Multimed Tools Appl (2016) 75:13077–13091
DOI 10.1007/s11042-015-2594-5

Privacy-preserving public auditing for educational multimedia data in cloud computing

Daeyeong Kim[1] · Hyunsoo Kwon[1] · Changhee Hahn[1] · Junbeom Hur[2]

Received: 20 December 2014 / Revised: 13 February 2015 / Accepted: 30 March 2015 /
Published online: 17 April 2015

Abstract Nowadays, as distance learning is being widly used, multimedia data becomes an effective way for delivering educational contents in online educational systems. To handle the educational multimedia data efficiently, many distance learning systems adopt a cloud storage service. Cloud computing and storage services provide a secure and reliable access to the outsourced educational multimedia contents for users. However, it brings challenging security issues in terms of data confidentiality and integrity. The straightforward way for the integrity check is to make the user download the entire data for verifying them. But, it is inefficient due to the large size of educational multimedia data in the cloud. Recently many integrity auditing protocols have been proposed, but most of them do not consider the data privacy for the cloud service provider. Additionally, the previous schemes suffer from dynamic management of outsourced data. In this paper, we propose a public auditing protocol for educational multimedia data outsourced in the cloud storage. By using random values and a homomorphic hash function, our proposed protocol ensures data privacy for the cloud and the third party auditor (TPA). Also, it is secure against lose attack and temper attack. Moreover, our protocol is able to support fully dynamic auditing. Security and

✉ Junbeom Hur
jbhur@korea.ac.kr

Daeyeong Kim
rlaeod@cau.ac.kr

Hyunsoo Kwon
khs910504@cau.ac.kr

Changhee Hahn
Mckinsey@cau.ac.kr

[1] Department of Computer Science and Engineering, Chung-Ang University, Seoul, Republic of Korea

[2] Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea

performance analysis results show that the proposed scheme is secure while guaranteeing minimum extra computation costs.

Keywords Privacy preserving auditing · Fully dynamic auditing · Cloud computing · Homomorphic hash · Educational multimedia

1 Introduction

Nowadays, distance learning has become one of the rapidly growing trends for both the students and educational institutions. It is an effective method of delivering educational contents to remote users. As traditional distance learning provides a text or picture at the designated time, students were only able to study during a fixed amount of time. Due to rapid growth of the multimedia Web [17], distance learning has enabled students to study without time and space constraints by using educational multimedia data. In addition, since students can learn more effectively when the learning materials are presented in multimedia data such as video, audio and text, utilization of multimedia data has become increasingly common in learning systems. However, as the volume of educational multimedia data is enriched, huge amounts of educational multimedia have been difficult to be handled in multimedia Web. e,g., storing/editing/deleting the data. To handle the educational multimedia data efficiently, distance learning systems are likely to use the cloud computing [10, 12]. Since, the cloud computing provides dynamically scalable infrastructure to support computing power, storage and communication capabilities as services, it is suitable for distance learning environments. Data users, that is educational contents owners, can easily access and store educational multimedia data in the cloud storage. However, it also brings new and challenging security issues in terms of confidentiality and integrity of the user's outsourced data. Since the cloud has full control of the stored educational multimedia data, a malicious cloud may be able to perform attacks on the data so as to maintain reputation and financial benefit. Therefore, the data user needs to check that educational multimedia data is correctly stored in the cloud.

Since size of educational multimedia data in the cloud is very large, it is a very inefficient in terms of the communication and computation overhead for a verifier, that is the data user or owner, to retrieve entire data from the cloud and check the integrity of it. Therefore, to save the computation and communication cost, a public auditing protocol, which enables a third party auditor (TPA) to verify it on behalf of users is critically important. Recently, many integrity public auditing schemes [1, 2, 5, 7, 8, 11, 13–16, 18–20] have been proposed to check the integrity of the data without retrieving the whole data. These schemes perform the partial integrity check by random sampling using challenge queries. Another issue is supporting dynamic data operation for outsourced data in the cloud. Supporting dynamic data is a one of the most challenging and practical requirements since the cloud data could be dynamically updated during their lifetime, such as data insertion/deletion/modification. Unfortunately, many schemes [1, 2, 8, 13] focus on static data and cannot support data update. In addition, most of these schemes [1, 2, 5, 8, 11, 13–15] do not support the privacy protection of data against the cloud. These schemes provide only the data privacy for the TPA. If the data and index are encrypted to ensure data privacy for the cloud, they cannot support fully dynamic data.

In this paper, we focus on the way to provide the data privacy and integrity for the educational multimedia data in the cloud. We propose a public auditing scheme that supports fully dynamic data as well as data privacy against both the untrusted cloud server and the TPA under the loss and tamper attacks [11] by using the random values and homomorphic hash function [6, 9]. Specifically, in the proposed scheme, we allow the data user to combine data block with random value which is not known to the cloud, thereby providing data privacy against the cloud. Similarly, we allow the cloud to combine homomorphic authenticator with random value which is not known to the TPA, thereby providing data privacy against the TPA. Since the TPA checks the integrity using homomorphic property, the TPA cannot learn any information about the cloud data during the auditing process. On the basis of the security and efficiency analysis results, the proposed scheme is secure against the attacks while guaranteeing minimum extra computation costs.

The rest of this paper is organized as follows. We overview the related work in Section 2, and we introduce the system and threat model, and our design goal in Section 3. Then we provide the preliminaries and the detailed description of our scheme in Section 4. In Section 5, we give security and performance analysis. Finally we conclude our work in Section 6.

2 Related work

Ateniese et al.'s [1] proposed Provable Data Possession (PDP) that allows public auditing without retrieving entire data. They utilize the RSA-based homomorphic linear authenticator (HLA). However, this scheme does not consider dynamic data. To support dynamic data, Ateniese et al.'s [2] designed an improved PDP scheme using symentric keys. However, their scheme is limited number of verification challenge queries. And it does not support fully dynamic data operation. i.e., block insertion cannot be supported. In addition, these two schemes did not consider data privacy. Thus, these schemes may leak users' data contents to the cloud and the TPA. Juels and Kaliski [8] defined another scheme called proof of retrievability (POR) that ensures both possession and retrievability of data file by using error correcting. However, the number of challenge queries is fixed. In addition, public auditing is not supported in their scheme. To support public auditing, Shacham and Waters [13] proposed an improve POR scheme based on BLS (Boneh-Lynn-Shacham) signature [4]. It provides proof of security for the POR scheme. However this scheme only consider static data files. Erway et al.'s [5] designed a dynamic provable data possession (DPDP) scheme based on rank-based authenticated skip list. Wang et al.'s [15] proposed a public auditing scheme that combines HLA with Merkle hash tree to support fully dynamic data. However, these schemes suffer from weak privacy against the cloud and the TPA. In other related works, Wang et al.'s [14] proposed a public auditing scheme for data privacy against the TPA using random mask technique. Liu et al.'s [11] proposed a secure and efficient public auditing scheme using homomorphic hash function [6, 9]. However, their protocols did not provide data privacy at the cloud. Thus, these schemes may leak users' data contents to the cloud. To provide the data privacy against the cloud, Zhu et al.'s [19] proposed a scheme that supports data integrity auditing in hybrid cloud. This scheme is suitable for providing integrity of the data users' important data. Govinda et al.'s [7] proposed a scheme that guarantees both the data integrity and confidentiality using somewhat homomorphic encryption. Thus, this protocol considers the data privacy against both the cloud and the TPA. However, this protocol did not support data dynamics including block insertion and deletion because the index and data block are encrypted.

3 Problem statement

3.1 System model

As illustrated in Fig. 1, the system model includes three entities: (1) Users who have educational multimedia data files to be stored in the cloud. (2) Cloud which provides data storage services and computing resources for user data. (3) TPA that checks the integrity of data stored in the cloud on behalf of users upon request. Our scheme consists of six algorithms. (**Setup**, **KeyGen**, **SigGen**, **Challenge**, **ProofGen**, **ProofVerify**)

A user is given a public key and a secret key by executing **KeyGen**, and generates a signature by using **SigGen**. The user divides original data into multiple blocks of data before storing it in the cloud. And the user sends the blocks and the corresponding signatures to the cloud. The user deletes data blocks and signatures from the local storage. When the user wants to verify the stored data in the cloud, the user sends auditing request to the TPA. After receiving the auditing request from the user, the TPA generates a challenge message by executing **Challenge**. Then, the TPA sends a challenge message to the cloud server. The cloud server will derive a proof from the stored data by executing **ProofGen**. The TPA verifies the correctness of the proof by **ProofVerify**.

3.2 Threat model

In this paper, we assume that the cloud is not fully trusted in terms of the data confidentiality and integrity, and the TPA is semi-trusted such that it is trusted for auditing process but untrusted for data confidentiality.

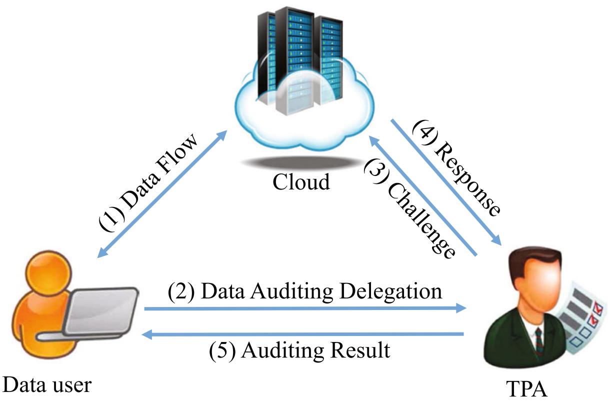

Fig. 1 The system model of the cloud data storage service

Data integrity threats Malicious cloud may discard the data that have not been accessed or rarely accessed, or tampered the data so as to maintain reputation. It is called loss attack and tamper attack respectively [11].

Data privacy threat During the auditing process, the cloud and the TPA may try to reveal the users information. We also assume that there is no collusion between the cloud and the TPA.

3.3 Design goal

The proposed scheme should achieve the following properties: (1) **Public auditing**: The TPA is able to check the integrity of data without retrieving the whole data. (2) **Privacy preserving**: The cloud and the TPA cannot obtain users plain data during the auditing process. (3) **Fully dynamic data**: The user is able to perform block-level operations on the outsourced data including block insertion, deletion and modification.

4 Public auditing scheme

4.1 Preliminaries

4.1.1 Bilinear maps

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative cyclic groups of order p. g is the generator of $\mathbb{G}_1$. A bilinear mapping $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ must satisfy the following properties:

– Bilinearity: $e(u^a, v^b) = e(u, v)^{ab}$ for all $u, v \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p$.
– Non-degeneracy: There are $u, v \in \mathbb{G}_1$ such that $e(u, v) \neq 1$.
– Computability: There is an efficient algorithm to compute $e(u, v)$ for any $u, v \in \mathbb{G}_1$.

4.1.2 Homomorphic hash function

Homomorphic hash function [6, 9] is hash function satisfying:

Homomorphism For any two message m_1, m_2 and scalars α_1, α_2, it holds

$$H(\alpha_1 m_1 + \alpha_2 m_2) = H(m_1)^{\alpha_1} \cdot H(m_2)^{\alpha_2}.$$

Collision resistance There is no probabilistic polynomial-time (PPT) adversary capable of forging $(m_1, m_2, m_3, \alpha_1, \alpha_2)$ satisfying both

$$m_3 \neq \alpha_1 m_1 + \alpha_2 m_2, \quad and \quad H(m_3) = H(m_1)^{\alpha_1} \cdot H(m_2)^{\alpha_2}.$$

4.2 Proposed scheme

In the section, we present our proposed scheme that ensures data privacy against both the cloud and the TPA. In our protocol, since the user and the cloud choose random values and combine them with date file F and $proof$ respectively, contents of data fila F and $proof$ cannot be exposed. To perform the operation of the homomorphic hash function, our scheme uses the signature of Liu et al.'s scheme [11]. Homomorphic hash function can support the operations that is required for the verification. Our scheme consists of six algorithms; **Setup**, **KeyGen**, **SigGen**, **Challenge**, **ProofGen**, **ProofVerify** and are defined as follows:

Setup To store user data file F in the cloud, F is divided into n blocks as $m_1, \ldots m_n, m_i \in \mathbb{Z}_p{}^*$

KeyGen(1^λ) →{Sk,Pk} A user chooses a large prime p randomly. Define $\mathbb{G}_1$ and $\mathbb{G}_2$ to be multiplicative cyclic groups with order p. Let g be a generator of $\mathbb{G}_1$. H: $\mathbb{Z}_p{}^* \rightarrow \mathbb{G}_1$ is a homomorphic hash function. The user chooses $\mathbb{Z}_p{}^* \rightarrow x$ randomly, and computes $g \in \mathbb{G}_1$. Public key is Pk=$\{g, v\}$ and secret key is Sk = $\{x\}$.

SigGen(F, Sk) → $\{F', \sigma\}$ Define the identity of the file F to be $id \in \mathbb{Z}_p{}^*$ for each $i \in \{1, \ldots, n\}$. Then, the user computes the signature as

$$\sigma_i \leftarrow \left(H(id||i)H(m_i))^x\right) \in \mathbb{G}_1$$

To protect data privacy against the cloud, the user chooses a n random value $r_n \leftarrow \mathbb{Z}_p{}^*$. Then, the user combines random r-values with each block as $F' = \{m_1 + r_1, \ldots, m_n + r_n\}$. Then, the user sends F' and the corresponding signature $\{\sigma_1, \ldots, \sigma_n\}$ to the cloud. Finally the user deletes the data file F from local storage.

Challenge(request, index table)→ $\{F', \sigma\}$ To verify the integrity of outsourced data, the user sends auditing request and the index table to the TPA which includes index, random r-value and r-index the user selected. Subsequently, the TPA defines the subset of set [1,n] to be $I = \{s_j\}(1 \le j \le c)$ and $s_1 \le \ldots \le s_c$. For each $i \in I$, the TPA chooses a random $\nu_i \leftarrow \mathbb{Z}_p{}^*$, and generates $chal$=$\{i, \nu_i\}_{i \in I}$ as a challenge message. Then, the TPA sends $chal$ to the cloud.

ProofGen($\{m_i + r_i\}_{i\in I}$, $\{\sigma_i\}_{i\in I}$, $chal$, Pk) → $\{proof\}$ Upon receiving the $chal$, the cloud computes as

$$\mu = \sum_{i=s_i}^{s_c} \nu_i(m_i + r_i) + r', \qquad \sigma = \prod_{i=s_1}^{s_c} \sigma_i^{\nu_i}.$$

To prevent the TPA from learning the data content stored in the cloud during the auditing process, the cloud chooses a random $r' \leftarrow \mathbb{Z}_p{}^*$ the cloud combines homomorphic authenticator with random r'-value. Then, the cloud sends $proof$=$\{\sigma, \mu, H(r')\}$ to the TPA.

ProofVerify($proof$, Pk, r) The TPA combines the challenge message with r-value received from the user as $H(\sum_{i=s_1}^{s_c} \nu_i r_i)^{-1}$. Upon receiving $proof$, the TPA can verify the integrity as a follow:

$$e(\sigma, g) \stackrel{?}{=} e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H(\mu) \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1} \cdot H\left(r'\right)^{-1}, v\right). \tag{1}$$

by using the properties of the homomorphic hash properties, the correctness of above equation can be shown as follows:

$$\begin{aligned}
e(\sigma, g) &= e\left(\prod_{i=s_1}^{s_c} \sigma_i^{\nu_i}, g\right) \\
&= e\left(\prod_{i=s_1}^{s_c} (H(id||i)H(m_i))^{\nu_i}, g^x\right) \\
&= e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot \prod_{i=s_1}^{s_c} H(m_i)^{\nu_i}, v\right) \\
&= e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i m_i\right) \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right) \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1}\right. \\
&\quad \left. \cdot H\left(r'\right) \cdot H\left(r'\right)^{-1}, v\right) \\
&= e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i (m_i + r_i)\right) \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1} \cdot H\left(r'\right) \cdot H\left(r'\right)^{-1}, v\right) \\
&= e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i (m_i + r_i) + r'\right) \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1} \cdot H\left(r'\right)^{-1}, v\right) \\
&= e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H(\mu) \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1} \cdot H\left(r'\right)^{-1}, v\right).
\end{aligned}$$

4.3 Dynamic update

We show how to design our proposed scheme to support fully dynamic data including block level operations of insertion, modification and deletion. Figures 2 and 3 show some examples which demonstrate the changes using index table. Figure 2a describes that the cloud stores the block $m_2^{'} + r_2^{'}$ and signature $\sigma_2^{'}$. Figure 2b shows that the TPA inserts a new

Index	Block	Signature
1	m_1+r_1	σ_1
2	m_2+r_2	σ_2
3	m_3+r_3	σ_3
⋮	⋮	⋮
n	m_n+r_n	σ_n

Insert →

Index	Block	Signature
1	m_1+r_1	σ_1
2	m_2'+r_2'	σ_2'
3	m_2+r_2	σ_2
4	m_3+r_3	σ_3
⋮	⋮	⋮
n+1	m_n+r_n	σ_n

(a) Insert block $m_2^{'} + r_2^{'}$ in the cloud

Index	r-index	r-value
1	1	r_1
2	2	r_2
3	3	r_3
⋮	⋮	⋮
n	n	r_n

Update →

Index	r-index	r-value
1	1	r_1
2	2	r_2'
3	3	r_2
4	4	r_3
⋮	⋮	⋮
n+1	n+1	r_n

(b) Update r-value in the TPA

Fig. 2 Block insertion process using index table

Index	Block	Signature
1	m_1+r_1	σ_1
2	m_2+r_2	σ_2
3	m_3+r_3	σ_3
4	m_4+r_4	σ_4
⋮	⋮	⋮
n	m_n+r_n	σ_n

Modify → Delete →

Index	Block	Signature
1	m_1'+r_1	σ_1'
2	m_2+r_2	σ_2
3	m_4+r_4	σ_4
⋮	⋮	⋮
n-1	m_n+r_n	σ_n

(a) Modify block m_1+r_1 and delete block m_3+r_3

Index	r-index	r-value
1	1	r_1
2	2	r_2
3	3	r_3
⋮	⋮	⋮
n	n	r_n

Update →

Index	r-index	r-value
1	1	r_1
2	2	r_2
3	3	r_4
4	4	r_5
⋮	⋮	⋮
n-1	n-1	r_n

(b) Update r-value in the TPA

Fig. 3 Block modification and deletion process using index table

random r-value $r_2^{'}$ in the index table. Figure 3a describes that the cloud replaces m_1+r_1 and deletes m_3+r_3. Figure 3b shows that the TPA deletes a random r-value r_3 in the index table.

4.3.1 Block insertion

When a user inserts data, the user must choose a new random r-value r_x , add it to the corresponding block m_x for $1 \leq x \leq n$, and computes the signature σ_x. Then, the user sends an index information i, σ_x and $m_x + r_x$ to the cloud. Upon receiving these, the cloud stores σ_x and $m_x + r_x$ on the index i, and moves all the subsequent blocks of index, say from j to $j + 1$. Also, the user sends an update message which includes i, r-value and r-index, to the TPA so as to update the r-value and r-index in the index table stored in the TPA.

4.3.2 Block modification

When a user wants to modify a specific block of index i in the cloud, the user modifies the block m_i for $1 \leq i \leq n$ to $m_i{}^*$ and and adds existing r_i-value to m_i^*, and generates the signature σ_i^* for $m_i{}^*$. Then, the user sends an index information i, σ_i^* and $m_i{}^* + r_i$ to the cloud. Upon receiving these, the cloud replaces $m_i + r_i$, σ_i with $m_i{}^* + r_i$, σ_i^*.

4.3.3 Block deletion

Block deletion is the opposite operation of block insertion. When a user wants to delete data block of index i from the cloud, the user sends the index information i to the cloud. Then, the cloud deletes the corresponding block and signature, and moves all the blocks of index j into $j - 1$, for $j > i$. Also, the user sends update message to the TPA so as to update the the r-value and r-index in the index table stored in the TPA.

5 Security and performance analysis

5.1 Security analysis

In this section, we provide a security analysis for our scheme into two parts. The first part deals with data integrity guarantee. The second part deals with data privacy guarantee. Security of the proposed scheme is based on the computational Diffie-Hellman assumption (CDH) [3]. The following definition recalls CDH.

Definition 1 CDH problem states that, given $g, g^a, g^b \in \mathbb{G}$, it is computationally intractable to compute the value as g^{ab}.

Data integrity guarantee We need to prove that the untrusted cloud cannot loss and tamper the users' data. This is equivalent to prove the Theorem 1.

Theorem 1 *The untrusted cloud cannot generate a forgery of an auditing proof.*

Proof Our proof is based on Liu et al.'s scheme [11].

Suppose A malicious cloud losses the user's data $m_j + r_j$, and tries to pass an auditing from the TPA.

The malicious cloud outputs $Proof$={σ^*, μ^*, $H(r^*)$} as

$$\sigma^* = \prod_{i=s_1, i\neq j}^{s_c} \sigma_i^{\nu_i}, \quad \mu^* = \sum_{i=s_1, i\neq j}^{s_c} \nu_i(m_i + r_i) + r^*.$$

Since, the *proof* of malicious cloud can also satisfy the (1), the following equation holds.

$$e\left(\sigma^*, g\right) = e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H\left(\mu^*\right) \cdot H\left(r^*\right)^{-1} \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1}, v\right). \tag{2}$$

If we multiply (2) by the expected signature $\sigma_j^{\nu_j}$ which is derived from (1), the following equation holds.

$$e\left(\sigma^*\sigma_j^{\nu_j}, g\right) = e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H\left(\mu^*\right) \cdot H\left(r^*\right)^{-1} \cdot H\left(\nu_j\left(m_j + r_j\right)\right) \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1}, v\right). \tag{3}$$

Then, we divide (3) by (2), we obtain:

$$e\left(\sigma_j^{\nu_j}, g\right) = e\left(H\left(\nu_j(m_j + r_j)\right), v\right). \tag{4}$$

If (4) is succeeds, we can compute $H(id||i)^x$ in forged signature σ^*, it is a violation of the assumption the assumption that the CDH assumption is hard.

Likewise, suppose A malicious cloud tampers the users data $m_j + r_j$ to $\left(m_j + r_j\right)^*$, the user's signature σ_j to σ_j^*, and tries to pass an auditing from the TPA.

The malicious cloud outputs $Proof$={σ^*, μ^*, $H(r^*)$} as

$$\sigma^* = \prod_{i=s_1, i\neq j}^{s_c} \sigma_i^{\nu_i} \cdot \left(\sigma_j^*\right)^{\nu_j}, \quad \mu^* = \sum_{i=s_1, i\neq j}^{s_c} \nu_i\left(m_i + r_i\right) + r^* + \nu_j\left(m_j + r_j\right)^*.$$

Since, the *proof* of malicious cloud can also satisfy the (1), the following equation holds.

$$e\left(\sigma^*, g\right) = e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H(\mu^*) \cdot H\left(r^*\right)^{-1} \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1}, v\right). \tag{5}$$

When we multiply (5) by the signature $\left(\sigma_j \sigma_j^{*-1}\right)^{\nu_j}$, we obtain:

$$e\left(\sigma^* \left(\sigma_j \sigma_j^{*-1}\right)^{\nu_j}, g\right) = e\left(\prod_{i=s_1}^{s_c} H(id||i)^{\nu_i} \cdot H\left(\mu^*\right) \cdot H\left(r^*\right)^{-1} \cdot H\left(\nu_j\left(m_j + r_j\right)\right)\right.$$
$$\left. \cdot H\left(\nu_j\left(m_j + r_j\right)^*\right)^{-1} \cdot H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1}, v\right). \quad (6)$$

When we divide (6) by (5), we obtain:

$$e\left(\sigma_j \sigma_j^{*-1}, g\right) = e\left(H\left(m_j + r_j\right) \cdot H\left(\left(m_j + r_j\right)^*\right)^{-1}, v\right)^{\nu_j}. \quad (7)$$

If (7) is succeeds, we can compute $H(id||i)^x$ in forged signature σ^*, it is a violation of the assumption that the CDH assumption is hard. In short, if success probability of the loss and tamper attacker in our scheme is non-negligible, it solves the CDH problem. This implies that the attacker cannot forge signature σ_i in the proposed scheme. In addition, The TPA combines the challenge message with r-value received from the user as $H\left(\sum_{i=s_1}^{s_c} \nu_i r_i\right)^{-1}$ to verify the *proof* in **ProofVerify**. Accordingly, when the r-value of data file F' in the cloud is equal to r-value in the TPA, the TPA can verify the *proof*. Thus, if the cloud forges the data, r-value of data file F' in the cloud is changed. Therefore, the TPA cannot calculate the correct verification result for the *proof*. □

Data privacy guarantee We want to make sure that the cloud and the TPA cannot derive user's data information during auditing process. This is equivalent to Theorem 2.

Theorem 2 *During auditing process, no information of μ will be leaked to the TPA and no information of data file F will be leaked to the cloud*

Proof We show that the TPA and the cloud cannot derive information from μ. Since the data is hashed in *proof* and obfuscated by random r' as $\mu = \sum_{i=s_i}^{s_c} \nu_i(m_i + r_i) + r'$, where r' is chosen randomly by the cloud. In addition, r' is unknown to the TPA. Therefore, the TPA cannot learn user data. Likewise, Since user data is obfuscated as $F' = \{m_1 + r_1, \ldots, m_n + r_n\}$, where r is random value chosen by the user. In addition, r is unknown to the cloud. Thus, the cloud cannot learn user data. □

5.2 Performance analysis

Communication cost As shown in Table 1, we analyze the communication cost at data auditing process. The size of the challenge message $chal$=$\{i, \nu_i\}_{i \in I}$ is $c \cdot (|s| + |p|)$ bits, where c is the number of select blocks, $|s|$ is the size of set $\{s_1, \ldots, s_c\}$ and $|p|$ is the size of an element of $\mathbb{Z}_p$. The size of an auditing *proof* is $2|p| + c \cdot (|id|)$ bits, where $|id|$ is the size of a block identifier. Communication costs of between the cloud and the TPA are the same as the previous scheme (Wang et al.'s scheme [14] and Liu et al.'s scheme [11]). However, the additional communication cost occurs between the user and the TPA as $n(|p|)$ which is necessary to ensure the security.

Table 1 Communication costs

	Wang et al. [10]	Liu et al. [11]	Proposed protocol
TPA ↔ Cloud	$c \cdot (\lvert s\rvert + \lvert p\rvert)$ $+2\lvert p\rvert + c \cdot (\lvert id\rvert)$	$c \cdot (\lvert s\rvert + \lvert p\rvert)$ $+2\lvert p\rvert + c \cdot (\lvert id\rvert)$	$c \cdot (\lvert s\rvert + \lvert p\rvert)$ $+2\lvert p\rvert + c \cdot (\lvert id\rvert)$
User ↔ TPA	-	-	$n(\lvert p\rvert)$
Total	$c \cdot (\lvert id\rvert + \lvert s\rvert + \lvert p\rvert)$ $+2\lvert p\rvert$	$c \cdot (\lvert id\rvert + \lvert s\rvert + \lvert p\rvert)$ $+2\lvert p\rvert$	$c \cdot (\lvert id\rvert + \lvert s\rvert + \lvert p\rvert)$ $+(n+2)\lvert p\rvert$

Computation cost We will focus on the additional computation overhead as compared to Wang et al.'s scheme [14] and Liu et al.'s scheme [11]. The experiment is conducted using JAVA in windows 7 with an Intel Core i7 processor running at 3.4 GHz, 8GB of RAM. Computation costs compared to the previous scheme [11, 14] are summarized in Table 2. Let $Hash$ is hash value, Add is additions, $Mult$ is multiplications, Exp is exponentiations, $MultExp$ is $\prod$ exponentiation and $Pair$ is paring. Then, suppose there are c random blocks specified in the $chal$ during auditing process. The size of block can be set 20 $byte$, and c is set to be 300 or 460 for high assurance of auditing process. According to Wang et al.'s scheme [14], if the server is missing 1 % of the data file F, the TPA only needs to audit for $c = 300$ or 460 randomly chosen blocks of data file F in order to detect misbehavior with probability lager than 95 % or 99 %, respectively.

Under this setting, we provide the quantify of additional computation cost into the user side and the TPA side. On the user side, additional computation costs of user are $nAdd$ in **SigGen**. As shown in Fig. 4, we implemented the additional computation cost on the user side. The time of addition reaches 8.1 sec for data file with size 1GB, and it increases linearly. It is a small additional computation cost for ensuring the security. Similarly, on the TPA side, additional computation costs of the TPA are $1Hash$, $(c+1)Mult$ in **ProofVerify**. According to previous work [9], we conducted a benchmarking for hash generation time. As a result, when c is 300 and 460, homomorphic hash generation time is $0.509ms$ and $0.780ms$ respectively. It require computationally negligible overhead. Thus, our scheme preserves data privacy against both the cloud and the TPA while requiring a small amount of additional computation overhead.

Table 2 Computataion costs

	Wang et al.'s scheme	Liu et al.'s scheme	Proposed protocol
SigGen	$2Hash$	$2Hash$	$2Hash + n\ Add$
ProofGen	$1Hash + c\ Add$ $+(c+1)\ Mult + 1Exp$ $+c\ MultExp$	$1Hash + c\ Add$ $+c\ Mult + 1Exp$ $+c\ MultExp$	$1Hash + c\ Add$ $+c\ Mult + 1Exp$ $+c\ MultExp$
ProofVerify	$(c+1)\ Hash + 2Mult$ $+2Exp + c\ MultExp$ $+2pair$	$(c+1)\ Hash + 2Mult$ $+1Exp + c\ MultExp$ $+2pair$	$(c+2)\ Hash$ $+(c+3)\ Mult + 2Exp$ $+c\ MultExp + 2pair$

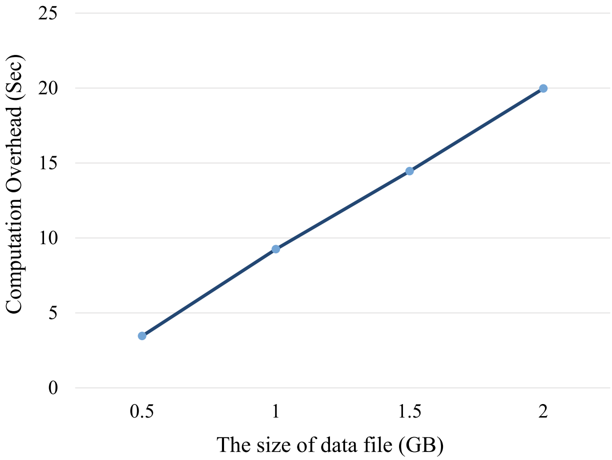

Fig. 4 Additional computation cost on the user side

6 Conclusion

In this paper, we propose a privacy preserving public auditing scheme for educational multimedia data without retrieving the whole data. The proposed scheme is secure against the loss attack and tamper attack. Also, both the cloud and the TPA could not learn user data content by utilizing the random value and homomorphic hash function. Furthermore, our proposed scheme is able to support fully dynamic data. Our scheme ensures data privacy with a small amount of additional computation costs.

Acknowledgments This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2013R1A2A2A01005559). This research was supported by the Chung-Ang University Research Scholarship Grants in 2014.

References

1. Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, Song D (2007) Provable data possession at untrusted stores. In: Proceedings of ACM CCS, pp 598–610
2. Ateniese G, Pietro RD, Mancini LV, Tsudik G (2008) Scalable and efficient provable data possession. In: Proceedings of SecureComm, pp 1–10
3. Bao F, Deng R, Zhu H (2003) Variations of Diffie-Hellman problem. Inf Commun SecurityICICS:301–312
4. Boneh D, Lynn B, Shacham H (2001) Short signatures from the weil pairing, ASIACRYPT 2001. LNCS 2248:514–532
5. Erway C, Kupcu A, Papamanthou C, Tamassia R (2009) Dynamic provable data possession. In: Proceedings of CCS, pp 213–222
6. Gennaro R, Katz J, Krawczyk H, Rabin T (2010) Secure network coding over the integers. Public key cryptography pkc, Springer LNCS, 6056, pp 142–160
7. Govinda Y, Vijaya G (2013) Complete privacy preserving auditing for data integrity in cloud computing. IEEE international conference on trust, security and privacy in computing and communications (TrustCom), pp 1559–1566

8. Juels A, Kaliski BS (2007) PORs: proofs pf retrievability for large files. In: Proceedings of ACM CCS, pp 584–597
9. Krohn M, Freeman M, Mazieres D (2004) On-the-fly verification of rateless erase codes for efficient content distribution. In: Proc. IEEE symposium on security and privacy
10. Li J Study on the development of mobile learning promoted by cloud computing (2010) information engineering and computer science (ICIECS). Second international conference, pp 1–4
11. Liu H, Zhang P, Liu J (2013) Public data integrity verification for secure cloud storage. J Netw:373–380
12. Masud M, Huang X (2012) A novel approach for adopting cloud-based e-learning system. In: IEEE/ACIS 11th international conference on computer and information science (ICIS), pp 37–42
13. Shacham H, Waters B (2008) Compact proofs of retrievability. In: Proc. ASIACRYPT. Springer-Verlag, pp 90–107
14. Wang C, Wang Q, Ren K, Lou W (2010) Privacy preserving public auditing for data storage security in cloud computing. IEEE InfoCom:1–9
15. Wang Q, Wang C, Ren K, Lou W (2011) Enabling public auditability and data dynamics for storage security in cloud computing. IEEE Trans Parallel Distrib Syst:847–859
16. Yang K, Jia X (2013) An efficient and secure dynamic auditing protocol for data storage in cloud computing. IEEE Trans Parallel Distrib Syst 24(9):1717–1726
17. Yu H, Pedrinaci C, Dietze S, Domingue J (2012) Using linked data to annotate search educational video resources for supporting distance learning. IEEE Trans Learn Technologies:130–142
18. Zeng K (2008) Publicly verifiable remote data integrity. In: Proceedings of the international conference on information and communications security, ICICS, pp 419–434
19. Zhu Y, Wang H, Hu Z, Ahn GJ, Hu H, Yau SS (2010) Efficient provable data possession for hybrid clouds. In: Proceedings of the 17th ACM conference on computer and communications security, pp 756–758
20. Zhu Y, Wang H, Hu Z, Ahn GJ, Hu H, Yau SS (2011) Dynamic audit services for integrity verification of outsourced storage in clouds. In: the proceedings of ACM SAC, pp 1550–1557

Daeyoung Kim received the B.S. degree of computer science and engineering from Daejeon University, Daejeon, Korea, in 2014. He is currently pursuing the M.S. degree in the School of Computer Science and Engineering, Chung-Ang University, Korea. His research interests include information security, mobile computing security, cyber security, and applied cryptography.

Hyunsoo Kwon received the B.S. degree of computer science and engineering from Chung-Ang University, Seoul, Korea, in 2014. He is currently pursuing the M.S. degree in the School of Computer Science and Engineering, Chung-Ang University, Korea. His research interests include information security, mobile computing security, cyber security, and applied cryptography.

Changhee Hahn received the B.S. degree of computer science and engineering from Chung-Ang University, Seoul, Korea, in 2014. He is currently pursuing the M.S. degree in the School of Computer Science and Engineering, Chung-Ang University, Korea. His research interests include information security, mobile computing security, cyber security, and applied cryptography.

Junbeom Hur received the B.S. degree from Korea University, Seoul, South Korea, in 2001, and the M.S. and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST) in 2005 and 2009, respectively, all in Computer Science. He was with the University of Illinois at Urbana-Champaign as a postdoctoral researcher from 2009 to 2011. He was with the School of Computer Science and Engineering at the Chung-Ang University, South Korea as an Assistant Professor from 2011 to 2015. He is currently an Assistant Professor with the Department of Computer Science and Engineering at the Korea University, South Korea. His research interests include information security, cloud computing security, mobile security, and applied cryptography.