COMPUTER SCIENCE | RESEARCH ARTICLE

Verification of data integrity and co-operative loss recovery for secure data storage in cloud computing

Paul R Rejin¹* and Raj D Paul²

Abstract: In Cloud Computing, the data stored in the external servers may be tampered or deleted by unauthorized persons or selfish Cloud Service Providers (CSPs). Hence, the Cloud Data Owners (CDOs) have to provide assurance to the integrity and correctness of the stored data in the server. In this paper, a Verification of Data Integrity and Co-operative Loss Recovery technique for secure data storage in Cloud Computing is proposed. In this technique, a ciphertext file is split into various cipher blocks and distributed to randomly selected cloud service providers (CSPs) by the cloud data owner (CDO). If a cloud data user (CDU) wants to access any file, the corresponding ciphertext file is reconstructed from the blocks and downloaded by the user. The file can be decrypted if the attribute set of the user matches the access policy of the application. By simulation results, we show that the proposed technique enhances the data integrity and confidentiality.

Subjects: Physical Sciences; Computer Science; Engineering & Technology

Keywords: Cloud computing; data integrity; loss recovery; security; CDU; VA

ABOUT THE AUTHOR

Prof. Raj D Paul received his PhD degree in computer science and engineering from Anna University. Dr. Paulraj D has 20 years of experience including 9 years industrial experience. He has received his B.E (CSE) degree first class from Bangalore University in 1993, M.E (CSE) and Ph.D in Service Oriented Architecture from Anna University respectively in the year 2004 and 2012. During his Ph.D he has proved that Semantic Web Services can be composed using Process Model Ontology instead of Service Profile Ontology. He has published several research publications in refereed International Journals with high impact factor and International Conferences as well. He is the life member of ISTE and a member of IET. He had received IET Men Engineer Award in the year 2012. His research interests include Service Oriented Architecture (SOA), Semantic Web Services, Computer Network and Interface, Cloud and Grid Computing, Big Data Analytics.

PUBLIC INTEREST STATEMENT

This research investigates the security aspects of cloud data storage. Cloud computing offers customers a more flexible way to obtain computation and storage resources on demand. In this environment, customers can now rent the necessary resources as soon as, and as long as, they need. However, the data stored in the external servers may be tampered or deleted by unauthorized persons or the owner of the server. This research work designs a technique to protect the stored data from data loss and unauthorized access. In this technique, the data owner encodes a plain text file and splits it into many blocks. Each block is then distributed to randomly selected servers. If any part of the file is tampered, it can be detected by applying some verification techniques. By conducting experiments, it has been proved that the time involved in distributing and reconstructing the blocks is significantly less.
1. Introduction
The recent development of cloud computing has shown its potential to reshape the current way that IT hardware is designed and purchased. Cloud computing offers customers a more flexible way to obtain computation and storage resources on demand. Rather than owning (and maintaining) a large and expensive IT infrastructure, customers can now rent the necessary resources as soon as, and as long as, they need. It provides rich benefits to the cloud clients such as costless services, the elasticity of resources, easy access through the Internet, etc (Vurukonda & Thirumala Rao, 2016).

Cloud computing requires data security mechanisms which can prevent leakage and loss of user data. Different from the traditional computing model in which users have full control of data storage and computation, cloud computing entails that the management of physical data and machines are delegated to the CSPs (Wei et al., 2013). However, the CSPs are usually not trustworthy. They may conceal the data loss or error from the users for their own benefit. Even more, they might delete rarely accessed user data for conserving storage space (Cao, He, Guo, & Feng, 2016).

The traditional cryptographic technology cannot be implemented for cloud data security, as the users lose their control on data storage. Hence, it is required to verify the correctness of data stored in external storage. It needs to more advanced technology to avoid data loss from the cloud storage (El Mrabti, Ammari, & De Montfort, 2016) and (Chatterjee, Sarkar, & Dhaka, 2015).

1.1. Problem identification and objectives of the work
The main threats for data storage security are

(i) Modifying the stored data to compromise the data integrity
(ii) Unauthorized access of stored data
(iii) Loosing part of data or whole data.

Hence, existing solutions should address these threats for efficient storage security. A detailed discussion of existing solutions is presented in the next section. Based on the problems identified in those works, an efficient solution is required with the following objectives:

- The stored data should be protected from data loss and unauthorized access
- The integrity of data blocks should be ensured
- The confidentiality of data should be ensured.
- The storage cost should be reduced.

In order to meet the these objectives, Verification of Data Integrity and Co-operative Loss Recovery technique for secure data storage in Cloud Computing is proposed.

2. Related works
While the works (Cao et al., 2016) and (Zhu, Hu, Ahn, & Yu, 2012) deal with data integrity verification, they did not ensure recovery of lost blocks and confidentiality.

For ensuring the correctness of data and prevent data losses, the works (El Mrabti et al., 2016) and (Al-Anzi, Salman, & Jacob, 2014) split the encrypted data into various cipher blocks and distribute among different service providers. But the data blocks are distributed equally to all CSPs leading to a chance of fetching the blocks by any external adversary or malicious CSP in future. Moreover, the integrity of each block was not checked so that there may be a possibility of corrupted blocks.
A Steganographic Approach using Huffman Coding (SAHC) (Chatterjee et al., 2015) was applied to prevent data access by unauthorized users from cloud storage. In (Nikam & Potey., 2016), authorization is achieved in various levels such as providing static username-password as the entry-level authentication, followed by OTP based on token generator technique. Both the works (Chatterjee et al., 2015) and (Nikam & Potey., 2016) protect unauthorized access and ensure confidentiality, but result in huge storage overhead.

The privacy-preserving auditing protocol (Yang & Jia, 2012) handles frequent data updates and maintain consistency of data, but did not provide data confidentiality. In (Yin, Qin, Zhang, Ou, & Keqin, 2017) a secure, easily integrated, and fine-grained query results verification scheme for secure search over encrypted cloud data was explained. This scheme can verify the correctness of each encrypted query result or further accurately find out how many or which qualified data files are returned by the dishonest cloud server. A short signature technique is designed to guarantee the authenticity of the verification object itself.

In (Liu, Li, & Li, 2018), an iterative proximal algorithm (IPA) to compute a Nash equilibrium solution is proposed. The convergence of the IPA algorithm is also analyzed and we find that it converges to a Nash equilibrium. But in here the communication cost is not clearly analyzed. In (Liu, Li, Tang, & Li, 2018), a CA which can find the NBS very efficiently. For the general case, the author proposed an IA which is based on duality theory. The convergence of the IA algorithm is also analyzed. But here dynamic configuration of the multiple servers in cloud is not explained.

3. Verification of data integrity and co-operative loss recovery (VDI-CLR) technique

3.1. Overview
In this paper, a VDI-CLR technique for secure data storage in Cloud Computing is proposed. In this technique, CDO encrypts the original file using CP-ABE scheme and splits it into n/2 cipher blocks, where n denotes the number of CSPs. The cipher blocks are then distributed to randomly selected n/2 CSPs. In order to reconstruct the data blocks, the parity information P corresponding to these data blocks is stored in k CSPs, where k = n-n/2. Each block is represented by a block tag pair \((m_i, \beta_i)\) where \(\beta_i\) is a short signature tag of block \(m_i\) generated by a set of secrets. A probabilistic verification scheme is applied to prove the integrity of each block. If a CDU wants to access the file, the cipher file \(F'\) is reconstructed from the respective CSPs, after verifying the integrity of each block. Then, CDU decrypts the cipher file, if and only if its attribute set satisfies the access policy.

3.2. System model
The scheme consists of a Multiple Cloud Server Providers such as CSP_1, CSP_2, ..., CSP_n, Cloud Data Owner (CDO) and Cloud Data User (CDU). A Trusted Server (TS) is responsible for the system initialization and the generation of the master secret key. The Verification Server (VS) is used to verify the integrity of user data. Figure 1 shows the system model and their basic operations.

Figure 1 represents the proposed system model. Let \(F\) be the original data, the client wants to store in the cloud storage. The original data \(F\) is encrypted to form \(F'\). Then, \(F'\) is split into various cipher blocks and are stored in n/2 randomly chosen CSPs.

3.3. File fragmentation and distribution by CDO
The CDO encrypts the file, fragments and distributes it to randomly selected CSPs. The steps involved in this process are presented in the following algorithm.
Algorithm-1

Notations | Definition
--- | ---
$H: \{0,1\}^* \rightarrow G_0$ | Hash function
$n$ | Total number of CSPs
$(\text{CSP})_n$ | Set of all CSPs
$(\text{CSP})_r$ | set of randomly selected $n/2$ CSPs
$(\text{CSP})_k$ | set of $k$ CSPs such that $(\text{CDO})_k = (\text{CDO})_n - (\text{CDO})_r$
$k$ | 
$F_i$ | File $i \in W$, $i=1,2,\ldots$
$B_{ij}$ | Blocks of $F_i$, $j=1,2,\ldots,n/2$
$p_{ij}$ | Parity information of $B_{ij}$
$T_m$ | Secrets for block $B_{ij}$, $m=1,2,\ldots,k$
$\beta_{ij}$ | Signature tag of $B_{ij}$
$H_k(.)$ | Collision-resistant hash function
$\alpha_1, \alpha_2$ | two random numbers

1. For each $F_i \in W$,
2. CDO selects a file $F_i$ for uploading
3. CDO splits $F_i$ as $(B_{i1}, B_{i2}, \ldots, B_{in/2})$
4. Select random secrets $(T_{i1}, T_{i2}, \ldots, T_{ik})$ for $F_i$
5. Create a tag $\beta_{ij}$ for each $B_{ij}$ as
6. $\xi(1) = H(\sum_{k=1}^{k} T_m(F_i))$
7. $\xi(2) = H_1(\{C_g\})$
8. $\xi(3) = H_2(B_{ij})$
9. $u_m = g^{\alpha_k}$, $m=1,2,\ldots,k$
10. $\beta_{ij} = (\xi(3))^{\alpha_1} \cdot \left( \prod_{m=1}^{k} u_m^{\alpha_k} \right)$
11. Form the fragment structure as $(B_{i1}, \beta_{i1}), (B_{i2}, \beta_{i2}), \ldots, (B_{in/2}, \beta_{in/2})$
12. Form the set $(\text{CDO})_b$
13. For each CDO $j \in (\text{CDO})_r$
14. CDO uploads the block \( (B_{ij}, \beta_{ij}) \) to CSP \( j \)
15. End For
16. For each CDO \( j \) \( \in \) \( \{\text{CDO}\} \)
17. CDO stores the parity information \( P_{ij} \) to CDO \( j \)
18. End For
19. End For

For example, consider Figure 2. Suppose \( n = 4 \). Let the ciphertext File \( F_1 \) is split into 2 cipher blocks \( B_{11} \) and \( B_{12} \). Suppose data block \( B_{11} \) is stored in CSP \( 1 \) and data block \( B_{12} \) is stored in CSP \( 3 \). The parity information \( A \) associated with \( B_{11} \) is stored in CSP \( 2 \). Similarly, the parity information \( B \) associated with block \( B_{12} \) is stored in CSP \( 4 \). If any CDU request for the file \( F_1 \), it can be reconstructed using parity information \( A \) in CSP \( 2 \) and of parity \( B \) in CSP \( 4 \). This results in effective reconstruction of data with the help of this parity scheme, if a double disk failure occurs.

3.4. Integrity verification by VA
The VA applies a probabilistic verification method to prove the integrity of the blocks stored in various CSPs. The steps involves in this process are shown in the following algorithm.

**Algorithm-2**

| Notations | Definition |
|-----------|------------|
| CA        | Cloud administrator |
| F_REQ     | File request |
| V_REQ     | Verification request |
| \( (IP) \) | Challenge set of block index coefficient pairs |
| \( R_k \) | Response for the challenge from CSP \( k \), \( k=1,2 \ldots r \) |
| \( c_j \) | Co-efficient of block \( B_{ij} \) of file \( F_i \) |

1. If a F_REQ has been received from the CDU, then
2. CA initiates V_REQ to VA
3. VA computes the challenge
4. \( IP = (B_{ij}, c_j) \)
5. VA returns \( IP \) to CA
6. CA forwards \( IP \) to each CSP \( k \)
7. Each CSP \( k \) returns \( R_k \) to CA
8. If CA receives \( R_k \), then
9. CA aggregates \( (R_k, k=1,2 \ldots r) \) into \( R \) using homomorphic property.
10. CA transmits \( R \) to VA
11. End if
12. VA checks the response \( R \) and checks the integrity of blocks
13. If integrity is failed for any \( b_{ij} \) \( \in \) CSP \( k \), then
14. That CSP is considered as not trusted
15. End if
16. End if
3.5. Data access by CDU

The steps involved in this technique are as follows:

**Algorithm-3**

| Notations | Definition |
|-----------|------------|
| AP        | Access policy created by CDO |
| \( K_{\text{pub}}, K_{\text{pr}} \) | Public and private keys |
| Kms       | Master secret key |
| \( A_i \) | Attributes of CDU |
| \( F_r \) | Requested file by CDU |
| \{CSP\}_r | CSPs which stores \( F_r \) |
| \{CSP\}_n | Set of all CSPs |
| \{CSP\}_k | set of \( k \) CSPs such that \( \{CSP\}_k = \{CSP\}_n - \{CSP\}_r \) |

1. Start
2. For each file \( F_i \)
3. \( \text{CDO encrypts } F_i \) as
4. \( \text{Enc}(F_i) = \text{CP-ABE} \left( K_{\text{pub}}, F_i, AP \right) \)
5. \( \text{CDO applies fragmentation structure for } \text{Enc}(F_i) \) and distributes to \( \{CSP\}_r \)
   (as explained in Algorithm-1)
6. End For
7. CDU generates \( K_{\text{pr}} \) as
   \[ K_{\text{pr}} = \text{Keygen}(K_{\text{pub}}, K_{\text{ms}}, A_i) \]
8. CDU sends request for File \( F_r \) to CA
9. CA obtains the parity information from \( \{CSP\}_r \)
10. If CSP, found, then
11. \( \text{CSP}_r \) transmits \( b_{ij} \) of \( F_r \) to VA
12. End if
13. VA verifies the integrity of each blocks (using Algorithm-2)
14. If integrity of all the blocks are verified, then
15. \( \text{VA reconstructs the file } \text{Enc}(F_r) \)
16. Else
17. The corresponding block is considered as tampered
18. End if
19. CDU downloads Enc(F_r) from VA
20. If K_pr satisfies AP, then
21. Dec(F_r) = Decrypt (K_pub, Enc(F_r), K_pr)
22. End if
23. Stop

Initially, based on an access policy (AP), the CDO encrypts a file and creates a cipher text file F'. If a CDU wants to access the file F, it sends a request to the VA. VA reconstructs the cipher file F' after verifying the integrity of each block. The CDU downloads the cipher file F' and decrypts it, if and only if its attribute set satisfies AP. If CDO modifies the file F, it again creates an updated cipher file F'' and repeats the above process.

4. Experimental results
To validate the VDI-CLR technique presented in the previous section, we have implemented it in Java based on the CP-ABE toolkit and the Java Pairing-Based Cryptography library (JPBC) (De Caro & Iovino, 2011). To compare experimental results of the proposed scheme, we also simulate the typical CSS-FRS (El Mrabti et al., 2016) and RSSNS (Al-Anzi et al., 2014) schemes.

The following experiments are conducted by using Java on the system with Intel Core processor at 3.00 GHz and 4GB RAM running Windows 7 Ultimate. All the results are averages of 10 trials. As illustrated in (Yin et al., 2017), the complexity of the encryption and decryption in CP-ABE scheme can be measured from the two factors: (i) The time cost of encryption and decryption. (ii) The storage cost of ciphertext.

For example, assume that the patient shares three files, i.e., M = {m1,m2, m3}, with three access levels, the access policy is designed as {\(a_1, a_2, ..., a_i, i \text{ of } i\) AND \(a_{i+1}\) AND \(a_{i+2}\)) in FH-CP-ABE scheme. Accordingly, he should construct three access policies for CP-ABE scheme, where the policies are \{(a_1, a_2, ..., a_i, i \text{ of } i) \text{ AND } a_{i+1} \text{ AND } a_{i+2}\},

\{(a_1, a_2, ..., a_i, i \text{ of } i) \text{ AND } a_{i+1}\), and \((a_1, a_2, ..., a_i, i \text{ of } i)\). The policies only contain AND gate to ensure that all the ciphertext components are computed in the decryption algorithm.

4.1. Results for varying the number of files
The experimental results in this section are obtained by varying the number of files to be downloaded by the CDU. The number of attributes used in the weighted policy is fixed as 4. The average file size is fixed around 5Mb. The number of CSPs maintained is 4.

The time cost is measured in terms of the data distribution and reconstruction times. Data distribution time involves the encryption time, splitting time and distributing into various CSPs. Similarly, data reconstruction time involves the time for fetching the blocks from various CSPs, time for combining the blocks as per parity information and decryption time.

Figures 3 and 4 shows the results of data distribution and reconstruction times measured for files 1 to 5.

As it can be seen from Figure 3, the CSS-FRS scheme has the highest distribution time which increases from 10.15 s to 15.25 s when the number of files is increased from 1 to 5. The distribution time of RSSNS falls in the range of 11.45 s to 13.5 s. Since in both CCS-FRS and RSSNS, the data blocks are distributed equally to all CSPs, they need higher distribution time. The proposed VDI-CLR scheme attains the lowest distribution time in the range of 8.05 to 10.2 s, since it replicates only in half of the available number of CSPs.
Hence, the percentage improvement of VDI-CLR over CSS-FRS and RSSNS schemes are 31.1% and 26.9%, respectively.

As it can be seen from Figure 4, the CSS-FRS scheme has the highest data reconstruction time which increases from 12.15 s to 17.45 s when the number of files is increased. But the reconstruction time of RSSNS falls in the range of 13.45 s to 15.57 s. Since in both CSS-FRS and RSSNS, the data blocks are distributed equally to all CSPs, they need higher reconstruction time. The proposed VDI-CLR scheme attains the lowest reconstruction time in the range of 8.25 to 11.21 s, since it can fetch the parity information from limited CSPs and need to obtain the blocks from half of the CSPs. Hence, the percentage improvement of VDI-CLR over CSS-FRS and RSSNS schemes are 36.2% and 32.1%, respectively.

The storage cost for the cipher text is measured in terms of the size of the encrypted files. Figure 5 shows the results of storage cost measured for files 1 to 5.

As it can be seen from Figure 5, the CCS-FRS scheme has the highest storage cost which increases from 2750 KB to 10480 KB when the number of files is increased. The storage cost of RSSNS falls in the range of 2175 KB to 9284 KB. However, the proposed VDI-CLR scheme attains the lowest storage cost in the range of 1270 KB to 6870 KB, since it stores the replicated data in half of the total available CSPs. Hence, the percentage improvement of VDI-CLR over CCS-FRS and RSSNS schemes are 45% and 37%, respectively.

Figure 6 shows the correctness of downloaded data at the CDUs. As it can be seen from the figure, the correctness of data degrades slightly for all the techniques, when the number of files is increased. Due to the integrity verification of each blocks, VDI-CLR has 10% and 7% increased correctness when compared to CCS-FRS and RSSNS techniques.
4.2. Results for varying file size

The experimental results in this section are obtained by varying the size of the files to be downloaded by the CDU. The number of attributes used in the weighted policy is fixed as 4. The average number of files is fixed as two files.

The time cost is measured in terms of the data distribution and reconstruction times. Figures 7 and 8 shows the results of data distribution and reconstruction times measured for file sizes from 5 to 25 MB.

As it can be seen from Figure 7, the RSSNS scheme has the highest distribution time which increases from 11.45 s to 12.9 s when the file size is increased. But the distribution time of RSSNS falls in the range of 10.15 s to 12.11 s only. However, the proposed VDI-CLR scheme attains the lowest distribution time in the range of 8.05 to 9.36 s, since it replicates only in half of the available number of CSPs.
Hence, the percentage improvement of VDI-CLR over CSS-FRS and RSSNS schemes are 20% and 27%, respectively.

As it can be seen from Figure 8, the RSSNS scheme has the highest data reconstruction time which increases from 13.45 s to 15.23 s when the file size is increased. The reconstruction time of CSS-FRS falls in the range of 12.15 s to 14.22 s only. However, the proposed VD-CLR scheme attains the lowest data reconstruction time in the range of 8.25 to 9.88 s, since it can fetch the parity information from limited CSPs and need to obtain the blocks from half of the CSPs. Hence, the percentage improvement of VDI-CLR over CSS-FRS and RSSNS schemes are 29% and 34%, respectively.

The storage cost for the ciphertext is measured in terms of the size of the encrypted files. Figure 9 shows the results of storage cost measured for file sizes 5 to 25 MB.

As it can be seen from Figure 8, the CCS-FRS scheme has the highest storage cost which increases from 2750 KB to 5380 KB when the file size is increased. The storage cost of RSSNS falls in the range of 2175 KB to 4394 KB. However, the proposed VDI-CLR scheme attains the lowest storage cost in the range of 1270 KB to 3770 KB, since it stores the replicated data in half of the total available CSPs. Hence, the percentage improvement of VDI-CLR over CSS-FRS and RSSNS schemes are 42% and 25%, respectively.

5. Conclusion and future work
In this paper, we have proposed a Verification of Data Integrity and Co-operative Loss Recovery (VDI-CLR) technique for secure data storage in Cloud Computing is proposed. In this technique, a ciphertext file is split into various cipher blocks and distributed to randomly selected cloud service providers (CSPs) by the cloud data owner (CDO). If a cloud data user (CDU) wants to access any file, the corresponding ciphertext file is reconstructed from the blocks and downloaded by the user. The proposed VDI-CLR technique has been compared with CCS-FRS and RSSNS techniques. The performance is measured in terms of data distribution time, data reconstruction time, the correctness of data and storage cost, for a different number of files and different file sizes.
the performance results, it can be concluded that VDI-CLR has the least data distribution and reconstruction time, least storage overhead along with improved correctness of data.

In cloud computing environment, both the cloud users and service providers should have a trusted relationship. But most of the works consider trusts of cloud users alone, failing to address the trust worthiness of cloud servers. In order to solve the above-mentioned issues, the future work aims to design a trust-based access control framework for a multi-cloud environment.

Funding
The authors received no direct funding for this research.

Author details
Paul R Rejin
E-mail: rejinpaulphd@gmail.com
Raj D Paul
E-mail: kingrajpaul@gmail.com
1 Department of Computer Science and Engineering, RMK College of Engineering and Technology, Chennai, India.
2 Department of Computer Science and Engineering, RMD Engineering College, Chennai, Tamil Nadu, India.

Citation information
Cite this article as: Verification of data integrity and co-operative loss recovery for secure data storage in cloud computing, Paul R Rejin & Raj D Paul, Cogent Engineering (2019), 6: 1654694.

References
Al-Anzi, F. S., Salman, A. A., & Jacob, N. K. (2014). New proposed robust, scalable and secure network cloud computing storage architecture. Journal of Software Engineering and Applications, 7, 347–353. doi:10.4236/jsea.2014.75031
Cao, L., He, W., Guo, X., & Feng, T. (2016). A scheme for verification on data integrity in mobile multicloud computing environment. Mathematical Problems in Engineering, 6. Article ID 9267608.
Chatterjee, T., Sarkar, M. K., & Dhaka, V. S. (2015, April). Steganographic approach to ensure data storage security in cloud computing using Huffman Coding (SAHC). IJCNS International Journal of Computer Science and Network, 4(2): 198–207.
De Caro, A., & Iovino, V. (2011). jPBC: Java pairing based cryptography. In Proc. IEEE Symp. Comput. Commun. (pp. 850–855) (Greece). June-July.
El Mrabti, A. A., Ammari, N., & De Montfort, M. (2016). New mechanism for cloud computing storage security. (IJACSA) International Journal of Advanced Computer Science and Applications, 7(7): 526–539.
Liu, C., Li, K., & Li, K. (2018). A game approach to multi-servers load balancing with load-dependent server availability consideration. IEEE Transactions on Cloud Computing, 1. (Early Access), doi: 10.1109/TCC.2018.2790404
Liu, C., Li, K., Tang, Z., & Li, K. (2018). Bargaining game-based scheduling for performance guarantees in cloud computing. ACM Transactions on Modeling and Performance Evaluation of Computing Systems, 3 (1), 1: 1–1.25. doi:10.1145/3141233
Nikam, R., & Potey, M. (2016). CLOUD STORAGE SECURITY USING MULTI-FACTOR AUTHENTICATION. IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2016), December 23-25, Jaipur, India.
Vurukonda, N., & Thirumala Rao, B. (2016). A study on data storage security issues in cloud computing. Elsevier, Procedia Computer Science, 92, 128–135. doi:10.1016/j.procs.2016.07.335
Wei, L., Zhu, H., Cao, Z., Dong, X., Jia, W., Chen, Y., & Vasilakos, A. V. (2013). Security and privacy for storage and computation in cloud computing. Elsevier, Information Science, 258(10):371–386
Yang, K., & Jia, X. (2012). An efficient and secure dynamic auditing protocol for data storage in cloud computing. IEEE Transactions on Parallel and Distributed Systems, 24(9): 1717–1726
Yin, H., Qin, Z., Zhang, J., Ou, L., & Keqin, L. (2017). Achieving secure, universal, and fine-grained query results verification for secure search scheme over encrypted cloud data. In IEEE transactions on cloud computing (Vols. 1-1, pp. 99). (Early Access), doi: 10.1109/TCC.2017.2709318
Zhu, Y., Hu, H., Ahn, G.-J., & Yu, M. (2012, DECEMBER). Cooperative provable data possession for integrity verification in multicloud storage. IEEE Transactions on Parallel and Distributed Systems, 23(12), doi:10.1109/TPDS.2012.66
