty is that of solving e^{th} roots mod a composite modulus n. The conditions determined the modulus n and the public key e are to guarantee that for every integer $c \in (0,1,...,n-1)$ there is just one $m \in (0,1,...,n-1)$ where $m^e = c \bmod n$. However, the RSA scheme is the most employed public key encryption compared with the other schemes. It can be hired for both encryption and digital signature schemes.

2. Security of RSA

This section introduce a security issue related to RSA encoding scheme, based on the small decoding key d

that we will discuss, as well as an appropriate measures to counteract the threat.

But before we discuss the efficiency of decryption, it is necessary to give a belief idea concerning the efficiency of encryption related to public key. it is enviable to select a small public key encryption as e=3. When an encryption public key e is selected arbitrarily, then the RSA encryption scheme employing the repeated square and multiply method takes k modular squaring and an expected $k/2$ less with optimizations, modular multiplication, where k is the size string length of the modulus n. Then encryption algorithm can be accelerate via choosing the encryption public key e as small as possible or via choosing the public key e with a small number of 1's in its binary representation. The encryption public key e=3 is generally used in scheme. In this situation, it is essential that both $p-1$ and $q-1$ is divisible via 3. This affords a very fast encryption operation because it just needs 1 modular squaring and 1 modular multiplication.

As is the case with the encryption exponent e, it may seem describe to select a small decryption exponent d in order to improve the efficiency of decryption. In this case, one would select d first and then compute e in RSA key generation algorithm for public key encryption as describe in section 4. However, if $\gcd(p-1, q-1)$ is small, as is typically the case, and if d has up to approximately one-quarter as many bits as the modulus n, then there is an efficient algorithm for computing d from the public information (n, e). This algorithm can not be extended to the case where d approximately the same size as is n. Hence, to avoid this attack, the decryption exponent d should be roughly the same size as modulus n.

3. Related Work

The RSA cryptography scheme was brought in in 1977 by Rivest, Shamir and Adleman [8]. Kaliski and Robshaw [10] give an outline of the main attack methods on RSA public key encryption and digital signature schemes, and the practical methods of counteracting these methods of attacks.

The computational correspondence of computing the decrypted key d and then factoring the modulus n was shown by RSA based on earlier work done by Miller [11]. The attack on RSA with small encryption public key is discussed by Håstad [12] who illustrated that sending an encryption of more than $e(e+1)/2$ linearly related messages of the type $(a_i * m + b_i)$, where a_i and b_i are known allows an adversary to decrypt the messages provided that the modulus n_i satisfy $n_i > 2^{(e+1)(e+2)/4} * (e+1)^{(e+1)}$.

On the other hand, the attack on RSA with small decryption exponent d is due to Wiener. [13]. Wiener showed that his attack can be avoided if the encryption exponent e is chosen to be at least 50% longer than the modulus n. In this case, d should be at least 160 bits in length to avoid the square root discrete logarithm algorithm such as Pollard's rho algorithm [14] and the parallelized variant of Menzes, Oorschot and Vanstone [15]. However, it was noted by [16, 17, 18, 19] that these theoretical bounds on d are not correct since some quantity which appears in the analysis is not negligible.

In this paper we suggest an efficient algorithm to break the RSA scheme. Through define a functional problem of attack taking in its account the low decryption key of the RSA scheme. But before that we are going to discuss the RSA scheme.

4. RSA Scheme

In 1978, RSA [8] developed a public key cryptosystem that is based on the difficulty of integer factoring. The RSA public key encryption scheme is the first example of a provably secure public key encoding scheme against chosen massage attacks. Assuming that the factoring problem is computationally intractable and it is hard to find the prime factors of $n = p * q$. The RSA scheme is as follows:

2.1 Key generation algorithm

To generate the keys entity A must do the following:

1. Randomly and secretly choose two large prime numbers p and q with equally likely.
2. Compute the modulus $n = p * q$.
3. Compute $\theta(n) = (p-1)(q-1)$
4. Select random integer $e, 1 < e < n$ where $\gcd(e, \theta) = 1$
5. Use Baghdad method Baghdad Method for Calculating Multiplicative Inverse [20] to compute the unique decrypted key $d, 1 < d < \theta(n)$ where $e * d \equiv 1 \bmod \theta(n)$
6. Determine entity A public and private key. The pair (d, θ) is the private key. While the pair (n, e) is the public key.

2.2 Public key encryption algorithm

Entity B encrypts a message m for entity A which entity A decrypts.

2.3 Encryption: entity B should do the following:

- Obtain entity A's public key (n, e).
- Represent the message m as an integer in the interval $[0...n-1]$
- Compute $c = m^e \bmod n$
- Send the encrypted message c to entity A

2.4 Decryption: To recover the message m from the cipher text c. Entity A must do the following:

- Obtain the cipher text c from entity B
- Recover the message $m = c^d \bmod n$

Example

Key generation: suppose that entity A selects the prime numbers $p = 23$ and $q = 71$. Then he finds the RSA modulus $n = p*q = 23*71 = 1633$ and $\theta(n) = (p-1)(q-1) = 1540$. Entity A chooses $e = 23$ and using the Baghdad method for multiplicative inverse [20] to find the decrypted key $d = 67$ where $e*d \equiv 1 \bmod \theta$. So $A's$ public key is the pair $(n = 1633, e = 23)$ while entity $A's$ private key is $(\theta = 1540,\ d = 67)$.

Encoding: Suppose entity B obtain $A'S$ public key $(n = 1633)$ and he determines a message $m = 741$ to be encrypted, entity B uses repeated square and multiply algorithm [21] of modular exponentiation to compute $c = 741^{23} \bmod 1633 = 1109$ and send this $c = 1109$ to entity A.

Decoding: To recover and obtain the original message m entity A should first obtain $c = 1109$ from entity B then recover the message $m = c^d \bmod n = 1109^{67} \bmod 1633 = 741$ using repeated square and multiply algorithm [21] for exponentiation.

5. The Proposed Attack Algorithm

In this section, we address the following question: is there a possible attack on the RSA cryptosystem other than factoring n. The answer is that yes there are few methods that attack the RSA scheme that does not involve finding the factoring of the modulus n but most of them carrying some deficiencies.

We will now prove the very interesting result that, as long as the exponent key e is known, then n can be factored in polynomial time by means of a randomized algorithm [6, 22]. Therefore we can say that computing this method is no easier than factoring n. However, this does not rule out the possibility of breaking the RSA cryptosystem without involving e. Notice that this result is of much more than theoretical interest.

In this paper we proposed a method that breaking the RSA scheme based on the knowing public key (e, n). This method will work efficiently if the decryption key d is small. It is possible therefore, to factor the modulus n.

Algorithm

Suppose entity A has public key $(n = 8927, e = 2621)$ where entity A private key $(d = 5, p = 113, q = 79)$. Then entity B encrypts the message m so the cipher text $c = 6948$ then he sent it to entity A. Suppose that the attacker entity T obtain the encrypted message $c = 6948$. By using the following algorithm entity T will recover the original message m. The steps of the proposed algorithm are as follows:

1. Obtain entity A public key (e, n)
2. Compute q_i and r_i as follows:
   a. $f = (e/n)$
   b. $q_0 = \lfloor f \rfloor, r_o = f - q_0, m = 0$
   c. *for* $i = 1,2,...$
   $\{m = m+1;$
   $q_i = \lfloor 1/r_{i-1} \rfloor$
   $r_i = (1/r_{i-1}) - q_i$
   if $(q_i = q_{i-1})$
   break$\}$
   $f = (q_0, q_1, ...q_n)$
3. Compute $k/(d*g)$ as follows:
   for $i = 2,3,..., m$ (Obtain m from step 2)
   {
   if (i is even)
   (* reconstruct f using algorithm number 1below with the following expansion $q_0, q_1, ..., q_{i-1}, q_{i+1}$ *)
   if (i is odd)
   (* reconstruct f using algorithm number 1 below with the following expansion $q_0, q_1, ..., q_i$ *)
   }
   So, $k/(d*g) = (n_m / d_m)$, in each iteration. Where $(k = n_m, d*g = d_m)$
4. Compute $d = (d*g)/d$. This will lead to break and obtain the message.

Algorithm Number 1

To reconstruct f from its expansion starting from q_0 and let n_i and $d_i, = i = 0,1,..., m$ is a sequence of numerators and denominators, so f can be computed as follows:

$n_0 = q_0, d_0 = 1$

$n_1 = q_0 q_1 + 1, d_1 = q_1$

for $i = 2,3,..., m$ (Obtain m from step 2)

$\{\ n_i = q_i n_{i-1} + n_{i-2}$

$d_i = q_i d_{i-1} + d_{i-2}$

$\}$

and this will lead to $f = n_m / d_m$

Example

1. Suppose that the public key ($e = 2621, n = 8927$)
2. Find q_i and r_i

2.1. $f = 2621/8927$

2.2. $q_0 = \lfloor 2621/8927 \rfloor = 0, r_0 = (2621/8927) - 0$
$= 2621/8927, m = 0$

2.3. $q_i = (0,3,2)$,
$f = (0,3,2)$,
$r_i = (2621/8927, 1064/2621, 493/1064), m = 2$

3. Compute $k/(d*g)$ as follows:

$k_0/(dg)_0 = n_0/d_0 = 1/1$, Where $k_0 = 1, (dg)_0 = 1$

$k_1/(dg)_1 = n_1/d_1 = 1/3$, Where $k_1 = 1, (dg)_1 = 3$

$k_2/(dg)_2 = n_2/d_2 = 1/10$, Where $k_2 = 3, (dg)_2 = 10$

4. Compute $d = (d*g)/d$ as follows:

First compute g_i

$g_0 = e(dg)_0 \bmod k_0$
$= 2621*1 \bmod 1 = 0$

$g_1 = e(dg)_1 \bmod k_1$
$= 2621*3 \bmod 1 = 0$

$g_2 = e(dg)_2 \bmod k_2$
$= 2621*10 \bmod 3 = 0$

Second compute d_i

$d_0 = (dg)_0 / g_0 = 1/0$ Does not applicable

$d_1 = (dg)_1 / g_1 = 3/0$ Does not applicable

$d_2 = (dg)_2 / g_2 = 10/2 = 5$ Catch solution

So $d = 5$. To decrypt the message m we use the following equation $m = c^d \bmod n$

$= 6948^5 \bmod 8927$
$= 1619194607\,7606355968 \bmod 8927$
$= 3$

6. Discussion

In this section, we will discuss the efficiency of the suggested attack method, trying to provide a clear idea to the reader about what it would take and what the requirements should be available for the proposed method.

To show the viability of the proposed scheme we have executed the key generation algorithm and measured the average running time for it. The laptop computer with 3.0 GHz CPU and 2 GB RAM is used. The algorithms were executed using C++ programming language and utilizing tools on the Windows operating system. For the proposed scheme we ran the key generation algorithm 100 times finding an average key generation time of only 93 ms compare with 1342 seconds of the RSA- small- e scheme. While the average numbers of iterations for each loop (steps 2-4) were 842 compare with an average number of iterations for each loop was 37656 ($\approx 2^{16}$) in the RSA-small- e scheme. In our execution the complete prime factoring of p and q was not computed. Instead, we only try to find small factors until the desired factoring $(P = k_{p_2}(p-1))$ is found. Thus the key generation for this scheme is efficient. Of course, a decryption time for the proposed scheme is longer than decryption time for the RSA-small- e scheme. Thus, there is a trade-off between the proposed scheme and the RSA-small- e scheme.

In Table 1, we compare an example of the proposed scheme with the other RSA variants in terms of the complexity of encryption and decryption for a 1024-bit modulus with security parameter $m = 80$. Suppose that a random $k-bit$ exponent will require $1.5*k$ multiplications. Suppose also that decryption is always executed using the Chinese Remainder Theorem. The complexity for each operation (encryption and decryption) is the expected number of bit operations. To compute $Z^a \bmod b$ for a random exponent a, we expect $1.5*|a|*|b|^2$ binary operations [10]. When the exponent is of the special form of $2^k + 12$, then the number of binary operations will be reduced to $(k+1)*|b|^2$ bit operations.

In the case of the proposed scheme, we use public exponents of the form $2^k + 1$. For the proposed scheme we use the example $(n_e, n_d) = (592,190)$. The example satisfies each of the security conditions listed in section 5. Compared to the original RSA-CRT our example for the proposed scheme encrypts about 2.6 times faster while decryption is about 1.2 times slower.

Table 1: Comparisons of the different RSA schemes in terms of encryption and decryption times for $n = 1024$ Bit modulus

Scheme name	RSA-small-e	RSA-short-e	RSA-CRT	Proposed
Public Key	$e = 2^{16} + 1$	n bits	n bits	$e = 2^{591} + 1$
Number of Multiplications	$0.0166 * n$	$1.5 * n$	$1.5 * n$	$0.58 * n$
Private Key	n bits	> 350 bits	> 160 bits	358 bits
Complexity	$17 * n^2$	$1536 * n^2$	$1536 * n^2$	$592 * n^2$

7. Conclusion

We are introduced a new algorithm for attacking RSA scheme. The new algorithm aims to obtain the factoring the modulus based on the small private key d of the RSA scheme. The new attack method is amazingly effective under certain circumstances, but rendered important with relative ease especially with large number of iterations. All to do is just to ensure that proper bounds of public key d are placed on. We claim that the proposed algorithm is more efficient than the already existed algorithms of attack since it is faster and takes less running times.

References

[1] Bonteh S, "Twenty Years of Attacks on the RSA Cryptosystem", Notices of the American Mathematical Society, 46(2):203-213, 1999

[2] Thomsan D. Bruce D. Arjen L. and Mark M., "On the Factoring of RSA-120", (169), pp. 166-174, 1994

[3] Cavallar S, Dodson B, Lenstra A, Leyland P, Lioen W, Montgemery P, Murphy B, and Zimmermann P, "Factoring of RSA-140 using the number field sieve", 1999

[4] Eric W "Prime Factorization Algorithm", Mathworld.woiframe.com/news/ 2003

[5] Bahr F, Boehm M, Franke J and Kleinjung T, "For the Successful Factorization of RSA-200" www.rsasecurity.com

[6] Coron J. S. and May A., "Deterministic polynomial time equivalent of computing the RSA secret key and factoring", Journal of Cryptology, 20(1):39-50, January 2007.

[7] Diffie W and Hellman M, "New Direction in Cryptography, IEEE Transaction on Information Theory", IT-22(6): 644-654, 1976

[8] Rivest R, Shamir A and Adelman L, "A Method for Obtaining Digital Signature and Public Key Cryptosystems", Communications of the ACM, 21, pp. 120-126, 1978

[9] Sarkar, S. Maitra, and Sarkar S., "RSA cryptanalysis with increased bounds on the secret exponent using less lattice dimension", Cryptology ePrint Archive, Report 2008/315, 2008. http://eprint.iacr.org.

[10] Kaliski B.S. and Robshaw, M.J.B. "Linear Cryptanalysis Using Multiple Approximation", Advances in Cryptology-CRYPTO '94 Proceedings, Springer-Verlag, 1994, pp.26-39.

[11] Miller G.L., "Riemann's Hypothesis and Tests for Primality", Journal of Computer Systems Science, volume13, Number 3, December 1976, pp. 300-317

[12] Håstad J., "On Using RSA with low exponent in a public key Network", Advances in Cryptology –CRYPTO '85, Springer-Velag LNCS 218, pp. 403-408, 1986

[13] Wiener M., "Cryptanalysis of short RSA secret exponents", IEEE Trans. Inform. Theory 36 (1990), 553–558.

[14] Pollard J. M, "Monte Carlo, Methods for Index Computation mod p", Mathematics of Computation, 32(1978), 918-924

[15] Menzes A, Oorschot P van and Vanstone S, "Handbook of Applied Cryptography", CRC, 1996.

[16] Hinek, M. J., "Low Public Exponent Partial Key and Low Private Exponent Attacks on Multi-prime RSA", Master's thesis, University of Waterloo, 2002.

[17] Hinek, M. J., Teske, E., "On some attacks on multi-prime RSA", Proceedings of SAC 2002, Lecture Notes in Computer. Science 2595 (2003), 385–404.

[18] Hinek M. J., "On the security of multi-prime RSA", Journal of Mathematical Cryptology, 2(2):117-147, 2008.

[19] Jason M., Hinek1 and Charles C. Y. Lam2, "Common Modulus Attacks on Small Private Exponent RSA and Some Fast Variants (in Practice)", Cryptology ePrint Archive, Report 2009/037, 2009. http://eprint.iacr.org

[20] Sattar Aboud, "Baghdad Method for Calculating Multiplicative Inverse", International Conference on Information Technology, Las Vegas, Nevada, USA. pp: 816-819, 2004

[21] Lam K. and Hui L, "Efficiency of square-and-multiply exponentiation algorithms", Electronics Letters, Vol. 30, Issue 25, pp.2115- 2116, 1994

[22] Itoh, K., Kunihiro N., and Kurosawa K., "Small secret key attack on a variant of RSA", (due to Takagi). In T. Malkin, editor, CT-RSA 2008, volume 4964 of Lecture Notes in Computer Science, pp. 387-406, 2008.