World of Computer Science and Information Technology Journal (WCSIT)
ISSN: 2221-0741
Vol. 1, No. 6, 260-263, 2011

A Fast Approach for Breaking RSA Cryptosystem

[1]Prof. Dr. Alaa H Al-Hamami
Dean of the Computer Sciences and Informatics College
Amman Arab University
Amman - Jordan
alaa_hamami@yahoo.com

[2]Bilal S O Al-Kubaysee
Computer Sciences and Informatics College
Amman Arab University
Amman - Jordan
belal_sadeq@yahoo.com

Abstract— Prime numbers play a very important role in the complexity and security of the public key cryptosystem. RSA is one type of public key algorithm and its security depends on the complexity of factoring (n) value. Any encryption algorithm depends on the length of the key and the computational effort required breaking the key.
This paper introduces an efficient algorithm to attack the RSA Scheme. Obtaining the private key of the RSA scheme is the target of the suggested algorithm by factoring the modulus based on the public key (*e, n)* of the RSA scheme. The suggested algorithm is very fast due to its treatments for the factorizing problem. It will limited the search for the p & q values especially when the value of *n* is small, since most of public key encryption schemes select a small encryption *n* in order to improve the efficiency and reliability of encryption. The suggested algorithm is more efficient than most existed algorithms of attack since it is break the search process and takes less running time.

Keywords - RSA Scheme; Factoring; GCD; Attack; Prime Numbers.

I. INTRODUCTION

The first proposed code in public key was by Diffie and Hellman in 1976, which is the first truly revolutionary development in encryption for thousands of years. The first form of public key algorithm is the RSA, which developed by Rivest – Shamir and Adlman by MIT in 1977 [1].
Any Encryption algorithm depends on: the length of the key and the computational effort required to break the key. RSA algorithm is a quantum code where the plain text and cipher text are numbers located between zero and the result of multiply two prime numbers that almost be a great numbers.
RSA complexity and security rely on the prime numbers (p & q), which they can have long numbers to make it very difficult for the attacker to factorize the value of (n) and getting p & q values (n = p*q).

II. THE PROBLEM

Rivest, Shamir and Adelman (RSA) crypto system [1] works as the following:

1. We must choose two un-equal prime numbers (p & q); these numbers should be as large as possible.
2. $n = p * q$.
3. $\phi(n) = (p - 1) * (q - 1)$.
4. We choose e (public key) between 1 and < ϕ (n) and must satisfy the following:

$$GCD(\phi(n), e) = 0.$$

5. To encrypt the plaintext M, we use the following equation:

$$C = M^e \bmod n$$

6. We extract the private key as the following:

$$d * e \equiv 1 \bmod \phi(n)$$

7. To decrypt the cipher text C, we use the following:

$$M = C^d \bmod n$$

8. The two keys are the following:
   Public Key = (*e, n*), Private Key = (*n, d*)

The weakness of RSA is to factorize (n) which is the product of two prime numbers (p & q). Due to the large prime numbers (p & q) that used for producing (n), it is very difficult to factorize (n). Several researchers tried to build strong method to find the composite numbers of value (*n)*, but they have not complete successes and the reason because the size of the number (*n)* is every time getting larger and larger.
In this research we suggest a new method for factoring the (n) number in order to get *p* and *q* and then calculate the private key (*d*).

III. LITERATURE SURVEYS

Aboud and AL-Fayoumi [2] tried analysis the algorithm reversely. By other way, find (*d)* value firstly, the *Euler*, and

then prime numbers (*p, q*) by converting the number type of *n* value to binary and assuming some equations to define the key. By applying this way, we will need many very big loops (like Repeat *k* from 1 to *e* until p2 - $s * p + n \equiv 0 \bmod 2^b$ is true) with much complex computing processes. They consider b represents the number of bits of *n*, *s* that has been computed from $ed \equiv 1 + k(n - s + 1) \bmod 2^b$. The approximated time is too long, and that could be inefficient and inapplicable, because one of best of the RSA properties is the speed of key exchange where the attacker has not an opportunity and time to find.

Hastad [4] made an attack on RSA with small key by 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; allowing an adversary to decrypt the messages provided that the Modulus n_i satisfy $n_i > 2^{(e+1)(e+2)/4} * (e+1)^{(e+1)}$.

Coppersmith [3] introduced a new type of attacks on RSA which capasitate a passive adversary to recover such message from the corresponding cipher text. This attack is of practical importance since many public key encryption schemes have been proposed which require the encryption of polynomial related messages. For instance include the key distribution protocol of Tatebayashi, Matsuzaki, and Newman.

Wiener [5] is an attack hinges about find the *d* value directly with special case of *d*, the RSA secret exponent *d* is chosen to be small compared to the RSA modulus *N*. A well-known attack on RSA with low secret-exponent *d* was given by Wiener about 15 years ago. Wiener showed that using continued fractions, one can efficiently recover the secret-exponent *d* from the public key (*n, e*) as long as $d < n^{1/4}$. Interestingly, Wiener stated that his attack may sometimes also work when *d* is slightly *larger* than $n^{1/4}$.

IV. THE PROPOSED METHOD

As we mentioned n = *p* * *q*, so if we discover the two prime numbers, this will lead us to gain the key *d*. When we discover any one of q & p, it is so easy to get the other number. It is known that, (*n)* and (*e)* are representing the public key. From this point we start to construct the mathematical equation as in the proposed Algorithm.

Now, we present the proposed algorithm, and then we show an example to view how the proposed algorithm is applied. The following algorithm is in Figure (1).

1. Find the square root of (n).
2. Collect prime numbers from 1 till the square root of (n).
3. Pick up first prime numbers that its length is the same length of square root of (n).
4. Collect the numbers from the first picked up number to last prime numbers in the array.
5. Fulfill $k = w(L \in N)$ statement.
6. Fulfill (*n* modular *k* = 0) equation.
7. Apply the equation GCD (value, n) = 0.
8. Find the prime number p or q.
9. Find p or q by this equation = n/q or n/p.
10. Find *p* value
11. Calculate *q* value

Figure. I: THE PROPOSED ALGORITHM BLOCK

Now, take an example for each step that shown in Algorithm of Figure (1);

1) Compute the square root of (*n)*.

Suppose we have the value of (n) which is equal to 1457. Now, the square root of (n) is almost = 38, and this value will be called sqt *of (n).*

2) Find all the prime numbers, which falls between [1and square root of (*n*)], and put them in array called (*v)*. This interval will be:

v = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37].

3) Find the first number that has length exactly as the length of square root number in step 1from *v* array (suppose it is *w*). As it mentioned at step 3 of the algorithm, then *w* = 2.

4) From (*v)* array, collect all numbers that start with (*w)* (length = 2) till last number of (*v)* (suppose it is *G*). On the safe side, collect the remaining numbers that except *G* to make feed back if there is no true answer of later step 6 (suppose it as *G'*) in the Algorithm. Following – up the example, the array will be: *G* = [11, 13, 17, 19, 23, 29, 31, 37].

5) Find the numbers that testify the following statement:

$k = G(L \in N)$, Where:

L = last digit of the prime numbers in *G* array elements,

N = set of numbers that if multiplied together, the first number of these results will be the same number that at last digit of *n* value. So, we can classify these sets into two collections as the following:

N1 = [1, 3, 7] when the last digit of *n* value will be 1, 3 or 7.

N2 = [1, 3, 5, 7] when the last digit of *n* value will be 5.

For the prime numbers that held 9 in the last digit of them will go to *G'*.

In other words, we can describe $k = (L \in N)$ statement as collection in *k* - as the residual numbers of probabilities database elements - all numbers that held the last digit they included into N, where N had been classified.

Note the results of each step and regard how much we are decreasing the space of search area to reach to desired factors with lowest range of probabilities for these factors.

As it displayed of the proposed algorithm, the variables of (*L)* of the example will be defined as the following:

L = [2, 3, 5, 7, 1, 3, 7, 9, 3, 9, 1, 7] and *N* = [1, 7], Then, *k* = [11, 17, 31, 37].

In this work we tried a method that aims at reducing the unwanted prime numbers as possible as to increase the possibility of a successful attack.

6) Find the indexes of special numbers that only investigated with the following equation: indexes = (*n* modular *k* = 0). The meaning is to save the locations for any prime number when the remainder with *n* values = 0 in array (*k*). So, the *index* value of the example is 3, (k=31).

7) The result is the smallest prime number (*p* or *q*) constructing *n*. this number is unique and corresponding to one index location.

As mentioned before, any value of (*n)* is constructed by multiplication of a pair of prime numbers *p* & *q*. Since any two prime numbers have no common factor, the result will be finding one part of *n*. Following – up the example, the value that hold index = 3 as in *k* array is 31.

8) Finally, since number 31 is define the *p*, immediately the *q* = $\frac{n}{p}$, q = 47.

V. EXPERIMENTAL WORKS

By using the suggested Algorithm to factorize the value of (n), we apply a collection of variant lengths of presuming (*n)* numbers on the proposed method. From this examination, we got the results as follows, Table (1).

It is a fact that length of p & q is getting longer and longer and this will make this type of attack harder and harder, due to the limitation of the machines and the processing time. So, we make an elapsed time equation that was evaluated and derived from the results that we got in Table I. The equation was:

$$T(n_k) = T(n_{k-1}) + (n_k - n_{k-1}) \times A$$

TABLE I. SAMPLE OF TRIAL RUN RESULTS OF THE PROPOSED METHOD

n value	p	q	Time in second
35	5	7	0.000000
5141	53	97	0.000000
191869	213	613	0.000000
10241339	2003	5113	0.000000
6498478561	77773	83557	10×10^{-3}
93767667359	120557	777787	18×10^{-3}
547014990083	666637	820559	30×10^{-3}
1524212467931	1234577	1234603	37×10^{-3}
45586366129943	8205529	5555567	43×10^{-3}
4694434085162951	55555573	84499787	55×10^{-3}
572994973100356993	667982233	857799721	64×10^{-3}
7724931571923045853	1953799537	3953799469	71×10^{-3}

This equation used to simulate the next lengths of (*n)* values to testify how the result of the proposed algorithm is if length of (*n)* was more than 100 digits. As we know, the current operating system does not configure to receive and generate a big process like an input of (*n)* could have length 150 digit or more (Window 32 bit or 64 bit). To solve this situation, and using the Table (1) results, we examine the equation to see how much it matched between the assuming and real result, and we got a closed similarity of results. As shown in Figure (2).

Although these numbers were not large as desired, where the max length of digit of *n* we have reached is 19, because the limitations of hardware and software of the computer are not suitable to manipulate more large numbers of (*n)*, although that, we were able to reach to a very encourage results by applying the proposed method as it shown in Table (1).

10^{-3} second

FIGURE.II TWO MIXED CURVE OF TRIAL RUN AND SIMULATED READING

Table II will show the reducing amount of the probable prime numbers elements before applying the proposed algorithm (approximately) and after.

TABLE II. REDUCING THE PRIME NUMBER

***n* value**	**No. of probable numbers - before**	**No. of probable numbers – after**
35	11	3
5141	685	8
191869	17318	30
10241339	679534	141
6498478561	≥ 292431535	3339
93767667359	≥ 2813030020	4227
547014990083	≥ 10393284811	24968
1524212467931	≥ 13717912211	8420
45586366129943	≥ 45586366129	191295
4694434085162951	≥ 2347217042581	1820515
572994973100356993	≥ 34379698386021	15838480
7724931571923045853	≥ 386246578596152	17006176

The results that we obtained proved that the proposed algorithm is reasonable and could be used for large numbers of (*n)*. We used a computer with limited specification such as the hard disk size (40 GB), CPU speed (1.46 GHz) and the RAM capacity is 512 MB. Also, there are many programs and

applications that could be considered affections on the operating systems speed. We have applied the proposed *Time* equation of the proposed algorithm to estimate the factorization time on a different large prime numbers (*n*), we got a very good result as shown in Table III.

TABLE III. SIMULATION OF ELAPSED TIME FOR BIG DIGIT LENGTHS OF *N*

Digit length of *n*	Elapsed time (millisecond)
25	92.25
30	118.5
50	224.5
60	276
70	330.5
85	444.25
95	466.75
101	493.25
115	571.75
128	652
135	686.75
140	713
145	739.25
150	775.5
156	799
165	856.25
170	870.5
175	896.75
180	953

VI. CONCLUSION

The aim of this research is to factorize (n) in a very fast way. We satisfied our objective by using the square root for (*n*) to get the half number of prime number values, and then by applying our rules we limited the search area by looking at the desired prime numbers only. TABLE.II proved our Idea in reducing the number of prime numbers we looking for. This, of course, will lead to a very fast method of factorizing.

ACKNOWLEDGMENT

I am delighted to thank Allah who prepare for me supervisor like Prof. Alaa AL-Hamami for his assistance and the use of his library and for his helping to write and design my thesis, and for helping me with a good ideas that make my thesis's more descanted and benefited. Also, first and foremost, I import my gratefulness to my family especially to my parents who were prepared for me the best environment and take me full incorporeal and materialistic support. I wish for them the happy and healthy life.

REFERENCES

[1] AL-Hamami AL-Ani, Technology of information security and protection systems, ISBN 978-9957-11-697-2, pp.173 - 223, Dar Wael , Jordan. 2007

[2] Aboud, AL-Fayoumi; "Efficient Method for Breaking RSA Scheme"; *Ubiquitous Computing and Communication Journal, vol. 4,no.2 , p:15-20,* 2008.

[3] Coppersmith, D. "Attack on the Cryptographic Scheme", Advances in Cryptology–CRYPTO '94, Springer-Verlag, LNCS 839, pp.294-307, 1994.

[4] Hastad, 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.

[5] Nguyen, H. Number Theory and the RSA Public Key Cryptosystem. http://cdn.bitbucket.org/mvngu/numtheory-crypto/downloads/numtheory-crypto.pdf. Accessed on 25/9/2009.

AUTHORS PROFILE

1. **Alaa Hamami** is a professor in database Security. He holds

a BS in physics from Baghdad University,1970, MSc in computer science from Loughborough University – England,1979 and a PhD degree in computer science – database security- from the University of East Anglia – England,1984. He has a membership in many different scientific societies including ACM and IEEE. He is a deanship of Computer Sciences and Informatics College -Amman Arab University, Jordan. Prof. Hamami has more than 31 years of experience including extensive project management experience in planning and leading a range of IT-related projects in addition to management posts. Prof. Hamami supervises more than 50 PhD and MSc students in computer science, information management and integration, security, and knowledge management.

2. Bilal AL-Kubaysee , Iraqi's born in Kuwait 1984. He holds the BS in computer science from Ahliyya Amman University,2008-Jordan, MSc in computer science from Amman Arab university, 2011 – Jordan. He interest in digital image process and database security and tag them together. he started writing in 2007, the writing genera for Geometric Image Processes and security matters, the most important his projects were Hand Geometric Authentication, PlateCar numbers Detection, and Fasr Approach for breaking the RSA Cryptosystem.