



International Journal of Software and Web Sciences (IJSWS)

www.iasir.net

An improved RSA Encryption Algorithm for Cloud Computing Environments: Two key Generation Encryption (2KGEA)

¹Ms. Shubhra Saggar, ²Dr. R.K. Datta

¹Research Scholar, Singhania University.

Faculty, Guru Nanak Institute of Manegment, New Delhi

²Director, M.E.R.I.T. New Delhi

Abstract: Cloud computing is an emerging technology that is still unclear to many security problems. Ensuring the security of stored data in cloud servers is one of the most challenging issues in such environments. The main aim of this project is to use the cryptography concepts in cloud computing communications and to increase the security of encrypted data in cloud servers with the least consumption of time and cost at the both of encryption and decryption processes. In the proposed model, the number of key generation exponents has been increased in comparison to the original RSA. Moreover, a dual encryption process has been applied in this algorithm to prevent common attacks against RSA algorithm.

Index Terms—Cloud computing, security, cryptography, RSA, sharing.

I. INTRODUCTION

With regard to data-in-transit, the primary risk is in not using a vetted encryption algorithm. Although this is obvious to information security professionals, it is not common for others to understand this requirement when using a public cloud, regardless of whether it is IaaS, PaaS, or SaaS. It is also important to ensure that a protocol provides confidentiality as well as integrity (e.g., FTP over SSL [FTPS], Hypertext Transfer Protocol Secure [HTTPS], and Secure Copy Program [SCP])—particularly if the protocol is used for transferring data across the Internet. Merely encrypting data and using a non-secured protocol (e.g., “vanilla” or “straight” FTP or HTTP) can provide confidentiality, but does not ensure the integrity of the data (e.g., with the use of symmetric streaming ciphers).

Although using encryption to protect data-at-rest might seem obvious, the reality is not that simple. If you are using an IaaS cloud service (public or private) for simple storage (e.g., Amazon’s Simple Storage Service or S3), encrypting data-at-rest is possible—and is strongly suggested. However, encrypting data-at-rest that a PaaS or SaaS cloud-based application is using (e.g., Google Apps, Salesforce.com) as a compensating control is not always feasible. Data-at-rest used by a cloud-based application is generally not encrypted, because encryption would prevent indexing or searching of that data.

Generally speaking, with data-at-rest, the economics of cloud computing are such that PaaS based applications and SaaS use a multitenancy architecture. In other words, data, when processed by a cloud-based application or stored for use by a cloud-based application, is commingled with other users’ data (i.e., it is typically stored in a massive data store, such as Google’s BigTable). Although applications are often designed with features such as data tagging to prevent unauthorized access to commingled data, unauthorized access is still possible through some exploit of an application vulnerability. Although some cloud providers have their applications reviewed by third parties or verified with third-party application security tools, data is not on a platform dedicated solely to one organization.

II. Cryptography

Cryptography is the most common technique for ensuring a secure communication between two parts in the presence of a third party. If *A* (Alice) and *B* (Bob) send messages to each other, and they do not want others to read or change the content of their messages, then it means that they want to have a secure communication. In this communication, a transmission medium *T* is used, i.e. *A* sends his message to *B* via *T*. A third party, who wants to interfere this communication by accessing/changing the message, is called an intruder *I*. whenever a message is on its way towards the destination, it is in danger of being accessed by *I*, who can perform the following actions:

1. He can *block* the message, so it never reaches its destination, and thus the availability is violated.
2. He can *intercept* the message, so it is not secret anymore, and thereby the confidentiality is destroyed.
3. He can *change* the content of the message, and by that the integrity is violated.

4. He can *fake* a message and impersonate the sender *A*, and send the message to *B*. This violates also the integrity of the message.

A. Encryption & Decryption

Encryption is the process of translating plain text data (plaintext) into something that appears to be random and meaningless (ciphertext). Decryption is the process of converting ciphertext back to plaintext.

To encrypt more than a small amount of data, symmetric encryption is used. A symmetric key is used during both the encryption and decryption processes. To decrypt a particular piece of ciphertext, the key that was used to encrypt the data must be used.

The goal of every encryption algorithm is to make it as difficult as possible to decrypt the generated ciphertext without using the key. If a really good encryption algorithm is used, there is no technique significantly better than methodically trying every possible key. For such an algorithm, the longer the key, the more difficult it is to decrypt a piece of ciphertext without possessing the key.

1. Symmetric Algorithms

There are two most famous symmetric algorithms, namely Data Encryption Standard (DES) and Advanced Encryption Standard (AES). Another symmetric algorithm, which is a public domain algorithm, is called Blowfish. We will briefly explain these algorithms and their security issues.

1.1. Data Encryption Standard (DES)

One of the well-known symmetric algorithms is Data Encryption Standard (DES). It is a block cipher with the block size of 64 bits.

1.2. Advanced Encryption Standard (AES)

AES is a block cipher with a block size of 128 bits. The key length for AES is not fixed, so it can be 128, 192, 256 and possibly more bits.

2. Asymmetric Algorithms

One of the main reasons why asymmetric cryptography was invented is because symmetric cryptography is not suitable for communication in a big network with a large number of users. There is a key distribution problem. Each user has to have/remember the secret key of all other users, with whom he communicates.

2.1. The RSA algorithm

The most commonly used asymmetric algorithm is Rivest-Shamir-Adleman (RSA). It was introduced by its three inventors, Ronald Rivest, Adi Shamir and Leonard Adleman in 1977. It is mostly used in key distribution and digital signature processes. RSA is based on a one-way function in number theory, called “integer factorisation”. A one-way function is a function, which is “easy” to compute one way, but “hard” to compute the inverse of it. Here easy and hard should be understood with regard to computational complexity, especially in terms of polynomial time problems.

2.2. RSA CRYPTOSYSTEM

The original RSA algorithm was publicly described in 1977 [13] and after that many similar algorithms were designed based of original RSA in order to rectify the weaknesses of the basic algorithm. The Original RSA scheme is as follows:

A. Key Generation Algorithm

- 1) Randomly and secretly choose two large primes: p, q and compute $n = p \cdot q$
- 2) Compute $\phi(n) = (p - 1)(q - 1)$.
- 3) Select Random Integer: e such as $1 < e < n$ and $\gcd(e, \phi) = 1$.
- 4) Compute d such as $e \cdot d \equiv 1 \pmod{\phi(n)}$ and $1 < d < \phi(n)$.
- 5) Public Key: (e, n)

6) Private Key: (d, n) .

B. Encryption Process

- 1) Suppose entity A needs to send message m to entity B (represent m as an integer in the range of $0 < m < n$).
- 2) Entity B should send his public key to entity A.
- 3) Entity A will encrypt m as $c = m^e \bmod n$ and will send c to entity B.

C. Decryption Process

- 1) Entity B will decrypt the received message as $m = c^d \bmod n$.

The most important advantage of RSA is ensuring about the privacy of the private key because this key will not be transmitted or revealed to another user. However, this algorithm has some considerable weaknesses. The main computational costs of the RSA are the modular exponentiations found during the key generation, encryption and decryption process [14]. Moreover, this algorithm has some weaknesses against certain attacks (i.e., Brute force, Mathematical attacks, Timing attacks and Chosen Cipher-text attacks) [15]. To reduce these problems, many algorithms have been designed and introduced based on original RSA. RSA Small- e and Efficient RSA are only two of the most popular algorithms identified for improving the main algorithm.

III. THE PROPOSED SCHEMA

The proposed schema will be introduced in this paper as Two key Generation Encryption (2KGEA) based on the strengths of Efficient RSA and RSA Small- e . In this algorithm, the number of exponents will be increased to 3. The steps of 2KGEA are as follows:

A. Key Generation Algorithm

- 1) Randomly and secretly choose two large primes: p, q and compute $n = p \cdot q$
- 2) Compute $\phi n = p - 1 \cdot q - 1$.
- 3) Compute $\gamma n, h = p h - p_0 p h - p_1 \dots p h - p h - 1 + q h - q_0 q h - q_1 \dots q h - q h - 1$.
- 4) Select Random Integer: r such as $1 < r < n$ and $\gcd r, \phi n = 1$ and $\gcd r, \gamma n = 1$ (r should be a small integer).
- 5) Compute e such as $r \cdot e \equiv 1 \bmod \phi n$ and $1 < e < \phi n$.
- 6) Compute d such as $d \cdot e \equiv 1 \bmod \gamma n$ and $1 < d < \gamma n$.
- 7) Public Key: e, n .
- 8) Private Key: r, d, n .

B. Encryption Process

- 1) Suppose entity A needs to send message m to entity B (represent m as an integer in the range of $0 < M < n$).
- 2) Entity B should send his public key to entity A.
- 3) Entity A will encrypt m as: $c = ((m^e \bmod n)^r \bmod n)$

After that Entity A will send c to entity B.

C. Decryption Process

- 1) Entity B will decrypt the received message as: $m = ((c^r \bmod n)^d \bmod n)$.

HE-RSA has used an additional exponent for increasing the security of the original RSA. In this algorithm, r (as third exponent) has been defined according to the RSA small- e .

This means, r in 2KGEA has the same role of e in RSA Small- e and has been defined as a small integer. As it was described earlier, it is expected that the exponent e in 2KGEA has been the same size as ϕn . Furthermore, exponent d has been computed according to the Efficient RSA algorithm and the value of $\gamma n, h$. nevertheless, in the proposed schema, the encryption process has been carried out by two consecutive steps but with the very same

exponents while the exponents for decrypting the cipher text have been different. The security of proposed algorithm has been affected due to this difference.

IV. SECURITY ANALYSIS

The security analysis of original RSA and 2KGEA has been investigated according to three attack approaches: the Brute Force, Mathematical Attacks, and Timing attacks.

A. Brute Force Attack

All possible combinations to guess the private key have been tried by the attacker during the brute force attack. In the original RSA, the probability of failure against this attack will be decreased considerably by choosing exponents larger than 2048 bits [15] but the combination of 2KGEA makes this algorithm has significant resistance brute force attack even with 1024 bits exponents.

B. Timing Attack

Timing attack is a side channel attack, in which the attacker determines private exponent by calculating the time with exploiting the timing variation of the modular exponentiation [12]. Timing attack in original RSA may be prevented by including a random delay to the exponentiation algorithm or multiplying the cipher-text with a random number [15] while dual encryption in 2KGEA will protect the transferred message from the timing attack and it is not necessary for multiplying the cipher-text.

C. Mathematical Attack

This attack will take place by determining p , q or ϕn and could be prevented by using 2048 bits exponents in original RSA while in the proposed model, by increasing the value of h the chance of successful mathematical attack would be decreased considerably [6]. Using 2 different keys for decryption process, has increased the security of the private exponents significantly and even after losing one private key by an attack the data has still been encrypted and unreachable. This is done while the algorithm has used same basic exponents in the generation process of these different keys.

V. 2KGEA AND CLOUD COMPUTING

The proposed algorithm were suggested for using in cloud computing environments and increasing the reliability of this new technology, but the most challenging issue in using 2KGEA encryption algorithm in cloud servers is time and memory limitations during the encryption and decryption process in servers according to the sharing performances. It is suggested to encrypt the stored data with a symmetric-key algorithm such as AES in cloud servers and after that encrypt the secret key with 2KGEA for sharing actions. This means, only the secret key would be encrypted with this algorithm and the encryption and decryption process will be more efficient and needless to say it would be less time consuming and memory deficient.

VI. CONCLUSION

In this paper, a hybrid asymmetric-key encryption algorithm has been suggested based on RSA Small-e and Efficient RSA according to the security issues in cloud computing environments. In the proposed algorithm, the number of exponents has been increased to three and a dual encryption process has been applied to raise the security level of the algorithm in comparison of original RSA. According the simulation results, the total execution time in 2KGEA was increased up to approximately 40 percent less than the original RSA and this increase may be reasonable and acceptable according to the security level and the efficiency of 2KGEA.

REFERENCES

- [1] *Conf. on Electronics Computer Technology (ICECT)*, Kanyakumari, 2011, pp. 236-239.
- [2] M.A. Alzain, E. Pardede, B. Soh, and J.A. Thom, "Cloud Computing Security: From Single to Multi-Clouds," in *Proc. 45th Hawaii International Conf. on System Science (HICSS)*, Hawaii, 2012, pp.5490-5499.
- [3] U. Somani, K. Lakhani, and M. Mundra, "Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing," in *Proc. 1st International Conf. on Parallel Distributed and Grid Computing (PDGC)*, Solan, 2010, pp. 211-216.
- [4] S. Alshehri, S. P. Radziszowski, and R. K. Raj, "Secure Access for Healthcare Data in the Cloud Using Ciphertext-Policy Attribute-Based Encryption," in *Proc. 28th International Conf. of Data Engineering Workshops (ICDEW)*, Virginia, 2012, pp. 143-146.
- [5] L. H. Ying, S. S. Tzuo, T. W. Guey, and B. S. P. Lin, "Toward Data Confidentiality via Integrating Hybrid Encryption Schemes and Hadoop Distributed File System," in *Proc. 26th International Conf. on Advanced Information Networking and Applications (AINA)*, Fukuoka, 2012, pp. 740-747.

- [6] S. J. Aboud, M. A. Alfayoumi, M. Alfayoumi, and H. Jabbar, "An Efficient RSA Public Key Encryption Scheme," in *Proc. 5th International Conf. Information Technology: New Generations (ITNG)*, Las Vegas, 2008, pp. 127-130.
- [7] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. *J. Cryptol.*, 21(3):350-391, 2008.
- [8] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. Order preserving encryption for numeric data. In *Proceedings of SIGMOD '04*, pages 563-574, 2004.
- [9] Georgios Amanatidis, Alexandra Boldyreva, and Adam O' Neill. New security models and provably-secure schemes for basic query support in outsourced databases.
- [10] D. Coppersmith, M. Franklin, J. Patarin, and M. Reiter, "Low-exponent RSA with Related Message," in *Proc. 15th Annual International Conf. on Theory and Application of Cryptographic Techniques (EUROCRYPT'96)*, Berlin, 1996, pp. 1-9.
- [11] D. Boneh and G. Durfee, "Cryptanalysis of RSA with private key d less than $N^{0.292}$," *IEEE Trans. on Information Theory*, vol. 46, no. 4, pp. 1339-1349, July 2000.
- [12] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, *Cryptology Conference on Advances in Cryptology (CRYPTO'96)*, London, 1996, pp. 104-113.
- [13] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *ACM Trans. On Communications*, vol. 21, pp. 120-126, 1978
- [14] H. M. Sun, M. E. Wu, W. C. Ting, and M. J. Hinek, "Dual RSA and Its Security Analysis," *IEEE Trans. on Information Theory*, vol. 53, no. 8, pp. 2922-2933, August 2007.
- [15] A. Alhasib and A. M. Haque, "A Comparative Study of the Performance and Security Issues of AES and RSA Cryptography," in *Proc. 3rd International Conference on Convergence and Hybrid Information Technology (ICCIT)*, Busan, 2008, pp. 505-510.
- [16] L. Ballard, S. Kamara, and F. Monroe. Achieving efficient conjunctive keyword searches over encrypted data. In *Proc. of ICICS*, 2005.
- [17] Feng Bao, Robert Deng, Xuhua Ding, and Yanjiang Yang. Private query on encrypted data in multi-user settings. In *Proc. of ISPEC*, 2008.
- [18] Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In *Proc. of CRYPTO*, 2007.
- [19] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Incremental cryptography: The case of hashing and signing. In *Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO ' 94*, pages 216-233, London, UK, UK, 1994. Springer-Verlag.
- [20] S. Berreti, A.D. Bimbo, and E. Vicario. Efficient matching and indexing of graph models in content-based retrieval. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2001.