Data Verification Using Block Level Batch Auditing on Multi-cloud Server

G. L. Prakash, Manish Prateek, and Inder Singh
(Corresponding author: G. L. Prakash)

School of Computer Science and Engineering, University of Petroleum and Energy Studies
Bidholi, Via Prem Nagar, Dehradun, Uttarakhand 248007, India
(Email: prakashgl@ddn.upes.ac.in)
(Received Mar. 20, 2017; revised and accepted June 26, 2017)

Abstract

In cloud computing, storage as a service provides an on-demand, flexible data sharing across the networks. This reduces the burden of local data storage management and avoidance of resource maintenance (Hardware or software). In this paradigm, data owner loses the control of the outsourced data, once the data leaves the data owner premises. Due to this, the data on an untrusted cloud server is at risk in terms of integrity, confidentiality and availability of the outsourced data. In order to maintain the outsourced data without corruption from the internal or external adversary, an efficient data auditing verification method is required for data verification. In this paper, we propose a flexible data auditing method using block level auditing of data distributed on multiple cloud servers. This method utilizes the Computational Diffie-Hellman (CDH) and Decisional Diffie-Hellman (DDH) problem solving techniques. The performance of data verification with different sizes of data blocks on multiple servers. Compared to the existing methods of data auditing the proposed method minimizes the computation, communication and storage overheads.

Keywords: Auditing; Cloud Computing; Corruption; Storage management; Verification

1 Introduction

In modern computing technology, cloud-computing paradigm is an important technology used to provide various remote services such as, computing, storage, memory and other services with reduced computing cost when compared with many traditional approaches. There are various cloud service providers available in recent days including Amazon Web Services, Microsoft Azure, Google Cloud Platform and IBM Cloud that provide storage as a service [5]. Storage as a Cloud service is one of the important features of cloud computing used to share user's data across the network [8,14]. The cloud servers examine the outsourced data very frequently because the data can be lost or corrupted due to hardware failure, software failure or from the assailants [12,17].

Maintaining the integrity of outsourced data in a cloud server is an important issue in cloud computing [19]. In order to maintain the reputation of the cloud service, the cloud service providers set access restrictions on the services it provides to the users [2]. To avoid losing of profit of the service and to maintain the quality of the cloud service, verification of the integrity of outsourced data becomes mandatory before data utilization. To verify the integrity of outsourced data, various traditional approaches such as RSA, hash functions, MAC, Digital signature [9–11] are proposed. These proposed methods retrieve the entire data from the servers to verify the correctness of the outsourced data so that the auditor can derive the user data from this information and it takes more computation and communication cost, which can degrade the efficiency of the system. Therefore, the traditional integrity checking approaches are not suitable in cloud computing to utilize the resources optimally [6]. In general, the size of the data is very large for downloading the server and verification of data integrity would demand availability of more resources.

There are many methods proposed for checking data integrity without downloading the entire data from the server. This verification can be either private verification, public verification or delegated verification also called as public auditing. In these methods, it is essential to divide data into smaller blocks. Random verification of data blocks is preferred instead of retrieving the entire data block for verification after the data owner had signed these data blocks.

Organisation of the rest of the paper is as follows; Section 2 explains the various existing data auditing methods. Sections 3 and 4 explains the statement of the problem and the detailed algorithms for the method proposed. Section 5 presents the performance analysis of the proposed method and finally we concluded in the Section 6.

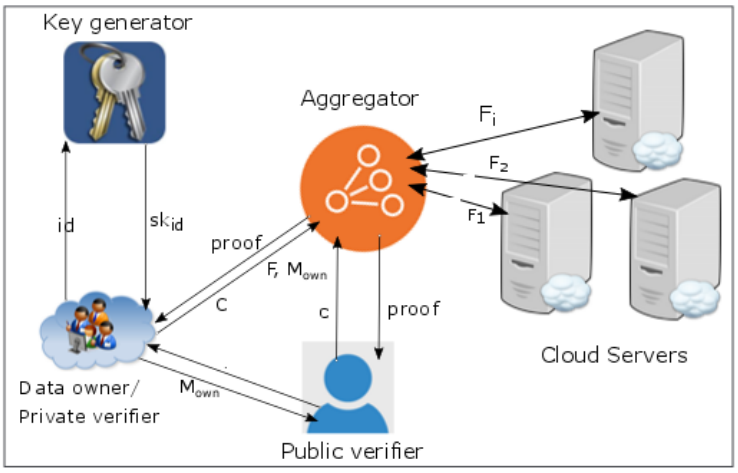

Figure 1: Data auditing system model

2 Related Work

In cloud computing, to ensure the integrity of the data on untrusted storage on single cloud the various public and private auditing schemes are proposed [1, 3, 4].

In Ateniese *et al.* [1], proposed a public auditing model as Provable Data Possession (PDP) for auditing data on untrusted storage entity. They introduced homomorphic linear authenticators to audit the selected outsourced data blocks of the outsourced file. This method may leak the user information to the auditor during auditing process and which may leads to helps to derive the user information, therefore it does not provide the security for the outsourced data.

In Juels *et al.* [4], introduced a Proof of Retrievability (PoR) model for verification and retrieval of data from remote data storage service using error-correcting codes. This method has the following drawbacks: 1) It has fixed data auditing challenges, 2) Suffers from public and delegate verification.

In Yujue *et al.* [20], addressed identity-based data outsourcing for distributed users. In this method of audit, the data users have to authorize the dedicated proxy before storing data on cloud server and which is more controlled way of outsourcing data on cloud server. To verify the integrity of outsourced data is more expensive.

In [7, 13, 15, 16] proposed a data auditing protocol on the cloud server to support the batch auditing on multi-servers. In these methods, individuals used data tags to the owner and these cannot help to combine multi-owner tags to conduct batch auditing. To combine these individual tags, third party auditor is introduced which takes additional computation and communication cost. Due to these overhead this method reduces the efficiency of the auditing system.

In [21, 22] proposed the data privacy protocol for auditing user's data in cloud storage server by using the bilinear privacy operations to verify the correctness of the response message. The drawback of this method is that, for multi-cloud auditing to segregate the data blocks of the multiple users, the auditor takes more computational task and which is a low end user entity in the cloud storage system. This method suffers from yet another drawback while using unencrypted used information for auditing process, thereby empowering the auditor to derive user data.

3 Statement of the Problem

3.1 System Model

The system model considered in the proposed work as shown in the Figure 1. It consists of five components such as; *key generator, cloud servers, verifier, cloud users and aggregators.*

Key generator: It is an entity, which receives the identity of the user(ID) and generate the secrete key(*sk_id*) for the user(ID) using a computational Diffie-Hellman (CDH) method.

Cloud users: It is an individual user or an organization which outsource the data on multi cloud servers for maintenance and management of shared data.

Verifier: It is an entity, either the data user or third party auditor to check the correctness of outsourced data using Decisional Diffie-Hellman method.

Cloud servers: It is a set of servers, which managed by the cloud service provider to provide the storage service, which has massive compute and storage facility.

Aggregator: It is an independent and trusted entity. Which distribute the requests to the servers and aggregate the responses from the servers.

3.2 Security Model

In cloud data storage model, the auditor and the cloud servers are semi trusted entities, which means they are

honest but it is curious about the received data. Due to this semi trusted entities this may create the following attacks.

User attack: Poor identity and access management procedures an unauthorized intruder can attack the user

Data segregation: Incomplete security perimeters and configuration of virtual machines could provide a threat against data integrity.

Response attack: The semi trusted server may generate the response message for the requested data blocks from the previous audits without using the actual owner's data.

Data attack: The server and auditor can derive the user's data using metadata information in the frequently auditing task.

3.3 Objectives

In the proposed method the following objectives are achieved for auditing outsourced data on cloud storage server.

In our proposed method we are achieved the following objectives for auditing data on multi cloud storage.

Flexible data auditing: The private or public data verification method can be applied based on the priority of the outsourced data.

Privacy of the data: In either of the auditing method, the verifier cannot derive the user data from the metadata.

Lightweight overhead: To optimize the storage, computation and communication overhead to perform the data auditing on cloud server.

3.4 Notations

The various symbols are used in this paper is listed in Table 1 as follows.

3.5 Cyclic Group Operation

Consider G1, G2, G_T are cyclic multiplicative groups of order prime number p, bilinear map using these groups is defined as $G1 \times G2 \longrightarrow G_T$ of order p.

The property of the map e is that for all a, $b \in p$ is defined as that $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for all $g_1 < G_1$ and $g_2 < G_2$. The group G_T is distinct from input groups G_1 or G_2 and all the elements of this map is also elements of group G_T.

Table 1: Notations

Symbol	Meaning
F	erasure encoded file $F = \{F_{ij}\}$
t	file tag
n	number of data blocks
s	number of sectors per block
σ	block authenticator
M_{owner}, M_{agg}	File metadata for owner and aggregator
$H(), h()$	hash functions
Φ	set of authenticators
r and x	random number
c	challenge message
CS_{l_i}	set of cloud servers
p	prime number
msk_{id}	master secret secret key

4 Data Auditing Design Methodology

4.1 Basic Auditing Method

The basic data integrity verification scheme consists of three stages such as; key generation, metadata generation and data auditing. The detail of these stages has shown in Figure 2.

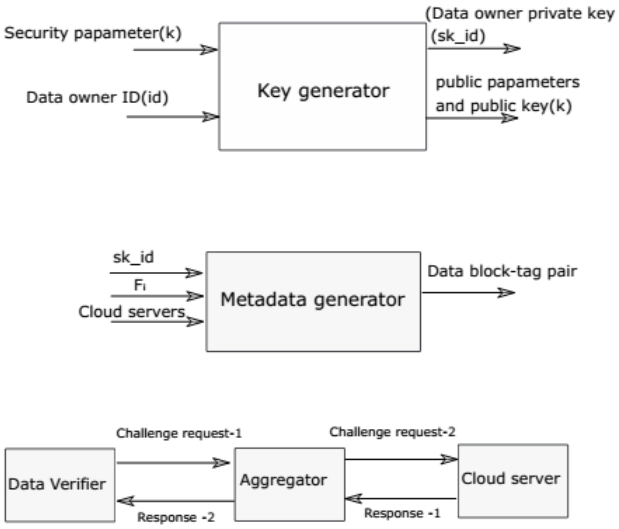

Figure 2: Basic auditing model

4.1.1 Key Generation

It is an entity, which takes the input as security parameter(k) and the data owner identifier (id) and generates the public parameters, secrete key, public key and owners private key(pk_id).

4.1.2 Metadata Generation

To generate the metadata for a given file (F_i) of the owner (id), the metadata generator takes input as owners private key (pk_id) and the file (F_i) as input and generates the signature of file blocks interns of $block - tag$ pair.

4.1.3 Data Auditing

The data auditing process consists of five steps of request, response and verification among verifier, aggregator and cloud servers.

1) Verifier send the challenge request to aggregator for verification of selected number of data blocks stored on cloud servers.

2) The aggregator searches the requested data blocks meta data from the metadata table and then distribute the request to the corresponding cloud server.

3) After receiving the responses from the cloud servers, the aggregator combines all the responses.

4) Aggregator sends the final combined response to the verifier.

5) The verifier verify the response message using bilinear map operation. If the response is valid, verifier confirms data blocks are not modified, otherwise he declares data blocks are modified.

4.2 Proposed Batch Auditing Method

The proposed data auditing method consists of key generation, metadata generation and data verification procedure. The detailed explanation of these procedure as follows.

4.2.1 Key Generator

The key generator select two random positive integer numbers r and x and calculate $A = g^x$ and $B = g^r$ where g is the generator i.e $g <$ group G_1, keeps x has secrete key and $\{g, A\}$ as public parameters.

For the given data owner (id) key generator calculate the signature using Equation (1) and sends the private key $sk_{id} = (\sigma and B)$ to the data owner. Then the data owner verifies the private key using Equation (2).

$$\sigma = r + x(H(id, B)) \bmod q \quad (1)$$

$$g^{\sigma} \stackrel{?}{=} BA^{H(id,B)} \quad (2)$$

The detailed algorithm for key generation and authorization is explained in Algorithm 1. The proof for the correctness equation and example as follows;

$$g^{\sigma} \stackrel{?}{=} BA^{H(id,B)}$$

$$\begin{aligned} LHS &= g^{\sigma} \\ &= g^{r+xH(id,B)} \\ &= g^{r}.g^{xH(id,B)} \\ &= BA^{H(id,B)} \\ &= RHS. \end{aligned}$$

For example, $g = 3$, $q = 11$, $r = 3$, $x = 4$:

$$\begin{aligned} A &= g^x = 3^4 \\ B &= g^r = 3^3 \\ \sigma &= r + x(H(id, B)) \bmod q \\ &= 3 + 4.H(id.B) mod 11 \\ g^{\sigma} &\stackrel{?}{=} BA^{H(id,B)} \\ 3^{3+4.H(id.3^3) mod 11} &= 3^3.3^{4.H(id.3^3) \bmod 11} \\ &= 3^{3+4.H(id.3^3) \bmod 11} \end{aligned}$$

Algorithm 1 Key Generation

input: User identity (id)
output: Master secrete key (x), Public parameters (p, q, g, A, H)

1: Select a random number x, r from a set of positive integer numbers Z_q^*
2: compute A, B and σ
$A = g^x$ and $B = g^r$ $\sigma = x + r(H(id, B)) \bmod q$
3: keeps x has master secrete key.
4: sends private key $sk_{id} = (B, \sigma)$ and public parameters to user.
5: Verify the user identity id by solving the DDH problem $g^{\sigma} \stackrel{?}{=} BA^{H(id,B)}$
6: If the equation is verifies then accept the user (id) private key sk_{id}, otherwise reject it.

4.2.2 Metadata Generation

The data owner with the valid private key (sk_{id}) prepares the metadata for the file F and stores the metadata and the corresponding file on cloud server CS. Consider the file F is split in to n blocks and each block of s sectors i.e; $F = F'_{ij}$, where $i = 1ton$, $j = 1tos$ and F' is the encrypted data block. The Data owner calculates the hash value for each sector using MD5 algorithm i.e. $h1(F'_{ij})$ and prepare the metadata M_i for the file block F_i using Equation (3). Then the owner sends the file blocks and metadata $\{F_i$ and $M_i\}$ to the cloud server CS_{l_i}. The procedure of metadata generation is explained in Algorithm 2.

$$M_i = (h(CS_{l_i}, i, name_i).\Pi_{j=1}^{s} u_{ij}^{F_{ij}})^{\sigma} \quad (3)$$

Where name i is the identifier of each block, j is the sector number in the data block and u_{ij} is the random number.

Algorithm 2 M_i

input: File (F), sk_{id}
output: Metadata M_i for the file block F_i

1: Data owner split the file F in to n blocks $\{F_i\}$, and each encrypted data block in to s sectors i.e., $\{F'_{ij}\}$ where $i \leq n$, and $j \leq s$
2: Data owner selects s random number vector $\{u_i\}$ where $j \leq s$
3: calculate the hash values for each encrypted file block i.e., $F_{ij} = h(F'_{ij})$
4: calculate the metadata for the ith file block i.e, $M_i = (h(CS_{l_i}, i, name_i).\Pi_{j=1}^{s} u_{ij}^{F_{ij}})^{\sigma}$
5: Data owner adds $\phi_i = (i, u, CS_{l_i}, name_i)$ to the M_i table
6: Data owner sends M_i to aggregator, then the aggregator stores in his metadata table, M_{agg})

and stores file blocks in cloud server CS_{l_i}

4.2.3 Data Verification

To verify the data stored in cloud server CS, the auditor sends a request for selected number of blocks c to aggregator and aggregator identifies the corresponding cloud server using the metadata table M_{agg} and further sends a request to the corresponding cloud server CS_{l_i}. After receiving the request from the aggregator, cloud server prepares the response message and send to the aggregator. The aggregator combines all the responses and sends the aggregated response to the auditor. The auditor verifies the correctness of the response message using the Equation (4). The details of the data auditing algorithm is explained in Algorithm 3.

The block diagram for private and public data verification as shown in the Figure 3 and Figure 4 respectively.

$$e(M, g) \stackrel{?}{=} e(\Pi_{i=1}^{c} h_i^{F_i} \Pi_{j=1}^{s} u_j, BA^{H(id,B)}) \quad (4)$$

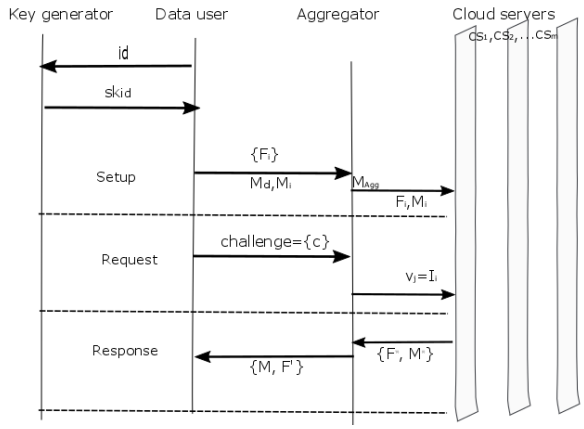

Figure 3: Batch auditing setup and private verification

The proof for data auditing response verification as follows.

$$\begin{aligned} LHS &= e(M, g) \\ &= e(\prod_{cs_i} M^{(i)}, g) \\ &= e(\prod_{cs_i} \prod_{v_l \in M_i} M_{v_l}, g) \\ &= e(\prod_{cs_i} \prod_{v_l \in M_i} h_l \prod_{j=1}^{s} u_j^{F_{v_l j}}, g^{\sigma}) \\ &= e(\prod_{i=1}^{c} h_i \prod_{j=1}^{s} u_j^{F_{v_l j}}, BA^{H(id,B)}) \\ &= RHS \end{aligned}$$

Algorithm 3 Data_Audit(c)

input: Challenge message c data blocks
output: Metadata for the file block F_i

1: The private or public verifier sends the challenge message (c) to the aggregator
2: Aggregator prepares the index set(I_i) for the corresponding c request blocks using $block - tag$ pairs stored in cloud server.
3: Aggregator sends the index set(I_i to the cloud servers.
4: for each cloud server v_l calculates $(M^{(i)}, F_j^{(i)})$ and send to the aggregator.
$M^{(i)} = \Pi_{v_l \in I_i}\{M_{v_l,j}\}$
$F_j^{(i)}) = \Sigma_{v_l \in I_i}\{F_{v_l,j}\}$
5: Aggregstor prepsres the aggregated message (M and F') then sends to the verifier.
$M = \Pi_{cs_i} M^{(i)}$ and $F'_l = \Sigma_{cs_i} F_l^{'(i)}$
6: Verifier solves the following DDH problem and returns the status of the file block.
$e(M, g) \stackrel{?}{=} e(\Pi_{i=1}^{c} h_i \Pi_{j=1}^{s} u_j, BA^{H(id,B)})$
7: If the above equation holds then it responds success, otherwise it responds failure.

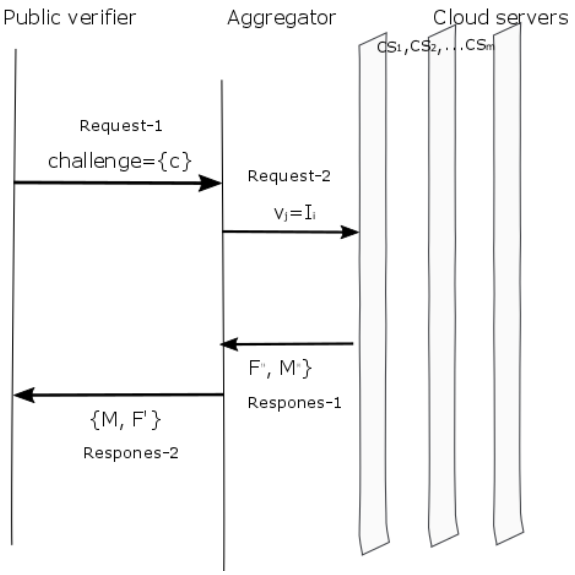

Figure 4: Data auditing using public verification

Table 2: Tag generation time for different file sizes

File Size(MB)	Tag generation Time(Seconds)
1	1.87
2	3.17
4	4.27
6	6.85
8	10.05
10	14.65
12	18.59
14	19.87

5 Performance Analysis

5.1 Simulation Setup

To evaluate the performance of our proposed data auditing method the following simulation setup are considered. All the algorithms are implemented using Python programming language with built in cryptographic functions in Python library. The simulation result is tested on Amazon Web Service virtual machines (VM).

Two VMs is run as a cloud server and one VM run as a aggregator with Ubuntu operating system, t2.small vCPU and 2GB EBS storage configuration. The data owner and the public verifier runs on a Laptop with 64 bits Windows 10 operating system, i3 intel processor and 4GB physical memory configuration.

5.2 Result Analysis

In ordered to show the performance of the proposed method, we compared with existing ID-DPDP [18] data auditing method. The performance of the algorithm is evaluated based on the tag generation, tag verification cost and file auditing cost for different data block size with different batch size.

Table 2, shows the comparison of tag generation cost for different sizes of files with 256KB of data blocks size. First column represent the different block size in Megha Bytes and Second column represents the tag generation cost in seconds. It is observed that, for smaller sized file sizes tag generation cost is linear than the larger file sizes. The proposed method shoes that, it has lightweight computation overheads for the larger file sizes. Table 3 shows the tag generation cost for different sizes of the data blocks of 1MB file size.

Table 5, shows the comparison of tag verification cost on cloud servers. To verify the different sizes of data block for 1MB file tag verification cost shows that, for smaller size of data blocks both the methods has same order of growth for tag verification cost. In case of larger size of data blocks our proposed method has better performance than the ID-DPDP method.

In Table 5, shows that, the performance comparison to audit 10MB data file with 256KB of data blocks for different number of batch size. For smaller batch size of auditing both the method has same order of auditing cost and for larger batch size our proposed method performance is better than the existing method.

Table 3: Tag generation time for different data block sizes

Data block Size(KB)	Tag generation Time(Seconds)
256	1.856
500	1.047
768	0.677
1024	0.384

Table 4: Tag verification cost with different block size

Data block Size(KB)	Tag generation Time(Seconds)
256	0.0856
500	0.0447
768	0.0327
1024	0.0154

The cost performance comparison for auditing data on multiple servers interms of computation, communication and storage overheads with Zhu, ID-DPDP as shown in Table 6,

6 Conclusions

In this paper, the flexible data verification method for outsourced data integrity checking is proposed. This introduce the data aggregator, which distributes the requests and aggregates the response from the servers during the auditing process. The proposed method utilizes the CDH and DDH problem to verify the correctness of the outsourced data. Finally, the performance comparison of the proposed data verification method with the existing method. The experimental results and security analysis shows that, the proposed method is better than the existing method

References

[1] G. Ateniese, R. Burns, R. Curtmola, *et al.*, "Provable data possession at untrusted stores," in *Proceedings*

Table 5: Tag verification cost with different block size

Number of Blocks	Data Verification Time (Seconds)
4	3.1856
8	5.0127
12	8.5327
24	10.0154

Table 6: Computation performance of auditing methods

Auditing Method	Computation overhead	Communication overhead	storage overhead
Zhu [23]	expo:(2s+ns+2n+c) mul:(c+s+2) pairing: 3	$c.log_2 n + (1+s)log_2 q$	$T_{cl} + T_o$
ID-DPDP [18]	expo:(2s+ns+2n+c) mul:(sn+s+1), hash:(ns+cs) pairing: 2	$log_2 n + (2+s)log_2 q$	$T_{cl} + T_o$
Proposed method	expo:(2ns+2n+c) mul:(sn), hash:(ns+cs) pairing: 3	$log_2 n + s log_2 q$	$M_{owner} + M_{Aggre}$

of the 14th ACM Conference on Computer and Communications Security (CCS'07), pp. 598–609, 2007.

[2] B. Balusamy, P. V. Krishna, G. S. T. Arasi, and V. Chang, "A secured access control technique for cloud computing environment using attribute based hierarchical structure and token granting system," *International Journal of Network Security*, vol. 19, no. 4, pp. 559–572, 2017.

[3] J. Han, H. Yan, J. Li and Y. Zhang, "A novel efficient remote data possession checking protocol in cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 78–88, 2017.

[4] A. Juels, Jr. Kaliski, S. Burton, "Pors: Proofs of retrievability for large files," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pp. 584–597, 2007.

[5] C. W. Liu, W. F. Hsien, C. C. Yang, and M. S. Hwang, "A survey of attribute-based access control with user revocation in cloud data storage", *International Journal of Network Security*, vol. 18, no. 5, pp. 900–916, 2016.

[6] S. K. Madria, "Security and risk assessment in the cloud," *IEEE Compurter*, vol. 49, no. 9, pp. 110–113, Sept. 2016.

[7] J. Mao, J. Zhang, P. Li, "An oriented-group supporting multi-user public auditing for data sharing," in *IEEE International Conference on Smart City/SocialCom/SustainCom*, pp. 592–599, 2015.

[8] A. Mosa, H. M. El-Bakry, S. M. Abd El-Razek, S. Q. Hasan, "A proposed E-government framework based on cloud service architecture," *International Journal of Electronics and Information Engineering*, vol. 5, no. 2, pp. 93–104, 2016.

[9] Z. Ren, L. Wang, Q. Wang, M. Xu, "Dynamic proofs of retrievability for coded cloud storage systems," *IEEE Transactions on Services Computing*, 2016.

[10] R. L. Rivest, *The MD5 Message-digest Algorithm*, Apr. 1992. (`https://tools.ietf.org/html/rfc1321`)

[11] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of ACM*, vol. 21, pp. 120–126, Feb. 1978.

[12] D. Song, I. Fischer and U. Shankar, "Cloud data protection for masses," *IEEE Computing*, vol. 45, no. 1, pp. 39–45, 2012.

[13] S. Tan, L. Tan, X. Li, Y. Yan, "An efficient method for checking the integrity of data in the cloud," *China Communications*, vol. 11, no. 9, pp. 68–81, Sept. 2014.

[14] Y. Tian, Y. Peng, G. Gao, X. Peng, "Role-based access control for body area networks using attribute-based encryption in cloud storage," *International Journal of Network Security*, vol. 19, no. 5, pp. 720–726, 2017.

[15] B. Wang, B. Li, and H. Li, "Panda: Public auditing for shared data with efficient user revocation in the cloud," *IEEE Transactions on Services Computing*, vol. 8, no. 1, pp. 92–106, 2015.

[16] B. Wang, H. Li, X. Liu, F. Li, X. Li, "Efficient public verification on the integrity of multi-owner data in the cloud," *Journal of Communications and Networks*, vol. 16, no. 6, pp. 592–599, 2014.

[17] C. Wang, K. Ren and Q. Wang, "Security challenges for public cloud," *IEEE Internet Computing*, vol. 16, no. 1, pp. 69–73, 2012.

[18] H. Wang, "Identity-based distributed provable data possession in multi-cloud storage," *IEEE Transactions on Service Computing*, vol. 8, no. 2, pp. 328–340, 2015.

[19] Q. Wang, C. Wang, K. Ren, *et al.*, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 847–859, May 2011.

[20] Y. Wang, Q. Wu, B. Qin, *et al.*, "Identity-based data outsourcing with comprehensive auditing in clouds," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 940–952, 2017.

[21] L. Xia, L. Yang, "An efficient and secure public batch auditing protocol for dynamic cloud storage data," in *IEEE International Computer Symposium*, pp. 671–675, 2016.

[22] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 9, pp. 1717–1726, 2013.

[23] Y. Zhu, H. Hu, G. J. Ahn and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," *IEEE Transations on Parallen and Distributed Systems*, vol. 23, no. 12, pp. 2231–2244, 2012.

Biography

Prakash G L is an Assistant Professor with the Department of Computer Science and Engineering of University of Petroleum and Energy Studies, Dehradun, India. He received his B.E and M.E degrees in Computer Science and Engineering from Bangalore University, Bangalore. He is presently pursuing his Ph.D program in the area of data storage and security in Cloud Computing from University of Petroleum and Energy Studies. He has published more than five international journals and ten international conference papers.

Dr. Manish Prateek did his Undergrad and Post Grad degree in the field of Computer Science from South West State University, (formerly known as Kursk State Technical University), Russia, in 1996. Since then he worked at different level in the IT industry in India as well as in Middle East. He did his PhD in the area of Manufacturing & Robotics and was awarded the degree in the year 2007. In 2005 he took up his career into technical education and started as Associate Professor and Head, Dept. of Information Technology at GRIET Hyderabad and later became a full Professor at the age of 36. Currently he is working as Professor & Associate Dean at UPES, Dehradun. His area of research includes Robotics and Automation, Image Processing and Patter Recognition, Cyber Security, Machine Vision etc. So far, more than 40 research papers have been published by him in different International Journals. He is also a recipient of Lifetime Achievement Award for his contribution in the field of education and research by the prestigious scientific society Pentagram Research Centre. He is also a member at the prestigious International Federation for Systems Research (IFSR), Austria

Dr. Inder Singh, M.Sc.(IT), M. Tech. (IT), Ph.D., Microsoft Certified Professional, IBM DB2 certified, e-commerce certification from Asset International, Dell EMC's Certified Data Science Associate. He is an Assistant Professor (S.G.) at School of Computer Science and Engineering, UPES, Dehradun. He has over 16 years of working experience. He has started his career as Systems Administrator and switched to teaching profession after 6 years. His area of research includes Computer Networks, Cloud Computing and Virtualization, and Data Science. So far, he has published and presented more than 18 research papers in different International Journals and conferences.