INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research

International Journal of Cloud Computing

ISSN online: 2043-9997 - ISSN print: 2043-9989
https://www.inderscience.com/ijcc

Towards P2P dynamic-hash-table-based public auditing for cloud data privacy, security and integrity

Raziqa Masood, Nitin Pandey, Q.P. Rana

DOI: 10.1504/IJCC.2023.10054984

Article History:

Received:	21 April 2020
Accepted:	03 May 2020
Published online:	27 March 2023

Copyright © 2023 Inderscience Enterprises Ltd.

Towards P2P dynamic-hash-table-based public auditing for cloud data privacy, security and integrity

Raziqa Masood* and Nitin Pandey

Amity Institute of Information Technology,
Sector-125, Noida – 201313 (UP), India
Email: raziqa.masood@student.amity.edu
Email: npandey@amity.edu
*Corresponding author

Q.P. Rana

Jamia Hamdard University,
Hamdard Nagar, New Delhi – 110062, India
Email: qprana@jamiahamdard.ac.in

Abstract: Cloud storage is the most demanded feature of cloud computing to provide outsourced data on-demand. However, users are in a dilemma to trust over the cloud service providers regarding whether privacy is preserved, integrity is maintained, and security is guaranteed, towards the outsourced data. Therefore, it requires developing an efficient auditing technique to provide confidence upon the data present in cloud storage. We propose a peer-to-peer (P2P) public auditing scheme to audit outsourced data using a dynamic-hash-table (DHT) to strengthen the users' trust, and availability over the data. Each DHT maintains the information of outsourced data, which helps the auditors to provide safety and integrity while auditing them. Moreover, these auditors are organised into a structured P2P to accelerate the auditing along with the auditor's availability. Thus, the proposed scheme overcomes from a single point of failure, and found to be computational less as compared with the existing methods.

Keywords: cloud computing; dynamic hash table; DHT; outsourced data storage; peer-to-peer; P2P; privacy; public auditing.

Reference to this paper should be made as follows: Masood, R., Pandey, N. and Rana, Q.P. (2023) 'Towards P2P dynamic-hash-table-based public auditing for cloud data privacy, security and integrity', *Int. J. Cloud Computing*, Vol. 12, No. 1, pp.72–89.

Biographical notes: Raziqa Masood is presently pursuing her PhD in Information Technology from the Amity University, Noida, India. She received her MTech from the Integral University, Lucknow, India. Her research interest includes security and privacy issues in cloud data storage.

Nitin Pandey is an Assistant Professor (Grade 3) at the Amity Institute of Information Technology, Amity University, Uttar Pradesh, India. His areas of research are coding theory, cryptography, network and cyber security.

Copyright © 2023 Inderscience Enterprises Ltd.

He received his PhD from the Mewar University Chittorgarh. He instructs courses covering CCNA routing and switching, CCNA security, CCNA cyber operation, cybersecurity essentials, IoT fundamentals, big data and analytics, and NDG Linux essentials.

Q.P. Rana is a Professor and the Director of Computer Center at the Jamia Hamdard University, New Delhi. He has 15 years of experience in cryptography and network security. He received his PhD from the Jamia Hamdard University, New Delhi.

This paper is a revised and expanded version of a paper entitled 'DHT-PDP: a distributed Hash table based provable data possession mechanism in cloud storage' presented at 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2020.

1 Introduction

Cloud computing is accepted as the next evolution of information technology (IT) industries with its several cutting edge facilities, such as geographical independence for resource pooling, on-demand availability to use the cloud resources, the cost for only accessed resources and services (Liu et al., 2011). These services are possible after deploying centralised frameworks of the cloud (web-servers), which manage and handle the outsourced data. A cloud service provider (CSP) is responsible for managing cloud services, such as retrieval and maintenance of the outsourced data along with worldwide availability. It does provide a powerful platform more than personal computer systems with an inexpensive solution to users for storing, updating, and computing data without any investment (Fox et al., 2009).

CSP manages all cloud services on the outsourced data without having any control of its owners who have only faith in CSP for the correctness and privacy of their outsourced data. Many reports reveal the several threats and privacy leakage on cloud data through the misbehaving nature of CSP in the past (Amazon S3 Team et al., 2008; Shah et al., 2008; Wang et al., 2009; Krebs, 2009; Nayak and Tripathy, 2018a). In general, CSP hides the information such as deletion of the unused cloud data, and data lost during processing, to maintain its reputation.

Moreover, a user cannot verify the outsourced data due to cryptographic primitives on the cloud data (Juels and Kaliski, 2007), expensive input/output operations, and requirement of massive network bandwidth. Therefore, it is impractical to verify the vast outsourced data from the user's end (Zhang et al., 2010). Thus, the data owners and users require a third-party authority to perform verification on the requested data efficiently without belief on CSP blindly.

A visual solution to verify the outsourced data in cloud computing is to place a auditor between users and CSP. The third-party public auditor is resource capable and considered as an expert to perform operations on the cloud data. It audits and confirms the correctness to users for each outsourced data transaction. On the other hand, CSP uses the public auditor for its independent arbitration purposes of its cloud-based service platform (Shah et al., 2008). Therefore, the deployment of a public auditor within cloud

infrastructure provides a practical and economical solution for cloud operations and builds trust towards the remotely stored cloud data (Nayak and Tripathy, 2018b).

However, existing public auditing schemes have major issues that have to be addressed to make them effective and efficient. First, these schemes extract user's information during data auditing, which leads to data leakage (Ateniese et al., 2007; Wang et al., 2009; Shacham and Waters, 2008). Thus, public auditors must include a mechanism to preserve data privacy, which is free from data encryption (Wang et al., 2011). Second, these schemes should address the efficiency and scalability issues to audit huge auditing requests from various users. An efficient multi-user auditing mechanism can be introduced in public auditors to support batch-auditing (Yang and Jia, 2012). Third, public auditing schemes have another issue to make an efficient distribution of auditing workload on different auditors. These auditors are independently assigned the auditing works, which may lead some auditors are overloaded with auditing works and unavailable for new users. Therefore, auditing scheme includes a mechanism to distribute works uniformly within the auditors. Another issue of auditing schemes is to audit the dynamic updates at CSP, which is required by many cloud computing applications. Thus, auditing schemes must support data dynamics (Wang et al., 2010; Zhu et al., 2011).

These auditing requirements would be achieved using Merkle-hash tree (MHT) in public auditing scheme as in Wang et al. (2010), but it suffers from computation costs and communication costs to perform updating and verification. Further, an index-hash table (IHT)-based public auditing has been proposed to reduce the overheads on auditors (Zhu et al., 2011; Nayak and Tripathy, 2018b). Recently, Chen et al. (2018) proposed an adjacency-hash table (AHT) to achieve dynamic auditing in fewer computation costs and communication overheads. Unfortunately, these existing schemes do not address the organising issue of public auditors, which would be an emerging challenge to handle a bundle of auditing requests from different cloud users.

Therefore, we propose a peer-to-peer (P2P) organisation for public auditors. P2P organisation provides an efficient solution for work distribution over different auditors and overcomes from a single point of failure. For efficient public auditing, we use a dynamic-hash-table (DHT) implementation in our proposed public auditing scheme. Furthermore, we use Boneh, Lynn, and Shacham (BLS) signature technique (Boneh et al., 2004), and bilinear map to implement batch auditing. Our contributions in this work are summarised below:

1 We introduce a P2P chord-ring (Stoica et al., 2001) organisation of public auditors to make them self-organise, self-distribute workload, and empowered with a collaborative auditing approach.

2 We use the DHT data structure at each peer auditor to store cloud meta-data, which helps to provide an effective and secure dynamic auditing solution.

3 We show our propose public auditing scheme is well suitable to perform privacy protection, batch user-data auditing, group peer auditing, and dynamic data auditing.

4 We formally prove and analyse that our propose scheme can mitigate different attacks such as forge attack, replacing attack, and replay attack.

5 Finally, we evaluated and analyse the performance of our propose protocol regarding its computation costs, communication costs, and auditing time consumption and found to be using less overhead as compared to the existing schemes.

The rest of the paper starts with a brief explanation of the existing public auditing schemes (Section 2). Problem statement and our proposed system model are mentioned in Section 3. Section 4 explains the proposed P2P dynamic public auditing protocol, which includes the verification of privacy protection (Subsection 4.1), dynamic updating (Subsection 4.2), batch and group auditing (Subsection 4.3). Further, Sections 5 and 6 are formally prove the security and analyse the evaluation results of our scheme respectively. Finally, we conclude our work in Section 7.

2 Related work

Many research works have been proposed on public auditing in cloud computing to build an efficient and secure platform for cloud users and increase the trust towards CSP. In this direction, Ateniese et al. (2007) defined a privacy-preserving model for outsourced data in the untrusted cloud storage environment. They proposed a public auditing mechanism based on RSA homomorphic linear authenticator. It requires a linear combination of data blocks to the public auditors. Further, they proposed a partially dynamic auditing scheme using symmetric cryptography with limited public auditing (Ateniese et al., 2008). To make a full public auditing scheme, Wang et al. (2009) combinations BLS-based homomorphic linear authenticator and MHT. However, these schemes are not able to protect data privacy in the auditing process due to the requirement of a linear combination of sampled blocks.

These traditional public auditors have a critical issue to manage huge certificates as the number of outsourced data grows significantly. Wang et al. (2013) proposed a secure identification-based public auditing to overcome from public-key cryptography issues. Tan and Jia (2014) included an aggregate signature scheme in identification-based public scheme. However, these auditing schemes suffered from key escrow. Therefore, these auditing schemes are inappropriate for cloud-assisted mechanisms to provide batch auditing or dynamic updates.

For dynamic data auditing, a dynamic provable data possession scheme has been proposed using skip-list (Ateniese et al., 2007). This list helps in rank-based authentication, which supports dynamic verification with a general pattern. Wang et al. (2010) implemented MHT to perform dynamic auditing to support batch verification along with data privacy. However, these schemes require high computational costs of auditors and significant communication overheads while performing dynamic updates and verification (Zhu et al., 2011). Therefore, Zhu et al. (2011) IHT-based public auditing scheme (IHT-PA) to organise cloud data and its meta-data at auditors instead of CSP. This structure reduces the overheads but not efficient in dynamic updates due to its sequence processing on block elements, which leads to a change in the sequence index of a few blocks. Consequently, it performs recalculation of tags at CSP with extra computation costs and communication overheads.

Tian et al. (2015) designed a DHT, which is based on a two-dimensional data structure to support efficient outsourced data updates and audits. This structure consists

of an array and a linked-list, where array stores file identifier with a pointer to points its first data block in the link-list. To improve the dynamic updates, Shen et al. (2017) proposed a doubly link-list implementation along with a location array. This auditing scheme supports sampling block-less and global verification. This auditing scheme supports sampling block-less and global verification. Moreover, Chen et al. (2018) proposed an enhanced version of dynamic auditing using an adjacency-list instead of a link-list for maintaining block element information of the outsourced files at auditors. Dynamic operations of these schemes are restricted due to the sequential stickiness of file identifiers with an index of the file location array.

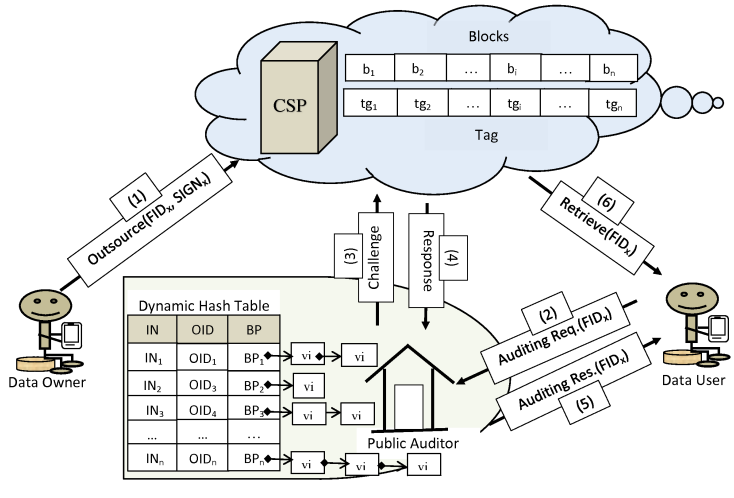
3 Problem statement and our system model

3.1 Problem statement

DHT-based public auditing architecture consists of data-owner/data-user, CSP, and third party auditor (public auditor) as shown in Figure 1. Data-owner/data-user can be an individual or an organisation which are retrieving a huge quantity of cloud data. Each outsourced data/file includes its file identification (FID) and corresponding signature (SIGN) as a block. Each blocks are tagged with a corresponding tag at CSP. CSP manages a number of cloud servers to provide scalability and on-demand accessibility of the outsourced data blocks. Public auditors keep these blocks using a DHT which is a two-dimensional data structure as shown in Figure 1.

Each DHT maintains an index number (IN) for a file having an overlay file identification (OID). All version information (vi) of a file is maintained by a link-list, which has starting block point (BP) in the DHT. The auditors track the latest version of file blocks. These information help auditors to check data integrity and reliability of storage services of CSP in the auditing process on the behalf of the user's data request. Thus, users avail burden-less storage and computation services to access their outsourced data along with data integrity from public auditors.

Figure 1 Architecture for DHT-based public auditing (see online version for colours)



Public auditors are credible but may be curious for the outsourced data. These auditors may be unavailable due to huge data auditing requests. We consider cloud storage services are available but CSP is not work honestly. CSP may hide some corrupted data and try to deceive users due to self-interest. It would perform the following dishonest activities as an attack to auditors during auditing process:

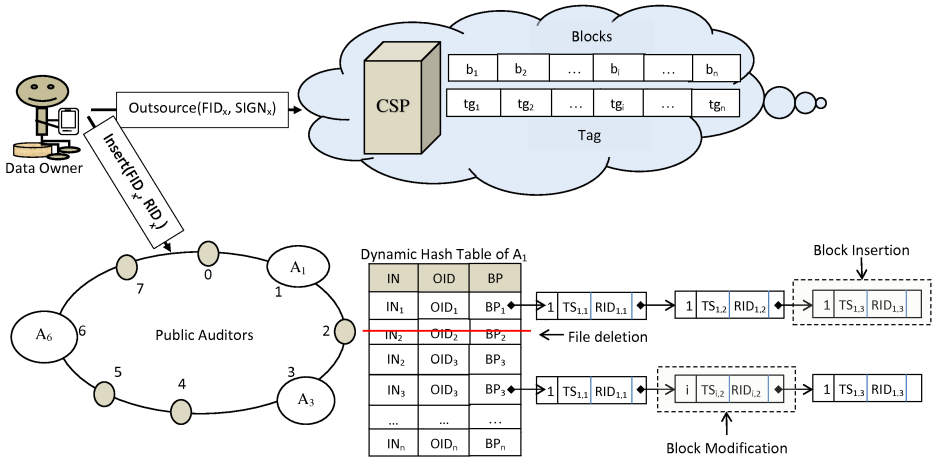
- Forge attack: CSP would forge data blocks along with their associated tags to deceive public auditors.
- Replacing attack: CSP would try to pass the verification process to replace some corrupted data blocks with required blocks and respective tags.
- Reply attack: CSP would use previously generated verification information to successful completion of their verification at an auditor.

These problems motivate us to develop an efficient and a secure auditing scheme with ensuring the auditors availability.

3.2 System model

Our proposed system model organises auditors into a P2P structure to make an efficient and secure performance for public auditors. It manages the outsourced data information on CSP and public auditors using a DHT to helps in integrity checking of stored cloud data.

Figure 2 System model for outsourcing data (see online version for colours)



3.2.1 P2P organisation of public auditors

The proposed system model implements a P2P [Chord-based (Stoica et al., 2001)] organisation of public auditors to ensure accessibility, availability and security during

huge auditing work load of outsourced data. It uses SHA1 (Karger et al., 1997) hash function to generate an overlay identification (m-bit) using physical identification of peers (auditors). It organises a ring structure through zero to 2^{m-1} address space using modulo 2^m. Figure 2 shows A_1, A_3, and A_6 overlay ids (OIDs) for auditors having physical ids 1, 3 and 6 respectively, where m is 3-bit. This organisation makes an efficient environment for each auditor to provide self-organising, load-balancing, accessibility of shared resources within log(n) searching cost, where n is the number of overlay members.

3.2.2 DHT-based outsourced information management

Each P2P auditor maintains a DHT to manage the outsourced information for ensuring efficient and secure auditing for cloud data. DHT tracks two types of basic information (file and its block) using two-dimensional data structure. As shown in Figure 2, each file with an OID (OID_i) is store at respective auditor's (A_i) DHT as an array data structure, which has its IN (IN_i) and block pointer (BP_i). (BP_i) points the first block of OID_i in DHT, similar as a link-list data structure.

Each node of DHT link-list refers as a block (b_{ij}) of a file, where i and j represent as j^{th} block elements of i^{th} file. Each block element keeps four information, they are current version number, issued time stamp (TS_{ij}), root overlay id (RID_i) and the pointer of next issued block element as shown in Figure 2. These DHT information are used to audit n number of outsourced files b_i ($\in \{1, n\}$) at CSP. DHT provides basic operations (insert, search, modify, and delete) of file and blocks as per their respective data structure.

4 The proposed P2P auditing scheme

This section presents our proposed P2P public auditing scheme based on DHT to provide an efficient and a secure auditing environment for users.

4.1 Dynamic verification to ensure privacy preserving

The proposed dynamic verification uses two multiplicative cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$, which are a group of large prime order p. These groups are mapped using a bi-linear map e ($\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$). SHA-1 $\mathbb{H}$ ($\{0,1\}^* \rightarrow \mathbb{G}_1$) is used to hash n blocks or chunks ($b_1, b_2, ..., b_n$) of a file before inserting into cloud server. These chunks are used to generate a root overlay id (ROI) at each data owner to provide the verification meta-data at auditors. The proposed verification process performs in two phases: setup phase and verification phase. The setup phase consists of four steps as follows.

Step 1 *Key initialisation:* Users generates two set of keys SK (= {s, sk}) and PK (= {k, l, m, pk}), where s ($\in \mathbb{Z}_{\mathbb{P}}^*$) is a random number, ($sk$, pk) is a key pair for signature, k and m are randomly selected element of generator $\mathbb{G}_1$, and l computed as k^s.

Step 2 *Data information:* Each file chunks ($b_1, b_2, ..., b_n$) of a file (FID_i) is hashed to generate m-bit chunk keys. Finally, these keys are used to generate an

overlay root id (RID_i) of m-bit, as shown in Figure 3. Further, a root version identifier (RVI) is generated to identify the version id of the generated RID_i as:

$$RVI = \{(FID_i, TS_i, RID_i) | 1 \leq i \leq n\} \tag{1}$$

where TS_i is the issued time stamp of RVI. Data owner sends these data infromation $(H(FID_i), RVI)$ to P2P auditors. An appropriate auditor stores these information into its DHT table.

Step 3 *Signature generation:* A signature σ_i is generated for each block b_i using data owner's public key m as follows:

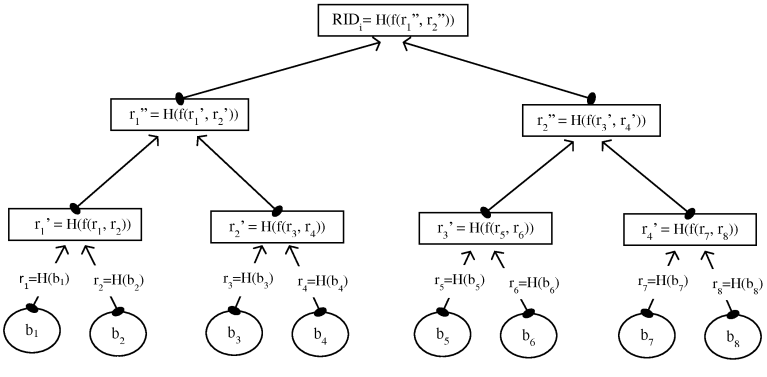
$$\sigma_i = \mathbb{H}(FID_i, TS_i, RID_i) m^{(r_i + \mathbb{H}(FID_i, TS_i, RID_i))}. \tag{2}$$

To ensure integrity of FID_i, data owner generates a file tag Γ $(= OID_i || SIGN(sk, FID_i))$, where $SIGN(sk, FID_i)$ represents a signature for FID_i using the owner's private key sk. Finally, data owner uploads FID_i, σ_i, and Γ to cloud storage and clean its local storage space.

Step 4 *Tag generation:* CSP generates a tag TG_i for each block b_i $(1 \leq i \leq n)$ using the bilinear map e after receiving outsourced block signature σ_i as follows:

$$TG_i = e(\sigma_i, l). \tag{3}$$

Figure 3 Generation of a root overlay id of a file using its chunks



Our proposed verification phase consists of four steps as follows:

Step 1 *File integrity check:* The verification starts at P2P auditor (A_x) after receiving an auditing request of a file tag Γ from a data user. A_x extracts the file id from $SIGN(sk, FID_i)$, and verifies $H(FID_i) \stackrel{?}{=} OID_i$. Unsuccessful result shows the requested file has been modified, hence the process can be terminated. Otherwise, A_x forwards the request to a P2P auditor A_i which has closest OID of $H(FID_i)$. A_i confirms metadata of $H(FID_i)$ in its DHT table, and goes to step 2; otherwise, report a file modification error to data requester.

Step 2 *Challenge:* Auditor A_i generates a challenge $chal = (OIDX = \{odix_i | 1 \leq i \leq k, k \leq n\}, RN = \{rn_i | i \in OIDX\}, RM)$ and sends to CSP, where $OIDX = \{odix_i | 1 \leq i \leq k\}$ is a random set file chunks to be checked, $RN = \{rn_i | i \in OIDX\}$ is a set of random numbers ($\in \mathbb{Z}_{\mathbb{P}}^*$), and RM is a random masking computed as $RM = l^r$, where r is a random number ($\in \mathbb{Z}_{\mathbb{P}}^*$).

Step 3 *Verification proof:* After receiving the challenge, CSP has to produce a challenge proof for data correctness. This process consists tag proof (TGP) and data proof (DP). In tag proof, CSP must be generate the aggregated authenticate of each corresponding tag of $OIDX$, i.e.,

$$TGP = \prod_{i \in OIDX} \theta_i^{RN_i}. \tag{4}$$

For data proof, CSP computes linear combination (LC) of the required chunks for $chal$, and then λ as follows:

$$LC = \sum_{i \in OIDX} RN_i.\mathbb{H}(b_i). \tag{5}$$

and

$$\lambda = e(m, RM)^{LC} \tag{6}$$

Finally, CSP sends these information (TGP, λ) to A_i as a response proof of the given challenge ($chal$).

Step 4 *Proof check:* After receiving the response from CSP, auditor A_i computes hash value (HV) of all blocks of the challenge and combined them as:

$$HV = \prod_{i \in OIDX} \mathbb{H}((FID_i, TS_i, RID_i))^{RN_i}. \tag{7}$$

Finally, A_i checks the received proof using following equation (8).

$$\lambda.e\left(HV.m^{\sum_{i \in OIDX} RN_i.\mathbb{H}(FID_i, TS_i, RID_i)}, RM\right) \stackrel{?}{=} TGP^r. \tag{8}$$

A_i replies to the data requester about the correctness of data request as per the equation outputs. The correctness of the equation (8) is demonstrated as follows:

$$\lambda.e\left(HV.m^{\sum_{i \in OIDX} RN_i.\mathbb{H}(FID_i, TS_i, RID_i)}, RM\right)$$

$$= e(m, RM)^{\sum_{i \in OIDX} RN_i.\mathbb{H}(b_i)}.e\left(\prod_{i \in OIDX} \mathbb{H}((FID_i, TS_i, RID_i))^{RN_i}\right.$$

$$\left..m^{\sum_{i \in OIDX} RN_i.\mathbb{H}(FID_i, TS_i, RID_i)}, RM\right)$$

$$
\begin{aligned}
&= e\left(m^{\sum_{i\in OIDX} RN_i.\mathbb{H}(b_i)}, RM\right).e\left(\prod_{i\in OIDX}\mathbb{H}((FID_i, TS_i, RID_i))^{RN_i}\right.\\
&\left.m^{\sum_{i\in OIDX} RN_i.\mathbb{H}(FID_i,TS_i,RID_i)}, RM\right)\\
&= e\left(\prod_{i\in OIDX}\mathbb{H}((FID_i, TS_i, RID_i))^{RN_i}\right.\\
&\left.m^{\sum_{i\in OIDX} RN_i.(\mathbb{H}(b_i)+\mathbb{H}(FID_i,TS_i,RID_i))}, k^{r-s}\right)\\
&= e\left(\prod_{i\in OIDX}(\mathbb{H}((FID_i, TS_i, RID_i))\right.\\
&\left.m^{\sum_{i\in OIDX} RN_i.(b_i+\mathbb{H}(FID_i,TS_i,RID_i))}\right)^{RN_i}, k^{r-s}\Bigg)\\
&= e\left(\prod_{i\in OIDX}\sigma_i^{RN_i}, k^{r-s}\right)\\
&= e\left(\prod_{i\in OIDX}\sigma_i^{RN_i}, k^{s}\right)^{r}\\
&= e\left(\prod_{i\in OIDX}\sigma_i^{RN_i}, l\right)^{r}\\
&= \prod_{i\in OIDX} TG_i^{RN_i.r}\\
&= TGP^r.
\end{aligned}
$$

4.2 Dynamic updating

The basic idea of dynamic updating such as block modification ($\mathbb{M}_b$), block insertion ($\mathbb{I}_b$), and block deletion ($\mathbb{D}_b$) in our scheme is based on P2P Chord protocol. In this, each files are hashed and placed at an appropriate P2P audior's (A_i) DHT table. It helps to distribute the auditing workload among peer auditors. block is update at an auditor's DHT which has closest OID as file of the block has (FID_i).

4.2.1 Block modification

To modify a outsourced data block (b_i) of the file (FID_i), user generates a new root id (RID_i') for modify/new data block (b_i') and sends a modification request ($FID_i, \mathbb{M}_b, TS_i', RID_i'$) to an auditor ($A_i$). After receiving the modification request, (A_i) computes $\mathbb{H}(FID_i)$ and finds the corresponding index at its DHT table. It replaces TS_i' and RID_i' accordingly in a link-list entries of the DHT. To update CSP, user computes signature (σ_i') of the modified block (b_i') using equation (2) and sends a modify request ($FID_i, \mathbb{M}_b, b_i, b_i', \sigma_i'$) to CSP. For each modification request, CSP generates a new tag (tg_i') for modified block (b_i') using equation (3) and updates its database.

4.2.2 Block insertion

To insert a block (b_j) of a file (FID_j), a root value of FID_j (RID"$_j$) is generated at user. User sends a block insertion request ($FID_j, \mathbb{I}_b, b_j, TS_j, RID_j$) to an auditor ($A_j$). A_j inserts the new block information in the link-list of its DHT table. User sends an insert message ($FID_i, \mathbb{I}_b, b_i, b'_i, \sigma'_i$) of b_j to CSP after computing a corresponding signature (σ_j) applying equation (2). CSP inserts the information in its database after generating a tag value (tg_j) with the help of received insertion information using equation (3).

4.2.3 Block deletion

To delete a k^{th} block (b_k) of a file FID_k, user generates deletion request (FID_k, $\mathbb{D}_b$, k^{th}) and sends to an auditor A_k. A_k computes the hash of FID_k and deletes the k^{th} block in the link-list of its DHT table. To update the deletion of block b_k at CSP, user sends a block delete information ($FID_k, \mathbb{D}_b$) to CSP. CSP computes and store a new file version and its tag of FID_k.

4.3 Verification of batch and group peer auditing

Our proposed auditing scheme combines all challenges of the requested data at each P2P auditor into a batch to send at CSP. CSP generates verification proof for each challenge ($chal$) and aggregate them using the aggregate BLS signature method (Boneh et al., 2003).

Let n number of P2P auditors receive auditing requests from k (on average) number of different users. An auditor ($A_x, 1 \leq x \leq n$) combines all the generated $chal$ ($n.k$) and sends to CSP. CSP generates tag proof (TGP_i) and data proof (λ_i) for each $chal_i$, where i = 1, 2, 3, ..., $n.k$. These generated proofs TGP_i and λ_i are aggregated into TGP_B [equation (9)] and λ_B [equation (10)] respectively.

$$TGP_B = \prod_{i=1}^{n.k} TGP_i \tag{9}$$

$$\lambda_B = \prod_{i=1}^{n.k} \lambda_i \tag{10}$$

Finally, CSP sends (TGP_i and λ_i) to A_i as a response of the challenges. A_i computes hash value ($HV_i, 1 \leq i \leq k$) for all users and requests other members to send their respective hash values. A_i generates a batch hash value (HV_B) after combining the all challenge hash values [equation (11)] and verify the received proof by checking equation (12). Finally, A_i responses the verification result to respective member peers.

$$HV_B = \prod_{i=1}^{n.k} \prod_{j \in OIDX} \mathbb{H}((FID_{i,j}, TS_{i,j}, RID_{i,j}))^{RN_{i,j}}. \tag{11}$$

$$\lambda_B . \prod_{i=1}^{n.k} e\left(HV_B . m_i^{\sum_{j \in OIDX} RN_{i,j} . \mathbb{H}(FID_{i,j}, TS_{i,j}, RID_{i,j})}, RM_i\right) \stackrel{?}{=} TGP_B^r. \tag{12}$$

The correction of equation (12) is demonstrated below.

$$
\begin{aligned}
&\lambda_B.\lambda_B.\prod_{i=1}^{n.k} e\left(HV_B.m_i^{\sum_{j\in OIDX} RN_{i,j}.\mathbb{H}(FID_{i,j},TS_{i,j},RID_{i,j})}, RM_i\right)\\
&= \lambda_B.\prod_{i=1}^{n.k} e(m, RM_i)^{\sum_{j\in OIDX} RN_{i,j}.\mathbb{H}(b_{i,j})}.\prod_{i=1}^{n.k} e\left(\prod_{j\in OIDX}\mathbb{H}((FID_{i,j},\right.\\
&\left.TS_{i,j}, RID_{i,j}))^{RN_{i,j}}.m_i^{\sum_{i\in OIDX} RN_{i,j}.\mathbb{H}(FID_{i,j},TS_{i,j},RID_{i,j})}, RM_i\right)\\
&= \prod_{i=1}^{n.k} e\left(m^{\sum_{j\in OIDX} RN_{i,j}.\mathbb{H}(b_{i,j})}, RM_i\right).\prod_{i=1}^{n.k} e\left(\prod_{j\in OIDX}\mathbb{H}((FID_{i,j},\right.\\
&\left.TS_{i,j}, RID_{i,j}))^{RN_{i,j}}.m_i^{\sum_{j\in OIDX} RN_{i,j}.\mathbb{H}(FID_{i,j},TS_{i,j},RID_{i,j})}, RM_i\right)\\
&= \prod_{i=1}^{n.k} e\left(\prod_{j\in OIDX}\mathbb{H}((FID_{i,j}, TS_{i,j}, RID_{i,j}))^{RN_{i,j}}\right.\\
&\left.m_i^{\sum_{j\in OIDX} RN_{i,j}.(\mathbb{H}(b_{i,j})+\mathbb{H}(FID_{i,j},TS_{i,j},RID_{i,j}))}, k^{r-s}\right)\\
&= \prod_{i=1}^{n.k} e\left(\prod_{j\in OIDX}(\mathbb{H}((FID_{i,j}, TS_{i,j}, RID_{i,j}))\right.\\
&\left.m_i^{\sum_{j\in OIDX} RN_{i,j}.(b_{i,j}+\mathbb{H}(FID_{i,j},TS_{i,j},RID_{i,j}))}\right)^{RN_{i,j}}, k^{r-s}\Bigg)\\
&= \prod_{i=1}^{n.k} e\left(\prod_{j\in OIDX}\sigma_{i,j}^{RN_i}, k^{r-s}\right)\\
&= e\left(\prod_{i\in OIDX}\sigma_i^{RN_i}, k^s\right)^r\\
&= \prod_{i=1}^{n.k} e\left(\prod_{j\in OIDX}\sigma_i^{RN_i}, l\right)^r\\
&= \prod_{i=1}^{n.k}\left(\prod_{j\in OIDX} TG_i^{RN_{i,j}.r}\right)\\
&= \prod_{i=1}^{n.k} TGP_i^r\\
&= TGP_B^r.
\end{aligned}
$$

5 Security analysis

Our proposed auditing mechanism is resistant against the following attacks.

5.1 Forge attack

We design a security game theory (based on Shacham and Waters, 2008) to prove the resistance against forge attack. Auditor (A_i) sends a $chal = (OIDX = \{odix_i | 1 \leq i \leq k, k \leq n\},\ RN = \{rn_i | i \in OIDX\}, RM)$ to CSP. CSP should response (TGP, λ) for correct auditing of file FID_i using equation (8) at A_i. However, the CSP generates an incorrect response (TGP, λ') using incorrect data blocks $(b'_k | 1 \leq k \leq c, c \leq n)$, where $\lambda = e(m, RM)^{B'}$, $B' = \sum_{k \in OIDX} (\zeta_k)$, and $\zeta_k = RN_k.\mathbb{H}(b'_i)$. The forge generated proof sends to A_i for checking. If this proof is passed by A_i, the CSP wins the game and lunches forge attack successfully; otherwise it fails.

Let CSP wins the game at auditor A_i, i.e., from equation (8), we have

$$e(m, RM)^{\lambda'}.e\left(HV.m^{\sum_{i \in OIDX} RN_i.\mathbb{H}(FID_i, TS_i, RID_i)}, RM\right) = TGP^r. \tag{13}$$

Moreover, the correct proof is (TGP, λ), then we have

$$e(m, RM)^{\lambda}.e\left(HV.m^{\sum_{i \in OIDX} RN_i.\mathbb{H}(FID_i, TS_i, RID_i)}, RM\right) = TGP^r. \tag{14}$$

To verify successfully at A_i, the following equation (15) should be hold.

$$m^{\sum_{i \in OIDX} \zeta_i} = m^{\sum_{i \in OIDX} \lambda} \tag{15}$$

In ζ_i computation, CSP uses forge data (b'_i) which produces different hash value. Thus, the above equation (15) can not hold and CSP lost the game.

5.2 Replacing attack

The replacing-attack game is defined as follows: auditor (A_i) sends a $chal = (OIDX = \{odix_i | 1 \leq i \leq c, c \leq n\},\ RN = \{rn_i | i \in OIDX\}, RM)$ to CSP. To response back, CSP generates an auditing proof (TGP', λ') after replacing the j^{th} index block b_j to the k^{th} index block b_k, where (b_j, $b_k \in OIDX$). CSP launches a replacing attack in the case of passing the generated proof (TGP', λ') at auditor A_i; otherwise, it fails.

As per bi-linear map properties, left side of the equation (8) can be re-written as:

$$\begin{aligned}
&\lambda'.e\left(HV.m^{\sum_{i \in OIDX} RN_i.\mathbb{H}(FID_i, TS_i, RID_i)}, RM\right) \\
&= e(m, RM)^{\sum_{i,j \in OIDX} (RN_i.\mathbb{H}(b_i)) + (RN_j.\mathbb{H}(b_k))} \\
&.e\left(\prod_{i \in OIDX} \mathbb{H}((FID_i, TS_i, RID_i))^{RN_i}.m^{\sum_{i \in OIDX} RN_i.\mathbb{H}(FID_i, TS_i, RID_i)}, RM\right)
\end{aligned}$$

$$= e\left(m^{i,j\in OIDX^{\sum}(RN_i.r_i)+(RN_j.r_k)}, RM\right)$$
$$.e\left(\prod_{i\in OIDX}\mathbb{H}((FID_i,TS_i,RID_i))^{RN_i}.m^{i\in OIDX^{\sum}RN_i.\mathbb{H}(FID_i,TS_i,RID_i)}, RM\right)$$
$$= e\left(\prod_{i\in OIDX}\mathbb{H}((FID_i,TS_i,RID_i))^{RN_i}\right.$$
$$\left..m^{i,j\in OIDX^{\sum}RN_i(r_i+RVI_i)+RN_j(r_k+RVI_j)}, RM\right)$$

Next, the right side of equation (8) can be re-written as:

$$(TGP')^{rm} = \prod_{i\in OIDX}\theta_i^{RN_i.r}.\theta_k^{RN_j.r}$$
$$= e\left(\prod_{i\in OIDX}\sigma_i^{RN_i}.\sigma_k^{RN_j}, RM\right)$$
$$= e\left(\prod_{i\in OIDX}\left(\mathbb{H}(FID_i,TS_i,RID_i).m^{(r_i+\mathbb{H}(FID_i,TS_i,RID_i))}\right)^{RN_i}\right.$$
$$\left.\cdot\prod_{i\in OIDX}\left(\mathbb{H}(FID_k,TS_k,RID_k).m^{(r_k+\mathbb{H}(FID_k,TS_k,RID_k))}\right)^{RN_j}, RM\right)$$
$$= e\left(\prod_{i\in OIDX}\mathbb{H}((FID_i,TS_i,RID_i))^{RN_i}\right.$$
$$\left..m^{i,j\in OIDX^{\sum}RN_i(r_i+RVI_i)+RN_j(r_k+RVI_j)}, RM\right).$$

A successful verification at A_i of the replacing-attack must has RVI_i equals to RVI_k, i.e., $\mathbb{H}(FID_i,TS_i,RID_i) = \mathbb{H}(FID_k,TS_k,RID_k)$. The computation of RID is based on the sequence of data blocks as discussed earlier (Figure 3). Therefore, root id of replacing block index can be identify in our proposed scheme.

5.3 Replay attack

The replay-attack game is defined as follows: auditor (A_i) sends a $chal = (OIDX = \{odix_i|1 \le i \le c, c \le n\},\ RN = \{rn_i|i \in OIDX\}, RM)$ to CSP. CSP generates information (TGP', λ') after substituting a k^{th} block information (b_k) with previously generated k^{th} block information (b_k*) as a proof of the received $chal$. The replay attack is launched in the case of passing the generated proof (TGP', λ') at auditor A_i; otherwise, it fails.

To pass a verification proof $((TGP', \lambda'))$ using old information at A_i, $RVI_{Current}$ root version id using current time stamp of FID_i is equal to RVI_{Old}. It is not possible to generate in our P2P-based DHT public auditing protocol due to replacement of a data chunk can change the root overlay id (RID).

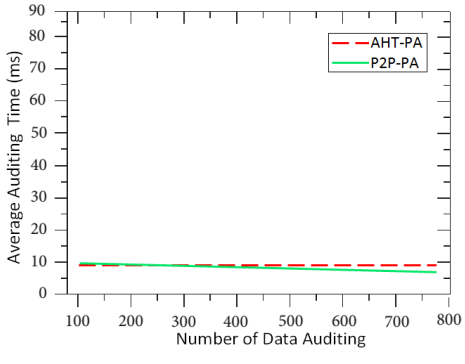
6 Performance analysis

6.1 Computation costs

We chose the Amazon cloud (CSP) to measure the effectiveness of our P2P dynamic auditing scheme on the computation costs during the accomplishment of auditing requests. The configuration of this cloud includes a micro instance elastic-compute cloud (EC2), a variable EC2 compute unit, one CPU having a clock speed of 2.5 GHz, an Intel Xenon Family (physical processor), and a cloud-storage-server with 1 GB memory. We chose five personal computers, and each machine has an Intel Core i5 processor (3.2 GHz) and 4 GB RAM as P2P public auditors. Each P2P auditors maintains a DHT table to maintain files meta-data. We implement the schemes with the help of the pairing-based cryptography (PBC) library (version – 0.5.14) using the C programming language. This library helps to achieve the cryptographic operations with type A pairing parameters.

Figure 4 depicts a result of average computation costs of per auditing work of our scheme and an existing scheme (Chen et al., 2018), where each 10 K files having 50 K blocks. Initially, our scheme has more computation cost, but our scheme performs outstanding as auditing workload increases due to a batch and group auditing with the help of peer auditors.

Figure 4 Comparison of average computation cost of per auditing work from experimental results (see online version for colours)



6.2 Communication costs

We use a theoretical network trafficking model of DHT-based cellular network as defined in Tetarave et al. (2018) to compute the overall bandwidth consumption to audit the outsourced data in cloud computing. This model uses C_w and C_f parameters for wireless and fixed communication cost respectively, where $C_f = k.C_w$ ($k \in \{1, 2, 3, ...\}$). In our experiments, we consider C_w cost to communicate from DO/DU to CSP/auditor, whereas C_f cost within CSP and public auditors. The experiment from this model has been implemented, and its performance evaluation is carried out using MATLAB R2016a on a PC with Intel Core i7 processor with 8 GB of main memory. The comparisons are based on the retrieval of 100 K file chunks among 1 K cloud users.

We consider the number of public auditors is 10, and the number would be increased as per simulation.

Figure 5 Comparison of communication cost as the number of data user increases (see online version for colours)

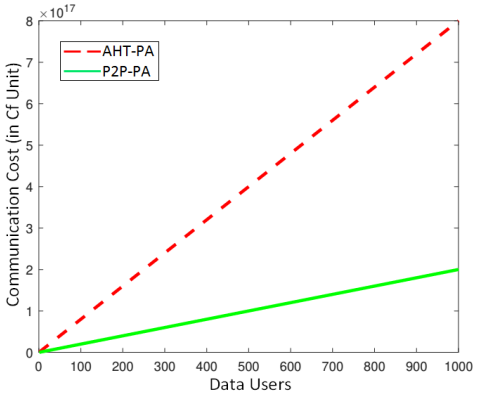
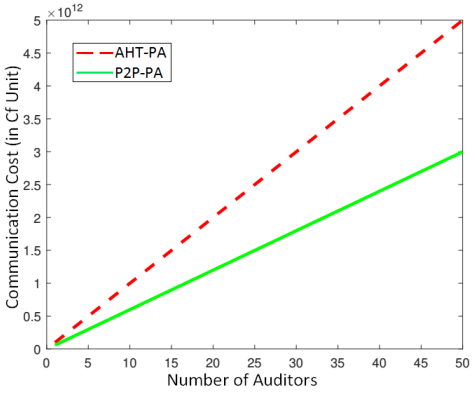


Figure 6 Comparison of communication cost as the number of public auditor increases (see online version for colours)



In traditional auditing schemes, $(N.F.c.k.A_n.(3C_w + 2C_f))$ communication cost $(T_{tradAudit})$ is required for auditing F outsourced files with c number of chunks per file, where N and A_n are the total number of DO/DU and public auditors respectively. In this, each outsourced file chunks $(F.c)$ have been stored, verified, and accessed. On the other hand, few chunks (ϵ) of a file are needed to verify in our proposed DHT-based auditing scheme. In this, request for the verification of file chunks ϵ is carried out from DU to the public auditor. After receiving the request, the target DHT auditor communicates with CSP. Finally, CSP provides the parameters for preserving the privacy of the file. Therefore, three times of fixed communication cost (C_f) are required for data verification, while two times of wireless cost C_w are necessary for data outsourcing and downloading. Thus, the total communication cost in our proposed scheme would be

$$T_{dhtAudit} = N.F.c.A_n.k.2C_w + N.\epsilon.log(A_n).3C_f \tag{16}$$

In Figure 5, $T_{dhtAudit}$ consumes significantly less bandwidth using expensive radio signal. Figure 6 shows the effect of auditors after organising into DHT ring. As the auditors increase, our scheme audits the requested files to distribute the auditing load efficiently.

7 Conclusions

This work presents a new public auditing scheme for efficient and secure cloud storage services. We introduced a P2P mechanism with a DHT data structure to the efficient management of meta-data for each outsourced file. Formal verification of different attacks proves our proposed auditing mechanism mitigates them. Using P2P consistent hash properties and DHT structural benefits make our scheme fewer computation costs and communication overheads from the existing works, which are confirmed through experiments and simulations.

References

Amazon S3 Team et al. (2008) *Amazon S3 Availability Event: July 20, 2008*, November, Vol. 15.

Ateniese, G., Burns, R., Curtmola, R., Herring, J., Kissner, L., Peterson, Z. and Song, D. (2007) 'Provable data possession at untrusted stores', in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ACM, pp.598–609.

Ateniese, G., Pietro, R.D., Mancini, L.V. and Tsudik, G. (2008) 'Scalable and efficient provable data possession', in *Proceedings of the 4th International Conference on Security and Privacy in Communication Networks*, ACM, p.9.

Boneh, D., Gentry, C., Lynn, B. and Shacham, H. (2003) 'Aggregate and verifiably encrypted signatures from bilinear maps', in *International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, pp.416–432.

Boneh, D., Lynn, B. and Shacham, H. (2004) 'Short signatures from the weil pairing', *Journal of Cryptology*, Vol. 17, No. 4, pp.297–319.

Chen, W., Tian, H., Chang, C-C., Nan, F. and Lu, J. (2018) 'Adjacency-hash-table based public auditing for data integrity in mobile cloud computing', *Wireless Communications and Mobile Computing*, Vol. 2018, Article ID 3471312, 12pp.

Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A. and Stoica, I. (2009) *Above the Clouds: A Berkeley View of Cloud Computing*, Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, Vol. 28, No. 13.

Juels, A. and Kaliski Jr., B.S. (2007) 'PORs: proofs of retrievability for large files', in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ACM, pp.584–597.

Karger, D., Lehman, E., Leighton, T., Panigrahy, R., Levine, M. and Lewin, D. (1997) 'Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web', in *Proceedings of the Twenty-Ninth Annual ACM symposium on Theory of Computing*, ACM, pp.654–663.

Krebs, B. (2009) *Payment Processor Breach May Be Largest Ever* [online] http://voices.washingtonpost.com/securityfix/2009/01/payment_processor_breach_may_b.html (accessed 29 January 2020).

Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L. and Leaf, D. (2011) *NIST Cloud Computing Reference Architecture*, NIST Special Publication, Vol. 500, No. 2011, pp.1–28.

Nayak, S.K. and Tripathy, S. (2018a) 'SEMKC: secure & efficient computation over outsourced data encrypted under multiple keys', *IEEE Transactions on Emerging Topics in Computing*, ISSN: 2168-6750, DOI: 10.1109/TETC.2018.2859051.

Nayak, S.K. and Tripathy, S. (2018b) 'SEPDP: secure and efficient privacy preserving provable data possession in cloud storage', *IEEE Transactions on Services Computing*, ISSN: 1939-1374, DOI: 10.1109/TSC.2018.2820713.

Shacham, H. and Waters, B. (2008) 'Compact proofs of retrievability', in *International Conference on the Theory and Application of Cryptology and Information Security*, Springer, pp.90–107.

Shah, M.A., Swaminathan, R., Baker, M. et al. (2008) 'Privacy-preserving audit and extraction of digital contents', *IACR Cryptology ePrint Archive*, Vol. 186.

Shen, J., Shen, J., Chen, X., Huang, X. and Susilo, W. (2017) 'An efficient public auditing protocol with novel dynamic structure for cloud data', *IEEE Transactions on Information Forensics and Security*, Vol. 12, No. 10, pp.2402–2415.

Stoica, I., Morris, R., Karger, D., Kaashoek, M.F. and Balakrishnan, H. (2001) 'Chord: a scalable peer-to-peer lookup service for internet applications', *ACM SIGCOMM Computer Communication Review*, Vol. 31, No. 4, pp.149–160.

Tan, S. and Jia, Y. (2014) 'NaEPASC: a novel and efficient public auditing scheme for cloud data', *Journal of Zhejiang University Science C*, Vol. 15, No. 9, pp.794–804.

Tetarave, S.K., Tripathy, S. and Ghosh, R.K. (2018) 'GMP2P: mobile P2P over GSM for efficient file sharing', in *International Conference on Distributed Computing and Internet Technology*, Springer, pp.217–231.

Tian, H., Chen, Y., Chang, C-C., Jiang, H., Huang, Y., Chen, Y. and Liu, J. (2015) 'Dynamic-hash-table based public auditing for secure cloud storage', *IEEE Transactions on Services Computing*, Vol. 10, No. 5, pp.701–714.

Wang, C., Chow, S.S.M., Wang, Q., Ren, K. and Lou, W. (2011) 'Privacy-preserving public auditing for secure cloud storage', *IEEE Transactions on Computers*, Vol. 62, No. 2, pp.362–375.

Wang, Q., Wang, C., Li, J., Ren, K. and Lou, W. (2009) 'Enabling public verifiability and data dynamics for storage security in cloud computing', in *European Symposium on Research in Computer Security*, Springer, pp.355–370.

Wang, Q., Wang, C., Ren, K., Lou, W. and Li, J. (2010) '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.

Wang, H., Wu, Q., Qin, B. and Domingo-Ferrer, J. (2013) 'Identity-based remote data possession checking in public clouds', *IET Information Security*, Vol. 8, No. 2, pp.114–121.

Yang, K. and Jia, X. (2012) '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.

Zhang, X., Wuwong, N., Li, H. and Zhang, X. (2010) 'Information security risk management framework for the cloud computing environments', in *2010 10th IEEE International Conference on Computer and Information Technology*, IEEE, pp.1328–1334.

Zhu, Y., Ahn, G-J., Hu, H., Yau, S.S., An, H.G. and Hu, C-J. (2011) 'Dynamic audit services for outsourced storages in clouds', *IEEE Transactions on Services Computing*, Vol. 6, No. 2, pp.227–238.