Cluster Comput (2017) 20:1561–1573
DOI 10.1007/s10586-017-0809-4

Secure and efficient online data storage and sharing over cloud environment using probabilistic with homomorphic encryption

N. Jayapandian[1] · **A. M. J. Md. Zubair Rahman**[2]

Received: 11 December 2016 / Revised: 6 February 2017 / Accepted: 22 February 2017 / Published online: 29 March 2017

Abstract Cloud computing is one of the great tasks in the business world nowadays, which provides shared processing resources. In cloud area network, security is the main challenge faced by cloud providers and their customers. The advantage of cloud computing includes reduced cost, re-provisioning of resources etc. The cloud network makes use of standard encryption method to secure documents while storing in online. In this paper, we have depicted two efficient encryption algorithms that meet security demand in cloud. Probabilistic encryption, one of these algorithms may be used to produce randomness of text encryption. With this algorithm, if the same message is encrypted twice it should yield different secret coded texts on both calculations. Another crucial algorithm is homomorphic encryption, is a cryptographic method to define the sample system and to provide a software implementation. In order to maintain quality of service (QoS) and improve customer satisfaction, we are going to propose an efficient algorithm which combines the characteristics of both probabilistic and homomorphic encryption techniques, to provide high level of security. Our proposed scheme will yield better encryption techniques reduce security attacks, increased throughput and improve the QoS.

Keywords Cloud computing · Probabilistic · Homomorphic · Encryption · Data storage

✉ N. Jayapandian
jayapandiann@zoho.com

[1] Department of Computer Science & Engineering, Knowledge Institute of Technology, Salem, India

[2] Al-Ameen Engineering College, Erode, India

1 Introduction

Cloud computing is a model for permitting pervasive, suitable, on-demand network access to a common typical pool of configurable computing assets (e.g., servers, applications, etc.). For defining the best way of encryption we propose this paper for betterment and quality of service (QoS) in cloud security. In modern cloud encryption, NULL is a block cipher created for analysis of loss in antiquity there is already found of rumors that the National Security Agency suppressed publication of this algorithm, there is no correct proof for such action on their part null encryption other efficient way on selecting NONE or null cipher is choosing not to use crypt in a system where different type of encryption options are offers, such as for testing and debugging, or authentication-only communication. Thus the data is remaining same after encryption. In mathematics such a function is known as the identity function. NULL is defined mathematically by the use of the Identity function applied to a block of data N such that:

$$\mathrm{NULL}(N) = I(N) = N, \tag{1}$$

where I(N) is the encrypted text and N is the original text so it remain the original after the encryption. Here we can name null encryption as the null ciphers. Null ciphers provide a way of concealing a message within a larger content of plain text without the need for a complicated crypt system, and they can also be more secure depends on the memory size of the text compared to the amount of plain text. The secret text is encoded as every character of a plain text, but there's enormous error and attacks of discovered systems. To make more efficient we minimize the chances of that revealing, various method should be used to determine the way of secret characters, ideally that terms gives the appearance of ran-

domness. Though NULL encryption technique is faster than other ordinary encryption, the implementations of the base methods are available only for commonly used hardware and OS platforms. The NULL encryption algorithm offers neither confidentiality nor does it offer any security service [1]. It is simply a convenient way to determine integrity of the other use of applying encryption within encapsulating security payload (ESP). When it comes under randomness we shall use probabilistic encryption, a probabilistic encryption scheme is polynomially secure and it has the effectiveness of deterministic structures, encryption schemes were the topmost area of concern in cryptography [2]. They compact with providing means to allow private statement over an apprehensive channel. A sender sends information to the receiver via an insecured channel that is a channel which may be nominated by an opponent. The data and facts that are need to be intersected, which we call it as a actual text must be put into an unusual or distinct code (encrypt) to a cipher text (encoded information). Only the authorized person must be given the idea to convert the secret code (i.e., cipher text) back to the actual original message (decrypt) [3], while the unauthorized person cannot have permission to access this message. The authorized person will be given a key at his disposal, which enables him to recover the actual message. Probabilistic encryption is a modern design technique for encryption where a message is encrypted into one of many conceivable cipher texts, in such a way that it is also provably as tough to attain fractional evidence about the message from the cipher text [4], as it is to solve some tough problematic function. In early approaches to encryption, even though it was not always known whether one could acquire such fractional information, neither was it proved that also one could not do so. The structure had considerable message development due to the bit-by-bit encryption of the message which overall makes the system not practical. In this we develop a practical encryption system which combines both the security measures schemes and the efficiency of the deterministic schemes by using one-way function with a base. For high efficiency of the process we undergo homomorphic encryption, the basic concept to encrypt the data before sending to the cloud provider. The client needs to provide the private key to decrypt the data from the server before calculation execute and the confidential data to encrypt without decrypt [5]. We consider homomorphic encryption technique with esteem of Boolean circuits comprising of gates for performing calculations like addition and subtraction [6]. The modern aspect of cryptography has the criteria to secure the data storage to respond the client request and the size of the data that is encrypted and get stored in the cloud server [7]. If the encryption scheme is homomorphic, it can still perform some meaningful computation to the data when encrypted. In the real world applications like medical, financial domains the fully homomorphic scheme is used. But somewhat homomorphic is better can be faster and more compact than other schemes. Homomorphic encryption is the primary application in the field of cloud. The recognition of cloud computing is reliable with unblemished technological and financial trends [8]. The residue number system (RNS) that creates multi shares and the operations on these data shares are homomorphic. There are two properties in RNS to design the functions. The growth of the end-users is increasing in cloud service provider (CSP) and the impact of cloud services in business sector is tremendous.

In Sect. 1 the introduction deals with the general description about the cloud and the proposed algorithm. In Sect. 2 the related work explains the trap door function which is widely used in the cryptography, symmetric key and asymmetric key encryption should be discussed. In Sect. 3 the existing system deals with the present techniques used for encryption and decryption process. In Sect. 4 explains working of the proposed algorithm of probabilistic with homomorphic algorithm which gives the effective outcome as compare to the existing methodologies. Section 5 explains the step by step mathematical relation of the hybrid algorithm. Section 6 deals the experimental result and analysis of the hybrid algorithm. Section 7 which gives conclusion and future works of the proposed algorithm.

2 Related work

A probabilistic algorithm is an algorithm, which result to approach the results obtained depend upon the probability. These algorithms also are typically known as randomized algorithms. In some applications the utilization of probabilistic algorithms is natural, example simulating the behavior of some existing or planned system over time. During this case the result naturally a stochastic. In some cases the matter to be resolved is deterministic however is be changed into a stochastic one and resolved by applying a probabilistic algorithm example is numerical integration and optimization. For these numerical applications the result obtained is usually approximate, however its expected accuracy improves because the time offered to use the algorithm will increase. The techniques of applying probabilistic algorithms to numerical problems were originally known as Monte Carlo method. There also a variety of discrete problems for which only the accurate result's acceptable example is sorting and searching, wherever the introduction of randomness influences only on the benefit and efficiency finding the answer. For some issues where trivial in depth search isn't possible. Probabilistic algorithms will be applied giving a result that's correct with a probability but that is smaller than one example is primarily testing and string equality testing. The probability of failure will be created arbitrary tiny by frequent applications of the algorithm.

Table 1 Comparison of probabilistic and homomorphic encryption

Parameters	Probabilistic encryption	Homomorphic encryption	Probabilistic with homomorphic encryption
Algorithm type	Symmetric	Symmetric	Asymmetric
Key length (bits)	256	256	512
Security	Secure	Secure	High secure
Power consumption	Medium	Medium	Medium
Speed	Fast	Moderate	Very fast
Memory usage	High	High	Medium
Encryption throughput	High	Medium	Very high
Decryption throughput	High	Medium	Very high
Implementation	Complex	Simple	Simple

Homomorphic encryption could be a structure of encryption that enables computations to be accepted out on cipher text, so generating an encrypted result that, once decrypted, matches the results of operations performed on the plaintext. This is generally a needed feature in modern communication system architectures. Homomorphic encryption would enable the chaining together with various services does not includes the exposing the data to every of these services. For instance, a series of different from various services from different company may calculate the tax, currency rate of exchange and shipping, on a contract action without exposing the unencrypted data to every of these services [9]. Homomorphic encryption schemes are malleable by design. This permits their use in cloud computing environment for making the confidentiality processed data. Additionally the homomorphic property of a variety of cryptosystems may of be able to create many secure system, for example secure voting systems, collision-resistant hash functions, private information retrieval schemes and many [10]. There are many partly homomorphic crypto-systems, and as well as variety of fully homomorphic crypto-systems. Although a crypto-system that is coincidence malleable may be subject to attacks on this basis, if treated safely carefully homomorphism can also be used to perform computations securely (Table 1).

The probabilistic and homomorphic encryption are symmetric key type, when both the algorithm is being combined, the algorithm type is asymmetric. The key length for probabilistic and homomorphic is 256 bits, when these two algorithm are combined the key length will be doubled that is 512 bits. The probabilistic algorithm has high security level when compare to both homomorphic and the algorithm which we proposed. The probabilistic with homomorphic algorithm has less power consumption as compare to probabilistic and homomorphic algorithm, but the probabilistic algorithm's speed is fast when compare to homomorphic algorithm, it works at a moderate speed. The combined algorithm which we proposed is very fast when compare to probabilistic algorithm. The probabilistic and homomorphic algorithm consumes high memory usage whereas the combined algorithm consumes less memory usage. The throughputs of encryption of the combined algorithm are very high as compare to both the algorithms. Similarly the decryption throughputs of the proposed algorithm are higher as compare to both algorithms. The implementation of probabilistic algorithm is little complex whereas the homomorphic and the proposed algorithm is very simple.

The main advantage of probabilistic with homomorphic algorithm is power consumption and higher security. The memory usage in both the algorithms is very high as compare to the combined algorithm of probabilistic and homomorphic algorithms. The encryption and decryption throughput is very high in the combined algorithm. The implementation of this hybrid algorithm is very simple as compare to other methods.

The major public-key Data Encryption Structures are deterministic algorithms and is based on trapdoor functions. Trapdoor functions are widely used in cryptography [11], the two main disadvantages of encryption systems based on trapdoor functions are, inverting may be tranquil for actual texts only for some distinct form, similar to encrypting the messages 1 and 0 to themselves and it could be easy to estimate at least fractional evidence of the actual text. Additionally, for a deterministic structure it is easy to perceive if a message is sent twice. In this section, we discuss about some of the trapdoor functions hired in data encryption schemes. So far it's not known whether these functions are actually one way, but research has shown that there is no efficient inverting algorithm for any of them unless one has fractional evidence or the trapdoor (Fig. 1).

The function f takes inputs as two prime numbers A and B in binary notation and gives their product. This function can be calculated in $O(n^2)$ time where n is the overall length of the inputs. Inverting this function needs discovering the factors of integer N that has been given. The finest factoring algorithms known sequence in $O(2^{(\log N)1/3(\log\log N)2/3})$ time, and it is the only pseudo-polynomial of log N, the total number of

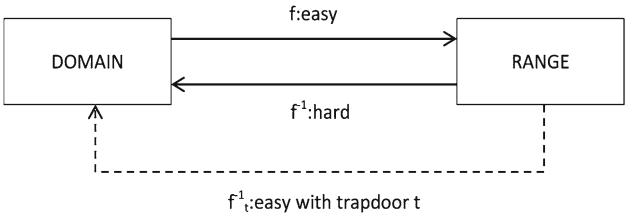


Fig. 1 Domain and range of trapdoor system

bits required to denote N. This function can be widespread by permitting p and q to range over a appropriate set of semi-primes. Note that f is not one-way for subjective p, q > 1, since the product has 2 as a factor with probability 3/4. Probabilistic public-key data encryption scheme used the predicate called "is quadratic remainder modulo compound n" [12]. In this scheme, each and every message had lot of conceivable encodings and every bit of a message is encrypted individualistically. Due to this last property, this scheme is not practical. If k is the security constraint then each bit of the information is encoded separately by a k-bit extended sequence and even inferior, resulting in at least a k-bit data expansion factor. The main aspect of cloud computing is to encrypt the data before sending it to the cloud provider. At every operation data has to decrypt when needed. The client has to provide the private key to the server for the data to decrypt before executing the calculations required, which may affect the privacy of data stored in cloud computing has different names like "server hosting" and "outsourcing". It has different virtualization techniques and adding a new server for the many new applications at risk. The homomorphism encryption systems perform the operation of encrypting the data without decrypting (knowing the private key) for the secret key client in the only holder.

By convention, the cloud computing storage of personal or the professional information which has to secure and it realized the concern structure (calculation) the enterprises uses the virtualization on their cloud platform on the same server and the providers such as IBM, Google and Amazon can consist the virtualized storage and the treatment space that belongs to intervene and protect the data.

RSA cryptosystem realize the properties of the multiplicative homomorphism encryption and it has still lake of security because if two ciphers C_1 and C_2 and corresponding messages as $n_1, n_2, \ldots$

$$C_1 = n_1^e \bmod m, \tag{2}$$

$$C_2 = n_2^e \bmod m. \tag{3}$$

Then the client sends the pair of (C_1 C_2) to the server to perform the calculations and to encrypt the result to the client ($C_1 \times C_2$). The main application of RSA multiplicative process of encryption in homomorphism on two messages n_1 n_2 and their ciphers C_1 C_2, respectively, using the RSA encryption.

$$N_1 = 579,638 \quad C_1 = 00\,05\,00\,07\,00\,09\,00\,06\,00\,03\,00\,08,$$
$$N_2 = 365,892 \quad C_2 = 00\,03\,00\,06\,00\,05\,00\,08\,00\,09\,00\,02.$$

Symmetric-key algorithms are cryptography that uses equivalent cryptographic keys for each encryption of plaintext and decryption of ciphertext. The keys could also be identical or there could also be an easy transformation to travel between the two keys. The keys, in patience, represent a shared secret between two and a lot of parties which will be wont to maintain a private information link [13]. This demand that each parties have access to the secrete keys one amongst the most drawbacks of symmetric key encryption, as compared to public-key public key encryption [14]. In general symmetric key encryption is two types first one is stream cipher and second one is block cipher [15].

The advanced encryption system (AES), additionally known by its original name Rijndael, may be a specification for the encryption of electronic data established by the US National Institute of Standards and Technology (NIST) in 2001 [16]. AES may be a subset of the Rijndael cipher developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, who are submitted a proposal to NIST at the time the AES selection process [17]. Rijndael may be a family of ciphers with completely different key and block sizes. For AES, NIST selected three members of the Rijndael family, every one with a block size of 128 bits, however three completely different key lengths: 128, 192 and 256 bits. AES has been adopted by the US Government and is currently used worldwide. It supersedes data encryption standard (DES), which was revealed in 1977 [18]. The algorithm explained by AES could be a symmetric-key algorithm, which means same key is used for each encrypting and decrypting the data. High speed and low RAM necessities were criteria of the AES selection process, because the chosen algorithm, AES performed well on a large variety of hardware, from 8-bit good cards to superior computers.

DES is fixed-length and symmetric-key algorithm. The block size is 64 bits. DES also uses a key to customize the transformation. Hence the decryption will supposedly only be performed by those that may know the actual key used for the encryption. The key apparently consists 64 bits; but, only 56 of those are literally employed by the algorithm. Eight bits are used for checking parity, and are henceforth discarded. Hence the effective key length is 56 bits. Like other block ciphers, DES by itself is not a secure suggests that of encryption however should instead be employed in a mode of operation. FIPS-81 specifies many modes to be used with DES [19]. Additional comments on the usage of DES are contained in FIPS-74. Decryption uses similar structure

as encryption however with the keys employed in reverse order.

Asymmetric encryption, is also called public-key encryption, utilizes a couple of keys a public key and a non-public key. If you encrypt the data with the general public key, only the holder of the corresponding private key will decrypt the data, therefore making certain confidentiality. The first realistic algorithm for asymmetric encryption was developed by Diffie and Hellman in 1976. Later on, RSA became the most usually deployed asymmetric encryption algorithm. Many secure on-line dealings systems believe asymmetric encryption to begin a secure channel. SSL, for instance, may be a protocol that utilizes asymmetric encryption to produce communication security on the web. Asymmetric encryption algorithms generally involve exponential operations; they're not light-weight in terms of performance. For that reason, asymmetric algorithms are usually used to secure key exchanges instead of using for bulk encryption.

RSA is an algorithm employed by new computers to encrypt and decrypt messages. It's an asymmetric cryptographic algorithm. Asymmetric means there are two totally different keys. This is often known as public key cryptography, as a result of one amongst them will be given to everybody. The other key should be private. It is based on the truth that capturing the factors of an integer is difficult (the factoring problem). RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman, the persons who described it in 1978. A user of RSA creates and to publishes the creations of two massive prime numbers, also with the auxiliary value, as their public key. The prime factors should be kept secret. Anyone will use the general public key to encrypt a message, however with presently published methods, if the general public key is massive enough, only somebody with information of the prime factors will feasibly decrypt the message [20].

3 Existing system

As a substantial study area for system security, data access control has been developed. Various practices have been established to implement fine-grained access control efficiently, which permits flexibility in stipulating discrepancy access rights of individual users. Also it contains NULL algorithm which is not much confidential as well as not secure, this algorithm does not offer security service also. It is just an appropriate way to epitomize the voluntary use of applying encryption within ESP. As this algorithm is of less confidentiality, ESP has been enhanced. So without confidentiality ESP can be used to provide authentication and integrity. A probabilistic algorithm is a type of algorithm which contains an additional command RANDOM that returns "1" or "0", both with a probability series of $\frac{1}{2}$. Many encryption and decryption schemes are there, such as a private key versus public key, Probabilistic Public-Key Encryption, Semantic Security, the Quadratic Residuosity Problem and so on. Initially the encryption schemes had one single key for encryption and decryption, so that it has to be made confidentially secured. The private key is also called as symmetric encryption scheme. Before using this encryption scheme, once the key is given it must be replaced securely, hence the private-key encryption is a "method of encompassing a private network over a specified time". After this, a new indication of decryption called public key encryption has been enhanced, which is also called as asymmetric encryption scheme. Another security algorithm called homomorphic encryption on cloud storage holds a recent work done in the context of cloud security. The analysis of data security consists of data with a life cycle of seven phases such as generation, use, transfer, share, storage, archival, and destruction. In data storage, key management is considered as most essential and important point. Again in data destruction phase it is important to ensure that data is securely erased using methods that makes it unrecoverable. Otherwise an adversary can take advantage of the physical characteristics of the storage medium to access sensitive information. It addresses security issues related to single and multi-cloud models. Authors have evaluated the security of single clouds based on three factors, namely data reliability, data interloping, and service availability. When the total number of users increases, then efficiency of the work automatically increases. The performance of the system also will remain unchanged [21]. Data reliability is important since the data can get corrupted during transmission or transfer between data source and the cloud. Another security factor is the data interloping, in which the adversary gains access to a cloud service through stolen passwords and then can cause damage to the services being used by the genuine users. Service availability is also another factor to be considered.

The existing system which is the probabilistic and homomorphic encryption which does not have the high security level as it is the algorithm type is symmetric for both the probabilistic and homomorphic algorithms, it shown in Fig. 2. When we combine both the algorithms that is homomorphic with probabilistic algorithm the security level is increased which have the type of algorithm in the asymmetric algorithm. The speed of the transformation is fast and moderate in probabilistic and homomorphic algorithm whereas the hybrid algorithm has very fast speed.

4 Proposed algorithm for probabilistic with homomorphic encryption

Encryption has been largely studied in the framework of cryptography. Among those works, improving the efficiency and security definition reinforcements are mostly focused. The

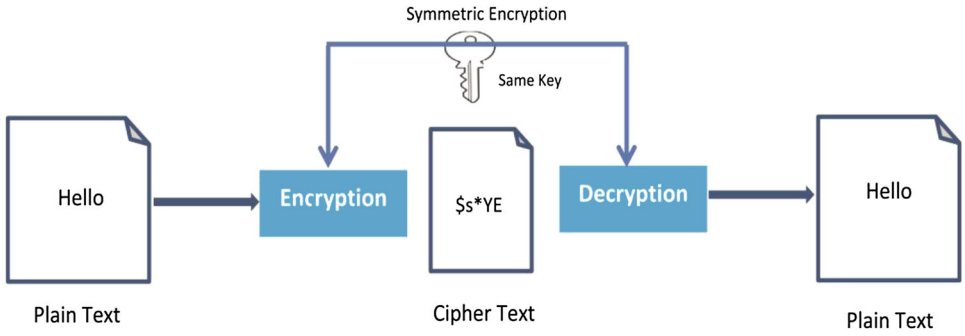


Fig. 2 Symmetric encryption architecture

major construction is one in which each and every word in the document is encrypted autonomously under a special two-layered encryption construction. So the proposal for filters to construct indexes for data files has emerged. To every file, all unique words are built up in a filter that contains trapdoors which stored on the server. To survey a word, the consumer generates the quest demand by computing the trapdoor of the word and sends it to the server. After receiving the request, the server examines if any filter comprises the trapdoor of the request word and proceeds the consistent file identifiers. To achieve more efficient encryption, the "index" approaches would help to find single encrypted hash table directory which is built for the whole file collection. In the index table, all the entry consists of the trapdoor of a keyword and an encrypted set of file identifiers whose consistent data files contain the keyword. As a balancing approach, a public-key based encryption scheme, with an equivalent to that of probabilistic has been reinforced. In their construction, any person with the public key can inscribe to the data stored on the server but only the users who are authorized with the private key can search. As an attempt to improve query centers, subcategory query and array of query over encrypted data, have also been proposed in. If all documents such as personal, health, financial, etc., warehoused in the cloud were encrypted, that would efficiently resolve problems. However, if the data is not decrypted first then the user would be unable to influence the power of the cloud to carry out computation. The cloud benefactor thus has to decrypt the documents first (invalidating the issue of secrecy and privacy), complete the computation then send the outcome to the user. What if the user could carry out any random computation on the hosted data without the cloud benefactor learning about the user's data computation is done on encrypted data without prior decryption.

Probabilistic encryption is the use of arbitrariness in an encryption algorithm, so that when the same message is encrypted many times, generally it will yield different types of cipher texts. The "probabilistic encryption" is a term generally used in encryption algorithms with reference to public key, there are various symmetric key encryption algorithms which achieve a comparable property, that is, to hide even incomplete information about the original text, the probabilistic encryption algorithm can be used. When using public key cryptography probabilistic encryption is mainly important. Suppose that the opponent observes a cipher text, and suspects that the actual text is either "YES" or "NO", or has a forecast that the actual text might be "SOME OR MANY".

This is the promise of probabilistic encryption schemes which allow the transformation of secret texts S(m) of message m, to secret texts S(f(m)) of a calculation/function of message m, without disclosing the message. This idea was, referred to probabilistic privacy schemes and multiplicative probabilistic (we could compute a secret code text which is the product of plaintexts) and above the next 30 years, researchers came up with partially probabilistic cryptosystems, a survey of probabilistic encryption schemes that can be found in. Though probabilistic encryption is an effective method, in one or the other way data hacking takes place to improve the efficiency and QoS, we enhance homomorphic encryption, in the encryption of homomorphic, if: from encry(a) and encry(b) is possible to compute encry(g(a, b)), where f can be: +, x without the usage of private key. Among the homomorphic, the operations are allowed to access the raw data. The ideal of homomorphic encryption has been arising when it is around 30 years and the significant breakthrough in 2009. There is fully homomorphic encryption that exists today because of the limitations that is related to the complexity of computations and they are not considered to be practical in the usage of today's applications. Our proposal is to encrypt the data before which we are sending to the server to execute the calculations that are provided. In homomorphic encryption, the improvement in the complexity that compares to the response time and with the length of the public key. The advantage of computing that includes resources of re-provisioning reduces cost and easy maintenance and thereby increased profit. The fully homomorphic encryption is an important factor in cloud security. Then we analyze the performance of the existing homomorphic encryption cryptosystems and outsource the calculations on the confidential data to the server, with the secret key to decrypt the calculation results (Fig. 3).

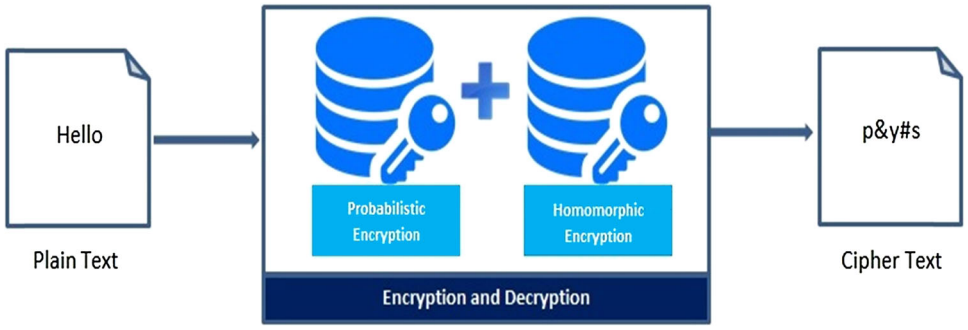


Fig. 3 Probabilistic with homomorphic encryption

Generally when we undergo probabilistic algorithm, we achieve successful encryption. As in this algorithm we can encrypt any test at any order because there is no restriction for randomness. When we encrypt different tests we achieve several cipher texts. In every encryption algorithm, the main aim will be encoding making the original message into a secret code in order to define high security. In this case, homomorphic encryption also deals with converting actual text into cipher text. The performance of this encryption would have high security, as the client needs their data to be secures and it should not get hacked. The personal and computational information that are stored in cloud should be safe and secure. These two encryption algorithms probabilistic and homomorphic separately gives security for data encryption, as the level of hacking is less. So, when we combine these algorithms and encrypt the test, it will be strongly encrypted into a cipher text and the possibility for hacking is very low. Because of this high level data storage, security, confidentiality, accuracy, availability everything will be maintained. Combination of these two algorithms will yield a good result for encryption in cloud storage scheme.

5 Mathematical relation

Algorithm 1 [Key Generation K**]:**

1. Select any two large random primes a and b, a≠b.
2. Set n ←ab.
3. Select a pseudo square P ∈ Qn (i.e. P is quadratic non-residue and $(\frac{P}{n}) = 1$).
4. The public key is (n, P), the private key is (a, b).

Algorithm 2 [Encryption E]:

Let text t be a binary string t = t1, t2 . . .t_l ,
let (n, p) be the public key.
1. For i = 1. . .l do:
(a) Select a ∈ q_n at random.
(b) If mi = 0, set ri ←a^2 mod n; otherwise set ri ← pa^2 mod n.
2. The Encrypted text is e = (e1, e2, . . . , e_l).

Algorithm 3 [Decryption D]:

Let e = (e1, e2, . . . , e_l) be a Encrypted text and (a, b) be the private key.
1. For i = 1. . .l do:
(a) Compute ci = $(\frac{ei}{p})$ using Proposition 1.
(b) If ci = 1, set ti ←0; otherwise set ti ←1.
2. The decrypted text is t = (t1,t2, . . . ,t_l).

Remark 1 Key generation the pseudo square P required in *Step 3* can be found by *a* probabilistic algorithm that picks *p* at random until

$$\left(\frac{y}{a}\right) = \left(\frac{y}{b}\right) = -1.$$

Encryption if ti = 0, then it is encrypted to a random quadratic residue modulo n, while for ti = 1 a random pseudo square is chosen. In fact, conferring to Lemma 4 and multiplicativity of the Jacobi symbol, a pseudo square times a quadratic residue yields a pseudo square.

Decryption knowing the factors of n, it is easy to decide whether a ci is a quadratic residue or not: $\left(\frac{\mathrm{ci}}{\mathrm{p}}\right) = 1$ if and only if ci is a quadratic residue, which is the case if and only if ti = 0.

Remark 2 [**Security of the scheme**]: assuming the hardness of QRP (Definition 5), the Goldwasser–Micali encryption scheme is semantically secure: for a € Q_n^* is picked at random, a^2 is a random quadratic residue and pa^2 is a random pseudo square modulo n. So, in order to decrypt a single bit of the coded text, an attacker would have to solve the quadratic residuosity problem. For a detailed proof based on mathematical definitions. The Goldwasser–Micali cryptosystem was the first system based upon the concept of probabilistic encryption and furthermore the first systems proven to be semantically protected. It is not a feasible system since in general, one plaintext-bit is expanded into n bits of cipher text.

In the algebraic homomorphism, the structure and the isolation part of the processing model and the dependency of the execution engine. Homomorphic filter algorithm which is used for the remote sensing. Our implementation in the circuit that comprises for an encrypted program in the runtime environment to generate an assembler in the machine code.

In the first case, the homogeneous give as with $S_i = 45.5$ for all K. In the second case, we allow for heterogeneity with S_i = j for all K. In the later case, as with the full-scale attack if only S > 50. In some business applications, the market size and the privacy inhibitors for the adoption of which it protects the data. The utilization of homomorphic authenticators are to reduce the arbitrarily a wide communication that over headed with the divided blocks as P_i (i=1,…,n) and each block of P_i for the cloud server as the data owner to submit challenges chale = {(I, q_i)} for the random blocks and with the linear combinations as $\mu = \sum_i q_i \cdot p_i$ where, the large fraction of block size $|q_i|$ where, $|q_i| \gg \log(n)$.

The aggregate authenticator of $\sigma = \pi_{i\Omega i}$ and computed as $\{p_i, \sigma_i, Vi\}$. Let G be denote the multiplicative set of which is under the operation of (x, y £ R -> x, y £ R) and does not contain zero. The elements of $e_1, e_2, \ldots, e_1 | \geq |$ y G are the multiplicative generators in the e_i, i = 1, 2,. . .,l. The graph which is undirected with the graph- theoretical as V(R) X G(R) for the format of $v_i G v_{i+1}$ for i = 1,. . .,G-1 is the pare of the graph. If x £ G, then (x) £ Q and [x] £ L.

It the co-ordinate of the points and the lines are

$$[q] = \left(q_1, q_{11}, q_{12}, q_{21}, q_{22}, q_{23}, \ldots, q_{ii}, q_{ii}, q_i, i+1, \ldots\right), \tag{4}$$

$$[m] = \left(m_1, m_{11}, m_{12}, m_{21}, m_{22}, m_{23}, \ldots, m_{ii}, m_{ii}, m_i, i+1, \ldots\right). \tag{5}$$

Here the encryption process generate the group of key elements, each group will have a separate nodes and mid-values. Here q denotes points and m demotes lines. qii, qi, i+1 are the incremental values of the node. mii, mi, i+1 denotes the increase in the number of lines. This system is used to randomly encrypt the data. Which will automatically increase the security level.

$$\text{The subset of } q_r = q_r(\mu) \sum_{i=0}^{r} \left(\mu_{ii}, \mu_{r-i} - \mu_i, i+1\mu_{r-i}, r-i-1\right) \quad \text{and } d = d(u) = \left(d_2, d_{3,\ldots}, d_t\right). \tag{6}$$

The public key algorithm for the t for the linear transformation T: x-> Bx where B is the sparse matrix of the condition as $B \neq 0$ as $\mathrm{Map}_T E_n, \alpha, \beta, T^{-1}$ as the multiplicative public rule.

$$x_1 -> g_1\left(x_1, x_[2], \ldots..x_n\right), \tag{7}$$

$$x_2 -> g_2\left(x_1, x_2, \ldots..x_n\right). \tag{8}$$

$x_n -> g_n$ (x_1, x_2,…,x_n) for the linear transformation of the part (β_1, α_1, α_2, β_2,…,β_k, α_k) as secret. We consider Diffie-Hellman algorithm for D(fn) for the key exchange in the case of group. Let ALn(Fq) be the group of affine transformation of the vector space F n q, i.e., maps τA, B: xe → xAe + B, where xe = (x1, x2,…,xn), C = (C1, C2,…,Cn) and A is invertible sparse matrix with A 5 = 0. Let h k n be the new public rule obtained via k iterations of hn = Fn, α, β = LG, n, β1 PG, n,α1 LG, n, β3 PG, n, α2,…,LG, n, βk PG, n, αk. Correspondents Alice and Bob have different information for making computation. Element hn as above, affine transformation τ ∈ AGLn(K). So she obtains the base b = τhk n τ − 1 and sends it in the form of polynomial standard map to Bob. So, Alice chooses rather large number nA computes cA = b nA and send. On his turn Bob chooses his own key nC and computes cB = b nC . He and Alice get the collision map as cA nA and cB nB respectively. Notice that the position of adversary is similar to Bob's position. The need to solve one of the equations b x = cB or b x = cA. The algorithm is implemented in the case of finite fields and rings Zn for family of groups C(fn).

Table 2 Security attack comparison

Data sizes (mb)	Null (%)	Probabilistic (%)	Homomorphic (%)	Probabilistic with homomorphic (%)
001–1000	90	65	30	25
1001–2000	85	60	20	16
2001–3000	92	71	22	15
3001–4000	87	60	24	21

For notation, let ri + pj + 1 = a − bi, so R = R1,…,Rp2. For a code word u = (P1,…,Pn) ∈ Rn and P_N ∈ R, we define the counting function $K_I(u) := \#\{j: uj = Ui\}$. The complete weight enumerator of the D code P is the polynomial. we C (K1, K2,…,Kp2) = X u∈D z n1(u_1) 1 z n2(u_2) 2,…,z np1 (u_N) p2.

The operation for the encrypted data, that render the decision branches in the circuit. The private cloud has the high degree of transparency and control security policy standard in the established enablers and regulatory.

Definition 1 (*Probabilistic with Homomorphic Public-Key Encryption Scheme*) A probabilistic public-key bit-encryption scheme (X, ε, D) with Data security factor m consists of:

- X, the key: a probabilistic algorithm that on input m outputs a pair (a, b), where a is the public key and bis the private key.
- ε, the encryption scheme, with three inputs: the public key a, the plaintext bit s ε {0, 1}, and a random string l of length p(n) for some polynomial p(·). We will write ε_a(s, l).
- Homomorphic scheme: in this function cyclic group G of order q with generator g, if the public key is (G, q, g, j), where $j = g^k$, and k is the secret key, then the encryption of data S is $\varepsilon_a(s, l) = (g^r, s \cdot j^r\ l)$.

$$\begin{aligned} &\varepsilon_a(s_1, l) \cdot \varepsilon_a(s_2, l) = \left(g^{r1},\ s_1 \cdot j^{r1} l\right)\left(g^{r2},\ s_2 \cdot j^{r2} l\right) \\ &= \left(g^{r1+r2},\ (s_1 \cdot s_2)\ j^{r1+r2} \cdot l\right) \\ &= \varepsilon_a(s_1 \cdot s_2). \end{aligned} \tag{9}$$

- D, the decryption scheme, with two inputs: the private key b and the ciphertext c. we will write $D_b(c)$.

$$D_b(\varepsilon_a(s, l)) = S. \tag{10}$$

Equations 1 and 2 combination of probabilistic and homomorphic encryption scheme which provides better result compare to individual performance.

6 Experimental result and analysis

The main objective of our proposed scheme is to encrypt the data securely and increase customer satisfaction by developing high level storage. This factor helps to reduce hacker's attack on the stored data. For this purpose, a virtual function that satisfies security attacks, time complexity, throughput and key analysis are tested with the algorithms. After that the efficiency of the system is calculated and the best algorithm is chosen for encryption of data in the storage. When it is encrypted only the CSP can access the data for decryption.

6.1 Security attacks

Security in the sense describes the safeguarding of data that are stored in the cloud storage unit, though there are many encrypting algorithms used for minimizing hackers attack, here is the probabilistic with homomorphic algorithm which is best for data security and attacks are less when compared to Null encryption, probabilistic and homomorphic algorithm (Fig. 4, Table 2).

When comparing with the data that is defined, the security attack on Null encryption is high as large number of hacking has occurred in this scenario. The hacking level when we encrypt with probabilistic algorithm is less when compared with Null, and in homomorphic encryption the security attack is little bit low, combination of probabilistic and homomorphic encryption attacker ratio is very less compare to other algorithms. For instance when we transmit 001–100 mb of data 90% hacking occur in Null encryption 65% hacking occur in probabilistic, 30% occur in homomorphic and only 30% occur in probabilistic with homomorphic encryption.

6.2 Time complexity

Time complexity of the algorithm is calculated based on its execution time, so when transmitting 001–100 mb of data the Null encryption takes more time to execute, the probabilistic algorithm some time less than null and finally probabilistic and homomorphic encryption executes faster than all the algorithms (Fig. 5; Table 3).

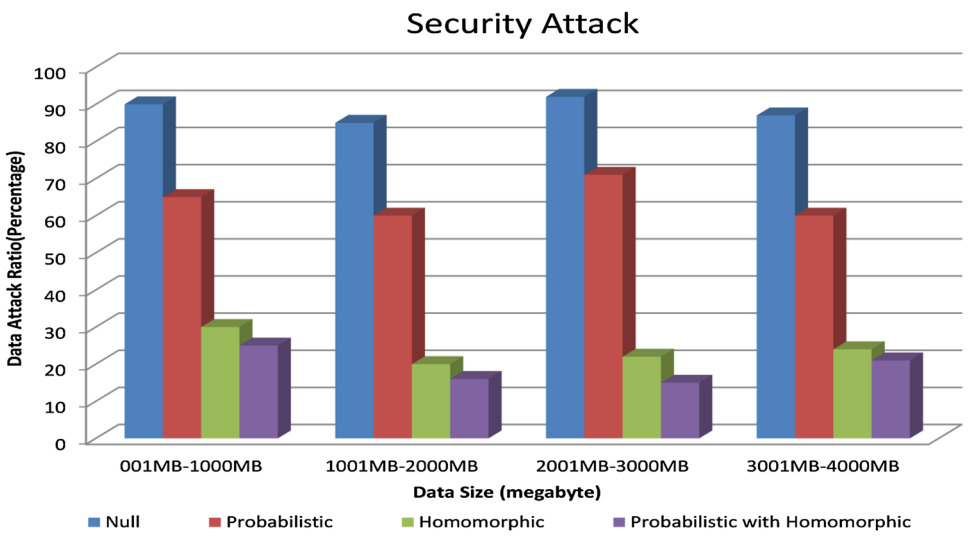


Fig. 4 Comparative of security attacks

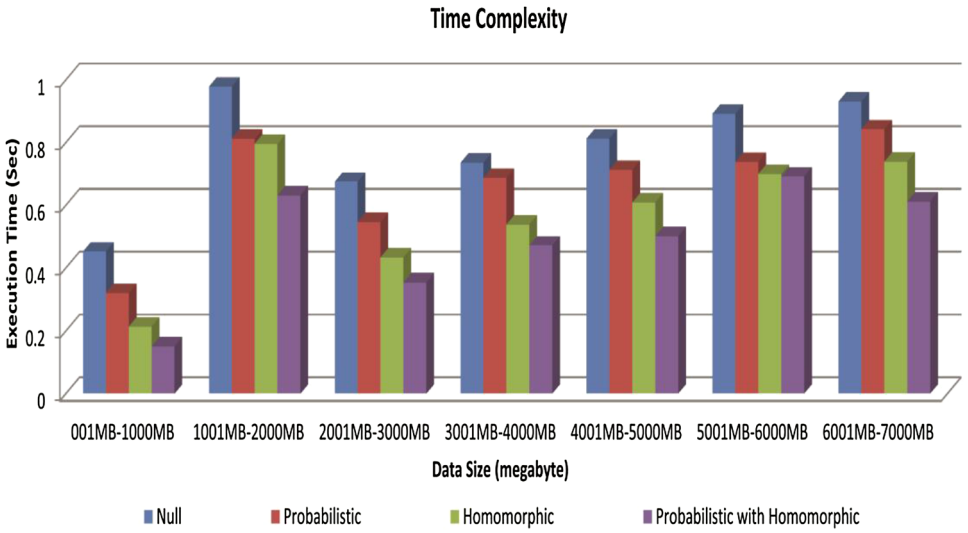


Fig. 5 Execution time comparison

Table 3 Time complexity analysis

Data sizes (mb)	Null (s)	Probabilistic (s)	Homomorphic (s)	Probabilistic with homomorphic (s)
001–1000	0.4527	0.3214	0.2142	0.1512
1001–2000	0.9786	0.8126	0.7962	0.6321
2001–3000	0.6754	0.5467	0.4342	0.3548
3001–4000	0.7364	0.6874	0.5389	0.4720
4001–5000	0.8132	0.7129	0.6092	0.5013
5001–6000	0.8923	0.7382	0.7001	0.6921
6001–7000	0.9313	0.8427	0.7391	0.6123

6.3 Throughput

In the data encryption system, the quantity of the particular data is encrypted in a given amount of time is known to be as the throughput. The throughput is measured in seconds. Consider the above table consists of data size, throughputs of homomorphic, probabilistic and also the hybrid algorithm. In the Table 4 data set 1, the Null encryption algorithm encrypt the data at the rate of 2488.707562 MB/s. Probabilistic and homomorphic algorithm encrypt the data at the rate of 2824.541095 MB/s. The data size is 854 MB which have the null encryption of 343.15 ms, the data is encrypted with the probabilistic algorithm to give the throughput of 326.26 ms and the throughput given by the homomorphic algorithm is

Fig. 6 Throughput analysis

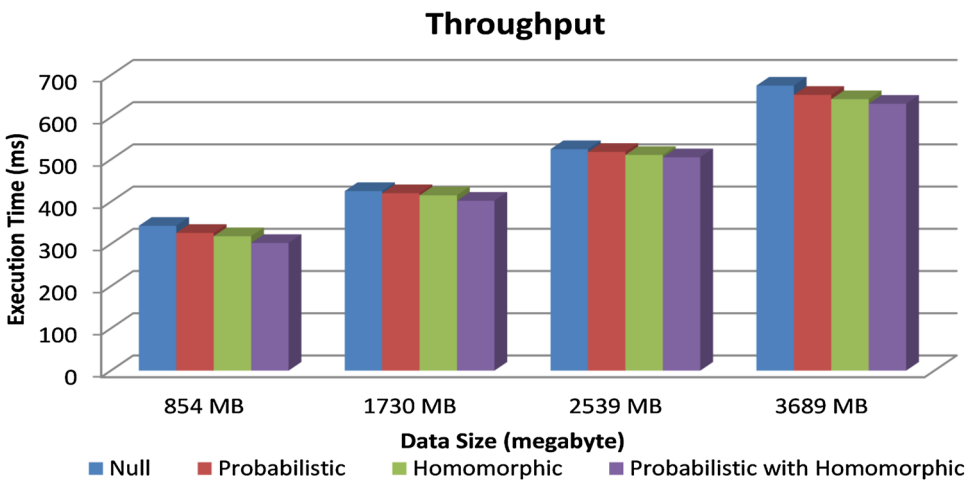
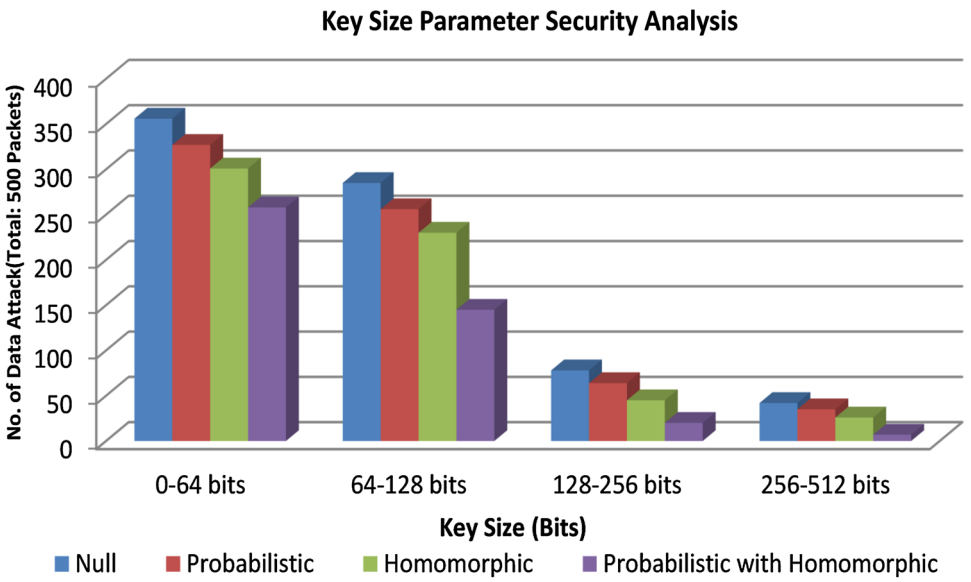


Table 4 Time comparison of throughput

Data sizes (MB)	Null (ms)	Probabilistic (ms)	Homomorphic (ms)	Probabilistic with homomorphic (ms)
854	343.15	326.26	318.46	302.35
1730	425.45	420.35	415.32	402.42
2539	524.67	518.24	510.90	505.39
3689	675.30	653.29	643.18	632.25

Fig. 7 Key size parameter analysis



318.46 ms. The throughput of the hybrid algorithm is 302.35 ms. Consider the data size of 2539 MB, which has the null encryption of 524.64 s, the throughput of the probabilistic and homomorphic algorithm is 518.24, 510.90 ms, respectively. The hybrid algorithm has the throughput of 505.39 ms. This shows that the throughput in the hybrid algorithm is much better as compare to the probabilistic and the homomorphic algorithms. The above figure represents the pictorial representation of the data encryption system. The encryption system includes the throughput of the data (Fig. 6; Table 4).

6.4 Key size parameter security analysis

When data is sent in terms of packets, if a total of 500 packets are sent after encryption, 356 were hacked in Null encryption and 285 were hacked in probabilistic algorithm, 78 packets

Table 5 Encryption key size analysis

Key size (bits)	Total no. of packets	Null	Probabilistic	Homomorphic	Probabilistic with homomorphic
0–64	500	356	285	78	42
64–128	500	327	256	64	35
128–256	500	301	230	45	26
256–512	500	258	145	20	07

were hacked in homomorphic encryption but only 43 packets were hacked in probabilistic with homomorphic encryption and vice versa the remaining packets were also hacked in the same way. While comparing with other algorithms homomorphic produces better result (Fig. 7; Table 5).

7 Conclusion

Security is the major aspect in all terms. When security level in cloud becomes high then confidentiality, integrity and privacy will be more convenient for the users and service providers. Our proposed system assures that the synthesis of probabilistic with homomorphic algorithm gives a better result for encryption. Also security attacks are low in the technique with fast execution time transmitting only limited data. Based on our proposed system the throughput, security attacks and time complexity is efficient on probabilistic with homomorphic encryption algorithm. Hence, this system of encryption is needed for the documents that are stored in the cloud storage. When all these levels increase then the QoS will get increased automatically. All these tests and experiments show that our construction is efficient in encryption with provable verification.

References

1. Glenn, R., Kent, S.: The NULL Encryption Algorithm and Its Use with IPsec. NIST and BBN Corp. (1998)
2. Luna, J., Abdallah, C.T., Heileman, G.L.: Probabilistic optimization of resource distribution and encryption for data storage in the cloud. IEEE Trans. Cloud Comput. **6**(1), 1–13. doi:10.1109/TCC.2016.2543728
3. Park, N.: Secure data access control scheme using type-based re-encryption in cloud environment. In: Semantic Methods for Knowledge Management and Communication, pp. 319–327. Springer, Berlin (2011)
4. Bendlin, R., Damgård, I., Orlandi, C., Zakarias, S.: Semi-homomorphic encryption and multiparty computation.' In: Advances in Cryptology–EUROCRYPT 2011, pp. 169–188. Springer, Berlin (2011)
5. Fontaine, C., Galand, F.: A survey of homomorphic encryption for nonspecialists. EURASIP J. Inf. Secur. **2007**, 15 (2007)
6. Van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Advances in Cryptology—EUROCRYPT 2010, pp. 24–43. Springer, Berlin (2010)
7. Koo, D., Hur, J., Yoon, H.: Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage. Comput. Electr. Eng. **39**(1), 34–46 (2013)
8. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Pelosi, G., Samarati, P.: Encryption-based policy enforcement for cloud storage. In: 2010 IEEE 30th International Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 42–51. IEEE (2010)
9. Micciancio, D.: A first glimpse of cryptography's holy grail. Commun. ACM **5**(3), 96–97 (2010)
10. Rivest, R., Scribe Ledlie: Lecture Notes 15: Voting, Homomorphic Encryption (2002)
11. Gong, L., Li, S., Mao, Q., Wang, D., Dou, J.: A homomorphic encryption scheme with adaptive chosen ciphertext security but without random oracle. Theor. Comput. Sci. **609**, 253–261 (2016)
12. Pasupuleti, S.K., Ramalingam, S., Buyya, R.: An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing. J. Netw. Comput. Appl. **64**, 12–22 (2016)
13. Delfs, H., Knebl, H.: Symmetric-key encryption. In: Introduction to Cryptography, pp. 11–31. Springer, Berlin (2007)
14. Mullen, G.L., Mummert, C.: Finite Fields and Applications. Student Mathematical Library, vol. 41, pp. 19–20. American Mathematical Society, Providence (2007)
15. Paar, C., Pelzl, J.: Understanding Cryptography: A Textbook for Students and Practitioners. Springer, New York (2009)
16. Daemen, J., Rijmen, V.: AES proposal: Rijndael (1999)
17. Schwartz, J.: US selects a new encryption technique. N. Y. Times **3** (2000)
18. Westlund, H.B.: NIST reports measurable success of Advanced Encryption Standard. J. Res. Natl Inst. Stand. Technol. **1**, 56–57 (2002)
19. Katz, J., Yung, M.: Unforgeable encryption and chosen ciphertext secure modes of operation. In: International Workshop on Fast Software Encryption, pp. 284–299. Springer, Berlin (2000)
20. Baldi, M., Bianchi, M., Chiaraluce, F., Rosenthal, J., Schipani, D.: Enhanced public key security for the McEliece cryptosystem. J. Cryptol. **29**(1), 1–27 (2016)
21. Lee, C.-C., Chung, P.-S., Hwang, M.-S.: A survey on attribute-based encryption schemes of access control in cloud environments. IJ Netw. Secur. **15**(4), 231–240 (2013)

N. Jayapandian has received his ME(CSE) from Kongu Engineering College, Erode, Tamilnadu at 2009. He has completed his BTech(IT) from Institute of Road and Transport Technology, Erode, Tamilnadu at 2006. He is active Life Member of ISTE. He is currently doing his research in Cloud Computing in Anna University, Chennai. Currently, he is working as Assistant Professor in the Department of Computer Science and Engineering at Knowledge Institute of Technology, Salem. In his 7 years of teaching experience and 1 year of Industry Experience. His research interests are Grid Computing and Cloud Computing. He has published papers in 8 International Journal, 15 international Conference and 6 National Conferences.

A. M. J. Md. Zubair Rahman has completed his PhD, in Association Rule Mining algorithms: Data Mining in 2010 in Anna University, Chennai, India and has completed his MS(Software Systems) in BITS Pilani, India in 1995. He completed ME(Computer Science and Engineering) in Bharathiar University, Tamilnadu, India in 2002. Currently, he is working as Principal at Al-Ameen Engineering College, Erode. Previously he has worked as Professor in the Department of Computer Science and Engineering, Kongu Engineering College, India. He has completed 25 years of teaching service. He has published 30 Research papers in International/National journals.