DOI: http://dx.doi.org/10.21123/bsj.2021.18.2.0338

A New Methodology to Find Private Key of RSA Based on Euler Totient Function

Kritsanapong Somsuk

Department of Computer and Communication Engineering, Faculty of Technology, Udon Thani Rajabhat University, UDRU, Udon Thani, Thailand.
E-mail: kritsanapong@udru.ac.th
ORCID ID: https://orcid.org/0000-0002-1311-8222

Received 11/11/2019, Accepted 26/10/2020, Published Online First 11/1/2021, Published 1/6/2021

Abstract:

The aim of this paper is to present a new methodology to find the private key of RSA. A new initial value which is generated from a new equation is selected to speed up the process. In fact, after this value is found, brute force attack is chosen to discover the private key. In addition, for a proposed equation, the multiplier of Euler totient function to find both of the public key and the private key is assigned as 1. Then, it implies that an equation that estimates a new initial value is suitable for the small multiplier. The experimental results show that if all prime factors of the modulus are assigned larger than 3 and the multiplier is 1, the distance between an initial value and the private key is decreased about 66%. On the other hand, the distance is decreased less than 1% when the multiplier is larger than 66. Therefore, to avoid attacking by using the proposed method, the multiplier which is larger than 66 should be chosen. Furthermore, it is shown that if the public key equals 3, the multiplier always equals 2.

Key words: Euler totient function, Private key, Public key, RSA

Introduction:

Nowadays, communication which is sent through opening a network such as internet and the machine is very popular because data is rapidly transmitted. However, opening a network is known as unsecure channel. With this problem, security and confidentiality of information becomes exceedingly important. Cryptography (1), which is one of security methods, is a technique to protect information by converting original message or plaintext as the unreadable message, or ciphertext. It is called the encryption process. In fact, ciphertext will be transmitted via the channel instead of plaintext. That means intruders cannot understand data which is trapped on the network. After ciphertext is arrived to receivers, they can use the decryption process to recover original plaintext.

The first generation of cryptography is called symmetric key cryptography (2). The secret key is selected for both of senders and receivers. Advanced Encryption Standard (AES) (1, 2) is the highest performance in this group. However, the problem is about how to exchange the secret key over unsecure channel. Later, this problem was solved when W. Diffie and M. E. Hellman proposed the new technique which is called asymmetric key cryptography or public key cryptography (3) in 1976. A pair of keys, the public key and the private key, is selected for the processes. Nevertheless, this algorithm can be chosen for only exchanging the secret key. In 1978, RSA (4) which is the best well known of public key cryptography was presented by R.L. Rivest, A. Shamir and L. Adleman. In fact, RSA can be chosen to solve many problems such as data encryption, digital signature and key exchange. In addition, if the modulus is a big number, then it becomes very difficult to recover both prime numbers by using mathematical techniques such as factoring. However, the difficulty to break this system is also based on the type of computer and updating method. Therefore, RSA is still very hard to be attacked.

Assuming all parameters of RSA are strong and at least 4096 bits of modulus is chosen (5), no one can break RSA in polynomial time. One the other hand, if one of parameters becomes a weakness, RSA may be broken by using some of disclosed algorithms. The examples of weak parameters which are already solved consist of prime numbers (6), small private key (7, 8), small public key with some disclosed parameters (9) and common modulus attack (10) etc.

In this paper, it is shown that if the multiplier, *k*, of Euler totient function is very small, time to break RSA can be decreased by using the proposed method. In fact, the proposed method which is suitable for small value of *k* is about estimating a new initial value, *f*, for the private key before finding the private key by using brute force attack. In addition, the distance is decreased about 66% when *k* equals 1. Furthermore, it is shown that when the public key equals 3 and prime factors are larger than 3, *k* always equals 2.

RSA Cryptosystem

RSA is the best well known public key cryptography. It can be applied with many tasks such as data encryption, digital signature and key exchange. For data encryption, it is divided into 3 processes as follows:

1) Key Generation: First, two secret prime numbers (p and q), $p > q$, are randomly generated to compute modulus, $n = p*q$ and Euler totient function, $\Phi(n) = (p-1)*(q-1)$. Next, the public key, e, is randomly chosen with the following condition, $1 < e < \Phi(n)$ and $\gcd(e, \Phi(n)) = 1$. The last process is to find the private key, d, from $ed \equiv 1 \bmod \Phi(n)$ or $ed = 1 + k\Phi(n)$, by using Extended Euclidean algorithm (11, 12). The published parameters are $\{e, n\}$ and the secreted parameters are $\{d, \Phi(n), p, q\}$.

2) Encryption process: Assuming m is represented as plaintext, the encryption equation is:

$$c = m^e \bmod n \qquad (1)$$

where c is ciphertext which will be sent to receiver.

3) Decryption process: After c is arrived, m is recovered by using the decryption equation:

$$m = c^d \bmod n \qquad (2)$$

However, for digital signature (13, 14), the process is different from data encryption. Assuming z is represented as a hash value of the signature and h is the signed text. Therefore, h is computed from the following equation:

$$h = z^d \bmod n \qquad (3)$$

In addition, the equation to verify the signed text is shown in the equation (4):

$$z = h^e \bmod n \qquad (4)$$

Exploit parameters to break RSA

In this section, the techniques to break RSA are presented. In fact, RSA is simply attacked when some parameters are weak.

Factoring

If n is factored, p and q are disclosed. After that, d can be easily recovered. In fact, it is very difficult to find p and q from at least 4096 bits of n when both of them are assigned as a strong parameter. On the other hand, RSA becomes an unsecured algorithm in the case that prime factors are a weak parameter because they are found by using some of factoring algorithms. The examples of factorization algorithms are as follows:

1) Trial division algorithm (TDA) (15, 16) is the simplest algorithm. It chooses 3 as the first divisor and it will be increased by 2 when the result has a remainder. Therefore, TDA is suitable for small value of q. In addition, there is a different technique (17) to implement TDA by changing the sequence of divisor. The value of $\lfloor \sqrt{n} \rfloor$ is selected as the first divisor and it will be decreased by 2 when the result has a remainder. In fact, $\lfloor \sqrt{n} \rfloor$ is very close to p and q when $p - q$ is small.

2) Fermat's Factorization algorithm (FFA) (18, 19, 20, 21) was discovered by P. der Fermat in 1600. He found that n can be rewritten as the difference between two perfect square numbers which are mathematically relative with p and q. In fact, FFA can find p and q very fast when the result of $p - q$ is small.

3) Pollard's p - 1 (22, 23) was proposed by J. Pollard in 1974. Fermat's little theorem (24) is the main theorem of this algorithm. In fact, Pollard's p-1 has a very high performance when all prime factors of $p - 1$ or $q - 1$ are small.

4) Generalized Trial Division (25) was presented by M. Sahin. The technique behind this algorithm is to find the result of $\gcd(x, n)$, where $x \in$ □$^+$, which does not equal 1, because it is one of two prime factors of n. In fact, ip and jq, where $i, j \in$ □$^+$, $1 < i < q$ and $1 < j < p$, are all integers that $\gcd(ip, n) = p$ and $\gcd(jq, n) = q$.

Wiener's attack

In 1990, M. Wiener (7, 8) showed that d will be recovered very simple by using continued fraction to find $\frac{k}{d}$ which is a convergence of $\frac{e}{n}$, when it is a small value, $d < \frac{1}{3}n^{\frac{1}{4}}$. Moreover, in 1999, D. Boneh and G. Durfee (26) showed that if the following condition occurred, $\frac{1}{3}n^{\frac{1}{4}} < d < n^{0.292}$, d is simply recovered.

Hastad Broadcasting Attack

Hastad Broadcasting Attack (9) is the technique to find d when $e = 3$. The condition for this method to finish the process is that the same message must be selected to be encrypted with $e = 3$ and the different values of modulus. For example, $c_1 = m^e \bmod n_1$, $c_2 = m^e \bmod n_2$ and $c_3 = m^e \bmod n_3$. Then, m can be recovered by using Chinese Remainder Theorem (CRT) (27, 28).

Common Modulus Attack

Common Modulus Attack (10) is the idea to find m when it is encrypted two times with different public keys and common modulus, from $c_1 = m^{e1} \bmod n$ and $c_2 = m^{e2} \bmod n$, where $\gcd(e_1, e_2) = 1$.

Partial Key Exposure attack

Assuming x is represented as bit length of n and the $\frac{x}{4}$ least significant bits of d is disclosed. The technique which is called Partial Key Exposure attack (10) can be chosen to recover d.

Brute force Attack

In fact, d should be assigned in the following condition, $1 < d < \Phi(n)$, and it is always an odd number. Therefore, the concept of brute force attack is to find d by choosing $d = 3$ as the first value to compute $t = ed \bmod \Phi(n)$. If $t = 1$, then d is the private key. On the other hand, d must be increased by 2 when the result is not equal to 1 until the correct answer is found.

The proposed method to attack RSA

In this paper, a new method to recover d in order to attack RSA is proposed. The key is to find an integer (f), where $3 < f < d$, to be a new initial value instead of 3 for brute force attack. In general, if brute force attack is selected to find d, then the initial value is begun as 3. On the other hand, if f is chosen as an initial value, then the distance between the f to d decreased when it is compared with the distance between 3 and d. In fact, f can be estimated by finding the smallest integer which is possible to be $\Phi(n)$.

Lemma 1 The highest value of $p + q$ is $3 + \frac{n}{3}$

Proof: Because $p > q$, then assigning $p = \sqrt{n} + a$ and $q = \sqrt{n} - b$, where $a, b \in \mathbb{Z}^+$

So, $p*q = (\sqrt{n} + a)*(\sqrt{n} - b) = n + a\sqrt{n} - b\sqrt{n} - ab$

Because $n = p*q$, then $a\sqrt{n} - b\sqrt{n} - ab = 0$. That means it implies that a is always larger than b. Therefore, assuming n is very close to $p'*q'$, where $p' = p + x$, $q' = q - y$ and $x, y \in \mathbb{Z}^+$, the result of $p' + q'$ must be larger than $p + q$. The reason is that x is always larger than y.

Because 3 is the smallest prime number which is an odd integer and the result of $3*\frac{n}{3}$ equals n, the highest value of $p + q$ is $3 + \frac{n}{3}$.

□

Theorem 1 Always d equals or larger than $\left\lceil \frac{2n-3}{3e} \right\rceil$

Proof:

From $\Phi(n) = (p - 1)*(q - 1) = pq - (p + q) + 1 = n - (p + q) + 1$

From Lemma 1, $\Phi(n) > n - (3 + \frac{n}{3}) + 1$

From $ed = 1 + k\,\Phi(n) = 1 + k(n - (p + q) + 1)$

Then, $ed > 1 + k(n - (3 + \frac{n}{3}) + 1)$

In fact, $k \in \mathbb{Z}^+$, that means the minimum of k is 1.

Assuming $k = 1$, then $ed > 1 + n - (3 + \frac{n}{3}) + 1$

$3ed > 3 + 3n - 9 - n + 3$

Or $d > \frac{2n-3}{3e}$

d is always an integer, then $d \geq \left\lceil \frac{2n-3}{3e} \right\rceil$

□

From Theorem 1, f can be assigned as $\left\lceil \frac{2n-3}{3e} \right\rceil$. Moreover, the convergence of d equals this value whenever k is small.

In addition, total loops which should be left out of the computation can be calculated by using the following equation:

$$s = \frac{f-3}{2} \qquad (5)$$

Baghdad Science Journal
P-ISSN: 2078-8665
E-ISSN: 2411-7986

where, s is the reduced loops and 3 is the smallest prime number which is an odd number.

Example 1: Finding f when n = 174279334060020221413 and e = 212441, (p = 41964266303, q = 4153041371, d = 53323777947306261 ($d \approx 5.33 \text{ x } 10^{16}$) and k = 65)

Sol. From $f = \left\lceil \dfrac{2n-3}{3e} \right\rceil$

$$= \left\lceil \frac{2\text{x}174279334060020221413-3}{3\text{x}212441} \right\rceil$$

$= 546910543194017 \approx 5.47 \text{ x } 10^{14}$

From, $s = \dfrac{f-3}{2} = \dfrac{546910543194017-3}{2}$

$= 273455271597007$

$s \approx 2.73 \text{ x } 10^{14}$

The information in Fig. 1 is shown that the distance between d and f is smaller than the distance between d and 3.

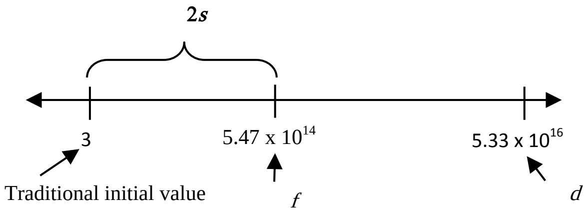

Figure 1. Some of parameters in example 1 on Numbers Line

Example 2: Finding f when n = 174279334060020221413 and e = 212439, (d = 135050712179508995819 ($d \approx 1.35 \text{ x } 10^{20}$) and k = 164621)(Fig.2)

Sol. From $f = \left\lceil \dfrac{2n-3}{3e} \right\rceil$

$$= \left\lceil \frac{2\text{x}174279334060020221413-3}{3\text{x}212439} \right\rceil$$

$= 546915692065394$

d should be an odd number, $f = f + 1$ = 546915692065395

Then, $f \approx 5.47 \text{ x } 10^{14}$

Then, it implies that,

$s = \dfrac{546915692065395-3}{2} = 273457846032696$

$s \approx 2.73 \text{ x } 10^{14}$

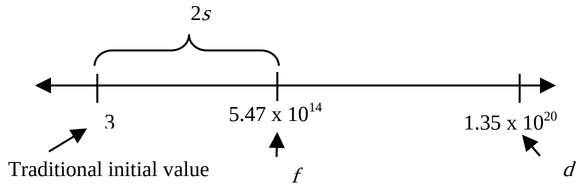

Figure 2. Some of parameters in example 2 on Numbers Line

From both Examples 1 and 2, it implies that the difference between d and f in example 2 is very high when it is compared with the result in example 1. The reason is that k in example 2 is larger than the same parameter in example 1. In fact, the proposed technique is suitable for small value of k, because k = 1 is assigned in Theorem 1. Moreover, it is highly efficient with small value of q, because q = 3 is assigned in the same theorem.

In general, k should be assigned as 1, because it is unknown value. However, in the next section, it is shown that if e = 3 and q > 3, then k always equals 2. Therefore, f can be expanded when the parameters are fallen in this case.

Lemma 2 Assigning $k \in \mathbb{Z}^{+}$, if

1. e = 3 and gcd(e, $\Phi(n)$) = 1, then only k = 1 or 2 can be chosen to generate different values of d.
2. e = 3, q = 3 and gcd(e, $\Phi(n)$) = 1, then d can be generated from only k = 1.
3. e = 3, q > 3 and gcd(e, $\Phi(n)$) = 1, then d can be generated from only k = 2.

Proof 1: From $ed = 1 + k\Phi(n)$

Assuming k = 3, $3d = 1 + 3\Phi(n)$

$d = \dfrac{1+3\Phi(n)}{3} = \dfrac{1}{3} + \Phi(n)$

That means, d is always larger than $\Phi(n)$ when k is higher than 2. However, it is impossible to occur. Therefore, assigning e = 3, the result which is fallen in range during 3 to $\Phi(n)$ must be generated from k = 1 or 2. In addition, it is shown in Table 1 that k = 2, k = 5 and k = 8 has the same value of d when e = 3 and n = 174279334060020221413.

Assuming k = 2, $3d = 1 + 2\Phi(n)$

$$d = \frac{1+2\Phi(n)}{3}$$

Therefore, it is possible to be occurred.

Assuming k = 1, $3d = 1 + \Phi(n)$

$$d = \frac{1+\Phi(n)}{3}$$

Therefore, it is possible to occur.

Therefore, if e = 3 and gcd(e, $\Phi(n)$) = 1, then only k = 1 or 2 can be chosen to generate different value of d. □

Proof 2:

From, $ed = 1 + k\Phi(n)$

Assuming k = 2, $3d = 1 + 2(3 - 1)*(\frac{n}{3} - 1)$

Because q = 3, then $n = 3p$, $= 1 + 4(p - 1)$

$$d = \frac{4p-3}{3},\ 4p - 3 > 3p$$

It implies that d is larger than p. Because $n = 3p$, $d*e = 3d > 9p > n$. Therefore, this case is impossible to occur. Therefore, k cannot be assigned as 2.

Next, assuming k = 1, $3d = 1 + (3 - 1)*(\frac{n}{3} - 1)$

Because q = 3, then $n = 3p$, $= 1 + 2(p - 1)$

$$d = \frac{2p-1}{3},\ d < p < \Phi(n)$$

Then, it is possible to occur.

Therefore, if e = 3, q = 3 and gcd(e, $\Phi(n)$) = 1, then d is always generated from k = 1. □

Proof 3:

From, $ed = 1 + k\Phi(n)$

Assuming k = 1, $3d = 1 + \Phi(n)$

If there is a solution, then$(1 + \Phi(n)) \bmod 3 = 0$

Because all odd prime numbers except 3 have the remainder as 1 or 2 when all of them are divided by 3, then there are 4 cases to consider d.

Case 1: $p \bmod 3 = 1$ and $q \bmod 3 = 1$

Then, $(p - 1) \bmod 3 = 0$ and $(q - 1) \bmod 3 = 0$

Implies, $\Phi(n) \bmod 3 = 0$

Therefore, $(1 + \Phi(n)) \bmod 3 = 1$, the contradiction occurred, then there is no solution.

Case 2: $p \bmod 3 = 1$ and $q \bmod 3 = 2$

Then, $(p - 1) \bmod 3 = 0$ and $(q - 1) \bmod 3 = 1$

Implies, $\Phi(n) \bmod 3 = 0$

Therefore,$(1 + \Phi(n)) \bmod 3 = 1$, the contradiction occurred, then there is no solution.

Case 3: $p \bmod 3 = 2$ and $q \bmod 3 = 1$

Then, $(p - 1) \bmod 3 = 1$ and $(q - 1) \bmod 3 = 0$

Implies, $\Phi(n) \bmod 3 = 0$

Therefore,$(1 + \Phi(n)) \bmod 3 = 1$, the contradiction occurred, then there is no solution.

Case 4: $p \bmod 3 = 2$ and $q \bmod 3 = 2$

Then, $(p - 1) \bmod 3 = 1$ and $(q - 1) \bmod 3 = 1$

Implies, $\Phi(n) \bmod 3 = 1$

Therefore, $(1 + \Phi(n)) \bmod 3 = 2$, the contradiction occurred, then there is no solution.

From all cases above, there is no result of $(1 + \Phi(n)) \bmod 3 = 0$. Therefore, it is impossible to find d from e = 3 and k = 1.

In fact, there is a solution in case 4 when e = 3 and k = 2 as follows:

From, $\Phi(n) \bmod 3 = 1$

Then, $k\Phi(n) \bmod 3 = 2\Phi(n) \bmod 3 = 2$

Therefore,$(1 + 2\Phi(n)) \bmod 3 = (1 + 2) \bmod 3 = 0$

Therefore, if e = 3, q > 3 and gcd(e, $\Phi(n)$) = 1, then d is always generated from k = 2. □

Theorem 2 Assigning e = 3 and q > 3, then $d \geq \left\lceil \frac{4n-9}{9} \right\rceil$

Proof:

From Lemma 1, $\Phi(n) > n - (3 + \frac{n}{3}) + 1$

and $ed > 1 + k(n - (3 + \frac{n}{3}) + 1)$

From Lemma 2, $3d = 1 + 2\Phi(n)$

$$3d > 1 + 2(n - (3 + \frac{n}{3}) + 1)$$
$$9d > 3 + 6n - 18 - 2n + 6$$
$$9d > 4n - 9$$
$$d > \frac{4n-9}{9}$$

Therefore, if e = 3 and q >3, then $d \geq \left\lceil \frac{4n-9}{9} \right\rceil$ □

In fact, assuming e = 3, q > 3 and $f' = \left\lceil \frac{4n-9}{9} \right\rceil$, then it implies that $f' \approx 2f$.

Example 3: Finding f' when n = 174279334060020221413 and e = 3, (d = 116186222675935275827 ($d \approx 1.16 \times 10^{20}$) and k = 2)

Sol. Start, $f' = \left\lceil \frac{4n-9}{9} \right\rceil = \left\lceil \frac{4 \times 174279334060020221413 - 9}{9} \right\rceil$
$= 77457481804453431738$

d should be an odd number, $f' = f' + 1 = 77457481804453431739$

$f' \approx 7.75 \times 10^{19}$

Then, it implies that,

$$s = \frac{77457481804453431739 - 3}{2}$$

$$= 38,728,740,902,226,715,868$$

$s \approx 3.87 \text{ x } 10^{19}$

In addition,

$$f = \left\lceil \frac{2n-3}{3e} \right\rceil = \left\lceil \frac{2\text{x}17427933406002022141 3-3}{9} \right\rceil$$

$= 38,728,740,902,226,715,870$

d should be an odd number, $f = f + 1 = 38,728,740,902,226,715,871$

$$f \approx 3.87 \text{ x } 10^{19}$$

Therefore, it implies that $f' \approx 2f$.

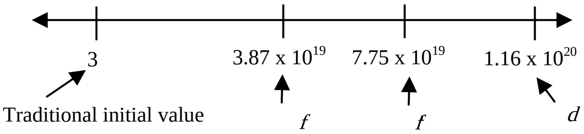

Figure 3. Some of parameters in example 3 on Number Line

From Fig. 3, f' which is about $2f$ is chosen to be a new initial value of d. In fact, f' can be estimated as $2f$ when $e = 3$ and $q > 3$ are selected as parameters of RSA algorithm.

In addition, if RSA is chosen for digital signature, both z and h must be disclosed. Moreover, h can be also recovered by using the following equation:

$$t = h*z^{-(d-1)} \bmod n \qquad (6)$$

In fact, z is always recovered by using equation (6), because

$t = h*z^{-(d-1)} \bmod n = z^d*z^{-(d-1)} \bmod n = z$

However, it is not necessary to compute $d - 1$ times of the multiplication. The process to find d is improved as follows:

First, the process to find u,

$$u = c * z^{-f} \bmod n \qquad (7)$$

And find t from,

$$t = u * (z^{-1})^{d-f-1} \bmod n \qquad (8)$$

Therefore, it requires only $d - f - 1$ times of modular multiplication and 1 times of modular exponentiation. In addition, loops of modular multiplication decreased as $\frac{d - f - 1}{2}$.

Assuming RSA is applied with digital signature, then both z and h can be found. On the other hand, only c will be known when RSA is chosen for data encryption. Because both of original plaintext and ciphertext must be chosen for the proposed algorithm, the proposed method can be chosen to find d when RSA is applied with digital signature.

Algorithm: Finding d
Input: z, h, e, n

```
f ← ⌈(2n-3)/(3e)⌉
IF f is an even number then
    f ← f + 1
End IF
z ← h^e mod n
u ← z^-1 mod n
t ← h * u^f mod n
i ← 0
While t ≠ z do
    t ← t * u mod n
    i ← i + 1
End While
d ← f + i - 1
```

Output: d

Experimental Results and Analysis

The aim of this section is to consider the performance of f which is generated from different values of k. The experiment is divided into 4 tables: Table 1 is the considering f from n = 174279334060020221413, which is the modulus from examples 1 to 3; Table 2 is to consider n = 10813049747177129789 (p = 85142123993520707, q = 127 and $\Phi(n)$ = 10727907623183608956). The difference between Table 1 and Table 2 is the aspect of $p - q$. The result of $p - q$ is small in Table 1 but it is high in Table 2;

Table 3 is to consider 1024 bits of n,

n = 47775655623228230693495843508585842635266289021821771194418882620388172201209154955484249943919022487592148795538508194918531874607677181664356175542109001519934694377115275083292209693792507992595437880750895759466882341935810398088919065228053860053663604443268752287945579270039001704658847760341877529027

(p = 67118165566638937993700076171567008082031652748859133971516327656546525619999363350799345711514830757422061256195763731620182519999242693617018374066371 79

q = 71181408520162393304242624527679308000083338769708327062720139108736082960607087468531877576454374082085372874929801690803930440001524540032369704945483 13

$\Phi(n)$ = 477756556232282306934958435085858426352662

890218217711944188826203881722012091549554
842499439190224875921487955385081949185318
746076771816643561755421089876899772856969
821452890221397691608997810962860240047923
358202058136749523374438371321057250631331
796536998556397314033368587430016279354828
21533976343536)

Table 1. The pair of (*e*, *d*) generated from *n* = 174279334060020221413 which small result of *p* – *q*

Row	*k*	Public Key (*e*)	Private Key (*d*)	The new initial value (*f*)	*d* - *f*	Decreased Distance (%)
1	1	11	15843575819445719431	10562383882425467964	5281191937020251467	66%
2	1	253	688851122584596497	459234081844585563	229617040740010934	66%
3	1	688851122584596497	253	168	85	66%
4	2	3	116186222675935275827	38728740902226715869	77457481773708559958	33%
5	2	148691	2344181342702691	781393781107667	1562787561595024	33%
6	2	74196809	4697758201809	1565919401017	3131838800792	33%
7	3	1039	503212706488651339	111825045915957793	391387660572693546	22%
8	3	20107	26002785201258703	5778396712919885	20224388488338818	22%
9	4	101	6902151842134768861	1150358640660199481	5751793201474569380	16%
10	4	10949	63669498224094589	10611583040157105	53057915183937484	16%
11	5	3	116186222675935275827	38728740902226715869	77457481773708559958	33%
12	5	19	45862982635237608879	6115064352983165663	39747918282254443216	14%
13	5	171	5095886959470845431	679451594775907295	4416435364694938136	14%
14	7	173	7051764960100117897	671596663044393917	6380168297055723981	10%
15	7	3827279	318752653803739	30357395608389	288395258195350	10%
16	8	3	116186222675935275827	38728740902226715869	77457481773708559958	33%
17	8	2297	606980701833357993	50581725166164627	556398976667193366	9%
18	8	6891	202326900611119331	16860575055388208	185466325555731123	9%
19	11	17	112768980832525414773	6834483688628243977	105934497143897170796	7%
20	11	714463	2683235764697307	162620349418627	2520615415278680	7%
21	15	271	9646457602245548731	428731449102140767	9217726153143407964	5%
22	15	11653	224336223308036017	9970498816328855	214365724491707162	5%
23	22	3613	1061208233685542237	32157825271707763	1029050408413834474	4%
24	22	213167	17986580231958343	545047885961149	17441532345997194	4%
25	45	14341	546863540243053561	8101682079818711	538761858163234850	2%
26	45	157751	49714867294823051	736516552710791	48978350742112260	2%
27	66	9883	1163860775565880027	11756169453271287	1152104606112608738	1%
28	67	42053	277666643971452577	2762852179551521	274903791791901056	< 1%
29	769682	1453971	92257456553458647011	79909587403517	92257376643871243494	< 1%
30	5268695	17731261	51785637565336171641	6552620409043	51785631012715762598	< 1%

Table 2. The pair of (*e*, *d*) generated from *n* = 10813049747177129789 which high result of *p* – *q*

Row	*k*	Public Key (*e*)	Private Key (*d*)	The new initial value (*f*)	*d* - *f*	Decreased Distance (%)
1	1	11	975264329380328087	655336348313765441	319927981066562646	66%
2	1	187	57368489963548711	38549196959633261	18819293003915450	66%
3	1	688851122584596497	187	125	62	66%
4	2	1961	10941262236801233	3676032550459673	7265229686341560	33%
5	2	307877	69689568387269	23414220066623	46275348320646	33%
6	2	2328283	9215295239611	3096144167805	6119151071806	33%
7	3	566107	56850953741167	12733811508163	44117142233004	22%
8	4	5	8582326098546887165	1441739966290283971	7140586132256603194	16%
9	4	25	1716465219709377433	288347993258056795	1428117226451320639	16%
10	5	131	409462123022275151	55028242988178777	354433880034096374	14%
11	7	8713	8618771188142461	827349917531439	7791421270611022	10%
12	7	136751	549139336182443	52714055703077	496425280479366	10%
13	7	374659	200436539259127	19240695756545	181195843502582	10%
14	7	5880293	12770682236801	1225908272165	11544773964636	10%
15	9	955	101100700113772231	7548376786860125	93552323326912106	8%
16	9	12557	7689031505029265	574078189969851	7114953315059414	8%
17	9	334805	288380306771561	21531039952961	266849266818600	8%
18	11	7943	14856727162913219	907553799754679	13949173363158540	7%
19	11	729769	161704572070093	9878057072103	151826514997990	7%
20	11	2624063	44971094007659	2747151966797	42223942040862	7%
21	15	32969	4880906741112989	218650848720051	4662255892392937	5%
22	15	626411	256889828479631	11507939406319	245381889073312	5%
23	22	6949	33963731142616117	1037372259526755	32926358883089362	4%
24	25	326171	822260993710631	22100983323015	800160010387616	3%
25	45	40073	12046910464483877	179889197999935	11867021266483942	2%
26	45	915673	527214238099477	7872570045695	519341668053782	2%
27	66	7001	101134395533512097	1029667166326441	100104728367185656	1%
28	67	13079	54956021924711507	551165978396775	54404855946314732	< 1%
29	123	1605341	821964079688729	4490447718865	817473631969864	< 1%
30	769682	60844009	135708963477853777	118478383491	135708844999470286	< 1%

Table 3. The pair of (*e*, *d*) generated from 1024 bits of *n*

Row	*k*	Public Key (*e*)	Private Key (*d*)	The new initial value (*f*)	*d - f*	Decreased Distance (%)
1	1	7	6.82×10^{306}	4.55×10^{306}	2.27×10^{306}	66%
2	1	2275679	2.09×10^{301}	1.39×10^{301}	6.99×10^{300}	66%
3	1	2.09×10^{301}	2275679	1517119	758560	66%
4	2	2623	3.64×10^{304}	1.21×10^{304}	2.42×10^{304}	33%
5	2	1757281	5.43×10^{301}	1.81×10^{301}	3.62×10^{301}	33%
6	2	2492887	3.83×10^{301}	1.27×10^{301}	2.55×10^{301}	33%
7	3	457	3.14×10^{305}	6.96×10^{304}	2.43×10^{305}	23%
8	4	709	2.69×10^{305}	4.49×10^{304}	2.69×10^{305}	16%
9	4	3545	5.39×10^{304}	8.98×10^{303}	4.49×10^{304}	16%
10	5	23	1.04×10^{307}	1.38×10^{306}	9.00×10^{306}	14%
11	8	203	1.88×10^{306}	1.56×10^{305}	1.72×10^{306}	9%
12	8	2291	1.67×10^{305}	1.39×10^{304}	1.52×10^{305}	9%
13	8	43529	8.78×10^{303}	7.31×10^{302}	8.04×10^{303}	9%
14	8	304703	1.25×10^{303}	1.04×10^{302}	1.14×10^{303}	9%
15	9	25	1.72×10^{307}	1.27×10^{306}	1.59×10^{307}	8%
16	9	6895519	6.24×10^{301}	4.61×10^{300}	5.77×10^{301}	8%
17	9	34477595	1.24×10^{301}	2.92×10^{299}	1.15×10^{301}	8%
18	13	103	6.03×10^{306}	3.09×10^{305}	5.72×10^{306}	6%
19	13	18121	3.43×10^{304}	1.75×10^{303}	3.25×10^{304}	6%
20	16	37	2.07×10^{307}	8.61×10^{305}	1.97×10^{307}	5%
21	16	1155547	6.62×10^{302}	2.75×10^{301}	6.34×10^{302}	5%
22	22	353	2.98×10^{306}	9.02×10^{304}	2.89×10^{306}	4%
23	22	2471	4.25×10^{305}	1.29×10^{304}	4.12×10^{305}	4%
24	25	22399	5.33×10^{304}	1.42×10^{303}	5.19×10^{304}	3%
25	47	393947	5.69×10^{303}	8.08×10^{301}	5.61×10^{303}	2%
26	47	10743361	2.09×10^{302}	2.96×10^{300}	2.06×10^{302}	2%
27	66	3269449	9.64×10^{302}	9.74×10^{300}	9.54×10^{302}	1%
28	67	4971403	6.44×10^{302}	6.41×10^{300}	6.37×10^{302}	< 1%
29	100	761	7.72×10^{306}	4.18×10^{304}	7.72×10^{306}	< 1%
30	769682	44059	2.24×10^{307}	7.22×10^{302}	2.24×10^{307}	< 1%

The results from Table 1, Table 2 and Table 3 imply that:

1) The ratio between the decreased distance, $d - f$, and d is quite stable for the same value of k when $d = \frac{1 + k\Phi(n)}{e} < \Phi(n)$.

2) In fact, an exception is occurred when $d > \Phi(n)$, because d must be decreased in the following condition, $1 < d < \Phi(n)$, and it also decreases value of k. Therefore, the ratio of this case is similar to the same pair of (e, d) which is generated from a smaller value of k. The example that the same pair of (e, d) can be generated from the different values of k, $k = 2$, $k = 5$ and $k = 8$, and the ratio of them are 33% is shown in 4th row, 11th row and 16th row of Table 1.

3) The size of e does not affect the ratio; it is shown in the 2nd Row and the 3rd Row that although e is alternated with d, the ratio is not changed. In fact, only size of k affects the ratio.

4) The maximum decreased distance is 66% for all the values of $q > 3$ and $k = 1$.

5) The ratio that is less than 1% is begun at $k > 66$. Therefore, to avoid attacking by using the proposed method $k > 66$.

6) From Theorem 2 and the results in Table 1, Table 2 and Table 3, it implies that d is rapidly recovered when $e = 3$, because k is stable ($k = 2$) and the ratio is always 33%. Therefore, we can estimate 99% of the ratio by using the equation $f = \left\lceil f * \frac{99}{34} \right\rceil \approx \lceil f * 2.91 \rceil$, 34 is from $\lfloor 33.xx \rfloor$% to prevent $f > d$. The example is as follows:

From 4th Row of Table 1, f = 38728740902226715869, then

$$f = \lceil 38728740902226715869 * 2.91 \rceil = 112700636025479743179 = 1.12 * 10^{20}$$

In Table 4, $q = 3$ (n = 24473764731015097203, p = 8157921577005032401 and $\Phi(n)$ = 16315843154010064800) is chosen for the experiment to consider the performance of f. The reason is that $q = 3$ is assigned with the main equation in Theorem 1.

Table 4. The pair of (e, d) generated from n = 244737647331015097203 with q = 3 (p = 81579215770050324 01)

Row	k	Public Key (e)	Private Key (d)	The new initial value (f)	$d - f$	Decreased Distance (%)
1	1	3779	4317502819267019	4317502819267019	0	100%
2	1	43019	379270628187779	379270628187779	0	100%
3	1	379270628187779	43019	43019	0	100%
4	2	761671	42842232811831	21421116405915	21421116405916	50%
5	3	64849	754792355503249	251597451834417	503194903668833	33%
6	3	453943	107827479357607	35942493119203	71884986238404	33%
7	3	1094957	44702695596293	14900898532097	29801797064196	33%
8	4	4049	16118392841699249	4029598210424813	12088794631274436	25%
9	4	28225579	2312206690819	578051672705	1734155018114	25%
10	5	221	36913672294140 4181	7382734458828 0837	29530937835312 3344	20%
11	5	182579	446815985245019	89363197049003	357452788196016	20%
12	7	47	2430019193150435183	347145599021490741	2082873594128944442	15%
13	7	23206801	4921441006801	703063000971	4218378005830	15%
14	8	907	143910413706814243	17988801713351781	125921611993462462	13%
15	8	7841	16646696241816161	2080837030227021	14565859211589140	13%
16	9	28211	5205153606256091	578350400695121	4626803205560970	12%
17	11	12572633	14274995117897	1297726828899	12977268288998	10%
18	15	101	2423145022872781901	161543001524852127	2261602021347929774	7%
19	15	437	560040382860757373	37336025524050491	522704357336706882	7%
20	15	44137	5544954285750073	369663619050005	5175290666700068	7%
21	16	1523	171407413305424187	10712963331589011	160694449973835176	7%
22	16	11420977	22857369423313	1428585588957	21428783834356	7%
23	22	108613	3304839654444877	150219984292949	3154619670151928	5%
24	22	6300011	56975860738691	2589811851759	54386048886932	5%
25	45	696359	1054359808562039	23430217968045	1030929590593994	3%
26	45	4874513	150622829794577	3347173995435	147275655799142	3%
27	66	263	4094470145112791927	62037426441102907	4032432718671689020	2%
28	100	329	5008815071595794969	49592228431641535	4959222843164153434	1%
29	102	5471	304188631275640031	2982241483094511	301206389792545520	< 1%
30	769682	769682	1289426730838273392 1	249932494202143	1289401737588853177 8	< 1%

The results in Table 4 imply that,
1) $f = d$ when $k = 1$, because $q = 3$ is chosen in the experiment.
2) For the same value of k, the ratio considered from $q = 3$ is always larger than the others which are generated from $q > 3$.

In addition, the proposed method will be compared with some other algorithms. In general, the prominent point of each method is different from each other. However, $k = 1$ and a small private key are assigned for all values in Fig. 1 to ensure that the proposed method is efficient when k is small. There are 4 compared methods as follows:
1) Brute force attack that the initial value is 3, this method is efficient when d is small.
2) The improvement of FFA in (21), this method is efficient when the result of $p - q$ is close to 0.
3) The improvement of TDA in (17), this method is efficient when q is close to $\sqrt{n}$
4) Pollard's $p - 1$, this method is efficient when all prime factors of $p - 1$ or $q - 1$ are small.

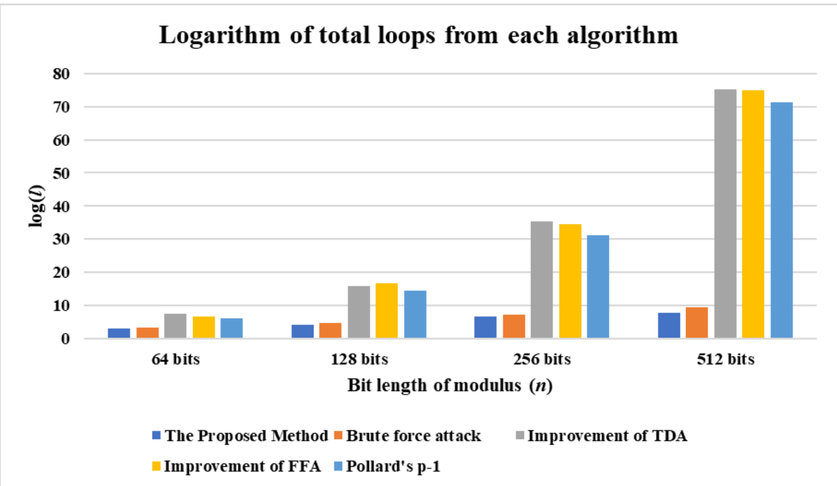

Figure 4. Logarithm of total loops from each algorithm

Assuming l is represented as total loops to finish the process of each algorithm, Fig. 4 shows the result of $\log_{10}(l)$ from the proposed method and all compared methods. The experimental results show that the proposed method requires the smallest loops when k equals 1 and d is small. On the other hand, it cannot guarantee that the proposed method is the most efficient algorithm whenever k and d are not assigned in the conditions suitable for this method.

Therefore, the conclusion is that the proposed method is a special proposed method that is suitable for the small values of k and d.

Conclusion:

In this paper, the new technique to recover the private key (d) is proposed by estimating the new initial value, f, before using brute force attack. In fact, f can be computed by choosing the smallest value which may be Euler totient function, $\Phi(n)$, instead of the real value of $\Phi(n)$ and selecting 1 instead of k. The method is suitable for the small value of k, especially k equals 1. In fact, assuming a prime factor (q) is higher than 3 and k equals 1, the distance between f and d decreased about 66%. On the other hand, the decreased distance is less than 1% when k is larger than 66. Therefore, to avoid attacking RSA by using the proposed technique, k should be assigned very large. Furthermore, d which is computed from $\frac{1+k\Phi(n)}{e}$ must be less than $\Phi(n)$. In addition, it is shown that k always equals 2 when e equals 3. However, if e and q equal 3, then k always equals 1.

Author's declaration:

- Conflicts of Interest: None.
- I hereby confirm that all the Figures and Tables in the manuscript are mine. Besides, the Figures and images, which are not mine, have been given the permission for re-publication attached with the manuscript.
- Ethical Clearance: The project was approved by the local ethical committee in Udon Thani Rajabhat University.

References:

1. Alka R, Gupta A, Jaiswal M. An enhanced AES algorithm using cascading method on 400 bits key size used in enhancing the safety of next generation internet of things (IOT), Proceedings of the International Conference on Computing, Communication and Automation, India, 2017, pp. 422 - 427.
2. Yu L, Zhang, D, Wu L, Xie S, Su D, Wang X. AES Design Improvements Towards Information Security Considering Scan Attack, Proceedings of the IEEE International Conference On Trust, Security and Privacy In Computing And Communications/ IEEE International Conference On Big Data Science And Engineering, Communication and Automation, USA, 2018, pp. 322 - 326.
3. Diffie W, Hellman M. New directions in cryptography. IEEE Transactions on Information Theory. 1976; 22(6): 644-654.
4. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public key cryptosystems. Communications of ACM. 1978; 21: 120 – 126.
5. Priyadarshini P, Prashant N, Narayan DG, Meena SM. A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish. Procedia Computer Science. 2015; 78: 617 – 624.
6. Hosung J, Heejin P. Fast Prime Generation Algorithms using proposed GCD test on Mobile Smart Devices, Proceedings of the International Conference on Big Data and Smart Computing, China. 2016; pp. 374-377.
7. Wiener, M. Cryptanalysis of short RSA secret exponents. Proc. IEEE. 1990; 36: 553-558.
8. Thuc D N, Than DN, Long DT. Attacks on Low Private Exponent RSA: An Experimental Study, Proceedings of the International Conference on Computational Science and Its Applications, Vietnam. 2013; pp. 162-165.
9. Hastad J. On using RSA with Low Exponent in a Public-Key Network. Advances in Cryptology. 1986; 218: 404-408.
10. Imad KS, Abdullah D, Saleh O. Mathematical Attacks on RSA Cryptosystem. Journal of Computer Sciences. 2006; 2(8): 665 – 671.
11. Dongmuanthang P, Prabir S. Redesigned the Architecture of Extended-Euclidean Algorithm for Modular Multiplicative Inverse and Jacobi Symbol, Proceedings of the International Conference on Trends in Electronics and Informatics, India. 2018; pp. 1345 - 1349.
12. Ibrahim H, Fayez G, Atef I. High Speed and Low Area Complexity Extended Euclidean Inversion Over Binary Fields. IEEE Trans. Consum. Electron. 2019; 65: 408 – 417.
13. Farah J, Endroyono, Achmad A. Security System Analysis in Combination Method: RSA Encryption and Digital Signature Algorithm, Proceedings of the International Conference on Science and Technology, Indonesia. 2018; pp. 1- 5.
14. Muhammad R P, Deden IA, Riri FS. Comparison of ECDSA and RSA signature scheme on NLSR performance, Proceedings of EEE Asia Pacific Conference on Wireless and Mobile, Indonesia. 2018; pp. 7- 11.
15. Nidhi L, Anurag P, Shishupal K. Modified Trial Division Algorithm Using KNJ-Factorization Method To Factorize RSA Public Key Encryption, Proceedings of the International Conference on Contemporary Computing and Informatics, India. 2014; pp. 992 – 995.
16. Ambedkar BR, Gupta A, Gautam P, Bedi SS. An Efficient Method to Factorize the RSA Public Key

Encryption, Proceedings of the International Conference on Communication Systems and Network Technologies. Katra. 2011; pp. 108 -111.
17. Somsuk K, Chiawchanwattana T, Sanemueang C. Estimating the new Initial Value of Trial Division Algorithm for Balanced Modulus to Decrease Computation Loops, Proceedings of the International Joint Conference on Computer Science and Software Engineering. Thailand. 2019; pp. 137 – 141.
18. Wu ME, Tso R, Sun HM. On the improvement of Fermat factorization using a continued fraction technique. **Future** Gener Comput Syst. 2014; 30(1): 162 – 168.
19. Omar K, Szalay L. Sufficient conditions for factoring a class of large integers. J. Discret. Math. Sci. Cryptogr. 2010; 13: 95-103.
20. Somsuk K, Tientanopajai K. An Improvement of Fermat's Factorization by Considering the Last m Digits of Modulus to Decrease Computation Time. Int. J. Netw. Secur. 2017; 19: 99 – 111.
21. Somsuk K. The improvement of initial value closer to the target for Fermat’s factorization algorithm. J. Discret. Math. Sci. Cryptogr. 2013; 7-8: 1573 – 1580.
22. Bishop D. Introduction to Cryptography with java Applets, Jones and Bartlett Publisher, London, England, 2003.
23. Sarnaik S, Bhakkad R, Desai C. Comparative study on Integer Factorization Algorithm-Pollard's RHO and Pollard's P-1, Proceedings of the International Conference on Computing for Sustainable Global Development. India, 2015; pp.677 – 679.
24. Nikita YM, Ivan YS, Vasiliy IY, Olga AS, Irina VR, Angrey GL, et al. Modification and Optimization of Solovey–Strassen’s Fast Exponentiation Probablistic Test Binary Algorithm. Proceedings of the IEEE East-West Design & Test Symposium, Georgia. 2019, pp.1-3.
25. Murat S. Generalized Trial Division. IJCMS.2011; 6(2): 59 – 64.
26. Boneh D, Durfee G. Cryptanalysis of RSA with Private Key d less than $N^{0.292}$. Lect. Notes Comput. Sci. 1999; 1592: 1 – 11.
27. Kong F, Zhou D, Jiang Y, Shang J, Yu J. Fault Attack on an Improved CRT-RSA algorithm with the Modulus Chaining Method, Proceedings of IEEE International Conference on Computational Science and Engineering and IEEE International Conference on Embedded and Ubiquitous Computing, China. 2017; pp. 866 – 869.
28. Somsuk K, Chiawchanwattana T, Sanemueang C. Speed up RSA’s Decryption Process with Large sub Exponents using Improved CRT, Proceedings of International Conference on Information Technology. Thailand, 2018, pp. 1-5. IEEE.

بالاعتماد على دالة مؤشر أويلر RSA منهجية جديدة للعثور على المفتاح الخاص ل

كرتسانتبونك سومسوك

قسم الحاسوب وهندسة الاتصالات، كلية التكنلوجيا، جامعة ادون تاني رجبات، ادون ثاني تايلاند.

الخلاصة:

الهدف من هذه البحث هو تقديم منهجية جديدة للعثور على المفتاح الخاص لــ RSA .القيمة الاولية الجديدة يتم إنشاؤها من معادلة جديدة لتسريع العملية. في الواقع ، بعد العثور على هذه القيمة ، يتم اختيار هجوم القوة القاسية لاكتشاف المفتاح الخاص. بالإضافة إلى ذلك ، بالنسبة إلى المعادلة المقترحة ، تم تعيين مضاعف دالة مؤشر أويلر لايجاد كلا من المفتاح العام والمفتاح الخاص على أنه 1. ومن ثم ، حصلنا على أن المعادلة التي تقدر قيمة أولية جديدة مناسبة للمضاعف الصغير. النتائج التجريبية تبين أنه إذا تم تعيين جميع العوامل الأولية للمعامل أكبر من 3 وكان المضاعف 1 ، فإن المسافة بين القيمة الأولية والمفتاح الخاص تنخفض بنحو 66٪. من ناحية أخرى ، تقل المسافة عن 1٪ عندما يكون المضاعف أكبر من 66. لذلك ، لتجنب الهجوم باستخدام الطريقة المقترحة ، يجب اختيار المضاعف الأكبر من 66. علاوة على ذلك ، يتضح أنه إذا كان المفتاح العمومي يساوي 3 ، فإن المضاعف دائمًا يساوي 2.

الكلمات المفتاحية: دالة مؤشر أويلر ، مفتاح خاص ، مفتاح عام ، RSA.