Encrypted Data Searching Techniques and Approaches for Cloud Computing: A Survey

Lija Mohan and M. Sudheep Elayidom

Abstract Today, Cloud Computing has paved the way for enormous computing and storage. Cloud servers are third party systems which could be rented on demand basis and paid on usage basis. More and more users are adopting cloud based applications but the only factory that hinders its development is security issue. Users have a fear of trusting a third party system like cloud and they show reluctance to outsource their sensitive information to cloud. Encryption seems to be a direct solution but it limits the computability on data. Hence encryption schemes should be chosen based on the application they need to implement. In this article we study the basic encryption schemes which are widely used in cloud scenario. We compare these schemes in terms of their computational complexity, security, performance etc.

Keywords Cloud · Security · Survey · Encryption schemes · Encrypted data search

1 Introduction

Cloud Computing [1] is synonymous to 'Internet Based Computing' where users could do any type of computations if they have internet connectivity and a web browser to provide the interface. These types of computations are made possible by connecting a lot of virtual resources together and granting access to authorized users. Merits of Cloud computing includes elasticity, reliability, economical computing, pay per usage policy, global accessibility, usability and ease of maintenance.

Lija Mohan (✉) · M. Sudheep Elayidom
Department of Computer Science, School of Engineering, Cochin University of Science & Technology, Kochi, Kerala, India
e-mail: lija@cusat.ac.in; joinlija@gmail.com

M. Sudheep Elayidom
e-mail: sudheep@cusat.ac.in

J.K. Mandal et al. (eds.), *Proceedings of the First International Conference on Intelligent Computing and Communication*, Advances in Intelligent Systems and Computing 458, DOI 10.1007/978-981-10-2035-3_11

1.1 Challenges Associated with Cloud Computing

In spite all the merits we discussed above, normal users still keep away from cloud fearing that the security and privacy needed by their data will not be met. The cloud model follows a highly dynamic environment where the data will be partitioned and stored in multiple locations. Data will be kept replicated as well. Neither the service providers nor the owners of the data will know about the details of machines where their data will be stored. Providing Fine Grained Access Control [2] is another problem to deal with.

1.2 Motivation for a Solution

To implement solutions for BigData as well as problems involving severe computations, cloud computing is the most appropriate one. But use of public cloud has resulted in a lot of security and privacy issues. Several reports [3–9] reveal the security breaches and data theft took place in real world scenarios.

'Encryption of data' seems to be a first hand solution to ensure secrecy and privacy. But encryption limits the computations that can be performed on data like retrieving a particular file containing a specific keyword or extracting features from an image etc.

There exists a trade-off between security and usability. But the solution here is to apply the security mechanism in a way that it will not limit the functionality as well. This article aims to provide an insight to some encryption schemes that can be applied to cloud system based on the specific applications.

2 Review of Existing Solutions to Enable Encrypted Data Search

Basically we identified six different ways to search on encrypted data, each based on one of the following cryptographic primitives:

2.1 Property Preserving Encryptions (PPE)

PPE schemes [10] encrypt text in such a way that it leaks certain properties of the underlying data.

Different PPE schemes are proposed based on the property that is leaked. The basic one is 'Deterministic Encryption' [11] in which one message always generates same cipher text after encryption. Thus by comparing the cipher text one can determine whether the messages are same. These types of encryptions are hence applicable to problems where similarity is compared.

For e.g. If ‘m1’ encrypts to ‘c1’ and ‘m2’ encrypts to ‘c2’, then by comparing the value of c1 and c2 we can determine whether m1 is equal to m2.

Order Preserving Encryption (OPE) [12–16], Orthogonality Preserving Encryption etc are some variations of Property Preserving Encryption. Bellare et al. [14] proposed a method where PPE scheme can efficiently applied on securing databases.

Computational Complexity
Search complexity is O(nm), where ‘m’ is the number of documents, i.e., linear complexity. But data structures like Binary search trees can improve the speed.

Security of PPE
Since encryption on m1 always generate same cipher text security is limited since this can lead to some statistical leakages.

2.2 Functional and Identity-Based Encryption

The concept behind Functional Encryption was first proposed by Sahai and Waters in a conference and later formalized and proved to be practical by Boneh, Sahai, Waters and by O’Neill [17]. Identity Based Encryption, Attribute Encryption, Predicate Encryption etc can be considered as variations of Functional Encryption.

The working of Identity encryption can be explained by a simple real world application: Alice want to send some secret message to Bob. Alice knows that Bob works at Google.

According to Functional Encryption, Google will initialize the security system by generating a pair of master keys (msk, mpk), where one is a secret key and other is public. Google then distributes mpk together with a valid certificate to its authorized employees.

To encrypt a message ‘m’, Alice will collect Google’s Master Public key mpk, and apply the encryption algorithm on ‘m’ using mpk and Bob’s public identity, ‘bob@google.com’.

c=E(mpk,‘bob@google.com’, m).

For Bob, to decrypt the message ‘c’, Bob generates his secret key using Google’s master key and his own id.

sk=KeyGen(msk,‘bob@google.com’).

Bob recovers the message by applying the decryption algorithm.

m=Dec(sk,c).

The advantage of this method is its simplicity. Without revealing any public key of Bob, Alice can send encrypted messages to him or any person in the organization knowing only the public key of that organization.

In case of attribute based encryption, some attributes approved by Organization will be utilized for encryption. For e.g. Consider a hospital domain. Alice needs to upload a file which can be viewed by a person if he is a ‘doctor specialized in oncology with masters degree’. Hence the attributes can be ‘doctor’, ‘MD’, ‘Oncology’ etc.

Computational Complexity
Complexity is O(nm) as the algorithms has to try to decrypt each cipher text in the Encrypted domain. But always $m \ll n$, hence the time complexity needed will always be more compared to PPE.

Security
This approach substantially ensures security since neither statistical leakages or brute force attacks exist in the system.

2.3 Fully Homomorphic Encryption

A cryptosystem that supports both addition and multiplication operations on encrypted data is called fully homomorphic encryption (FHE) and is far more powerful. Homomorphic encryption schemes process data in its encrypted form itself. No decryption is needed. Thus these types of applications are best suited for third party computations like cloud computing. Encryption does not reveal any information to external agents.

Homomorphism with respect to addition or multiplication has been made possible since the development of RSA [18] and paillier encryption [20]. They are called partial homomorphic systems. The concept of Fully Homomorphic encryption which made possible additions and multiplications over encrypted data was first proposed by Gentry [19].

Craig Gentry [19] developed lattice based cryptosystem to achieve fully homomorphic property and he was successful in evaluating arbitrary depth circuits. The scheme was also bootstrappable meaning as the circuit grows, the noise rises and ultimately the circuit will get capable of decrypting its own encrypted data i.e., the circuit gains self referential property. Hence in 2012 Gentry along with Vaikundanathan [7] proposed a variation of the original scheme using the property of ideal lattices over integers.

Let us illustrate fully homomorphic symmetric encryption scheme with an example:

Let the shared secret key be an odd number, 101. The domain consist of bits {0, 1}. To encrypt $m = 1$; Choose a random small prime number $r = 5$, and large $q = 9$.

Table 1 Complexity comparison

Dimension	KeyGen	PK size	Re-crypt
512 200,000-bit integers	2.4 s	17 MB	6 s
2048 800,000-bit integers	40 s	70 MB	31 s
8192 3,200,000-bit integers	8 min	285 MB	3 min
32728 13,000,000-bit integers	2 h	2.3 GB	30 min

$$\text{Encryption}\,(\text{m}) = \text{c} = \text{m} + 2\text{r} + \text{pq} = 11 + 909 = 920$$

Here cipher text will always be close to a multiple of p.
Therefore, m ≈ LSB of distance to nearest multiple of p.

$$\text{Decryption is m} = (\text{c} \,\%\, \text{p}) \,\%\, 2 = 11 \,\%\, 2 = 1.$$

Computational Complexity
Comparing Fully homomorphic encryption using integers and ideal lattices, the flatter method have exponential complexity which is not at all tolerable. Integer method is assumed to have complexity λ5. The table below describes the complexity details (Table 1).

2.4 *Oblivious RAM*

Oblivious RAM concept was first proposed by Goldreich [18] as a method to implement software protection on third party servers. But at that time it seems to be irrelevant because cloud computing or third party computing were not at all in practice. But now the work has gained so much application context related to cloud storage.

An ORAM scheme basically consist of 3 stages Setup, Read and Write.

- Setup: inputs are
  - security parameter 1K.
  - RAM (memory array) of N items.
  - Outputs: Secret key K and an oblivious memory ORAM.
- Read: A two-party algorithm run between client and server. The client runs the Read function with a secret key K and an index i as input while the server runs the Read Function with an oblivious memory ORAM as input. At the end of the execution, the client receives RAM[i] while the server receives Ɛ, i.e., null.

  Read((K, I),ORAM) = (RAM[i], Ɛ).

Table 2 Critical comparison of searching schemes

Scheme	Search complexity	Search type	Recalculation?
PPE	O(n)	Linear	No
Functional encryption	O(d)	Pre-processed index	No
SSE	O(1)	Pre-processed index	Yes
PEKS	O(n)	Linear	No
Rank ordered	O(d)	Pre-processed index	Yes

Table 3 Summary of major search schemes and their ability to perform certain search options

Scheme	Exact match	Sub match	Case insensitivity	Regex	Proximity	Stemming
Practical technique	Yes	No	No	No	Yes	No
Secure indexes	Yes	Maybe	Maybe	No	No	Maybe
SSE	Yes	Maybe	Maybe	No	No	Maybe
PEKS	Yes	Maybe	Maybe	No	No	Maybe
Rank ordered	No	No	Yes	No	No	Yes

- Write: Two party protocol executed between the client and a server. The client runs the Write function with a key K, an index i and a value v as input and the server runs the Write function with an oblivious memory ORAM as input. At the end of the protocol, the client receives nothing (again denoted as Ɛ) and the server receives an updated oblivious memory ORAM' such that the *i*th location now holds the value v. We represent this as

  Write ((K, i, v), ORAM) = (Ɛ, ORAM').

Security of ORAM
ORAM is constructed such that server is unable to derive any information about RAM. Read and Write functions do not leak information about the index and values either.

Computational Complexity
Since FHE has to be implemented in Read and Write phase, ORAM is the slowest of all techniques mentioned above (Tables 2 and 3).

3 Conclusion

Cloud computing is gaining so much interest due to the huge amount of data generated and need for computations to be performed on these data. Security and privacy is the only factor that hinders the usability of cloud. Users of data do not

Table 4 Summary of different encryption schemes

Scheme	Summary
PPE	Fast search, but at the expense of small information leakage
Functional encryption	Easy implementation, Secure but slow search time
FHE	Secure but Application Dependant, We should choose a homomorphic function based on the application context in which it is implemented
ORAM	Most secure solution which hides even the access pattern

trust a third party agent like cloud to store their sensitive data. The solution is encryption. But encryption limits the computability of data. To eliminate such limitations we can choose encryptions that properly match each application. This article surveys the different encryption schemes available in literature and compare them based on factors like security, complexity etc. The table below provides a short summary of all the schemes mentioned (Table 4).

Acknowledgments The authors sincerely thank Department of Science & Technology, India for supporting the research work by granting Inspire Fellowship.

References

1. M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A.Konwinski, G. Lee, D. Patterson, A. Rabkin, and M. Zaharia, “A View of Cloud Computing,” Comm. ACM, vol. 53, no. 4, pp. 50–58, 2010.
2. Shashank Agrawal, Shweta Agrawal, Saikrishna Badrinarayanan, Abishek Kumarasubramanian, Manoj Prabhakaran, and Amit Sahai. Function Private Functional Encryption and Property Preserving Encryption: New Definitions and Positive Results. Cryptology ePrint Archive, Report 2013/744, 2013.
3. C. Leslie, “NSA Has Massive Database of Americans’ Phone Calls,” http://usatoday30.usatoday.com/news/washington/2006-05-10/, 2013.
4. R. Curtmola, J.A. Garay, S. Kamara, and R. Ostrovsky, “Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions,” Proc. ACM 13th Conf. Computer and Comm. Security (CCS), 2006.
5. C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, “Secure Ranked Keyword Search over Encrypted Cloud Data,” Proc. IEEE 30th Int’l Conf. Distributed Computing Systems (ICDCS), 2010.
6. S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski, “Zerber+r: Top-k Retrieval from a Confidential Index,” Proc. 12th Int’l Conf. Extending Database Technology: Advances in Database Technology (EDBT), 2009.
7. M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, “Fully Homomorphic Encryption over the Integers,” Proc. 29th Ann. Int’l Conf. Theory and Applications of Cryptographic Techniques, H. Gilbert, pp. 24–43, 2010.
8. M. Perc, “Evolution of the Most Common English Words and Phrases over the Centuries,” J. Royal Soc. Interface, 2012.
9. O. Regev, “New Lattice-Based Cryptographic Constructions,” J. ACM, vol. 51, no. 6, pp. 899–942, 2004.

10. Mihir Bellare, Thomas Ristenpart, Phillip Rogaway, and Till Stegers. Format-preserving encryption. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors, Selected Areas in Cryptography, volume 5867 of Lecture Notes in Computer Science, pages 295–312. Springer, 2009.
11. Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'Neill. Order-preserving symmetric encryption. In Antoine Joux, editor, EUROCRYPT, volume 5479 of Lecture Notes in Computer Science, pages 224–241. Springer, 2009.
12. Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam ONeill. Order-preserving symmetric encryption. Cryptology ePrint Archive, Report 2012/624, 2012. http://eprint.iacr.org/.
13. Alexandra Boldyreva, Nathan Chenette, and Adam O'Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In Phillip Rogaway, editor, CRYPTO, volume 6841 of Lecture Notes in Computer Science, pages 578–595. Springer, 2011.
14. Alexandra Boldyreva, Nathan Chenette, and Adam ONeill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. Cryptology ePrint Archive, Report 2012/625, 2012. http://eprint.iacr.org/.
15. Dan Boneh and Xavier Boyen. Efficient selective identity-based encryption without random oracles. J. Cryptology, 24(4):659–693, 2011.
16. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, EUROCRYPT, volume 3494 of Lecture Notes in Computer Science, pages 440–456. Springer, 2005.
17. E. Bach and J.O. Shallit. Algorithmic Number Theory. Foundations of computing. MIT Press, 1996.
18. Goldreich, O. "Towards a Theory of Software Protection and simulation by Oblivious RAMs" STOC 87.
19. C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," Proc. 41st Ann. ACM Symp. Theory of computing (STOC), pp. 169–178, 2009.
20. Lija Mohan, Sudheep Elayidom, "Fine Grained Access Control and Revocation for secure cloud environment- a polynomial based approach", International Conference on Information and Communication Technologies, to be published in Elsevier Procedia, December 2014.