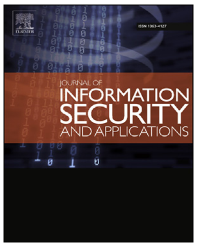
Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/jisa



An Enhanced and Secured RSA Key Generation Scheme (ESRKGS)

M. Thangavel*, P. Varalakshmi, Mukund Murrali, K. Nithya

Department of Information Technology, MIT Campus, Anna University, Chromepet, Chennai, 600044, Tamilnadu, India

Keywords:
Cryptosystem
RSA algorithm
Public key
Private key
Encryption
Decryption

A B S T R A C T

Public-key cryptography can be claimed as the greatest and an excellent revolution in the field of cryptography. A public-key cryptosystem is used for both confidentiality and authentication. One such public-key cryptosystem is the RSA cryptosystem. In this paper, a modified and an enhanced scheme based on RSA public-key cryptosystem is developed. The proposed algorithm makes use of four large prime numbers which increases the complexity of the system as compared to traditional RSA algorithm which is based on only two large prime numbers. In the proposed Enhanced and Secured RSA Key Generation Scheme (ESRKGS), the public component n is the product of two large prime numbers but the values of Encryption (E) and Decryption (D) keys are based on the product of four large prime numbers (N) making the system highly secured. With the existing factorization techniques, it is possible only to find the primes p and q. The knowledge of n alone is not sufficient to find E and D as they are based on N. The time required for cryptanalysis of ESRKGS is higher than traditional RSA cryptosystem. Thus the system is highly secure and not easily breakable. A comparison is done between the traditional RSA scheme, a recent RSA modified scheme and our scheme to show that the proposed technique is efficient.

1. Introduction

Security is the concept of keeping information secret by protecting it from unauthorized users. In order to keep a data secured it must be hidden from unauthorized access (confidentiality), prevented from modifications (integrity) and available to authorized persons when needed (availability). Confidentiality, integrity and availability are thus, the three security goals (Forouzan, 2007,). Security goals can be implemented using several techniques among which cryptography is the most general and widely prevalent one. Cryptography is a term of Greek origin meaning *secret writing*. In earlier days,

* *Corresponding author.*
E-mail address: thangavelmuruganme@gmail.com (M. Thangavel).

http://dx.doi.org/10.1016/j.jisa.2014.10.004
2214-2126/© 2014 Elsevier Ltd. All rights reserved.

cryptography referred only to the process of encryption and decryption, but today cryptography involves more complex processes and procedures.

Any cryptographic system is usually characterized by three main dimensions, viz., the operations involved in the transformation of plain text to cipher text, the number of secret keys used and the method of processing the plain text (Stallings, 2011). Based on this, there are two broad classifications of cryptosystems. One is the symmetric-key cryptosystem (or shared-key cryptosystem) and the other is the asymmetric-key cryptosystem (or public-key cryptosystem). In symmetric key cryptosystem, encryption and decryption are done using the same key whereas in a public-key cryptosystem, encryption and decryption are done using two different keys. Among the two keys, one key is private and the other is public. Generally encryption is done using public key and decryption is done using private key but some schemes do the vice versa. In public-key cryptosystems, finding the private key from the public key is computationally infeasible. A public-key cryptosystem has six components – 1) Plain text; 2) Encryption algorithm; 3) Public Key; 4) Private Key; 5) Decryption algorithm; 6) Cipher text. The entire cryptosystem evolves around these key components. Public-key cryptosystems can be used for encryption/decryption, to create digital signatures and for key exchange. Among the public-key cryptosystems, the most popular one is the RSA cryptosystem developed by Rivest et al. (1978) at MIT. The RSA algorithm can be described as follows.

ALGORITHM 1.1. RSA ALGORITHM

```
RSA_Keygen()

INPUT:
Two large prime numbers $p$ and $q$.

OUTPUT:
Public Key Components: $\{e, n\}$
Private Key Components: $\{d, n\}$

PROCEDURE:
$n \leftarrow p * q$

/*Compute Euler phi value of $n$ */
$\Phi(n) \leftarrow (p-1) * (q-1)$

Find a random number $e$, satisfying $1 < e < \Phi(n)$ and $\gcd(e, \Phi(n)) = 1$.

Compute a random number $d$, such that,
$d \leftarrow e^{-1} mod(\Phi(n))$

RSA_Encrypt()

INPUT:
Plain text message, $M (< n)$

OUTPUT:
Cipher text, $C$

PROCEDURE:
•  $A$ transfers a message (or Plain text) $M (< n)$ along with the public keys to$B$.
•  $B$ encrypts the message using $A$'s public key $e$ to generate a Cipher text, $C$.

        Cipher Text, $C \leftarrow M^e mod\ n$

RSA_Decrypt()

INPUT:
Cipher text message, $C$

OUTPUT:
Decrypted plain text, $P$

PROCEDURE:
•  $B$ transfers the Cipher text $C$, to $A$.
•  $A$ decrypts the Cipher text using its private key $d$ to derive the Plain text $P$.

        Plain Text, $P \leftarrow C^d mod\ n$
```

To strengthen the security, RSA algorithms based on more than two prime numbers can be seen in the works of Al-Hamami and Aldariseh (2012) and Ivy et al. (2012) but the drawback with both approaches is that the original message can be easily obtained by the knowledge of the public key transmitted as no complexity is involved in encryption and decryption. Yet another variant of RSA algorithm was proposed by Sharma et al. (2011) in which a modified subset-sum over RSA was used for encryption. The subset sum problem is a good introduction to the NP-complete class of problems. This approach was significantly complex in terms of its operations. So the security level was higher than the previous approach. However, the method cannot work efficiently against brute-force attacks as it was based on factoring. The work by Segar and Vijayaragavan (2013)introduced a new technology named as "modify trial division technique" to implement RSA algorithm for large numbers. The work by Minni et al. (2013) introduced an enhanced RSA algorithm in which the distribution of n was eliminated from the key so that one cannot trace back to the factors p and q. A systolic RSA cryptosystem based on modified Montgomery's algorithm and the Chinese Remainder Theorem (CRT) technique was developed by Wu et al. (2001) based on hardware implementations. Another variant of RSA called the BEAIRSA (Batch Encrypt Assistant Improved RSA) was developed by Liu et al. (2011) to improve the Batch RSA decryption performance by transferring some decryption computations to encryption in modular exponentiation. In this paper the RSA system was divided into four phases: Setup, Percolate-Up, Exponentiation-Phase and Percolate-Down. This scheme speeded up the decryption process. Rama Chandra Rao et al. (2012) developed a novel modular multiplication algorithm and its application to RSA Decryption. The basic idea behind the work was that RSA can be speeded up by using the Chinese Remainder Theorem (CRT) and using strong prime criterion and thus an efficient modulo n multiplication algorithm was introduced. A new algorithm (SA_RSA) which presented design of Jordon-Totient function and applied them to RSA public-key cryptosystem with one public key and one private key used for development of protocols to provide secured communication was introduced by Mahajan and Easo (2012). The scheme tells that increase of key size increases the security of the system.

In this paper, an Enhanced RSA Key Generation Scheme (ESRKGS) is proposed to reduce the direct attacks possible in the case of RSA. The scheme is based on four large prime numbers instead of two. Also, the keys are not directly dependent on the public key n. Therefore any kind of brute-force attack is difficult on the proposed method.

The rest of the paper is organized as follows. Section 2 gives an overview of few recent modified RSA algorithms and the disadvantages in those methods. The proposed algorithm ESRKGS is explained in detail in Section 3. The mathematical proof of the proposed method is explained in Section 4. In Section 5, a comparison of performance of the proposed method with other methods is done to show that the ESRKGS is not easily attacked by brute-force when compared to other systems. Finally, the conclusions of the present algorithm are discussed in Section 6.

2. Related works

Many researchers have given several proposals to modify the RSA algorithm. Few recent and major modifications proposed are discussed with the algorithms used in each case.

2.1. *RSA cryptosystem based on 'n' prime numbers (Ivy et al., 2012)*

In this work, RSA was modified with the introduction of four prime numbers. The algorithm is described as follows.

ALGORITHM 2.1. RSA CRYPTOSYSTEM BASED ON 'N' PRIME NUMBERS

```
Keygen()

INPUT:
Four large prime numbers $p$, $q$, $r$ and $s$.

OUTPUT:
Public Key Components: $\{e, n\}$
Private Key Components: $\{d, n\}$

PROCEDURE:
$n \leftarrow p * q * r * s.$

/*Compute Euler phi value of $n$ */
$\Phi(n) \leftarrow (p-1) * (q-1) * (r-1) * (s-1)$

Find a random number $e$, satisfying $1 < e < \Phi(n)$ and $\gcd(e, \Phi(n)) = 1$.

Compute a random number $d$, such that,
$d \leftarrow e^{-1} mod(\Phi(n))$.
```

The encryption and decryption algorithm are similar to the traditional RSA algorithm. The approach increases the security level of RSA because the time taken to find four prime numbers with the knowledge of n is greater than the time required to find two prime numbers of RSA. The complexity of the system lies in factoring of the public key n. With the existing factorization techniques (Ali and Salami, 2004; Lenstra, 1987), it is possible to factorize n and subsequently find the private key, making the system insecure.

2.2. *Encryption and decryption using secure RSA (Jamgekar and Joshi, 2013)*

In this approach, complexity was brought in the computation of cipher text from plain text. A similar level complexity was also used in the decryption part. The algorithm can be described as follows.

ALGORITHM 2.2. ENCRYPTION AND DECRYPTION USING SECURE RSA

```
Keygen()

INPUT:
Four large prime numbers $p$, $q$, $r$ and $s$.

OUTPUT:
Public Key Components: $\{n, m, g, e\}$
Private Key Components: $\{d, \lambda, \mu\}$

PROCEDURE:
$n \leftarrow p * q$
$m \leftarrow r * s$

/*Compute Euler phi value of $n$ */
$\varphi \leftarrow (p-1) * (q-1)$

/*Compute Euler phi value of $m$ */
$\lambda \leftarrow (r-1) * (s-1)$

Find a random number $e$, satisfying $1 < e < \varphi$ and $\gcd(e, \varphi) = 1$.

Compute the secret exponent $d$, satisfying $1 < d < \varphi$ such that, $e * d\ mod(\varphi) = 1$.

Select an integer $g \leftarrow m + 1$

/*Compute the modular multiplicative inverse */
$\mu \leftarrow \lambda^{-1} mod\ m$

Encrypt()

INPUT:
Plain text message, $S\ (< n)$

OUTPUT:
Cipher text, $C$

PROCEDURE:
Select random number $r$, such that $r < m$.
```

```
Compute Cipher Text,
$C \leftarrow g^{S\wedge(e\ mod n)} * r^m mod\ m^2$

Decrypt()

INPUT:
Cipher text message, $C$

OUTPUT:
Decrypted plain text, $S$

PROCEDURE:
Compute the original message $S$ as,
    $S \leftarrow (((C^{\lambda} mod\ m^2 - 1)/m) * \mu\ mod\ m)^d mod\ n$
```

This approach overcomes the disadvantage of the previous algorithm. This algorithm also uses four prime numbers and the key generation phase is similar to RSA. The system is highly secure because the encryption and decryption are not only dependent on n but also other new factors computed. The disadvantage of the algorithm is that the encryption and decryption computation is very complex. There are several parameters introduced whose need was not clearly justified. It thus increases the overhead of the system.

2.3. *Modified RSA algorithm* (Chhabra and Mathur, 2011)

The approach eliminated the need to transfer n. So it becomes difficult for the hackers to derive at the prime numbers used. The algorithm can be described as follows. The disadvantage of the system is that the attacker can easily attack the system with the value ofk_p and d known.

ALGORITHM 2.3. MODIFIED RSA ALGORITHM

```
Keygen()

INPUT:
Two large prime numbers $p$ and $q$.

OUTPUT:
Public Key Component: $k_p$
Private Key Component: $k_s$

PROCEDURE:
$n \leftarrow p * q$.

/*Compute Euler phi value of $n$ */
$\Phi(n) = (p-1) * (q-1)$
Find a random number $k_p$, satisfying $lg\ n < k_p < n$and
$gcd(k_p, n) = 1$.

Compute a random number $d$, such that,
If $p > q$, $n - p < d < n$ and $d$ must be co-prime to $n$
Else if $p < q$, $n - q < d < n$ and $d$ must be co-prime to
$n$
A general formula to find $d$ is, $k_p * k_s mod\ (d) = 1$
$k_s$can be found using the formula $k_s * k_p = 1\ mod\ (d)$
Encrypt()

INPUT:
Plain text message, $M (< n)$

OUTPUT:
Cipher text, $C$

PROCEDURE:
Compute Cipher Text,
$C \leftarrow m^{k_p} mod\ d$

Decrypt()

INPUT:
Cipher text message, $C$

OUTPUT:
Decrypted plain text, $M$

PROCEDURE:
Compute the original message $M$ as,
$M \leftarrow C^{k_s} mod\ d$
```

3. Proposed model

The proposed ESRKGS scheme focuses on removing the major issues of above discussed RSA systems. Most RSA systems are easily breakable because the computation of keys is based on n. This n can be easily found by factoring methods (Ali and Salami, 2004; Lenstra, 1987)because it is only a product of 2 primes. If this n is obtained, the hacker can easily find the keys and thus break the system. The major modifications which make the proposed system an efficient RSA algorithm are discussed in the following sections.

3.1. *ESRKGS key generation*

The proposed ESRKGS key generation involves the usage of four prime numbers. The value of E, D depends on the value of N, which is the product of 4 prime numbers. The computation of E is also not direct. The values e_1, e_2 are needed to find the value of E_1 thus increasing the time taken to attack the system. Only the value of n is kept as public and private component. Thus the attacker with the knowledge of n cannot determine all the primes which are the basis for finding the value of N and subsequently D. The parameter E_1 also increases the complexity of the system. For security purposes, the bit length of all the primes chosen is of same length as in case of traditional RSA. The algorithm is presented below.

ALGORITHM 3.1. ESRKGS KEY GENERATION

```
ESRKGS_Key_gen()

INPUT:
Four prime numbers p, q, r and s.

OUTPUT:
Public Key Components: {E, n}
Private Key Components: {D, n}

PROCEDURE:
$n \leftarrow p * q$
$m \leftarrow r * s$
$N \leftarrow n * m$
/*Compute Euler phi value of n and m */
$\Phi(n) \leftarrow (p-1) * (q-1)$
$\Phi(m) \leftarrow (r-1) * (s-1)$
/*Compute Euler phi value of N */
$\Phi(N) \leftarrow \Phi(n) * \Phi(m)$

Find a random number $e_1$, satisfying $1 < e_1 < \Phi(n)$ and $gcd(e_1, \Phi(n)) = 1$.

Find a random number $e_2$, satisfying $1 < e_2 < \Phi(m)$ and $gcd(e_2, \Phi(m)) = 1$

Compute $E_1 \leftarrow {e_1}^{e_2} mod\ N$

Find a random number E, satisfying $1 < E < \Phi(N) * E_1$ and $gcd(E, \Phi(N) * E_1) = 1$.

Compute a random number D, such that,
$D \leftarrow E^{-1} mod(\Phi(N) * E_1)$.
```

3.2. *ESRKGS encryption and decryption*

The encryption is done with the help of the public components and decryption is done with the help of the private components. The encryption and decryption are based on *n* but the computation of keys is not based on *n* but on N. This makes the system secure and not easily breakable. The algorithm is presented below.

ALGORITHM 3.2. ESRKGS ENCRYPTION & DECRYPTION

```
ESRKGS_Encrypt()

INPUT:
Plain text message, $M (< n)$
Public Key Components {E, n}

OUTPUT:
Cipher text, C

PROCEDURE:
$C \leftarrow M^E mod\ n$

Decrypt()

INPUT:
Cipher text message, C
Private Key Components: {D, n}

OUTPUT:
Decrypted plain text, P

PROCEDURE:
$P \leftarrow C^D mod\ n$
```

Let us discuss an example problem using the proposed ESRKGS algorithm.

ESRKGS-EXAMPLE.

- Take four large prime numbers $p = 79$, $q = 101$, $r = 109$ and $s = 89$.
- Compute, $n = p * q$ and $m = r * s$. $n = 7979$and $m = 9701$
- Compute, $N = m * n.N = 77404279$
- Compute Euler phi value of *n* and *m*,
  $\Phi(n) = (p-1) * (q-1)$
  $\Phi(m) = (r-1) * (s-1)$
  $\Phi(n) = 7800$and$\Phi(m) = 9504$
- Compute $\Phi(N) = \Phi(n) * \Phi(m)$.
  $\Phi(N) = 74131200$
- Find a random number e_1, satisfying
  $1 < e_1 < \Phi(n)$and $gcd(e_1, \Phi(n)) = 1$.
  $e_1 = 2761$
- Find a random number e_2, satisfying
  $1 < e_2 < \Phi(m)$and $gcd(e_2, \Phi(m)) = 1$.
  $e_2 = 587$
- Compute $E_1 = {e_1}^{e_2} mod\ N.\ E_1 = 74034755$
- Find a random number *E*, satisfying
  $1 < E < \Phi(N) * E_1$and $gcd(E, \Phi(N) * E_1) = 1$.
  $E = 4425692186722853$
- Compute a random number *D*, such that,
  $D = E^{-1} mod(\Phi(N) * E_1)$.
  $D = 4707099085177517$
- Input message, $M = 59$.
- Encryption,$C = M^E mod\ n.C = 2883$.
- Decryption, $P = C^D mod\ n.P = 59$

Table 1 – Performance of ESRKGS.

Length of p, q, r and s (in bits)	Key generation time (in ms)	Encryption time (in ms)	Decryption time (in ms)	Total execution time (in ms)
100	113	1.5	1.3	115.8
128	165	2	2	169
256	237	3	2	242
512	389	16	16	421
1024	1168	105	106	1379
2048	11,164	784	745	12,693
4096	181,811	6620	6647	195,078

4. Mathematical proof of ESRKGS

The proposed ESRKGS algorithm is proved mathematically in the following way.

The Cipher text is found using,

$$C = M^E \bmod n \tag{1}$$

and the plain text can be obtained by decryption of the Cipher text using

$$P = C^D \bmod n \tag{2}$$

The aim now is to get back the original message M from$C^D \bmod n$.

$$C^D \bmod n = M^{ED} \bmod n \tag{3}$$

From Algorithm 3.1,

$D \leftarrow E^{-1} \bmod (\Phi(N)*E_1)$.

Hence,

$$\begin{aligned} ED &= 1 \cdot \bmod (\Phi(N)*E_1) = 1 \cdot \bmod ((p-1)(q-1)(r-1)(s-1)*E_1) \\ &= 1 + k((p-1)(q-1)(r-1)(s-1)*E_1) \end{aligned} \tag{4}$$

(k is any positive integer)

Substituting (4) in (3)

$$\begin{aligned} C^D \bmod n &= M^{1+k((p-1)(q-1)(r-1)(s-1)*E_1)} \bmod n \\ &= \left(M*M^{k((p-1)(q-1)(r-1)(s-1)*E_1)}\right) \bmod n \\ &= M*\left(M^{(p-1)}\right)^{k(q-1)(r-1)(s-1)*E_1} \bmod p*q \\ &= M*\left(M^{(p-1)}\right)^{k(q-1)(r-1)(s-1)*E_1} \bmod p*q \\ &= M*M^{(p-1)k(q-1)(r-1)(s-1)*E_1} \bmod p*q \end{aligned}$$

(By Fermat's Little Theorem {Ribenboim, 1995}$M^{p-1} \equiv 1 \bmod p$)

$$= M*1^{k(q-1)(r-1)(s-1)*E_1} \bmod n = M \bmod n = M(\text{Since } M < n)$$

5. Implementation and results

For simulation purpose, ESRKGS is implemented using Java BigInteger library functions (Wagner, 2003). The user is allowed to enter the prime numbers or specify the bit length of the prime numbers to generate automatically using random function. BigInteger library provides operations for modular arithmetic, GCD calculation, primarily testing, prime generation, bit manipulation, and a few other miscellaneous operations. The implementation of the algorithm, is implemented in JAVA, running on a 2.50 GHz Intel ® Core ™ i5-3120M Processor and 8 GB RAM.

5.1. *Performance analysis*

The proposed algorithm ESRKGS was tested on varying bit sizes of inputs. The performance of the ESRKGS system in terms of key generation time, encryption time and decryption time is shown in Table 1.

Also, the performance of RSA algorithm by Rivest et al. (1978) (denoted as RSA1 henceforth)and RSA by Ivy et al. (2012) (denoted as RSA2 henceforth) is depicted in Table 2 and Table 3 respectively.

From the above tables, it can be seen that the time for key generation of ESRKGS is slightly greater than that of RSA1 and RSA2. The increased key generation time of ESRKGS can be justified by the fact that the time to break the system is high because of the complexity introduced. The encryption and

Table 2 – Performance of RSA1.

Length of p, q, r and s (in bits)	Key Generation Time (in ms)	Encryption Time (in ms)	Decryption Time (in ms)	Total Execution Time (in ms)
100	72	1	1	74
128	92	1.1	1.1	94.2
256	133	1	1.1	135.1
512	352	3	3	358
1024	889	21	22	932
2048	4315	183	169	4667
4096	91,542	1380	1381	94,303

Table 3 – Performance of RSA2.

Length of p, q, r and s (in bits)	Key generation time (in ms)	Encryption time (in ms)	Decryption time (in ms)	Total execution time (in ms)
100	110	2	1.7	113.7
128	144	2.5	2.2	148.7
256	216	4	3	223
512	313	21	23	357
1024	922	170	169	1261
2048	7471	1393	1379	10,243
4096	93,899	10,907	10,957	115,763

decryption time of all the RSA methods are compared in Figs. 1 and 2. From the graphs it can be seen that, the encryption and decryption time is higher than RSA1 because of the usage of 4 primes but it is less than RSA2. The increase in time is tolerable because the security is increased to a great extent in the proposed method. For example, for input primes of bit length 512, the encryption time of ESRKGS is 16 ms whereas for RSA1 it is only 3 ms but for RSA2 it is 21 ms. Similarly the decryption time for input of 512 bits in all the three cases is almost same as their respective encryption times.

5.2. *Security analysis*

There are various attacks possible on RSA which include the timing attack as shown in Carl (1996). The time to break an RSA system is equivalent to the time taken for finding the prime numbers used. This requires the factorization of the parameter 'n'. For this purpose, Elliptic Curve factorization Method (ECM) (Ali and Salami, 2004) and General Number Field Sieve (GNFS) (Lenstra, 1987) are used in common. GNFS and ECM are the first and third fastest known factoring methods respectively. ECM is commonly used for smaller number factoring whereas GNFS is capable of factoring integers larger than 100 digits. But in the proposed RSA algorithm (ESRKGS) even though the public key 'n' can be factored using any of the methods, this parameter is not sufficient enough in the computation of private key D. The above factoring techniques can be used to find p and q but the other two primes can be found only using a brute-force. In other words,

$$\tau_{\text{system}} = \tau_{p,\,q} + \tau_{\text{brute force}}$$

where,

τ_{system} = Time taken to break the system
$\tau_{p,\,q}$ = Time taken to find p and q using GNFS or ECM
$\tau_{\text{brute force}}$ = Time taken for brute-force attack

So, brute-force analysis is employed for this. The time taken for brute-force attack on ESRKGS, RSA1 and RSA2 is shown in Table 4. A graph is plotted based on the readings as shown in Fig. 3. For example, the time needed to find n by brute-force in the case of ESRKGS for input primes of bit length 12 is 234.95032s whereas in the case of RSA1 and RSA2 it is only 75.70676086s and 123.498494s respectively.

From the graph, it is clear that the time taken for brute-force attack on ESRKGS is far higher than that of other RSA schemes. This is based on the fact that only 'n' is known to the attacker but to find E and D the value of 'N' is needed thus making the system difficult to break.

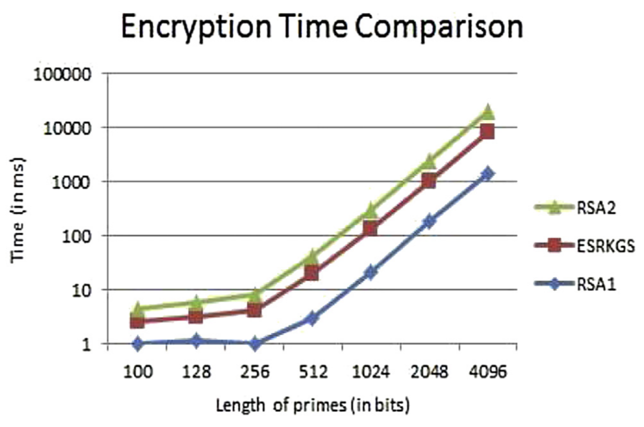


Fig. 1 – Encryption time comparison.

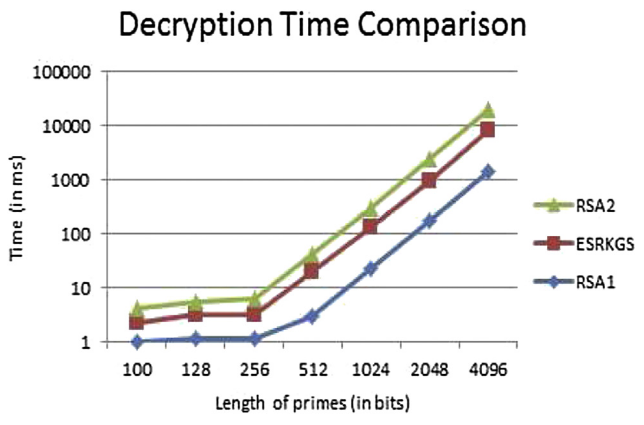


Fig. 2 – Decryption time comparison.

Table 4 – Brute-force attack time.

Length of p, q, r and s (in bits)	Time to find a 'n' by Brute-force (in s)		
	RSA1	RSA2	ESRKGS
6	1.083344	2.645654	7.855621
8	1.574581	5.383928	12.383894
10	4.047046	18.146578	32.492047
12	75.706760	123.498494	234.950321
16	14,961.466536	61,928.389156	126,473.290471

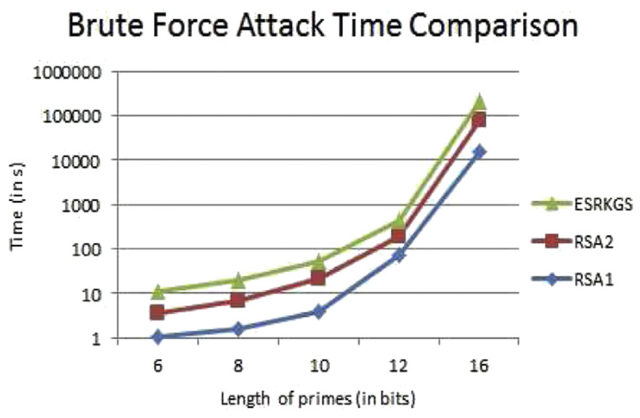


Fig. 3 – Brute-force time comparison.

6. Conclusion

In this paper, an enhanced RSA key generation algorithm called ESRKGS is proposed. The proposed work uses four large prime numbers instead of two prime numbers thereby increasing the attacking time needed to find these primes. The computation of keys *E*, *D* depends on the value of *N* (not *n*), which is the product of 4 prime numbers and the computation of *E* is also not direct. As a result the key generation time of ESRKGS is higher than traditional RSA and another compared RSA. The higher key generation time in turn increases the time needed to break the system, thereby making the system strong. The encryption and decryption time of ESRKGS is higher than RSA but significantly less than the other modified RSA which used four primes. Thus there is not much overhead or burden on the system. The performance of the algorithm is measured in terms of the time taken for brute-force attack. From the experiments, it is proved that the proposed algorithm is highly secure and not easily breakable as compared to RSA and the compared modified RSA algorithm.

REFERENCES

Al-Hamami AH, Aldariseh IA. Enhanced method for RSA cryptosystem algorithm. In: International Conference on Advanced Computer Science Applications and Technologies, Kuala Lumpur; 2012. p. 402–8.

Ali H, Salami MA. Timing attack prospect for RSA cryptanalysts using genetic algorithm technique. Int Arab J Inf Technol 2004;1(1):80–4.

Carl PA. Tale of two sieves. Notices Amer Math Soc 1996;43(12):1473–85.

Chhabra A, Mathur S. Modified RSA algorithm: a secure approach. In: International Conference on Computational Intelligence and Communication Networks, Gwalior; 2011. p. 545–8.

Forouzan BA. Cryptography and network security. Special Indian Edition. Tata McGraw-Hill; 2007. p. 2–11.

Ivy PU, Mandiwa P, Kumar M. A modified RSA cryptosystem based on ‘n’ prime numbers. Int J Eng Computer Sci 2012;1(2):63–6.

Jamgekar RS, Joshi GS. File encryption and decryption using secure RSA. Int J Emerg Sci Eng (IJESE) 2013;1(4):11–4.

Lenstra Jr HW. Factoring integers with elliptic curves. Ann Math 1987;126(3):649–73.

Liu Q, Li Y, Li T, Hao L. The research of the batch RSA decryption performance. J Comput Inf Syst 2011;7(3):948–55.

Mahajan S, Easo S. Performance evolution of RSA and new cryptosystem. Int J Emerg Technol Adv Eng 2012;2(3):279–83.

Minni R, Sultania K, Mishra S, Vincent DR. An algorithm to enhance security in RSA. In: Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode; 2013. p. 1–4.

Rama Chandra Rao GAV, Lakshmi PV, Ravi Shankar N. A novel modular multiplication algorithm and its application to RSA decryption. Int J of Comput Sci Issues 2012;9(6):303–9.

Ribenboim P. In: The New Book of Prime Number Records. 3rd ed., Vol. 49. New York: Springer-Verlag; 1995. p. 22–5.

Rivest RL, Shamir A, Adleman LA. Method for obtaining digital signatures and public-key cryptosystems. Commun ACM 1978;21(2):120–6.

Segar TC, Vijayaragavan R. Pell's RSA key generation and its security analysis. In: Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode; 2013. p. 1–5.

Sharma S, Sharma P, Dhakar RS. RSA algorithm using modified subset sum cryptosystem. In: International Conference on Computer & Communication Technology (ICCCT), Allahabad; 2011. p. 457–61.

Stallings W. Cryptography and network security: principles and practice. 5th ed. Pearson Education; 2011. p. 121–44. 253–97.

Wagner NR. The Laws of Cryptography with Java Code. Tech Rep 2003:78–112.

Wu CH, Hong JH, Wu CW. RSA cryptosystem design based on the Chinese remainder theorem. In: Design Automation Conference, Proceedings of the ASP-DAC, Yokohama; 2001. p. 391–5.