Magnifying the Key Stability of RSA Algorithm in Data Security for Public Cloud

M. Kamal[1†] and Dr. G. Ravi[2††],
[1]Research Scholar, [2]Associate Professor and Head

PG & Research Department of Computer Science, Jamal Mohamed College (Autonomous)
(Affiliated to Bharathidasan University)
Tiruchirappalli-620 020, Tamilnadu

Abstract
Cloud computing is a web-based application that provides users with computing, software, infrastructure, platform, devices, and other resources on a pay-as-you-go basis. Security is the most important factor in cloud computing to ensure that cloud data is kept in safe mode. Data security is a challenging research part in any cloud environment. When cloud users upload their confidential and secret data over the cloud and security must be rendered to such secret data. In this article, a new algorithm is introduced to generate 'n' prime numbers to compute both public and private keys for strengthening the data security. The encryption and, decryption time, and security level are analysed using the Hackman tool. Encryption and decryption power are measured by OPNET tool. The results show that the highest value for the proposed algorithm in all five different file sizes all the parameters. In this proposed algorithm, brute force attack and dictionary attack have been used to analyse the ciphertext.

Keywords:
Cloud computing, Data security, RSA, Encryption, Decryption and Ciphertext, MKRSA.

1. Introduction

The term cloud states to a Network or the Internet. Cloud can provide services through public and private network, such as Wide Area Network, Local Area Network or Virtual Private Network. Applications like E-Mail, Web Conferencing, and Customer Relationship Management (CRM) are running in the cloud. Cloud computing refers to remotely managing, configuring and accessing hardware and software resources, which provides an online data storage, infrastructure and application. It contains many unique security issues and challenges. In the cloud, data is stored to a third-party provider and accessed over the Internet which means that the visibility and control of the data are limited. It also raises the question of how to properly protect it. It is inevitable that everyone understands the respective roles and security issues inherent in cloud computing.

Most of the security risks of cloud computing are related to data security. Whether it is due to lack of data visibility, inability to control data, or data theft in the cloud. Most of the problems are due to cloud data customers. Data security is a set of policies and technologies that protect data against intentionally or inadvertently damaging, altering or disclosing data. Data protection can be implemented by a variety of technologies and techniques, including administrative controls, physical security, logical checks, organization and other security techniques that restrict access to unauthorized or malicious users or processes. Cryptography provides an essential security to achieve data security. Cryptography makes the data vulnerable to malicious attacks and has made incredible strides in maintaining confidentiality in stored data. The term 'crypt' refers to 'hidden/vault' whereas 'graph' refers to 'write'. Cryptography is an application to secure the information from many suspicious and malicious third parties - enemies. Cryptography is the process of converting data into a public network into unpredictable code [1]. Symmetric algorithm is used to encrypt the plain text and decrypt the ciphertext using the public key. The asymmetric algorithm is also used to encrypt the plain text using the public key and decrypt the ciphertext using the private key. The asymmetric algorithm takes longer to encrypt the text than the symmetrical algorithm [2].

RSA cryptosystem is based on the dramatic difference between the ease of detecting large primes and the difficulty of factoring in the production of only two prime numbers. In encryption process of RSA, public key can encrypt a message, and private key is used to decrypt it. It provides a method to ensure the confidentiality, integrity, and reliability of electronics communication and data storage [3].

Manuscript received June 5, 2022
Manuscript revised June 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.6.101

Hackman tool is a main tool that provides simple text files for the applications, which is evaluated, and tries to decode each encoded file into all plain text file. Hackman has an embedded hacking mechanism. It can be enabled or disabled with the help of BCCC.LIB. It can also change the depth of the attack if the programmer decides to do so. But the default level is maximum strength hacking. If one plans to evaluate the strength of the cryptographic algorithm, it is strongly recommended not to change the default settings.

A brutal attack is a cryptographic hack that relies on guessing possible combinations of the target password until the correct password is found. Password is long, and combinations must be checked. However, if the password is weak, it may take seconds without any effort. Weak passwords are like fishing in a barrel for attackers, which is why all companies must implement a strong password policy on all users and systems.

Dictionary attack is the attack vector that an attacker uses to break into the system, and it is password protected by technically placing each word in the dictionary as a password for that system. The dictionary may contain words from an English dictionary and some leaked lists of passwords that are commonly used, and can be useful and quick at times if combined with common letters that replace numbers.

2. Literature Review

Rahul Saxena et.al., presented the conversion of OpenMP based algorithms to the RSA algorithm code to improve the execution time using the parallel processing power of modern multi-core architecture-based machines [4]. Abdeldaym et.al., used two keys, namely public key and private key which takes four prime numbers for calculating keys. Instead of sending a public key directly, sends two public keys to the receiver. So that the speed of the encryption process decreases [5]. Shereek et.al., found that the generation of the public and private key in RSA is optimized using the prime test, can be surpassed by Fermat's little theorem [6].

Kanasei et al, used a combination of asymmetric and symmetric encryption techniques (i.e., RSA and AES encryption methods) to achieve the guarantees of cloud data security [7]. Patel et.al., deals the random prime numbers used to create keys of public and private for encrypt and decrypt the data. Therefore, it improved the data security individually or collectively in a cloud environment [8]. Amalarethinam et.al., reduces the time of encryption and decryption processes by dividing the file into blocks and enhances the strength of the algorithm by increasing the key size. This strength allows users to store data on the cloud without any hassle [9].

In Enhanced RSA (ERSA) algorithm offers four prime numbers, used to create the public key process, and two prime numbers were used to create a private key. Data was encrypted and decrypted using those keys. As compared with RSA, ERSA gives better results [10]. Hybrid Security algorithm for RSA (SRSA) provides a new hybrid protection algorithm for the RSA cryptosystem called, Hybrid RSA (HRSA). Here the calculation of public key and private key depends on the value product of four prime numbers. In this method, the calculation of public key and private key still involves for some intermediate factors. From this, the generation of keys very complicated. So that the encryption and decryption process of data takes a very long time [11]. In Enhanced and Secured RSA public-key cryptosystem (ESRPKC) algorithm deals with the Chinese residual theorem. It offers four prime numbers instead of two primes to create public and private key pairs, again it increases the system complexity [12]. In Improved Key Generation Scheme using RSA [IKGSR] algorithm focused the speed of the system was increased as compared to the other RSA algorithms. But the use of four prime numbers takes extra time to calculate both public and private-key [13].

3. Proposed Algorithm

The proposed algorithm is Magnifying the Key Stability of the RSA algorithm (MKRSA) for Data Security in Public Cloud Storage and to reduce the encryption and decryption time, encryption and decryption power of the data as well.

It composes eight steps for generating keys in encryption and decryption processes.

Step 1: Generate a set ‘P’ prime numbers ranging from X and Y.

Step 2: Select ‘N’ prime numbers (p1, p2, p3… pn) from P.

Step 3: Calculate the values of N_1 and N_2

$$N_1 = \prod_{i=1}^{n} p_i \qquad Eq.(1)$$

If (n mod 2 == 0)

$$N_2 = \prod_{i=1}^{\frac{n}{2}} p_i \qquad Eq.(2)$$

else

$$N_2 = \prod_{i=1}^{n-2} p_i \qquad Eq.(3)$$

Step 4: Compute $\Phi(N_1)$

$$\Phi(N_1) = \prod_{i=1}^{n}(p_i - 1) \qquad Eq.(4)$$

Step 5: Calculate the Public Key (E):

$$\text{GCD}\left(E, \Phi(N_1)\right) = 1 \qquad Eq.(5)$$

Step 6: The Private Key (D) is derived from

$$D \times E = 1 \times \text{mod}\left(\Phi\left(D \times E = 1 \times \text{mod}\left(\Phi(N_1)\right)\right)\right) \qquad Eq.(6)$$

Through E and N_1, the public key component is formed. Similarly, the pairs D and N_2 were used for framing the private key.

Step 7: Encryption Process

The Ciphertext (CT) from the given Plaintext (PT) is

$$CT = PT^E \ mod \ N_1 \qquad Eq.(7)$$

Step 8: Decryption process

PT can be originating by

$$PT = CT^D \ mod \ N_2 \qquad Eq.(8)$$

The proposed methodology of MKRSA is explained as bellow:

Step 1: 'P' is a set of prime numbers.

Step 2: The 'N' prime numbers are extracted from 'P' set.

Step 3: The N_1 value is calculated using equation 1. When the value of N is even, N_2 is calculated from equation 2 or else N_2 is calculated from equation 3.

Step 4: The Euler Totient value is calculated using step 3.

Step 5: The public key (E) is picked in such a way that the Greatest Common Divisor of E and the Euler Totient value of r is equal to 1.

Step 6: The private key (D) is calculated using step 5.

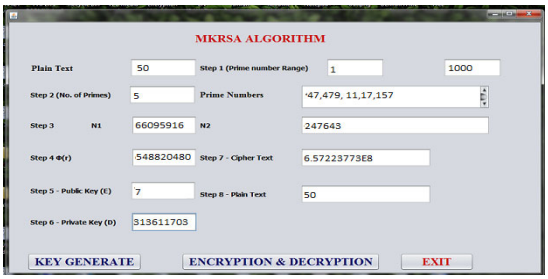
Step 7: This step performs the process of converting PT into CT. In this process, CT is generated through the pair of E and N_1 values.

Step 8: PT is retrieved from CT using the private key pair of D and N_2.

4. Illustrative Example

The MKRSA algorithm was implemented using Java language. The Steps 1 to 6 were used to generate both E and D keys and Step 7 & 8 were used to encrypt and decrypt the PT. Figure 1 shows the computational process of MKRSA.

Figure 1: Computational process of MKRSA algorithm



Step 1: Calculate a set of prime numbers ranging 1 to 1000.

In Step1, the values 1 & 1000 are assigned to X & Y respectively. The following prime numbers are found namely 1, 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59… 967, 971, 983, 991 and 997.

Step 2: If 'N' value is 5, then any five prime numbers are randomly chosen from step 1. For an example, 47, 479, 11, 19 and 157 are selected prime numbers.

Step 3: Compute the values of N_1 and N_2 by

$$N_1 = \prod_{i=1}^{n} p_i$$

$N_1 = 47 \times 479 \times 11 \times 17 \times 157$

$N_1 = 660959167$

Here N_1 is calculated from selected prime numbers. Then, N_2 value is calculated using 3 prime numbers from the selected number (Step 2) because 'N' is an odd number.

$$N_2 = \prod_{i=1}^{n-2} p_i$$

$N_2 = 47 \times 479 \times 11$

$N_2 = 247643$

Step 4: Compute $\Phi(N_1)=\Phi(N_1) = \prod_{i=1}^{n}(p_i - 1)$

$\Phi(N_1) = (47-1) \times (479-1) \times (11-1) \times (17-1) \times (157-1)$

$\Phi(N_1) = 548820480$

Step 5: Find the Public Key (E), i.e. Greatest Common Divisor $(E, \Phi(N_1)) = 1$

GCD (E, 040) = 1

E = 7

Step 6: 'D' is calculated from D × E = 1 × mod (Φ (N_1))

D × 7 = 1 mod 617423040 = 313611703

Here, E & N_1 are representing the public key component (7, 660959167). Similarly, D & N_2 are representing the component of private key (313611703, 247643).

Step 7: The CT was found through the equation 7. Here the PT is assumed as 50.

$CT = PT^E \bmod N_1$

$CT = 50^7 \bmod 660959167$

$CT = 6.57223773E8$

Step 8: The PT is decrypted through the equation 8.

$PT = CT^D \bmod N_2$

$PT = 6.57223773E8^{176406583} \bmod 247643$

$PT = 50$

Based on the above steps, the PT was decrypted. Figure 1 shows the diagrammatic representation of the proposed algorithm.

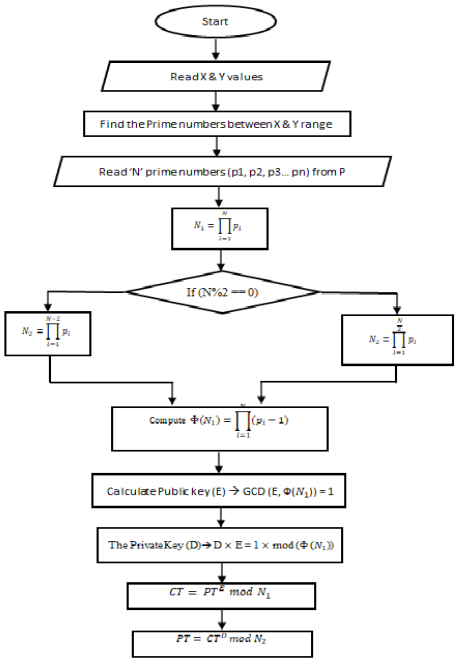


Figure 2: Figure 1: Work Flow diagram of MKRSA

Figure 3 shows the implementation of the MKRSA algorithm in cloud computing. SaaS is also known as "on-demand software". It is a software distribution model that is facilitated by a cloud service provider. These features are available to end users via the Internet, so end users do not need to set up any software on their devices to access these features.

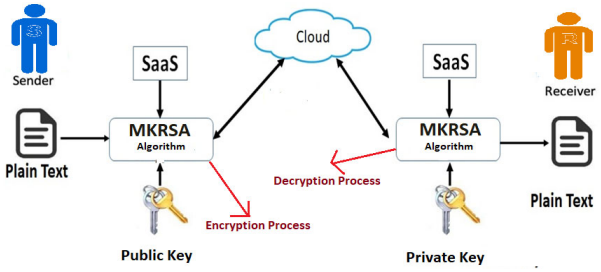


Figure 3: Implementation of the MKRSA in Public Cloud

5. Methodology Evaluation

The results were analyzed by two simulators called All Block Cipher Hackman tool and OPNET tool. The CT was generated using the proposed MKRSA algorithm. The Hackman tool was used to measure the encryption time, decryption time and security level of CT. The OPNET tool was used to calculate the encryption power and decryption power of CT which was generated by the proposed MKRSA algorithm.

The level of security is measured through Hackman tool by creating a random hash table with hacking methods such as Brute Force and Dictionary attacks. These attacks expose encrypted data blocks during the hacking process. Hackman tool analyzes the security level of the proposed algorithm of MKRSA with existing algorithms.

The Security level is calculated as

$$\frac{\partial_c}{\partial_t} \times 100 \qquad Eq.(9)$$

where, ∂_c is the compromised data blocks and ∂_t is the total number of blocks in the encrypted data.

The OPNET simulator tool is used to view and analyze the results [14]. Using this tool, two parameters encryption power and decryption power are calculated for five different encryption algorithms, including the proposed MKRSA algorithm.

OPNET is capable of measuring the average power applied to one node by monitoring the power consumption of all nodes. It supports multiple network environments with different values for voltage V and current I. It handles the voltage and current information of all nodes as a whole.

$$P = \{(V_1, I_1), (V_2, I_2) \dots (V_n, I_n)\}$$

The average power consumption for a network transaction is calculated using the following formula.

$$P_a = \frac{1}{n}\sum_{i=1}^{n}(V_i \times I_i) \qquad Eq.(10)$$

where, n is the number of nodes

6. Experimental Results and Discussion

This section displays the results obtained by running the simulation program using different sizes of data loads. The results show the correlation for changing the data load in each algorithm and the impact of the cyber mode.

6.1 Performance Evaluation

Five different sizes of files are taken and implemented in the MKRSA algorithm. The considered file sizes are 1 MB, 2 MB, 3 MB, 4 MB and 5 MB. The MKRSA technique is compared with other four algorithms on five different parameters of encryption time, decryption time, encryption power, decryption power and security level. The results are obtained by running all the four existing techniques on the same computing system.

Table 1a: Encryption Time (mS)

Data (MB)	ERSA	HSRSA	ESRSA	IKGSR	MKRSA
1	2668	2794	2435	2889	2351
2	4888	5164	4587	5222	4233
3	7693	7940	6976	8227	6618
4	10312	10663	9330	10984	8975
5	12947	13588	11701	13963	11457

Table 1b: Time Differences of Encryption - MKRSA vs Existing Algorithms (mS)

Data / Existing Algorithms	1 MB	2 MB	3 MB	4 MB	5 MB
ERSA	317	655	1075	1337	1490
HSRSA	443	931	1322	1688	2131
ESRSA	84	354	358	355	244
IKGSR	538	989	1609	2009	2506

Encryption time is the time taken by the processor to encrypt a given text by performing various functions. These functions are defined in the respective encryption algorithms. Encryption and decryption power are measured in milliseconds (mS). Table 1a shows the comparison of encryption time of existing algorithm with that of the proposed algorithm. The proposed algorithm takes very less time as compared to existing algorithms (Refer Table 1b). When the data size is increased, the MKRSA algorithm takes minimum encryption time as compared to that of the existing algorithms. Figure 4 shows the encryption time of all the algorithms including proposed algorithm. The MKRSA algorithm gives the lowest encryption time than the other techniques.

Table 2a: Decryption Time (mS)

Data (MB)	ERSA	HSRSA	ESRSA	IKGSR	MKRSA
1	2592	2787	2446	2919	2325
2	4962	5219	4580	5274	4370
3	7681	7945	6855	8301	6778
4	10309	10742	9370	11196	9084
5	13038	13474	11688	14091	11488

Table 2b: Time Difference of Decryption - MKRSA vs Existing Algorithms (mS)

Data / Existing Algorithms	1 MB	2 MB	3 MB	4 MB	5 MB
ERSA	267	592	903	1225	1550
HSRSA	462	849	1167	1658	1986
ESRSA	121	210	77	286	200
IKGSR	594	904	1523	2112	2603

Decryption is an important process to predict the plain text. Figure 5 shows the various files sizes (Data size) vs decryption time of all algorithms. The proposed method gives the lowest decryption time

than other algorithms. Table 2a shows that the comparison of decryption time with the proposed algorithm. The proposed algorithm takes very less time as compared to other algorithms (Refer Table 2b). Decryption time is the time taken by the processor to convert the CT into PT.

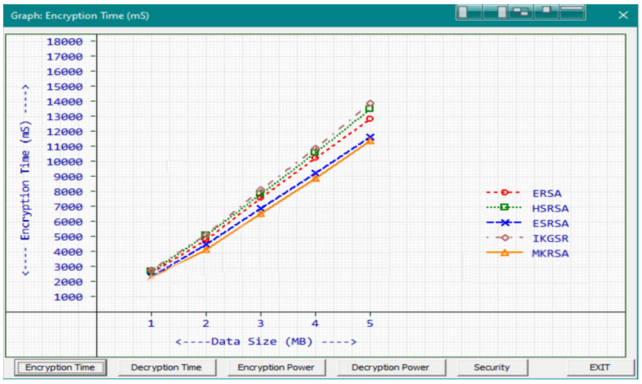


Figure 4: Encryption Time (mS)

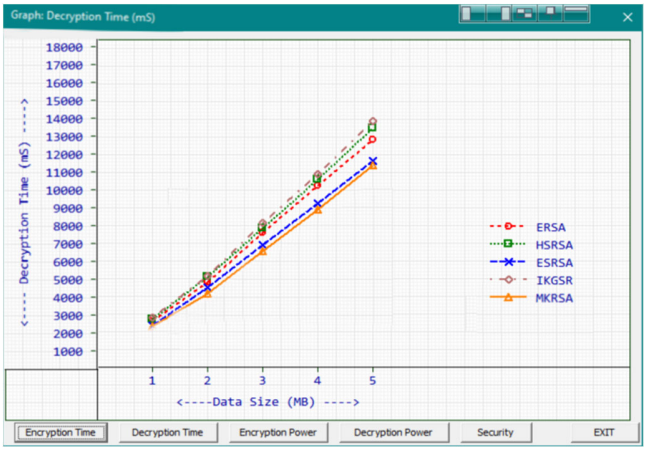


Figure 5: Decryption Time (mS)

From equation 10, the average encryption power and decryption power are calculated. They are measured in Milli Watts (mW). **Table 3a & 4a** show the computational power taken by the tool during the conversion process. Tables 3a & 4a indicate the comparison of the power taken by the machine during the encryption and decryption process. **Table 3b & 4b** shows the power differences of encryption and decryption process of the MKRSA algorithm and other existing algorithms. Figure 6 & 7 show the data size vs power taken by the machine at the time of encryption and decryption process. The MKRSA algorithm takes high power consumption as compared to the existing algorithms. The proposed algorithm allocates the least amount of time to encrypt and decrypt the given data. Due to this reason, the MKRSA algorithm reduces the machine power.

Table 3a: Encryption Power (mW)

Data (MB)	ERSA	HSRSA	ESRSA	IKGSR	MKRSA
1	992	1034	905	1087	874
2	1814	1931	1699	1946	1578
3	2872	2954	2601	3057	2454
4	3839	3971	3469	4071	3330
5	4815	5040	4357	5171	4258

Table 3b: Encryption Power Differences- MKRSA vs Existing Algorithms (mW)

Data / Existing Algorithms	1 MB	2 MB	3 MB	4 MB	5 MB
ERSA	118	236	418	509	557
HSRSA	160	353	500	641	782
ESRSA	31	121	147	139	99
IKGSR	213	368	603	741	913

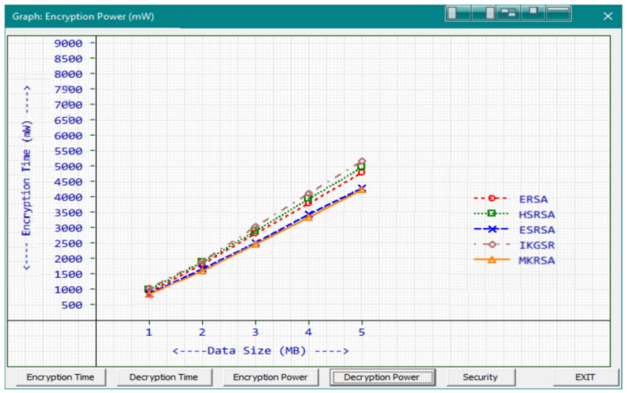


Figure 6: Encryption Power (mW)

The security levels of encryption techniques are measured using equation 9. The strength of an encryption algorithm is determined by the computational logic that provides the incomprehensible encryption.

Table 4a: Decryption Power (mW)

Data (MB)	ERSA	HSRSA	ESRSA	IKGSR	MKRSA
1	962	1033	907	1086	883
2	1852	1942	1700	1969	1630
3	2867	2949	2551	3086	2522
4	3838	3995	3476	4168	3378
5	4837	5008	4342	5221	4276

Table 4b: Decryption Power Difference- MKRSA vs Existing Algorithms (mW)					
Data / Existing Algorithms	1 MB	2 MB	3 MB	4 MB	5 MB
ERSA	79	222	345	460	561
HSRSA	150	312	427	617	732
ESRSA	24	70	29	98	66
IKGSR	203	339	564	790	945

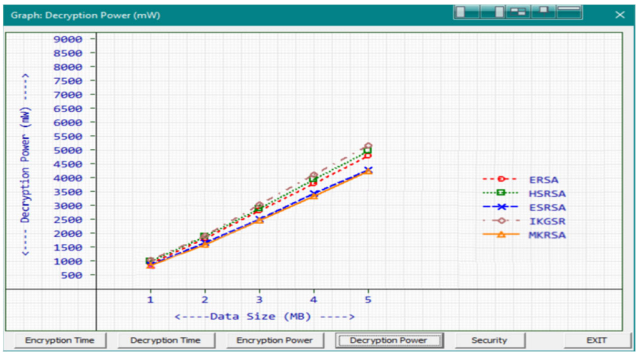


Figure 7: Decryption Power (mW)

The security levels of encryption techniques are measured using equation 9. The strength of an encryption algorithm is determined by the computational logic that provides the incomprehensible encryption. The security measure of encryption techniques such as ERSA, HSRSA, ESRSA, IKGSR and MKRSA are shown in Table 5. In Table 5, MKRSA algorithm shows the high security level, since MKRSA algorithm takes more than four prime numbers for calculating public and private keys. When the number of N values are increased, the N1 and N2 values are also increased. So that, E and D are strengthen. As a result, the security level of PT is enhanced. Figure 8 shows the comparison of security level of various existing algorithms with proposed approach.

Table 5: Security Level (%)					
Data (MB)	**ERSA**	**HSRSA**	**ESRSA**	**IKGSR**	**MKRSA**
1	89.02	90.15	88.65	90.59	91.97
2	89.38	89.57	86.62	91.89	91.91
3	89.18	90.41	87.67	89.27	92.43
4	87.70	91.16	88.62	89.72	93.75
5	88.16	89.30	87.47	89.09	92.38

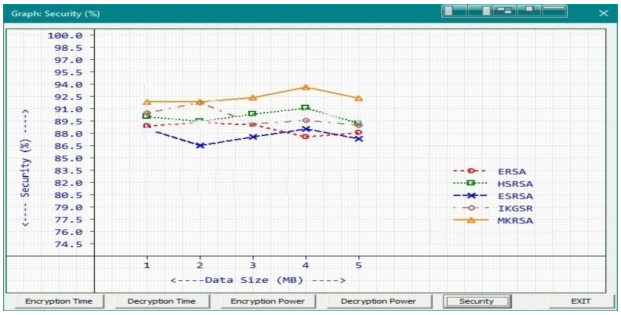


Figure 8: Security Level (%)

7. Conclusion

The proposed MKRSA algorithm uses 'n' prime numbers to generate the private and public keys. While increasing the key size of MKRSA algorithm, it manages complicated computations to the unauthorized users to identify the keys. Since, the key is very strong, the security of data will be increased automatically. The proposed MKRSA algorithm is very efficient in encryption and decryption time, encryption and decryption power and security level as compared with ERSA, HSRSA, ESRSA, and IKGSR algorithms.

Appendix

Hardware Configuration:

Processor: Intel ® Core ™ i5 7200U CPU @ 2.70 GHz
RAM: 8GB
Architecture: x64-bit based processor
Hard disk: 1 TB

Software Configuration:

OS: Windows 10 64-bit
IDE: Visual Studio
Programming Language: Visual C++
Run Time Library: Advanced C
Simulation Tool: OPNET

Simulation Parameters:

Area: 1000 x 1000 meters
Number of Nodes: 50 Wired & Wireless Random Mix
Node Placement: Random Distribution
Traffic Type: Typical real-world random traffic

Parameters for Comparison:

1. Encryption Time
2. Decryption Time

3. Encryption Power
4. Decryption Power
5. Security

Data Size: 1MB to 5MB in step of 1MB

References

[1] Kamal, M., Ravi, G. A survey on data security in cloud computing using cryptography algorithms. International Journal of Innovations in Engineering and Technology (IJIET), 16(1), 1-5, 2020.

[2] Sharma, J., & Thapa, R. Hybrid approach for data security using RSA and LSB Algorithm. In Proceedings of IOE Graduate Conference (pp. 181-186), 2019.

[3] Singh, S. K., Manjhi, P. K., & Tiwari, R. K. Data Security Using RSA Algorithm in Cloud Computing. International Journal of Advanced Research in Computer and Communication Engineering, 5(8), 11-16, 2016.

[4] Saxena, R., Jain, M., Singh, D., & Kushwah, A. An enhanced parallel version of RSA public key crypto based algorithm using OpenMP. In Proceedings of the 10th International Conference on Security of Information and Networks (pp. 37-42), 2017.

[5] Abdeldaym, R. S., Abd Elkader, H. M., & Hussein, R. Modified RSA algorithm using two public key and chinese remainder theorem. IJ of Electronics and Information Engineering, 10(1), 51-64, 2019.

[6] Shereek, B. M. Improve Cloud Computing Security Using RSA Encryption With Fermat's Little Theorem. IOSR Journal of Engineering, 4, 1, 2014.

[7] Khanezaei, N., & Hanapi, Z. M. A framework based on RSA and AES encryption algorithms for cloud computing services. In 2014 IEEE Conference on Systems, Process and Control (ICSPC 2014) (pp. 58-62). IEEE, 2014.

[8] Patel, S. R., & Shah, K. Security Enhancement and Speed Monitoring of RSA Algorithm. International Journal of Engineering Development and Research, 2(2), 2057-63, 2014.

[9] Amalarethinam, I. G., & Leena, H. M. (2017, February). Enhanced RSA algorithm with varying key sizes for data security in cloud. In 2017 World Congress on Computing and Communication Technologies (WCCCT) (pp. 172-175). IEEE, 2017.

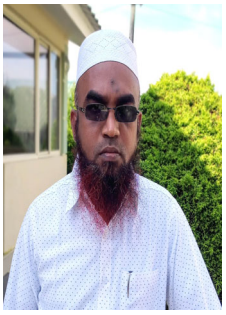
[10] Amalarethinam, D. G., & Leena, H. M. Enhanced RSA Algoirithm for Data Security in Cloud. International Journal of Control Theory and Applications, 9, 147-152, 2016.

[11] Panda, P. K., & Chattopadhyay, S. A hybrid security algorithm for RSA cryptosystem. In 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 1-6). IEEE, 2017.

[12] Kumar, V., Kumar, R., & Pandey, S. K. An enhanced and secured RSA public key cryptosystem algorithm using Chinese remainder theorem. In International Conference on Next Generation Computing Technologies (pp. 543-554). Springer, Singapore, 2017.

[13] Chinnasamy, P., & Deepalakshmi, P. Improved key generation scheme of RSA (IKGSR) algorithm based on offline storage for cloud. In Advances in Big Data and Cloud Computing (pp. 341-350). Springer, Singapore, 2018.

[14] Menaka, R., Ramesh, R., & Dhanagopal, R. Behavior based fuzzy security protocol for wireless networks. Journal of Ambient Intelligence and Humanized Computing, 1-16, 2020.



M. Kamal has received his Master of Science in Computer Science from Jamal Mohamed College, Tiruchiarappalli, Tamilnadu. Currently, he is a Research Scholar in Computer Science and working as an assistant professor in Computer Science, Jamal Mohamed College (Autonomous), Tiruchirappalli affiliated to Bharathidasan University, Tamilnadu. He has published many research papers in National and International Journals. His research area is Cloud Computing. He has attended several National and International Conferences and workshop.

Dr. G. RAVI is working as Associate Professor & Head, Department of Computer Science, Jamal College, Tiruchirappalli, Tamil Nadu, India. He has 33 years of experience in teaching and 24 years of experience in research. He has published more than 35 research articles in the International & National Conferences and Journals. He has acted as a Resource Person many Colleges in and around Tamilnadu. He is also the Member of the Board of Studies in many Universities and many Autonomous Colleges. His research areas are Cloud Computing, Big Data, Data Mining, and Mobile Networks, Wireless Sensor Networks and Machine and Artificial Intelligence.