Pers Ubiquit Comput (2017) 21:923–936
DOI 10.1007/s00779-017-1051-z

ORIGINAL ARTICLE

Privacy-preserving identity-based file sharing in smart city

Xiling Luo[1] · Yili Ren[1] · Jiankun Hu[2] · Qianhong Wu[1] · Jungang Lou[3]

Received: 30 August 2016 / Accepted: 28 February 2017 / Published online: 28 June 2017

Abstract People are experiencing an evolution of smart cities. Building a smart city will enhance economic competitiveness, social cohesion, and quality of life of its citizens. But smart cities accumulate and process large amount of files, which raises security and privacy concerns at individual and community levels. In the case of file sharing in smart cities, security should be considered to embrace file confidentiality, file integrity, receiver privacy, and sender privacy. In this paper, we propose a privacy-preserving identity-based file sharing (PIFS) scheme to meet these security goals. In PIFS, identity managers for receivers and senders designate them secret keys associated with their identities, respectively. Receivers and senders can register their identities to the group managers without leaking their secret keys. Then a sender can share confidential files with a peer, leaking neither identity of them. However, in case of dispute, the receiver group manager and sender open authority can trace the receiver and the sender, respectively. The security properties of our scheme are formally proven. Analysis shows that our scheme is efficient and practical.

✉ Qianhong Wu
qianhong.wu@buaa.edu.cn

[1] School of Electronic and Information Engineering, Beihang University, No. 37, Xueyuan Road, Haidian District, Beijing, 100191, China

[2] School of Engineering and IT, University of New South Wales, Sydney, Australia

[3] School of Information Engineering, Huzhou University, Huzhou, 313000, China

Keywords Smart city · Group signature · Group encryption · Identity based · Knowledge proof

1 Introduction

With the majority of the population living in city environments nowadays, the cities face more and more challenges, such as economic growth, mobility, energy, safety, and governance. These challenges are difficult to tackle as they grow in size and complexity. There is an urgent need for cities to become smarter in how they manage their infrastructures and resources. Therefore, the concept of smart cities [18] comes into being. A smart city is an urban development vision to integrate multiple information and communication technology (ICT) solutions in a secure fashion to manage a city's assets. Building a smart city is aimed to enhance economic competitiveness, social cohesion, and quality of life of its citizens. Smart city technologies are used to managing various city infrastructures such as transportation systems, hospitals, power plants, water supply networks, and other community services. In smart cities, every device and service are linked to an information network through the Internet. These devices include traditional static sensors and personal wearable devices. Data are collected from citizens and devices. These allow city officials to interact directly with the city infrastructures and to monitor the city.

The smart city will take advantage of communication and sensor capabilities sewn into the cities' infrastructures to optimize electrical, transport, and other logistical operations supporting daily life, thereby improving the quality of life for everyone. Therefore, there must be many files to be produced, transmitted, and shared. For example, mobile phone files with traffic congestion condition can be sent to the

traffic department of the city. The majority of the files that are used to communicate the city applications are as follows: user files for monitoring the public behavior, document files for better statistical and feasibility studies, industry files for monitoring the market demand, and business files for more commerce and financial analysis.

The smart city will require higher degrees of network connectivity to support sophisticated features. This has the potential to open up new vulnerabilities. For this reason, one of the biggest challenges facing smart city development is related to file security and user privacy. File confidentiality is needed to protect the privacy of citizens and valuable information of stakeholders in the city. File integrity protects file against modifications that can lead to harmful decisions [2]. In actuation requests, it confirms the authenticity of the request to avoid unauthorized changes in the city's physical infrastructure. User's sensitive information (privacy) can detail much about a person's life they do not wish to be revealed as to medical, political, or social contexts. The priority must be to establish user confidence in the smart city, as otherwise users will hesitate to accept the services provided by smart cities.

1.1 Our contribution

Motivated by the above scenarios, we propose a new file sharing scheme for smart city with file confidentiality, file integrity, receiver privacy, and sender privacy, referred to as privacy-preserving identity-based file sharing (PIFS). We first contribute the model and framework of PIFS. Second, we construct a concrete PIFS scheme in a modular way. Then we prove its relevant security properties. Finally, we give a detailed efficiency analysis for our scheme.

The privacy-preserving identity-based file sharing framework involves nine entities: two group managers for sender and receiver, respectively; a group of senders; a group of receivers; a sender open authority; and four identity managers. It also involves six procedures: system initialization, community setup, user join, file upload, file access, and peer tracing. One procedure will have at least one participant. Note that the sender group manager and receiver group manager can be implemented as one manager in practice. Likewise, four identity managers can also be implemented as one manager.

We design a concrete PIFS scheme in a modular way. In order to get an efficient and practical scheme, we use four primitives, i.e., a public-key encryption scheme which satisfies CCA2 security, an identity-based encryption scheme which satisfies anonymity and semantic security, knowledge proofs which satisfy special properties, and an identity-based group signature scheme. Our proposal is a new identity-based file sharing scheme for smart city with receiver anonymity, sender anonymity, semantic security, peer traceability, and file integrity. Receiver anonymity and sender anonymity protect users from a hostile environment where the attacker may want to extract information about their identities. Semantic security protects files from being attacked. Peer traceability ensures the tracking is reliable and prevents collusion attacks. File integrity protects files against modifications.

We prove the security of our concrete PIFS scheme according to our security definitions. And we give the analysis of probability as well as time complexity. Our scheme has constant complexity in computation and communication. This means the scheme will be efficient in practice. Our PIFS scheme is a fully functional file sharing scheme for smart city.

1.2 Related work

Some experts have researched smart cities. Zanella characterized an urban Internet of Things system according to specific application domains [26]. Al-Hader et al. believed that a smart city provides interoperable, Internet-based government services that enable ubiquitous connectivity to transform key government processes, both internally across departments and employees and externally to citizens and businesses [1]. To close the gap in the literature about smart cities and in response to the increasing use of the concept, Chourabi et al. proposed a framework to understand the concept of smart cities [5]. Su, Li, and Fu mainly focused on recent research and the concept of "smart city," summarizing the relationship between "smart city" and "digital city," putting forward the main content of application systems in the smart city [21].

Efforts have been devoted to ensure the security and privacy in smart cities. Suciu proposed a new platform for using cloud computing capacities for provision and support of ubiquitous connectivity and real-time applications and services for smart cities' needs [20]. Elmaghraby and Losavio examined two important and entangled challenges for smart city: security and privacy [7]. They also presented a model representing the interactions between person, servers, and things. Khan, Pervez, and Ghafoor presented a security and privacy framework for secure and privacy-aware service provisioning in smart cities [13]. Their framework aimed to provide end-to-end security and privacy features for trustable data acquisition, transmission, processing, and legitimate service provisioning. Bohli presented a data platform for management of a smart city and pointed out the main security and privacy threats [2]. They also presented use cases showing the benefits of such a platform for realizing typical smart city application. Wang looked into security issues in smart city infrastructure from both technical and business operation perspectives and proposed an approach to analyze threats and to improve data security of smart

city systems [24]. Wu et al. proposed a privacy-preserving system that guarantees message trustworthiness between vehicles. This work offered the possibility of tracing the message generator and its endorsers [23]. Chen et al. proposed a new verifiable database framework from vector commitment based on the idea of commitment binding [6].

As for file sharing, a large amount of works have been proposed. Hoßfeld et al. examined the feasibility of the eDonkey file-sharing service in GPRS networks, detected problems of the interaction between P2P and the mobile network, and found solutions to overcome them. Furthermore, this work measured and analyzed the characteristics of mobile P2P and gave first empirical performance values [11]. Shen presented an efficient and adaptive decentralized file replication algorithm that achieves high query efficiency and high replica utilization at a significantly low cost [22]. In [17], Lu et al. proposed an EigenTrust evolutionary game model based on the renowned EigenTrust reputation model. In this model, they used evolutionary game theory to model strategic peers and their transaction behaviors, which is close to the realistic scenario. Iamnitchi et al. proposed a novel perspective in [12], for analyzing data access workloads that considers the implicit relationships that form among users based on the data they access. Huang et al. proposed a notion called forward secure ID-based ring signature, which is an essential tool for building cost-effective authentic and anonymous data sharing system [10].

Although there are many works for smart city and file sharing, none of them has taken into the consideration of file confidentiality, file integrity, receiver privacy, and sender privacy at the same time. We will construct a new group encryption (GE) and apply a group signature to achieve this goal. Kiayias, Tsiounis, and Yung provided the conception of group encryption [14] and a modular design including zero-knowledge proofs, digital signature schemes, public-key encryption schemes with CCA2 security, and key-privacy and commitment schemes. Cathalo, Libert, and Yung [4] proposed a group encryption with non-interactive realization in the standard model. Independently, Qin, Wu, Susilo, and Mu [19] considered a similar primitive called group decryption. The group decryption has non-interactive proofs and short ciphertexts. Libert, Yung, Joye, and Peters proposed a traceable GE [16] which can trace all the ciphertexts encrypted by a specific user without abolishing the anonymity of the others. In [15], Luo et al. presented an identity-based group encryption, but they did not protect the privacy of the senders or the integrity of the encrypted files.

1.3 Outline of paper

In Section 2, we formally describe the system model and security requirements of PIFS. In Section 3, we provide a overview of PIFS and a concrete PIFS scheme. In Section 4, we prove some related security properties. In Section 5, we give our conclusion of this work.

2 Modeling PIFS

In this section, we formalize the model of PIFS system and the security requirements of this system.

2.1 The PIFS system

PIFS system involves nine entities. There are two group managers (GM_R, GM_S) in this system. GM_R administers the receiver group and traces the receiver when it is necessary. Likewise, GM_S administers the sender group. A sender open authority (OA_S) traces the sender's identity when it is necessary. Due to the sender group manager, sender members, and sender open authority are identity based, we add three identity managers for sender group manager, sender members, and sender open authority, respectively. They are denoted as IM_{GM_S}, IM_S, and IM_{OA_S}. A group of legitimate receivers receive messages from the senders anonymously. A group of legitimate senders have secret messages to be sent to the legitimate receivers anonymously. A receiver identity manager (IM_R) can issue the private keys to the users. For instance, as shown in Fig. 1, anonymous sender 2 sends a file to anonymous receiver 2.

We emphasize that the sender group manager and receiver group manager can be implemented as one manager in practice. Similarly, the identity managers for sender group manager, sender members, receiver members, and sender open authority can also be implemented as one manager. We here describe them separately so that one can realize our system in a flexible way.

PIFS consists of the following algorithms.

– $(Params, MSK_R) \leftarrow$ **SysInit**(λ). System initialization is a polynomial time algorithm which takes a security parameter λ as input. It outputs system parameter $Params$ and a receiver master-key MSK_R. For sender group manager, the identity manager IM_{GM_S} has a master-key pair (MSK_{GM_S}, MPK_{GM_S}) and one-way NP-relation $\langle \mathscr{R}_{GM_S} \rangle$ with trapdoor MSK_{GM_S}. For sender members, the identity manager IM_S has a master-key pair (MSK_S, MPK_S) and one-way NP-relation $\langle \mathscr{R}_S \rangle$ with trapdoor MSK_S. For sender open authority, the identity manager IM_{OA_S} has a master-key pair (MSK_{OA_S}, MPK_{OA_S}) and one-way NP-relation $\langle \mathscr{R}_{OA_S} \rangle$ with trapdoor MSK_{OA_S}. A samplable family of one-way NP-relation $\mathscr{F} = \{\langle \mathscr{R}_{C,i} : i \rangle\}$ with trapdoor gsk_i. It is used to issue certificate for senders. $Params$

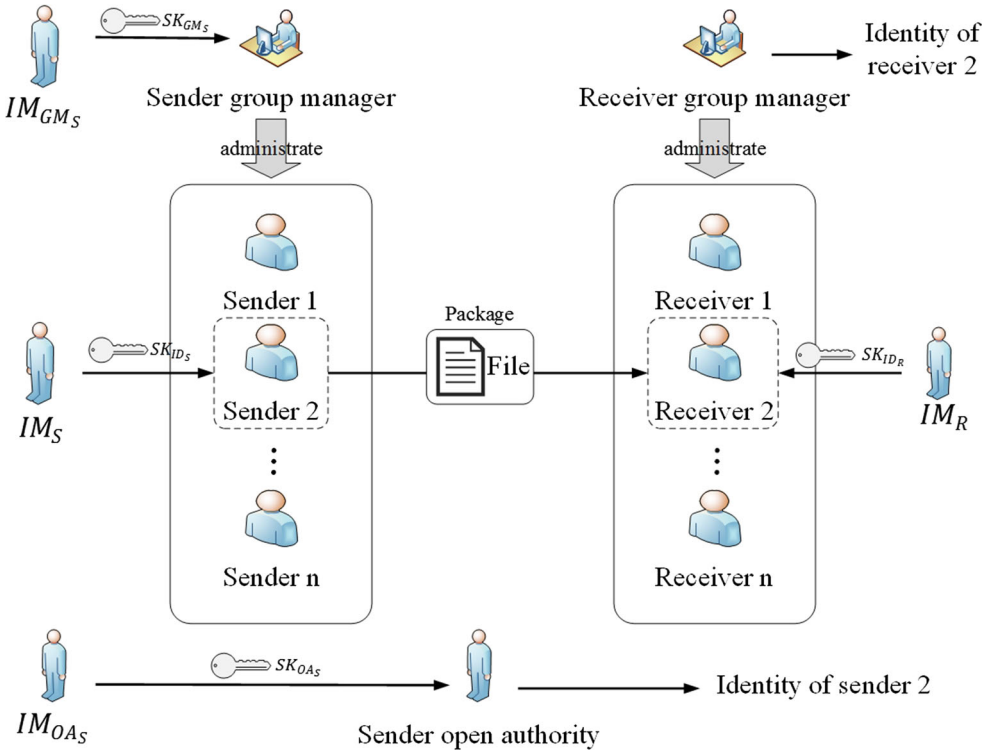

Fig. 1 System model

includes but is not limited to them: $(MPK_{GM_S}, MPK_{OA_S}, MPK_S, \mathscr{R}_{GM_S}, \mathscr{R}_S, \mathscr{R}_{OA_S}, \mathscr{F})$.

– $(PK_{GM_R}, SK_{GM_R}, SK_{OA_S}, aux_{OA_S}, SK_{GM_S}, aux_{GM_S}, \langle\mathscr{R}_{C,GM_S}\rangle) \leftarrow$ **ComSetup**$(Params, ID_{OA_S}, ID_{GM_S}, MSK_{OA_S}, MSK_{GM_S})$. Community setup is a polynomial time algorithm which takes system parameter $Params$, sender OA identity ID_{OA_S}, and sender group manager identity ID_{GM_S} as inputs. GM_R computes receiver group public key and receiver group private key (PK_{GM_R}, SK_{GM_R}). IM_{OA_S} uses his secret key MSK_{OA_S} to compute the secret key SK_{OA_S} and auxiliary information aux_{OA_S} for sender OA. IM_{GM_S} uses his secret key MSK_{GM_S} to compute the secret key SK_{GM_S} and auxiliary information aux_{GM_S} for GM_S. IM_{GM_S} also samples $\mathscr{F}$ to get a relation $\langle\mathscr{R}_{C,GM_S}\rangle$.

– $(SK_{ID_R}, SK_{ID_S}, aux_{ID_S}, cert_{ID_S}, reg) \leftarrow$ **UserJoin**$(Params, ID_R, MSK_R, ID_S, MSK_S)$. This is a polynomial time algorithm. For the receiver, it takes system parameter $Params$, receiver's identity ID_R, and MSK_R as inputs. IM_R computes the receiver's corresponding private key SK_{ID_R}. Each receiver can register his identity as a group member to GM_R. GM_R maintains the receivers' ID list, denoted as $I = \{ID_{R1}, ..., ID_{Rj}\}$. For the sender, it takes system parameter $Params$, sender's identity ID_S, and MSK_S as inputs. IM_S computes the sender's corresponding private key SK_{ID_S} and auxiliary information aus_{ID_S}. There is a pair of interactive protocols $(Join, Issu)$ between the sender and GM_S with common inputs ID_{GM_S} and ID_S. $Issu$'s additional inputs are SK_{GM_S} and aux_{GM_S}. $Join$'s additional inputs are SK_{ID_S} and aux_{ID_S}. $Join$ obtains $cert_{ID_S}$ satisfying $((SK_{ID_S}, aux_{ID_S}, cert_{ID_S}), ID_S) \in \mathscr{R}_{C,GM_S}$, and $Issu$ stores $(ID_S, cert_{ID_S})$ in a registration table reg.

– $(P) \leftarrow$ **FileUpload**$(M, F, Params, ID_R, PK_{GM_R}, ID_S, SK_{ID_S}, aux_{ID_S}, ID_{GM_S}, ID_{OA_S}, cert_{ID_S})$. This is a polynomial time algorithm which takes a session key M, system parameters $Params, PK_{GM_R}, ID_S, SK_{ID_S}, aux_{ID_S}, ID_{GM_S}, ID_{OA_S}, ID_R, cert_{ID_S}$, and a file F as input. It outputs a package P.

– $(F) \leftarrow$ **FileAccess**$(Params, P, SK_{ID_R}, ID_{GM_S}, ID_{OA_S})$. This is a polynomial time algorithm which takes $Params, ID_{OA_S}, SK_{ID_R}, ID_{GM_S}$, and P as inputs. It outputs 1 for valid verification or 0 for invalid verification. If it outputs 1, then it outputs the message F, else outputs "reject."

– $(ID_R, ID_S) \leftarrow$ **PeerTracing**$(SK_{GM_R}, ID_{GM_S}, SK_{OA_S}, ID_{OA_S}, reg, F)$. For receiver tracing, GM_R first verifies if the verifier outputs 1, then verifies the correctness of the encryption of ID_R. If both of them are valid, GM_R runs a polynomial time algorithm which takes F and SK_{GM_R} as inputs. It outputs ID_R of the receiver. For sender tracing, the OA_S with SK_{OA_S} has read access to reg and outputs identity ID_S for the corresponding sender using an $Open$ subprocedure. And ω is the proof of this claim. Then it checks if ω

is a valid proof of the sender's identity using a $Judge$ subprocedure.

2.2 Security requirements

We focus on two important and entangled challenges, i.e., security and privacy. Files in a smart city must be protected in order to reduce the risk of files theft and fake that can lead to a series of damage. Since digital citizens are more and more instrumented with data available about their identity, location, and activities, privacy seems to disappear. We propose the PIFS system to deal with security and privacy problems and satisfy security requirements as follows:

- **File confidentiality**. A file cannot be leaked to any unauthorized user, entity, or program. The characteristics of the file also cannot be utilized.
- **File integrity**. File integrity means the file cannot be changed without authorization. In the PIFS system, we propose a method to verify the file and ensure the file integrity.
- **Receiver privacy**. In the process of file transfer, identity of the receiver may be leaked. Any attacker should not get any useful identity information of the receiver from the file, even he colludes with others.
- **Sender privacy**. It is similar to receiver privacy protection. But we focus on sender privacy, specially, sender's identity protection.

3 The proposal

3.1 High-level description of the scheme

In this section, we provide a high-level description of our PIFS scheme in the smart city file sharing setting. We assume that our PIFS scheme works in end-to-end mode in a smart city. A peer (sender) shares a file with the other one (receiver). Since a file may be very large, it should be very inefficient using a public-key cryptography to encrypt a file directly. Instead, we can encrypt a session key and use a symmetrical encryption to encrypt the file. This is a relatively efficient solution.

The first question of a secure file sharing scheme in a smart city is the file security. The second question is how to protect identity of the receiver from being leaked, because the attacker may extract the receiver's identity and then obtains the total constituent of the receiver group. Group encryption is a good method to achieve both of the two goals. The existing GE schemes are all realized in the public key infrastructure (PKI) setting, in which complicated certificate management is required to ensure security. It seems appealing to use identity-based encryption (IBE) schemes to replace the PKI-based public-key encryptions in GE. Therefore, we employ an identity-based encryption with ANO-IND-ID-CPA security [9] and a public-key encryption with CCA2 security [3] to propose a new cryptographic primitive called identity-based group encryption (IBGE) [15]. As a component of our PIFS, IBGE ensures message confidentiality and receiver anonymity.

File integrity is another essential element of the file sharing scheme. Just like the protection of receiver privacy, the sender privacy should also be protected in the smart city file sharing setting. Obviously, sender traceability is necessary, too. As we all know, digital signature ensures file integrity. And group signature has properties of sender anonymity and traceability. To achieve these goals in identity-based setting, we apply an identity-based group signature [25] scheme as a component of our PIFS.

In order to verify that if the encrypted receiver's identity and the identity that forms IBE ciphertext are identical, we link the identity-based encryption with the public-key encryption using a zero-knowledge proof. This zero-knowledge proof indicates that the IBE ciphertext has not been tampered as well as the ciphertext is well formed. This means that the zero-knowledge proof makes our PIFS scheme achieve CCA2 security.

3.2 A concrete PIFS scheme

As illustrated in Fig. 1, nine entities are involved in our PIFS scheme: receiver group managers (GM_R), sender group managers (GM_S), a group of receiver, a group of sender, sender open authority (OA_S), four identity managers (IM_S, IM_R, IM_{OA_S}, IM_{GM_S}) for sender members, receiver members, sender open authority, and sender group manager. Two group managers administrate receiver group and sender group, respectively. Receiver group manager is able to identify the receiver. Sender open authority specially identifies the sender. Four identity managers issue the secret keys to the four entities. A sender shares a file with a peer.

We use bilinear groups to construct our scheme. Let p be a large prime. $\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T$ are three cyclic groups of prime order p. $g, \hat{g}$ are generators of $\mathbb{G}, \hat{\mathbb{G}}$ respectively. We say that $\mathbb{G}$, $\hat{\mathbb{G}}$ are bilinear groups if there is a bilinear map $e : \mathbb{G} \times \hat{\mathbb{G}} \rightarrow \mathbb{G}_T$ that satisfies the following properties. *Bilinear:* we say a map $e : \mathbb{G} \times \hat{\mathbb{G}} \rightarrow \mathbb{G}_T$ is bilinear if $e(u^a, \hat{u}^b) = e(u, \hat{u})^{ab}$ for all $u \in \mathbb{G}, \hat{u} \in \hat{\mathbb{G}}, a, b \in \mathbb{Z}_p$. *Non-degenerate:* if $g, \hat{g}$ are generators of $\mathbb{G}, \hat{\mathbb{G}}$ respectively, then $e(g, \hat{g})$ is a generator of $\mathbb{G}_T$. We use bilinear groups as a black box. If $\mathbb{G} = \hat{\mathbb{G}}$, the group is a symmetrical bilinear group. If $\mathbb{G} \neq \hat{\mathbb{G}}$, the group is a asymmetric bilinear group.

Now we are ready to describe our PIFS scheme. It works as follows.

SysInit. Let a receiver's identity be $ID_R \in \mathbb{Z}_p$. Let $\mathbb{G}, \mathbb{G}_T$ be two groups of order p, and let $\bar{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. Let $\bar{\mathbb{G}}$ be an Abelian group of order p in which the DDH problem [3] is hard. IM_R chooses random $g, h \xleftarrow{R} \mathbb{G}$ and random $\alpha \xleftarrow{R} \mathbb{Z}_p$. It sets $g_1 \leftarrow g^\alpha \in \mathbb{G}$. The program chooses random $g_2, g_3, t \xleftarrow{R} \bar{\mathbb{G}}$ and universal one-way hash functions H. The $MSK_R = \alpha$.

Let a sender's identity be $ID_S \in \mathbb{Z}_p$. Another pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \hat{\mathbb{G}}_T$ where the above three cyclic groups are of order p. The $IM_{GM_S}, IM_{OA_S}, IM_S$ secret keys are $MSK_{GM_S} = x_G, MSK_{OA_S} = x_O, MSK_S = x_S \in \mathbb{Z}_p^*$ and the public keys are $MPK_{GM_S} = g_G^{x_G}, MPK_{OA_S} = g_O^{x_O}, MPK_S = g_S^{x_S} \in \mathbb{G}_2$, where $g_G, g_O, g_S \in \mathbb{G}_2$. Let u be a generator in $\mathbb{G}_1$. Define hash functions $H_G : \{0,1\}^* \to \mathbb{Z}_p^*, H_O : \{0,1\}^* \to \mathbb{G}_1, H_S : \{0,1\}^* \to \mathbb{G}_1, \bar{H} : \{0,1\}^* \to \mathbb{Z}_p^*$.

For GM_S, define $\mathscr{R}_{GM_S} = \{((SK_{GM_S}, R), ID_{GM_S}) : g_G^{SK_{GM_S}} = R^{H_G(R \| ID_{GM_S})}_{MPK_{GM_S}}\}$. For OA_S, define $\mathscr{R}_{OA_S} = \{(SK_{OA_S}, ID_{OA_S}) : SK_{OA_S} = H_O(ID_{OA_S})^{x_O}\}$. For sender, define $\mathscr{R}_S = \{(x, i) : SK_{OA_S} = H_S(i)^{x_S}\}$. For certificate, define $\mathscr{F} = \{\langle \mathscr{R}_{C,i} \rangle : i\}$ with trapdoor x_i. And define the following:

$$\mathscr{R}_{C,ID_{GM_S}} = \{(ID_S, (A', e)) : A'^{e+SK_{GM_S}} H_S(ID_S) = u\}.$$

Let $g_4, g_5, g_6, g_7, g_8, u$ are generators in $\mathbb{G}_1$. Then the system parameter is as follows:

$$Pramas = (g, g_1, ..., g_8, u, h, t, \bar{e}, \hat{e}, g_G, g_O, g_S, MPK_{GM_S}, MPK_{OA_S}, MPK_S, H, \bar{H}, H_G, H_S, H_O, \mathscr{R}_{GM_S}, \mathscr{R}_{OA_S}, \mathscr{R}_S, \mathscr{F}).$$

ComSetup This procedure chooses random $x_1, x_2, y_1, y_2, z \xleftarrow{R} \mathbb{Z}_p$. Then it computes $w = g_2^{x_1} g_3^{x_2}, d = g_2^{y_1} g_3^{y_2}, l = g_2^z$. The receiver group public key and private key are $PK_{GM_R} = (g_2, g_3, w, d, l, H)$ and $SK_{GM_R} = (x_1, x_2, y_1, y_2, z)$. On input OA_S's identity ID_{OA_S}, the procedure uses $MSK_{OA_S} = x_O$ to compute OA_S secret key $SK_{OA_S} = H_O(ID_{OA_S})^{x_O}$. On input GM_S's identity ID_{GM_S}, the procedure defines $\mathscr{R}_{C,ID_{GM_S}} = \{(ID_S, (A', e)) : A'^{e+SK_{GM_S}} H_S(ID_S) = u\}$. Then it randomly chooses $r' \xleftarrow{R} \mathbb{Z}_p$ and computes the following:

$$aux_{ID_{GM_S}} = g_G^{r'}, SK_{GM_S} = r' + H_G(aux_{ID_{GM_S}} \parallel ID_{GM_S}) x_G.$$

UserJoin Let a receiver's identity be $ID_R \in \mathbb{Z}_p$. The procedure chooses random $r \xleftarrow{R} \mathbb{Z}_p$ and uses $MSK_R = \alpha$ to calculate the receiver's private key $SK_{ID_R} = (r, h_{ID_R})$, where $h_{ID_R} = (hg^{-r})^{1/(\alpha - ID_R)}$.

Let a sender's identity be $ID_S \in \mathbb{Z}_p$. The procedure uses $MSK_S = x_S$ to compute sender's private key $SK_{ID_S} = H_S(ID_S)^{x_S}$. And there is a pair of interactive protocols $(Join, Issu)$ between the sender and the procedure.

$Join$ runs a proof of knowledge of SK_{ID_S} for ID_S. $Issu$ uses SK_{GM_S}, aux_{GM_S} to compute $cert_{ID_S} = (A', e)$ satisfying $(ID_S, cert_{ID_S}) \in \mathscr{R}_{C,ID_S}$. $Issu$ chooses random $e \xleftarrow{R} \mathbb{Z}_p$ and computes $A' = (u/H_S(ID_S))^{1/(e+SK_{GM_S})}$. $Issu$ sends $(A', e, aux_{ID_{GM_S}})$ to $Join$. $Join$ accepts the certificate if and only if $\hat{e}(u, g_G) = \hat{e}(A', g_G)^e \hat{e}(A', S) \hat{e}(H_S(ID_S), g_G)$, where $S = g_G^{SK_{GM_S}} = aux_{ID_{GM_S}} \cdot MPK_{GM_S}^{H_G(aux_{ID_{GM_S}} \| ID_{GM_S})}$. $Join$ gets $cert_{ID_S}, aux_{GM_S}$. $Issu$ computes $W = \hat{e}(H_S(ID_S), g_G)$ and puts (ID_S, A', e, W) in the reg.

FileUpload This procedure can be divided into two subprocedures. They are encryption procedure and signature procedure.

Encryption procedure:

– Session key encryption. Given a session key $M \in \mathbb{G}_T$, the receiver's identity $ID_R \in \mathbb{Z}_p$, the procedure chooses random $s \xleftarrow{R} \mathbb{Z}_p$. Then it computes a part of ciphertext:

$$C_1 = (g_1^s g^{-s \cdot ID_R}, \bar{e}(g, g)^s, M \cdot \bar{e}(g, h)^{-s}) = (C_{10}, C_{11}, C_{12}).$$

– Receiver's identity encryption. Given receiver's identity $ID_R \in \mathbb{Z}_p$, the procedure chooses random $n \xleftarrow{R} \mathbb{Z}_p$. Then it computes the following:

$$k_1 = g_2^n, k_2 = g_3^n, \psi = l^n t^{ID_R}, \epsilon = H(k_1, k_2, \psi), v = w^n d^{n\epsilon}.$$

Another part of ciphertext is $C_2 = (k_1, k_2, \psi, v)$.

– We construct a zero-knowledge proof which can prove the encrypted receiver's identity and the identity that forms the IBE ciphertext are identical. It proves the IBE ciphertext has not been tampered as well as the the ciphertext is well formed. This is a non-interactive zero-knowledge proof protocol. We denote the protocol by

$$ZK\left\{s, n, ID_R \,\middle|\, \begin{array}{c} C_{10} = g_1^s g^{-s \cdot ID_R}, C_{11} = \bar{e}(g, g)^s, \\ k_1 = g_2^n, k_2 = g_3^n, \psi = l^n t^{ID_R}, v = w^n d^{n\epsilon} \end{array}\right\}$$

This zero-knowledge proof is difficult to constructed directly. We convert this zero-knowledge proof into an equivalent one as follows.

$$ZK\left\{s, n, ID_R \,\middle|\, \begin{array}{l} C_{10} = g_1^s g^{-s \cdot ID_R}, C_{11} = \bar{e}(g, g)^s, v = w^n d^{n\epsilon} \\ A = \psi^s, A = A_1 A_2, A_1 = l^{ns}, A_2^{-1} = t^{-s \cdot ID_R} \\ \psi = l^n t^{ID_R}, k = k_1^s, k = g_2^{ns}, k_1 = g_2^n, k_2 = g_3^n \end{array}\right\}$$

Sender randomly chooses integers $\bar{s}, \bar{ID}, \bar{n}$, and computes the following:

$$C\bar{}_{10} = g_1^{\bar{s}} g^{-\bar{s}\cdot \bar{ID}_R}, C\bar{}_{11} = \bar{e}(g,g)^{\bar{s}}, \bar{k_1} = g_2^{\bar{n}}, \bar{k_2} = g_3^{\bar{n}},$$
$$\bar{\psi} = l^{\bar{n}} t^{\bar{ID}_R}, \bar{v} = w^{\bar{n}} d^{\bar{n}\epsilon}, \bar{A} = \psi^{\bar{s}}, \bar{A} = \bar{A_1}\bar{A_2},$$
$$\bar{A_1} = l^{\bar{n}\bar{s}}, A_2\bar{}^{-1} = t^{-\bar{s}\cdot \bar{ID}_R}, \bar{k} = \bar{k_1^{\bar{s}}}, \bar{k} = g_2^{\bar{n}\bar{s}}$$

Then it sends these to a verifier. The procedure uses a hash function $\bar{H}$ to compute the following:

$$c = \bar{H}(\bar{C}_{10}, C_{10}, \bar{C}_{11}, C_{11}, \bar{k_1}, k_1, \bar{k_2}, k_2, \bar{\psi}, \psi,$$
$$\bar{v}, v, \bar{A}, A, \bar{A_1}, A_1, \bar{A_2}, A_2, A_2\bar{}^{-1}, A_2^{-1}, \bar{k}, k).$$

The sender computes $r_1 = \bar{s} + cs, r_2 = \bar{n} + cn, r_3 = \bar{ID}_R + c \cdot ID_R, r_4 = -\bar{s}\bar{ID}_R - c \cdot s \cdot ID_R, r_5 = \bar{n}\bar{s} + c \cdot ns$. Then it sends $r_1, r_2, r_3, r_4, r_5, c$ to a verifier. The verifier checks whether the equation holds the following:

$$c \stackrel{?}{=} \bar{H}(\psi^{r_1} A^{-c}, A, k_1^{r_1} k^{-c}, k, \bar{e}(g,g)^{r_1} C_{11}^{-c}, C_{11}, g_2^{r_2} k_1^{-c}, k_1,$$
$$g_3^{r_2} k_2^{-c}, k_2, (wd^{\epsilon})^{r_2} v^{-c}, v, l^{r_2} t^{r_3} \psi^{-c}, \psi, g_1^{r_1} g^{r_4} C_{10}^{-c},$$
$$C_{10}, t^{r_4} (A_2^{-1})^{-c}, A_2^{-1}, l^{r_5} A_1^{-c}, A_1, g_2^{r_5} k^{-c}, k).$$

The verifier outputs 1 if this equation holds; otherwise, it outputs 0. The ciphertext is $C = (C_1, C_2, C_3)$, where the $C_3 = (r_1, r_2, r_3, r_4, r_5, c)$.

– We use a symmetric encryption algorithm to encrypt file F with session key M. The algorithm outputs $E_M(F)$.

Signature procedure:

Given the sender's identity ID_S with private key SK_{ID_S} and certificate (A', e). The procedure computes a signature σ for ciphertext C, $E_M(F)$, and ID_{OA_S}. The procedure randomly selects $s_1, d' \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and computes $s_2 = es_1$. Then it computes the following:

$$t_0 = g_4^{s_1}, t_1 = SK_{ID_S} g_5^{s_1}, t_2 = H_S(ID_S) g_6^{s_1}, t_3 = A' g_7^{s_1}, t_5 = t_3^e g_8^{s_1}.$$

And we have the following:

$$ctxt = \hat{e}(H_S(ID_S), g_G)\hat{e}(H_O(ID_{OA_S}), MPK_{OA_S}),$$
$$U = g_O^{d'}, SK_{ID_S} = H_S(ID_S)^{x_S}, A'^{e+SK_{GM_S}} H_S(ID_S) = u,$$
$$S = g_G^{SK_{GM_S}} = aux_{ID_{GM_S}} \cdot MPK_{GM_S}^{H_G(aux_{ID_{GM_S}} \| ID_{GM_S})}.$$

– The procedure randomly selects $r_1', r_2', r_3', r_4' \in \mathbb{Z}_p$, R_1, R_2, $R_3 \in \mathbb{G}_1$ and computes the following:

$$\tau_0 = g_4^{r_1'}, \tau_1 = R_1 g_5^{r_1'}, \tau_2 = R_2 g_6^{r_1'}, \tau_3 = R_3 g_7^{r_1'},$$
$$\tau_4 = [\hat{e}(g_5, g_S)^{-1}\hat{e}(g_6, MPK_S)]^{r_1'}, \tau_5 = t_3^{r_3'} g_8^{r_1'},$$
$$\tau_6 = \hat{e}(g_7, g_G)^{r_2'}[\hat{e}(g_7, S)\hat{e}(g_6 g_8, g_G)]^{r_1'}, \tau_7 = g_4^{r_4'},$$
$$\tau_8 = \hat{e}(H_O(ID_{OA_S}), MPK_{OA_S})^{r_4'}\hat{e}(g_6', g_G)^{-r_1'}.$$

– Then the procedure uses a hash function $\bar{H}$ to compute the following:

$$c' = \bar{H}(t_0, ..., t_3, t_5, \tau_0, ..., \tau_8, aux_{GM_S}, ctxt, U, C, E_M(F)).$$

And the procedure computes the following:

$$z_0 = r_1' - c's_1, z_1 = R_1 SK_{ID_S}^{-c'}, z_2 = R_2 H_S(i)^{-c'}, z_3 = R_3 A'^{-c'},$$
$$z_4 = r_3' - c'e, z_5 = r_2' - c's_2, z_6 = r_4' - c'd.$$

– The signature is $\sigma = (t_0, ..., t_3, t_5, c', z_0, ..., z_6, aux_{GM_S}, ctxt, U)$.

Packaging procedure:

Finally, the package is $P = (C, \sigma, E_M(F))$, where (C, σ) is the header. The anonymous sender uploads the package.

FileAccess This procedure can be divided into signature verification procedure and decryption procedure.

Signature verification procedure:

Given the package $P = (C, \sigma, E_M(F))$, the procedure computes the following:

$$t_4 = \hat{e}(t_1, g_S)^{-1}\hat{e}(t_2, g_S)^{-1}, t_6 = \hat{e}(u, g_G)^{-1}\hat{e}(t_2 t_5, g_G)\hat{e}(t_3, S),$$
$$t_8 = ctxt \cdot \hat{e}(t_2, g_G)^{-1}, \tau_0 = g_4^{z_0} t_0^{c'}, \tau_1 = z_1 g_5^{z_0} t_1^{c'}, \tau_2 = z_2 g_6^{z_0} t_2^{c'},$$
$$\tau_3 = z_3 g_7^{z_0} t_3^{c'}, \tau_4 = [\hat{e}(g_5, g_S)^{-1}\hat{e}(g_6, MPK_S)]^{z_0} t_4^{c'},$$
$$\tau_5 = t_3^{z_4} g_8^{z_0} t_5^{c'}, \tau_6 = \hat{e}(g_7, g_G)^{z_5}[\hat{e}(g_7, S)\hat{e}(g_6 g_8, g_G)]^{z_0} t_6^{c'},$$
$$\tau_7 = g_G^{z_6} U^{c'}, \tau_8 = \hat{e}(H_O(ID_{OA_S}), MPK_{OA_S})^{z_6}\hat{e}(g_6', g_G)^{-z_0} t_8^{c'},$$
$$S = aux_{ID_{GM_S}} \cdot MPK_{GM_S}^{H_G(aux_{ID_{GM_S}} \| ID_{GM_S})}.$$

Then it computes $\hat{c}$ in the same way of c' and compares it to c' received in the signature. If they are equal, the procedure outputs 1 for valid signature, else outputs 0.

Decryption procedure:

If signature verification procedure outputs 1, execute the following steps, else return "reject."

Given $P = (C, \sigma, E_M(F)) = (C_1, C_2, \sigma, E_M(F))$, where $C_1 = (C_{10}, C_{11}, C_{12})$. The receiver's private key is $SK_{ID_R} = (r, h_{ID_R})$. Output the session key $M = C_{12} \cdot \bar{e}(C_{10}, h_{ID_R}) C_{11}^r$. Then the receiver uses the session key M to decrypt $E_M(F)$ and obtains file F.

PeerTracing This scheme can trace both receiver and sender if needs arise.

Receiver tracing:

- If ZK proof's verifier outputs 1, then the procedure executes the next step, else returns "reject."
- The procedure computes $t^{ID_R} = \psi / k_1^z$. For all $ID_{Rj} \in I$, compute $t^{ID_{Rj}}$ and test $t^{ID_{Rj}} \stackrel{?}{=} t^{ID}$. If $t^{ID_{Rj}} = t^{ID_R}$ holds, the procedure outputs receiver's identity ID_R, else returns "reject."

Sender tracing:

- $Open$ subprocedure. The sender open authority uses his secret key SK_{OA_S} to trace the sender's identity ID_S encrypted in the signature. Denote $Q_{OA_S} = H_O(ID_{OA_S})$. The procedure computes the following:

  $$m = \hat{e}(H_S(ID_S), g_G) = ctxt / \hat{e}(SK_{OA_S}, U).$$

  The open authority compares W with the registration table reg. If no such entry is found, returns "reject." If it is found to be sender ID_S, the sender open authority computes a proof of knowledge of SK_{OA_S} such that $\hat{e}(SK_{OA_S}, U) = ctxt/m$:

  1. Randomly picks $s_0' \stackrel{R}{\leftarrow} \mathbb{Z}_p$. Computes $t_0' = SK_{OA_S} h'^{s_0'}, t_1' = \hat{e}(h', U)^{s_0'}, t_2' = \hat{e}(h', g_O)^{s_0'}$.
  2. Randomly picks $r_0'', r_1'' \stackrel{R}{\leftarrow} \mathbb{Z}_p$. Computes $\tau_0' = Q_{OA_S}^{r_1''} h'^{r_0''}, \tau_1' = \hat{e}(h', U)^{r_0''}, \tau_2' = \hat{e}(h', g_O)^{r_0''}$.
  3. Computes $c'' = \bar{H}(t_0', t_1', t_2', \tau_0', \tau_1', \tau_2', ctxt, U, m)$.
  4. Computes $z_0' = r_0'' - c'' s_0', z_1' = Q_{OA_S}^{r_1''} SK_{OA_S}^{c''}$.

  Outputs the proof $\omega = (t_0', c'', z_0', z_1')$ to judge as follows.
- $Judge$ subprocedure. On input $ID_S, ID_{GM_S}, ID_{OA_S}, \sigma, \omega$, it computes the following:

  $$m = \hat{e}(H_S(ID_S), g_G), m' = ctxt/m, t_1' = \hat{e}(t_0', U)/m',$$
  $$t_2' = \hat{e}(t_0', g_O)\hat{e}(Q_{OA_S}, MPK_{OA_S}), \tau_0' = z_1' t_0'^{c''} h'^{z_0'},$$
  $$\tau_1' = \hat{e}(h', U)^{z_0'} t_1'^{c''}, \tau_2' = \hat{e}(h', g_0)^{z_0'} t_2'^{c''}.$$

  Then it compares if $c'' = \bar{H}(t_0', t_1', t_2', \tau_0', \tau_1', \tau_2', ctxt, U, m)$. If the equation holds, output 1, else output 0.

We show that the above scheme is correct. For the signature, the correctness is obvious. For the encryption, we first verify that the ciphertext can be decrypted correctly. $\bar{e}(C_{10}, h_{ID_R}) C_{11}^r = \bar{e}(g^{s(\alpha - ID_R)}, h^{1/(\alpha - ID_R)} g^{-r/(\alpha - ID_R)}) \bar{e}(g, g)^{sr} = \bar{e}(g, h)^s$. The receiver can decrypt because it possess an $(\alpha - ID_R)$-th root of h. When this is paired with an $(\alpha - ID_R)$-th root of g^s, the receiver obtains $\bar{e}(g, h)^s$. We then verify that the receiver can be traced correctly. Since $k_1 = g_2^n, k_2 = g_3^n$, we have $k_1^{x_1} k_2^{x_2} = g_2^{nx_1} g_3^{nx_2} = w^n$. Similarly, we have $k_1^{y_1} k_2^{y_2} = d^n$ and $k_1^z = l^n$. The equation $k_1^{x_1 + y_1 \epsilon} k_2^{x_2 + y_2 \epsilon} = v$ will hold. The output is $t^{ID_R} = \psi / l^n$.

Table 1 Storage complexity of our PIFS scheme

PK_{GM_R} size	5	SK_{GM_R} size	5
SK_{ID_R} size	2	MSK_R size	1
MSK_{GM_S} size	1	SK_{GM_S} size	1
MSK_{OA_S} size	1	SK_{OA_S} size	1
MSK_S size	1	SK_{ID_S} size	1
MPK_{GM_S} size	1	MPK_{OA_S} size	1
MPK_S size	1	Header size	29

3.3 Efficiency

In Tables 1 and 2, we denote τ_m as one multiplication operation time in $\mathbb{G}$ and $\mathbb{G}_T$, τ_e as one exponent operation time in $\mathbb{G}$ and $\mathbb{G}_T$, τ_p as one pairing operation time in $\mathbb{G}$ and $\mathbb{G}_T$, $\bar{\tau}_m$ as one multiplication operation time in $\bar{\mathbb{G}}$, $\bar{\tau}_e$ as one exponent operation time in $\bar{\mathbb{G}}$, $\bar{\tau}_p$ as one pairing operation time in $\bar{\mathbb{G}}$, $\hat{\tau}_m$ as one multiplication operation time in $\mathbb{G}_1, \mathbb{G}_2$, and $\hat{\mathbb{G}}_T$, $\hat{\tau}_e$ as one exponent operation time in $\mathbb{G}_1, \mathbb{G}_2$, and $\hat{\mathbb{G}}_T$, and $\hat{\tau}_p$ as one pairing operation time in $\mathbb{G}_1, \mathbb{G}_2$, and $\hat{\mathbb{G}}_T$. A number of pre-computations can be done to improve the efficiency of our PIFS scheme.

The storage complexity and computational complexity of our scheme are constant and unrelated to the number of users.

4 Security analysis

4.1 Formal security definition

File confidentiality and receiver privacy are protected by applying an identity-based encryption [9] scheme with ANO-IND-ID-CPA security and a public-key encryption [3] with CCA2 security. If a file cannot reveal information of the message, we say that the file is semantic secure. If a file cannot reveal information of the identity of the receiver, we say that the file receiver is anonymous. We consider the combination of these two definitions: receiver anonymity

Table 2 Computational complexity of our PIFS scheme

SysInit time	$3\tau_e + 3\bar{\tau}_e + 13\hat{\tau}_e$
ComSetup time	$5\bar{\tau}_e + 2\bar{\tau}_m + \hat{\tau}_e$
UserJoin time	$2\tau_e + \tau_m + 4\hat{\tau}_p + 2\hat{\tau}_e + \hat{\tau}_m$
FileUpload time	$12\tau_e + 5\tau_m + 61\hat{\tau}_e + 28\hat{\tau}_m$
FileAccess time	$9\hat{\tau}_p + 24\hat{\tau}_e + 21\hat{\tau}_m + \tau_p + \tau_e + 2\tau_m$
Receiver tracing time	$\bar{\tau}_e$
Sender tracing time	$13\hat{\tau}_e + 12\hat{\tau}_m + 3\hat{\tau}_p$

and semantic security. This definition will ensure file confidentiality and receiver privacy for our PIFS scheme in smart city. Receiver traceability is also a necessary definition for our scheme.

File integrity and sender privacy are protected by applying an identity-based group signature [25]. An identity-based group signature scheme implies message integrity and anonymous sender. Since integrity is inherent, we propose sender anonymity to ensure the sender privacy for our PIFS scheme. Likewise, sender traceability is required.

Formally, security definitions are defined through games between an adversary $\mathscr{A}$ and a challenger as follows.

Receiver anonymity and semantic security We have the following game for receiver anonymity and semantic security:

- **Setup.** The challenger builds the system. It takes security parameter λ as input and runs the algorithm **SysInit** which outputs system parameter $Params$ and MSK_R. It gives the adversary $Params$ but keeps MSK_R.
- **Phase 1.** The adversary can adaptively issue extraction query of $\langle ID_{Rj} \rangle$. Then it obtains the receiver's private key $SK_{ID_{Rj}}$.
- **Challenge.** After phase 1, adversary chooses two receiver identities ID_{R0}, ID_{R1} and two equal length plaintexts M_0, M_1. The only restriction is that the two identities did not appear in any private key extraction query in phase 1. Challenger chooses a random bit $b \in \{0, 1\}$ and a random bit $c \in \{0, 1\}$. It computes the package P using algorithm **FileUpload** and (ID_{Rb}, M_c). Then it sends P to adversary.
- **Phase 2.** It is similar to phase 1. The only constraint is $ID_{Ri} \neq ID_R$.
- **Guess.** The adversary outputs $b' \in \{0, 1\}$ and $c' \in \{0, 1\}$. The adversary wins the game if $b = b' \wedge c = c'$.

We define adversary $\mathscr{A}$'s advantage with security parameter λ in ANO-IND-CIA-CPA game as follows:

$$Adv_{\mathscr{A}}(\lambda) = \mid Pr[b = b' \wedge c = c'] - \frac{1}{4} \mid .$$

Definition 1 We say that our PIFS scheme has receiver anonymity and semantic security against chosen-identity attacks and chosen-plaintext attacks (ANO-IND-CIA-CPA) if no polynomially bounded adversary $\mathscr{A}$ has non-negligible advantage in the above game.

Receiver traceability We have the following game for receiver traceability:

- **Setup.** The challenger builds the system. It takes security parameter λ as input and runs the algorithm **SysInit** which outputs system parameter $Params$ and MSK_R. It gives the adversary $Params$ but keeps MSK_R.
- **Inspect phase.** The adversary can adaptively issue community setup query, user join query, file upload query, file access query and even colludes with the prover in the zero-knowledge proof.
- **Output.** The adversary outputs a valid package P^*. The adversary wins in the game if the receiver group manager outputs a wrong identity of the receiver. The adversary's advantage is its probability of winning.

Definition 2 We say our PIFS scheme is receiver traceable if no polynomially bounded adversary has non-negligible probability to win in the above game.

Sender anonymity We have the following oracles for the adversary to query:

- The Random Oracle $\mathscr{RO}$. Simulate the random oracle.
- The Key Extraction Oracle-GM_S $\mathscr{KEO}_{\mathscr{G}}$. Upon input ID_{GM_S}, outputs his secret key SK_{GM_S}.
- The Key Extraction Oracle-OA_S $\mathscr{KEO}_{\mathscr{O}}$. Upon input ID_{OA_S}, outputs his secret key SK_{OA_S}.
- The Key Extraction Oracle-Sender $\mathscr{KEO}_{\mathscr{S}}$. Upon input ID_S, outputs his secret key SK_{ID_S}.
- The Join Oracle $\mathscr{JO}$. Upon input ID_S of ID_{GM_S}, outputs $cert$ corresponding to an honest $Issu$-executing GM_S.
- The Issue Oracle $\mathscr{IO}$. Upon input ID_S of ID_{GM_S}, outputs $cert$ corresponding to an honest $Join$-executing sender.
- The Corruption Oracle $\mathscr{CO}$. Upon input ID_S of group ID_{GM_S}, outputs the secret keys $(SK_{ID_S}, aux_{ID_S}, cert)$.
- The Signing Oracle $\mathscr{SO}$. Upon input ID_S, ID_{GM_S}, ID_{OA_S} and ciphertext C, outputs a valid signature.
- The Open Oracle $\mathscr{OO}$. Upon input a valid signature σ for C under ID_{GM_S}, ID_{OA_S}, outputs the sender ID_S and the proof ω.

We have the following game for sender anonymity:

- **Setup.** The challenger builds the system. It takes security parameter λ as input and runs the algorithm **SysInit**. Then it invokes **UserJoin** q_u times to generate a set of honest senders (HS) with secret keys and certificates.
- **Phase 1.** Adversary adaptively queries $\mathscr{RO}, \mathscr{CO}, \mathscr{OO}$, $\mathscr{JO}, \mathscr{KEO}_{\mathscr{G}}, \mathscr{KEO}_{\mathscr{O}}, \mathscr{KEO}_{\mathscr{S}}$.
- **Challenge.** After phase 1, adversary chooses two sender identities $ID_{S0}, ID_{S1} \in$HS, ID_{GM_S}, ID_{OA_S}, and a package P. The only restriction is that ID_{OA_S} should not be input to $\mathscr{OO}, \mathscr{KEO}_{\mathscr{O}}$ before. Challenger chooses a random bit $\bar{b} \in \{0, 1\}$. It computes the signature $\sigma = \mathscr{SO}(ID_{S\bar{b}}, ID_{GM_S}, ID_{OA_S}, P)$ and sends the signature to adversary.

- **Phase 2.** It is similar to phase 1. The only restriction is that ID_{OA_S} should not be input to $\mathcal{OO}$, $\mathcal{KEO}_{\mathcal{O}}$.
- **Guess.** The adversary also has write access to registration table reg. It outputs its guess $\bar{b}' \in \{0, 1\}$. The adversary wins the game if $\bar{b} = \bar{b}'$.

We define adversary $\mathcal{A}$'s advantage with security parameter λ in sender anonymity game as follows:

$$Adv_{\mathcal{A}}(\lambda) = \mid Pr[\bar{b} = \bar{b}'] - \frac{1}{2} \mid .$$

Definition 3 We say our PIFS scheme has sender anonymity if no polynomially bounded adversary $\mathcal{A}$ has non-negligible advantage in the above game.

Sender traceability We have the following game for sender traceability:

- **Setup.** The challenger builds the system. It takes security parameter λ as input and runs the algorithm **SysInit**. Then it invokes **UserJoin** q_u times to generate a set of honest senders (HS) with secret keys and certificates.
- **Inspect phase.** Adversary adaptively queries $\mathcal{RO}, \mathcal{CO}, \mathcal{JO}, \mathcal{KEO}_{\mathcal{G}}, \mathcal{KEO}_{\mathcal{O}}, \mathcal{KEO}_{\mathcal{J}}$.
- **Output.** The adversary also has read access to reg. It outputs a valid signature σ. The adversary wins in the game if **PeerTracing**$(ID_{GM_S}), ID_{OA_S}, C, \sigma) = 1$, either $i = \perp$ or $Judge(ID_{GM_S}, ID_{OA_S}, i, m, \sigma, \omega) = 0$, where $(i, \omega) \leftarrow Open(ID_{GM_S}, MSK_{OA_S}, reg, C, \sigma)$. ID_{GM_S} has never been queried to $\mathcal{KEO}_{\mathcal{G}}$, and (i, ID_{GM_S}) has never been queried to $\mathcal{CO}$. The adversary's advantage is its probability of winning.

Definition 4 We say our PIFS scheme is sender traceable if no polynomially bounded adversary has non-negligible probability to win in the above game.

4.2 Complexity assumptions

Receiver anonymity and semantic security of our PIFS scheme rely on decisional augmented bilinear Diffie-Hellman exponent (decisional ABDHE) problem [8]. Let $\mathbb{G}, \mathbb{G}_T$ are two (multiplicative) cyclic groups of prime order p. g is a generator of $\mathbb{G}$. $\bar{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear map.

First, we review the q-BDHE problem: given a vector of $2q + 1$ elements

$$(g', g, g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q}, g^{\alpha^{q+2}}, ..., g^{\alpha^{2q}}) \in \mathbb{G}^{2q+1}$$

as input, output $\bar{e}(g, g')^{\alpha^{q+1}} \in \mathbb{G}_T$. Since the term $g^{\alpha^{q+1}}$ is missing in the input, it is intractable to compute $\bar{e}(g, g')^{\alpha^{q+1}}$.

The definition of the q-ABDHE problem is almost identical: given a vector of $2q + 2$ elements

$$(g', g'^{\alpha^{q+2}}, g, g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q}, g^{\alpha^{q+2}}, ..., g^{\alpha^{2q}}) \in \mathbb{G}^{2q+2}$$

as input, output $\bar{e}(g, g')^{\alpha^{q+1}} \in \mathbb{G}_T$. Since the term $g^{\alpha^{-1}}$ is missing in the input, it is intractable to compute $\bar{e}(g, g')^{\alpha^{q+1}}$, even though the term $g'^{\alpha^{q+2}}$ is added.

We will use a truncated version of the q-ABDHE problem, in which the terms $(g^{\alpha^{q+2}}, ..., g^{\alpha^{2q}})$ are omitted from the input, because of this version of q-ABDHE problem is more useful for our concrete IBGE scheme.

The truncated q-ABDHE problem: given a vector of q elements

$$(g', g'^{\alpha^{q+2}}, g, g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q}) \in \mathbb{G}^q$$

as input, output $\bar{e}(g, g')^{\alpha^{q+1}} \in \mathbb{G}_T$. The truncated q-ABDHE problem is hard if the q-ABDHE problem is hard, since the input vector of truncated q-ABDHE is less than q-ABDHE. $\mathcal{A}$ has advantage ϵ in solving truncated q-ABDHE if

$$Pr[\mathcal{A}(g', g'^{\alpha^{q+2}}, g, g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q}) = \bar{e}(g^{\alpha^{q+1}}, g')] \geqslant \epsilon$$

where the probability is over the randomly chosen $g, g' \overset{R}{\leftarrow} \mathbb{G}$, the randomly chosen $\alpha \overset{R}{\leftarrow} \mathbb{Z}_p$ and the randomly chosen bits by $\mathcal{A}$.

For ease of description, we use g_i and g'_i to denote g^{α^i} and g'^{α^i}. Now, it is easy to define the decisional version of truncated q-ABDHE. An algorithm $\mathcal{B}$ that outputs $b \in \{0, 1\}$ has advantage ϵ in solving truncated decision q-ABDHE if

$$\mid Pr[\mathcal{B}(g', g'_{q+2}, g, g_1, ..., g_q, \bar{e}(g_{q+1}, g')) = 0] - Pr[\mathcal{B}(g', g'_{q+2}, g, g_1, ..., g_q, Z) = 0] \mid \geqslant \epsilon$$

where the probability is over the randomly chosen $g, g' \overset{R}{\leftarrow} \mathbb{G}$, the randomly chosen $\alpha \overset{R}{\leftarrow} \mathbb{Z}_p$, the randomly chosen $Z \overset{R}{\leftarrow} \mathbb{G}_T$, and the randomly chosen bits of $\mathcal{B}$. We refer to the distribution on the left as P_{ABDHE} and the distribution on the right as R_{ABDHE}.

Definition 5 We say that the decisional version of truncated (t, ϵ, q)-ABDHE assumption holds in $\mathbb{G}$ if no t-time algorithm has advantage at least ϵ in solving the decisional version of truncated q-ABDHE problem in $\mathbb{G}$.

Sender anonymity of our PIFS scheme relies on coDBDH problem and Lockstep DDH+coDBDH problem. Let $\mathbb{G}_1, \mathbb{G}_2$ are two (multiplicative) cyclic groups of prime order p. g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$. Let ψ is a computable isomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$, with $\psi(g_2) = g_1$. $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a bilinear map.

Definition 6 The co-decisional Bilinear Diffie-Hellman problem (coDBDH problem) in $(\mathbb{G}_1, \mathbb{G}_2)$ is as follows: given $P, P^{\alpha}, P^{\beta} \in \mathbb{G}_1, Q \in \mathbb{G}_2, R \in \mathbb{G}_T$ for unknown $\alpha, \beta \in \mathbb{Z}_p$ to decide if $R = \hat{e}(P, Q)^{\alpha\beta}$.

Definition 7 Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a bilinear map. Given:

1. $g_1, g_1^{\alpha}, g_1^{\beta_i}, g_1^{\gamma_i}, \in \mathbb{G}_1$ for $1 \leqslant i \leqslant k$.
2. $g_2, g_2^{\delta_1}, g_2^{\delta_2}, R \in \mathbb{G}_T$.
3. $Pr\{\gamma_i = \alpha\beta_i, all\ i, 1 \leqslant i \leqslant k\ AND\ R = \hat{e}(g_1, g_2)^{\delta_1,\delta_2}\} = Pr\{\gamma_i \neq \alpha\beta_i, alli, 1 \leqslant i \leqslant k\ AND\ R \neq \hat{e}(g_1, g_2)^{\delta_1,\delta_2}\} = 1/2$.

The Lockstep DDH+coDBDH Problem to distinguish between the two non-zero probability events in (3) above with non-negligible probability over 1/2. The Lockstep DDH+coDBDH Assumption is that no polynomial time algorithm can solve the Lockstep DDH+coDBDH Problem. The proof can be seen in [25].

Sender traceability of our PIFS scheme relies on k-CAA2 problem.

Definition 8 The k-CAA2 problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is as follows: given $u, v \in \mathbb{G}_1, g_2, g_2^{\gamma} \in \mathbb{G}_2$ and pair (A_i, e_i, λ_i) with distinct and non-zero $e_i's$ satisfying $A_i^{\gamma+e_i} v^{\lambda_i} = u$ for $1 \leqslant i \leqslant k$ as input, outputs a pair $(A_{k+1}, e_{k+1}, \lambda_{k+1})$ satisfying $A_{k+1}^{\lambda+e_{k+1}} \cdot v^{\lambda_{k+1}} = u$, with $e_{k+1} \neq e_i$ for all $1 \leqslant i \leqslant k$.

4.3 Formal security results

We give the formal security results of our PIFS scheme according to the security definitions.

Theorem 1 *Our PIFS scheme satisfies $(t', \varepsilon', q_{ID})$ ANO-IND-CIA-CPA receiver anonymity and semantic security assuming the truncated decision $(t, \epsilon, q) - ABDHE$ assumption holds for $(\mathbb{G}, \mathbb{G}_T, \bar{e})$, where $q = q_{ID_R} + 1$, $t' = t - O(t_{exp} \cdot q^2)$, $\epsilon' = \epsilon + 2/q$, t_{exp} is the time required to exponentiate in $\mathbb{G}$.*

Proof Suppose $\mathscr{A}$ is an $(t', \varepsilon', q_{ID})$ ANO-IND-CIA-CPA adversary against our scheme. We construct a simulator $\mathscr{B}$ solves the truncated decision q-ABDHE problem. $\mathscr{B}$ takes as input $(g', g'_{q+2}, g, g_1, ...g_q, Z)$, where $Z = \bar{e}(g_{q+1}, g')$ or a random element of $\mathbb{G}_T$.

Setup. $\mathscr{B}$ generates a random polynomial $f(x) \in \mathbb{Z}_p[x]$ of degree q. It let $h = g^{f(\alpha)}$ and computes h from $(g, g_1, ..., g_q)$. It sends public parameters (g, g_1, h) to $\mathscr{A}$.

Phase 1. $\mathscr{A}$ adaptively issues receiver identity extraction query. $\mathscr{B}$ responds as follows. If $ID_R = \alpha$, $\mathscr{B}$ can solve the truncated decision q-ABDHE immediately. Otherwise, let $F_{ID_R}(x) = (f(x) - f(ID_R))/(x - ID_R)$ be the $(q-1)$-degree polynomial. $\mathscr{B}$ let $(f(ID_R), g^{F_{ID_R}(\alpha)})$ be the user's secret key (r, h_{ID_R}). Since $g^{F_{ID_R}(\alpha)} = g^{(f(\alpha)-f(ID_R)/(\alpha-ID_R))} = (hg^{-f(ID_R)})^{1/(\alpha-ID_R)}$, secret key (r, h_{ID_R}) is valid of ID_R.

Challenge. $\mathscr{A}$ outputs two identities ID_{R0}, ID_{R1} and two M_0, M_1. The restriction is that the two identities did not appear in any secret key extraction query. Note that if $\alpha \in \{ID_{R0}, ID_{R1}\}$, $\mathscr{B}$ can solve the truncated decision q-ABDHE immediately. Otherwise, $\mathscr{B}$ chooses bits $b, c \in \{0, 1\}$ and computes secret key $(r_b, h_{ID_{Rb}})$ for ID_b same to phase 1.

Let $f_2(x) = x^{q+2}$ and let $F_{2,ID_{b(x)}} = (f_2(x) - f_2(ID_{Rb}))/(x - ID_{Rb})$, which is a polynomial of degree of $q+1$. $\mathscr{B}$ sets

$$C_{10} = g'^{f_2(\alpha)-f_2(ID_{Rb})}, C_{11} = Z \cdot \bar{e}(g', \prod_{i=0}^{q} g^{F_{2,ID_{Rb},i}\alpha^i}),$$

$$C_{12} = M_c/\bar{e}(C_{10}, h_{ID_{Rb}})C_{11}^{r_b}$$

where $F_{2,ID_{Rb},i}$ is the coefficient of x^i in $F_{2,ID_{Rb}}(x)$. It sends $C_1 = (C_{10}, C_{11}, C_{12})$ as the ciphertext to be challenged.

Let $s = (\log_g g')F_{2,ID_{Rb}}(\alpha)$. If $Z = \bar{e}(g_{q+1}, g')$, then $C_{10} = g^{s(\alpha-ID_{Rb})}$, $C_{11} = \bar{e}(g, g)^s$, and $M_c/C_{12} = \bar{e}(C_{10}, h_{ID_{Rb}})C_{11}^{r_b} = \bar{e}(g, h)^s$. Let $C_1 = (C_{10}, C_{11}, C_{12})$ be an effective ciphertext of identity ID_{Rb} and message M_c under random value s.

Phase 2. $\mathscr{A}$ adaptively issues receiver identity extraction query as in phase 1. The restriction is that the two identities did not appear in any identity extraction query.

Guess. Finally, adversary $\mathscr{A}$ outputs guesses $b', c' \in \{0, 1\}$ of b, c. If $b' = b \wedge c' = c$, $\mathscr{B}$ outputs 1 else 0. □

The analysis of probability and time complexity is as follows.

Analysis of probability. If $Z = \bar{e}(g_{q+1}, g')$, the simulation is perfect. Adversary $\mathscr{A}$ can guess the bits (b, c) correctly with probability $\frac{1}{4} + \epsilon'$. Otherwise, Z is uniformly random, so (C_{10}, C_{11}) is a uniformly random and independent element of $(\mathbb{G}, \mathbb{G}_T)$. When this happens, the inequalities

$$C_{11} \neq \bar{e}(C_{10}, g)^{1/(\alpha-ID_{R0})}, C_{11} \neq \bar{e}(C_{10}, g)^{1/(\alpha-ID_{R1})}$$

both hold in the same time with probability $1 - 2/p$. When the two inequalities hold,

$$\bar{e}(C_{10}, h_{ID_{Rb}})C_{11}^{r_b} = \bar{e}(C_{10}, (hg^{-r_b})^{1/(\alpha-ID_{Rb})})C_{11}^{r_b}$$
$$= \bar{e}(C_{10}, h)^{\alpha-ID_{Rb}}(C_{11}/\bar{e}(C_{10}, g)^{1/(\alpha-ID_{Rb})})^{r_b}$$

is a uniformly random and independent value from the view of adversary $\mathscr{A}$, because r_b is a uniformly random and independent value from the view of adversary $\mathscr{A}$. So, C_{12} is uniformly random and independent. C_1 will not reveal any information of the bits (b, c). Assuming that no queried identity equals α, it is easy to see that $| Pr[\mathscr{B}(g', g'_{q+2}, g, g_1, ..., g_q, Z) = 0] - \frac{1}{4} | \leqslant \frac{2}{p}$ when $(g', g'_{q+2}, g, g_1, ..., g_q, Z)$ is sampled

from R_{ABDHE}. To the contrary, we can see that $|\ Pr[\mathscr{B}(g', g'_{q+2}, g, g_1, ..., g_q, Z) = 0] - \frac{1}{4}\ | \geqslant \epsilon'$ when $(g', g'_{q+2}, g, g_1, ..., g_q, Z)$ is sampled from P_{ABDHE}. Thus, we have that

$$|\ Pr[\mathscr{B}(g', g'_{q+2}, g, g_1, ..., g_q, \bar{e}(g_{q+1}, g')) = 0]$$
$$-Pr[\mathscr{B}(g', g'_{q+2}, g, g_1, ..., g_q, Z) = 0]\ | \geqslant \epsilon' - \frac{2}{p}.$$

Analysis of time complexity. In the simulation procedure, the overhead of $\mathscr{B}$ is computing $g^{F_{ID_R}(\alpha)}$ in order to response $\mathscr{A}$'s extraction query for the ID_R, where $F_{ID_R}(x)$ is polynomial of $q-1$ degree. Every computation requires $O(q)$ exponentiation in $\mathbb{G}$. $\mathscr{A}$ makes at most $q-1$ queries, thus, $t = t' + O(t_{exp} \cdot q^2)$.

Theorem 2 *Our PIFS scheme satisfies receiver traceability.*

Proof Setup is same as the above proof. In inspect phase adversary can adaptability issue queries. The challenger will respond adversary. The adversary will choose a receiver group public key $PK'_{GM_R} = (g_2, g_3, w', d', l', H)$ and obtain secret key are $SK'_{GM_R} = (x'_1, x'_2, y'_1, y'_2, z')$. Adversary will choose a receiver identity ID_R and obtain his private key SK_{ID_R}, as well as an other ID'_R. Adversary computes C'_1 using ID_R and computes C'_2 using ID'.

Let $C_{10} = g_1^s g^{-s \cdot ID_R}, A_2^{-1} = t^{-s \cdot ID'_R}, ID_R \neq ID'_R$. Prover chooses $-\bar{s} \cdot I\bar{D}_{R1}, -\bar{s} \cdot I\bar{D}_{R2}, I\bar{D}_{R1} \neq I\bar{D}_{R2}$ (if $I\bar{D}_{R1} = I\bar{D}_{R2}$, since $g_1^{r_1} g^{r_4} = \bar{C_{10}} C_{10}^c, t^{r_4} = \bar{A_2^{-1}}(A_2^{-1})^c$, we obtain $r_4 \equiv -\bar{s} I\bar{D}_1 + (-s \cdot ID_R)c \bmod p, r_4 \equiv -\bar{s} I\bar{D}_{R2} + (-s \cdot ID'_R)c \bmod p, ID_R = ID'_R$), then computes $\bar{C}_{10} = g_1^{\bar{s}} g^{-\bar{s} \cdot I\bar{D}_{R1}}, \bar{A}_2^{-1} = t^{-\bar{s} \cdot I\bar{D}_{R2}}$. $g_1^{r_1} g^{r_4} = \bar{C_{10}} C_{10}^c$ and $t^{r_4} = \bar{A_2^{-1}}(A_2^{-1})^c$ both hold, if and only if $-\bar{s} I\bar{D}_{R1} + (-s \cdot ID_R)c \equiv -\bar{s} I\bar{D}_{R2} + (-s \cdot ID'_R)c \bmod p$ holds. This means that $c \equiv \frac{\bar{s}(I\bar{D}_{R1} - I\bar{D}_{R2})}{s(ID'_R - ID_R)}$. This equation holds if and only if the verifier chooses this c exactly. We get $C'_3 = (r_1, r_2, r_3, r_4, r_5, c)$ and a valid ciphertext $C' = (C'_1, C'_2, C'_3)$. Thus, the adversary gets a valid file P^* which the GM_R cannot trace correctly. But the probability is negligible. □

Theorem 3 *Our PIFS scheme is sender anonymous if and only if the DDH assumption in* $\mathbb{G}_1$ *and the coDBDH assumption in* $(\mathbb{G}_1, \mathbb{G}_2)$ *both hold.*

Proof Suppose $\mathscr{A}$ is a polynomial time algorithm that breaks the sender anonymity of our scheme. Then we show how to construct a polynomial time algorithm $\mathscr{S}$ that solves the Lockstep DDH+coDBDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$, which is equivalent to the coDBDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ and the DDH problem in $\mathbb{G}_1$.

$\mathscr{S}$ is given $g'_1, g'^{\alpha}_1, g'^{\beta_i}_1, g'^{\gamma_i}_1 \in \mathbb{G}_1$ for $1 \leqslant i \leqslant 4$; $g'_2, g'^{\delta_i}_2, g'^{\delta_i}_2 \in \mathbb{G}_1$ and $R \in \mathbb{G}_T$ for unknown $\alpha_i, \beta_i, \delta_1, \delta_2 \in \mathbb{Z}_p$. $\mathscr{S}$ sets the public parameter $g_O = g'_2, MSK_{OA_S} = g'^{\delta_1}_2, g_4 = g'_1, g_5 = g'^{\beta_1}_1, g_6 = g'^{\beta_2}_1, g_7 = g'^{\beta_3}_1, g_8 = g'^{\beta_4}_1$. $\mathscr{S}$ generates $g_G, x_G, MPK_{GM_S} = g_G^{x_G}, g_S, x_S, MPK_S = g_S^{x_S}$ and $u = g_G$. $\mathscr{S}$ randomly picks $\ell \in \{1, ..., q_H\}$, where q_H is the number of query to H_O. $\mathscr{S}$ provides $\mathscr{A}$ the parameters.

The oracles are simulated as follows:

- H is random oracle.
- $H_G(aux_i \parallel i)$: On input new aux_i, i, randomly pick $\lambda \in \mathbb{Z}_p$ and return λ. Store (aux_i, i, λ) in tape $\mathscr{L}_G$.
- $H_S(i)$: On input new i, randomly pick $\lambda \in \mathbb{Z}_p$ and return g_S^{λ}. Store (i, λ) in tape $\mathscr{L}_S$.
- $H_O(i)$: On input new i, randomly pick $\lambda \in \mathbb{Z}_p$ and return g_O^{λ}. Store (i, λ) in tape $\mathscr{L}_O$. For the ℓ-th query, return $Q = g'_1$ and back patch (i, Q) in $\mathscr{L}_O$. Denote this identity as i_g.
- $\mathscr{KEO}_{\mathscr{S}}(i)$: Computes $H_S(i)$. Then $SK_{ID_{Si}} = MPK_S^{\lambda}$, where $(i, \lambda) \in \mathscr{L}_S$.
- $\mathscr{KEO}_{\mathscr{G}}(ID_{GM_S})$: On input ID_{GM_S}, randomly pick $h, SK_{GM_S} \in \mathbb{Z}_p$ and computes $aux_{GM_S} = g_G^{SK_{GM_S}} MPK_{GM_S}^{-h}$. $\mathscr{S}$ back patches $H_G(aux_{GM_S} \parallel ID_{GM_S}) = h$. Store $(aux_{GM_S}, ID_{GM_S}, h)$ in tape $\mathscr{L}_O$. Return (SK_{GM_S}, aux_{GM_S}).
- $\mathscr{KEO}_{\mathscr{G}}(ID_{OA_S})$: Computes $H_O(ID_{OA_S})$. Then $SK_{OA_S} = MPK_{OA_S}^{\lambda}$, where $(ID_{OA_S}, \lambda) \in \mathscr{L}_O$. If $ID_{OA_S} = i_g$, declare failure and exit.
- $\mathscr{IO}(i, ID_{GM_S})$: It interacts with the honest sender i. Computes (SK_{GM_S}, aux_{GM_S}) as in $\mathscr{KEO}_{\mathscr{G}}(ID_{GM_S})$. Randomly selects $e \in \mathbb{Z}_p$, and computes

$$ID_{OA_S} = i_g, A' = (u/H_S(i))^{1/(e+SK_{GM_S})}, W = \hat{e}(H_S(i)), g_G).$$

  Store (i, A', e, W) in reg. Returns (A', e, aux_{GM_S}) to honest sender i.
- $\mathscr{CO}(i, ID_{GM_S})$: On input the identity, this oracle outputs the sender's secret keys. Computes $H_1(ID_{OA_S})$. Computes $H_S(i)$ as in $\mathscr{KEO}_{\mathscr{S}}(i)$. Computes $cert_i$ as in $\mathscr{IO}(i, ID_{GM_S})$. Returns $(ID_{Si}, cert_i)$.
- $\mathscr{OO}(ID_{GM_S}, ID_{OA_S}, m, \sigma)$: Computes $H_1(ID_{OA_S})$. Then $(SK_{OA_S}, \lambda) \in \mathscr{L}_1$. Return$(i, \omega) \leftarrow Open(ID_{GM_S}, SK_{OA_S}, reg, m, \sigma)$. If $ID_{OA_S} = i_g$, declare failure and exit.

Anytime $\mathscr{A}$ can query the oracles above. At some point, it sends the sender identity $i_0, i_1, ID_{GM_S}, ID_{OA_S}$ and C to the $\mathscr{S}$. $\mathscr{S}$ flips a coin $\bar{b} \in \{0, 1\}$ and computes $(SK_{ID_{S\bar{b}}}, A'_{\bar{b}}, e_{\bar{b}}) \leftarrow \mathscr{CO}(i_{\bar{b}}, ID_{GM_S}$. $\mathscr{S}$ sets $t_0 = g'^{\alpha}_1, t_1 = SK_{ID_{S\bar{b}}} g'^{\gamma_1}_1, t_2 = H_S(i_{\bar{b}}) g'^{\gamma_2}_1, t_3 = A'_{\bar{b}} g'^{\gamma_3}_1 m, t_5 = t_3^{e_{\bar{b}}} g'^{\gamma_4}_1$. $\mathscr{S}$ randomly chooses a c' and $z_0, ..., z_6$. It computes $\tau_0, ..., \tau_8$. It sets $U = g'^{\delta_2}_2$ and computes $ctxt =$

$\hat{e}(H_S(i_{\bar{b}}), g_G)R$. Then back patch c' to H. $\mathscr{S}$ returns signature σ_g as the gauntlet to $\mathscr{A}$.

Finally, $\mathscr{A}$ outputs a bit $\bar{b}'$. If $\bar{b}' = \bar{b}$, $\mathscr{S}$ returns "yes" for the Lockstep DDH+coDBDH problem. Otherwise, $\mathscr{S}$ returns "no." By the back patch above, if $\mathscr{A}$ has a non-negligible advantage ε in winning the game, $\mathscr{S}$ has advantage ε/q_H in solving the Lockstep DDH+coDBDH problem. □

Theorem 4 *Our PIFS scheme is sender traceable if and only if the k-CAA2 assumption holds.*

Proof Let $\mathscr{A}$ be a polynomial time adversary attacking the sender traceability. We show that given a colluding group of k senders, with the knowledge of the opening key and access to some oracles, we can use $\mathscr{A}$ to solve the k-CAA2 problem.

$\mathscr{S}$ is given the tuple $u, v \in \mathbb{G}, g_2, g_2^{\gamma} \in \mathbb{G}_2$ and pair (A_i', e_i, λ_i) with distinct and non-zero e_i's satisfying $A_i'^{\lambda+e_i} v^{\lambda_i} = u$ for $1 \leqslant i \leqslant k$ as input. The value $s = \log_u(v)$ is also given to it.

$\mathscr{S}$ sets $g_G = g_6, g_S = v$ and g_O. It randomly selects $x_G, MPK_{GM_S} = g_G^{x_G}, x_S, MPK_S = g_S^{x_S}, x_O, MPK_{OA_S} = g_O^{x_O}$. It randomly selects μ and sets $g_7 = v^{\mu}$. It sets up the rest of the parameters and provides to $\mathscr{A}$. It randomly selects $\ell \in \{1, ..., q_c\}$, where q_c is the number of query to $\mathscr{CO}$.

The oracles are simulated as follows:

- $H_S(i)$: On input new i, randomly pick λ_j from the given k-CAA2 tuple and return v^{λ_j}. Store (i, λ_j) in tape $\mathscr{L}_S$.
- $\mathscr{JO}(i, ID_{GM_S})$: It interacts with honest issuer ID_{GM_S}. Computes ID_{Si} as in $\mathscr{KEO}_{\mathscr{S}}(i)$. Then interacts with ID_{GM_S} with ID_{Si}. Finally, it returns $cert_i$.
- $\mathscr{CO}(i, ID_{GM_S})$: On input the sender identity, this oracle outputs the sender's secret keys. Computes ID_{Si} as in $\mathscr{KEO}_{\mathscr{S}}(i)$. Randomly selects $e \in \mathbb{Z}_p$, and computes $A' = (u/H_S(i))^{1/(e+SK_{GM_S})}, W = \hat{e}(H_S(i)), g_G)$. Store (i, A', e, W) in reg. Returns (A', e, aux_{GM_S}) to honest sender i. For the $\ell-$th query, randomly selects $h \in \mathbb{Z}_p$ and computes $aux_{GM_S} = g_2^{\lambda} MPK_{GM_S}^{-h}$. $\mathscr{S}$ back patch $H_G(aux_{GM_S} \parallel ID_{OA_S}) = h$. Pick a pair of (A_i', e_i, λ_i) from the k-CCA2 tuple. Back patches (i, λ_i) to $\mathscr{L}_S$. Then we have $ID_{Si} = MPK_S^{\lambda_i}$. Returns $(ID_{Si}, A_i', e_i, aux_{GM_S})$. Computes $W = \hat{e}(H_S(i), g_G)$. Stores $(i, A_i', e_{i,W})$ in reg. Denote this identity as $ID_{GM_{Sg}}$. If $ID_{GM_S} = ID_{GM_{Sg}}$ in future queries, also runs the above steps.

Other oracles are similar to the proof of theorem 3. Suppose $\mathscr{A}$ can output a valid signature σ such that the sender open authority cannot trace the identity of the sender, or the sender open authority can find the identity but cannot prove that to $Judge$. Below we proof the soundness of the proof system between $Open$ and $Judge$. Rewind the simulation to obtain the following:

$$1 = \Delta z' h_1^{z_0'} t_0'^{\Delta c''}, 1 = \hat{e}(h, U)_1^{z_0'} t_1'^{\Delta c''}, 1 = \hat{e}(h, g_O)_1^{z_0'} t_2'^{\Delta c''}$$
$$t_0' = \Delta z_1'^{1/\Delta c''} h^{z_0'/\Delta c''}, t_1' = \hat{e}(h, U)^{z_0'/\Delta c''}, t_2' = \hat{e}(h, g_O)^{z_0'/\Delta c''}.$$

Notice that we have the following:

$$t_1' = \hat{e}(h, U)^{s_0'} = \hat{e}(t_0', U)m'^{-1}, t_1' = \hat{e}(h, U)^{s_0'} = \hat{e}(t_0' SK_{OA}^{-1}, g_O).$$

Let $\tilde{s_0'} = -\Delta z_0'/\Delta c''$. Hence, $m' = \hat{e}(t_0', U)t_1'^{-1} = \hat{e}(h^{\tilde{s_0'}} t_0', U)$. Since we have $t_0' SK_{OA}^{-1} = h^{-\tilde{s_0'}}$, then $m' = \hat{e}(SK_{OA_S}, U)$. Therefore, we extract the witness $SK_{OA_S} = t_0' h^{\tilde{s_0'}}$. Hence, for a open authority with secret key SK_{OA_S}, he can always output a valid proof to the $Judge$ if he knows the identity of the sender.

If finally $\mathscr{A}$ returns a signature with $ID_{GM_S} = ID_{GM_{Sg}}$, then we rewind the simulation to the point where c' is computed. We get the following: $g_4^{\Delta z_0} t_0^{\Delta c'} = 1, \Delta z_1 g_5^{\Delta z_0} t_1^{\Delta c'} = 1, \Delta z_2 g_6^{\Delta z_0} t_2^{\Delta c'} = 1, t_3^{\Delta z_4} t_5^{\Delta c'} = 1, g_G^{\Delta z_6} U \Delta c' = 1$. Let $\tilde{s}_1 = -\Delta z_0/\Delta c', I\tilde{D}_S = \Delta z_1^{-1/\Delta c'}, \tilde{H} = \Delta z_2^{-1/\Delta c'} = H_1(i), \tilde{A}' = \Delta z_3^{-1/\Delta c'}, \tilde{e} = -\Delta z_4/\Delta c', \tilde{s}_2 = -\Delta z_5/\Delta c', d' = -\Delta z_6/\Delta c'$. We have the following:

$$\hat{e}(g_7, g_G)^{\Delta z_5}[\hat{e}(g_7, S)\hat{e}(g_6, g_G)]^{\Delta z_1} t_6^{\Delta c'} = 1$$
$$\hat{e}(g_7, g_G)^{\tilde{s_2}}[\hat{e}(g_7, S)\hat{e}(g_6, g_G)]^{\tilde{s_1}} = t_6$$
$$= \hat{e}(u, g_G)^{-1}\hat{e}(t_2 t_3, g_G)\hat{e}(t_3, S).$$

After rearranging, we have the following:

$$\hat{e}(u, g_G) = \hat{e}(\tilde{A}', g_G)^{\tilde{e}}\hat{e}(\tilde{A}', S)\hat{e}(\tilde{H}, g_G)\hat{e}(g_7, g_G)^{\tilde{e}\tilde{s_1} - \tilde{s_2}}.$$

If $\tilde{e}\tilde{s_1} = \tilde{s_2}$, then we get a pair of $(\tilde{A}', \tilde{e}, \tilde{H})$ which satisfy $\tilde{A}'^{\tilde{e}+\lambda}\tilde{H} = u$. Then we have $(\tilde{A}', \tilde{e}, \lambda)$, where $(i, \lambda) \in \mathscr{L}_1$ that solves the k-CAA2 problem. If $\tilde{e}\tilde{s_1} \neq \tilde{s_2}$, we have $\tilde{A}'^{\tilde{e}+\lambda}\tilde{H} g_7^{\tilde{e}\tilde{s_1} - \tilde{s_2}} = u$. Then we have $\lambda^* = \lambda + \mu(\tilde{e}\tilde{s_1} - \tilde{s_2})$, where $(i, \lambda) \in \mathscr{L}_1$, such that $(\tilde{A}', \tilde{e}, \lambda^*)$ solves the k-CAA2 problem.

Hence, if $\mathscr{A}$ has a non-negligible advantage ε in winning the game, $\mathscr{S}$ has advantage ε/q_c in solving the k-CAA2 problem. □

5 Conclusion

We formalized a new file sharing scheme for smart cities, referred to as privacy-preserving identity-based file sharing which embraces file confidentiality, file integrity, receiver privacy, and sender privacy. It allows an anonymous sender to share a file with any group member. The receiver of

the ciphertext also remains anonymous. Identities of the sender and receiver can be traced if the need arises. We propose a concrete construction of our PIFS scheme and prove the security properties formally. Our scheme has constant complexity in computation and communication.

Acknowledgments This paper is supported by the Natural Science Foundation of China through projects 61672083, 61370190, 61532021, 61472429, and 61402029, by the Beijing Natural Science Foundation through project 4132056.

References

1. Al-Hader M, Rodzi A, Sharif AR, Ahmad N (2009) SOA of smart city geospatial management. In: 3rd UKSim European symposium on computer modeling and simulation, pp 6–10. doi:10.1109/EMS.2009.112
2. Bohli JM, Skarmeta A, Victoria Moreno M, Garcia D, Langendorfer P (2015) SMARTIE project: secure IoT data management for smart cities. In: International conference on recent advances in internet of things, pp 1–6. doi:10.1109/RIOT.2015.7104906
3. Cramer R, Shoup V (1998) A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk H (ed) CRYPTO 1998. LNCS, vol 1462. Springer, Berlin, Heidelberg, pp 13–25
4. Cathalo J, Libert B, Yung M (2009) Group encryption: Noninteractive realization in the standard model. In: Matsui M (ed) ASIACRYPT 2009. LNCS, vol 5912. Springer, Berlin, Heidelberg, pp 179–196
5. Chourabi H, Nam T, Walker S, Gil-Garcia JR, Mellouli S, Nahon K, Pardo TA, Scholl HJ (2012) Understanding smart cities: an integrative framework. In: Hawaii international conference on system science (HICSS), pp 2289–2297. doi:10.1109/HICSS.2012.615
6. Chen X, Li J, Huang X, Ma J, Lou W (2015) New publicly verifiable databases with efficient updates. IEEE Trans Dependable Sec Comput 12(5):546–556
7. Elmaghraby AS, Losavio MM (2014) Cyber security challenges in smart cities: safety, security and privacy. J Adv Res 5(4):491–497
8. Groth J (2006) Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai X., Chen K (eds) ASIACRYPT 2006. LNCS, vol 4284. Springer, Berlin, Heidelberg, pp 444–459
9. Gentry C (2006) Practical identity-based encryption without random oracles. In: Vaudenay S (ed) EUROCRYPT 2006, LNCS, vol 4004. Springer, Berlin, Heidelberg, pp 445–464
10. Huang X, Liu JK, Tang S, Xiang Y, Liang K, Xu L, Zhou J (2015) Cost-effective authentic and anonymous data sharing with forward security. IEEE Trans Comput 64(4):971–983
11. Hoßfeld T, Tutschku K, Andersen FU (2005) Mapping of file-sharing onto mobile environments: feasibility and performance of eDonkey with GPRS. In: Wireless communications and networking conference, vol 4, pp 2453–2458. doi:10.1109/WCNC.2005.1424899
12. Iamnitchi A, Ripeanu M, Santos-Neto E, Foster I (2011) The small world of file sharing. IEEE Trans Parallel Distrib Syst 22(7):1120–1134
13. Khan Z, Pervez Z, Ghafoor A (2014) Towards cloud based smart cities data security and privacy management. In: International conference on utility and cloud computing, pp 806–811. doi:10.1109/UCC.2014.131
14. Kiayias A, Tsiounis Y, Yung M (2007) Group encryption. In: Kurosawa K (ed) ASIACRYPT 2007. LNCS, vol 4833. Springer, Berlin, Heidelberg, pp 181–199
15. Luo X, Ren Y, Liu J, Hu J, Liu W, Wang Z, Xu W, Wu Q (2016) Identity-based group encryption. In: Liu JK, Steinfeld R (eds) ACISP 2016. LNCS, vol 9723. Springer, Berlin, Heidelberg, pp 87–102
16. Libert B, Yung M, Joye M, Peters T (2014) Traceable group encryption. In: Krawczyk H (ed) PKC 2014. LNCS, vol 8383. Springer, Berlin, Heidelberg, pp 592–610
17. Lu K, Wang J, Li M (2016) An Eigentrust dynamic evolutionary model in P2P file-sharing systems. Peer-to-Peer Netw Appl 9(3):599–612. doi:10.1007/s12083-015-0416-1
18. Monzon A (2015) Smart cities concept and challenges: bases for the assessment of smart city projects. In: International conference on smart cities and green ICT systems, pp 1–11
19. Qin B, Wu Q, Susilo W, Mu Y (2009) Publicly verifiable privacy-preserving group decryption. In: Yung M, Liu P, Lin D (eds) Inscrypt 2008. LNCS, vol 5487. Springer, Berlin, Heidelberg, pp 72–83
20. Suciu G, Vulpe A, Halunga S, Fratu O, Todoran G, Suciu V (2013) Smart cities built on resilient cloud computing and secure Internet of Things. In: International conference on control systems and computer science, pp 513–518. doi:10.1109/CSCS.2013.58
21. Su K, Li J, Fu H (2011) Smart city and the applications. In: International conference on electronics, communications and control, pp 1028–1031. doi:10.1109/ICECC.2011.6066743
22. Shen H (2010) An efficient and adaptive decentralized file replication algorithm in P2P file sharing systems. IEEE Trans Parallel Distrib Syst 21(6):827–840
23. Wu Q, Domingo-Ferrer J, González-Nicolás Ú (2010) Balanced trustworthiness, safety, and privacy in vehicle-to-vehicle communications. IEEE Trans Vehicular Technol 59(2):559–573
24. Wang P, Ali A, Kelly W (2015) Data security and threat modeling for smart city infrastructure. In: International conference on cyber security of smart cities, industrial control system and communications, pp 1–6. doi:10.1109/SSIC.2015.7245322
25. Wei VK, Yuen TH, Zhang F (2005) Group signature where group manager, members and open authority are identity-based. In: Boyd C., Nieto JMG (eds) ACISP 2005. LNCS, vol 3574. Springer, Berlin, Heidelberg, pp 468–480
26. Zanella A, Bui N, Castellani A, Vangelista L, Zorzi M (2014) Internet of things for smart cities. IEEE Intern Things J 1(1):22–32