Cluster Computing (2021) 24:1989–2000
https://doi.org/10.1007/s10586-021-03239-x

Efficient TPA-based auditing scheme for secure cloud storage

Bilin Shao[1] · Yanyan Ji[1]

Received: 12 November 2019 / Revised: 2 July 2020 / Accepted: 13 January 2021 / Published online: 29 January 2021

Abstract
In recent years, how to design efficient auditing protocol to verify the integrity of users' data, which is stored in cloud services provider (CSP), becomes a research focus. Homomorphic message authentication code (MAC) and homomorphic signature are two popular techniques to respectively design private and public auditing protocols. On the one hand, it is not suitable for the homomorphic-MAC-based auditing protocols to be outsourced to third-party auditor (TPA), who has more professional knowledge and computational abilities, although they have high efficiencies. On the other hand, the homomorphic-signature-based ones are very suitable for employing TPA without compromising user's signing key but have very low efficiency (compared to the former case). In this paper, we propose a new auditing protocol, which perfectly combines the advantages of above two cases. In particular, it is almost as efficient as a homomorphic-MAC-based protocol proposed by Zhang et al. recently. Moreover, it is also suitable for outsourcing to TPA because it does not compromise the privacy of users' signing key, which can be seen from our security analysis. Finally, numerical analysis and experimental results demonstrate the high-efficiency of our protocol.

Keywords Cloud storage · Auditing protocol · Homomorphic MAC · Homomorphic signature

1 Introduction

Mobile networks, such as 3G/4G, vehicular ad hoc network (VANET) and Internet of Things (IoTs), generate tremendous amount of data every day [1]. To mitigate the burden of local data storage, subsequent updates and maintenance, people would like to have their data transferred through networks and eventually stored in cloud service provider (CSP) (see [2–4]). However, for some subjective or objective reasons, the data stored in CSP may be corrupted or even lost. In terms of the subjective aspects, a CSP may secretly delete user's data in order to save its own storage space and try to hide the data-lost fact to maintain reputation [5]. From the view of objective aspects, a honest CSP may become vulnerable when encountering the external rival attacks, or internal hardware and software failures [6]. For users, a natural way to detect the cheating of cloud server is to download all their data and audit its integrity. Obviously, it is unrealistic because of the limitations of local space and communication resource. As a result, how to design secure cloud storage system with a reliable audit mechanism becomes extremely important, and hence has attracted a lot of attention of the researchers.

Early systems to consider the integrity of remote data were suggested by Deswarte et al. in [7], Gazzoni Filho and Barreto in [8], and Schwarz and Miller [9]. Later researches focused on the design of security model, including authenticators [10], provable data possession (PDP) [11] and proofs of retrievability (PoR) [12].

Many recent works mainly focused on the problem that the auditing process is private or public. Private auditing, in which the operation of verification is secretly performed between the user and CSP, is the initial model for remote checking of data integrity [12]. Homomorphic message authentication code (MAC) is a popular cryptographic technique to construct private auditing protocols (see [13–15]). Due to high efficiency and low cost, this model is preferred by researchers. Note that, in homomorphic MAC scheme, the signing and verifying keys are same and hence it is unsuited to outsource the auditing process to third

✉ Yanyan Ji
yany_ji@163.com

1 School of Management, Xi'an University of Architecture and Technology, Xi'an 710055, Shaanxi, People's Republic of China

party auditor (TPA) unless the user would like to share its secret key with TPA.

Compared to the original data owner, TPA may have more professional knowledge about auditing and more powerful ability of computing. Hence, outsourcing the process of auditing to TPA will greatly reduce the heavy and unnecessary burdens of users. This is also a popular choice of public auditing protocol and believed to be the right direction of future development. In current public auditing protocols, a basic component is the primitive of homomorphic signature (see [16–19]), which is a public key model of homomorphic MAC and includes a signing key *sk* and a verifying key *vk*. The user authenticates its original data file with this *sk* and transmits *vk* to TPA. Then using *vk*, TPA will audit the integrity of this user's data. Meanwhile, the known of *vk* by TPA does not break the privacy of user's signing key *sk*.

Unfortunately, public auditing protocols have lower computational efficiencies and larger redundancies than homomorphic-MAC-based private protocols. For example, Zhang et al. in [15] compared the performances of those two kinds of models. For a 124M data file, the time consumptions of outsourcing, proving and verifying for a public auditing protocol are 2162, 461 and 632 times of the corresponding processes in their private key auditing.

Hence, how to perfectly combine high efficiency of private auditing protocol with the properties of protecting signer's secret key and being suitable for employing TPA of public auditing protocol becomes an interesting direction of research.

Our Contribution. In this paper, we try to answer the above question. In particular, we propose a new auditing protocol, which has the following features:

- It is almost the same as the homomorphic-MAC-based auditing protocol in terms of efficiency.
- It is suitable for employing TPA to audit data's integrity but does not compromise the privacy of user's signing key.
- It is based on a new technique that falls into neither homomorphic MAC nor homomorphic signature.

In fact, our protocol is mainly based on basic knowledge of linear algebra. More specifically, assume that F is the original data file and can be divided into n blocks, in which each block is an m-dimension vector in $\mathbb{Z}_p$. That is, F has the form of

$$F = \{\widetilde{\mathbf{v}}_1, \widetilde{\mathbf{v}}_2, \cdots, \widetilde{\mathbf{v}}_n\} \in \mathbb{F}_p^{m\times n}.$$

In our auditing protocol, a user first generates the signing key *sk*, which consists of two secret vector $\mathbf{X}_1$ and $\mathbf{X}_2 \in \mathbb{F}_p^{m+n}$, and then linearly derives the corresponding verifying key *vk* from a linear combination of $\mathbf{X}_1, \mathbf{X}_2$. Using this *sk*, the user can compute the tags $\mathbf{t}_i$ for each vector $\mathbf{v}_i$, which, in fact, is the "inner product" between $\mathbf{v}_i$ and $\mathbf{X}_1, \mathbf{X}_2$. When secretly receiving *vk* from the user, TPA can perform the task of auditing by checking a "linear relationship" of *vk* and the responses from CSP. The specific but secret relationship between *vk* and *sk* ensures the correctness of this protocol.

For TPA, it is infeasible to compute *sk* from the derived verification key *vk* since it can not collect enough linear equations and solve $\mathbf{X}_1, \mathbf{X}_2$ from them.

The high efficiency of our protocol lies in that all the processes, including authenticating, generating proof as well as verification, are only the linear operations of vectors.

Related Works. One of the earliest related works about cloud auditing is called "Proof of Retrievability (PoR)", which is proposed by Juels et al. in 2007 [12]. In this system, they used the technique of error-correcting coding to ensure the retrievability of user's data. In fact, this PoR is a typical private auditing solution and does not support employing TPA. At the same time, Atenises et al. originally proposed a public auditing system named provable data possession (PDP) [11], which generates authentication tags based on RSA and audits data's integrity by randomly sampling [20]. In [21], Zhang et al. presented a general framework to design secure PDP schemes using homomorphic encryption schemes [22].

In addition, other factors, including dynamic auditing, privacy protection and batch auditing, are also important considerations in cloud auditing system.

Erway et al. extended Ateniese et al.'s PDP model by proposing an authenticated skip list, which is based on rank, to support dynamic auditing [23]. Their main contribution lies in that they demonstrated a general model to dynamically audit (for TPA): Combining dynamic data structures with the corresponding algorithm of verification. Based on this work, Wang et al. [24] suggested TPA to dynamically audit by using Merkle hash tree, which satisfies not only public auditing but also privacy-protection and batch verification. Nevertheless, the computational costs for TPA are very heavy and communication overheads for updating and verification are also very large. Later, Zhu et al. used index-hash-table (IHT) to design a new public auditing system [25], which maintains data's properties in the party of TPA instead of CSP. As a result, it greatly improves the efficiency for auditing. However, the operation of updating, including insertion and deletion, needs to move (in average) half of the elements in IHT because its structure is sequent. Hence, in [26], Chen and Liu introduced a public auditing scheme based on dynamic hash table (DHT) for TPA to change the situation. Moreover, Shen et al. further presented a new structure, which consists of doubly linked information table and a location array [27]. With this new structure, the computational and communication overheads of their scheme are substantially reduced.

In [28], Chen et al. discussed the possibility of constructing secure cloud auditing protocol from secure network coding scheme (see [29]), although they seem to belong to completely different research areas. Inspired by their work as well as [30], we design our TPA-based auditing protocol, which is not only almost same as the homomorphic-MAC-based auditing protocol in terms of efficiency but also perfectly protects user's signing key.

Organizations. This paper is organized as follows. First, we introduce the system model, the notion of TPA-based auditing protocol and its security definition in Sect. 2. Then, in Sect. 3, we propose our protocol and present the detailed security proof. Discussions on its performance analysis are presented in Sect. 4. Finally, we conclude the whole paper in Sect. 5.

2 Preliminaries

Notations. In the whole paper, we denote by λ the security parameter of algorithms. A boldface such as $\mathbf{v}$ means a row vector and v_i is its i-th component. $\mathbf{v}^T$ is the transposition of $\mathbf{v}$. For two same-dimension vectors $\mathbf{v}_1, \mathbf{v}_2$, $\mathbf{v}_1 \cdot \mathbf{v}_2$ denotes the dot-product operation of them. PPT is an abbreviation of probabilistic polynomial time. A function $f(\lambda)$ is called negligible if, for any $c > 0$, there exists a $\lambda_0 \in \mathbb{Z}$ such that for any $\lambda > \lambda_0$, it holds that $f(\lambda) < \lambda^{-c}$.

2.1 System model

2.1.1 Private key auditing protocol

In a private key auditing protocol, a user, such as a smartphone, tablet computer or laptop, owns a secret key K and wants to store original data file F to cloud. First, it computes the authenticated file F' from K, F and outsources F' to cloud. Then, when intending to check the integrity of F, it generates a challenge query q and gives it to cloud, who will return a proof Γ based on this q and F'. Of course, this process can be run in many times. Finally, the user can know whether its original data file is integral or not by running a verification algorithm based on q, Γ and K. A simple description can be found in Fig. 1.

2.1.2 Public key auditing protocol

In a public key auditing protocol, a user has a pair of keys (sk, vk), in which sk is a secret signing key and vk is the corresponding public verification key. If it wants to employ TPA auditing the integrity of its data file, send vk to this TPA. First, it computes the authenticated F' from sk and a data file F and transmits it to a cloud. Then, the TPA sends a challenge query q and receives a proof Γ from the cloud. Finally, this TPA audits the integrity of F' by running verification algorithm based on q, Γ and vk. A simple description of TPA-based public key auditing protocol can be found in Fig. 2.

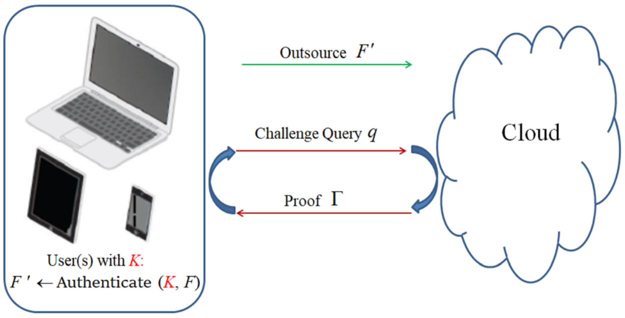


Fig. 1 Private key auditing protocol

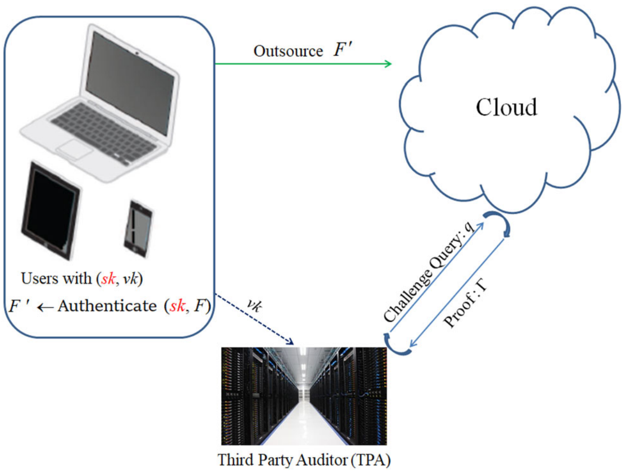


Fig. 2 TPA-Based Public Key Auditing Protocol

2.2 TPA-based auditing protocol

Informally, a TPA-based auditing protocol contains five algorithms: Key-generation algorithm `KeyGen`, authenticating algorithm `Authenticate`, auditing algorithm `Audit`, proof-generation algorithm `ProofGen` and verification algorithm `Verify`. We remark that the key-generation algorithm is run by a KGC or user, the authenticating algorithm is run by a user who will store his/her data file to cloud. Then the auditing and verification algorithms are run by the TPA employed by this user. Finally, the remaining proof-generation algorithm is run by CSP.

Formally, we describe the five algorithms for a TPA-based auditing protocol $\mathtt{TPA-AP}$ as follows.

- $\texttt{KeyGen}$: Take the security parameter λ as input. Output: (sk, vk), where sk and vk will be transmitted to user and TPA, respectively.
- $\texttt{Authenticate}$: The inputs are sk and an original data file F. Assume that F can be divided into n blocks and each block is an m-dimension vector in $\mathbb{Z}_p$. That is

  $$F = (\mathbf{v}_1, \cdots, \mathbf{v}_n) \in \mathbb{F}_p^{m\times n}.$$

  It outputs the processed file

  $$F' = \{(\mathbf{v}_1, t_1), \cdots, (\mathbf{v}_n, t_n)\},$$

  where t_i is the authentication tag of $\mathbf{v}_i$ for $1 \le i \le n$. Then the generated F' will be given to CSP.
- $\texttt{Audit}$: For the input $\ell \in [n]$, it generates the challenge query q. Then the query q will be given to CSP.
- $\texttt{ProofGen}$: Take F' and q as inputs, this algorithm generates a proof $\Gamma = (\mathbf{v}, t)$. Then the generated Γ will be returned to TPA.
- $\texttt{Verify}$: Based on the inputs $vk, q,$ and Γ, it outputs 1 (accept) or 0 (reject). Finally, the auditing result will be informed to user.

The correctness requires that for any $(sk, vk) \leftarrow \texttt{KeyGen}(\lambda)$, all F, $F' \leftarrow \texttt{Authenticate}(sk, F)$, any $q \leftarrow \texttt{Audit}(vk)$ and $\Gamma \leftarrow \texttt{ProofGen}(F', q)$, it holds that

$$1 \leftarrow \texttt{Verify}(vk, q, \Gamma).$$

2.3 Security model

As for the security of $\texttt{TPA}-\texttt{AP}$ in the above section, we consider it from three aspects: Integrity, restorability (of the original data file) and the security of user's signing key against TPA.

Integrity. First, same as previous work [28, 15], the CSP is modeled as a potentially malicious adversary. In practice, the CSP has the outsourced data file F' and it can also see many past auditing queries q_i from TPA and the corresponding answers Γ_i. In addition, it is also reasonable to stipulate that the CSP knows if TPA accepts those proofs from its manners. In terms of security, we consider the cheating behavior of CSP. That is, it may not correctly generate the proof according to the challenge query from TPA since it may lost the user's original data. Hence, we consider the first experiment played by an adversary $\mathcal{CSP}$ (standing for a malicious CSP) and a challenger CH as follows.

$\underline{\mathrm{Exp}_{\texttt{TPA-AP}}^{\mathcal{CSP}}(\lambda)}$:

- **Setup.** The challenger first runs

  $$(sk, vk) \leftarrow \texttt{KeyGen}(\lambda), \text{ and } F' \leftarrow \texttt{Authenticate}(sk, F),$$

  where $F = (\mathbf{v}_1, \cdots, \mathbf{v}_n) \in \mathbb{F}_p^{m\times n}$ is a data file chosen by CH. Then give $F' = \{(\mathbf{v}_1, t_1), \cdots, (\mathbf{v}_n, t_n)\}$ to $\mathcal{CSP}$, and initialize an empty list L, which will be used to store the following queries.
- **Queries.** This process can be run for polynomial times. First, the challenger computes

  $$q \leftarrow \texttt{Audit}(\ell),$$

  and return q to $\mathcal{CSP}$. Then $\mathcal{CSP}$ generates a proof $\Gamma = (\mathbf{v}, t)$ and gives it to CH. Finally, the challenger runs

  $$\delta \leftarrow \texttt{Verify}(vk, q, \Gamma),$$

  and gives δ to $\mathcal{CSP}$. Add (q, Γ) to L.
- **Challenge.** $\mathcal{CSP}$ outputs a pair of $(q^*, \Gamma^*) = (q^*, (\mathbf{v}^*, t^*))$, which does not appear in L, and $\mathbf{v}^*$ is not computed from F' according to q^*.

We call $\mathcal{CSP}$ wins the game if

$$1 \leftarrow \texttt{Verify}(vk, q^*, \Gamma^*).$$

Define $\mathrm{Adv}_{\texttt{TPA-AP}}^{\mathcal{CSP}}(\lambda)$ as the advantage of $\mathcal{CSP}$ winning the game. That is,

$$\mathrm{Adv}_{\texttt{TPA-AP}}^{\mathcal{CSP}}(\lambda) := \Pr[\mathcal{CSP} \text{ wins}].$$

Restorability. Next, we consider the restorability of user's original data if the proofs provided by CSP can pass the verifications of TPA. In other words, for a secure auditing protocol, the user or TPA should have the ability to extract original data from the auditing queries and their corresponding proofs. Therefore, we define the following experiment, which is similar to that of [28] except that it is played by TPA (with a challenger) instead of a user, to estimate the probability of extracting for original data.

$\underline{\mathrm{Exp}_{\texttt{TPA-AP}}^{\mathrm{TPA-Extract}}(\lambda)}$:

- **Setup.** A challenger first runs

  $$(sk, vk) \leftarrow \texttt{KeyGen}(\lambda), \text{ and } F' \leftarrow \texttt{Authenticate}(sk, F).$$

  Then give vk to TPA.
- **Queries.** This process can be run for polynomial times. First, TPA computes

  $$q \leftarrow \texttt{Audit}(\ell),$$

  and return q to the challenger, who generates a proof Γ by running $\texttt{ProofGen}(F', q)$. Then TPA obtains this Γ and computes

  $$\delta \leftarrow \texttt{Verify}(vk, q, \Gamma).$$

- **Output.** Finally, TPA outputs F^*.

If $F = F^*$, then TPA is called winning the experiment. Here, we denote by $\mathrm{Adv}_{\mathtt{TPA-AP}}^{\mathrm{TPA-Extract}}(\lambda)$ the probability of TPA winning this experiment.

Signing Key's Security. Finally, we consider the security of user's signing key against TPA. That is, TPA is now modeled as a honest but curious adversary: Although honestly executing the audit protocol, he is also curious about user's signing key and tries to analyze it according to his current knowledge. It is rather important to protect user's key since anyone does not hope agent (i.e. TPA) knowing his/her signing key (if possible). Therefore, we define the following experiment, which is played by an honest but curious adversary $\mathcal{TPA}$ and a challenger.

$\underline{\mathrm{Exp}_{\mathtt{TPA-AP}}^{\mathcal{TPA}}(\lambda):}$

- **Setup** and **Queries** are same as the above experiment $\mathrm{Exp}_{\mathtt{TPA-AP}}^{\mathrm{TPA-Extract}}(\lambda)$.
- **Output.** Finally, $\mathcal{TPA}$ outputs sk'.

If $sk = sk'$, then we call TPA wins the game. Here, we denote by $\mathrm{Adv}_{\mathtt{TPA-AP}}^{\mathcal{TPA}}(\lambda)$ the probability of TPA winning this game.

Summarizing the above discussions, we formally give the security of TPA-based auditing protocol $\mathtt{TPA-AP}$ as follows.

Definition 1 A (p, m, n) TPA-based auditing protocol $\mathtt{TPA-AP}$, consisting of $\mathtt{KeyGen}$, $\mathtt{Authenticate}$, $\mathtt{Audit}$, $\mathtt{ProofGen}$ and $\mathtt{Verify}$, is secure if for any PPT $\mathcal{CSP}$, $\mathcal{TPA}$, and some TPA, the probabilities $\mathrm{Adv}_{\mathtt{TPA-AP}}^{\mathcal{CSP}}(\lambda)$ and $\mathrm{Adv}_{\mathtt{TPA-AP}}^{\mathcal{TPA}}(\lambda)$ are negligible, and $\mathrm{Adv}_{\mathtt{TPA-AP}}^{\mathrm{TPA-Extract}}(\lambda)$ is overwhelming.[1]

3 Our proposed auditing protocol and its security analysis

3.1 The proposed protocol

First, we present our proposed TPA-based auditing protocol $\mathtt{TPA-AP}$ as follows.

- $\mathtt{KeyGen}$: For the input of λ, it randomly chooses

  $$\mathbf{X}_1,\ \mathbf{X}_2 \in \mathbb{F}_p^{m+n},\ \mathbf{Z} = (Z_1, Z_2) \in \mathbb{F}_p \times \mathbb{F}_p$$

  and computes

  $$\mathbf{Q} = Z_1\mathbf{X}_1 + Z_2\mathbf{X}_2 \in \mathbb{F}_p^{m+n}. \tag{1}$$

  Define $sk = X := \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix}$ and $vk = (\mathbf{Q}, \mathbf{Z})$.
- $\mathtt{Authenticate}(sk, F)$: For the data file F, parse it into

  $$F = \{\widetilde{\mathbf{v}}_1, \widetilde{\mathbf{v}}_2, \cdots, \widetilde{\mathbf{v}}_n\} \in \mathbb{F}_p^{mn}.$$

  For $1 \le i \le n$, augment $\widetilde{\mathbf{v}}_i$ as

  $$\mathbf{v}_i := \left(\widetilde{\mathbf{v}}_i, \underbrace{\overbrace{0, \cdots, 0, 1}^{i}, 0, \cdots, 0}_{n} \right) \in \mathbb{F}_p^{m+n}.$$

  Then for $\mathbf{v}_i$, compute

  $$t_{\mathbf{v}_i,1} = \mathbf{v}_i \cdot \mathbf{X}_1 \in \mathbb{F}_p,\ t_{\mathbf{v}_i,2} = \mathbf{v}_i \cdot \mathbf{X}_2 \in \mathbb{F}_p.$$

  Define $\mathbf{t}_i := (t_{\mathbf{v}_i,1}, t_{\mathbf{v}_i,2})$ as the tag of $\mathbf{v}_i$. Finally, output the processed data file

  $$F' = \{(\widetilde{\mathbf{v}}_1, \mathbf{t}_1), \cdots, (\widetilde{\mathbf{v}}_n, \mathbf{t}_n)\}.$$

- $\mathtt{Audit}(\ell)$: Randomly choose $1 \le i_1 < i_2 < \cdots < i_\ell \le n$ and $c_1, c_2, \cdots, c_\ell$ from the finite field $\mathbb{F}_p$, and set the challenge message as

  $$q = \{(i_\tau, c_\tau)\}_{\tau=1,\cdots,\ell}.$$

- $\mathtt{ProofGen}(F', q)$: Parse F' as $\{(\widetilde{\mathbf{v}}_1, \mathbf{t}_1), \cdots, (\widetilde{\mathbf{v}}_n, \mathbf{t}_n)\}$ and q as $\{(i_\tau, c_\tau)\}_{\tau=1,\cdots,\ell}$. Then compute

  $$\widetilde{\mathbf{v}} = \sum_{\tau=1}^{\ell} c_\tau \widetilde{\mathbf{v}}_{i_\tau},$$

  and

  $$\mathbf{t} = \sum_{\tau=1}^{\ell} c_\tau \mathbf{t}_{i_\tau} = \sum_{\tau=1}^{\ell} \left(c_\tau t_{\mathbf{v}_{i_\tau},1}, c_\tau t_{\mathbf{v}_{i_\tau},2} \right)$$
  $$= \left(\sum_{\tau=1}^{\ell} c_\tau t_{\mathbf{v}_{i_\tau},1}, \sum_{\tau=1}^{\ell} c_\tau t_{\mathbf{v}_{i_\tau},2} \right).$$

  Finally, output the proof $\Gamma = (\widetilde{\mathbf{v}}, \mathbf{t})$.
- $\mathtt{Verify}(vk, q, \Gamma)$: Parse $vk = (\mathbf{Q}, \mathbf{Z})$, $q = \{(i_\tau, c_\tau)\}_{\tau=1,\cdots,\ell}$, and $\Gamma = (\widetilde{\mathbf{v}}, \mathbf{t})$. Then augment $\widetilde{\mathbf{v}}$ as $\mathbf{v} := (\widetilde{\mathbf{v}}, \mathbf{e}) \in \mathbb{F}_p^{m+n}$. Here, $\mathbf{e}$ is an n-dimensional vector and computed as follows:

  $$\mathbf{e} := c_1\mathbf{e}_{i_1} + \cdots + c_\ell \mathbf{e}_{i_\ell},$$

  where $\mathbf{e}_{i_\tau} = (\overbrace{0, \cdots, 0, 1}^{i_\tau}, 0, \cdots, 0) \in \mathbb{F}_p^n$. Finally, check if

  $$\mathbf{v} \cdot \mathbf{Q} = \mathbf{Z} \cdot \mathbf{t}. \tag{2}$$

  If it is, output 1; else output 0.

The correctness of this scheme can be verified as follows. For clarity, we only present the correctness for single vector $\mathbf{v}_1$ instead of the above "combined" $\mathbf{v}$, whose derivation can be easily known from the linearity of (2). In particular, for the tag $\mathbf{t}_1 = (t_{\mathbf{v}_1,1}, t_{\mathbf{v}_1,2})$ of $\mathbf{v}_1$, it holds that

[1] Here, a function $h(\lambda)$ of λ is called overwhelming if $1 - h(\lambda)$ is negligible.

$t_{\mathbf{v}_1,1} = \mathbf{v}_1 \cdot \mathbf{X}_1 = \mathbf{X}_1\mathbf{v}_1^T,\ t_{\mathbf{v}_1,2} = \mathbf{v}_1 \cdot \mathbf{X}_2 = \mathbf{X}_2\mathbf{v}_1^T.$

Hence,

$$\begin{aligned}\mathbf{v}_1 \cdot \mathbf{Q} = \mathbf{Q}\mathbf{v}_1^T &= (Z_1\mathbf{X}_1 + Z_2\mathbf{X}_2)\mathbf{v}_1^T \\ &= Z_1\mathbf{X}_1\mathbf{v}_1^T + Z_2\mathbf{X}_2\mathbf{v}_1^T = \mathbf{Z}\cdot\mathbf{t}_1.\end{aligned} \tag{3}$$

Remark 1 In practical auditing protocol, the current key generation may bring heavy work for user or KGC since the secret vectors $\mathbf{X}_1, \mathbf{X}_2$ are of the same sizes as each *augmented* block $\mathbf{v}_i$, which may be several hundreds or even more. However, we can use cryptographic pseudo-random function (PRF) to generate the long keys. Concretely, let $F : \mathcal{K}_F \to \mathbb{F}_p^{m+n}$ be some public known PRF and k_1, k_2 are two seeds randomly chosen from $\mathcal{K}_F$. Then the (long) keys $\mathbf{X}_1, \mathbf{X}_2$ can be generated by running

$\mathbf{X}_1 \leftarrow F(k_1),\ \mathbf{X}_2 \leftarrow F(k_2).$

3.2 Security analysis

Since the security model in Sect. 2.3 consists of three aspects, we give the security analysis of $\mathtt{TPA-AP}$ by using the following lemmas.

Lemma 1 (***Integrity***) *For a malicious $\mathcal{CSP}$ attacking on the (p, m, n) TPA-based protocol $\mathtt{TPA-AP}$, it holds that*

$$\mathrm{Adv}^{\mathcal{CSP}}_{\mathtt{TPA-AP}}(\lambda) = \frac{1}{p}. \tag{4}$$

Proof Consider the security experiment $\mathrm{Exp}^{\mathcal{CSP}}_{\mathtt{TPA-AP}}(\lambda)$ on our proposed scheme $\mathtt{TPA-AP}$ as follows.

- **Setup.** The challenger first generates $sk = X$ and $vk = (\mathbf{Q}, \mathbf{Z})$ according to the algorithm $\mathtt{KeyGen}$. Then choose a data file F, which is divided into

  $F = (\tilde{\mathbf{v}}_1, \cdots, \tilde{\mathbf{v}}_n) \in \mathbb{F}_p^{m\times n},$

  and run

  $F' := \{(\tilde{\mathbf{v}}_1, \mathbf{t}_1), \cdots, (\tilde{\mathbf{v}}_n, \mathbf{t}_n)\} \leftarrow \mathtt{Authenticate}(sk, F),$

  where, for $1 \le i \le n$, $\mathbf{t}_i = (t_{\mathbf{v}_i,1}, t_{\mathbf{v}_i,2})$ is the authentication tag of $\tilde{\mathbf{v}}_i$. Give F' to $\mathcal{CSP}$, and initialize an empty list L, which will be used to store the following queries.
- **Queries.** This process can be run for polynomial times. First, the challenger randomly chooses $1 \le i_1 < i_2 < \cdots < i_\ell \le n$ and $c_1, c_2, \cdots, c_\ell$ from the finite field $\mathbb{F}_p$, and gives

  $q = \{(i_\tau, c_\tau)\}_{\tau=1,\cdots,\ell}$

  to $\mathcal{CSP}$. Then $\mathcal{CSP}$ responses with a proof $\Gamma = (\tilde{\mathbf{v}}, \mathbf{t}) \in \mathbb{F}_p^{m+2}$. After that, the challenger runs

  $\delta \leftarrow \mathtt{Verify}(vk, q, \Gamma),$

  and gives δ to $\mathcal{CSP}$. Add (q, Γ) to L.
- **Challenge.** $\mathcal{CSP}$ outputs a pair of

  $(q^*, \Gamma^*) = \left(\{(i_\tau^*, c_\tau^*)\}_{\tau=1,\cdots,\ell}, (\tilde{\mathbf{v}}^*, \mathbf{t}^*)\right),$

  which does not appear in L, and $\tilde{\mathbf{v}}^*$ is not computed from F' according to q^*. Or equivalently, the augmented vector $\mathbf{v}^* = (\tilde{\mathbf{v}}^*, \mathbf{e}^*)$ of $\tilde{\mathbf{v}}^*$, where

  $\mathbf{e}^* = c_1^*\mathbf{e}_{i_1^*} + \cdots + c_\ell^*\mathbf{e}_{i_\ell^*},$

  does not belong to

  $\mathrm{Span}\{\mathbf{v}_1, \cdots, \mathbf{v}_n\}.$

Now, we analyze the advantage of $\mathcal{CSP}$. First, from the processed

$F' = \{(\tilde{\mathbf{v}}_1, \mathbf{t}_1), \cdots, (\tilde{\mathbf{v}}_n, \mathbf{t}_n)\},$

$\mathcal{CSP}$ can get the following system:

$$\begin{cases}\mathbf{v}_1 \cdot \mathbf{X}_1 = \mathbf{v}_1\mathbf{X}_1^T = t_{\mathbf{v}_1,1},\\ \mathbf{v}_1 \cdot \mathbf{X}_2 = \mathbf{v}_1\mathbf{X}_2^T = t_{\mathbf{v}_1,2},\\ \qquad\vdots \\ \mathbf{v}_n \cdot \mathbf{X}_1 = \mathbf{v}_n\mathbf{X}_1^T = t_{\mathbf{v}_n,1},\\ \mathbf{v}_n \cdot \mathbf{X}_1 = \mathbf{v}_n\mathbf{X}_2^T = t_{\mathbf{v}_n,1},\end{cases} \tag{5}$$

where $\mathbf{v}_i$ is the augmented vector of $\tilde{\mathbf{v}}_i$ for $1 \le i \le n$. Let A be the coefficient matrix of (5). Then its rank $r(A) \le 2n$.

If its output (q^*, Γ^*) can pass the verification of $\mathtt{Verify}$, then it holds that

$\mathbf{v}^* \cdot \mathbf{Q} = \mathbf{Z} \cdot \mathbf{t}^*.$

Combing it with (1), we know that

$$Z_1\mathbf{v}^* \cdot \mathbf{X}_1 + Z_2\mathbf{v}^* \cdot \mathbf{X}_2 = \mathbf{v}^* \cdot \mathbf{Q} = Z_1 t_{\mathbf{v}^*,\mathbf{1}} + Z_2 t_{\mathbf{v}^*,2}. \tag{6}$$

Since $\mathbf{v}^* \notin \mathrm{Span}\{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$, we can easily know that (6) is not the linear combination of the equations in (5). That is, the rank of the new system consisting of (5) and (6) equals to $r(A) + 1\ (\le 2n + 1)$. However, there are totally $2(n + m)$ unknowns in $\mathbf{X}_1$ and $\mathbf{X}_2$. Therefore, the event that the output of $\mathcal{CSP}$ satisfies (6) occurs with probability at most

$$\frac{p^{2n+1}}{p^{2(n+m)}} = \frac{1}{p^{2m-1}} \le \frac{1}{p}\ (\text{for } m \ge 1).$$

Moreover, $\mathcal{CSP}$ can win the game with probability at least $1/p$ by randomly outputting (q^*, Γ^*) since the two sides of (6) are both elements of $\mathbb{F}_p$.

As a result, we know that (4) holds. This also ends the proof of Lemma 1.

Lemma 2 (***Restorability***) *For the proposed protocol* $\texttt{TPA}-\texttt{AP}$, *there exists an honest TPA that can extract original data file with probability* 1. *That is,*

$$\mathrm{Adv}_{\texttt{TPA}-\texttt{AP}}^{\mathrm{TPA-Extract}}(\lambda)=1. \tag{7}$$

Proof Now, we consider the experiment $\mathrm{Exp}_{\texttt{TPA}-\texttt{AP}}^{\mathrm{TPA-Extract}}(\lambda)$ on our proposed scheme $\texttt{TPA}-\texttt{AP}$. Concretely,

- **Setup.** A challenger first generates $sk=X$ and $vk=(\mathbf{Q},\mathbf{Z})$ according to the algorithm $\texttt{KeyGen}$ and gives vk to TPA. Then randomly choose a data file F, which is divided into

  $$F=(\widetilde{\mathbf{v}}_1,\cdots,\widetilde{\mathbf{v}}_n)\in\mathbb{F}_p^{m\times n},$$

  and run

  $$F':=\{(\widetilde{\mathbf{v}}_1,\mathbf{t}_1),\cdots,(\widetilde{\mathbf{v}}_n,\mathbf{t}_n)\}\leftarrow\texttt{Authenticate}(sk,F),$$

  where, for $1\le i\le n$, $\mathbf{t}_i=(t_{\mathbf{v}_i,1},t_{\mathbf{v}_i,2})$ is the authentication tag of $\widetilde{\mathbf{v}}_i$.
- **Queries.** For the ηth query ($\eta\le n$), TPA chooses

  $$\mathbf{c}_\eta=(c_{\eta,1},c_{\eta,2},\cdots,c_{\eta,n}),$$

  which is linearly independent of the vectors $\mathbf{c}_1,\cdots,\mathbf{c}_{\eta-1}$ chosen in former queries, and submits $q=\{(j,c_{\eta,j})_{j=1}^n\}$ to the challenger. Then the challenger computes

  $$\widetilde{\mathbf{v}}^{(\eta)}:=\sum_{j=1}^n c_{\eta,j}\widetilde{\mathbf{v}}_j,\ \text{and}\ \mathbf{t}^{(\eta)}:=\sum_{j=1}^n c_{\eta,j}\mathbf{t}_j.$$

  Set $\Gamma^{(\eta)}=(\widetilde{\mathbf{v}}^{(\eta)},\mathbf{t}^{(\eta)})$ and give it to TPA.
- **Output.** After n times queries to the challenger, TPA obtains the following system:

  $$\begin{cases} c_{1,1}\widetilde{\mathbf{v}}_1+c_{1,2}\widetilde{\mathbf{v}}_2+\cdots+c_{1,n}\widetilde{\mathbf{v}}_n=\widetilde{\mathbf{v}}^{(1)},\\ c_{2,1}\widetilde{\mathbf{v}}_1+c_{2,2}\widetilde{\mathbf{v}}_2+\cdots+c_{2,n}\widetilde{\mathbf{v}}_n=\widetilde{\mathbf{v}}^{(2)},\\ \qquad\vdots\\ c_{n,1}\widetilde{\mathbf{v}}_1+c_{n,2}\widetilde{\mathbf{v}}_2+\cdots+c_{n,n}\widetilde{\mathbf{v}}_n=\widetilde{\mathbf{v}}^{(n)}.\end{cases} \tag{8}$$

  Since the chosen vectors $\mathbf{c}_1,\mathbf{c}_2,\cdots,\mathbf{c}_n$ are linearly independent, the coefficient matrix of (8) has rank n. Therefore, the original data vectors $\widetilde{\mathbf{v}}_1,\widetilde{\mathbf{v}}_2,\cdots,\widetilde{\mathbf{v}}_n$ can be restored by solving the linear system (8).

In other words, for this TPA, it can restore the data file F with probability 1. This ends the proof of Lemma 2.

Lemma 3 (***Signing Key's Security***) *If $n<m$, then for an honest but curious adversary $\mathcal{TPA}$ against the above protocol* $\texttt{TPA}-\texttt{AP}$, *the probability of recovering user's signing key is at most $1/p^{m-n}$. That is,*

$$\mathrm{Adv}_{\texttt{TPA}-\texttt{AP}}^{\mathcal{TPA}}(\lambda)\le\frac{1}{p^{m-n}}. \tag{9}$$

Proof Consider the experiment $\mathrm{Exp}_{\texttt{TPA}-\texttt{AP}}^{\mathcal{TPA}}(\lambda)$ on $\texttt{TPA}-\texttt{AP}$ as follows.

- **Setup.** Same as that of Lemma 2.
- **Queries.** This process can be run for polynomial times. First, $\mathcal{TPA}$ randomly chooses $1\le i_1<i_2<\cdots<i_\ell\le n$ and $c_1,c_2,\cdots,c_\ell$ from the finite field $\mathbb{F}_p$, and gives

  $$q=\{(i_\tau,c_\tau)\}_{\tau=1,\cdots,\ell}$$

  to the challenger. Then the challenger computes

  $$\widetilde{\mathbf{v}}=\sum_{\tau=1}^{\ell}c_\tau\widetilde{\mathbf{v}}_{i_\tau},\ \text{and}\ \mathbf{t}=\sum_{\tau=1}^{\ell}c_\tau\mathbf{t}_{i_\tau}.$$

  Set $\Gamma=(\widetilde{\mathbf{v}},\mathbf{t})$ and return it to $\mathcal{TPA}$.
- **Output.** Finally, $\mathcal{TPA}$ outputs a guess sk' of user's signing key sk.

Now, we bound the probability of $sk=sk'$. From the queries and the corresponding responses, $\mathcal{TPA}$ can *at most* collects

$$F'=\{(\widetilde{\mathbf{v}}_1,\mathbf{t}_1),\cdots,(\widetilde{\mathbf{v}}_n,\mathbf{t}_n)\}.$$

In other words, it can at most get the equations:

$$\begin{cases} \mathbf{v}_1\cdot\mathbf{X}_1=\mathbf{v}_1\mathbf{X}_1^T=t_{\mathbf{v}_1,1},\\ \mathbf{v}_1\cdot\mathbf{X}_2=\mathbf{v}_1\mathbf{X}_2^T=t_{\mathbf{v}_1,2},\\ \qquad\vdots\\ \mathbf{v}_n\cdot\mathbf{X}_1=\mathbf{v}_n\mathbf{X}_1^T=t_{\mathbf{v}_n,1},\\ \mathbf{v}_n\cdot\mathbf{X}_1=\mathbf{v}_n\mathbf{X}_2^T=t_{\mathbf{v}_n,1},\end{cases} \tag{10}$$

in which, for $1\le i\le n$, $\mathbf{v}_i$ is the augmented vector of $\widetilde{\mathbf{v}}_i$ and $t_{\mathbf{v_i},\mathbf{1}},t_{\mathbf{v}_i,1}$ are two components of $\mathbf{t}_i$.

In addition, $\mathcal{TPA}$ also has the verification key $vk=(\mathbf{Q},\mathbf{Z})$, which satisfies

$$\mathbf{Q}=Z_1\mathbf{X}_1+Z_2\mathbf{X}_2\in\mathbb{F}_p^{m+n}. \tag{11}$$

Combining (10) with (11), $\mathcal{TPA}$ can at most obtain a system including $2n+(m+n)=3n+m$ equations and $2(n+m)$ variables. Hence, it can correctly find user's signing key $X=(\mathbf{X}_1,\mathbf{X}_2)$ with probability at most

$$\frac{p^{3n+m}}{p^{2(n+m)}}=\frac{1}{p^{m-n}}.$$

This ends the proof of Lemma 3.

Remark 2 Here, we discuss the rationality of the assumption "$n<m$" in the conditions of Lemma 3. In

general, for a data file F with a fix size, such as dozens of Kilobytes (K) or hundreds of Megabytes (M), the parameter n is usually set as its unit-number while m is its concrete number in this unit. For example, a file F with size of 20 K can be divided into 20 blocks and each block has size of 1024 bytes. That is, $n = 20$ and $m = 1024$. Hence, the assumption of "$n < m$" is habitual but crucial in this lemma.

Putting all the facts of Lemmas 1, 2 and 3 together, we have the following:

Theorem 1 *If $n < m$, then our proposed auditing protocol* $\mathtt{TPA-AP}$ *is secure. Concretely, for any PPT $\mathcal{CSP}, \mathcal{TPA}$ and some TPA, their advantages in the corresponding experiments* $\mathrm{Exp}^{\mathcal{CSP}}_{\mathtt{TPA-AP}}(\lambda)$, $\mathrm{Exp}^{\mathrm{TPA-Extract}}_{\mathtt{TPA-AP}}(\lambda)$ *and* $\mathrm{Exp}^{\mathcal{TPA}}_{\mathtt{TPA-AP}}(\lambda)$ *satisfy* (4), (7) *and* (9), *respectively.*

3.3 Domain extension for authentication tags

Note that in the security analysis of the proposed protocol $\mathtt{TPA-AP}$ presented in Sect. 3.2, a malicious $\mathcal{CSP}$ can cheat the user or TPA with probability $1/p$, and an honest but curious $\mathcal{TPA}$ can recover user's signing key with probability $1/p^{m-n}$. However, in practice, the predetermined parameter p is typically set as $2^8 = 256$. Therefore, this security level for $\mathtt{TPA-AP}$ may not always be sufficient. In order to enhance its security, we can extend authentication tag for each vector to multiple ones. Hence, we introduce the following extended auditing protocol $\mathtt{E-TPA-AP}$.

- $\mathtt{E-KeyGen}$: For the input of λ, it randomly chooses

  $$\left(\mathbf{X}_1^{(1)}, \mathbf{X}_2^{(1)}\right), \cdots, \left(\mathbf{X}_1^{(r)}, \mathbf{X}_2^{(r)}\right) \in \mathbb{F}_p^{m+n},$$

  $$\mathbf{Z}^{(1)} = (Z_1^{(1)}, Z_2^{(1)}), \cdots, \mathbf{Z}^{(r)} = (Z_1^{(r)}, Z_2^{(r)}) \in \mathbb{F}_p \times \mathbb{F}_p,$$

  and computes

  $$\begin{cases} \mathbf{Q}^{(1)} = Z_1^{(1)}\mathbf{X}_1^{(1)} + Z_2^{(1)}\mathbf{X}_2^{(1)} \in \mathbb{F}_p^{m+n}, \\ \vdots \\ \mathbf{Q}^{(r)} = Z_1^{(r)}\mathbf{X}_1^{(r)} + Z_2^{(r)}\mathbf{X}_2^{(r)} \in \mathbb{F}_p^{m+n}. \end{cases}$$

  Define

  $$sk = X := \left(\left(\mathbf{X}_1^{(1)}, \mathbf{X}_2^{(1)}\right), \cdots, \left(\mathbf{X}_1^{(r)}, \mathbf{X}_2^{(r)}\right)\right)$$

  and

  $$vk = \left((\mathbf{Q}^{(1)}, \mathbf{Z}^{(1)}), \cdots, (\mathbf{Q}^{(r)}, \mathbf{Z}^{(r)})\right).$$

- $\mathtt{E-Authenticate}(sk, F)$: For the data file F, parse it into

  $$F = \{\widetilde{\mathbf{v}}_1, \widetilde{\mathbf{v}}_2, \cdots, \widetilde{\mathbf{v}}_n\} \in \mathbb{F}_p^{m\times n}.$$

  For $1 \le i \le n$, augment $\widetilde{\mathbf{v}}_i$ as

  $$\mathbf{v}_i := \left(\widetilde{\mathbf{v}}_i, \underbrace{\overbrace{0, \cdots, 0, 1}^{i}, 0, \cdots, 0}_{n}\right) \in \mathbb{F}_p^{m+n}.$$

  Then for $\mathbf{v}_i$, compute

  $$\begin{cases} t^{(1)}_{\mathbf{v}_i,1} = \mathbf{v}_i \cdot \mathbf{X}_1^{(1)}, \ t^{(1)}_{\mathbf{v}_i,2} = \mathbf{v}_i \cdot \mathbf{X}_2^{(1)}, \\ \vdots \\ t^{(r)}_{\mathbf{v}_i,1} = \mathbf{v}_i \cdot \mathbf{X}_1^{(r)}, \ t^{(r)}_{\mathbf{v}_i,2} = \mathbf{v}_i \cdot \mathbf{X}_2^{(r)}. \end{cases}$$

  Define

  $$\mathbf{t}_i = \left(\mathbf{t}_i^{(1)}, \cdots, \mathbf{t}_i^{(r)}\right) = \left((t^{(1)}_{\mathbf{v}_i,1}, t^{(1)}_{\mathbf{v}_i,2}), \cdots, (t^{(r)}_{\mathbf{v}_i,1}, t^{(r)}_{\mathbf{v}_i,2})\right)$$

  as the tag of $\mathbf{v}_i$. Finally, output the processed data file $F' = \{(\widetilde{\mathbf{v}}_1, \mathbf{t}_1), \cdots, (\widetilde{\mathbf{v}}_n, \mathbf{t}_n)\}$.

- $\mathtt{E-Audit}(vk)$: Same as $\mathtt{Audit}(vk)$.
- $\mathtt{E-ProofGen}(F', q)$: Same as $\mathtt{ProofGen}(F', q)$ except that

  $$\begin{aligned} \mathbf{t} &= \sum_{\tau=1}^{\ell} c_\tau \mathbf{t}_{i_\tau} \\ &= \sum_{\tau=1}^{\ell}\left(\left(c_\tau t^{(1)}_{\mathbf{v}_{i_\tau},1}, c_\tau t^{(1)}_{\mathbf{v}_{i_\tau},2}\right), \cdots, \left(c_\tau t^{(r)}_{\mathbf{v}_{i_\tau},1}, c_\tau t^{(r)}_{\mathbf{v}_{i_\tau},2}\right)\right) \\ &= \left(\left(\sum_{\tau=1}^{\ell} c_\tau t^{(1)}_{\mathbf{v}_{i_\tau},1}, \sum_{\tau=1}^{\ell} c_\tau t^{(1)}_{\mathbf{v}_{i_\tau},2}\right), \right. \\ &\quad \left. \cdots, \left(\sum_{\tau=1}^{\ell} c_\tau t^{(r)}_{\mathbf{v}_{i_\tau},1}, \sum_{\tau=1}^{\ell} c_\tau t^{(r)}_{\mathbf{v}_{i_\tau},2}\right)\right) \\ &:= \left((\mathbf{t}^{(1)}, \cdots, \mathbf{t}^{(r)}\right). \end{aligned}$$

  Finally, output the proof $\Gamma = (\widetilde{\mathbf{v}}, \mathbf{t})$.
- $\mathtt{E-Verify}(vk, q, \Gamma)$: Same as $\mathtt{Verify}(vk, q, \Gamma)$ except that in the final phase, check if

  $$\begin{cases} \mathbf{v} \cdot \mathbf{Q}^{(1)} = \mathbf{Z}^{(1)} \cdot \mathbf{t}^{(1)} \\ \vdots \\ \mathbf{v} \cdot \mathbf{Q}^{(r)} = \mathbf{Z}^{(r)} \cdot \mathbf{t}^{(r)}. \end{cases}$$

  If all of them hold, output 1; else output 0.

The correctness of this protocol can be easily verified. As for its security, we have the following:

Theorem 2 *If $n < m$, then our proposed auditing protocol* $\mathtt{E-TPA-AP}$ *is secure. Concretely, for any PPT $\mathcal{CSP}, \mathcal{TPA}$ and some TPA, their advantages in the corresponding experiments* $\mathrm{Exp}^{\mathcal{CSP}}_{\mathtt{E-TPA-AP}}(\lambda)$, $\mathrm{Exp}^{\mathrm{TPA-Extract}}_{\mathtt{E-TPA-AP}}(\lambda)$ *and* $\mathrm{Exp}^{\mathcal{TPA}}_{\mathtt{E-TPA-AP}}(\lambda)$ *respectively satisfy*

$$\mathrm{Adv}^{\mathcal{CSP}}_{\mathtt{E-TPA-AP}}(\lambda) = \frac{1}{p^r},$$
$$\mathrm{Adv}^{\mathrm{TPA-Extract}}_{\mathtt{E-TPA-AP}}(\lambda) = 1,$$

and

$$\mathrm{Adv}^{\mathcal{TPA}}_{\mathtt{E-TPA-AP}}(\lambda) \le \frac{1}{p^{r(m-n)}}.$$

Proof Since the proof of Theorem 1 can be easily extended to this theorem, we omit it here. $\square$

4 Performance analysis

In this section, we analyze the computational efficiency and other properties of our proposed protocols.

4.1 Computational efficiency

In order to estimate the computational efficiency of our protocol, we do some experiments on a laptop with the configuration of Intel(R) Core(TM) i7-7500 CPU @ 2.70GHz 2.9 GHz and 8 GB RAM. Since all the algorithms mainly consist of the linear operations of vectors, the experiments are run in Matlab with version R2014a.

For the parameter p, we choose it as $p = 2^8 = 256$ so that each element in $\mathbb{F}_p$ can be represented in 1 Byte. For m, n, we set them as different ones since the choosing of them depends on the size of the outsourced file F. Here, we generate five files with sizes of 25K, 230K, 2.2M, 24M, 102M. For the five files, respectively choose the pair of (m, n) as $(1\mathrm{K}, 25)$, $(1\mathrm{K}, 230)$, $(1\mathrm{M}, 2.2)$, $(1\mathrm{M}, 24)$ and $(1\mathrm{M}, 102)$. Then the running times for `Authenticate`, `Audit`, `ProofGen`, and `Verify` are presented in Table 1.

Comparisons with Private Verification. Since the protocol proposed by Zhang et al. in [15] is based on homomorphic MAC scheme and is extremely efficient in terms of computation, we make a comparison with it. Moreover, because the algorithms `Audit`, `ProofGen` for both protocols are same, we only need to compare the efficiencies of `Authenticate` and `Verify`. In addition, note that they also used PRF and PRG to shorten their key. In order to present a fair comparison, we get rid of the actions of PRF and PRG. After running Zhang et al.'s authentication and verification algorithms with respect to the above five data files, we can draw two figures: Fig. 3 and Fig. 4.

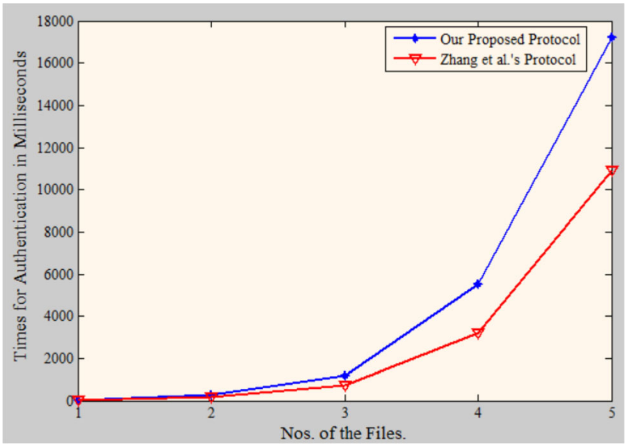


Fig. 3 Authentication Time for Our and Zhang et al.'s Protocols

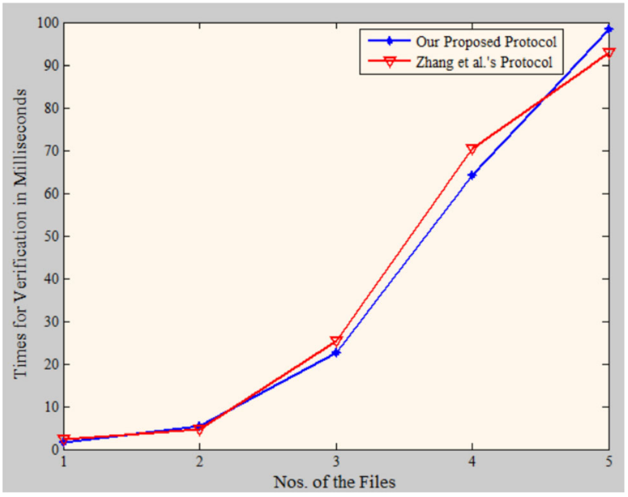


Fig. 4 Verification Time for Our and Zhang et al.'s Protocols

From the simulation results, we know that the authentication time of our protocol is slightly longer than that of Zhang et al. The reason lies in for the same vector, it needs

Table 1 Running Time of Our Proposed Protocol

No.	m (B)	n	`Authenticate` (ms)	`Audit` (ms)	`ProofGen` (ms)	`Verify` (ms)
1	1K	25	26.3	2.3	1.4	1.6
2	1K	230	257.7	3.1	7.6	5.4
3	1M	2.2	1204.8	2.4	30.2	22.7
4	1M	24	5513.4	3.6	73.7	64.2
5	1M	102	19201.3	5.1	104.5	98.3

Table 2 Comparisons of Computational Efficiency with Chen et al.'s Public Auditing Protocol

Data File	Protocol	`Authenticate` (s)	`Audit` (ms)	`ProofGen` (ms)	`Verify` (ms)
No. 4	Our Protocol	5.51	3.6	73.7	64.2
	Chen et al.'s Protocol [28]	2837.6	4.2	27612.4	18203.5
No. 5	Our Protocol	19.2	5.1	104.5	98.3
	Chen et al.'s Protocol [28]	12610.5	6.4	96624.4	49213.5

to compute two tags in our algorithm instead of one tag in Zhang et al.'s algorithm. However, the time of other processes is almost same for both protocols. Hence, our protocol is extremely computational efficient for practical data storage.

Comparisons with Public Verification. Here, in order to show the efficiency advantage of our proposed protocol, we also compare it with the public auditing protocol proposed by Chen et al. in [28]. Concretely, we download the source code and conduct an experiment on the same laptop for our data files with No. 4 and 5 in TABLE 1. Then the concrete running time of these two protocols is presented in TABLE 2. Obviously, our proposed protocol has a considerable advantage in computation efficiency.

4.2 Storage and communication overheads

Here, we only consider the overhead of generated tags for one data file (or its divided vectors) and the sizes of keys transmitted between user and TPA.

In particular, for each (divided) vector $\mathbf{v}$, in our protocol, it needs to generate a two-dimensional authentication tag

$$\mathbf{t} = (t_{\mathbf{v},1}, t_{\mathbf{v},2}) \in \mathbb{F}_p^2,$$

whose size is obviously two times of that appeared in Zhang et al.'s protocol. Then for a data file F consisting of n vectors, the overhead of authentication tags totally equals to $2n\lceil\log_2 p\rceil$ bits. If $p = 2^8$, then it is $2n$ bytes. Of course, the corresponding value of overhead for Zhang et al.'s protocol is just $n\lceil\log_2 p\rceil$ bits (or n bytes when $p = 2^8$). As for the tag overhead of Chen et al.'s public auditing protocol, we know that for each block $\mathbf{v} \in \mathbb{Z}_e^m$, the generated tag is $t = (s, x) \in \mathbb{Z}_e \times \mathbb{Z}_N$. Hence, for the whole data file with n blocks, the tag overhead equals to $n \times (|\mathbb{Z}_e| + |\mathbb{Z}_N|)$.

Assume that the algorithm `KeyGen` is run by user and vk is securely transmitted to TPA from this user. Then the communication overhead naturally equals to the length of vk. For $vk = (\mathbf{Q}, \mathbf{Z}) \in \mathbb{F}_p^{m+n} \times \mathbb{F}_p^2$, it consists of $m + n + 2$ elements in $\mathbb{F}_p$ and hence has length of $(m + n + 2)\lceil\log_2 p\rceil$ bits. Similarly, we can calculate the corresponding communication overhead of public key (i.e. $|vk|$) is $(m + 2n)\lceil\log_2 p\rceil$ for Zhang et al.'s protocol. In addition, from [28], we know the verification key vk' consists of $(m + n)$ group elements, which are coprime with N, and e, N. Thus, the communication cost of the verification key for Chen et al.'s equals to $(m + n + 1) \cdot |\mathbb{Z}_N| + |\mathbb{Z}_e|$. Obviously, our proposed protocol is superior to the other ones in terms of communication costs of verification key.

Meanwhile, the sizes of the signing key sk for all the three protocols can also be computed as $2(m + n)$, $m + 2n$, and $2 \cdot |\mathbb{Z}_N|$, respectively.

In order to give a response to verifier's challenge message, the CSP in our protocol will compute and return the proof $\Gamma = (\mathbf{v}, \mathbf{t}) \in \mathbb{F}_p^{m+n+2}$. Therefore, the communication cost of Γ equals to $(m + n + 2) \cdot |\mathbb{F}_p|$. Similarly, we can know the lengths of returned proofs in [15] and [28] are $(m + n + 1) \cdot |\mathbb{F}_p|$ and $(m + 1) \cdot |\mathbb{Z}_e| + |\mathbb{Z}_N|$, respectively.

However, we emphasize that the main superiority of our proposed protocol lies in that the asymmetry of signing and verification keys, as well as the high computational efficiency. Compared to the homomorphic-MAC-based auditing protocol, our protocol is more suitable to be outsourced to TPA although it slightly sacrifices computational efficiency, storage and communication overheads.

Table 3 Comparisons on Storage and Communication Overheads

Protocols	Tag Overhead	$\lvert sk \rvert$	$\lvert vk \rvert$	Proof Overhead	Suitable for TPA
Our Proposed Protocol	$2n \cdot \lvert\mathbb{F}_p\rvert$	$2(m+n) \cdot \lvert\mathbb{F}_p\rvert$	$(m+n+2) \cdot \lvert\mathbb{F}_p\rvert$	$(m+n+2) \cdot \lvert\mathbb{F}_p\rvert$	Yes
Zhang et al.'s Protocol [15]	$n \cdot \lvert\mathbb{F}_p\rvert$	$(m+2n) \cdot \lvert\mathbb{F}_p\rvert$	$(m+2n) \cdot \lvert\mathbb{F}_p\rvert$	$(m+n+1) \cdot \lvert\mathbb{F}_p\rvert$	No
Chen et al.'s Protocol [28]	$n \cdot (\lvert\mathbb{Z}_e\rvert + \lvert\mathbb{Z}_N\rvert)$	$2 \cdot \lvert\mathbb{Z}_N\rvert$	$(m+n+1) \cdot \lvert\mathbb{Z}_N\rvert + \lvert\mathbb{Z}_e\rvert$	$(m+1) \cdot \lvert\mathbb{Z}_e\rvert + \lvert\mathbb{Z}_N\rvert$	Yes

Finally, our proposed protocol is obviously more computational efficient than Chen et al.'s public auditing protocol.

All the above facts can be summarized as the TABLE 3.

5 Conclusions

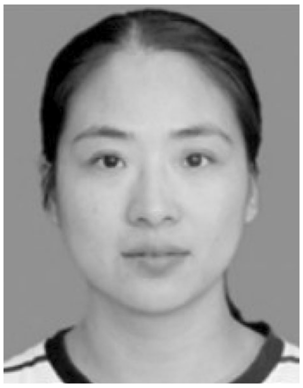
In this paper, we propose an efficient TPA-based auditing protocol. Considering that the current homomorphic-signature-based public auditing protocol are very inefficient in terms of computational and communication overhead, we adopt a technique of similar homomorphic MAC to greatly reduce them. From the experimental analysis, we can easily know that our protocol is almost as efficient as the homomorphic-MAC-based one proposed by Zhang et al. in [15].

Acknowledgements The authors would like to thank anonymous referees for their valuable suggestions and comments. This work is supported in part by National Natural Science Foundation of China (No. 61672416; No. 61872284), and in part by Project of Natural Science Research in Shaanxi (2019JM118).

References

1. Jyoti, A., Shrimali, M.: Dynamic provisioning of resources based on load balancing and service broker policy in cloud computing. Clust. Comput. **23**(1), 377–395 (2020)
2. Kalai Arasan, K., Anandhakumar, P.: A new GLoSM embedded virtual machine model for big data services in cloud storage systems. Clust. Comput. **22**(1), 399–405 (2019)
3. Kumar Bedi, R., Singh, J., Kumar Gupta, S.: Design and implementation of an efficient multi cloud storage approach for resource constrained modile devices. Clust. Comput. **22**, 13143–13157 (2019)
4. Tchernykh, A., Miranda-Lóṕez, V., Babenko, Mikhail G. et al.: Performance evaluation of secret sharing schemes with data recovery in secured and reliable heterogeneous multi-cloud storage. Clust. Comput. **22**(4), 1173–1185 (2019)
5. Ni, J., Yu, Y., Mu, Y., et al.: On the security of an effieicent dynamic auditing protocol in cloud storage. IEEE Trans. Parall. Distr. **25**(10), 2760–2761 (2014)
6. Xue, J., Xu, C., Zhao, J. et al.: Identity-based public auditing for cloud storage systems against malicious auditors via blockchain. In: Science China Information Sciences, vol. 62(3) (2019)
7. Deswarte, Y., Quisquater, J. Saı̈dane, A.: Remote integrity checking. In: Proceedings of IICIS 2003, vol. 140, pp. 1–11 (2003)
8. Gazzoni Filho, D., Barreto, P.: Demonstrating Data Possession and Uncheatable Data Transfer. In: Cryptology ePring Archive, Report 2006/150 (2006)
9. Schwarz, T., Miller, E.: Store, forget, and check: using algebraic signatures to chek remotely administered storage. In: Proceedings of ICDCS 2006.
10. Naor, M., Rothblum, G.: The complexity of online memory checking. Proc. FOCS **2005**, 573–584 (2005)
11. Atenises, G., Burns, R., Curtmola, R. et al.: Provable Data Possession at untrusted Stores. In: Proceedings of CCS, pp. 598–609 (2007)
12. Juels, A., Kaliski, B., Pors.: Proofs of retrievability for large files. In: Proceedings of CCS, pp. 584–597 (2007)
13. Chang, J., Ji, Y., Xu, M., et al.: General transformations from single-genearation to multi-generation for homomorphic message authentication schemes in network coding. Future Gener. Comput. Syst. **91**, 416–425 (2019)
14. Shacham, H., Waters, B.: Compact proofs of retrievability. J. Cryptogr. **26**, 442–483 (2013)
15. Zhang, R., Ma, H., Lu, Y., et al.: Provably secure cloud storage for mobile networks with less computation and smaller overhead. Sci. China Inf. Sci. **60**(12), 122104 (2017)
16. Dan Boneh, D. Freeman, J. Katz, et al.: Signing a Linear Subspace: Signature: Signature Schemes for Network Coding. In: PKC, vol. 5443, pp. 68-87. Springer, Berlin, Germany (2009)
17. Chang, J., Ma, H., Zhang, A., Xu, M., Xue, R.: RKA security of identity-based homomorphic signature scheme. IEEE Access **7**, 50858–50868 (2019)
18. Chang, J., Wang, H., Wang, F., et al.: RKA security for identity-based signature scheme. IEEE Access **8**, 17833–17841 (2020)
19. Lin, Q., Yan, H., Huang, Z., et al.: An ID-based linearly homomorphic signature scheme and its application in blockchain. IEEE ACCESS **6**, 20632–20639 (2018)
20. Ji, Y., Shao, B., Chang, J. et al.: Privacy-Preserving Certificateless Provable Data Possession Scheme for Big Data Storage on Cloud, Revisited. In: Applied Mathematics and Computation, vol. 386, 125478 (2020)
21. Zhang, J., Yang, Y., Chen, Y., et al.: A general framework to design secure cloud storage protocol using homomorphic encryption scheme. Comput. Netw. **129**, 37–50 (2017)
22. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC'2009, pp. 169–178
23. Erway, C., Küpçü, A., Papamanthou, C., Tamassia, R. : "Dynamic Provable Data Possession," in CCS'2009, pp. 213–222 (2009)
24. Wang, Q., Wang, C., Ren, K., et al.: Enabling public audititability and data dynamics for storage security in cloud computing. IEEE Trans. Parallel Distrib. Syst. **22**(5), 847–859 (2011)
25. Zhu, Y., Ahn, G., Hu, H., et al.: Dynamic audit services for outsourced storage in clouds. IEEE Trans. Serv. Comput. **6**(2), 227–238 (2013)
26. Chen, Y., Liu, J.: Dynamic-hash-table based public auditing for secure cloud storage. IEEE Trans. Serv. Comput. **10**(5), 701–714 (2017)
27. Shen, J., Shen, J., Chen, X. et al.: An efficient public auditing protocol with novel dynamic sturcture for cloud data. In: IEEE Transactions on Information Forensics and Security, vol. 12(10), (2017)
28. Chen, F., Xiang, T., Yang, Y., et al.: Secure Cloud Storage Meets with Secure Network Coding. IEEE Trans. Comput. **65**(6), 1936–1948 (2016)
29. Chang, J., Shao, B., Ji, Y., et al.: Secure network coding from secure proof of retrievability. SCI. CHINA Inf. Sci. (2020). https://doi.org/10.1007/s11432-020-2997-0
30. Wu, X., Xu, Y., Yuen, C., Xiang, L.: A tag encoding scheme against pollution attack to linear network coding. IEEE Trans. Parallel Distrib. Syst. **25**(1), 33–42 (2014)

Bilin Shao received the B. S. Degree from the School of Management, Xi' an University of Architecture and Technology (XAUAT), Shaanxi, China. He is currently a professor in XAUAT. His research includes information security, information management technology, cloud computing security, and VANETS security. He is a member of the China Computer Federation.



Yanyan Ji received the B. S. and M. S. degrees in school of electrical engineering and automation from Henan Polytechnic University in 2005, and school of information and communication engineering from North University of China in 2018. Now, she is pursuing the Ph. D. degree with the School of Management, Xi'an University of Architecture and Technology (XAUAT), Shaanxi, China. Her research interests mainly focus on communication and information security