Securing Cloud Data Storage through Encryption

Ishuta U. Wankhede

Dr. Sandeep R. Sirsat

Abstract: Cloud computing is a modern technology that realizes the dream of on demand, virtualized, scalable and economical internet based computing. However, there are number of threats which make cloud vulnerable. Data Storage and Backup service of cloud can suffer due to data loss, external security breach or malicious attack from within the cloud itself. The virtualized and multitenant nature of cloud make data security attacks more intense and diversified. Encryption is the most effective method to protect integrity and security of cloud data storage. Many novel encryption techniques have been developed especially keeping cloud in view. In this paper we have discussed various approaches of cloud data encryption and popular methods under these approaches with benefits and drawbacks of each method. We have also pointed out some open problems and highlighted the scope of research in this field.

Keywords: Attribute-based Encryption, Cloud Computing, Data Storage, Encryption, Homomorphism Encryption, Searchable Encryption

I. INTRODUCTION

Cloud computing is a modern technology that realizes the dream of on demand, virtualized and economical internet based computing. It offers tempting benefits like low cost infrastructure, online data storage and back up, virtualized platform, secured on demand computing services and much more. At present, many organizations and individuals use cloud for Infrastructure management, Project Hosting, Financial Transactions, and lots of other services like Social Networking and E-shopping. However, implementation of cloud has introduced many security threats like Inter-virtual machine attacks, account hijacking, abusive use of cloud, Insecure application interfaces, technical issues in shared environment and data loss or leakage.

The virtualized and multitenant nature of cloud make data security attacks more intense and diversified. The cloud usually contains valuable private information of customers, companies, business firms and government agencies. When such data is outsourced to cloud, the user trusts the cloud service provider (CSP) for its protection. However, the details cloud implementation are not disclosed to the user. So CSP can examine or modify the data without any knowledge to user. This leads to possibility of internal intrusion and data breach. The intruder can also physically tamper or steal the storage unit. Sensitive information can be recovered from storage devices even after they are formatted. This can cause serious data theft which would be hard to track. Hence a security mechanism is needed which must be capable to protect data during its transfer, storage and also ensure secure data deletion.

Encryption has always been effective solution to protect the data. This notion works with cloud as well. Cloud users can ensure the integrity and security of data by first encrypting it and then moving the encrypted data to cloud. However, the characteristics of cloud makes data storage in cloud environment stand out different from traditional data storage. Encryption mechanism for a cloud must not only secure the data but also safeguard its confidentiality and ensures its availability. Thus traditional encryption techniques often prove inefficient for cloud. Many researchers have attempted to modify these schemes so as to make them suitable for cloud. Novel cryptographic approaches have been proposed which support operations and search directly on the encrypted data. In this paper we have discussed different encryption approaches with popular methods and their strengths and drawbacks. We have also highlighted some open problems and pointed out the scope of research in this field. We have also presented an overview of potential approaches other than encryption which provide data confidentiality and make sure that only legitimate user has access to the data.

II. RELATED WORK

Many innovative approaches have been proposed to guarantee the security, integrity and privacy of the cloud data storage. Researchers proposed that a separate authority should handle the security policies and formalize the security level agreements (SLAs) for cloud service [1], [2] and [3]. This authority was named Trusted Third Party (TTP) which works as credit rating agency. It ranks the CSPs for security by identifying current assessable security risks and the user feedback. The TTP based cloud environment ensured data protection and helped user develop trust into cloud services. A set of protocols called PasS-Privacy as Service [4] was introduced to ensure privacy and legal compliance of customer data in cloud computing architectures. It was also proposed that the encryption and data storage tasks should be carried out by different service providers. A secure model for cloud computing based storage and retrieval [5] made a provision of separate encryption/decryption service provider which encrypts data being outsourced to CSP and decrypts the same for the user. A different approach called Fog Computing [6] was proposed to secure data in the cloud using decoy technology. This technique monitors data access in the cloud for abnormal data access patterns. When unauthorized access is suspected it is first verified using challenge questions and then disinformation attack is launched. Large amount of decoy information is passed to the attacker and thus the actual user data is protected against misuse.

Data watermarking techniques [7], [8], [9] were proposed for multi-way authentications and strengthening security of confidential data access in both public and private clouds.
Crypto- and steganosystems [10] were developed and their implemented on cloud computing was analyzed. However, these schemes had limited functionality.

Cloud integrity checking mechanisms were developed that allow a set of servers to prove client that a stored file is intact and retrievable. Proofs of Retrievalability (POR) and HAIL (High-Availability and Integrity Layer) [11] were major distributed cryptographic systems that ensured data integrity. Some researchers developed new technique for intrusion detection. They stated that security attacks could be diverted to a “Honeypot”—a special resource which is designed to be attacked [12], [13]. It acts a trap set to collect details about attack patterns, purpose of attack and programs launched by attacker.

A novel fountain code based hybrid storage system [14] was proposed that combines cloud storage with Peer to Peer storage. This system provides speedy data irretrievability and short data upload time ensuring the data privacy. A secure private encryption system [15] was implemented with private cloud which achieves the security of storage resource using eCryptfs (enterprise file encryption system).

Traditional cryptographic techniques have been implemented to cloud but they have limited functionality. The virtualized and multitenant nature of cloud has led to evolution of many new encryption approaches. The concept of N-cloud [16] was proposed in which file is first split into several components which are encrypted separately and then stored into different clouds in parallel. This approach was implemented on Amazon cloud and results shown better performance with regards to data access timing. Elliptic Curve cryptography (ECC) [17] can replace the modular arithmetic in some encryption schemes with elliptic curve multiplication. ECC have smaller key sizes and are believed to be very secure because we do not yet have sub exponential algorithms required to break the ECC. Quantum Cryptography uses quantum mechanical effects to perform cryptographic tasks or to break cryptographic systems. Quantum cryptography protocol was proposed for secure communication between servers in the cloud. Hybrid encryption is used to combine two or more cryptographic techniques to reduce their individual drawbacks. One scheme blended Advanced Encryption Standard (AES) with Quantum Cryptography [18]. It uses AES for actual encryption of data while keys are distributed under quantum cryptographic primitives. Another attempt was made to combine Advanced Encryption Standard and Diffie-Hellman key Exchange [19]. New cryptosystems called Proxy re-encryption schemes [20], [21] were developed which allow third parties to alter a ciphertext which has been encrypted for one party, so that it may be decrypted by another. Such schemes facilitate access control and scalable user revocation in unreliable cloud environment.

III. CLOUD DATA ENCRYPTION TECHNIQUES

Generally, encryption algorithms are classified as private or public on basis of usage of a single secret key or two keys—one kept secret and other made public.

![Cloud Encryption](image)

<table>
<thead>
<tr>
<th>Symmetric</th>
<th>Asymmetric</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Data Encryption Standard</td>
<td>1. Homomorphic Encryption</td>
</tr>
<tr>
<td>2. Advanced Encryption Standard (AES)</td>
<td>2. Searchable Encryption</td>
</tr>
<tr>
<td>3. Triple DES</td>
<td>3. Attribute-based Encryption</td>
</tr>
<tr>
<td>4. Blowfish Encryption</td>
<td></td>
</tr>
</tbody>
</table>

**Fig. 1 Classification of Cloud Encryption Techniques**

As per their application to cloud encryption techniques can also be categorized as-

1) techniques that require prior decryption of encrypted data before operation on the encrypted data
2) techniques that facilitate operation on encrypted data directly without prior decryption

The encryption process converts the readable plaintext to unintelligible ciphertext. This ciphertext is stored into the cloud. When traditional encryption schemes are used, any modification or search on this encrypted data requires prior decryption. Before any operation, user first has to download database (on which the operation is to be performed) to client machine and then decrypt it. Once the operation is performed, modified database is re-encrypted and finally stores it back to the cloud. This approach is quite naive and suffers from huge overheads. The new encryption techniques allow operations on data without actual decryption of the ciphertext. Homomorphic encryption facilitates direct manipulation of the ciphertext while Searchable encryption and Attribute based encryption facilitates direct search over encrypted database using searchable tokens.

A. Symmetric Encryption techniques

Symmetric Encryption approach involves the use of a single key for the encryption and decryption purpose. The secret key is shared between two communicating parties. In this section, some popular symmetric data encryption techniques are discussed.

Data Encryption Standard (DES) and Triple DES: DES [22] was designed by IBM and first published in 1977. It is a block-cipher algorithm which converts 64 bits block of plaintext into same size ciphertext. The key also consists of 64 bits; however, only 56 of these are actually used by the algorithm. Eight bits are used solely for checking parity, and are thereafter discarded. DES is now considered insecure as brute force attack is easily possible on DES based cryptosystem. The differential
cryptanalysis (DC), linear cryptanalysis (LC), and Davies' attack can also break the full 16 rounds of DES.

![Fig. 2 Data Encryption Standard](image1)

Triple DES uses a key bundle that comprises three keys which improves strength of algorithm. It provides increased key sizes-168, 112 or 56 bits in order to protect against the brute force attack. However, the 168 bit key falls prey to meet-in-the-middle attack and 112 bit key is susceptible to certain chosen-plaintext or known-plaintext attacks.

**Advanced encryption standard**: The Advanced Encryption Standard (AES) [23] was established by the U.S. National Institute of Standards and Technology (NIST) in 2001. AES has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. The numbers of cycles are 10, 12 and 14 for 128, 192 and 256 bit keys respectively.

![Fig. 3 Encryption and Decryption using Advanced Encryption Standard](image2)

Many attacks like side-channel attacks, XSL attacks, related-key attack, known-key distinguishing attack were theoretically published on AES. But none of them has been implemented successfully as far as 10 round AES with 128 bit key is taken into account. Many popular cloud encryption applications available today like SafeSync, Boxcryptor, Cloudfoggger and Wuala use 256-bit AES for encryption.

**Blowfish algorithm and Twofish algorithm**: Blowfish [24] has 64-bit block size and a variable key-length from 32 to 448 bits. It uses a 16-round Fiestal cipher and four S-boxes which are arrays that accept 8-bit input and produce 32-bit output.

![Fig. 4 The round function of Blowfish](image3)

Blowfish proves to be very secure if the user manages to use strong keys. It is also faster than rest of symmetric algorithms. Two fish [25] is related to the blowfish algorithm. It is 128 bit block cipher with 16 rounds and 128,192, or 256 bits key sizes.
The cloud encryption tool TrueCrypt uses hybrid encryption combinations of AES and Twofish algorithm.

B. Asymmetric Encryption techniques

Asymmetric Encryption involves the use of two separate keys. The data is encrypted with the public key of the receiver while the receiver decrypts the data using his private key. The asymmetric encryption approaches like Searchable Encryption (SE), Attribute-based Encryption (ABE), Homomorphic encryption (HE) facilitate direct search and operation or search on the encrypted data without decrypting it. This not only strengthens the security of data but also reduces the unnecessary overheads for decryption and re-encryption of data.

Diffie-Hellman Key exchange: Diffie-Hellman key exchange [26] is one of the earliest practically implemented public key exchange. The method allows two parties to establish individual public keys and subsequently a shared secret key is generated. This key can then be used to encrypt subsequent communications using any symmetric key cipher.

The original implementation of the approach uses modular arithmetic and the Discrete Logarithm. We start with two fixed numbers p and b where p is a large prime number and b is any number less than p. Let Alice and Bob be the two users who wish to communicate. The process of key establishment between two users is as follows:

<table>
<thead>
<tr>
<th>User1</th>
<th>User2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agree on prime no p = 11 and random base say b = 6</td>
<td></td>
</tr>
<tr>
<td>Fix a secret ( S_A = 3 )</td>
<td>Fix a secret ( S_B = 5 )</td>
</tr>
<tr>
<td>Compute ( R_A = b^{S_A} \mod p ) [ 6^3 \mod 11 = 7 ]</td>
<td>Compute ( R_B = b^{S_B} \mod ) [ 6^5 \mod 11 = 10 ]</td>
</tr>
<tr>
<td>Send result ( R_A ) to User2</td>
<td>Send result ( R_B ) to User1</td>
</tr>
<tr>
<td>( K = (R_A)^{S_B} \mod p ) [ 10^5 \mod 11 = 10 ]</td>
<td>( K = (R_B)^{S_A} \mod p ) [ 7^5 \mod 11 = 10 ]</td>
</tr>
</tbody>
</table>

The result 10 is fixed as a secret key.

**Fig. 5 Diffie-Hellman Key Exchange**

All quantities except the \( S_A, S_B \) and K are made public. The numbers p, \( S_A \), and \( S_B \) chosen for practical implementation are hundreds of digits long.

The Diffie-Hellman scheme lacks authentication while establishing the secret key. Moreover, the process of fixing a key might suffer from man in the middle attack. Also, choosing the prime number and the base is computationally expensive task.

i. Homomorphic Encryption Schemes

Homomorphic encryption schemes possess a special property. These schemes facilitate direct manipulation over encrypted data. Homomorphic crypto-system is defined as a 6-tuple algebra \( H_1 = (P, C, t, t', +, \times) \) where P and C denote the plain-text space and the cipher-text space, respectively, whereas \( t \) and \( t' \) denote the encryption- and decryption functions. \( + \) and \( \times \) tag the two algebraic operations addition and multiplication respectively. Scheme is Partially Homomorphic if either + or \( \times \) operations is Homomorphic and Fully Homomorphic if both operations satisfy homomorphism.

**Partially Homomorphic Encryption (PHE) schemes:** A PHE scheme supports either additive or multiplicative operations on data but not both. Hence in spite of being secure, it is not efficient to be implemented to the cloud. Many practical implemented and secure PHE schemes have been proposed out of which [27], [29] are multiplicatively homomorphic while [30]-[34] are additively homomorphic.

Public-key cryptosystem was first practical implemented in Rivest-Shamir-Adleman (RSA) scheme [26]. This scheme is based on the problem of factoring the product of two large prime numbers called as the factoring problem and is multiplicatively homomorphic.

**Key Generation**

<table>
<thead>
<tr>
<th>( p ) and ( q ) both prime, ( p \neq q )</th>
<th>( \phi(n) = (p - 1)(q - 1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( e ) ( \leq \phi(n) )</td>
<td>( e ) ( \leq \phi(n) )</td>
</tr>
<tr>
<td>( \text{gcd}(e, \phi(n)) = 1 )</td>
<td>( \text{gcd}(e, \phi(n)) = 1 )</td>
</tr>
<tr>
<td>( d = e^{-1} \mod \phi(n) )</td>
<td>( d = e^{-1} \mod \phi(n) )</td>
</tr>
<tr>
<td>Public key ( PU = [e, n] )</td>
<td>Private key ( PR = [d, e] )</td>
</tr>
</tbody>
</table>

**Encryption**

<table>
<thead>
<tr>
<th>Plain text</th>
<th>( \text{Ciphertext} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M &lt; n )</td>
<td>( C = M^e \mod n )</td>
</tr>
</tbody>
</table>

**Decryption**

<table>
<thead>
<tr>
<th>( \text{Ciphertext} )</th>
<th>Plain text</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C )</td>
<td>( M = C^d \mod n )</td>
</tr>
</tbody>
</table>

**Fig. 6 RSA Encryption**

RSA uses variable length key ranging from 1.024 to 4,096 bit and has single round for encryption process. The block size taken for encryption is also variable and its length must be less than that of the key length. The size of the ciphertext generated is same as that of the key length.

In data encryption tools for clouds like Wuala and BoxCryptor, RSA is generally used to encrypt a secret key which is then used to encrypt the message. RSA is believed to be very secure if the sufficiently long key size is chosen. RSA provides safety against chosen plaintext attack, padding the message m. The timing attacks gain information from the implementation of the encryption scheme. RSA can avoid such attacks by ensuring that the every decryption operation takes a constant amount of time.
RSA scheme used a trapdoor function which is easy to compute in one direction, but computation of its inverse function is very difficult without special information. Later researcher proposed three new trapdoor mechanisms: a trapdoor permutation and two homomorphic probabilistic encryption schemes based on trapdoor function used in RSA [30]. A public key cryptography was also proposed which uses digital signature scheme [29] based on the difficulty of computing discrete logarithms over finite fields. However, no method supported both addition and multiplication computations.

Fully Homomorphic Encryption (FHE) schemes: FHE schemes exhibit both additive and multiplicative homomorphism. This property makes them quite suitable for clouds. The first fully homomorphic scheme was developed by Gentry (2009) [35] using ideal lattices which was almost bootstrappable. He proposed that it is sufficient to construct an encryption scheme that can evaluate its own decryption circuit to construct an encryption scheme that permits evaluation of arbitrary circuits. Such circuit that can evaluate its augmented decryption circuit is called bootstrappable circuit.

FHE with Bootstrapping functionality allowed arbitrary number of operations but was very complex to construct. So later, the ideal lattice-based scheme was replaced with a very simple somewhat homomorphic scheme (SHE) [36]. The idea behind SHE is to compromise few operations and reduce depth and complexity of the circuit. It is conceptually simpler and possesses similar properties with regards to homomorphic operations. It is as efficient as that of previous scheme. It uses hardness of approximate integer greatest common divisor.

Gentry's scheme was refined to give smaller key and ciphertext sizes [38], but it does not support fully homomorphic encryption for practical key sizes. Later on, practical FHE schemes were derived from Gentry cryptosystem [37], [39] which use elementary modular arithmetic. Researchers attempted to replace the bootstrapping function [36], [38], [40]. The new schemes were based on the Learning with Errors (LWE) and Ring LWE and elementary theory of algebraic number fields that do not require lattices to understand encryption and decryption operations. This made them very efficient and reasonably reduced the ciphertext sizes. However, they suffer from a particular drawback. The polylogarithmic factors in the per-gate computation are quite large.

Many schemes were developed to support a particular operation on ciphertext for unlimited number of times. Two such methods are- based on bilinear group [41] and another which uses matrix and vector [42]. These schemes support arbitrary addition operations on data. However multiplication or division operation could be performed only once. So such schemes were improved by using invertible Diagonal matrix [39]. The improved scheme provides all four basic arithmetic operations over ciphertext any number of times. It embeds the result sign into the diagonal matrix so that the user no more needs to record the operation type during applying it to ciphertext. It also gives stable form of ciphertext.

ii. Searchable Encryption schemes

Encrypted can preserve data confidentiality and integrity but it has some limitations. One cannot conduct direct search over traditionally encrypted data. To carry out such search operation, the customer has to store an index locally, or download all the encrypted data, decrypt it and search locally. Also, to verify the integrity of the data, the organization has to first retrieve all the data in order to verify the signatures. If the data is large, this verification procedure is obviously undesirable. Various solutions based on (keyed) hash functions can be used in this case, but they allow only fixed number of verifications.

Searchable Encryption (SE) [44] scheme provides a way to encrypt a search index so that contents are accessible to one who has appropriate tokens. It guarantees that without any token the server learns nothing about data except its length and given the token, it just can identify which encrypted document contains the token without knowing what the keyword actually is. The processor can index data and encrypt it with a symmetric encryption scheme (e.g., AES) under a unique key. It can then encrypt the index using a Searchable Encryption scheme and encrypt the unique key under an appropriate policy. Finally, it can encode the encrypted data and index in such a way that the data integrity can later be verified using a proof of storage. To conduct a search, the user has to submit the keyword and encryption key. The keyword is modified to generate trapdoor functions which search the entire stored index. Once the search is completed, all matching entries are returned to the user.

Searchable capabilities on encrypted data have been extensively investigated. Different techniques for efficient search of keywords have been used. On the basis of particular application scenario, SE has various subtypes.

Symmetric searchable encryption (SSE): SSE [44] is used in situation where the party that searches over the data is also the one who generates it. Symmetric encryption is very efficient for both parties- one who encrypts the data and other who wants to perform search over the encrypted data as it allows data to be preprocessed and stored into data structures called the secure search indexes [46]. SSE schemes guarantee better security than the asymmetric counterparts. However all these schemes use deterministic tokens. The server can keep track of the queries and determine which one is repeated. Such tracked search patterns may lead to disclosure of particular keywords.

Asymmetric Searchable Encryption (ASE): ASE schemes are appropriate in any situation where the party searching over the data is different from the party that generates it. Public-key encryption with a symmetric keyword search (PEKS) [47] was introduced to carry out search on asymmetrically encrypted database. Efforts have been made to make PEKS versatile include Conjunctive Search and Subset Search, Range Search,
Time-scope Search and Similarity Search by Boneh and Waters (2007), Shi et. al. (2007), Abdala et. al. (2008) and Zhang et. al. (2010).

The ASE schemes built their index files [53] based on the actual keywords extracted which uses predefined edit distance value. Such indexes can only support search with keywords that are identical or have similar structures to the actual keyword. The users were provided with facility to increase the edit distance value to a larger number to increase the range of the search results. However, this degraded the search quality. Wildcard based fuzzy construction [54] was proposed which let the index file hold possible variations of keyword by inserting wildcard characters. However, the increase in the edit distance of the keywords led to massive sized indexes. The size of the index was reduced with the use Dictionary based fuzzy set wildcard character [55]. It uses dictionary to call valid words within predefined edit distance value. However, practical implementations proved these schemes inefficient to handle specific queries where the Keywords are spelt different but conceptually they have similar meanings. As a solution to this problem, three different schemes were developed- Synonym-Based Keyword Search (SBKS): “Wikipedia-Based Keyword Search (WBKS)”, and “Wikipedia-Based Synonym Keyword Search (WBSKS) [56]. The SBKS first adds the keyword to the index and then checks the keyword for spelling mistakes before conducting search. The WBKS uses the style of web-based Wikipedia Encyclopedia for keyword search in index. These schemes performed better in terms of storage requirements, performance, and search quality while preserving the data security and the privacy of the uploaded data. WBSKS combines both techniques.

ii. Attribute-based Encryption (ABE)

ABE allows the specification of a decryption policy to be associated with a ciphertext. Attributes are qualities of a party that can be established through relevant credentials. Each user in the system is provided with a decryption key that has a set of attributes associated with it. A user can then encrypt a message under a public key and a policy. Decryption will only work if the attributes associated with the decryption key match the policy used to encrypt the message. These schemes are highly resistant against collusion attacks.

The first Attribute based Encryption [57] scheme was called ‘Fuzzy Identity Encryption’. In this scheme user is identified by a set of attributes, and some function of this attributes are used to determine decryption ability for each ciphertext. However, such schemes use fixed distance metrics between identities which restricts the range of searching. Key-policy Attribute-based Encryption scheme [58] has solved this problem to facilitated fine-grained sharing of encrypted data. This scheme labels each ciphertext with a set of descriptive attributes. Each private key is associated with an access structure that specifies which type of ciphertexts the key can decrypt. The access structure is specified in the private key, while the ciphertexts are simply labeled with a set of descriptive attributes.

ABE schemes used attributes to describe the encrypted data and built policies into user’s keys until Ciphertext policy Attribute-based Encryption (CP-ABE) scheme [59] was proposed which uses attributes to describe a user’s credentials while party performing the encryption determined policy about who can decrypt. This scheme is secure against collusion attacks and can keeps encrypted confidential even if the storage server is untrusted. While this scheme. Access structures in his system are described by a monotonic “access tree”, where nodes of the access structure are composed of threshold gates and the leaves describe attributes.

No ABE scheme possessed satisfactory method to represent negative constraints in a key’s access formula. Attribute-based encryption with non-monotonic access structure overcame this weakness [60]. It allowed a private key to be expressed in terms of any access formula over attributes by describing access policies in terms of monotonic access structures with negative attributes. This scheme represented ABE non-monotonic access structures in terms of ABE monotonic access structures with negative attributes. Hierarchical attribute-based encryption scheme [61] extended CP-ABE with hierarchical structure of users by means of compound attributes. Flexible and Direct Revocation (FDR-CP-ABE) [62] formalized the notion of Ciphertext policy ABE and gave out a concrete construction, which supports direct attribute and user revocation and is applicable to the data sharing architecture. The proposed scheme outperforms the previous revocation related methods. It and has constant-size ciphertexts and only partial ciphertexts need to be updated whenever revocation events occur.

IV. CONCLUSION

Many approaches have been suggested for protection of cloud data till date. However, encryption comes out to be most efficient of all. The traditional schemes like AES and RSA are currently used to encrypt cloud data in practice. However, these schemes hamper the benefits of cloud due to their inefficiencies in cloud adoption.

Homomorphic encryption schemes have huge potential for cloud adoption but they need to be more agile before they are practically implemented for cloud security. One of the biggest drawbacks of FHE is their complexity even for basic operations. The ciphertext sizes and public keys of this size are very large. Another drawback is that FHEs are vulnerable to malwares in some cases. Work is expected on FHE to make them practically faster, compact, secure and more efficient to be used on cloud data by reducing the size of generated ciphertexts. The bootstrapping function in FHE needs to be more efficient or to be replaced by some alternative. SHE schemes are proven resistant to all known attacks; the only problem is to balance the tradeoff between efficiency of the scheme and preservation of hardness of the approximate-gcd problem.
SSE poses tradeoff between efficiency and functionality. The development of an efficient technique with better functionality still remains a challenge. ASE is functionally strong but the main disadvantages are inefficiency and weaker security guarantees. It requires the evaluation of pairings on elliptic curves which is a relatively slow operation compared to evaluations of (cryptographic) hash functions or block ciphers. In addition, in the typical usage scenarios for ASE data cannot be stored in efficient data structures. Attacks on searchable encryption have proven that these schemes fail prey to tracking of searchable patterns. Work is expected to find a secure the usage of these schemes.

Attribute based encryption techniques are quite efficient and secure. However work is needed to make ABE policies more efficient in expression.

REFERENCES


20. Q. Liu , G. Wang and J. Wu, "Clock-Based Proxy Re-encryption Scheme in Unreliable Clouds", 41st International Conference on Parallel Processing Workshops 2012, pp: 304-305


43. S. Kamara and K. Lauter, “Cryptographic Cloud Storage”, Revised Selected Papers from FC 2010 Workshops, RLCP5, WECSR, and WLC 2010, IEEE.


AUTHOR’S PROFILE

Ishuta U. Wankhede
She received Bachelor of Science Degree in 2012 and Masters Degree in Computer Science from Sant Gadge Baba Amravati University, Amravati, M.S., India in 2014. She is currently working as an Instructor for Computer Science at G. S. Science Arts and Commerce College, Khamgaon, India. Her areas of interest for research include Cloud computing and Network Security.

Dr. Sandeep R. Sirsat
He received Masters Degree in Computer Science in 1991 and completed PhD in 2014 from Sant Gadge Baba Amravati University, Amravati, M.S., India in 2014. He is currently working as Associate Professor and Head at Department of Computer Science, Shri Shivasji Science and Arts College, Chikhli, India. His research interests include Data Mining, Text Mining, Web Mining, Cyber Crime and Cloud Computing.