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/TBDATA.2017.2701347, IEEE Transactions on Big Data

NPP: A New Privacy-Aware Public Auditing Scheme for Cloud Data Sharing with Group Users

Anmin Fu, *Member, IEEE,* Shui Yu, *Senior Member, IEEE,* Yuqing Zhang, Huaqun Wang, and Chanying Huang

***Abstract*—Today, cloud storage becomes one of the critical services, because users can easily modify and share data with others in cloud. However, the integrity of shared cloud data is vulnerable to inevitable hardware faults, software failures or human errors. To ensure the integrity of the shared data, some schemes have been designed to allow public verifiers (i.e., third party auditors) to efficiently audit data integrity without retrieving the entire users' data from cloud. Unfortunately, public auditing on the integrity of shared data may reveal data owners' sensitive information to the third party auditor. In this paper, we propose a new privacy-aware public auditing mechanism for shared cloud data by constructing a homomorphic verifiable group signature. Unlike the existing solutions, our scheme requires at least *t* group managers to recover a trace key cooperatively, which eliminates the abuse of single-authority power and provides non-frameability. Moreover, our scheme ensures that group users can trace data changes through designated binary tree; and can recover the latest correct data block when the current data block is damaged. In addition, the formal security analysis and experimental results indicate that our scheme is provably secure and efficient.**

***Index Terms*—Data Integrity; Homomorphic Verifiable; Non-frameability; Provable Security.**

I. Introduction

DUE to the increasing number of applications of shared data, such as iCloud, Google Docs, and so on, users can upload their data to a cloud and share it with other peers as a group. Unfortunately, since cloud servers are vulnerable to inevitable hardware faults, software failures or human errors, data stored in the cloud may be spoiled or lost [1]. In the worst cases, a cloud owner may even conceal data error accidents in order to preserve its reputation or avoid profit losses [2],[3]. In addition, users who lose direct control over their data are not sure whether their cloud-stored data is intact or not. Therefore, integrity verification for the shared data in the cloud is an important, yet timely issue for a large number of cloud users.

To ensure the integrity of data stored in cloud servers, a number of mechanisms based on various techniques have been proposed. In particular, in order to reduce the burden on users, a trusted third-party auditor (TPA) is engaged to conduct the verification, which is called public auditing [4]. However, the TPA may have unnecessary access to private information during the auditing process [5]. Therefore, researchers proposed some new schemes to protect privacy, including data privacy [6], and identity privacy [7]-[9]. To be specific, the TPA cannot learn each block that is signed by a particular user in the group by constructing homomorphic authenticable ring signatures [7] or computing tags based on common group private key [8]. However, since both methods concern about unconditional privacy, the real identity of the signer can no longer be traced. A later development is the homomorphic authenticable group signature scheme based on group signatures [9], which is designed to protect privacy. On one hand, the identity of each signer is anonymous; and on the other hand, the group manager can trace a signer's real identity after a dispute. Unfortunately, in all existing public auditing schemes, the tracing process is accomplished by a single entity. As a result, that entity has the privilege of tracing, which may lead to abuse of single-authority power. Therefore, an innocent user may be framed or a malicious user may be harbored.

Meanwhile, to support data dynamics, the data structure based on index hash table [7]-[11] or Merkle Hash Tree (MHT) has been utilized [12], [15]. However, this kind of data structure merely records the newest data block with the corresponding signature, which prevents users from tracing the changes of the data blocks. When the current data has been corrupted, users cannot recover the old data from the records. Therefore, the problem of data traceability and recoverability also should be considered.

Moreover, a necessary authentication process is missing between the auditor and the cloud in most existing public auditing schemes, hence anyone can challenge the cloud for the auditing proofs. This problem will trigger network congestion and unnecessary waste of cloud resources. Although Liu *et al*. [12] designed an authorized public auditing scheme to solve the problem, it is only suitable for a single client, and cannot be applied to group-shared data. Since the malicious or pretended auditors/users might constantly request cloud access for the auditing proof by utilizing TPA, unauthorized auditing is another important issue that should be addressed in integrity verification for shared cloud data.

At present, all the existing public auditing schemes only consider a single group manager when applied to shared data with group users. However, in real-world applications, there might be multiple managers in a group.For instance, the shared data of a project team is created by multiple managers together,

This work is supported by National Science Foundation of China (61572255), Natural Science Foundation of Jiangsu Province, China (BK20141404 and BK20150787), and Six Talent Peaks Project of Jiangsu Province, China (XYDXXJS-032).

Anmin Fu and Chanying Huang are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, PR China.

Shui Yu is with the School of Information Technology, Deakin University, Melbourne, Australia.

Yuqing Zhang is with the National Computer Network Intrusion Protection Center, UCAS, Beijing, PR China.

Huaqun Wang is with the Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, School of Computer Science, Nangjing University of Posts and Telecommunications, Nanjing, PR China.

whats more, any of them can maintain the shared data. Another important practical problem is that the group users should be able to dynamically enroll and revoke the group, which will be managed by the group managers. And significantly, when tracing the real identity of the signer, a specified number of managers can work together, which ensures the fairness of the tracing process.

In this paper, we propose a new privacy-aware public auditing mechanism, called *NPP*, for the shared cloud data with multiple group managers. Our contributions can be summarized as follows.

1) We establish a model for data (in a group) shared with multiple group managers, and propose a new privacy-preservation public auditing scheme for multiple group managers in shared cloud storage. Our proposed scheme can not only provide multi-levels privacy-preservation abilities (including identity privacy, traceability and non-frameability), but also can well support group user revocation.

2) We design a data structure based on a binary tree for clouds to record all the changes of data blocks. Group users can trace the data changes through the binary tree and recover the latest correct data block when the current data block is damaged.

3) We utilize an authorized authenticate process to verify TPA's challenge messages. Therefore, only the TPA who has been authorized by the group users can pass the authentication and then challenge the cloud, which protects clouds from malicious challenges.

4) Our formal security analysis and experimental results show that *NPP* is provably secure and efficient.

The rest of this paper is organized as follows. Section II presents a review of related work on public auditing schemes in cloud storage. Then we introduce our system model, threat model and design objectives in Section III. Section IV briefly introduces the cryptographic knowledge applied in our scheme. In Section V, we describe the proposed public auditing scheme *NPP* in detail. Section VI analyzes its security and Section VII evaluates its performance. Finally, this paper is concluded in Section VIII.

II. RELATED WORK

Ateniese *et al.* [16] firstly proposed the Provable Data Possession (PDP) model, utilizing homomorphic verifiable tags, and the process of data integrity checking was a kind of "challenge-response" protocol. In order to support data retrievability, Juels *et al.* [17] proposed the Proofs of Retrievability (POR) model. Many extended schemes based on PDP or POR have been proposed to solve different problems in public auditing [7]-[14], [18]-[23].

Considering the application of cloud data shared by group users, Wang *et al.* [7] proposed a privacy-preserving public auditing scheme, called *Oruta*, for shared data in the cloud. Their scheme was based on a homomorphic authenticable ring signature, which allows a public auditor to audit the shared data without retrieving all data from the cloud. However, the auditing overhead linearly increases with the number of group users, hence it is not suitable for large groups in the cloud. To support large groups, Wang *et al.* [8] proposed a new auditing scheme, called *Knox*. The auditing overhead is independent of the number of group users, hence *Knox* can support shared data with large groups. Moreover, any group manager can reveal the identity of the signer. Unfortunately, the scheme cannot support user revocation.

Many schemes have been proposed in order to deal with this problem. In [9]-[11], homomorphic authentications based on proxy re-signature were constructed. With the cooperation of cloud and revoked users, these schemes converted the signatures of the revoked users into those of the existing users. As the cloud has powerful computation ability, this method has no effect on the existing users. The problem is that it cannot resist collusion attacks. If a revoked user colludes with the cloud, the private keys of the existing users can be obtained by the cloud. Therefore, the cloud can tamper with the shared data stored in it arbitrarily. Besides, Yu *et al.* [15] pointed out that the scheme in [11] is vulnerable to replace attacks.

Recently, to solve the problem of collusion attacks, Yuan *et al.* designed polynomial-based authentication tags, allowing aggregation of tags for different data blocks [19]. Their scheme allows secure delegation of user revocation operations to the cloud, permitting the cloud itself to conduct revocation without the participation of revoked users. Unfortunately, their scheme is also vulnerable to resist collusion attacks. If a revoked user colludes with the cloud, the cloud server can update the data as many times as the revoked user requests until it finally returns a legal data [22], [24]. Another attempt to solve the issue is the combination of vector commitments and group signatures with verifier-local revocation [22]. However, the computation cost of user revocation grows with the number of revoked users.

In addition, to eliminate threats of unauthorized audit challenges from malicious or pretended third-party auditors, Liu *et al.* [12] proposed an authorized auditing scheme by adding an additional authentication process between the cloud and the TPA. Additionally, to support fine-grained update requests, the authorized scheme employed BLS signatures and MHT. However, the scheme can only be applied to a single client.

III. PROBLEM STATEMENT

In this section, we describe the system model and the threat model of this paper, and give the design objectives of our public auditing scheme.

A. System Model

As shown in Fig. 1, the system model contains four entities: cloud, TPA, trusted private key generator (PKG), and group users. The cloud has powerful storage space and computing capacity, and provides services (e.g., data storage, data sharing, etc.) for group users. The TPA can verify the integrity of the shared data on behalf of the group users. The PKG generates the system public parameters and group key pair for group users. The group users include two types of users: GMs (Group Managers) and ordinary members.

Unlike existing system models, the GMs contain multiple members who create the shared data together and share them with the ordinary members through the cloud. Therefore, the

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/TBDATA.2017.2701347, IEEE Transactions on Big Data

GMs act as the common owners of the original data, and their identities are equal. Meanwhile, any of the GMs can add new members or revoke members from the group. In addition, either a GM or an ordinary member can access, download, and modify the shared data in the cloud. Note that multiple managers in a group is very common in practice. For instance, the shared data of a project team is created by multiple managers together. Later, any of the GMs can maintain the shared data and manage the group users. When tracing the real identity of the signer, a given number of managers can cooperate to trace the real identity, which ensures the fairness of the tracing process.

When a group user wants to check the integrity of the shared data, she/he first submits an auditing request message to the TPA. After receiving the request, the TPA challenges the cloud for an auditing proof. Once the cloud receives the auditing challenge, it firstly authenticates the TPA. If valid, the cloud will return the auditing proof to the TPA. Otherwise the cloud will refuse the request. Finally, the TPA verifies the validity of the proof and sends an auditing response to the group user.

B. Threat Model

1) Integrity Threat. There are two kinds of threats related to shared data integrity. One is that external attackers might corrupt the shared data in the cloud, so that group users can no longer access the correct data. The other is that the cloud may corrupt or delete the shared data due to the hardware/software faults or human errors. What's worse, the cloud may conceal the fact of data damage from users in order to maintain self-interest and service reputation.

2) Privacy Threat. As a trusted and inquisitive verifier, a TPA might obtain some privacy information from the verification metadata during the auditing process. For instance, the TPA might analyze which data block has been modified most or which user has modified the data most, and finally conclude which particular data block or which group user is of a higher value than the others. Then the TPA might directly obtain the data or the identity of the group user from the signatures of the data blocks.

3) Challenge Threat. Because the auditing challenge message is very simple and has not been authorized, any other entity can utilize the TPA to challenge the cloud for auditing proofs. In this case, a malicious entity might launch denial of service attacks on the cloud by sending massive challenge messages continuously, which will lead to network congestion and unnecessary waste of the clouds resources.

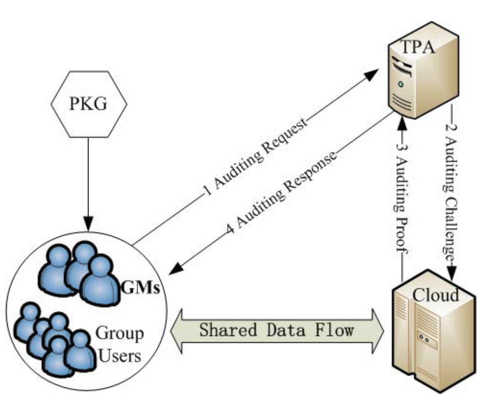


Fig. 1. The system model of NPP.

TABLE I
NOTIONS

Notions	Description
mpk	shared group public key
msk_l	the secret key of GM_l
{*spk,ssk*}	public/private key pair
usk_i	user signing key
upk_i	user membership key
rvk_i	user revocation key
$(V_{j,1}, V_{j,2}, \theta_j)$	the signature of the block m_j
AUTH	authorization
t	timestamp

C. Design Objectives

To achieve integrity checking of the shared data in the cloud, *NPP* is expected to the following design objectives: **1) Public auditing:** Besides the group users, the TPA can also correctly check the integrity of the shared data in the cloud without retrieving entire users' data from cloud. **2) Authorized auditing:** Only the TPA that has been authorized by the group users can challenge the cloud. **3) Identity privacy:** During the process of auditing, the TPA cannot learn the identity of the group user from the signatures of the data blocks. **4) Traceability:** Under certain conditions, the group managers can reveal the signer's identity from the signatures and decide which group user has modified the data block. **5) Non-frameability:** Group managers can guarantee the fairness of the tracing process, i.e., innocent group user won't be framed and the misbehaved user won't be harbored by the group managers. **6) Support data traceability and recoverability:** Group users can easily trace the data changes and recover the latest correct data once current data is damaged. **7) Support group dynamics:** Group dynamics include two aspects. One is that GMs can easily join or leave the group, the other is that new users can be easily added into the group and misbehaved users can be efficiently excluded from the group.

IV. PRELIMINARIES

In this section, we briefly introduce the cryptographic knowledge applied in *NPP*. The main notations used in this paper are described in Table I.

A. Homomorphic Verifiable Tags

Homomorphic Verifiable Tags [16] (HVTs) acting as the verification metadata of file blocks have been widely used in integrity checking for data stored in the cloud.

Definition 1 (Homomorphic verifiable signature). If an HVT based on signatures can satisfy the following two properties simultaneously, then the signature scheme is a homomorphic verifiable signature scheme [7], [11].

Supposing (*pk,sk*) are the public/private key pair of the signer, σ_1 and σ_2 denote the tags of data block $m_1, m_2 \in Z_q$, respectively.

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/TBDATA.2017.2701347, IEEE Transactions on Big Data

1) Blockless verification: A verifier can judge the correctness of all data through the linear combination of the data without retrieving it from the cloud. Specifically, given σ_1, σ_2, two random numbers $y_1, y_2 \in Z_p$ and data block $m = y_1 m_1 + y_2 m_2$, a verifier can check the correctness of m without knowing m_1, m_2.

2) Non-malleability: Any entity without the secret key cannot generate a new and valid tag through combining the known tags. Specifically, given σ_1, σ_2, two random numbers $y_1, y_2 \in Z_p$ and data block $m = y_1 m_1 + y_2 m_2$, an entity who has no *sk* cannot generate the valid tag σ for m by combining σ_1 and σ_2.

B. Discrete Logarithm (DL) Problem

Definition 2 (DL Problem). Let $a \in Z_p^*$, given $g, g^a \in G_1$ as input, output a. The advantage of probabilistic polynomial time algorithm $\mathring{A}$ in solving the DL problem in G_1 is defined as $AdvDL_{\mathring{A}} = Pr[\mathring{A}(g, g^a) = a : a \xleftarrow{R} Z_p^*]$, where the probability is over the choice of a, and the coin tosses of $\mathring{A}$. In this case, for any probabilistic polynomial time algorithm $\mathring{A}$, the advantage of solving the DL problem in G_1 is negligible.

V. The NPP Scheme

A. Overview

We assume that there is *S* group managers $GM_l (1 < l \leq S)$, and *d* users $U_i (1 \leq i \leq d)$ in *NPP*. The shared data M is divided into w data blocks, i.e. $M = \{m_1, m_2, \cdots, m_w\}$. In order to support dynamic operations on the shared data, we index each data block by leveraging index hash table [9]. Specifically, *NPP* consists of eight algorithms: {*Setup, Enroll, Revoke, Sign, Authorize, ProofGen, ProofVerify, Open*}.

In *Setup* phase, the PKG sets parameters for the entire system, distributes the group key pair $\{mpk_l, msk_l\}$ and a shared public/private key pair {*spk,ssk*} used to authorize each GM_l, and initializes the membership information Ω . Then, any GM generates a user signing key usk_i, a (public) user membership key upk_i, and a user revocation key rvk_i for U_i. GM also shares the authorization key pair {*spk, ssk*} with U_i in the *Enroll* procedure. Once a group user is revoked, GM invokes the *Revoke* algorithm to update Ω. The group user can compute the signatures of the shared data block from the issued keys in the *Sign* process. With the *Authorize* algorithm, the group authorizes TPA to generate authorized auditing challenges, and then the valid TPA can check the integrity of the shared data on behalf of the group user. Once the cloud receives a challenge from TPA, the cloud verifies whether the challenge has been authorized and decides whether to generate the audit proof via *ProofGen*. TPA checks the correctness of the proof via *ProofVerify*. Finally, in the *Open* process, at least t GMs work together to trace the real identity of the signer.

B. Support Data Traceability and Recoverability

Since the identity of each data block can be described by the index hash table, i.e., $id_j = \{v_j, r_j\}$, where v_j is denoted as the virtual index of block m_j, and r_j is a random number generated by a collision-resistant hash function, every group user can easily perform dynamic operations on the shared data, the details of which can be found in [7]. However, if the data block has been changed maliciously, the group user cannot trace the changes and recover the right data.

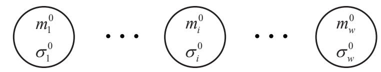


Fig. 2. The original records.

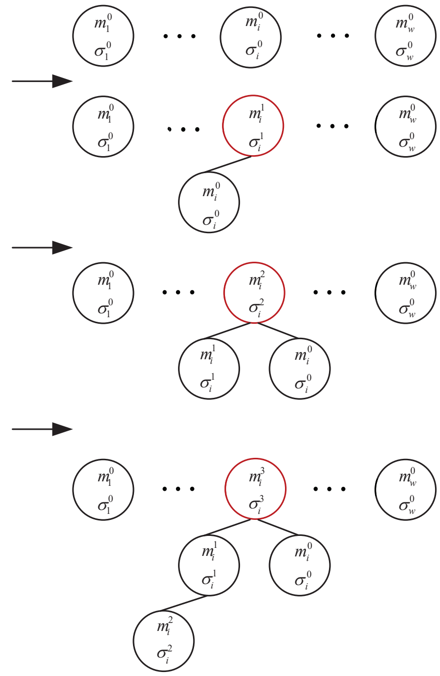


Fig. 3. The records when the *i*th block has been updated three times.

To support data tracing and recovery, we have designed an additional data structure based on binary tree for the cloud server to record every change of data block. Through the records, group users can easily trace data changes. When the damaged has been found, group users can recover the right data by the records. As the group users can verify the older blocks one by one until discover the latest correct block. As shown in Fig. 2, original data blocks $\{m_1, m_2, \cdots, m_w\}$ with the corresponding signatures $\{\sigma_1, \sigma_2, \cdots, \sigma_w\}$ are stored as the roots of w binary trees respectively. $\{m_i^j, \sigma_i^j\}(1 \leq i \leq w)$ denotes the *i*th block has been modified *j* times, hence $\{m_i^0, \sigma_i^0\}$ means the data block is the original one. We will use some examples to show different records when group users perform dynamic operations on the shared data later. Fig. 3 and 4 describe update operation and insert operation respectively. In addition, when group users want to delete a block, the cloud server still keeps the records related to this block, without any other additional operations. Whats more, the cloud server does not need to know which block has been deleted.

As shown in Fig. 3, the *i*th block has been updated for three times, and the latest one is always the root of the binary tree; the old ones are the nodes of the binary tree. If we define the depth of the binary tree as N, then the number of nodes

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/TBDATA.2017.2701347, IEEE Transactions on Big Data

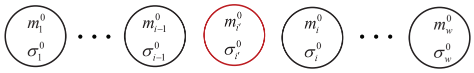


Fig. 4. The record when a block has been inserted.

belongs to range $[2^{N-1}, 2^N - 1]$, and the number of times for the cloud to record the updates belongs to range $[2^{N-1} - 1, 2^N - 2]$.

Once the current signature σ_i^3 has been damaged, group users can trace the changes $\{(m_i^2, \sigma_i^2), (m_i^1, \sigma_i^1), (m_i^0, \sigma_i^0)\}$ by implementing the postorder traversal to the final binary tree. As the damaged signature cannot pass the verification, the group users are able to verify the signature σ_i^2, if it can pass the verification, then the latest right block has been found. Otherwise, the group users continue verifying the signatures one by one according to the order of traversal tree with the help of TPA until the latest right block is discovered. The verification algorithm can be found in the next section.

As shown in Fig. 4, when group users want to insert a block, for example a new block m_i' is inserted between the block m_{i-1} and the block m_i, and $id_i' = \{v_i', r_i'\} = \{\lfloor (v_{i-1} + v_i)/2 \rfloor, r_i'\}$, the cloud server will create a new root (i.e., $\{m_{i'}^0, \sigma_{i'}^0\}$) for this new block.

C. Construction of NPP

In this section, we describe the details of the eight algorithms included. To protect data privacy, the data can be encrypted by the means of symmetrical encryption technology and attribute-based encryption technology before shared data is outsourced to the cloud [25]; however, this is outside the scope of our paper.

1) Setup: With the input security parameters $\varepsilon > 1$, $k, l_p \in N$, PKG randomly chooses parameters $\lambda_1, \lambda_2, \gamma_1$ and γ_2 such that $\lambda_1 > \varepsilon(\lambda_2 + k) + 2, \lambda_2 > 4l_p, \gamma_1 > \varepsilon(\gamma_2 + k) + 2$ and $\gamma_2 > \lambda_1 + 2$, two multiplicative cyclic groups G_1, G_2 with the same order q, and g_0 is the generator of G_1. Then PKG chooses a bilinear map $e : G_1 \times G_1 \longrightarrow G_2$ and two one-way hash functions: $H_1\{0,1\}^* \longrightarrow Z_q, H_2\{0,1\}^* \longrightarrow G_1$, and defines two intervals: $A = [2^{\lambda_1} - 2^{\lambda_2}, 2^{\lambda_1} + 2^{\lambda_2}]$, $B = [2^{\gamma_1} - 2^{\gamma_2}, 2^{\gamma_1} + 2^{\gamma_2}]$. The parameters above are all public.

Then, PKG computes the shared group public key $mpk = (n, a, a_0, Y, g_0, g, h, g_1, g_2, \eta_1, \eta_2)$ and each GM_l's secret key $msk_l = (p', q', X_l)$ as follows:

- Select l_p-bit primes p', q' such that $P = 2p' + 1$, and $Q = 2q' + 1$. Set the modulus $n = PQ$ (Note that all the following arithmetic operations are modulo n unless specified otherwise).
- Choose random elements $a, a_0, g, h, g_1, g_2, \eta_1, \eta_2 \in_R QR(n)$ (of order q), where $QR(n)$ denotes the set of quadratic residues of group Z_n^* .
- Choose a random secret $X \in_R Z_q^*$, and set $Y = g^X$.
- Choose a *t-1* degree polynomial $f(x) = b_0 + b_1 x + \cdots + b_{t-1} x^{t-1}$ with $b_0 = X, b_1, \cdots, b_{t-1} \in Z_q$, compute $X_l = f(l)(l = 1, 2, \cdots, S$ and $2t - 1 \geq S)$, i.e. X is divided into S pieces X_l [26].
- Initialize the membership information $\Omega = (c, u)$, where c is initialized to g_1, and u is initialized to 1.

Next, PKG chooses a public/private key pair $\{spk, ssk\}$ used for authorization only.

Finally, PKG sends $\{mpk, msk_l\}$ along with $\{spk, ssk\}$ to GM_l securely.

2) Enroll: The usk_i, rvk_i, and upk_i of a new member U_i are generated as follows:

- U_i generates a secret exponent $\tilde{x_i} \in_R [0, 2^{\lambda_2}]$, a random integer $\tilde{r_i} \in_R [0, n]$, computes $C_1 = g^{\tilde{x_i}} h^{\tilde{r_i}}$ and broadcasts C_1 to any GM.
- GM who has received C_1 checks whether $C_1 \in QR(n)$. If this is the case, the GM chooses $\alpha_i, \beta_i \in_R [0, 2^{\lambda_2}]$ and sends $\{\alpha_i, \beta_i\}$ to U_i.
- U_i computes $x_i = 2^{\lambda_1} + (\alpha_i \tilde{x_i} + \beta_i \ mod \ 2^{\lambda_2})$, $C_2 = a^{x_i}$ and broadcasts C_2 to the GM.
- GM checks whether $C_2 \in QR(n)$. If this is the case, GM chooses $e_i, \pi \in_R B$, computes $A_i = (C_2 a_0)^{1/e_i} = (a^{x_i} a_0)^{1/e_i}$, $\rho = g_0{}^{\pi}(\rho$ is public) and sends $\{A_i, e_i, \pi\}$ along with $\{spk, ssk\}$ to U_i.
- U_i checks $a^{x_i} a_0 \stackrel{?}{=} A_i^{e_i}$. If the equation holds, U_i sets $usk_i = (x_i, \pi), rvk_i = e_i, upk_i = A_i$ and shares the key pair $\{spk, ssk\}$ with all the GMs.
- GM maintains a users-list, which contains all related keys and the valid time of the group users. Different users may be assigned with different valid times. Finally, GM adds U_i, U_i's related keys and valid time to the list.

3) Revoke: Supposing user $U_k (1 \leq k \leq d)$ is to be revoked, the revocation key is rvk_k and the current membership information is $\Omega = (c, u)$, then any GM should update $c = c^{rvk_k}, u = u \cdot rvk_k$.

Suppose there are revoked users $\{U_e, \cdots, U_k\}(1 \leq e < k \leq d)$, the latest $c = c^{\prod_{i=e}^{k} rvk_i}, u = \prod_{i=e}^{k} rvk_i$.

Then PKG distributes a new key pair $\{spk', ssk'\}$ to all GMs, and the GM shares it with the existing group users. Meanwhile, the GM updates the revoked users' valid time as a negative value in the users-list.

4) Sign: Group user U_i computes the signature $\sigma_j = (V_{j,1}, V_{j,2}, \theta_j)$ [27] for block $m_j \in Z_q (1 \leq j \leq w$, id_j is the identifier) as follows:

i. Compute $V_{j,1}$.

- Randomly choose $r_j \in_R \{0,1\}^{2l_p}$ and compute $T_{j,1} = Y^{r_j} A_j, T_{j,2} = g^{r_j}, T_{j,3} = g^{rvk_i \cdot h^{r_j}}$.
- Randomly choose $r_{j,1} \in_R \pm \{0,1\}^{\varepsilon(\gamma_2 + k)}$, $r_{j,2} \in_R \pm \{0,1\}^{\varepsilon(\lambda_2 + k)}$, $r_{j,3} \in_R \pm \{0,1\}^{\varepsilon(\gamma_1 + 2l_p + k + 1)}$, $r_{j,4} \in_R \pm \{0,1\}^{\varepsilon(2l_p + k)}$ and then compute $d_{j,1} = T_{j,1}^{r_{j,1}} / (a^{r_{j,2}} \cdot Y^{r_{j,3}})$, $d_{j,2} = T_{j,2}^{r_{j,1}} / g^{r_{j,3}}$, $d_{j,3} = g^{r_{j,4}}$, $d_{j,4} = g^{r_{j,1}} \cdot h^{r_{j,4}}$.
- Compute $v_{j,1} = \eta_1^{m_j} H_1(g \| h \| Y \| a_0 \| a \| T_{j,1} \| T_{j,2} \| T_{j,3} \| d_{j,1} \| d_{j,2} \| d_{j,3} \| d_{j,4})$.
- Compute $s_{j,1} = r_{j,1} - v_{j,1}(rvk_i - 2^{\gamma_1})$, $s_{j,2} = r_{j,2} - v_{j,1}(x_i - 2^{\lambda_1})$, $s_{j,3} = r_{j,3} - v_{j,1} \cdot rvk_i \cdot r_j$, $s_{j,4} = r_{j,4} - v_{j,1} \cdot r_j$.
- Output $V_{j,1} = (v_{j,1}, s_{j,1}, s_{j,2}, s_{j,3}, s_{j,4}, T_{j,1}, T_{j,2}, T_{j,3})$.

ii. Compute $V_{j,2}$.

- Find $f, b \in Z$ such that $f \cdot u + b \cdot rvk_i = 1$, and then set $d = g_1^{-b}$ (because U_i has not been revoked, rvk_i is not included in $u = \prod_{i=1}^{k} rvk_i$ and $gcd(rvk_i, u) = 1$).
- Compute $T_{j,4} = d \cdot g_2^{r_j}$.

- Randomly choose $r_{j,5} \in_R \pm \{0,1\}^{\varepsilon(\gamma_2+k)}$, $r_{j,6} \in_R \pm \{0,1\}^{\varepsilon(\lambda_2+k)}$, $r_{j,7} \in_R \pm \{0,1\}^{\varepsilon(\gamma_1+2l_p+k+1)}$, $r_{j,8} \in_R \pm \{0,1\}^{\varepsilon(2l_p+k)}$ and compute $d_{j,5} = T_{j,4}^{r_{j,5}}/(c^{r_{j,6}} \cdot g_2^{r_{j,7}})$, $d_{j,6} = g^{r_{j,5}} \cdot h^{r_{j,8}}$.
- Compute $v_{j,2} = \eta_2^{m_j} H_1(g\|h\|g_1\|g_2\|c\|T_{j,3}\|T_{j,4}\|d_{j,5} \|d_{j,6})$.
- Compute $s_{j,5} = r_{j,5} - v_{j,2}(rvk_i - 2^{\gamma_1})$, $s_{j,6} = r_{j,6} - v_{j,2}(f - 2^{\lambda_1})$, $s_{j,7} = r_{j,7} - v_{j,2} \cdot rvk_i \cdot r_j$, $s_{j,8} = r_{j,8} - v_{j,2} \cdot r_j$.
- Output $V_{j,2} = (v_{j,2}, s_{j,5}, s_{j,6}, s_{j,7}, s_{j,8}, T_{j,3}, T_{j,4})$.

iii. Compute tag $\theta_j = [H_2(id_j) g_0^{m_j}]^\pi$.

iv. Output the signature $\sigma_j = (V_{j,1}, V_{j,2}, \theta_j)$.

5) Authorize: Any group member can authorize the TPA on behalf of the group to challenge the cloud through the shared key pair {*spk, ssk*} as follows:

- The group member asks the ID of the TPA (for security, the ID is used for authorization only). Then the TPA returns its ID encrypted with the public key *spk*.
- The group member decrypts it with *ssk* to get ID, computes $sig_{AUTH} = Sig_{ssk}(AUTH\|t\|ID)$ ($AUTH$ means authorization and t is the timestamp), and sends sig_{AUTH} as the auditing authorization message to the TPA. Then the TPA can challenge the cloud on behalf of the group users.

Note that after message $\{AUTH, t, sig_{AUTH}\}$ is stored in the cloud along with the signatures of the data blocks, it will be deleted from local storage.

6) ProofGen: In this phase, the TPA first sends a challenge message to the cloud, and then the cloud generates a auditing proof message if the TPA is authorized.

i. The TPA challenges the cloud as follows:

- Randomly choose a subset Γ from the set $[1, w]$, where Γ contains D elements, i.e. $|\Gamma| = D$.
- Generate random numbers $y_j \in Z_q, j \in \Gamma$.
- Send an auditing challenge message $\{sig_{AUTH}, \{ID\}_{PK_{cloud}}, \{(j, y_j)\}_{j\in\Gamma}\}$ to the cloud. Because the PK_{cloud} is the public key of the cloud, the cloud can decrypt $\{ID\}_{PK_{cloud}}$ with the corresponding private key SK_{cloud} to get ID.

ii. The cloud checks whether the TPA has been authorized as follows:

- Compute ID by decrypting $\{ID\}_{PK_{cloud}}$ with its private key SK_{cloud}.
- Decrypt sig_{AUTH} with the group's public key spk to get $ID, AUTH$ and *t*. If $ID = ID'$, the computed $AUTH$ is equal to the $AUTH$ stored in the cloud and t is valid, the cloud will generate the auditing proof. Otherwise, the cloud will refuse to generate the proof.

iii. The cloud generates the auditing proof message to the TPA as follows:

- Compute $\lambda = \sum_{i\in\Gamma} y_j m_j \in Z_q$ and aggregate the selected tags as $\Theta = \prod_{j\in\Gamma} \theta_j^{y_j} \in G_1$.
- Output $\Phi_j = \{V_{j,1}, V_{j,2}\}_{j\in\Gamma}$ based on the selected blocks, where $V_{j,1} = (v_{j,1}, s_{j,1}, s_{j,2}, s_{j,3}, s_{j,4}, T_{j,1}, T_{j,2}, T_{j,3})$, $V_{j,2} = (v_{j,2}, s_{j,5}, s_{j,6}, s_{j,7}, s_{j,8}, T_{j,3}, T_{j,4})$.
- Send the auditing proof $\{\{id_j\}_{j\in\Gamma}, \{\Phi_j\}_{j\in\Gamma}, \lambda, \Theta\}$ to the TPA.

7) ProofVerify: The TPA checks the correctness of the proof as follows.

- Compute $d'_{j,1} \sim d'_{j,6}$ as follows:

$$d'_{j,1} = (a_0^{v_{j,1}} \cdot T_{j,1}^{s_{j,1}-v_{j,1}\cdot 2^{\gamma_1}})/(a^{s_{j,2}-v_{j,1}\cdot 2^{\lambda_1}} \cdot Y^{s_{j,3}}) \quad (1)$$

$$d'_{j,2} = T_{j,2}^{s_{j,1}-v_{j,1}\cdot 2^{\gamma_1}}/g^{s_{j,3}} \quad (2)$$

$$d'_{j,3} = T_{j,2}^{v_{j,1}} \cdot g^{s_{j,4}} \quad (3)$$

$$d'_{j,4} = T_{j,3}^{v_{j,1}} \cdot g^{s_{j,1}-v_{j,1}\cdot 2^{\gamma_1}} \cdot h^{s_{j,4}} \quad (4)$$

$$d'_{j,5} = ((g_1^{-1})^{v_{j,2}} \cdot T_{j,4}^{s_{j,5}-v_{j,2}\cdot 2^{\gamma_2}})/(c^{s_{j,6}-v_{j,2}\cdot 2^{\lambda_1}} \cdot g_2^{s_{j,7}}) \quad (5)$$

$$d'_{j,6} = T_{j,3}^{v_{j,2}} \cdot g^{s_{j,5}-v_{j,2}\cdot 2^{\gamma_1}} \cdot h^{s_{j,8}} \quad (6)$$

- Verify the correctness of the following equations:

$$\prod_{j\in\Gamma} v_{j,1}^{y_j} \stackrel{?}{=} \eta_1^\lambda \prod_{j\in\Gamma} H_1(g\|\cdots\|d'_{j,4})^{y_j} \quad (7)$$

$$\prod_{j\in\Gamma} v_{j,2}^{y_j} \stackrel{?}{=} \eta_2^\lambda \prod_{j\in\Gamma} H_1(g\|\cdots\|d'_{j,6})^{y_j} \quad (8)$$

$$e(\Theta, g_0) \stackrel{?}{=} e(\prod_{j\in\Gamma} H_2(id_j)^{y_j} \cdot g_0^\lambda, \rho) \quad (9)$$

Note that for simplicity, we use $H_1(g\|h\|\cdots\|d'_{j,4})$ and $H_1(g\|\cdots\|d'_{j,6})$ instead of $H_1(g\|h\|Y\|a_0\|a\|T_{j,1}\|T_{j,2}\|T_{j,3}\|d'_{j,1}\|d'_{j,2}\|d'_{j,3}\|d'_{j,4})$ and $H_1(g\|h\|g_1\|g_2\|c\|T_{j,3}\|T_{j,4}\|d'_{j,5}\|d'_{j,6})$, repetitively, in the following parts.

- If the equations (7) (8) (9) all hold, then the auditing proof is valid. Otherwise, it is not.
- If the proof is valid, TPA will send a positive report to the user. Otherwise, a negative report will be sent instead.

8) Open: When users have performed malicious actions on the shared data, at least *t* GMs work together to trace the real identity of the group user as follows:

- Negotiate with each other to construct a polynomial $y(x) = \sum_{l=1}^t f(l) \cdot F_l(x) = \sum_{l=1}^t X(l) \cdot F_l(x)$, where the Lagrange polynomial interpolation $F_l(x) = \prod_{0\le h'\le t, h'\ne l} \frac{x-h'}{l-h'}$.
- Compute the trace key $X = y(0) = \sum_{l=1}^t X(l) \cdot F_l(0)$.
- Compute $upk_i = T_{j,1}/T_{j,2}^X = A_i$, and then reveal the real identity of the signer through upk_i.

This procedure guarantees the current user can be found. When GMs need to find previous users who have affected the shared data, they can trace the data changes by implementing the postorder traversal to the additional binary tree, and then reveal the real user identities of each data change through the above procedure.

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/TBDATA.2017.2701347, IEEE Transactions on Big Data

D. Discussions

1) Group Managers Dynamics:

- GM joining

If a new GM wants to join the group, the PKG computes $S' = S+1$, and tests whether $2t-1 \geq S'$. If it holds, the PKG will compute a new piece $(S', X_{s'})$ with polynomial $f(x)$ and distribute it to the new GM'_S ; otherwise, the PKG chooses a new $(t'-1)$-degree polynomial $f'(x) = b'_0 + b'_1 x + \cdots + b'_{t'-1} x^{t'-1}$, where $2t'-1 \geq S', b'_0 = X, b'_1, \cdots, b'_{t'-1} \in Z_q$, and computes $X'_l = f'(l)(l = 1, 2, \cdots, S')$, i.e. X is divided into S' pieces X'_l and then distributed to GM_l.

In addition, the PKG generates a new key pair $\{$*spk', ssk'*$\}$, and broadcasts it to all the GMs, who can then share it with the existing group users. Note that the process of updating $\{$*spk, ssk*$\}$ has no effect on auditing, because the signing keys, the membership keys and the revocation keys of the existing users do not need to be updated. Nor do the signatures of the data blocks.

- GM leaving

If an existing GM_l wants to leave the group, the PKG first sets $S' = S - 1$, chooses a new $(t'-1)$-degree polynomial $f'(x) = b'_0 + b'_1 x + \cdots + b'_{t'-1} x^{t'-1}$, where $2t'-1 \geq S', b'_0 = X, b'_1, \cdots, b'_{t'-1} \in Z_q$, and then computes and distributes new $X'_l = f'(l)(l = 1, 2, \cdots, S')$ to each GM_l.

In addition, the PKG generates a new key pair $\{$*spk', ssk'*$\}$, and broadcasts it to all the GMs, who can then share it with the existing group users.

2) User Revocation: GMs maintain a users-list, which is composed of each user's related key and expiration time. Once a user's service subscription expires, their signing key should become invalid from then on. In this case, any GM can invoke the *Revoke* algorithm by updating the membership information Ω and the key pair $\{$*spk, ssk*$\}$ and setting the value of the revoked user's expiration time to be negative.

There might be misbehaving users in the group. In this case, any GM can invoke the *Revoke* algorithm as mentioned above. Note that when a user is revoked from a group, GMs do not need to re-compute and re-distribute new keys to the valid users, since the revoked user U_i cannot find $f, b \in Z_q$ such that $f \cdot u + b \cdot rvk_i = 1$, U_i cannot compute the partial signature V_2 any more.

If the revoked user U_i maliciously reveals their signing key $usk_i = (x_i, \pi)$, then the partial signature of other users can be discerned because of the common key π. However, it is not enough to forge a valid signature as the secret key x_j of the other users is still unknown. Therefore, the partial signature V_1 cannot be computed.

As we have demonstrated, valid users do not need to update their keys and the existing signatures. Signatures belonging to the revoked users can be re-computed by the GMs. Specifically, the existing user interacts with GMs to generate a proxy signature key, then GMs use the proxy key to compute the signatures of the revoked users. That transforms them into the signatures which sign by the private key of the existing user.

VI. SECURITY ANALYSIS

The correctness analysis and security analysis of our proposed NPP protocol are established by the following theorems:

Lemma 1. *NPP* is a homomorphic authenticable group signature scheme.

Proof: According to the **Definition 1**, if *NPP* is a homomorphic verifiable, it must satisfy both blockless verification and non-malleability.

- Blockless verification

When TPA selects the subset $\Gamma = \{1, 2\}$ and computes $\lambda = y_1 m_1 + y_2 m_2$, Equations (7), (8) and (9) are all correct (the specific proof process can be referred to Equations (10), (11) and (12) below). Therefore, *NPP* satisfies the property of blockless verification.

- Non-malleability

An attacker without the private key cannot generate the valid tag σ' of m' by combining σ_1 and σ_2, because $\theta_1^{y_1} \cdot \theta_1^{y_2} = [H_2(id_1)^{y_1} \cdot H_2(id_2)^{y_2} \cdot g_0^{m'}]^\pi$, $\theta' = [H_2(id') \cdot g_0^{m'}]^\pi$. If $\theta' = \theta_1^{y_1} \cdot \theta_1^{y_2}$, then $H_2(id') = H_2(id_1)^{y_1} \cdot H_2(id_2)^{y_2} = C$. Once a value id' can be found such that $H_2(id') = C$, it disproves that H_2 is a one-way hash function. Therefore, *NPP* has the property of non-malleability.

Therefore, from **Lemma 1**, we can demonstrate that *NPP* has the properties of public auditing and correctness. ■

Theorem 1 (Public Auditing). Given a message *M* and its group signature σ, the TPA is able to publicly and correctly check the integrity of message *M* under *NPP*.

Proof: Besides the group users, the TPA can execute auditing by randomly choosing a subset Γ of $[1, w]$ without the need of retrieving all data blocks from the cloud, which satisfies the object of public auditing.

The correctness of the verifying process relies on the correctness of Equations (7), (8) and (9). Specific proofs are as follows:

$$\begin{aligned}\prod_{j\in\Gamma} v_{j,1}^{y_j} &= \prod_{j\in\Gamma} (\eta_1^{y_j m_j} \cdot H_1(g\| \cdots \|d'_{j,4})^{y_j}) \\ &= \eta_1^{\sum_{j\in\Gamma} y_j m_j} \cdot \prod_{j\in\Gamma} H_1(g\| \cdots \|d'_{j,4})^{y_j} \\ &= \eta_1^{\lambda} \cdot \prod_{j\in\Gamma} H_1(g\| \cdots \|d'_{j,4})^{y_j}\end{aligned} \tag{10}$$

$$\begin{aligned}\prod_{j\in\Gamma} v_{j,2}^{y_j} &= \prod_{j\in\Gamma} (\eta_2^{y_j m_j} \cdot H_1(g\| \cdots \|d'_{j,6})^{y_j}) \\ &= \eta_2^{\sum_{j\in\Gamma} y_j m_j} \cdot \prod_{j\in\Gamma} H_1(g\| \cdots \|d'_{j,6})^{y_j} \\ &= \eta_2^{\lambda} \prod_{j\in\Gamma} H_1(g\| \cdots \|d'_{j,6})^{y_j}\end{aligned} \tag{11}$$

$$\begin{aligned}e(\Theta, g_0) &= e(\prod_{j\in\Gamma} \theta_j^{y_j}, g_0) \\ &= e(\prod_{j\in\Gamma} (H_2(id_j) g_0^{m_j})^{y_j \pi}, g_0) \\ &= e(\prod_{j\in\Gamma} H_2(id_j)^{y_j} \cdot g_0^{\lambda}, \rho)\end{aligned} \tag{12}$$

From Equations (10), (11) and (12), we conclude that TPA can correctly check the integrity of the shared data without retrieving all the data blocks on behalf of the group users. ■

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/TBDATA.2017.2701347, IEEE Transactions on Big Data

Theorem 2 (Unforgeability). Given shared data *M* and its group signatures σ, it is computationally unfeasible that an untrusted cloud or adversary can generate an invalid auditing proof that can pass the verification under *NPP*.

Proof: According to the security game defined in [7], we firstly define **Game 1** as follows:

Game 1: TPA sends auditing challenge message $\{(j, y_j)\}_{j\in\Gamma}$ of shared data M to the cloud, and the correct auditing proof should be $\{id_j, \Phi_j, \lambda, \Theta\}_{j\in\Gamma}$, which can pass the verification. Now, instead of generating the correct auditing proof, the untrusted cloud generates an invalid auditing proof of $\{id_j, \Phi_j, \lambda', \Theta\}_{j\in\Gamma}$ based on the corrupted shared data *M'*, where $\lambda' = \sum_{j\in\Gamma} y_j m'_j$. Define $\Delta m_j = m'_j - m_j$ for $j \in \Gamma$, and at least one element of $\{\Delta m_j\}_{j\in\Gamma}$ is nonzero (because $M' \neq M$). If the invalid proof still can pass the verification performed by the TPA, then the cloud wins this game. Otherwise, it fails.

Now we show that, if the untrusted cloud wins the above game, we can find a solution to the DL problem. We first assume the untrusted cloud could win **Game 1**. Then, according to Equation (6), we have $e(\Theta, g_0) = e(\prod_{j\in\Gamma} H_2(id)^{y_j} \times g_0^{\lambda'}, \rho)$, where $\lambda' = \sum_{y\in\Gamma} y_j m'_j$. Since $\{id_j, \Phi_j, \lambda, \Theta\}_{j\in\Gamma}$ is the correct auditing proof, we also have $e(\Theta, g_0) = e(\prod_{j\in\Gamma} H_2(id)^{y_j} \times g_0^{\lambda}, \rho)$

Then we learn that $g_0^{\lambda'} = g_0^{\lambda}$, $g_0^{\sum_{j\in\Gamma} y_j m_j} = g_0^{\sum_{j\in\Gamma} y_j m'_j}$, $g_0^{\sum_{j\in\Gamma} y_j \Delta m_j} = \prod_{j\in\Gamma} (g_0^{y_j})^{\Delta m_j} = 1$.

As G_1 is a cyclic group, given two random elements $g, h \in G_1$, there exists $x \in Z_p$ such that $g = h^x$. Without loss of generality, given $g, h \in G_1$, each $g_0^{y_j}$ can be randomly and correctly generated by computing $g_0^{y_j} = g^{\varepsilon_j} h^{\gamma_j}$, where ε_j and γ_j are random values of Z_p. Then, we have $1 = \prod_{j\in\Gamma} (g_0^{y_j})^{\Delta m_j} = \prod_{j\in\Gamma} (g^{\varepsilon_j} h^{\gamma_j})^{\Delta m_j} = g^{\sum_{j\in\Gamma} \varepsilon_j \Delta m_j} \cdot h^{\sum_{j\in\Gamma} \gamma_j \Delta m_j}$.

Clearly, we can find a solution to the DL problem. More specifically, given $h, g = h^x \in G_1$, we can output $g = h^x = h^{-\frac{\sum_{j\in\Gamma} \gamma_j \Delta m_j}{\sum_{j\in\Gamma} \varepsilon_j \Delta m_j}}$, $x = -\frac{\sum_{j\in\Gamma} \gamma_j \Delta m_j}{\sum_{j\in\Gamma} \varepsilon_j \Delta m_j}$, unless the denominator is zero. However, as we defined in **Game 1**, at least one element of $\{\Delta m_j\}_{j\in\Gamma}$ is nonzero, and ε_j is a random element of Z_p; therefore, the probability of the denominator being zero is $1/p$, which is negligible because p is a large prime. It means that once the untrusted cloud wins **Game 1**, we can find a solution to the DL problem with a probability of $1 - 1/p$, which contradicts the assumption that the DL problem is computationally unfeasible in G_1. Therefore, it is computationally unfeasible for the untrusted cloud to generate an invalid auditing proof that can pass the verification. ■

Theorem 3 (Authorized Auditing). *NPP* supports authorization verification.

Proof: Since the *ID* of the TPA is encrypted with the public key *spk*, any other entity excluded by the group cannot obtain the valid *ID* without the private key *ssk*. Therefore, they cannot forge a valid message sig_{AUTH} to pass the authentication. In addition, the timestamp *t* included in sig_{AUTH} ensures that a previous authorization message cannot be utilized as a valid message. Therefore, only the TPA who has been authorized by the group can challenge the cloud. ■

Theorem 4 (Identity Privacy). Given a message *M* and its group signature σ, it is computationally unfeasible for a verifier to reveal the identity of the signer.

Proof: Because TPA cannot infer the secret value X from the known $Y = g^X$ and g, it is computationally unfeasible for the TPA to infer the real identity of the signer from signatures $V_{j,1}$ and $V_{j,2}$. In addition, although ρ, used to verify the partial signature θ, is public, users in the group share the same secret value π, hence the partial signatures θ of all users in the group are the same. Therefore, TPA cannot infer the real identity of the signer from the signature θ. ■

Theorem 5 (Traceability and Non-frameability). At least t GMs can work together to recover the identity of the signer from the signatures.

Proof: The secret value X is divided into S pieces by the PKG based on (t, s) secret sharing scheme, and the S pieces are distributed to S GMs respectively. GM_l owns piece X_l of X. By Lagrange polynomial interpolation, at least t GMs work together to recover $X = y(0) = \sum_{l=1}^{t} X_l \cdot F_l(0)$ and then compute $upk_i = T_{j,1}/T_{j,2}^X = A_i$. Therefore, the identity of the signer can be traced by at least t GMs after recovering upk_i. That means at least t GMs work together, the group managers can expose the signers identity from the signatures. Note that since the tracing process is performed by multiple GMs instead of a single entity, it eliminates the potential risks brought by power centralization and ensures non-frameability during the tracing process.

Moreover, by implementing the postorder traversal to the additional binary tree and revealing the real identities from the signatures, GMs can trace each user who has performed actions on the shared data. Finally, if the current data block has been damaged, through postorder traversal, GMs can verify all the previous records of this data block one by one with the help of TPA until they find the latest right data block. ■

Theorem 6 (Data Traceability and Recoverability). *NPP* supports data traceability and recoverability.

Proof: According to the data structure based on the binary tree, the cloud server can record every change of the shared data blocks. Through the records, the group users can trace the data changes. Even if the current data block has been damaged, the users can recover the latest right data by verifying the older blocks one by one in the records.(We have made a detailed description in the subsection V.B) ■

VII. EVALUATION

A. Functionality Comparison

Table II lists the features of *NPP* compared with other auditing schemes *Knox* [8] and *PDM* [19] for shared data in the cloud. As Table II shows, the two very important properties non-frameability and data traceability and recoverability are all not found in other two schemes. Whats more, comparing to their schemes, *NPP* adds a lot of features. Hence, *NPP* has wider application than *Knox* and *PDM*.

B. Performance Analysis

In our experiments, we utilize Pairing Based Cryptography (PBC) library [28] to simulate the cryptographic operations in

TABLE II
FUNCTIONALITY COMPARISON

	Knox [8]	*PDM* [19]	*NPP*
Public Auditing	Yes	Yes	Yes
Authorized Auditing	Yes	No	Yes
Identity Privacy	Yes	No	Yes
Traceability	Yes	No	Yes
Non-frameability	No	No	Yes
Data Traceability and Recoverability	No	No	Yes
User Revocation	No	Yes	Yes

TABLE III
COMMUNICATION COST COMPARISON

Scheme	Communication Cost(KB)
Knox [8]	$D\|w\| + (10D + k + 1)\|q\| + D(\|id\| + \|T\|) = 106.4$
PDM [19]	$(d + 5)\|q\| + D\|id\| = 4.6 + 0.02d$
NPP	$D\|w\| + (16D + 2)\|q\| + D\|id\| = 149.4$

TABLE IV
COMPUTATION COST COMPARISON

Scheme	Computation Cost(s)
Knox [8]	$D(2Exp_{Z_q} + 4Mul_{Z_q}) + kMul_{Z_q} + D(17Exp_{G_1} +11Mul_{G_1}) + D(Exp_{G_T} + Mul_{G_T}) + 4Pair = 1.351$
PDM [19]	$(k + 6)Exp_{G_1} + (k + D + 3)Mul_{G_1} + (D + 3)Pair = 1.446$
NPP	$D(22Exp_{Z_q} + 14Mul_{Z_q}) + D(2Exp_{G_1} + 2Mul_{G_1}) +2Pair = 0.236$

EXP_{Z_q} and MUL_{Z_q} are one exponentiation operation and one multiplication operation on Z_q respectively; EXP_{G_1} and MUL_{G_1} are one exponentiation operation and one multiplication operation on Group G_1 respectively; EXP_{G_T} and MUL_{G_T} are one exponentiation operation and one multiplication operation on Group G_T respectively; Pair is a bilinear pairing operation.

the schemes. All tests are applied to a Ubuntu system with i7 3.40GHz-Intel Core and 4GB-memory over 1,000 times. We set the size of elements in G_1, G_2, G_T, Z_q as 160 bits (i.e., $|q| = 160bit, |T| = 160bit$), the identity of each data block as 50 bits (i.e., $|id| = 50bit$), and the number of the shared data blocks as 1,000,000 (i.e., $w = 1,000,000$ and $|w| = 20bit$). Each data block contains 100 elements (i.e., $k = 100$), the size of each data block is 2KB and the shared data is 2GB in all the experiments. Based on random sampling method [7], if the TPA select $D = 460$ data blocks, the detection probability is greater than 99%, and if $D = 300$, the detection probability is greater than 95%. To keep a higher detection probability, we chose $D = 460$. The performance analysis and the experiment results are as follows.

1) Communication cost: As Table III shows, the communication costs of *NPP* and *Knox* are both constant, but that of *PDM* linearly increases with the number of the group users. To support user revocation, *NPP* adds V_2 as a partial signature, which brings additional overhead $|V_2| = 7D|q| = 62.89$KB compared with *Knox*. However, compared with the shared data size of 2GB, the additional communication cost of 62.89KB is small and acceptable.

2) Computation cost: As Table IV shows, the computation costs of all the three schemes are constant, which are independent of the number of the group users. Obviously, *NPP* outperforms *Knox* and *PDM*. Because the operations on G_T and the pairing operations are time-consuming, *NPP* has no operations on G_T and has the fewest pairing operations. In contrast, *Knox* has several operations on G_T and more pairing operations. Although the *PDM* has no operations on G_T, the number of pairing operations in *PDM* linearly increase with D.

3) Performance results: From the above analysis, *NPP* has the lowest computation cost compared with *Knox* and *PDM*. Specifically, the computation cost of *Knox* is almost 5.7 times that of *NPP*, and *PDM* is almost 6.1 times that of *NPP*. Therefore, in terms of computation cost, *NPP* significantly outperforms *Knox* and *PDM*. As for communication cost, although the cost of *NPP* is a bit more than that of *Knox*, the additional overhead 63KB is small and acceptable compared to the size of shared data with 2GB.

VIII. Conclusion

In this paper, we propose a novel multi-level privacy preserving public auditing scheme for cloud data sharing with multiple managers. During the process of auditing, the TPA cannot obtain the identities of the signers, which ensures the identity privacy of the group users. Moreover, unlike the existing schemes, the proposed *NPP* requires at least *t* group managers to work together to trace the identity of the misbehaving user. Therefore, it eliminates the abuse of single-authority power and ensures non-frameability. Exceptionally, group users can trace the data changes through the designed binary tree and recover the latest correct data block when the current data block is damaged. In addition, the analysis and the experimental results show that *NPP* is provably secure and efficient.

References

[1] D. Fernandes, L. Soares, J. Gomes, et al, "Security issues in cloud environments: a survey," *International Journal of Information Security*, vol. 12, no. 2, pp. 113-170, 2014.

[2] W. Hsien, C. Yang, and M. Hwang, "A survey of public auditing for secure data storage in cloud computing," *International Journal of Network Security*, vol.18, no.1, pp. 133-142, 2016.

[3] J. Yu, K. Ren, C. Wang, et al, "Enabling Cloud Storage Auditing with Key-Exposure Resistance," *IEEE Transactions on Information Forensics and Security*, vol.10, no.6, pp. 1167-1179, 2015.

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

[5] S. Yu, "Big privacy: challenges and opportunities of privacy study in the age of big data," *IEEE Access*, vol. 4, no. 6, pp. 2751-2763, 2016.

[6] C. Wang, Q. Wang, K. Ren, et al, "Privacy-preserving public auditing for data storage security in cloud computing," *Proceedings of IEEE INFOCOM*, pp. 1-9, 2010.

[7] B. Wang, B. Li, and H. Li, "Oruta: privacy-preserving public auditing for shared data in the cloud," *IEEE Transactions on Cloud Computing*, vol.2, no.1, pp.43-56, 2014.

[8] B. Wang, B. Li, and H. Li, "Knox: privacy-preserving auditing for shared data with large groups in the cloud," *Applied Cryptography and Network Security. Springer Berlin Heidelberg*, pp. 507-525, 2012.

[9] B. Wang, H. Li, and M. Li, "Privacy-preserving public auditing for shared cloud data supporting group dynamics," *Proceedings of IEEE ICC*, pp. 1946-1950, 2013.

[10] B. Wang, B. Li, and H. Li, "Public auditing for shared data with efficient user revocation in the cloud," *Proceedings of IEEE INFOCOM*, pp. 2904-2912, 2013.

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

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/TBDATA.2017.2701347, IEEE Transactions on Big Data

[12] C. Liu, J. Chen, L. Yang, et al, "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Transactions on Parallel and Distributed Systems*, vol.25, no.9, pp. 2234-2244, 2014.

[13] H. Wang, and Y. Zhang, "On the Knowledge Soundness of a Cooperative Provable Data Possession Scheme in Multicloud Storage," *IEEE Transactions on Parallel and Distributed Systems*, vol.25, no.1, pp. 264-267, 2014.

[14] L. Huang, G. Zhang, and A. Fu, "Privacy-preserving public auditing for dynamic group based on hierarchical tree," *Journal of Computer Research and Development*, vol.53, no.10, pp. 2334-2342, 2016.

[15] Y. Yu, J. Ni, M. Au, et al, "Comments on a public auditing mechanism for shared cloud data service," *IEEE Transactions on Services Computing*,vol.8, no.6, pp. 998-999 2015.

[16] G. Ateniese, R. Burns, R. Curtmola, et al, "Provable data possession at untrusted stores," *Proceedings of ACM CCS*, pp. 598-609, 2007.

[17] A. Juels, and B. Kaliski, "PORs: Proofs of retrievability for large files," *Proceedings of ACM CCS*, pp. 584-597, 2007.

[18] Y. Yu, M. H. Au, and Y. Mu, "Enhanced privacy of a remote data integrity-checking protocol," *International Journal of Information Security*, vol. 14, no. 4, pp. 307-318, 2015.

[19] J. Yuan, and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification," *Proceedings of IEEE INFOCOM*, pp. 2121-2129, 2014.

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

[21] L. Huang, G. Zhang, A. Fu, "Certificateless Public Verification Scheme with Privacy-preserving and Message Recovery for Dynamic Group," *Proceedings of ACSW*, 2017.

[22] T. Jiang, X. Chen, and J. Ma, "Public integrity auditing for shared dynamic cloud data with group user revocation," *IEEE Transactions on Computers*, vol.65, no.8, pp.2363-2373, 2016.

[23] H. Wang, "Proxy Provable Data Possession in Public Clouds," *IEEE Transactions on Services Computing*, vol.6, no.4, pp.551-559, 2013.

[24] Y. Yu, Y. L, J. N, et al, "Comments on public integrity auditing for dynamic data sharing with multiuser modification," *IEEE Transactions on Information Forensics and Security*, vol.11, no.3, pp.658-659, 2016.

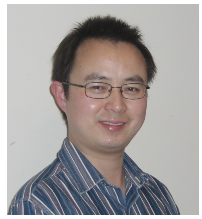
[25] H. Jin, D. Wong, and Y. Xu, "Efficient group signature with forward secure revocation," *Security Technology. Springer Berlin Heidelberg*, pp. 124-131, 2009.

[26] A. Shamir, "How to share a secret?" *Communications of the ACM*, vol.22, no.11, pp. 612-613, 1979.

[27] S. Yu, C. Wang, K. Ren, et al, "Achieving secure, scalable, and fine-grained data access control in cloud computing," *Proceedings of IEEE INFOCOM*, pp. 1-9, 2010.

[28] Pairing Based Cryptography (PBC) Library [Online]. Available: http://crypto.stanford.edu/pbc/.

Anmin Fu is an associate professor and supervisor of Ph.D. students of Nanjing University of Science and Technology, China. He received his B.S. degree in Communication Engineering from Lanzhou University of Technology, China, in 2005. He received his M.S. and Ph.D. degrees in Cryptography and Information Security from Xidian University in 2008 and 2011, respectively. His research interests include cloud computing security, wireless security and applied cryptography.

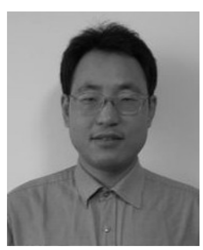


Shui Yu (M'05-SM'12) is currently a Senior Lecturer of School of Information Technology, Deakin University. He is a Senior Member of IEEE, and a member of AAAS and ACM, the Vice Chair of Technical Committee on Big Data Processing, Analytics, and Networking of IEEE Communication Society.

Dr Yu's research interest includes Cybersecurity, Networking Theory, Big Data, and Mathematical Modelling. He has published two monographs and edited two books on big data, more than 150 technical papers, including top journals and top conferences, such as IEEE TPDS, IEEE TCC, IEEE TCSS, IEEE TC, IEEE TIFS, IEEE TMC, IEEE TKDE, IEEE TETC, and IEEE INFOCOM. Dr Yu initiated the research field of Networking for Big Data in 2013. His h-index is 25.

Dr Yu actively serves his research communities in various roles. He served IEEE Transactions on Parallel and Distributed Systems as an AE (2013-2015), and is currently serving the editorial boards of IEEE Communications Surveys and Tutorials (exemplary editor for 2014), IEEE Access, IEEE Internet of Thing Journal, IEEE Communications Letters (exemplary editor for 2016), and a number of other international journals. Moreover, he has organized several Special Issues either on big data or cybersecurity. He has served more than 70 international conferences as a member of organizing committee, such as publication chair for IEEE Globecom 2015 and IEEE INFOCOM 2016 and 2017, TPC co-chair for IEEE BigDataService 2015, IEEE ITNAC 2015, and General chair for ACSW 2017.

Yuqing Zhang is a professor and supervisor of Ph.D. students of Graduate University of Chinese Academy of Sciences, China. He received his B.S. and M.S. degrees in computer science from Xidian University, China, in 1987 and 1990 respectively. He received his Ph.D. degree in Cryptography from Xidian University in 2000. His research interests include cryptography and network security.



Huaqun Wang received the BS degree in mathematics education from the Shandong Normal University and the MS degree in applied mathematics from the East China Normal University, both in China, in 1997 and 2000, respectively. He received the Ph.D. degree in Cryptography from Nanjing University of Posts and Telecommunications in 2006. Now, he is a full professor in Nanjing University of Posts and Telecommunications. His research interests include applied cryptography, network security, blockchain, and cloud computing security. He has published more than 70 papers. He has served in the program committee of several international conferences and the editor board of international journals.

Chanying Huang received the BS degree in School of Computer Science & Technology from Harbin Institute of Technology (HIT) in July 2009, and the PhD in Graduate School of Information Security from Korea University (KU) in February 2015. She is a lecturer in School of Computer Science & Engineering, Nanjing University of Science & Technology. Her research focuses on Sensor Network Security, Security and Privacy issues in BANs, and Security and Privacy issues in eHealth/mHealth.