SEPDP: Secure and Efficient Privacy Preserving Provable Data Possession in Cloud Storage

Sanjeet Kumar Nayak, *Student Member, IEEE,* and Somanath Tripathy, *Senior Member, IEEE*

Abstract—Cloud computing is an emergent paradigm to provide reliable and resilient infrastructure enabling the users (data owners) to store their data and the data consumers (users) can access the data from cloud servers. This paradigm reduces storage and maintenance cost of the data owner. At the same time, the data owner loses the physical control and possession of data which leads to many security risks. Therefore, auditing service to check data integrity in the cloud is essential. This issue has become a challenge as the possession of data needs to be verified while maintaining the privacy. To address these issues this work proposes a secure and efficient privacy preserving provable data possession (SEPDP). Further, we extend SEPDP to support multiple owners, data dynamics and batch verification. The most attractive feature of this scheme is that the auditor can verify the possession of data with low computational overhead.

Index Terms—Integrity verification, Storage-as-a-Service, Privacy preserving, Dynamic auditing, Batch auditing.

1 Introduction

Storage-as-a-service has emerged as a commercial alternative for local data storage due to its characteristics include less initial infrastructure setup, relief from maintenance overhead and universal access to the data irrespective of location and device. Though it provides several benefits like cost saving, accessibility, usability, syncing and sharing, it raises several security threats as data is under the control of the cloud service provider (CSP). CSP can discard the rarely accessed data to save space and earn more profit, or it can lie about the data loss and data corruption, as a result of software/ hardware failure to protect its reputation. Therefore, it is necessary to check the possession of data in the cloud storage [1], [2].

Traditional cryptographic solutions for integrity checking of data, either need a local copy of the data (which the data users (DUs) do not have) or allow the DUs to downloads the entire data. Neither of these solutions seems practical as earlier one requires extra storage and later alternative increases the file transfer cost. To address this issue, several schemes including [3], [4], [5], [6] are proposed which employ blockless verification to verify the integrity without downloading the entire data. One of the attractive features of these works is to allow the public verifier to verify. With public auditability, DUs can recourse the auditing task to a third party auditor (TPA). It has expertise and capabilities to convince both the CSP and the DU [4], [7]. These schemes use provable data possession (PDP) technique, which gives probabilistic data possession guarantee by randomly verifying few blocks for ensuring possession of data in the untrusted cloud storage.

Recently, several schemes [2], [3], [4], [5], [6], [8], [9], [10], [11], [11], [12], [13], [14], [15] have been proposed to allow TPA to check integrity of the data stored on the untrusted cloud. These schemes have their own pros and cons.

Privacy preserving is essential to prevent TPA to infer the data using the cloud server's response while auditing. However, the schemes proposed in [2], [3] do not achieve privacy preserving requirement. Though data dynamics is an important feature to facilitate the data owners to insert, modify, and delete on a particular block of data, without changing the meta-data of other blocks, the techniques proposed in [3], [4] do not achieve data dynamics requirement. Meanwhile, the schemes like [3], [10], [16] could not achieve batch auditing requirement which ensures that TPA should be capable enough to deal with the multiple numbers of simultaneous verification requests from different DUs. This property is to save computation and communication cost between CSP and TPA. Unfortunately, the schemes [2], [3], [4], [11], [12], [13], [15], [16] use pairing based cryptographic operations which are intensive computation and need more time.

In this work, we propose a secure and efficient privacy preserving provable data possession scheme (SEPDP) for cloud storage. It operates in three phases, namely, key generation, signature generation and auditing phase. Most attractive feature of SEPDP is that it does not use any intensive computation like pairing based operation. Further, we extend SEPDP to support multiple data owners, batch auditing, and dynamic data operations. A probabilistic analysis to detect the integrity of the blocks stored at CSP. We evaluated the performance of the proposed scheme and compared with some of the existing popular mechanisms. We observe that the total time for verification carried out by TPA in the proposed scheme is less than that of the existing schemes. This signifies that SEPDP is efficient and suitable to implement the verification at the low powered devices.

Remainder of this paper is organized as follows. Section 2 discusses the overview of related works in this field. System model and design goals are presented in Section 3. The proposed scheme is discussed in Section 4. Extension of SEPDP to support multiple DOs, batch auditing and data dynamics requirements are explained in Section 5, 6 and

• S. K. Nayak and S. Tripathy are with the Department of Computer Science and Engineering, Indian Institute of Technology Patna, India, 801106. E-mail: {sanjeet.pcs13, som}@iitp.ac.in

7 respectively. Security analysis of the proposed SEPDP is performed in Section 8. SEPDP is evaluated in terms of performance in Section 9. The concluding remarks are provided in Section 10.

2 Related Work

Remote data integrity checking protocols can be broadly categorized into two kinds. The deterministic guarantee based schemes like [17] [18] and [19], verify each block of data and therefore require a significant amount of storage and computation. Alternative kind of schemes called provable data possession (PDP) include [8], [3], [20] use probabilistic checking method, in which a few blocks are randomly selected to detect manipulation. PDP is introduced in [8], that uses random sampling of a few blocks for integrity verification. Shacham et al. [3] designed two different integrity verification mechanisms. One uses pseudo-random function (PRF) which fails to provide public verifiability, while the other one uses boneh–lynn–shacham (BLS) signatures [20]. Both the schemes support blockless verification but fail to provide privacy of the DO's data. Blockless verification requires linear combination of sampled blocks which gives a clue to TPA to extract the data [4]. To preserve privacy of the data owner supporting blockless verification, Wang et al. [4] proposed a public auditing scheme and extended that to support batch auditing further. As a result, TPA can simultaneously perform multiple auditing requests from different DUs. But, all these schemes [3], [4], [8] fail to support data dynamics. Moreover, as signatures of the data blocks contain index number of the corresponding blocks, if one block is updated (inserted/modified/deleted), the corresponding verification meta-data (signature) of all other blocks need to be updated. The scheme proposed in [16] uses index hash table (IHT) to support data dynamics in public auditing mechanism reducing the update overhead. Unfortunately, this scheme fails to support batch auditing property. later on, Wang et al. [7] extended their previous technique [4] to support data dynamics. Yang et al. [11] proposed an efficient and secure dynamic auditing protocol that achieves all essential features of public auditing. Also it consumes lesser computation and communication cost. A certificateless public auditing scheme for verifying data integrity in the cloud is proposed by Wang et al. [2]. Although this scheme does not require certificate for key generation, it fails to achieve privacy, data dynamics, and batch auditing properties. But, [2], [3], [4], [8], [11], [15], [16] schemes are based on pairing based cryptography, which requires more verification cost in audit phase.

3 System Model and Design Goals

3.1 System Model

A typical cloud data storage model for public auditing as depicted in Figure 1, consists of four entities, namely, data owner (DO), data user (DU), cloud service provider (CSP), and a third party auditor (TPA). Data owners are the entities who store their data in the cloud. Data users access and operate on those data kept at CSP. But, operating on the incorrect data lead to faulty result and create chaos which necessitate the integrity verification of remotely stored data. Therefore, the system consists of a third party auditor (TPA) to verify the integrity of outsourced data.

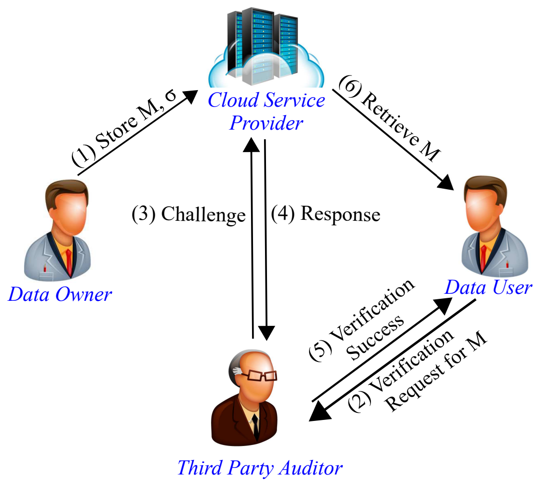


Fig. 1. Cloud data storage architecture for public auditing.

Initially, DO shares a secret key with TPA through a secure channel using any standard technique like SSL/TLS. Every block of the outsourced data (m_i) is tagged with a signature (σ_i) computed using the private key of DO. In the auditing phase, TPA sends a challenge to CSP and CSP returns a response to proof possession of the data. Thus, the public auditing schemes are a kind of challenge-response protocol.

CSP is assumed to be semi-trusted. It executes the protocol without polluting data integrity actively. At the same time, it may lie about the incorrectness of the data to save its reputation. Further, we consider that neither DU nor third party auditor is colluded with CSP to falsify the integrity check.

3.2 Design Goals

Following design goals are desired to enable remote data auditing in above system model.

1) *Guarantee for Storage Correctness:* CSP can pass the audit phase only if it possesses the outsourced data intact (same as uploaded by DO).
2) *Guarantee for Privacy Preserving:* TPA fails to infer the data m_i from the response(s) provided by CSP.
3) *Blockless Verification:* Auditor can be able to verify the integrity of all the desired blocks at once by checking a block (linear combination of all those blocks). This is to reduce the bandwidth consumption.
4) *Public Auditability:* Any third party other than DU should be able to correctly verify the integrity of the data stored in CSP without downloading the entire outsourced data.
5) *Guarantee for Unforgeability:* It must be computationally infeasible for CSP to forge a response in the auditing phase.
6) *Batch Auditing:* TPA should be capable enough to deal with the multiple number of verification requests from different DUs simultaneously. This

feature saves both the computation cost of TPA as well as bandwidth consumption between CSP and TPA.

7) *Data Dynamics:* The scheme should facilitate the data owners to perform insert, modify, and delete operations on a particular block of data, without changing meta-data of other blocks.

4 The Proposed scheme (SEPDP)

In this section, we present the proposed secure and efficient data possession scheme (*SEPDP*). SEPDP achieves all the design goals discussed in previous section. SEPDP consists of three phases, namely, key generation phase, signature generation phase, and audit phase. The operations of these phases are depicted in Figure 2 and discussed below. For the sake of simplicity, we describe the scheme with a single DO and extend the scheme to support multiple DOs in Section 5. Notations used in this work are stated in Table 1. G, g, p and $\mathbb{H}_{(.)}(.)$ are system wide parameters and available to all the entities.

TABLE 1
Notations used in proposed SEPDP

Notations	Meaning
p	Large prime in Z_p^*
g	Primitive Element in G
$\mathbb{H}_{(\mathbf{k})}(.)$	Keyed-hash function
$x \overset{R}{\leftarrow} X$	x is randomly selected from X
$a\|\|b$	a is concatenated with b

(i) Key Generation Phase

DO chooses a large prime number p such that computing discrete logarithm problem (DLP) is intractable in Z_p^*. Let G be a group of large prime order p and $g \in G$ is the primitive element. DO chooses a keyed-hash function, denoted as $\mathbb{H}_{(\mathbf{k})}(.)$, defined as $\{0,1\}^* \times \mathcal{K} \rightarrow {Z_p}^*$. She shares the key $\mathbf{k} \in \mathcal{K}$ with TPA through a secure channel. Also, she selects a random number $x \overset{R}{\leftarrow} {Z_p}^*$ as private key (SK), calculates $Y\ (= g^x)$ as public key and publishes.

(ii) Signature Generation Phase

In this phase, DO splits the file M into n blocks as $M = (m_1, m_2, ..., m_n)$, where $m_i \in {Z_p}^*$. She signs all the n blocks after choosing a secret random number $r \overset{U}{\leftarrow} {Z_p}^*$. The signature $\sigma\langle R, s_i\rangle$ is computed as

$$R = g^r \tag{1}$$

$$s_i = (m_i - \mathbb{H}_{\mathbf{k}}(R||i)x)r^{-1}, i = 1, 2, ..., n \tag{2}$$

and uploads M and σ to the CSP.

(iii) Auditing Phase

In this phase, TPA randomly selects a subset (with c elements) of set $[1, n]$. This set is represented as Q. For $i \in Q$ it will generate a random $v_i \in {Z_q}^*$ where $q << p$. Now, TPA sends $\{(i, v_i)\}_{i\in Q}$ to the cloud server as challenge.

After receiving the challenge message from TPA, CSP-computes α, β and γ as

$$\alpha = R^{\sum_{i\in Q} v_i s_i} \tag{3}$$

$$\beta = \sum_{i\in Q} v_i m_i \tag{4}$$

$$\gamma = g^{\beta} \tag{5}$$

and returns $(\{\alpha, \gamma, R\})$ as response.

TPA assures the integrity of M using Equation (6).

$$\gamma \overset{?}{=} \alpha Y^{\sum_{i\in Q} \mathbb{H}_{\mathbf{k}}(R||i)v_i} \tag{6}$$

4.1 Correctness

In SEPDP, TPA can verify the integrity of M correctly if Equation (6) holds. R.H.S. of Equation (6) can be simplified as follows.

$$\begin{aligned}
&R^{\sum_{i\in Q} v_i s_i} Y^{\sum_{i\in Q} \mathbb{H}_{\mathbf{k}}(R||i)v_i}\\
&= g^{r(\sum_{i\in Q} v_i s_i)}.g^{x(\sum_{i\in Q} \mathbb{H}_{\mathbf{k}}(R||i)v_i)}\\
&= g^{r(\sum_{i\in Q} v_i((m_i - \mathbb{H}_{\mathbf{k}}(R||i)x)r^{-1}))}.g^{x(\sum_{i\in Q} \mathbb{H}_{\mathbf{k}}(R||i)v_i)}\\
&= g^{(\sum_{i\in Q} v_i(m_i - \mathbb{H}_{\mathbf{k}}(R||i)x))}.g^{x(\sum_{i\in Q} \mathbb{H}_{\mathbf{k}}(R||i)v_i)}\\
&= g^{\sum_{i\in Q} v_i m_i}.g^{-x(\sum_{i\in Q} \mathbb{H}_{\mathbf{k}}(R||i)v_i)}.g^{x(\sum_{i\in Q} \mathbb{H}_{\mathbf{k}}(R||i)v_i)}\\
&= g^{\sum_{i\in Q} v_i m_i}\\
&= g^{\beta}\\
&= \gamma\\
&= \text{L.H.S.}
\end{aligned}$$

4.2 Blockless Verification

Blockless verification states that, if CSP has n blocks and corresponding signatures s_i, then TPA can verify the integrity of all the n blocks by verifying a random combination of those.

How it Works?

Let us consider an example for $(n =)2$ blocks of messages (m_1, m_2). In auditing phase, TPA chooses two random numbers v_1 and v_2 $(v_i \in {Z_q}^*)$ and sends those to CSP. Consequent upon receiving v_1 and v_2, CSP computes α and β using Equation (3) and (4) as follows.

$$\alpha = R^{v_1 s_1 + v_2 s_2} \tag{7}$$

$$\beta = v_1 m_1 + v_2 m_2 \tag{8}$$

It sends α and $\gamma\ (= g^{\beta})$ to TPA. It can verify the correctness of the two blocks without requiring m_1 or m_2 as follows.

$$\gamma \overset{?}{=} R^{v_1 s_1 + v_2 s_2} Y^{\mathbb{H}_{\mathbf{k}}(R||1)v_1 + \mathbb{H}_{\mathbf{k}}(R||2)v_2} \tag{9}$$

Similarly, we can extend the proof for n number of blocks and can show that using a single random combination of all the n blocks we can check their integrity without the knowledge of m_i separately. Hence, the proposed SEPDP is blockless verifiable.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2820713, IEEE Transactions on Services Computing

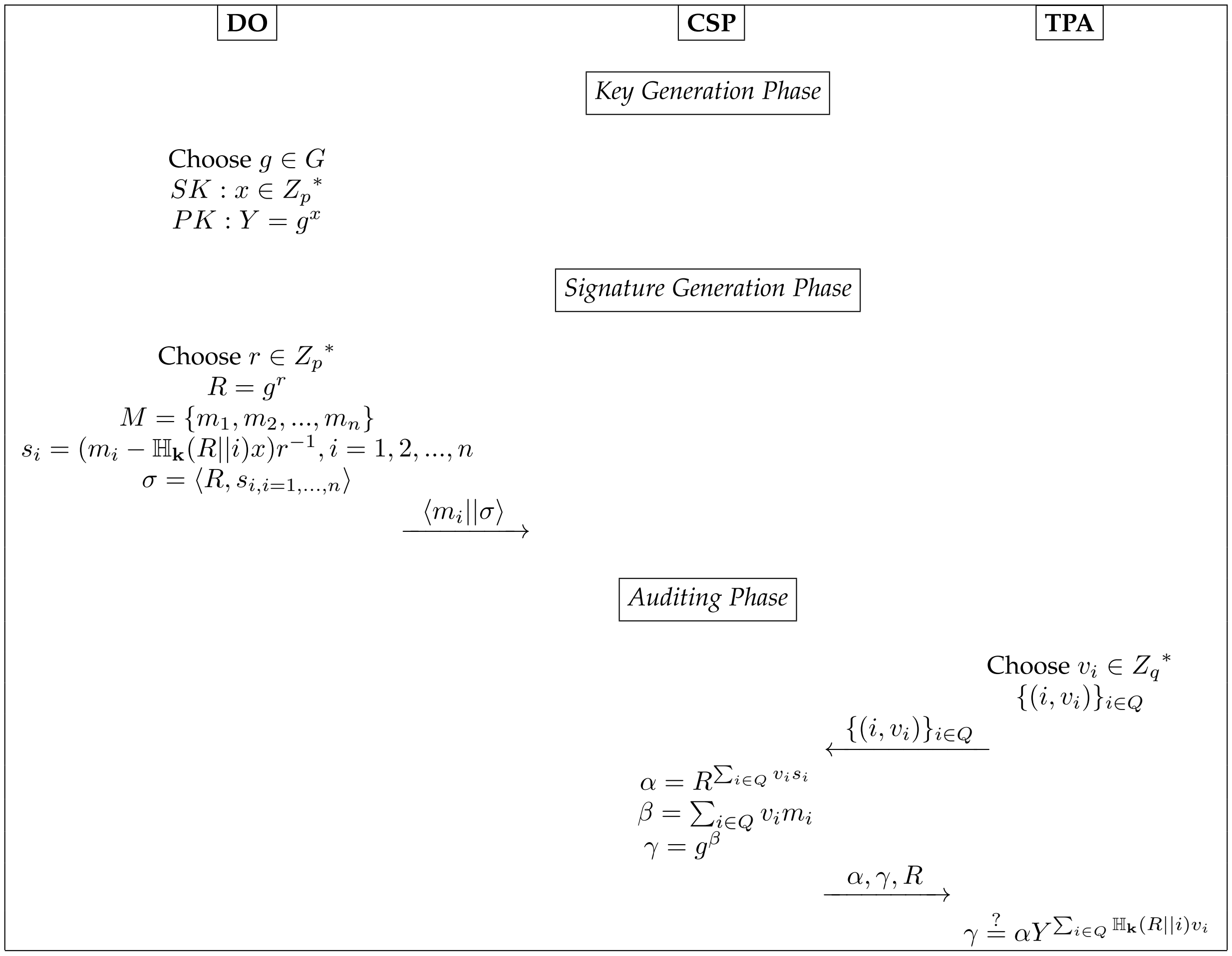

DO
CSP
TPA
Key Generation Phase
Choose $g \in G$
$SK : x \in {Z_p}^*$
$PK : Y = g^x$
Signature Generation Phase
Choose $r \in {Z_p}^*$
$R = g^r$
$M = \{m_1, m_2, ..., m_n\}$
$s_i = (m_i - \mathbb{H}_{\mathbf{k}}(R||i)x)r^{-1}, i = 1, 2, ..., n$
$\sigma = \langle R, s_{i,i=1,...,n} \rangle$
$\langle m_i || \sigma \rangle$
Auditing Phase
Choose $v_i \in {Z_q}^*$
$\{(i, v_i)\}_{i \in Q}$
$\{(i, v_i)\}_{i \in Q}$
$\alpha = R^{\sum_{i \in Q} v_i s_i}$
$\beta = \sum_{i \in Q} v_i m_i$
$\gamma = g^{\beta}$
α, γ, R
$\gamma \overset{?}{=} \alpha Y^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R||i) v_i}$

Fig. 2. Schematic diagram of proposed SEPDP.

5 Extension for Multiple DOs

Here, we extend SEPDP to support multiple data owners. Such a model is depicted in Figure 3 in which each data owner has its own public key and private key. Each DO signs their corresponding data and stores both data and signatures in the CSP. TPA uses challenge-response mechanism to check integrity of the data stored in CSP. Operation of the scheme is depicted in Figure 4 and discussed below.

(i) Key Generation Phase

Let d be the number of DOs present in the cloud storage system. During this phase, j^{th} data owner (DO$_j$) shares a key $\mathbf{k}_j \in \mathcal{K}$ with the TPA. She selects a unique random number $x_j \in {Z_p}^*$ as her private key ($SK_j, j = [1, d]$). Then she calculates Y_j ($= g^{x_j}$) and publishes it as public key.

(ii) Signature Generation Phase

In this phase, DO$_j$ signs $m_{j,i} \in {Z_p}^*$ resulting $s_{j,i} = (m_{j,i} - \mathbb{H}_{\mathbf{k}_j}(R||i)x_j)r^{-1}$ and $R = g^r$. She uploads $m_{j,i}$ and $\sigma_j = \langle R, s_{j,i} \rangle$ to CSP. Here, we assume that r is secretly shared among all the data owners using a secure group key sharing techniques like [21], [22].

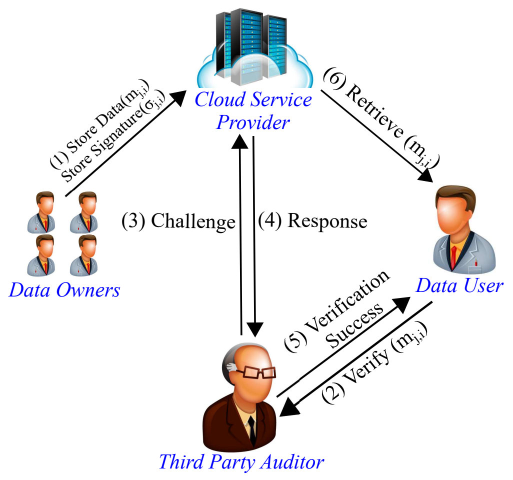


Fig. 3. Cloud data storage architecture for multiple DOs extension.

(iii) Auditing Phase

To audit integrity of M_j, TPA generates a challenge message $\{(i, v_i)_{i \in Q}\}_{j \in [1,d]}$ as in SEPDP. Then, CSP finds α_j, β_j and γ_j for DO$_j$'s data as, $\alpha_j = R^{\sum\limits_{i \in Q} v_i s_{j,i}}$, $\beta_j = \sum\limits_{i \in Q} v_i m_{j,i}$ and $\gamma_j = g^{\beta_j}$. CSP sends $\langle \alpha_j, \gamma_j, R \rangle$ as audit proof.

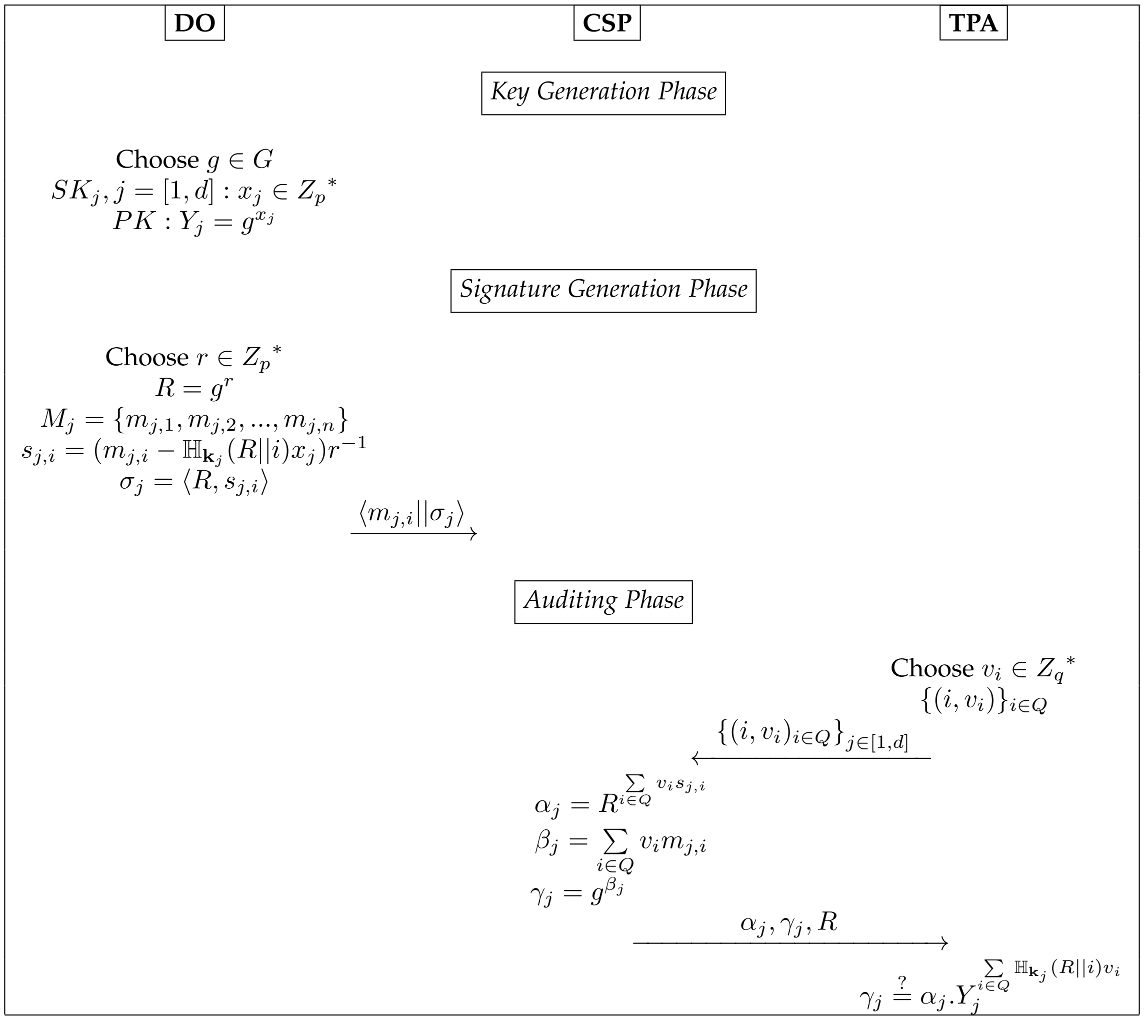

DO
CSP
TPA
Key Generation Phase
Choose $g \in G$
$SK_j, j=[1,d] : x_j \in {Z_p}^*$
$PK : Y_j = g^{x_j}$
Signature Generation Phase
Choose $r \in {Z_p}^*$
$R = g^r$
$M_j = \{m_{j,1}, m_{j,2}, ..., m_{j,n}\}$
$s_{j,i} = (m_{j,i} - \mathbb{H}_{\mathbf{k}_j}(R\|i)x_j)r^{-1}$
$\sigma_j = \langle R, s_{j,i} \rangle$
$\langle m_{j,i} \| \sigma_j \rangle$
Auditing Phase
Choose $v_i \in {Z_q}^*$
$\{(i, v_i)\}_{i \in Q}$
$\{(i, v_i)_{i \in Q}\}_{j \in [1,d]}$
$\alpha_j = R^{\sum_{i \in Q} v_i s_{j,i}}$
$\beta_j = \sum_{i \in Q} v_i m_{j,i}$
$\gamma_j = g^{\beta_j}$
α_j, γ_j, R
$\gamma_j \overset{?}{=} \alpha_j . Y_j^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}_j}(R\|i)v_i}$

Fig. 4. Schematic diagram of multiple DO extension of SEPDP.

TPA verifies the integrity of M_j as

$$\gamma_j \overset{?}{=} \alpha_j . Y_j^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}_j}(R\|i)v_i} \tag{10}$$

5.1 Correctness

In case of multiple DO, TPA can verify integrity of M correctly if Equation (10) holds. R.H.S. of Equation (10) can be simplified as follows.

$$\begin{aligned}
& R^{\sum_{i \in Q} v_i s_{j,i}} . Y_j^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}_j}(R\|i)v_i} \\
&= g^{r \sum_{i \in Q} v_i((m_{j,i} - \mathbb{H}_{\mathbf{k}_j}(R\|i)x_j)r^{-1})} . g^{x_j \sum_{i \in Q} \mathbb{H}_{\mathbf{k}_j}(R\|i)v_i} \\
&= g^{\sum_{i \in Q} v_i m_{j,i}} . g^{-x_j \sum_{i \in Q} v_i \mathbb{H}_{\mathbf{k}_j}(R\|i)} . g^{x_j \sum_{i \in Q} v_i \mathbb{H}_{\mathbf{k}_j}(R\|i)} \\
&= g^{\sum_{i \in Q} v_i m_{j,i}} \\
&= g^{\beta_j} \\
&= \gamma_j \\
&= \text{L.H.S.}
\end{aligned}$$

6 Extension for Batch Auditing

Sometimes requests from many data users arise for integrity verification within a short span of time. Verifying these requests by TPA one by one would not be a wise decision.

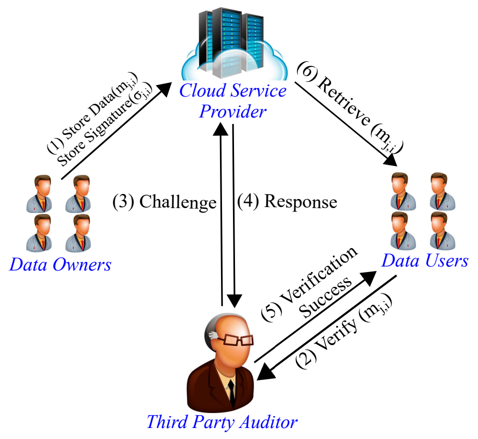


Fig. 5. Cloud data storage architecture for batch auditing extension.

Batch auditing is the process in which TPA verifies multiple auditing requests submitted by many DUs (in a batch) at a time which saves computation time. This can be achieved by aggregating d verification equations (requested by different DUs) like Equation (6) into one verification equation. Such a model is depicted in Figure 5, in which

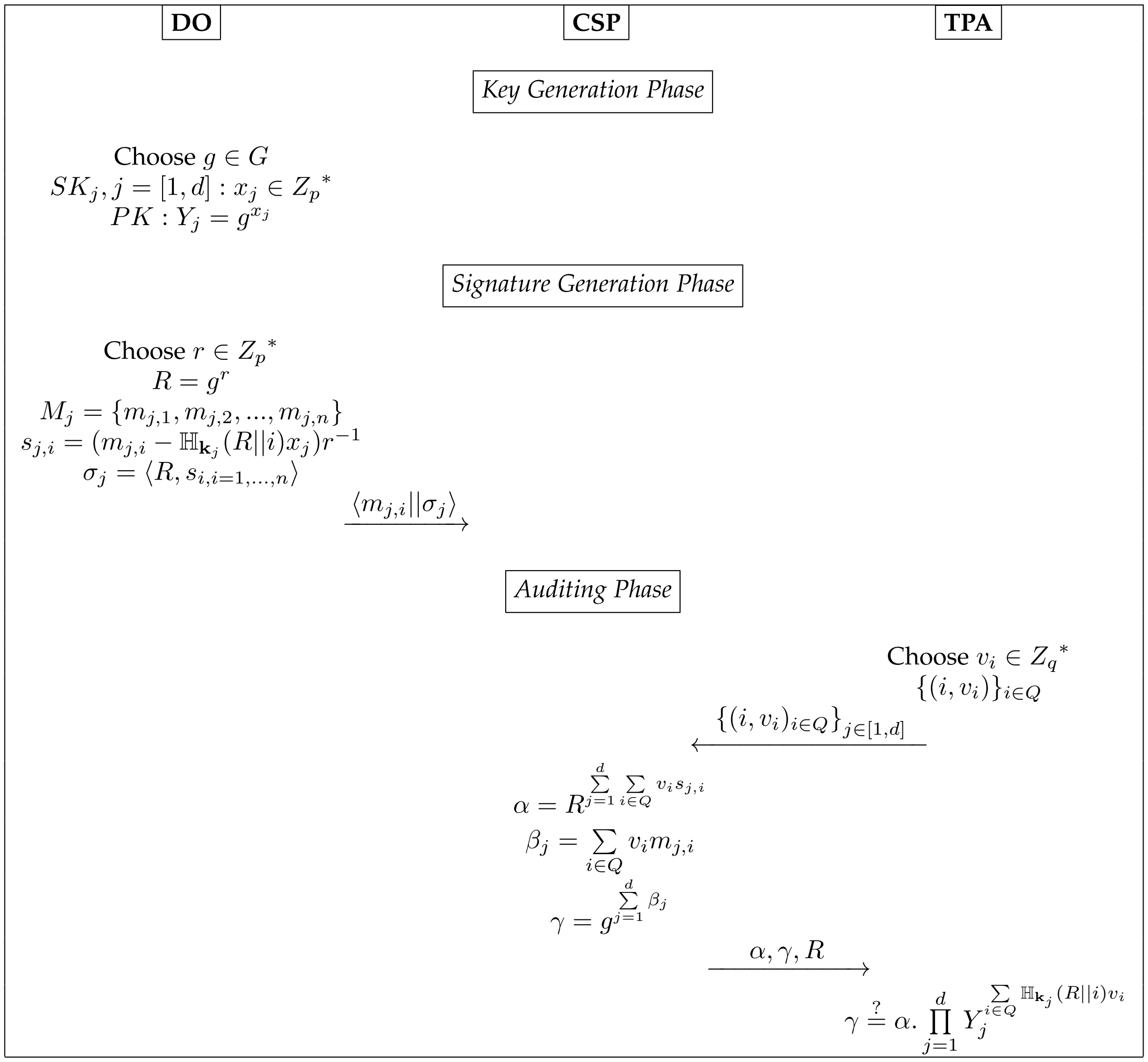

DO
CSP
TPA
Key Generation Phase
Choose $g \in G$
$SK_j, j = [1, d] : x_j \in {Z_p}^*$
$PK : Y_j = g^{x_j}$
Signature Generation Phase
Choose $r \in {Z_p}^*$
$R = g^r$
$M_j = \{m_{j,1}, m_{j,2}, ..., m_{j,n}\}$
$s_{j,i} = (m_{j,i} - \mathbb{H}_{\mathbf{k}_j}(R||i)x_j)r^{-1}$
$\sigma_j = \langle R, s_{i,i=1,...,n}\rangle$
$\langle m_{j,i}||\sigma_j\rangle$
Auditing Phase
Choose $v_i \in {Z_q}^*$
$\{(i, v_i)\}_{i\in Q}$
$\{(i, v_i)_{i\in Q}\}_{j\in[1,d]}$
$\alpha = R^{\sum_{j=1}^{d}\sum_{i\in Q} v_i s_{j,i}}$
$\beta_j = \sum_{i\in Q} v_i m_{j,i}$
$\gamma = g^{\sum_{j=1}^{d}\beta_j}$
α, γ, R
$\gamma \overset{?}{=} \alpha . \prod_{j=1}^{d} Y_j^{\sum_{i\in Q}\mathbb{H}_{\mathbf{k}_j}(R||i)v_i}$

Fig. 6. Schematic diagram of batch auditing extension of SEPDP.

each data owner has its own public key and private key. DO signs their corresponding data and stores both data and signatures in CSP. TPA uses challenge-response mechanism to check integrity of data stored in CSP. The operation of the scheme is depicted in Figure 6 and discussed below.

(i) Key Generation Phase

Let d be the number of DOs present in the cloud storage system. During this phase, j^{th} data owner (DO$_j$) shares a key $\mathbf{k}_j \in \mathcal{K}$ with the TPA. She selects random numbers $x_j \in {Z_p}^*$ as their private key ($SK_j, j = [1, d]$). Then she calculates Y_j $(= g^{x_j})$ and publishes publishes it as public key.

(ii) Signature Generation Phase

In this phase, DO$_j$ signs $m_{j,i} \in {Z_p}^*$ resulting $s_{j,i} = (m_{j,i} - \mathbb{H}_{\mathbf{k}_j}(R||i)x_j)r^{-1}$ and $R = g^r$. It uploads $m_{j,i}$ and $\sigma_j = \langle R, s_{j,i}\rangle$ to CSP. Here, we assume that r is secretly shared among all the data owners using a secure group key sharing techniques like [21], [22].

(iii) Auditing Phase

To check the integrity of M_j, TPA generates a challenge message $\{(i, v_i)_{i\in Q}\}_{j\in[1,d]}$ as mentioned in Section 4. Then, CSP finds $\alpha = R^{\sum_{j=1}^{d}\sum_{i\in Q} v_i s_{j,i}}$, $\beta_j = \sum_{i\in Q} v_i m_{j,i},\ j]_{i=1}^{d}$ and $\gamma = g^{\sum_{j=1}^{d}\beta_j}$. CSP sends $\{\alpha, \gamma, R\}$ as audit proof.

TPA verifies the integrity of M_j as

$$\gamma \overset{?}{=} \alpha . \prod_{j=1}^{d} Y_j^{\sum_{i\in Q}\mathbb{H}_{\mathbf{k}_j}(R||i)v_i} \tag{11}$$

6.1 Correctness

In case of batch auditing extension of SEPDP, TPA can verify the integrity of M_j correctly if Equation (11) holds. R.H.S. of

Equation (11) can be simplified as follows.

$$
\begin{aligned}
& R^{\sum\limits_{j=1}^{d}\sum\limits_{i\in Q} v_i s_{j,i}} \cdot \prod_{j=1}^{d} Y_j^{\sum\limits_{i\in Q}\mathbb{H}_{\mathbf{k}_j}(R||i)v_i} \\
&= g^{r\sum\limits_{j=1}^{d}\sum\limits_{i\in Q} v_i((m_{j,i}-\mathbb{H}_{\mathbf{k}_j}(R||i)x_j)r^{-1})} \cdot \prod_{j=1}^{d} g^{x_j\sum\limits_{i\in Q}\mathbb{H}_{\mathbf{k}_j}(R||i)v_i} \\
&= g^{\sum\limits_{j=1}^{d}\sum\limits_{i\in Q} v_i m_{j,i}} . g^{-\sum\limits_{j=1}^{d} x_j(\sum\limits_{i\in Q} v_i\mathbb{H}_{\mathbf{k}_j}(R||i))} . g^{\sum\limits_{j=1}^{d} x_j(\sum\limits_{i\in Q} v_i\mathbb{H}_{\mathbf{k}_j}(R||i))} \\
&= g^{\sum\limits_{j=1}^{d}\sum\limits_{i\in Q} v_i m_{j,i}} \\
&= g^{\sum\limits_{j=1}^{d}\beta_j} \\
&= \gamma \\
&= \text{L.H.S.}
\end{aligned}
$$

7 Extension for Dynamic Data Operation

Data owners not only access the data but also dynamically update (modify/insert/delete) it. In traditional integrity checking mechanisms, data blocks are mapped with index number. During the dynamic update of a single block, verification meta-data of unmodified blocks are also updated because the mapping of data block with index number is changed. So, efficient mechanism for public auditing scheme with dynamic data operations is required. Along with this, the scheme should be secure against forgery of verification meta-data. In this section, SEPDP is extended to support dynamic data operations. The scheme is depicted in Figure 7 and discussed below.

(i) Key Generation Phase

The DO shares a key $\mathbf{k} \in \mathcal{K}$ with the TPA and selects a random number $x \overset{R}{\leftarrow} {Z_p}^*$ as her private key (SK). She calculates $Y\,(= g^x)$ and publishes publishes it as public key.

(ii) Signature Generation Phase

In this phase, M is split into n blocks $\{m_1, m_2, ..., m_n\}$ and each block m_i is split into $\mathbb{s}$ sectors as $\{m_{i,1}, m_{i,2}, ..., m_{i,\mathbb{s}}\}$. The signature s_i on block m_i is computed using $\mathbb{s}$ random secrets $(\tau_1, \tau_2, ..., \tau_{\mathbb{s}}) \in Z_p(= \tau)$. $u_j]_{j=1}^{\mathbb{s}}$ is computed as g^{τ_j}. The DO builds index-hash table (IHT) Ψ as $\psi]_{i=1}^{n} = \{B_i = i, V_i = 1, R_i \in \{0,1\}^*\}$. IHT keeps the record of the changes in the blocks. It is used in the generation of verification meta-data. Records of IHT consists of serial number (S No.), block number (B_i), version number (V_i), and random integer (R_i). The unique combination of the table ($B_i||V_i||R_i$) is used for the generation of the verification meta-data. Verification meta-data (signature $\sigma = \langle R, s_i\rangle$) is computed as $s_i]_{i=1}^{n} = \left(\sum\limits_{j=1}^{\mathbb{s}} \tau_j.m_{i,j} + z_i^1 - \mathbb{H}_{\mathbf{k}}(R)x\right) r^{-1}$ and $R = g^r$. Here, $z_i^1 = \mathcal{H}_z(\psi_i)$, $z = \sum\limits_{j=1}^{\mathbb{s}} \tau_j$ and $\mathcal{H}$ is a collision resistant secure hash function. Finally, block-signature pairs are send to CSP and $u_j]_{j=1}^{\mathbb{s}}$, z and Ψ are send to the TPA.

(iii) Auditing Phase

In this phase, TPA generates a challenge message $(\{(i, v_i)\}_{i\in Q})$ as mentioned in Section 4 and CSP generates a response message as $\{\alpha, \beta, R\}$ where $\alpha = R^{\sum_{i\in Q} v_i s_i}$, $\beta = \beta_j]_{j=1}^{\mathbb{s}}$ and $\beta_j = \sum_{i\in Q} v_i m_{i,j}$.

TPA verifies Equation (12) to check the integrity of M with dynamic data operations.

$$
\prod_{j=1}^{\mathbb{s}} u_j^{\beta_j} . g^{\sum\limits_{i\in Q} v_i z_i^1} = \alpha . Y^{\sum\limits_{i\in Q}\mathbb{H}_{\mathbf{k}}(R)v_i} \tag{12}
$$

(iv) Data Updation Phase

To support dynamic data operation, SEPDP is extended to allow DO to update (modify/insert/delete) on M (outsourced). To perform this, DO updates the corresponding signatures and corresponding entry of IHT. It sends the signature and the corresponding file (not required in case of delete operation) to CSP and updated entry of IHT to TPA. After this, the auditing phase can be carried out to check the integrity of the updated data. A diagrammatic representation of this phase is depicted in Figure 8 and discussed below.

TPA keeps the real time status of dynamic data operations in Ψ. To update (modify/delete/insert) a data block in M, DO requests to TPA for i^{th} entry of Ψ, which is provided consequently.

Modification: If DO wants to modify a block, she increases V_i by one and chooses a new R_i for i^{th} entry in Ψ. Hence, z_i^1 is recomputed using the updated values of the i^{th} entry in Ψ. She re-computes s_i with updated m_i as $s_i^{'} = \left(\sum\limits_{j=1}^{\mathbb{s}} \tau_j.m_{i,j}^{'} + z_i^1 - \mathbb{H}_{\mathbf{k}}(R)x\right) r^{-1}$ and sends $\psi_i{'}$ (updated i^{th} entry of IHT) to TPA and $(s_i^{'}, m_i^{'})$ to CSP.

Insertion: Similarly, when a new block is inserted, i^{th} position is created in Ψ. Other records are shifted by one place in Ψ. B_i is modified, V_i is set to 1 and a new R_i is chosen to constitute a new i^{th} entry in Ψ. z_i^1 is updated and s_i is re-computed with new m_i as $s_i^{'} = \left(\sum\limits_{j=1}^{\mathbb{s}} \tau_j.m_{i,j}^{'} + z_i^1 - \mathbb{H}_{\mathbf{k}}(R)x\right) r^{-1}$. She sends $\psi_i{'}$ (updated i^{th} entry of IHT) to the TPA and $(s_i^{'}, m_i^{'})$ to the CSP.

Deletion: After receiving IHT, DO replaces the version number with 0 in the i^{th} position of the IHT. z_i^1 is updated and s_i is re-computed as $s_i^{'} = \left(z_i^1 - \mathbb{H}_{\mathbf{k}}(R)x\right) r^{-1}$. She sends $\psi_i{'}$ (updated i^{th} entry of IHT) to the TPA and $s_i^{'}$ to the CSP.

As TPA receives $\psi_i{'}$, it replaces the original i^{th} entry of IHT with the new one. Following receipt of the updated signature and the block, CSP checks validity of the updated data as follows. For modify and insert operation, CSP verifies the equation $\prod\limits_{j=1}^{\mathbb{s}} u_j^{m_{i,j}^{'}} . g^{z_i^1} \overset{?}{=} R^{s_i^{'}} . Y^{\mathbb{H}_{\mathbf{k}}(R)}$ and for delete operation, it verifies the equation $g^{z_i^1} \overset{?}{=} R^{s_i^{'}} . Y^{\mathbb{H}_{\mathbf{k}}(R)}$.

7.1 Correctness

In case of extended SEPDP to support dynamic data operations, indeed TPA is able to verify the integrity of M correctly. To prove the correctness of this scheme, we have to show that Equation (12) is valid. R.H.S. of Equation (12) can be simplified as follows.

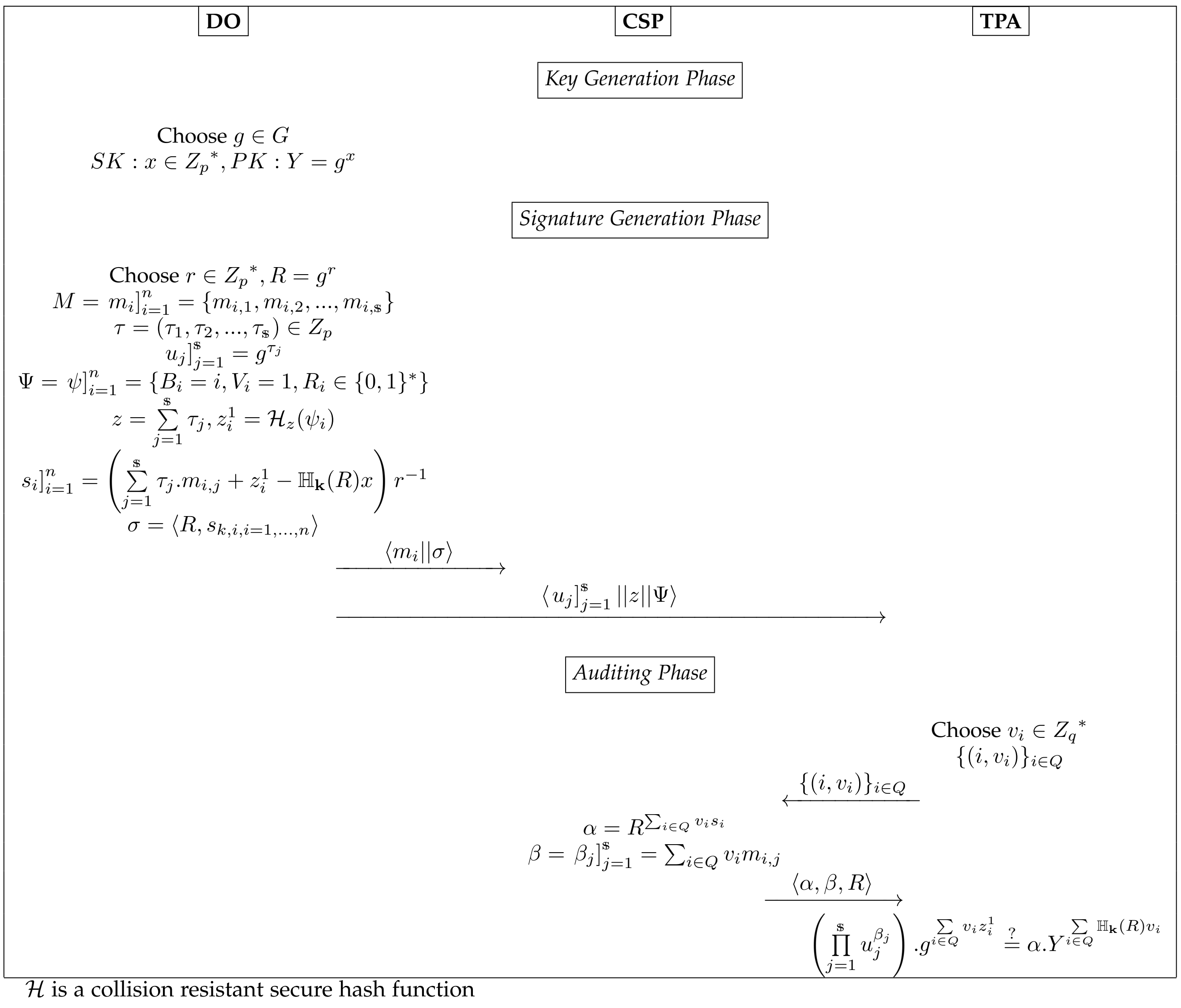

DO
CSP
TPA
Key Generation Phase
Choose $g \in G$
$SK : x \in Z_p{}^*, PK : Y = g^x$
Signature Generation Phase
Choose $r \in Z_p{}^*, R = g^r$
$M = m_i]_{i=1}^{n} = \{m_{i,1}, m_{i,2}, ..., m_{i,\$}\}$
$\tau = (\tau_1, \tau_2, ..., \tau_\$) \in Z_p$
$u_j]_{j=1}^{\$} = g^{\tau_j}$
$\Psi = \psi]_{i=1}^{n} = \{B_i = i, V_i = 1, R_i \in \{0,1\}^*\}$
$z = \sum_{j=1}^{\$} \tau_j, z_i^1 = \mathcal{H}_z(\psi_i)$
$s_i]_{i=1}^{n} = \left(\sum_{j=1}^{\$} \tau_j.m_{i,j} + z_i^1 - \mathbb{H}_{\mathbf{k}}(R)x\right) r^{-1}$
$\sigma = \langle R, s_{k,i,i=1,...,n}\rangle$
$\langle m_i || \sigma \rangle$
$\langle u_j]_{j=1}^{\$} || z || \Psi \rangle$
Auditing Phase
Choose $v_i \in Z_q{}^*$
$\{(i, v_i)\}_{i \in Q}$
$\{(i, v_i)\}_{i \in Q}$
$\alpha = R^{\sum_{i \in Q} v_i s_i}$
$\beta = \beta_j]_{j=1}^{\$} = \sum_{i \in Q} v_i m_{i,j}$
$\langle \alpha, \beta, R \rangle$
$\left(\prod_{j=1}^{\$} u_j^{\beta_j}\right) .g^{\sum_{i \in Q} v_i z_i^1} \overset{?}{=} \alpha.Y^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R) v_i}$
$\mathcal{H}$ is a collision resistant secure hash function

Fig. 7. Schematic diagram of dynamic data operation extension of proposed SEPDP.

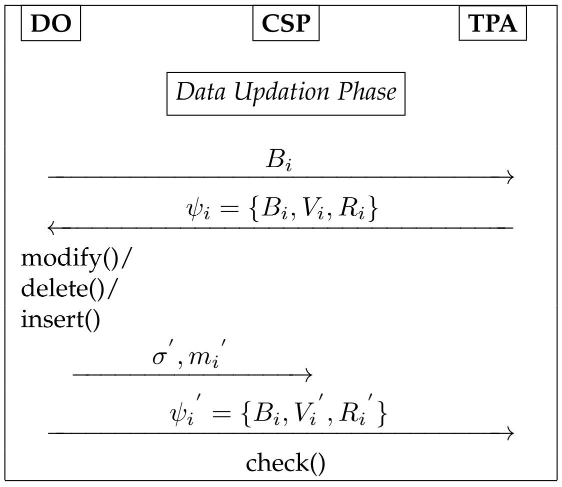


Fig. 8. Schematic diagram of data updation phase of extended SEPDP to support dynamic data operations

$$
\begin{aligned}
& R^{\sum_{i \in Q} v_i s_i} .Y^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R) v_i} \\
&= g^{r \sum_{i \in Q} v_i s_i} .g^{x \sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R) v_i} \\
&= g^{r \sum_{i \in Q} v_i \left(\left(\sum_{j=1}^{\$} \tau_j.m_{i,j} + z_i^1 - \mathbb{H}_{\mathbf{k}}(R)x\right) r^{-1}\right)} .g^{x \sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R) v_i} \\
&= g^{\sum_{i \in Q} v_i \left(\sum_{j=1}^{\$} \tau_j.m_{i,j} + z_i^1\right)} .g^{-x \sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R) v_i} .g^{x \sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R) v_i} \\
&= g^{\sum_{i \in Q} v_i \left(\sum_{j=1}^{\$} \tau_j.m_{i,j}\right)} .g^{\sum_{i \in Q} v_i z_i^1} \\
&= g^{\sum_{j=1}^{\$} \tau_j \left(\sum_{i \in Q} v_i.m_{i,j}\right)} .g^{\sum_{i \in Q} v_i z_i^1} \\
&= \prod_{j=1}^{\$} u_j^{\beta_j} .g^{\sum_{i \in Q} v_i z_i^1} \\
&= \text{L.H.S.}
\end{aligned}
$$

8 Security Analysis

In this section, SEPDP is analyzed against unforgeability and privacy to prove its security. Also, a probabilistic analysis for misbehavior detection of CSP is provided.

8.1 Unforgeability

CSP can attempt to break SEPDP in two alternative ways: (1) It generates a forge signature corresponding to a block of the file and subsequently forms the correct auditing response. (2) It generates a forge audit response message corresponding to (i, v_i) without having proper data, which passes the verification test at TPA. However, following two theorems prove that it is computationally infeasible for the CSP to succeed in either of these two ways.

Theorem 1. *Given a set of data and the corresponding signatures, it is computationally infeasible for CSP to generate a forgery of a signature.*

Proof: Based on the operations of the phases of SEPDP, algorithm $\mathscr{B}$ simulates a *security game*, which consists of two phases. In phase-1 of the security game, CSP can request three different kinds of queries to $\mathscr{B}$, which includes *Setup Query*, *Query for* $\mathbb{R}$, and *Sign Query*. In phase-2 of the security game CSP generates forgery of a signature. Algorithm $\mathscr{B}$ simulates the security game as follows and records the result of the queries in a tables corresponding to the appropriate query.

Setup Query: Consequent upon receiving the *Hash Query* from CSP, $\mathscr{B}$ sets $PK : \mathbb{Y} = g^{\mathbb{x}}$ and returns Y to the CSP.

Query for $\mathbb{R}$: CSP requests for $\mathbb{R}$ to $\mathscr{B}$. $\mathscr{B}$ chooses $\mathbb{r} \xleftarrow{U} {Z_p}^*$ and sets $\mathbb{R} = g^{\mathbb{r}}$. Then, it returns $\mathbb{R}$ to the CSP.

Sign Query: CSP requests for the result of the *Sign Query* for the message $\mathbb{m}_i$ and with it's identifier i to $\mathscr{B}$. To respond to the queries of this kind, algorithm $\mathscr{B}$ maintains a table ($T_{\mathfrak{s}}^{list}$), whose tuples are of the form $\langle \mathbb{m}_i, i, \rho_i, \mathfrak{s}_i \rangle$ as explained below. Initially, these entries of the table are empty. After receiving a *Sign Query* on input $(\mathbb{m}_i, i)$, algorithm $\mathscr{B}$ responds as follows.

1) If there is an entry corresponding to $(\mathbb{m}_i, i)$ in $T_{\mathfrak{s}}^{list}$, then $\mathscr{B}$ returns the corresponding $\mathfrak{s}_i$ to the CSP.
2) If there is no entry corresponding to $(\mathbb{m}_i, i)$ in table $T_{\mathfrak{s}}^{list}$, then the algorithm $\mathscr{B}$ chooses $\rho_i \xleftarrow{U} {Z_p}^*$ and sets $\mathfrak{s}_i = (\mathbb{m}_i - \rho_i.\mathbb{x}).\mathbb{r}^{-1}$ and sends $\mathfrak{s}_i$ to CSP. $\mathscr{B}$ inserts the tuple $\langle \mathbb{m}_i, i, \rho_i, \mathfrak{s}_i \rangle$ to the list $T_{\mathfrak{s}}^{list}$.

Forgery: After phase-1 of the security game is over, it outputs a forgery $\mathfrak{s}_i{}^*$ on $(\mathbb{m}_i{}^*, i^*)$. Below, we show that if CSP can successfully generate a forgery of a signature, then $\mathscr{B}$ can easily find $\mathbb{H}_{\mathbf{k}}(R||i^*)$ without knowing $\mathbf{k}$.

Let CSP has queried $\mathfrak{q}_{\mathfrak{s}}$ numbers of *Sign Query*. So, it has a set of $\mathfrak{q}_{\mathfrak{s}}$ numbers linear equations of the following form.

$$\begin{aligned}
\mathfrak{s}_1 &= (\mathbb{m}_1 - \rho_1.\mathbb{x}).\mathbb{r}^{-1} \pmod p \\
\mathfrak{s}_2 &= (\mathbb{m}_2 - \rho_2.\mathbb{x}).\mathbb{r}^{-1} \pmod p \\
\mathfrak{s}_3 &= (\mathbb{m}_3 - \rho_3.\mathbb{x}).\mathbb{r}^{-1} \pmod p \\
&\cdot \\
&\cdot \\
&\cdot \\
\mathfrak{s}_{\mathfrak{q}_{\mathfrak{s}}} &= (\mathbb{m}_{\mathfrak{q}_{\mathfrak{s}}} - \rho_{\mathfrak{q}_{\mathfrak{s}}}.\mathbb{x}).\mathbb{r}^{-1} \pmod p
\end{aligned}$$

Suppose CSP could solve these $\mathfrak{q}_{\mathfrak{s}}$ number of linear equations with $(\mathfrak{q}_{\mathfrak{s}} + 2)$ unknowns $(\rho_1, \rho_2, ..., \rho_{\mathfrak{q}_{\mathfrak{s}}}, \mathbb{x}, \mathbb{r})$. Consequently, it can generate forgery $\mathfrak{s}_i{}^*(= (\mathbb{m}_i{}^* - \rho_i{}^*.\mathbb{x}).\mathbb{r}^{-1})$ on $(\mathbb{m}_i{}^*, i^*)$. Then, $\mathscr{B}$ learns that $\rho_i{}^* = \frac{\mathfrak{s}_i{}^* - \mathbb{m}_i{}^*.\mathbb{r}^{-1}}{\mathbb{x}.\mathbb{r}^{-1}}$ $(= \mathbb{H}_{\mathbf{k}}(R||i^*))$.

This shows that, if CSP can successfully generate a forgery of a signature, then $\mathscr{B}$ can easily find $\mathbb{H}_{\mathbf{k}}(R||i^*)$ without knowing $\mathbf{k}$. But this contradicts the assumption of keyed-hash function. Hence, our assumption is wrong. So, in the proposed SEPDP, it is computationally infeasible to generate a forgery of a signature by the CSP. □

Theorem 2. *It is computationally infeasible for CSP to cheat TPA by generating forgery of an auditing response message, without having corresponding data.*

Proof: CSP can successfully generate a forgery of the auditing response message if it wins the following *security game*.

TPA sends a verification request $\{(i, v_i)\}_{i \in Q}$ to CSP. The response on original M would be $\{\alpha, \gamma, R\}$ where $\gamma = g^{\beta}$ and $\beta = \sum_{i \in Q} v_i m_i$. Instead of generating the correct response, CSP generates a forgery for the response over the corrupt data $M^{'}$ as $\{\alpha, \gamma^{'}, R\}$ where $\gamma^{'} = g^{\beta^{'}}$ and $\beta^{'} = \sum_{i \in Q} v_i m_i^{'}$ and $m_i^{'} \in M^{'}$ for $i \in Q$. Define $\Delta\beta = \beta^{'} - \beta$. Here, $\Delta\beta$ is non-zero as v_i's are random numbers and $M^{'} \neq M$. CSP wins this security game if this forgery on $M^{'}$ clears the verification Equation (6) at the TPA. Otherwise, it loses the game.

Let us assume that CSP wins the above security game. Hence, the corrupted response $\{\alpha, \gamma^{'}, R\}$ passes the verification Equation (6). So,

$$\gamma^{'} = R^{\sum_{i \in Q} v_i s_i} Y^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R||i) v_i} \tag{13}$$

But, according to the proposed scheme SEPDP the correct response $\{\alpha, \gamma, R\}$ also passes the verification Equation (6). Hence,

$$\gamma = R^{\sum_{i \in Q} v_i s_i} Y^{\sum_{i \in Q} \mathbb{H}_{\mathbf{k}}(R||i) v_i} \tag{14}$$

Now, from the Equations (13) and (14) it is clear that,

$$\gamma = \gamma^{'} \Rightarrow g^{\beta} = g^{\beta^{'}} \Rightarrow g^{\Delta\beta} = 1$$

In Z_p, for two elements $\mathfrak{m}, \mathfrak{n} \in Z_p, \exists \mathfrak{x} \in Z_p | \mathfrak{n} = \mathfrak{m}^{\mathfrak{x}}$. Hence, given $\mathfrak{m}, \mathfrak{n} \in Z_p$, $g = \mathfrak{m}^{\theta}\mathfrak{n}^{\eta}$ where θ and η are random numbers in Z_p. So,

$$\begin{aligned}
& g^{\Delta\beta} = 1 \\
& \Rightarrow \left(\mathfrak{m}^{\theta}\mathfrak{n}^{\eta}\right)^{\Delta\beta} = 1 \\
& \Rightarrow \left(\mathfrak{m}^{\theta.\Delta\beta}.\mathfrak{n}^{\eta.\Delta\beta}\right) = 1 \\
& \Rightarrow \mathfrak{n}^{\eta.\Delta\beta} = \mathfrak{m}^{-\theta.\Delta\beta}
\end{aligned}$$

Now taking logarithm and solving the above equation, we get

$$\mathfrak{n} = \mathfrak{m}^{-\frac{\theta\Delta\beta}{\eta\Delta\beta}}$$

Hence, the solution of the DL problem is $\mathfrak{x} = -\frac{\theta\Delta\beta}{\eta\Delta\beta}$ unless $\eta\Delta\beta$ is zero. However, according to the *security game*, $\Delta\beta$ can not be zero (as v_i's are random numbers). As η is a random element in Z_p, so $\eta\Delta\beta$ is zero with probability $\frac{1}{p}$ where p is a large prime. Hence, a solution to the DL problem can be found with a probability $1 - \frac{1}{p}$. This implies, if CSP wins the game, then we have the solution of the

DL problem with a probability of $1-\frac{1}{p}$, which is quite high. But this contradicts the DL assumption[1]. Hence, it is computationally infeasible for the CSP to cheat TPA without having corresponding data by generating forgery of a response message in the proposed scheme. □

Similarly, one can prove that the multiple DO extension, batch auditing extension and data dynamics extension of the SEPDP also satisfy the unforgeable property.

8.2 Privacy Preserving

We have to ensure that TPA can not find DO's data from the information exchanged during the auditing phase to preserve privacy of the data. If TPA can find the value of $\beta = \sum_{i\in Q} v_i m_i$, then it can obtain $\{m_i\}_{i\in Q}$ by solving a system of linear equations [4]. Using three cases, we show that no information regarding β will be leaked to TPA from the response message $\{\alpha, \gamma, R\}$ in SEPDP.

Case 1: According to Equation (5) and (4) $\gamma = g^{\beta}$ and $\beta = \sum_{i\in Q} v_i m_i$ respectively. So, TPA will try to solve the Equation (5) to find β, which further will be used in Equation (4) to find m_i. Equation of the form (5) is always equivalent to computing of discrete logarithm problem over $GF(p)$ as the unknown β appear in the exponent. As there is no efficient algorithm to solve discrete logarithm problem, it is computationally infeasible for TPA to derive the value of β from Equation (5). Thus, privacy of β is guaranteed from γ.

Case 2: Similarly, no information regarding β can be leaked from α and R which is described below. We know,

$$\begin{aligned}\alpha &= R^{\sum_{i\in Q} v_i s_i}\\ &= R^{\sum_{i\in Q} v_i((m_i - \mathbb{H}_{\mathbf{k}}(R||i)x)r^{-1})}\\ &= R^{\sum_{i\in Q} v_i(m_i r^{-1} - \mathbb{H}_{\mathbf{k}}(R||i)xr^{-1})}\\ &= R^{(\sum_{i\in Q} v_i m_i)r^{-1}} R^{-\sum_{i\in Q}\mathbb{H}_{\mathbf{k}}(R||i)xr^{-1}}\\ &= R^{\beta r^{-1}} R^{-\sum_{i\in Q}\mathbb{H}_{\mathbf{k}}(R||i)xr^{-1}}\\ &= R^{\beta r^{-1}} (g^{-\sum_{i\in Q}\mathbb{H}_{\mathbf{k}}(R||i)})^{x}\end{aligned}$$

Thus, $(g^{-\sum_{i\in Q}\mathbb{H}_{\mathbf{k}}(R||i)})^{x}$ blinds $R^{\beta r^{-1}}$. As TPA is only having $(g^{-\sum_{i\in Q}\mathbb{H}_{\mathbf{k}}(R||i)})^{x}$ and g is a public parameter, computing x from only these two pieces of information is infeasible due to discrete logarithm assumption.

Case 3: Guessing of β from the response is hard due to the following reason. We know that $\gamma = g^{\sum_{i\in Q} v_i m_i}$ and $v_i \overset{R}{\leftarrow} Z_q{}^*$ is a random number. Assume v_i's are of n_v bits, then there are $(2)^{n_v}$ possible values of v_i. if $|Q| = c$, then for g^{β} there are $c.\,(2)^{n_v}$ possibilities. Similarly, $\alpha = R^{\sum_{i\in Q} v_i s_i}$ and $v_i \overset{R}{\leftarrow} Z_q{}^*$ is a random number. So there are $c.\,(2)^{n_v}$ possible values of α. $R = g^r$ and $r \overset{R}{\leftarrow} Z_p{}^*$ is a random number. Assume r is of n_r bits, then there are $(2)^{n_r}$ possible values of R.

1. Let $\mathbb{x} \in Z_p^*$, given P and $P^{\mathbb{x}} \in G$, it is computationally infeasible for a polynomial time adversary $\mathscr{A}_{dl}$, to determine x. Mathematically,

$$\Pr[\mathscr{A}_{dl}(P, P^{\mathbb{x}}) = \mathbb{x} : \mathbb{x} \in Z_p^*] \le \epsilon$$

where ϵ is negligible.

As n_v and n_r are independent of each other, the probability that a guessed value of response message will be same as the original response message is

$$\mathscr{P} = \frac{1}{2c.\,(2)^{n_v} + (2)^{n_r}}$$

Therefore the larger the value of n_v and n_r, the stronger security we can achieve. If we choose r of 1024-bit, then probability $\mathscr{P} < \frac{1}{(2)^{1024}}$, which is very negligible. Hence, we can conclude that the privacy of β is protected against the TPA.

Similarly, one can prove that the multiple DO extension, batch auditing extension and data dynamics extension of the SEPDP also satisfy the privacy preserving property.

8.3 Probability of Misbehavior Detection

In the auditing phase of SEPDP, random sampling method is adopted for detecting the misbehavior of CSP, which reduces the workload on TPA. The random sampling technique divides the file M into n number of blocks. Let TPA randomly selects c $(c < n)$ number of blocks for challenge. Assume that CSP modifies x blocks out of n outsourced blocks by DO and percentage of modified blocks is denoted as p_1. Suppose P be the probability that at least one block picked by TPA out of n blocks which matches with one of the blocks modified by CSP. Hence,

$$P = 1 - \frac{{}^{n-x}C_c}{{}^{n}C_c} = 1 - \frac{(n-x)!(n-c)!}{(n-x-x)!n!} \qquad (15)$$

In Figure 9, results of probability of detection (P) with different number of queried blocks (c) with different percentage of corrupted blocks (p_1) of the file is shown.

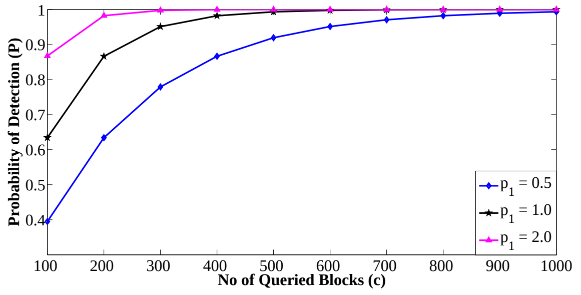


Fig. 9. Misbehavior detection probability under no. of queried blocks

Suppose DO divides a 500MB file (M) into 8KB blocks and uploads. Then, $n = 62,500$. Now, if CSP modifies 313 blocks out of 62, 500 blocks of M (i.e. $p_1 = 0.5\%$), then TPA has to randomly select 500 blocks to challenge the cloud server to achieve the detection probability (P) of at least 0.90 (using Equation (15)). Similarly, for $p_1 = 2.0\%$, TPA has to randomly select only 200 blocks out of the 62,500 blocks to achieve the detection probability (P) of at least 0.99. Hence, SEPDP can detect the corrupted blocks with higher probability by randomly selecting a few blocks.

TABLE 2
Notations used for performance evaluation

Notations	Meaning
T_p	One pairing operation
T_e	One exponentiation operation
T_m	One multiplication operation
T_h	One hashing operation
T_i	One inverse operation
n	Number of blocks in F
s	Number of sectors in a block
c	Number of selected blocks ($\|Q\|$)
$\|p\|$	Size of element in G or Z_p
$\|q\|$	Size of element in Z_q
$\|n\|$	Size of an element of set $[1, n]$

9 Performance Analysis

Performance of SEPDP is discussed in terms of communication and computation overhead. We have used the notations for basic operations as depicted in Table 2.

Table 3 compares the communication cost among several existing schemes. Here, we concentrate only on the communications occur in the auditing phase (i.e. challenge and response messages). Communication cost of key generation and signature generation phase is ignored as DOs are not involved in every challenge response communication that occurs between CSP and TPA. It is inferred that communication overhead of SEPDP is identical to that of [7] and [11] while it incurs less than that of [16].

TABLE 3
Communication overhead comparison

Schemes	**Challenge Message** (in bits)	**Response Message** (in bits)
Shacham et al. scheme [3]	$c(\|n\| + \|q\|)$	$2\|p\|$
Zhu et al. scheme [16]	$c(\|n\| + \|q\|)$	$4\|p\|$
Wang et al. scheme [7]	$c(\|n\| + \|q\|)$	$3\|p\|$
Yang et al. scheme [11]	$c(\|n\| + \|q\|) + \|p\|$	$2\|p\|$
SEPDP	$c(\|n\| + \|q\|)$	$3\|p\|$

A comparison of the computational overhead at different phases of the proposed scheme with that of the existing schemes is provided in Table 4. Result shows that, Shacham et al.'s scheme [3], Zhu et al.'s scheme [16], Wang et al.'s scheme [7], and Yang et al.'s scheme [11] require 2, 4, 2 and 3 number of bilinear pairing operation respectively. However, pairing operations (T_p) takes more computational time as compared to the other operations like T_e, T_m, T_h and T_i [23]. But, as SEPDP does not need pairing based operations, the verification process requires low computational overhead and therefore suitable to implement in low power devices.

Time consumed by TPA is computed for the popular existing schemes ([3] and [7]) and compared with SEPDP. These schemes are implemented using PBC library (version - 0.5.14) [23] with *type A* pairing parameters from the PBC archive. We choose amazon elastic compute cloud (EC2) cloud platform [24] as a cloud service provider. Figure 10(a) and 10(b) show the average computation time required by CSP and TPArespectively, varying the number of queried blocks (c) keeping the total number of blocks fixed ($n = 3000$). Figure 10(c) and 10(d) shows the average computation time consumed by CSP and TPA respectively, varying the number of blocks (n) keeping the number of queried blocks (c) fixed ($c = 50$). It is observed from the Figure 10(b) and 10(d) that the computation time of the TPA in the SEPDP is less than that of both Shacham et al.'s scheme [3] and Wang et al.'s scheme [7]. This is due to the following reason. Verification Equation (6) used in our scheme does not contain any time consuming pairing operation (T_p) and the number of exponentiation operations (T_e) in Equation (6) is independent of c (i.e. number of queried blocks). On the other hand, in both the schemes ([3] and [7]), number of exponentiation operations varies with c (as mentioned in Table 4). Hence, in Figures 10(a) and 10(b), the computation overhead time of SEPDP is nearly constant as c increases. This is almost constant in Figure 10(c) and 10(d) as the computation time of CSP and TPA do not depend on n.

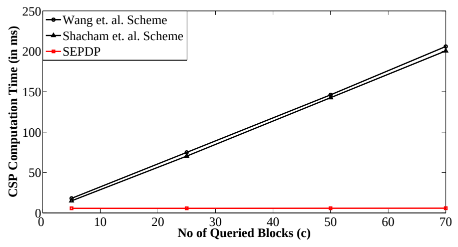
Computation time of the batch auditing extension of SEPDP is less than that of the individual auditing of the multiple tasks. This is simulated and the results are shown in Figure 10(e) and 10(f). In Figure 10(e), we varied number of auditing tasks from 5 to 95 with a gap of 10 keeping the number of blocks fixed ($n = 1000$). Here, the total computational time for the individual auditing for $c = 25$ and $c = 125$ is shown as a baseline. Figure 10(f) illustrates the comparison of total computation time for individual auditing of multiple tasks in SEPDP and the batch auditing extension of SEPDP. In this case, we varied number of auditing tasks from 5 to 95 keeping the number of queried blocks fixed ($c = 25$). It is observed (from both Figure 10(e) and 10(f)) that as the number of tasks increases, batch auditing scheme takes lesser time and saving in computation time increases with the number of auditing requests. This is due to the following reason. In case of batch auditing extension of SEPDP, CSP computes one group element α $\left(= R^{\sum_{j=1}^{d} \sum_{i \in Q} v_i s_{j,i}}\right)$ instead of d numbers of group elements α_j $\left(= R^{\sum_{i \in Q} v_i s_{j,i}},\ j]_{i=1}^{d}\right)$. Along with this, TPA verifies only 1 equation instead of d verification equations. So, the computation time is saved as the number of tasks increases.

We implemented the dynamic data operation extension of the proposed scheme and results are shown in Figure 10(g). For this, we varied the number of sectors from 5 to 95 by keeping the number of queried blocks as $c = 25$ and $c = 125$ and number of blocks as $n = 1000$ and $n = 3000$. Our scheme saves communication overhead time during the data updation phase as compared to Zhu et al. Scheme [16], because SEPDP sends only those entries of the IHT which correspond to the updated data (${m_i}'$) instead of the whole IHT.

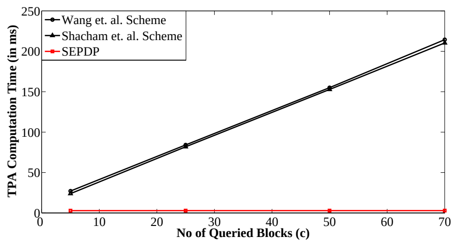
Storage correctness, blockless verification, unforgeability, public auditability, privacy preserving, data dynamics, batch auditing are the important features which would be considered for designing provable data possession in cloud storage. Table 5 summarizes the comparison of said features for different existing schemes and SEPDP. Shacham et al.'s Scheme [3] does not support privacy preserving, data dynamics, and batch auditing features while Wang et al.'s Scheme [4] fails to support data dynamics. Zhu et al.'s Scheme [16] does not support batch auditing feature. Verification time of SEPDP is faster as compared to the

TABLE 4
Computation overhead comparison

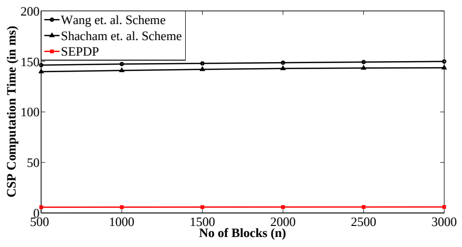
Schemes	Phase			Overall Computation
	KeyGen	Signing	Auditing	
Shacham et al. scheme [3]	T_e	$2nT_e + nT_m + nT_h$	$2T_p + (2c+1)T_e$ $+(3c-1)T_m + cT_h$	$2T_p + (2n+2c+2)T_e$ $+(n+3c-1)T_m + (n+c)T_h$
Zhu et al. scheme [16]	$2T_e$	$(2n+1)T_e+$ $2nT_m + (n+1)T_h$	$4T_p + (2c+3)T_e + (4c$ $-1)T_m + cT_h$	$4T_p + (2n+2c+6)T_e + (2n$ $+4c-1)T_m + (n+c+1)T_h$
Wang et al. scheme [7]	$2T_e$	$2nT_e + nT_m + nT_h$	$2T_p + (2c+3)T_e$ $+(3c+1)T_m + (c+2)T_h$	$2T_p + (2n+2c+5)T_e$ $+(n+3c+1)T_m + (n+c+2)T_h$
Yang et al. scheme [11]	T_e	$(n+2)T_e+$ nT_h	$3T_p + (2c+3)T_e$ $+2cT_m + cT_h$	$3T_p + (n+2c+6)T_e$ $+2cT_m + (n+c)T_h$
SEPDP	T_e	$T_e + 2nT_m$ $+nT_h + T_i$	$3T_e + (3c+1)T_m + cT_h$	$5T_e + (2n+3c+1)T_m + (n+c)T_h + T_i$



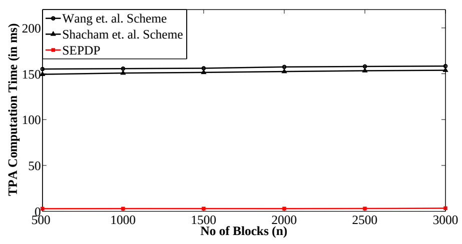
(a) Computation time of CSP under different no of queried blocks.



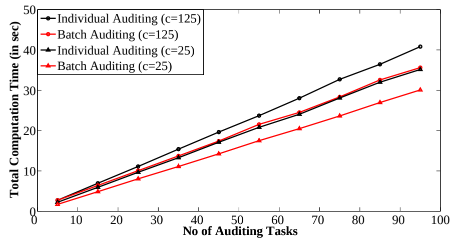
(b) Computation time of TPA under different no of queried blocks.



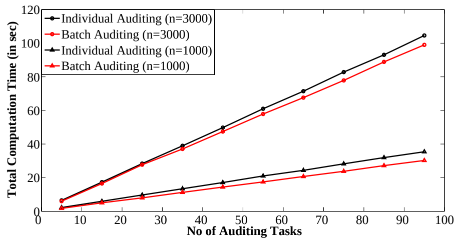
(c) Computation time of CSP under different no of blocks.



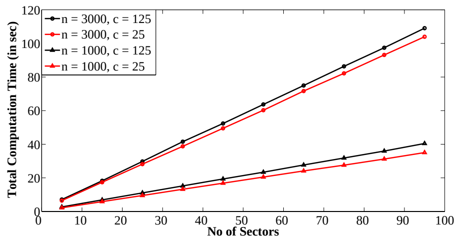
(d) Computation time of TPA under different no of blocks.



(e) Comparison of total computation time between individual auditing and batch auditing under different no of queried blocks.



(f) Comparison of total computation time between individual auditing and batch auditing under different no of blocks.



(g) Comparison of total computation time for different n and c values under different no of sectors.

Fig. 10. Experimental results of the proposed SEPDP

schemes ([3], [4], [7], [11], [16]).

10 Conclusion

In this paper, privacy preserving provable data possession scheme (named SEPDP) for untrusted and outsourced storage system is presented. Further, SEPDP is extended to support dynamic data updation by multiple owners and batch auditing. Security of the scheme is analyzed and showed that SEPDP protects data privacy from TPA while infeasible for CSP to forge the response without storing the appropriate blocks. The most appealing features of the proposed scheme is to support all the important features including blockless verification, privacy preserving, batch auditing and data dynamics with lesser computation overhead.

References

[1] K. Yang and X. Jia, "Data storage auditing service in cloud computing: challenges, methods and opportunities," *World Wide Web*, vol. 15, no. 4, pp. 409–428, 2012.

[2] B. Wang, B. Li, H. Li, and F. Li, "Certificateless public auditing for data integrity in the cloud," in *Proceedings IEEE Conference on Communications and Network Security (CNS)*, 2013, pp. 136–144.

TABLE 5
Requirement comparison of various public auditing scheme

Schemes	Storage Correctness	Blockless Verification	Unforgeability	Public Auditability	Privacy Preserving	Data Dynamics	Batch Auditing	Verification Speed
Shacham et al. Scheme [3]	Yes	Yes	Yes	Yes	No	No	No	Slow $(2T_p + (2c+1)T_e)$
Wang et al. Scheme [4]	Yes	Yes	Yes	Yes	Yes	No	Yes	Slow $(2T_p + (2n+2c+5)T_e)$
Zhu et al. Scheme [16]	Yes	Yes	Yes	Yes	Yes	Yes	No	Slow $(4T_p + (2n+2c+6)T_e)$
Wang et al. Scheme [7]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Slow $(2T_p + (2n+2c+5)T_e)$
Yang et al. Scheme [11]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Slow $(3T_p + (n+2c+6)T_e)$
SEPDP	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Fast $(5T_e)$

[3] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proceedings of 14th ASIACRYPT*, 2008, pp. 90–107.

[4] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proceedings of 29th IEEE Conference on Computer Communications (INFOCOM)*, 2010, pp. 1–9.

[5] L. Yuchuan, F. Shaojing, X. Ming, and W. Dongsheng, "Enable data dynamics for algebraic signatures based remote data possession checking in the cloud storage," *China Communications*, vol. 11, no. 11, pp. 114–124, 2014.

[6] A. F. Barsoum and M. A. Hasan, "Provable multicopy dynamic data possession in cloud computing systems," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 485–497, 2015.

[7] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.

[8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proceedings of the 14th ACM conference on Computer and communications security*, 2007, pp. 598–609.

[9] B. Wang, H. Li, X. Liu, F. Li, and 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.

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

[11] 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.

[12] H. Wang, "Proxy provable data possession in public clouds," *IEEE Transactions on Services Computing*, vol. 6, no. 4, pp. 551–559, 2013.

[13] ——, "Identity-based distributed provable data possession in multicloud storage," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 328–340, 2015.

[14] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, "Toward secure and dependable storage services in cloud computing," *IEEE transactions on Services Computing*, vol. 5, no. 2, pp. 220–232, 2012.

[15] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for outsourced storages in clouds," *IEEE Transactions on Services Computing*, vol. 6, no. 2, pp. 227–238, 2013.

[16] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in *Proceedings of the 2011 ACM Symposium on Applied Computing*. ACM, 2011, pp. 1550–1557.

[17] F. Sebé, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 8, pp. 1034–1038, 2008.

[18] D. L. Gazzoni Filho and P. S. L. M. Barreto, "Demonstrating data possession and uncheatable data transfer." *IACR Cryptology ePrint Archive*, vol. 2006/150, 2006.

[19] Z. Hao, S. Zhong, and N. Yu, "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability," *IEEE transactions on Knowledge and Data Engineering*, vol. 23, no. 9, pp. 1432–1437, 2011.

[20] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *Proceedings of 7th ASIACRYPT*, 2001, pp. 514–532.

[21] P. Adusumilli, X. Zou, and B. Ramamurthy, "Dgkd: Distributed group key distribution with authentication capability," in *Proceedings of the Sixth Annual IEEE SMC Information Assurance Workshop*. IEEE, 2005, pp. 286–293.

[22] M. Nabeel, M. Yoosuf, and E. Bertino, "Attribute based group key management," in *Proceedings of the 14th ACM symposium on Access control models and technologies*, 2014, pp. 115–124.

[23] B. Lynn, "The pairing-based cryptography library," *Internet: crypto. stanford. edu/pbc/[Mar. 27, 2013]*, 2006.

[24] Amazon Elastic Compute Cloud (Amazon EC2) Available: https://aws.amazon.com/ec2/.

Sanjeet Kumar Nayak Sanjeet Kumar Nayak is presently perusing his Ph.D degree in Computer Science & Engineering from Indian Institute of Technology, Patna. He received his B.Tech degree from Biju Patnaik University of Technology, Odisha and M.Tech degree from National Institute of Technology, Rourkela. His research interest includes security & privacy issues in cloud data storage, blind signature schemes. He has published one conference of international repute.

Somanath Tripathy Dr. Somanath Tripathy received his PhD in Computer Science and Engineering from IIT Guwahati in 2007. Currently, he is an Associate Professor in the Computer Science and Engineering Department of IIT Patna. He joined the faculty in December 2008. Prior to joining IIT Patna, he is the founder head of the Department of Information Technology at North-Eastern Hill University, Shillong. His research interest includes lightweight cryptography, Security issues in resource restrained devices, Cloud computing security and IOT Security. He has published more than 40 research papers in different journals and conferences of repute.