© 2015 IJSRSET | Volume 1 | Issue 5 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Enhanced File Security using Encryption and Splitting technique over Multi-cloud Environment

Prathamesh P. Kudtarkar*, Jayesh D. Pagare, Sujata R. Ahire, Tejaswini S. Pawar
Computer Engineering, L.G.N. Sapkal Collage of Engineering Nashik, Maharashtra, India

ABSTRACT

Cloud computing is a field which has been fast growing over the last few years. The fact that cloud can provide both computation and storage at low rates makes it popular among corporations and IT industries. This also makes it a very captivating proposition for the future. But in spite of its promise and potential, security in the cloud proves to be a cause for concerns to the business sector. This is due to the out sourcing of data onto third party managed cloud platform. These concerns security also make the use of cloud services not so much flexible. We provide a secure framework to stored data to be securely in the cloud, at the same time allowing operations to be performed on data without compromising of the sensitive parts of the data. A combination of searchable encryption with Partial Homomorphism is proposed.

Keywords: Cloud Computing, Cloud Security, Homomorphic Encryption, Searchable Encryption, Secure Socket Layer

I. INTRODUCTION

Cloud computing has become the synonym of anything that involves delivering of services over the Internet. The impact of cloud services on the business sector is tremendous. With the increase in the end-users, there is an increasing growth in the number of Cloud Service Providers (CSP) as well. The CSP is a third party that maintains and manages information about another entity. As users of both private and public sectors become more and more aware of cloud and it's plethora of services, they are searching for ways of making it more flexible and cost efficient. The strength of the cloud lies in that it offers both storage and computational power: a necessity for any company. Ideally, a company would want to use the CSP that provides the best package in their field of service. But as it happens, each CSP has different prices and strengths for each of their services and hence, a CSP who offers cheap storage may not offer good computation power. In such cases, a customer would opt to use multiple CSPs to make the best use of resources. Hence, in this scenario, one important factor that perhaps is given less importance is the security of data.

Ironically, the main reason for customers still opting out of cloud is the potential vulnerability of cloud when it comes to security. The Cloud Security Alliance (CSA) [1] in a report has concluded that security threats like malicious insiders, data breaches, etc. still hamper the popularity of cloud. Major security concerns are privacy, integrity and confidentiality.

Even in a single cloud, security is a major concern. However, this risk is multiplied when multiple clouds are involved. If there is data breach of a cloud user using multiple clouds for operations, it would be near impossible to determine at which CSP the data breach occurred or which malicious insider at which CSP sold this data. Hence, a possible solution is needed such that user can make use of multiple CSPs without fear of security breaches. In this paper, we propose a new approach to enhance security in cloud that particularly suits this scenario, ensuring that user can be sure of the security and confidentiality of their data.

The major problem in cloud is that when the user's data is stored on the cloud, the data could be physically located anywhere in the globe and it is not possible for the user to keep track of who has access to their data. Added to this is also the fact that storing raw data in the cloud implies an easy access to hackers and rivals, as the CSP's security offering would be the only barrier

between these entities and the raw data. Since the cloud hosts millions of other users, it could be possible that one CSP could collude with any other person and sell users data stored on the cloud. Unfortunately, this stored data may contain sensitive information, which is vital to the user's company or clients. The user loses direct control of his data and due to the raw nature of the data trusting the CSP becomes a rather forced choice.

Data in cloud could be broadly tagged as either static or computational. On the storage cloud, the data is at rest and in order to protect this data, the obvious option is to encrypt it. Encrypting this static data means, jumbling it up into gibberish that is unreadable or non-understandable. Several encryption algorithms have been proposed to be used for this purpose, the most popular being AES-256 bit. However, the disadvantage of conventional encryption standards is the necessary to decrypt the data before searching the same for partial retrieval. This search may be based on a small part of the data, such as any word or record. However, if decryption at the CSP is to be avoided, the only option left to the user is to retrieve the whole encrypted dataset, decrypt it, search and retrieve the necessary data, re-encrypt the data and store the same back in the cloud. In situations where it is necessary to retrieve a single user record from a database, this method would cause high overhead, since it requires the transfer of the data twice with additional cost for encryption and decryption. This overhead can only be avoided if there is a way to search on the encrypted data. Normal encryption schemes do not have the ability to search on encrypted data. To solve this problem Searchable Symmetric Encryption (SSE) was introduced. The advantage of this idea is to allow users to search for a word or record on encrypted text and retrieve that record. This saves a huge amount of time and effort taken by user.

When computation is required to be performed on the data, data is said to be dynamic, in other words, the value changes with every operation. In such cases, the data is expected to be raw for calculation. But as discussed, raw data is highly volatile. Conventional encryption encrypts the data, making it obtuse for hackers. However, it does not allow operations on it. A huge breakthrough while hunting for a solution was the introduction of Homomorphic Encryption. The idea is that, data can be encrypted in such a manner that allows computation to be done on it. This computed encrypted data upon decryption, returns the same answer as computations done on raw data.

Although solutions have been separately proposed for data at rest and data to be manipulated upon, it is important to identify a single system that handles both these cases simultaneously. This ensures the privacy of data in a multiple cloud environment. In this paper, it is proposed that it is enough to obscure only the sensitive part of the data, provided, the protection mechanism is strong. By doing so, even if a malicious user gets hold of the data, the document's integrity is not wholly lost, since the encrypted fields are not accessible.

We propose a solution that allows a user to search on Encrypted data, retrieve and perform some computations on it. This approach lets the users decide what parts of the data they are willing to reveal to the cloud while securely hiding the sensitive parts. A combination of Searchable Encryption along with Partial Homomorphic technique like Shamir's Secret Sharing Algorithm is chosen to support our proposal.

II. METHODS AND MATERIAL

A. Related Work

Searchable encryption can be achieved in two ways: Usingan index or by sequential search. Dan Boneh et al. [2], Yanjiang Yang et al. [3] proposed related research in the Searchable Encryption field. Most of the work is based on creating an index of keywords for the searchable encrypted file and mapping the indexes to the words when searched. When data users input a keyword, a trapdoor is generated for this keyword and then submitted to the cloud server. A comparison between the trapdoor and index is executed by the cloud server when it receives the trapdoor. All the files/records, which this keyword is a part of, are sent to the data user. Sequential scan of encrypted data allows for controlled searching [4]. All practical implementations can be built using this scheme since it offers less complexity in search. Wang et al. [5] proposed an encryption technique using a secure ranked keyword search by combining inverted index with order - preserving symmetric encryption (OPSE). They employed numerical relevance scores technique to order the retrieved files. Although this method enhances system usability and saves communication overhead, it supports only single

keyword ranked search and hence is not very useful for many applications.

Homomorphic crypto-systems can be broadly divided into two types: Partial Homomorphic systems and Fully Homomorphic systems. Fully Homomorphic systems are those systems, which do not have any limitation on the type of operation nor the number of operations that can be performed on the cipher texts. This is an ideal type of system and was considered impractical and was not even theoretically proved until in 2009, Craig Gentry [6], using lattice-based cryptography showed that fully Homomorphic system are theoretically achievable. However, this scheme did not allow for its implementation to be used practically as the complexity and length of cipher texts keeps increasing with increase in security levels. The key generated is also, too large. Marten van Dijk, Craig Gentry, Shai Halevi and Vinod Vaikuntana than [7] proposed the second Fully Homomorphic encryption scheme. This scheme uses many of the tools proposed in Gentry's construction, but does not require ideal lattices. This technique has almost the same efficiency as Gentry's original proposition. The HElib, a library released in GitHub [8], implements the Brakerski-Gentry-Vaikuntana than (BGV) [9] Homomorphic encryption scheme, along with many optimizations to make Homomorphic evaluations run faster.

A Homomorphic system having a limitation on the type of operation or the number of operations that can be performed on the cipher texts is called Partially Homomorphic. Examples of some such systems are RSA [10], Paillier [11] and Elgamal . When an allowed operation on the encrypted data is restricted to only multiplication, it is said to be multiplicatively Homomorphic. Both Elgamal and Unpadded RSA are such systems. On the other hand Paillier is additively Homomorphic since addition operation can be performed on the encrypted data. Although original Elgamal is multiplicative, a variant of Elgamal [12] is proposed where it could be made additive. Like most Homomorphic encryption mechanisms this also has a restriction on the size of the data that can be encrypted. This characteristic of Homomorphic encryptions essentially restricts the use of these mechanisms. However, it is observed that these mechanisms work well when applied on a smaller size of the data. In our work, we have used RSA Encryption and we have included the change to make RSA additively Homomorphic.

B. Proposed Work

This paper focuses on the problems related to the data security aspect of cloud computing. As information and data will be distribute with a third party. Cloud computing users desire to avoid an unreliable cloud provider. Protecting private and important information, such as patient's medical records or a credit card details from attackers or malicious insiders is of critical importance. In addition, the probable for relocation from a single cloud to a multi-cloud environment is examined and research related to security problems in single and multi-clouds in cloud computing is surveyed.

Advantages:

1. Service Availability.
2. Data Integrity.
3. The user runs custom applications using the service provider's resources.
4. 4. Cloud service providers should verify the security of their customers data and should be responsible if any security threat affects their customers service infrastructure.

- **Shamirs Secret Sharing Scheme**

In this paper, we use Shamir's secret sharing scheme, is based on polynomial evaluations. On input secrets computation operations are performs and distributes the resulting shares to other parties. When the secret has to be reconstructed then parties give their shares to the dealer, which can then combine the shares and retrieve the secret.

With this Shamir's secret sharing scheme, an intruder needs to retrieve at least three values to be able to find out the real value that desire to hide from the intruder. This depends on Shamir's secret sharing algorithm with a polynomial function technique which declare that even with full knowledge of (k, 1) clouds, the service provider will not have any knowledge of VS (VS is the secret value). The hackers need to retrieve all the information from the cloud providers to know the real value of the data in the cloud. If the attacker hacked one cloud providers password or even two cloud

providers passwords, they need to hack the third cloud provider (in the case where k = 3) to know the secret number which is the worst case scenario. Hence, for replicating data into multi-clouds use a multi-share technique, this may reduce the threat of data intrusion and increase data integrity.

Suppose that our secret is VS
We wish to divide the secret into 6 parts$(n = 6)$, where any subset of 3 parts $(k = 3)$is sufficient to reconstruct the secret. At random we obtain 2 $(k - 1)$
Numbers**:** ***R1*** and ***R2***.

Our polynomial to produce secret shares (points) is therefore:

$$f(x) = VS + R1x + R2x^2$$

We construct 6 points from the polynomial:

$$(x_0, y_0)(x_1, y_1)(x_2, y_2)(x_3, y_3)(x_4, y_4)(x_5, y_5)$$

We give each participant a different single point (Both x and $f(x)$).

Reconstruction
In order to reconstruct the secret any 3 points are enough.

Let us consider any Three Parts

$$(x_0, y_0); (x_1, y_1); (x_2, y_2)$$

We compute Lagrange basis polynomials:

$$\ell_0 = \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2}$$

$$\ell_1 = \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2}$$

$$\ell_2 = \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1}$$

Therefore:

$$f(x) = \sum_{j=0}^{2} y_j \cdot \ell_j(x)$$

So we get original polynomial

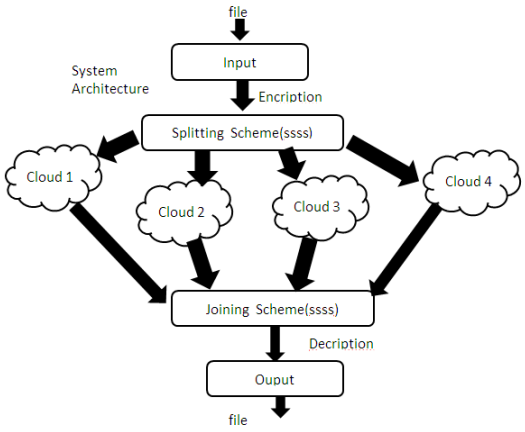

Figure 1: System Architecture

III. RESULTS AND DISCUSSION

ALGORITHM

A. Shamir's secret sharing scheme

Shamir's secret sharing scheme (ssss) is based on polynomial evaluations. The central party is the dealer that performs share compute operations on input secrets and distributes the resulting shares to other parties. When the secret has to be rebuild, the parties give their shares to the dealer, which can then combine the shares and retrieve the secret.

In Shamir's scheme shares are evaluations of a randomly generated polynomial. The polynomial f is generated in such a way that the evaluation f (0) reveals the secret value. If there are enough evaluations, the parties can reconstruct the polynomial and compute the secret. Algorithm 1 states how shares are computed in Shamir's scheme.

Algorithm 1: Share computation algorithm for Shamir's scheme
Data: finite field $\mathbf{F}$, secret data $s \in \mathbf{F}$, threshold k, number of shares n
Result: shares $s_1, \ldots, s_n$
Set $f_0 = s$
Uniformly generate coefficients $f_1, \ldots, f_{k-1} \in \mathbf{F}$
Construct the polynomial $f(x) = f_0 + f_1 x + \cdots + f_{k-1} x^{k-1}$
Evaluate the polynomial: $s_i = f(i)$, $(i = 1, \ldots, n)$

Note that the indices of the shares start from one, as the author cannot output s0 = f (0), because it is the secret value. The resulting shares s1….sn can be distributed to their holders. If the original value needs to be retrieved, they need a subset of at least k shares. Note that it is

important to store the index i together with the share si, because it is later needed for reconstruction.

The classical algorithm of Shamir's scheme reconstructs the whole polynomial, whereas they describe versions optimized for reconstructing only the secret f (0) = s. They only need to compute f (0) so for our purposes we can simplify the base polynomial's bi(x) as follows:

$$\beta_i = b_i(0) = \prod_{\substack{j=1 \\ i \neq j}}^{k} \frac{(-a_j)}{(a_i - a_j)}.$$

If the shares are computed using Shamir's scheme then algorithm 2 retrieves the secret value s.

Algorithm 2: Share reconstruction algorithm for Shamir's scheme

Data: finite field F, shares $s_{t_1}, \ldots, s_{t_k} \in F$ where $t_j \in \{1, \ldots, n\}$ are distinct indices
Result: secret data s
compute the reconstruction coefficients β_i according to equation (3)
compute $f(0) = s_{t_1}\beta_{t_1} + \cdots + s_{t_k}\beta_{t_k}$
Return $s = f(0)$

B. Secure computation with shares

They will now show what can be done with the shares once they have been distributed. They will investigate the possibility of using the homomorphic property of the secret sharing scheme to perform operations with the shares. In the following assume that a k-out-of-n threshold scheme is used. Assume that we have n parties p1….pn and the dealer gives each one of them a share according to its index.

C. Addition

Assume that we have shared values [u] = [u1….. un] and [v] = [v1……vn]. Because the evaluation mapping is a linear transformation, they can add the shares of [u] and [v] to create a shared value [w] so that u + v = w. Each party k has to run the protocol given in Algorithm 3 to add two shared values.

Algorithm 3: Protocol for adding two Shamir shares for node k

Data: shares u_k and v_k
Result: share w_k that represents the sum of $[u]$ and $[v]$
Round 1
$w_k = u_k + v_k$

D. Multiplication with a Scalar

Assume that we have a shared value [u] = [u1….. un] and a public value t. They can multiply the shares ui with t so that the resulting shares represent the value [w] = t[u]. Algorithm 4 shows the protocol for multiplication a share value by a scalar.

Algorithm 4: Protocol for multiplying Shamir shares by a scalar value for node k

Data: shares u_k and a public value t
Result: share w_k that represents the value of $t[u]$
Round 1
$w_k = tu_k$

E. Multiplication

Assume that they have shared values [u] = [u1…….. un] and [v] = [v1……vn]. Share multiplication, unfortunately, cannot be solved with the linear property of the transformation, as multiplying two polynomials with the same degree gives a polynomial with double the degree of the source polynomials. This means that they must use a k-out-of-n threshold scheme where $2k \leq n$ and the polynomials must have a degree of at most 2k. By multiplying the respective shares, the miners actually compute a share that represents the polynomial storing the product of the secrets. However, they must reconstruct the secret stored in the product polynomial and reshare it to make further multiplications possible.

Otherwise, the multiplication of the product polynomial with another one will give us a polynomial with a degree larger than n and we cannot reconstruct the secret from such polynomials anymore.

They can use pre-computed values of the optimized base polynomials βi needed in the protocol. This requires each node to know its number and also how many other nodes there are, but that is a reasonable assumption. Algorithm 5 gives the complete protocol for multiplying Shamir shares.

Algorithm 5: Protocol for multiplying two Shamir shares for node i

Data: shares u_i and v_i, precomputed value β_i
Result: share w_i that represents the value of $[u][v]$
Round 1
$z_i = u_i v_i \beta_i$
Share z_i to $z_{i_1}, \ldots, z_{i_n}$ using the same scheme as the dealer uses
Send to each other node $P_l, i \neq l$ the share z_{i_l}
Round 2
Receive shares $z_{ji}, j \neq i$ from other nodes
$w_i = z_{i_i} + \sum_{\substack{j=1 \\ j \neq i}}^{n} z_{ji}$

I. CONCLUSION

By implement the cloud based storage system that solves many business secure and safe storage issues. But on the other side many expert state that it is more dangerous to put the data over single cloud as it increase the adversary user attack prospect hence by designing the proposed system we are extending the storage cloud security by distributing and encrypting the data. A web portal which let the user manage his data and the managed data should be split over the multiple cloud drive as a small part of file along with encryption. Proposed system will be tested and demonstrate over a local network or on live storage cloud server**.**

II. REFERENCES

[1] The Notorious Nine: Cloud Computing Top Threat in 2013,CSA.,Online athttps://cloudsecurityalliance.org/download/the-notorious-ninecloud-computing-top-threats-in-2013/. (Assess date: 15 July 2014).

[2] D. Boneh, G. di Crescenzo, R. Ostrovsky, and G. Persiano. Public keyencryption with keyword search. In Advances in CryptologyEUROCRYPT '04, vol. 3027 of Lecture Notes in ComputerScience,pages 506-522, 2004

[3] Y. Yang, H. Lu, and J. Weng. Multi-user private keyword search forcloud computing. In Proc. IEEE Third International Conference onCloudCom, vol. 29, pp. 264-271, 2011.

[4] D.X. Song, D. Wagner and A. Perrig, "Practical Techniques for Searcheson Encrypted Data," In Proc. IEEE Symp. Security and Privacy, pp. 44-55, 2000.

[5] Cao, N., Wang, C., Li, M., Ren, K., Lou, W.: Privacy-preservingMultikeyword Ranked Search over Encrypted Cloud Data. In: 30thIEEE Conference on Computer Communications, pp. 829-837, 2011.

[6] Craig Gentry, Fully homomorphic encryption using ideal lattices,Symposium on the Theory of Computing, pp. 169-178, 2009.

[7] Vandijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V. Fully homomorphic encryption over the integers, 2009. http://eprint.iacr.org/.2009.

[8] Halevi, Shai. "An Implementation of homomorphic encryption",GitHubRepository, https://github.com/shaih/HElib, 2013.

[9] ZvikaBrakerski and Craig Gentry and VinodVaikuntanathan, FullyHomomorphic Encryption without Bootstrapping , Cryptology ePrint Archive, Report 2011/277, http://eprint.iacr.org/, 2011 .

[10] Rivest, R.; A. Shamir; L. Adleman, "A Method for Obtaining DigitalSignatures and Public-Key Cryptosystems". Communications of theACM. Feb 1978, vol. 21, no. 2, pp. 120-126, 1978.

[11] P. Pallier. Public-key cryptosystems based on composite degreeresiduosity classes. In Proc. of Eurocrypt vol. 1592 of LNCS, pages223–238, 1999.

[12] Markus Jakobsson, Ari Juels, Addition of ElGamal Plaintexts, In Proc.of the 6th International Conference on the Theory and Application of Cryptology and Information Security & Advances in Cryptology, pp.346-358, 2000.