

A Security Proxy Scheme Based on Attribute Node Mapping for Cloud Storage

Huakang Li, Zhenyu Wang, Yitao Yang, and Guozi Sun^(✉)

School of Computer Science and Technology, School of Software,
Institute of Computer Technology,
Nanjing University of Posts and Telecommunications, Nanjing 210023, China
{huakanglee,sun}@njupt.edu.cn

Abstract. Cloud storage provides convenient storage services with data leaking risk while the encryption and decryption keys are supported by cloud service. However, the traditional CP-ABE scheme cannot solve the problem of integrity of cloud service provider according to single attributes rules. In this paper, we design a prototype system for secure cloud storage which separates storage services and security service using Attribute node mapping based on CP-ABE scheme. The prototype system consists of four parts: a client, a key generation center, a security proxy and a storage system. We propose an innovative convergence encryption method and a shared access mechanism to improve the encryption against guessing attack. Hierarchical eliminate redundancy and parallel data access technologies are also proposed improving the data transmission efficiency.

Keywords: Cloud storage · Access control · Attribute-based Encryption · CP-ABE · Node mapping

1 Introduction

With the development of Internet and distributed computing in recent years, Cloud Computing has become an important technology for shared softwares and hardware resources. Cloud storage service is the most common and popular service (e.g. Google Drive, Dropbox, Huawei Cloud) for typical users. The bottleneck of limited storage space has become more and more significant, especially for mobile users while they take lots of pictures and videos.

Different with super computing system, inexpensive commodities are commonly used in cloud system due to the consideration of scalability [6]. The reliability issue of these systems is of particular relevance. To ensure the data reliability, redundancy scheme is a basic solution and has been extensively deployed [8]. With this scheme, the intuitive idea is to store copies of data objects over a set of network nodes for the successful recovery. At the same time, the cloud service provider could remove the extra redundancy data copies from data storage.

We also bear the risk of cloud storage, such as efficacy and security [7, 11] while we enjoy the convenience of cloud storage. One problem of cloud service

is verifying the integrity of the data since users cannot know how to handle their data in the cloud storage. Many researches introduced various systems and security models [1, 10, 16] to solve the problems of data integrity verification in the cloud storage. Private auditability [19], which is one efficient verifier, implies the data owner directly verifying data in the cloud storage service. However, the data owner could not verify the data frequently depending on the business requirements and spare time. Therefore, public auditability [20], which is another important verifier, implies the data owner allowing others to verify the owned data.

According to the basic requirements of security and performance, the existing scheme of security evaluation can be classified as follows:

- Blockless Verification: User can modify the data blocks to avoid retrieving all audited data blocks in cloud storage.
- Batch Auditing: User can verify the data from different client at the same time with a special token.
- Dynamic Data: The data can be continuously modified by competent users.
- Privacy Presenting: User can't access the delegated data in the cloud storage service.

In this paper, we propose a scheme for cloud storage combined with the security proxy and stochastic storage strategy to separate storage service and security service. The prototype system contains four parts: a client, a key generation center, a security proxy and a storage system. In the uploading process, client cuts the file into blocks with fixed size to calculate the hash fingerprints and sent to the security proxy. The security proxy compares the hash fingerprint to establish whether the data are redundant. Secret keys and random storage tables generated by the hash value and partial quantity value, are sent back to the client. The client uses the secret key to encrypt the data and upload to cloud storage with random storage table. In the downloading process, users need to pass the verification of access structure tree to achieve the secret key for decoding and random storage table when they are required to access the data. The client accesses the storage nodes according to the random storage table to decode the data block and reconstruct the data after the legitimate authentication. The method proposed under this paper solves the contradiction between data encryption and data redundancy. At the same time, it also can prevent the illegal use and data privacy from cloud storage service providers.

The article consists of the following parts: Sect. 2 introduces the related works of data encryption and data redundancy. The system design and improved ABE scheme are introduced in Sect. 3. Section 4 shows the access control scheme design. The experimental results of system performance are illustrated in Sect. 5. Section 6 concludes the main jobs and feature works.

2 Related Work

2.1 Data Redundancy

In cloud storage system, it's very straightforward to use redundancy scheme to achieve data reliability. Distributed data copy is the commonly used method to cope with data failure or missing. Research communities [14, 23] introduce redundancy schemes for the system performance. Generally, redundancy scheme can be classified into two types: replication [21] and erasure code [13].

Cloud storage service providers utilize data redundancy to ensure the reliability of data while they hope to reduce the repeated data copies to save the storage costs. Whole File Detection (WFD) technologies [9] use the hash value of the whole file to estimate the comparing fingerprint to implement the data repetition. Fixed-sized partition (FSP) [3] cuts the files into data blocks with fixed size to calculate the hash fingerprint. Content-defined chunking method [17] uses a dynamic sliding window to calculate the Robin fingerprint value. Sliding block method [12] uses Rsync sum function and fixed block sliding window to calculate the calibration value of cross data block in the file. Comparable data detection technology contrasts data one by one to eliminate duplicate data in the system. However, the high computational complexity problem is existed for these methods.

2.2 Data Encryption

Wang et al. [20] proposed the scheme to support public verification and fully dynamic data instead of modifying or deleting data files. The definition of public auditability which implies public verification is delegated by a trusted third party auditor (TPA) to verify. Li et al. [10] proposed a public auditability scheme in resource-constrained devices using third party auditor for data uploading and audit delegating. After that Wang et al. [19] proposed a privacy protection scheme which is considered user's data privacy in the public auditability.

Attribute-based Encryption (ABE), which is one of public key encryption systems, is proposed by Sahai [15] firstly. This method is based on fuzzy identity-based encryption and can achieve the fine-grained access control issues. ABE uses the user access policies set of users attributes and data together. The system enables users to access data only if users access property structures match the access control policy. It is ideal for cloud storage that data are shared among users. In the cloud area, many researchers [18, 24, 25] have applied the ABE to achieve a more fine-grained access control and data sharing goals.

In recent years, researchers have proposed a number of ABE schemes. Waters [22] and Daza [4] proposed ABE schemes independently, whose cipher text lengths are $n + (1)$ and $2(n - t) + (1)$, using threshold-based access control policy. However, effects of these ABE schemes are very low for mobile agents. A fixed length of the cipher text ABE encryption scheme [5] was proposed while the users private key attribute must be fully consistent. This significant limitation made the established policy properties cipher text cannot be widely promoted.

Bethencourt [2] proposed the Cipher Text-policy Attribute-based Encryption scheme (CP-ABE). The attribute set is a direct result of the user's private key, and access structure is related with the cipher text. If a user's set of attributes satisfy the cipher text access structure tree, the user can decrypt the cipher text. The sender can set the access structure to identify which users can access the appropriate data cipher text. This type of CP-ABE scheme is ideal for a distributed and share-based computing environment, especially for cloud storage.

3 Design of Cloud CP-ABE Scheme

3.1 Design of System Model

When the CP-ABE scheme is used in secure cloud storage environment, one problem is the fundamental structure of CP-ABE scheme supports attribute sets only constructed by single property in accordance of a certain number of rules. And it does not support the attributes of third-party, such as authorization center. Therefore we designed a new CP-ABE scheme (Fig. 1) for cloud storage with Key Generation Center and Security Proxy.

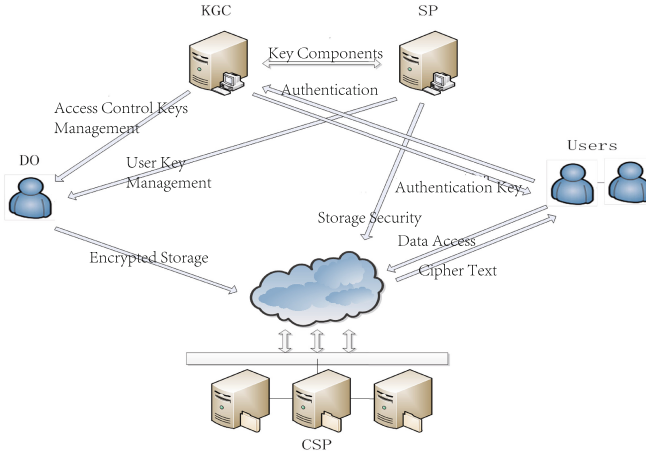


Fig. 1. System structure of cloud CP-ABE scheme.

- **Key Generation Center (KGC):** KGC is mainly responsible for the system to generate public and private keys. KGC is also responsible for the distribution of their corresponding properties of the component for different users with different access rights. In our work, KGC is deemed to be semi-credible (honest but curious) that the KGC will analyze users private information beside the default services.

- Security proxy (SP): SP is designed to separate the users security services from the cloud storage. It is responsible to store the fingerprint database and distribute the secret attributes. Also, SP is semi-credible.
- Cloud storage provider (CSP): CSP provides storage services for users, and control the data access according to the authentication of private key structure from users. Also, CSP is semi-credible.
- Data owner (DO): In order to reduce storage costs, data owners use the storage service from CSP, and upload their private data. DO are responsible for defining access control policy, and encrypt data before uploading to prevent illegal use of CSP.
- User: Users can get the data from CSP. If one user satisfies the access control policy with attribute structure, he can access the shared resources.

3.2 Sharing Degree-Based Authentication

We suppose that one file F is divided into N shared pieces stored on the cloud server. The server would reduce the redundancy blocks according to the data repetition and build an Access Structure Tree (AST) (Fig. 2) based on historic access frequency. Therefore, we have the definition as follows:

Definition 1: Sharing Degree (SD): If each piece data is shared by several documents, the SD can be estimated with the deeps of AST. The SD of leaf nodes which are on the bottom lay is $[1\ 10]$, and the root node has the largest SD, such as $(1000\ \infty)$;

Definition 2: Children Relationship (CR): CR presents the relationship between child nodes data blocks. Therefore, the CR of a leaf node amounts to

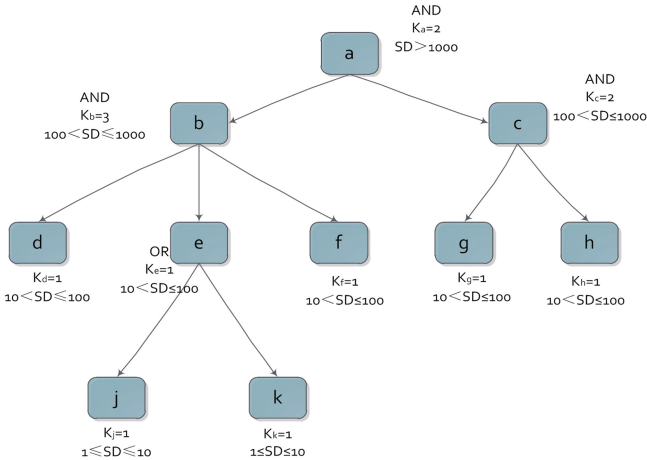


Fig. 2. A simple structure of sharing degree-based access tree.

Null. The CR of other nodes is defined as *AND* or *OR* or n/m , where m is the number of child nodes and $1 \leq n \leq m$.

Definition 3: Threshold Value (TV): TV is used to quantify the relation between child nodes. Therefore the TV (k_x) of a leaf node is 1 and the k_x of other nodes is $0 \leq k_x \leq m$.

When the owner gains access to the file F , the client applies the access structure λ which matches the access tree T generated by security proxy. In order to meet $T(\lambda) = 1$, we should satisfy the condition that $T_x(\lambda) = 1$ ($x = 1, \dots, m$), where T_x is the sub-tree of the AST.

3.3 ABE Scheme with SA

In order to import the CP for key management, we add Secret Attribute (SA) into the set for attribute keys (Fig. 3). Each user includes this property, and the values are very different for different data. Root of access structure tree must be AND gate, and the child node of the root must be a mapping node which including expiration and secret attributes. The operation (such as attribute addition and deletion) of attribute sets is not contained by these two attributes. So adversary cannot have all the users private key when update the key regardless of in cloud server of third-party security proxy.

3.4 Mapping Node

For the mapping node, a mapping function $e : \{SA, E\} \rightarrow \{\rho\lambda\}$ with Expiration (E) and Secret Attribute (SA). λ is SA set submitted by user. And the new key structure $S^{new} \{\rho\lambda, \rho i, \rho j, \rho k, \dots\}$, $\rho\lambda \in U - S$ is generated synchronously by the key generation center.

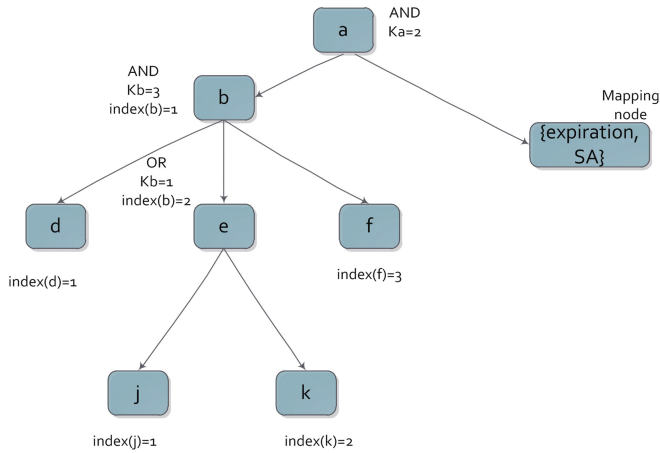


Fig. 3. A simple structure of access structure tree with SA.

Table 1. The Symbol descriptions of Cloud CP-ABE

Symbol	Description
KGC	Key generation center
SP	Security proxy
SA	Security attribute
EXPIRATION	Expiration attribute
PK	Public key
MK	Main key
SK	Private key
U	Attribute set
M	User data
CT	Cipher text

For the cloud service, if x is a leaf node, and key attribute set $|x| \in S^{new}$, $T_x(\lambda)$ will return value 1. If x is a non-leaf node, the value of $T_x(\lambda)$ will be calculated with its child nodes $\{y\}$. If x is a mapping node and the attribute of x satisfies $S_x \neq \phi$, S_x will be transformed to $\rho\lambda$ with the mapping function e . The mapping node becomes a leaf node and performs the leaf node matching operation.

4 Access Control Design

To accommodate de-duplication technology in cloud storage and reduce the calculate pressure of re-encryption by security agents. We firstly use the file division to cut files into a number of blocks of fixed size. The hash value and convergent encryption of each block was computed. The key generation center (KGC) assigns the attribute keys to n . Security proxy (SP) is responsible for allocating the confidential attributes (SA) and expiration attributes (E). The main symbols for Cloud CP-ABE can be presented as Table 1.

4.1 System Initialization

Assuming that q is the initial prime number for encryption algorithm and Z_p^* is a collection of a finite field. For any $i \in Z_p^*$, and $a \in Z_p^*$ (a is in set S), the definition of the Lagrangian Parameter $\Delta_{i,s}$ is as bellows:

$$\Delta_{i,s} = \prod_{j \in S, j \neq i} \frac{x - j}{i - j} \quad (1)$$

The bilinear group $\{G_x\}$ are generated by security parameters with generator g . We can define the bilinear mapping function $e : G_1 \times G_2 \rightarrow G_2$ while the pseudo-random function can be defined as:

$$\Upsilon(x) = g_2^{x^n} \prod_{i=1}^{n+1} t_i^{\Delta_{i,s}} \quad (2)$$

Therefore, Υ can be simplified as $g_2^{x^n} g^{h(x)}$, where $h(x)$ is a polynomial of degree n .

Attribute set U can be defined as $U = \{u_1, u_2, u_3, \dots, u_n, sa*, expiration\}$. Here $sa*$ is secret attribute associated with a specific data block, $expiration$ is key attribute generated by security proxy. For any property $u_i \in U$ is associated with $\Upsilon(u_i)$. Randomly selected $y \in Z_p^*$, the system public key parameter PK is generated as:

$$T_1 = g^{t_1}, \dots, T|u| = g^{t|u|}, Y = e(g, g)y \quad (3)$$

here, the main system secret key MK is $\{s_i : t_1, \dots, t|u|, y\}$.

4.2 Encryption Algorithm

The encryption algorithm proceeds from the root node r of the access tree T . We choose a polynomial P_x for each node x from root to leaf. For the root node, $P_r(0) = s$ where $s \in Z_p^*$ is randomly selected. For the non-leaf node, $P_x(0) = P_{parent}(x)(index(x))$. The final cipher text (CT) can be written as:

$$CT = \{M \cdot e(g, g)\alpha s, C = hs, \forall y \in L, C_y = H(|y|)P_y(0)\} \quad (4)$$

where M is input data, α is the source unit in Z_p^* , and L is the set of all leaf nodes of AST.

4.3 Authentication

When user accesses file F , security proxy will extract d ($d \leq f$) data blocks randomly to generate the access control tree Π . Here the original file F is divided into f blocks. The user must provide the full attribute set Π , otherwise its an illegal access from the current user.

4.4 Private Key Generation

For the PKG, the users private key SK is generated by attribute set U , primary key MK and public parameters PK .

$$SK = (D = g(\alpha + \gamma)/\beta, \forall j \in S : D_j = g\gamma \cdot H(j), D_j = g\gamma j) \quad (5)$$

here $\alpha, \beta, \gamma \in Z_p^*$ and $j \in S$ are selected randomly.

4.5 Decryption of Cipher Text

If and only if the attribute set meets the access tree, cipher text can be decrypted to plain-text. For the leaf-node x , we use $i = |x|$ and $i \in S$ to calculate as follows,

$$\begin{aligned} Decrypt(CT, SK, x) &= \frac{e(D_i, C_x)}{e(D'_i, C'_x)} \\ &= \frac{e(g^\gamma \cdot H(i)^{\gamma_i}, g^{P_x(0)})}{e(g^{\gamma_i}, H(i)^{P_x(0)})} \\ &= e(g, g)^{\gamma \cdot P_x(0)} \end{aligned} \quad (6)$$

If $i \notin S$, $Decrypt(CT, SK, x) = \perp$. For the non-leaf node x , we can use the return value of F_λ from its child node λ , then recursively calculate the F_x by polynomial interpolation:

$$F_x = \sum_{\lambda \in S_x} F_\lambda^{\Delta_{i,S_x}(0)} = e(g, g)^{s, P_x(0)} \quad (7)$$

S_x is a set of k_x child nodes, which make $F_\lambda \neq 0$, and $i = index(\lambda)$, $S'_x = \{index(\lambda) : \lambda \in S\}$, and

$$\Delta_{i,s}(x) = \prod_{j \in s, j \neq i} \frac{x-j}{i-j} \quad (8)$$

According to this method, we can recursive the root node to restore the blind factor

$$A = e(g, g)^{\gamma q_\gamma(0)} = e(g, g)^{\gamma^S} \quad (9)$$

Finally, plain text can be decrypted from the cipher text with:

$$C * / (e(C, D)/A) = C * / (e(h^S, g^{\alpha+\gamma/\beta})/e(g, g)^{\gamma^S}) = M \quad (10)$$

5 System Performance

The confidentiality of access structure and data in this paper can be evidenced by the security issue of cipher text of encryption key according to symmetric key encryption algorithm (such as DES, AES, etc.). Therefore, in this section we just discuss the time cost for system performance. Table 2 shows that we used a computer with 2.5 GHz CPU and 4G Memory. The system is Ubuntu 12.04 with JDK 1.7. We used the standard library PCB-0.5.14 from Stanford University. The encrypt data are generated randomly with [20 50] child nodes. The number of users attribute set is 10 uniformly. We calculated attribute set using KEK function. The finite field was set at 512, and 160 bit elliptic curve functions ($y^2 = x^3 + x$) for decryption were used from PBC library.

Table 2. System parameters for experiments

System environment		Experiment parameters	
CUP	2.5G	leaf nodes	[20 50]
Memory	4G	Attribute set	10
System	Ubuntu 12.04	$SK_{Eq.5}$	KEK
SDK	JDK 1.7.0	Z_p^*	512
Lib	PBC-0.5.14	$Decrypt_{Eq.6}$	$\{y^2 = x^3 + x\}$

Figures 4 and 5 show the encryption and decryption times with CP-ABE and our Cloud CP-ABE algorithm. Encryption times of the two schemes are significantly linear relationship with leaf nodes. The average time of our scheme

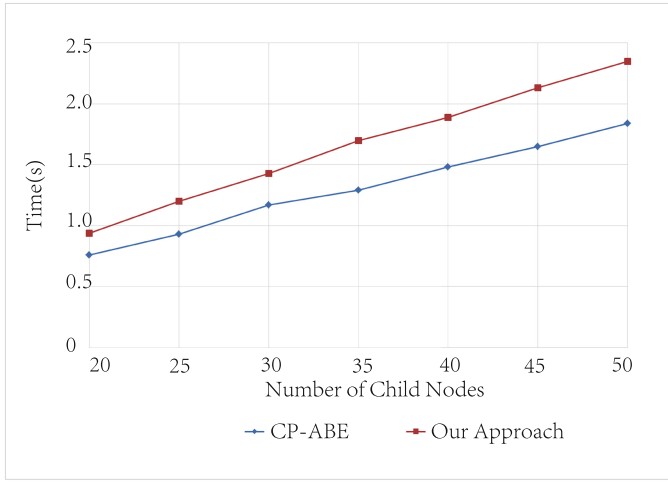


Fig. 4. Encryption time results for CP-ABE and our approach.

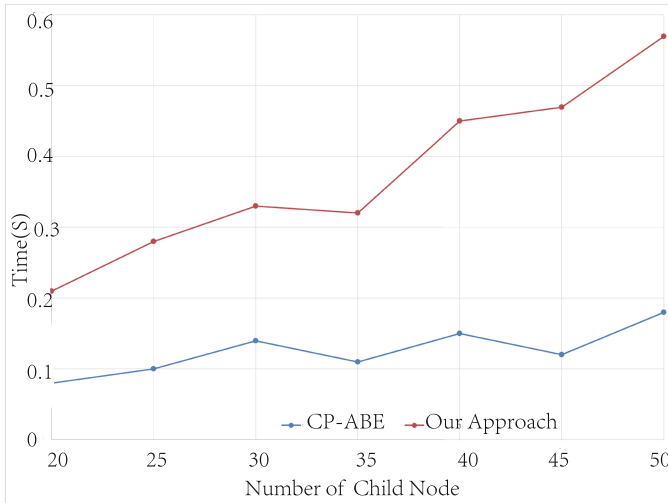


Fig. 5. Decryption time results for CP-ABE and our approach.

is 0.36 s more than the basic CP-ABE. The average time consuming of the basic CP-ABE program is 0.126 s, and the average time for our scheme is 0.376 s for decryption. Compared with the basic CP-ABE scheme, the time consuming is added within the acceptable range.

6 Conclusion

In this paper, we proposed an attributed-based access control model for the encryption scheme. The prototype system consists of four parts: a client, a key generation center, a security proxy and a storage system. Based on the traditional CP-ABE scheme, a de-duplication function, which makes access control tree of CP-ABE scheme more expressive, to solve defection that the user attribute sets must come from the user. The experimental results illustrated that hierarchical eliminate redundancy and parallel data access technologies were in a position to improve the data transmission efficiency. However, all of our work is based on the cloud storage providers and security agents are separated. In the future, we could consider the mutual authentication mechanisms among user, agent and cloud service provider to make the cloud storage more secure.

Acknowledgments. This work was supported by the Foundation of Nanjing University of Posts and Telecommunications (Grant No. NY213085 and No. NY214069), the NSFC (No. 61502247, 11501302, 61502243), Natural Science Foundation of Jiangsu Province (BK20140895, BK20130417).

References

1. Ateniese, G., Di Pietro, R., Mancini, L.V., Tsudik, G.: Scalable and efficient provable data possession. In: Proceedings of the 4th International Conference on Security and Privacy in Communication Networks, p. 9. ACM (2008)
2. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, SP 2007, pp. 321–334. IEEE (2007)
3. Bobbarjung, D.R., Jagannathan, S., Dubnicki, C.: Improving duplicate elimination in storage systems. *ACM Trans. Storage (TOS)* **2**(4), 424–448 (2006)
4. Daza, V., Herranz, J., Morillo, P., Ràfols, C.: Extensions of access structures and their cryptographic applications. *Appl. Algebra Eng. Commun. Comput.* **21**(4), 257–284 (2010)
5. Emura, K., Miyaji, A., Nomura, A., Omote, K., Soshi, M.: A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In: Bao, F., Li, H., Wang, G. (eds.) *ISPEC 2009*. LNCS, vol. 5451, pp. 13–23. Springer, Heidelberg (2009). doi:[10.1007/978-3-642-00843-6_2](https://doi.org/10.1007/978-3-642-00843-6_2)
6. Ford, D., Labelle, F., Popovici, F.I., Stokely, M., Truong, V.A., Barroso, L., Grimes, C., Quinlan, S.: Availability in globally distributed storage systems. In: *OSDI*, pp. 61–74 (2010)
7. Hashem, I.A.T., Yaqoob, I., Anuar, N.B., Mokhtar, S., Gani, A., Khan, S.U.: The rise of big data on cloud computing: review and open research issues. *Inf. Syst.* **47**, 98–115 (2015)
8. Hwang, G.H., Lin, H.F., Sy, C.C., Chang, C.Y., et al.: The design and implementation of appointed file prefetching for distributed file systems. *J. Res. Pract. Inf. Technol.* **40**(2), 91 (2008)
9. Khasnabish, B., Jin, W., Li, M.: Content de-duplication for CDNI optimization. Internet-Draft (2013)

10. Li, J., Tan, X., Chen, X., Wong, D., Xhafa, F.: OPoR: enabling proof of retrievability in cloud computing with resource-constrained devices. *IEEE Trans. Cloud Comput.* **3**(2), 195–205 (2015)
11. Liu, C., Yang, C., Zhang, X., Chen, J.: External integrity verification for outsourced big data in cloud and IoT: a big picture. *Future Gener. Comput. Syst.* **49**, 58–67 (2015)
12. Policroniades, C., Pratt, I.: Alternatives for detecting redundancy in storage systems data. In: *USENIX Annual Technical Conference, General Track*, pp. 73–86 (2004)
13. Reed, I.S., Solomon, G.: Polynomial codes over certain finite fields. *J. Soc. Ind. Appl. Math.* **8**(2), 300–304 (1960)
14. Rodrigues, R., Liskov, B.: High availability in DHTs: erasure coding vs. replication. In: Castro, M., Renesse, R. (eds.) *IPTPS 2005. LNCS*, vol. 3640, pp. 226–239. Springer, Heidelberg (2005). doi:[10.1007/11558989_21](https://doi.org/10.1007/11558989_21)
15. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) *EUROCRYPT 2005. LNCS*, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). doi:[10.1007/11426639_27](https://doi.org/10.1007/11426639_27)
16. Singh, R., Kumar, S., Agrahari, S.K.: Ensuring data storage security in cloud computing. *Int. J. Eng. Comput.* **2**(12), 17–21 (2012)
17. Ungureanu, C., Atkin, B., Aranya, A., Gokhale, S., Rago, S., Calkowski, G., Dubnicki, C., Bohra, A.: Hydrafis: a high-throughput file system for the hydrastor content-addressable storage system. In: *FAST*, pp. 225–238 (2010)
18. Wan, Z., Liu, J., Deng, R.H.: Hasbe: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE Trans. Inf. Forensics Secur.* **7**(2), 743–754 (2012)
19. Wang, C., Wang, Q., Ren, K., Lou, W.: Privacy-preserving public auditing for data storage security in cloud computing. In: *2010 IEEE Proceedings of INFOCOM*, pp. 1–9. IEEE (2010)
20. Wang, Q., Wang, C., Ren, K., Lou, W., Li, J.: Enabling public auditability and data dynamics for storage security in cloud computing. *IEEE Trans. Parallel Distrib. Syst.* **22**(5), 847–859 (2011)
21. Wang, Y., Li, S.: Research and performance evaluation of data replication technology in distributed storage systems. *Comput. Math. Appl.* **51**(11), 1625–1632 (2006)
22. Waters, B.: Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) *PKC 2011. LNCS*, vol. 6571, pp. 53–70. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-19379-8_4](https://doi.org/10.1007/978-3-642-19379-8_4)
23. Weatherspoon, H., Kubiatowicz, J.D.: Erasure coding vs. replication: a quantitative comparison. In: Druschel, P., Kaashoek, F., Rowstron, A. (eds.) *IPTPS 2002. LNCS*, vol. 2429, pp. 328–337. Springer, Heidelberg (2002). doi:[10.1007/3-540-45748-8_31](https://doi.org/10.1007/3-540-45748-8_31)
24. Yang, K., Jia, X., Ren, K.: Attribute-based fine-grained access control with efficient revocation in cloud storage systems. In: *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, pp. 523–528. ACM (2013)
25. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving secure, scalable, and fine-grained data access control in cloud computing. In: *2010 IEEE Proceedings INFOCOM*, pp. 1–9. IEEE (2010)

Security, Privacy, and Anonymity in Computation,
Communication, and Storage
9th International Conference, SpaCCS 2016,
Zhangjiajie, China, November 16-18, 2016, Proceedings
Wang, G.; Indrakshi, R.; Alcaraz Calero, J.M.; Thampi,
S.M. (Eds.)
2016, XVI, 508 p. 149 illus., Softcover
ISBN: 978-3-319-49147-9