Secure Provenance: The Essential of Bread and Butter of Data Forensics in Cloud Computing

Rongxing Lu[†], Xiaodong Lin[‡], Xiaohui Liang[†], and Xuemin (Sherman) Shen[†]
[†]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada
[‡]Faculty of Business and Information Technology, University of Ontario Institute of Technology, Canada
{rxlu, x27liang, xshen}@bbcr.uwaterloo.ca; xiaodong.lin@uoit.ca

ABSTRACT

Secure provenance that records ownership and process history of data objects is vital to the success of data forensics in cloud computing, yet it is still a challenging issue today. In this paper, to tackle this unexplored area in cloud computing, we proposed a new secure provenance scheme based on the bilinear pairing techniques. As the essential bread and butter of data forensics and post investigation in cloud computing, the proposed scheme is characterized by providing the information confidentiality on sensitive documents stored in cloud, anonymous authentication on user access, and provenance tracking on disputed documents. With the provable security techniques, we formally demonstrate the proposed scheme is secure in the standard model.

Categories and Subject Descriptors

C.2.0 [**General**]: Security and protection

General Terms

Security, Privacy

Keywords

Cloud Computing, Data Forensics, Secure Provenance, Provable Security, Standard Model

1. INTRODUCTION

Cloud computing, as an emerging computing paradigm aiming to share storage, computation, and services transparently among a massive users, has gathered great momentum from not only industry but also academia [5]. In essence, cloud computing overlaps many existing concepts, such as distributed, grid and utility computing [17]. However, driven largely by marketing and service offerings from big corporate players like Google, IBM and Amazon, cloud computing has evolved out of these concepts and become a new buzz word focusing on "cloud" — more abstract resource and services' delivery [8]. Once cloud computing steps into our daily lives, any locally stored information, such as email, word processing documents and spreadsheets, could be remotely stored in a cloud [6]. Then, we can use any terminals, e.g., computer, laptop and PDA etc., to access these information at anytime, anywhere. Due to these promising characteristics, cloud computing has become increasingly attractive to the public [10, 20, 4, 19].

However, the flourish of cloud computing still hinges up fully understanding and managing the challenges that the public concerns, for instance, the confidentiality and privacy issues in cloud computing [10]. When a user stores some sensitive information in a cloud, the confidentiality of these sensitive information is of concern to the user. Without any protection on these sensitive information, e.g., personal financial information, health records, a user won't have confidence in storing his/her sensitive information in cloud. Similarly, when a company stores some business documents, e.g., business plans, in a cloud, the company also cares about the confidentiality and hopes only the relevant personnel can access these documents after they are authorized. Besides the confidentiality of these sensitive information, the user's identity privacy, a fundamental right to privacy, is also expected in cloud computing. If the access to a cloud discloses a user's real identity, the user could still be unwilling to accept this paradigm. Due to this reason, the user authentication without identifying the real identity, also called anonymous authentication [3], is desirable in cloud computing. Although anonymous authentication can provide user identity privacy, it is a two-edged sword to provide complete anonymous access in cloud computing. For example, when a group of users are authorized to some financial computing or data-intensive scientific collaborations in a cloud, if an important data modified by someone is disputed, it is hard to track the real user due to complete anonymous authentication. Therefore, to tackle this dilemma, cloud computing should also provide *provenance* [16] to record ownership and process history of data objects in cloud in order for wide acceptance to the public.

The concept of provenance has been extensively studied for a long time, and widely used in the archival theory to denote the documented history of some data objects [9]. Given its provenance, a data object can report who created and who modified its contents. Therefore, once a dispute rises in a document stored in a cloud, provenance is important for data forensics to provide digital evidences for post investigation. However, provenance is still an unexplored area in cloud computing [5], in which we need to deal with many

ASIACCS'10 April 13–16, 2010, Beijing, China.

challenging security issues. For example, in support of data forensics in cloud computing, the provenance information must be secured, i.e., they do not violate the information confidentiality and user privacy in cloud computing. Specifically, as the essential of bread and butter of data forensics in cloud computing, secure provenance should at least satisfy the following basic requirements:

- *Unforgeability*: a genuine provenance record in cloud computing can effectively attest the ownership and process history of data objects stored in a cloud, any adversary cannot forge a valid provenance records, i.e., modifying an item in a existing record or directly introducing a new forged record without being detected.

- *Conditional privacy preservation*: To ensure information confidentiality and anonymous authentication in cloud computing, a genuine provenance record should also be conditional privacy preserving [12, 14, 15, 13]. That is, only a trusted authority has the ability to reveal the real identity recorded in the provenance, while anyone else cannot.

Secure provenance is vital to the success of data forensics in cloud computing, yet it is still a challenging issue today [5]. Aiming at this, in this paper, we propose a secure provenance scheme based on the bilinear pairing technique to provide trusted evidences for data forensics in cloud computing. Concretely, this paper will make the following contributions:

- Firstly, the formal definition and security notions of secure provenance for cloud computing are introduced;

- Secondly, based on the bilinear pairings [2, 11], a concrete secure provenance scheme is proposed, which can achieve the information confidentiality, anonymous access to the cloud, and conditional provenance tracking;

- Thirdly, we use the provable security technique to validate its security in the standard model.

The remainder of this paper is organized as follows. In Section 2, we introduce the formal definition and security notions of secure provenance for cloud computing. In Section 3, we recall bilinear maps and the corresponding complex assumptions. Then, we present our secure provenance scheme in Section 4, followed by the formal security proofs in Section 5. In the end, we draw our conclusions in Section 6.

2. MODEL AND SECURITY NOTIONS

2.1 System Model

We consider a representative cloud computing architecture, which consists of a trusted system manager (SM), a cloud service provider (SP) and a large number of users $\mathcal{U} = \{U_1, U_2, \cdots\}$, as shown in Figure 1.

- System Manager (SM): SM is a trustable and powerful entity, and located at the top of the cloud computing system. The responsibility of SM is in charge of the management of the whole system, for example, registering the cloud service provider and users, and investigating the malicious or faulty users that had operated on some disputed data in cloud computing systems.

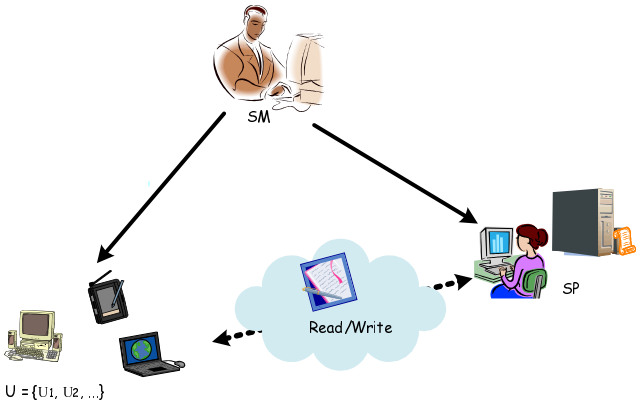

Figure 1: System model under consideration

- Service Provider (SP): SP is also trustable and has significant resources and monitors live cloud computing systems. When a user accesses to a cloud computing system, for example, reading/writing a file, SP will first authenticate the user and make the system resources available after the user authentication is passed.

- Users $\mathcal{U}$: $\mathcal{U} = \{U_1, U_2, \cdots\}$ are a group of registered users, which can operate all group resources stored at the cloud computing systems. Each user $U_i \in \mathcal{U}$ is privacy sensitive. When a user accesses the cloud computing system, it doesn't hope its real identity will be disclosed to neither the SP nor other users. However, when a dispute takes place on some sensitive operations, e.g., falsely modifying a file, the user's identity can be traced by the SM.

2.2 Definition of Secure Provenance

To fit for the above system model, we formally define the secure provenance ($\mathcal{SP}$) in this subsection.

Definition 1. ($\mathcal{SP}$ Scheme) A secure provenance scheme $\mathcal{SP}$ is defined by the following algorithms: *system setup*, *key generation*, *anonymous authentication*, *authorized access*, and *provenance tracking*.

- a system setup algorithm **Setup**: it is a probabilistic algorithm run by SM, which takes as input a security parameter k and outputs the system public parameters *params* and *master key*.

- a key generation algorithm **KGen**: it is an algorithm run by SM, which takes as input the public parameters *params*, *master key* and an identifier of either a user $U_i \in \mathcal{U}$ or the SP, and outputs a corresponding private key sk_i. This algorithm can be either probabilistic or deterministic.

- an anonymous authentication algorithm **AnonyAuth**: it is the first interaction run between a user U_i and the SP. First, SP sends a random number χ to U_i, and U_i takes as input a fresh nonce Y_i, the private key sk_i, and the public parameters *params*, and outputs an anonymous signature σ_A of $Y_i||\chi$. Then, the SP takes as input the purported signature σ_A, the fresh nonce Y_i, and the public parameters *params* and tests

whether σ_A is valid. If it is valid, the anonymous authentication is passed; otherwise rejected.

- an authorized access algorithm **AuthorAccess**: it is the second interaction between a user U_i and the SP to achieve *fine-grained* authorized access after a successful anonymous authentication. First, the SP grants a unique access secret key ask_i corresponding to the nonce Y_i to the user U_i. Then, the user U_i can operate (i.e., create, modify) some data M in cloud computing systems attested with a provenance signature σ_P generated with ask_i. Later anyone can check the validity, but only the authorized users in the same group can read the data M.
- a provenance tracking algorithm **ProveTrack**: Once a sensitive operation is disputed, the SP can provide $(\sigma_P, Y_i, \sigma_A)$ to the SM. Then, the SM can track the real identity of the user with $(\sigma_P, Y_i, \sigma_A)$.

Correctness: A secure provenance scheme $\mathcal{SP}$ should satisfy the following properties: a registered user must be anonymously authenticated by the **AnonyAuth** algorithm with σ_A; an authorized access operation must be verified by the **AuthorAccess** algorithm with σ_P; Once a user executed a disputed operation, the real identity of the user must be tracked by the **ProveTrack** algorithm.

2.3 Security Notions

2.3.1 Confidentiality on data M

We describe the security model of confidentiality on data M in the spirit of cpa indistinguishability in the $\mathcal{SP}$ scheme. In specific, in the $\mathcal{SP}$ setting, an adversary $\mathcal{A}$ is first given the public parameters, and outputs two data M_0, M_1 of the same length. Then, after one data M_b, $b \in \{0,1\}$, is encrypted, the adversary $\mathcal{A}$ must decide which message has been encrypted.

Definition 2. (Confidentiality) Let k and t be integers and ϵ be a real in $[0,1]$, let $\mathcal{SP}$ be a secure provenance scheme with security parameter k. Let $\mathcal{A}$ (excluded from $\mathcal{U}$) be a cpa adversary against $\mathcal{SP}$. We consider the following random experiments:

$\mathit{Experiment}\ \mathbf{Exp}^{\mathsf{cpa}}_{\mathcal{SP},\mathcal{A}}(k)$
  $params, masterkey \xleftarrow{R} \mathbf{Setup}(k)$
  $(M_0, M_1) \leftarrow \mathcal{A}$
  $b \xleftarrow{R} \{0,1\}$, $C \leftarrow M_b$
  $b' \leftarrow \mathcal{A}(params, C)$
  $\mathit{if}\ b = b',\ \mathit{then\ return}\ \ b^* \leftarrow 1\ \mathit{else}\ b^* \leftarrow 0$
  $\mathit{return}\ b^*$

We define the success probability of $\mathcal{A}$ via

$$\mathbf{Succ}^{\mathsf{cpa}}_{\mathcal{SP},\mathcal{A}}(k) = 2\cdot\Pr\left[\mathbf{Exp}^{\mathsf{cpa}}_{\mathcal{SP},\mathcal{A}}(k) = 1\right] - 1 = 2\cdot\Pr\left[b = b'\right] - 1$$

$\mathcal{SP}$ is said to be (k,t,ϵ)-cpa secure, if no adversary $\mathcal{A}$ running in time t has a success $\mathbf{Succ}^{\mathsf{cpa}}_{\mathcal{SP},\mathcal{A}}(k) \geq \epsilon$.

2.3.2 Unforgeability on signature σ_P

For digital signatures, the strong security notion was defined by Goldwasser, Micali and Rivest in [7] as existential forgery against adaptive chosen message attack (ef-cma). In the $\mathcal{SP}$ setting, an ef-cma adversary $\mathcal{A}$ is given the public parameters, as well as an access to the key generation oracle $\mathcal{O}_K$ on some registered users in $\mathcal{U} = \{U_1, U_2, \cdots\}$, to the signing oracle $\mathcal{O}_S$ of the service provider SP. As usual, in the adversary answer, there is the natural restriction that in the returned signature $\sigma_P^\star$, the signature $\sigma_P^\star$ on message $M^\star$ has not been obtained from $\mathcal{O}_S$.

Definition 3. (Unforgeability) Let $\mathcal{U} = \{U_1, U_2, \cdots\}$ be a group of registered users and SP be the service provider, k and t be integers and ϵ be a real in $[0,1]$, let $\mathcal{SP}$ be a secure provenance scheme with security parameter k. Let $\mathcal{A}$ be an ef-cma adversary against $\mathcal{SP}$. We consider the following random experiment:

$\mathit{Experiment}\ \mathbf{Exp}^{\mathsf{ef\text{-}cma}}_{\mathcal{SP},\mathcal{A}}(k)$
  $params, masterkey \xleftarrow{R} \mathbf{Setup}(k)$
  $(pk_{sp}, sk_{sp}) \xleftarrow{R} \mathbf{KGen}(params, masterkey)$ for SP
  for each $U_i \in \mathcal{U}$ do
    $sk_i \xleftarrow{R} \mathbf{KGen}(params, masterkey)$
  $(\sigma_P^\star, M^\star) \leftarrow \mathcal{A}^{\mathcal{O}_K, \mathcal{O}_S}(params, pk_{sp})$
  $\mathit{if}\ \sigma_P^\star$ is valid $\mathit{then}\ b^* \leftarrow 1\ \mathit{else}\ b^* \leftarrow 0$
  $\mathit{return}\ b^*$

We define the success probability of $\mathcal{A}$ via

$$\mathbf{Succ}^{\mathsf{ef\text{-}cma}}_{\mathcal{SP},\mathcal{A}}(k) = \Pr\left[\mathbf{Exp}^{\mathsf{ef\text{-}cma}}_{\mathcal{SP},\mathcal{A}}(k) = 1\right]$$

$\mathcal{SP}$ is said to be (k,t,ϵ)-ef-cma secure, if no adversary $\mathcal{A}$ running in time t has a success $\mathbf{Succ}^{\mathsf{ef\text{-}cma}}_{\mathcal{SP},\mathcal{A}}(k) \geq \epsilon$.

2.3.3 Full anonymity on signature σ_A

A typical approach to define full anonymity σ_A is the following experiment in the spirit of cpa-indistinguishability [2, 11]. In the $\mathcal{SP}$ setting, an adversary $\mathcal{A}$ is given the public parameters, as well as an access to the key generation oracle $\mathcal{O}_K$ on some registered users in $\mathcal{U} = \{U_1, U_2, \cdots\}$, to the anonymous signature oracle $\mathcal{O}_S$ on some valid signatures. Then, $\mathcal{A}$ outputs (U_0, U_1, m). After an anonymous signature σ_A^b corresponding to (U_b, m) is signed, where $b \in \{0,1\}$, the adversary $\mathcal{A}$ must decide which user in (U_0, U_1) signed the signature σ_A^b.

Definition 4. (Full Anonymity) Let $\mathcal{U} = \{U_1, U_2, \cdots\}$ be a group of registered users, k and t be integers and ϵ be a real in $[0,1]$, let $\mathcal{SP}$ be a secure provenance scheme with security parameter k. Let $\mathcal{A}$ be an adversary against the anonymity of $\mathcal{SP}$. We consider the following random experiment:

$\mathit{Experiment}\ \mathbf{Exp}^{\mathsf{anony}}_{\mathcal{SP},\mathcal{A}}(k)$
  $params, masterkey \xleftarrow{R} \mathbf{Setup}(k)$
  for each $U_i \in \mathcal{U}$ do
    $sk_i \xleftarrow{R} \mathbf{KGen}(params, masterkey)$
  $(U_0, U_1, M) \leftarrow \mathcal{A}$
  $b \xleftarrow{R} \{0,1\}$, $\sigma_A^b \leftarrow (U_b, M)$
  $b' \leftarrow \mathcal{A}^{\mathcal{O}_K, \mathcal{O}_S}(params, \sigma_A^b)$
  $\mathit{if}\ b = b'$ is valid $\mathit{then\ return}\ \ b^* \leftarrow 1\ \mathit{else}\ b^* \leftarrow 0$
  $\mathit{return}\ b^*$

We define the success probability of $\mathcal{A}$ via

$$\mathbf{Succ}^{\mathsf{anony}}_{\mathcal{SP},\mathcal{A}}(k) = 2\Pr\left[\mathbf{Exp}^{\mathsf{anony}}_{\mathcal{SP},\mathcal{A}}(k) = 1\right] - 1 = 2\Pr\left[b = b'\right] - 1$$

$\mathcal{SP}$ is said to be (k,t,ϵ)-anony secure, if no adversary $\mathcal{A}$ running in time t has a success $\mathbf{Succ}^{\mathsf{anony}}_{\mathcal{SP},\mathcal{A}}(k) \geq \epsilon$.

2.3.4 Full traceability on signature σ_A

Full traceability on signature σ_A can be described in the following game [2, 11]. In the $\mathcal{SP}$ setting, an adversary $\mathcal{A}$ is given the public parameters and tracking information (trackinfo), as well as an access to the key generation oracle $\mathcal{O}_K$ on some registered users in $\mathcal{U} = \{U_1, U_2, \cdots\}$, to the anonymous signature oracle $\mathcal{O}_S$ on some valid signatures, where the validity of signature and identity tracing can be checked by $\mathcal{A}$. At some point, $\mathcal{A}$ outputs a forged anonymous signature $\sigma_A^\star$ with its tracking identity $U^\star \in \mathcal{U}$ and a message $M^\star$. There is the natural restriction that in the returned signature $\sigma_A^\star$, the identity $U^\star$ has not been queried on $\mathcal{O}_K$ and $(U^\star, M^\star)$ has not been obtained from $\mathcal{O}_S$.

Definition 5. (Full Traceability) Let $\mathcal{U} = \{U_1, U_2, \cdots\}$ be a group of registered users, k and t be integers and ϵ be a real in $[0, 1]$, let $\mathcal{SP}$ be a secure provenance scheme with security parameter k. Let $\mathcal{A}$ be an adversary against the traceability of $\mathcal{SP}$. We consider the following random experiment:

Experiment $\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{SP},\mathcal{A}}(k)$
  $params, masterkey \xleftarrow{R} \mathbf{Setup}(k)$
  for each $U_i \in \mathcal{U}$ do
    $sk_i \xleftarrow{R} \mathbf{KGen}(params, masterkey)$
  $\sigma_A^\star \leftarrow \mathcal{A}^{\mathcal{O}_K, \mathcal{O}_S}(params, U^\star)$
  if $\sigma_A^\star$ is valid *then* $b^* \leftarrow 1$ *else* $b^* \leftarrow 0$
  return b^*

We define the success probability of $\mathcal{A}$ via

$$\mathbf{Succ}^{\mathsf{trace}}_{\mathcal{SP},\mathcal{A}}(k) = \Pr\left[\mathbf{Exp}^{\mathsf{trace}}_{\mathcal{SP},\mathcal{A}}(k) = 1\right]$$

$\mathcal{SP}$ is said to be (k, t, ϵ)-trace secure, if no adversary $\mathcal{A}$ running in time t has a success $\mathbf{Succ}^{\mathsf{trace}}_{\mathcal{SP},\mathcal{A}}(k) \geq \epsilon$.

3. BILINEAR MAPS AND COMPLEX ASSUMPTIONS

3.1 Bilinear Groups of Composite Order

Let p, q be two distinct large primes, and $n = pq$. Groups $(\mathbb{G}, \mathbb{G}_T)$ of composite order n are called *bilinear map groups of composite order* if there is a mapping $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with the following properties [2, 11]:

- **Bilinearity:** $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G}^2$ and $a, b \in \mathbb{Z}_n$;
- **Non-degeneracy:** $e(g, h) \neq 1_{\mathbb{G}_T}$ whenever $g, h \neq 1_{\mathbb{G}}$;
- **Computability:** there exists an efficient algorithm to compute $e(g, h) \in \mathbb{G}_T$ for all $(g, h) \in \mathbb{G}$.

3.2 Complex Assumptions

Let $\mathbf{g}$ be a generator of $\mathbb{G}$, then $g = \mathbf{g}^p \in \mathbb{G}$ can generate the subgroup $\mathbb{G}_p = \{g^0, g^1, \cdots, g^{p-1}\}$ of order q, and $g' = \mathbf{g}^q \in \mathbb{G}$ can generate the subgroup $\mathbb{G}_q = \{g'^0, g'^1, \cdots, g'^{q-1}\}$ of order q in $\mathbb{G}$. In the following, we define the quantitative notion of the complexity of the problems underlying the proposed secure provenance scheme, namely the SubGroup Decision (SGD) Problem [2],the Decisional Bilinear Diffie-Hellman (DBDH) Problem [2], the Strong Diffie-Hellman (SDH) Problem [1] and the Strong Diffie-Hellman-2 (SDH2) Problem.

Definition 6. (SGD Problem) The SubGroup Decision (SGD) problem in $\mathbb{G}$ is as follows: Given a tuple $(e, \mathbb{G}, \mathbb{G}_T, n, h)$, where the element h is randomly drawn from either $\mathbb{G}$ or subgroup $\mathbb{G}_q$, decide whether or not $h \in \mathbb{G}_q$.

Definition 7. (SGD Assumption) Let $\mathcal{A}$ be an adversary that takes an input of h drawn from either $\mathbb{G}$ or subgroup $\mathbb{G}_q$, and returns a bit $b' \in \{0, 1\}$. We consider the following random experiments.

Experiment $\mathbf{Exp}^{\mathrm{SGD}}_{\mathcal{A}}$
  $\widetilde{b} \leftarrow \{0, 1\}$
  if $\widetilde{b} = 0$, then $h \xleftarrow{R} \mathbb{G}_q$; else if $\widetilde{b} = 1$ then $h \xleftarrow{R} \mathbb{G}$
  $\widetilde{b}' \leftarrow \mathcal{A}(e, \mathbb{G}, \mathbb{G}_T, n, h)$
  return 1 if $\widetilde{b}' = \widetilde{b}$, 0 otherwise

We then define the advantage of $\mathcal{A}$ via

$$\mathbf{Adv}^{\mathrm{SGD}}_{\mathcal{A}} = \Big|\Pr\left[\mathbf{Exp}^{\mathrm{SGD}}_{\mathcal{A}} = 1 | \widetilde{b} = 0\right] - \Pr\left[\mathbf{Exp}^{\mathrm{SGD}}_{\mathcal{A}} = 1 | \widetilde{b} = 1\right]\Big| \geq \epsilon$$

Let $\tau \in \mathbb{N}$ and $\epsilon \in [0, 1]$. We say that the SGD is (τ, ϵ)-secure if no adversary $\mathcal{A}$ running in time τ has an advantage $\mathbf{Adv}^{\mathrm{SGD}}_{\mathcal{A}} \geq \epsilon$.

Definition 8. (DBDH Problem) The Decisional Bilinear Diffie-Hellman (DBDH) problem in $\mathbb{G}$ is as follows: Given an element g of $\mathbb{G}$, a tuple (g^x, g^y, g^z, T) for unknown $x, y, z \in \mathbb{Z}_n^*$ and $T \in \mathbb{G}_T$, decide whether $T = e(g, g)^{xyz}$ or a random element R drawn from $\mathbb{G}_T$.

Definition 9. (DBDH Assumption) Let $\mathcal{A}$ be an adversary that takes an input of (g^x, g^y, g^z, T) for unknown $x, y, z \in \mathbb{Z}_n^*$ and $T \in \mathbb{G}_T$, and returns a bit $b' \in \{0, 1\}$. We consider the following random experiments.

Experiment $\mathbf{Exp}^{\mathrm{DBDH}}_{\mathcal{A}}$
  $x, y, z \xleftarrow{R} \mathbb{Z}_n$; $R \xleftarrow{R} \mathbb{G}_T$
  $\widetilde{b} \leftarrow \{0, 1\}$
  if $\widetilde{b} = 0$, then $T = e(g, g)^{xyz}$; else if $\widetilde{b} = 1$ then $T = R$
  $\widetilde{b}' \leftarrow \mathcal{A}(g^x, g^y, g^z, T)$
  return 1 if $\widetilde{b}' = \widetilde{b}$, 0 otherwise

We then define the advantage of $\mathcal{A}$ via

$$\mathbf{Adv}^{\mathrm{DBDH}}_{\mathcal{A}} = \Big|\Pr\left[\mathbf{Exp}^{\mathrm{DBDH}}_{\mathcal{A}} = 1 | \widetilde{b} = 0\right] - \Pr\left[\mathbf{Exp}^{\mathrm{DBDH}}_{\mathcal{A}} = 1 | \widetilde{b} = 1\right]\Big| \geq \epsilon$$

Let $\tau \in \mathbb{N}$ and $\epsilon \in [0, 1]$. We say that the DBDH is (τ, ϵ)-secure if no adversary $\mathcal{A}$ running in time τ has an advantage $\mathbf{Adv}^{\mathrm{DBDH}}_{\mathcal{A}} \geq \epsilon$.

Definition 10. (SDH Problem) The Strong Diffie-Hellman (SDH) problem in $\mathbb{G}$ is as follows: Given an element g of $\mathbb{G}$, and a l-tuple $(g^x, g^{(x^2)}, \cdots, g^{(x^l)})$ for some unknown $x \in \mathbb{Z}_n$, compute a new tuple $(c, g^{\frac{1}{x+c}})$, where $c \in \mathbb{Z}_n$.

Definition 11. (SDH Assumption) Let $\mathcal{A}$ be an adversary that takes an input of $(g, g^x, g^{(x^2)}, \cdots, g^{(x^l)})$ for some unknown $x \in \mathbb{Z}_n$, and returns a new tuple $(c, g^{\frac{1}{x+c}})$. We consider the following random experiment.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{SDH}}$

$x \xleftarrow{R} \mathbb{Z}_n$, $(c, \alpha) \leftarrow \mathcal{A}\left(g, g^x, g^{(x^2)}, \cdots, g^{(x^l)}\right)$

if $\alpha = g^{\frac{1}{x+c}}$ *then* $b \leftarrow 1$ *else* $b \leftarrow 0$

return b

We define the corresponding success probability of $\mathcal{A}$ in solving the SDH problem via

$$\mathbf{Succ}_{\mathcal{A}}^{\mathrm{SDH}} = \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{SDH}} = 1\right]$$

Let $\tau \in \mathbb{N}$ and $\epsilon \in [0, 1]$. We say that the SDH is (τ, ϵ)-secure if no polynomial algorithm $\mathcal{A}$ running in time τ has success $\mathbf{Succ}_{\mathcal{A}}^{\mathrm{SDH}} \geq \epsilon$.

Definition 12. (SDH2 Problem) The strong Diffie-Hellman-2 (SDH2) problem in $\mathbb{G}$ is as follows: Let g be an element of $\mathbb{G}$. Given $(g^x, g^y, g^{\frac{1}{x+y}})$ for unknown $x, y \in \mathbb{Z}_n$, l_1 distinct tuples $(c_i, g^{\frac{1}{x+c_i}})$, where $c_i \in \mathbb{Z}_n$, $i \in \{1, 2, \cdots, l_1\}$, and another tuple $(g^{(y^2)}, \cdots, g^{(y^{l_2})})$, compute a new tuple $(m, g^{\frac{1}{y+m}})$, where $m \in \mathbb{Z}_n$.

Definition 13. (SDH2 Assumption) Let $\mathcal{A}$ be an adversary that takes an input of $g, g^x, g^y, g^{\frac{1}{x+y}}, c_1, g^{\frac{1}{x+c_1}}, c_2, g^{\frac{1}{x+c_2}}, \cdots, c_{l_1}, g^{\frac{1}{x+c_{l_1}}}$ for some unknown $x, y, c_1, \cdots, c_{l_1} \in \mathbb{Z}_n$, and another tuple $(g^{(y^2)}, \cdots, g^{(y^{l_2})})$, returns a new tuple (c, α). We consider the following random experiment.

Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{SDH2}}$

$x, y, c_1, \cdots, c_{l_1} \xleftarrow{R} \mathbb{Z}_n$

$$(m, \alpha) \leftarrow \mathcal{A}\left(\begin{array}{l} g, g^x, g^y, g^{\frac{1}{x+y}} \\ c_1, g^{\frac{1}{x+c_1}}, c_2, g^{\frac{1}{x+c_2}}, \cdots, c_{l_1}, g^{\frac{1}{x+c_{l_1}}} \\ g^{(y^2)}, \cdots, g^{(y^{l_2})} \end{array}\right)$$

if $\alpha = g^{\frac{1}{y+m}}$ *then* $b \leftarrow 1$ *else* $b \leftarrow 0$

return b

We define the corresponding success probability of $\mathcal{A}$ in solving the SDH2 problem via

$$\mathbf{Succ}_{\mathcal{A}}^{\mathrm{SDH2}} = \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{SDH2}} = 1\right]$$

Let $\tau \in \mathbb{N}$ and $\epsilon \in [0, 1]$. We say that the SDH2 is (τ, ϵ)-secure if no polynomial algorithm $\mathcal{A}$ running in time τ has success $\mathbf{Succ}_{\mathcal{A}}^{\mathrm{SDH2}} \geq \epsilon$.

4. A SECURE PROVENANCE SCHEME

In this section, based on the bilinear pairing, we construct a fully secure provenance $\mathcal{SP}$ scheme for cloud computing, which includes the following five parts: Setup, KGen, AnonyAuth, AuthAccess, and ProveTrack.

Setup: Given the security parameter k, the bilinear map groups $(\mathbb{G}, \mathbb{G}_T, e)$ of composite order $n = pq$ are chosen, where $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ and p, q are two large primes with $|p| = |q| = k$. Let $\mathbb{G}_p$, $\mathbb{G}_q$ denote $\mathbb{G}$'s two subgroups of orders p and q respectively. Then, the SM chooses four elements (g, u, h_1, h_2) of $\mathbb{G}$, one generator h of $\mathbb{G}_q$, two random exponents $\alpha, a \in \mathbb{Z}_n^*$, and a collision-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_n^*$. After these, the SM sets the *master key* (g^α, a, q) and the system public parameters $params = (\mathbb{G}, \mathbb{G}_T, e, n, g, u, h_1, h_2, e(g,g)^\alpha, A = g^a, h, H)$.

KGen: When the service provider (SP) registers to the system, the SM chooses a random number $s_p \in \mathbb{Z}_n^*$ as the private key sk_{sp} and computes the corresponding public key $pk_{sp} = g^{\frac{1}{a+s_p}}$. Here, if $a + s_p = 0$, s_p should be chosen again until $a + s_p \neq 0 \bmod n$.

When a user $U_i \in \mathcal{U}$ registers to the system, the SM first chooses three random numbers $(t_{i1}, t_{i2}, s_i) \in \mathbb{Z}_n^*$, and computes the private key $sk_i = (aask_i, agsk_i)$, where $aask_i = (s_i, g^{\frac{1}{a+s_i}})$ is the anonymous authentication key, $agsk_i = (g^\alpha g^{at_{i1}}, g^{t_{i1}}, g^{t_{i2}}, h_1^{t_{i1}} h_2^{t_{i2}})$ is the authorized group access key. In addition, SM also stores $(U_i, g^{s_i q})$ in the tracking list used for ProveTrack.

AnonyAuth: When a user U_i is ready to access the cloud computing system, U_i and SP will execute the following first round interaction protocol to achieve anonymous authentication.

- SP first picks a random number $\chi \in \mathbb{Z}_n^*$, and sends χ as a fresh challenge to U_i.
- After receiving χ, U_i also picks a random number $x \in \mathbb{Z}_n^*$, computes $Y = g^x$, and signs an anonymous authentication σ_A on $Y||\chi$ as follows.
  1. Use the anonymous authentication key $aask_i$ to compute ρ, where
  $$\rho = (\rho_1, \rho_2, \rho_3) = \left(g^{s_i}, g^{\frac{1}{a+s_i}}, u^{\frac{1}{s_i + H(Y||\chi)}}\right) \quad (1)$$
  2. Choose three random numbers $z_1, z_2, z_3 \in \mathbb{Z}_n^*$ and compute $(\sigma_1, \sigma_2, \sigma_3)$ and the proof (π_1, π_2)
  $$\sigma_1 = \rho_1 \cdot h^{z_1}; \sigma_2 = \rho_2 \cdot h^{z_2}; \sigma_3 = \rho_3 \cdot h^{z_3}; \quad (2)$$
  $$\begin{aligned} \pi_1 &= \rho_2^{z_1} (A\rho_1)^{z_2} h^{z_1 z_2}; \\ \pi_2 &= \rho_3^{z_1} (g^{H(Y||\chi)} \rho_1)^{z_3} h^{z_1 z_3}; \end{aligned} \quad (3)$$
  3. Send the anonymous authentication $\sigma_A||Y||\chi$ to the SP, where
  $$\sigma_A = (\sigma_1, \sigma_2, \sigma_3, \pi_1, \pi_2) \quad (4)$$
- Upon receiving $\sigma_A = (\sigma_1, \sigma_2, \sigma_3, \pi_1, \pi_2)$ and $Y||\chi$, SP verifies its validity as follows.
  1. Compute T_1 and T_2, where
  $$T_1 = \frac{e(\sigma_1 A, \sigma_2)}{e(g,g)}; T_2 = \frac{e(\sigma_1 g^{H(Y||\chi)}, \sigma_3)}{e(g,u)}; \quad (5)$$
  2. Verify T_1 and T_2,
  $$T_1 \stackrel{?}{=} e(\pi_1, h); T_2 \stackrel{?}{=} e(\pi_2, h); \quad (6)$$
  If the above two equations hold, SP continues to execute AuthAccess; otherwise SP rejects the access request.

AuthAccess: After the U_i is anonymously authenticated, SP will authorize U_i to access the data stored in the cloud computing system as follows.

- SP first computes the authorization certificate σ_P, where
$$\sigma_P = (\sigma_{P1}, \sigma_{P2}, \sigma_{P3}) = \left(g^{s_p}, g^{\frac{1}{a+s_p}}, u^{\frac{1}{s_p + H(Y)}}\right) \quad (7)$$
and sends σ_P to U_i.

- When U_i receives $\sigma_P = (\sigma_{P1}, \sigma_{P2}, \sigma_{P3})$, it first verifies its validity by checking

$$\begin{aligned} e(A\sigma_{P1}, \sigma_{P2}) &\stackrel{?}{=} e(g, g); \\ e(g^{H(Y)}\sigma_{P1}, \sigma_{P3}) &\stackrel{?}{=} e(g, u); \end{aligned} \tag{8}$$

If both of them hold, $Y = g^x$ is attested, then U_i can operate (i.e., create, modify) some data M in cloud computing systems with the authorized unique access secret key $ask_i = x$.

- After U_i finishes processing on M, it runs the following steps on M so that only authorized users in the same group can read M.

1. Choose a random number $s \in \mathbb{Z}_n^*$;
2. Encrypt M as $C = (C_1, C_2, C_3, C_4)$, where

$$\begin{aligned} C_1 &= M \cdot e(g, g)^{\alpha s}; C_2 = g^s; \\ C_3 &= A^s h_1^{-s}; C_4 = h_2^{-s}; \end{aligned} \tag{9}$$

3. Use the short signature [1] to authenticate C as

$$sig = g^{\frac{1}{x + H(C)}} \tag{10}$$

4. Send $C||sig$ to the SP.

- Upon receiving $C = (C_1, C_2, C_3, C_4)$ and sig, SP first verifies the validity by checking

$$e(Y g^{H(c)}, sig) \stackrel{?}{=} e(g, g) \tag{11}$$

Because the short signature sig is provably secure in the standard model under the SDH assumption. Once the above equation hold, SP accepts the operation, and stores (C, σ_A) in the cloud computing system. Otherwise, SP rejects C and sig.

Once C is stored in the cloud computing system, each authorized member U_j in the same group $\mathcal{U}$, with its authorized group access key $agsk_j = (g^\alpha g^{a t_{j1}}, g^{t_{j1}}, g^{t_{j2}}, h_1^{t_{j1}} h_2^{t_{j2}})$, can read M from C as follows,

$$\begin{aligned} &\frac{e(C_2, g^\alpha g^{a t_{j1}})}{e(g^{t_{j1}}, C_3) \cdot e(g^{t_{j2}}, C_4) \cdot e(h_1^{t_{j1}} h_2^{t_{j2}}, C_2)} \\ =&\frac{e(g^s, g^\alpha g^{a t_{j1}})}{e(g^{t_{j1}}, g^{as} h_1^{-s}) \cdot e(g^{t_{j2}}, h_2^{-s}) \cdot e(h_1^{t_{j1}} h_2^{t_{j2}}, g^s)} \\ =&\frac{e(g^s, g^\alpha) e(g^s, g^{a t_{j1}})}{e(g^{t_{j1}}, g^{as}) \cdot e(g, h_1^{-s t_{j1}} h_2^{-s t_{j2}}) \cdot e(h_1^{t_{j1}} h_2^{t_{j2}}, g^s)} \\ =&e(g^s, g^\alpha) \end{aligned} \tag{12}$$

$$\frac{C_1}{e(g^s, g^\alpha)} = \frac{M \cdot e(g, g)^{\alpha s}}{e(g^s, g^\alpha)} = M \tag{13}$$

Note that, the authorized group access key $agsk_j$ for each group member $U_j \in \mathcal{U}$ is distinct, while the distinct $agsk_j$ can decrypt the same ciphertext C[1]. Therefore, when $agsk_j$ has other purposes in the cloud computing systems, this kind of authorized access is more secure than sharing the same key in traditional group key management.

ProveTrack: Once the data M decrypted from C is in dispute, SP can track the $\sigma_A||Y||\chi$ with the clue (sig, Y) that authenticates C, and submit $\sigma_A = (\sigma_1, \sigma_2, \sigma_3, \pi_1, \pi_2)$ to the SM. Then, SM uses the master key q to compute

$$\sigma_1^q = \left(g^{s_i} h^{t_1}\right)^q = g^{s_i q} \quad (\because h \text{ is a generator of } \mathbb{G}_q) \tag{14}$$

and then can efficiently trace the real user U_i by looking up the entry $(U_i, g^{s_i q})$ in the tracking list.

[1]Note that this kind of encryption technique can be regarded as some kind of single attribute-based encryption [21].

5. SECURITY PROOF

In this section, under the complex assumptions in Section 3.2, we analyze the security of the proposed $\mathcal{SP}$ scheme in the standard model.

Theorem 1. (*Confidentiality of M*) Let $\mathcal{A}$ be a cpa adversary against the proposed $\mathcal{SP}$ scheme in the standard model. Assume that $\mathcal{A}$ has the success probability $\mathbf{Succ}^{\mathsf{cpa}}_{\mathcal{SP},\mathcal{A}} \geq \epsilon$ to break the semantic security of M within the running time τ. Then, there exist $\epsilon' \in [0, 1]$ and $\tau' \in \mathbb{N}$ as follows

$$\epsilon' \geq \frac{\epsilon}{2}; \quad \tau' \leq \tau \tag{15}$$

such that DBDH problem can be solved with probability ϵ' within time τ'.

Proof. We define a sequence of games Game_0, Game_1, $\cdots$ of modified attacks starting from the actual adversary $\mathcal{A}$ [18]. All the games operate on the same underlying probability space: the system parameters and master key, the coin tosses of $\mathcal{A}$. Let $(g^{a'}, g^{b'}, g^{s'}, T)$ be a random instance of DBDH, we will use these incremental games to reduce the DBDH instance to the adversary $\mathcal{A}$ against the semantic security of M in the proposed $\mathcal{SP}$ scheme.

Game_0: This is the real attack game. The SM chooses the master key (g^α, a, q) and the system public parameters $params = (\mathbb{G}, \mathbb{G}_T, e, n, g, u, h_1, h_2, e(g, g)^\alpha, A = g^a, h, H)$. The adversary $\mathcal{A}$ is fed with these system parameters, and at some point, outputs two equal-length data (M_0, M_1) for challenge. We then flip a coin $b \in \{0, 1\}$ and produce a ciphertext $C = (C_1, C_2, C_3, C_4)$ as the challenge to the adversary $\mathcal{A}$. The ciphertext comes from a random number $s \in \mathbb{Z}_n^*$, and $C_1 = M_b \cdot e(g, g)^{\alpha s}$, $C_2 = g^s$, $C_3 = A^s h_1^{-s}$, $C_4 = h_2^{-s}$. Finally, the adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$. In any Game_j, we denote by Guess_j the event $b = b'$. Then, by definition, we have

$$\begin{aligned} \epsilon \leq \mathbf{Succ}^{\mathsf{cpa}}_{\mathcal{SP},\mathcal{A}} = 2\Pr[b = b'] - 1 = 2\Pr[\mathsf{Guess}_0] - 1, \\ \Pr[\mathsf{Guess}_0] \geq \frac{\epsilon}{2} + \frac{1}{2}. \end{aligned} \tag{16}$$

Game_1: In this game, we modify the simulation by replacing the public system parameters $(A = g^a, h_1, h_2)$ with $(g^{a'}, g^{\beta_1} g^{a'}, g^{\beta_2})$ respectively, where β_1, β_2 are randomly chosen from $\mathbb{Z}_n^*$. In addition, we also choose a random number $\alpha' \in \mathbb{Z}_n^*$, and replace $e(g, g)^\alpha$ with $e(g^{a'}, g^{b'}) e(g, g)^{\alpha'}$, which implicitly denotes $\alpha = a'b' + \alpha' \bmod n$; Since the randomness of $(g^{a'}, g^{b'}, \alpha', \beta_1, \beta_2)$, the public system parameters' distribution is unchanged, we thus have

$$\Pr[\mathsf{Guess}_1] = \Pr[\mathsf{Guess}_0]. \tag{17}$$

Game_2: In this game, we modify the simulation by replac-

ing the ciphertext $C = (C_1, C_2, C_3, C_4)$ as

$$\begin{cases} C_1 = \underline{M_b \cdot T \cdot e(g^{s'}, g^{\alpha'})}; \\ C_2 = \underline{g^{s'}}; \\ C_3 = A^{s'} h_1^{-s'} = g^{a's'} (g^{\beta_1} g^{a'})^{-s'} = \underline{(g^{s'})^{-\beta_1}}; \\ C_4 = h_2^{-s'} = (g^{\beta_2})^{-s'} = \underline{(g^{s'})^{-\beta_2}}; \end{cases} \tag{18}$$

If the DBDH challenge $(g^{a'}, g^{b'}, g^{s'}, T)$ is actually a Bilinear Diffie-Hellman tuple $(g^{a'}, g^{b'}, g^{s'}, T = e(g,g)^{a'b's'})$, i.e., $\widetilde{b} = 0$ in experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{DBDH}}$, we know

$$\begin{aligned} C_1 &= M_b \cdot e(g,g)^{a'b's'} \cdot e(g^{s'}, g^{\alpha'}) \\ &= M_b \cdot e(g,g)^{(a'b'+\alpha')s'} = M_b \cdot e(g,g)^{\alpha s'} \end{aligned} \tag{19}$$

Then,

$$\Pr[\mathsf{Guess}_2 | \widetilde{b} = 0] = \Pr[\mathsf{Guess}_1]. \tag{20}$$

and

$$\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{DBDH}} = 1 | \widetilde{b} = 0\right] = \Pr[\mathsf{Guess}_2 | \widetilde{b} = 0] \tag{21}$$

If the DBDH challenge $(g^{a'}, g^{b'}, g^{s'}, T)$ is a random tuple (g, g^x, g^y, g^z), i.e., $(g^{a'}, g^{b'}, g^{s'}, T = R)$, i.e., $\widetilde{b} = 1$ in experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{DBDH}}$, M_b is masked by a random element in $\mathbb{G}_T$, and thus is independent on b. Therefore, we will have

$$\Pr\left[\mathbf{Exp}_{\mathbb{G}}^{\mathrm{DDH}}(\mathcal{A}) = 1 | \widetilde{b} = 1\right] = \Pr[\mathsf{Guess}_2 | \widetilde{b} = 1] = \frac{1}{2}. \tag{22}$$

As a result, from Eqs. (16)-(22), we have

$$\begin{aligned} \epsilon' &= \mathbf{Adv}_{\mathcal{A}}^{\mathrm{DBDH}} \\ &= \left| \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{DBDH}} = 1 | \widetilde{b} = 0\right] - \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{DBDH}} = 1 | \widetilde{b} = 1\right] \right| \\ &\geq \left| \frac{\epsilon}{2} + \frac{1}{2} - \frac{1}{2} \right| = \frac{\epsilon}{2} \end{aligned} \tag{23}$$

In addition, we can obtain the claimed bound for $\tau' \leq \tau$ in the sequence games. Thus, the proof is completed. □

Theorem 2. (*Unforability of* σ_P) Let $\mathcal{U} = \{U_1, U_2, \cdots\}$ be a group of registered users and SP be the service provider. Let $\mathcal{A}$ be an $\mathsf{ef\text{-}cma}$ adversary against the proposed $\mathcal{SP}$ scheme in the standard model. Assume that $\mathcal{A}$ has the success probability $\mathbf{Succ}_{\mathcal{SP},\mathcal{A}}^{\mathsf{ef\text{-}cma}} \geq \epsilon$ to existentially forge the signature σ_P (in a weak chosen message attack), within the running time τ, after making q_K, q_S queries to the key generation oracle $\mathcal{O}_K$ and the signature oracle $\mathcal{O}_S$, respectively. Then, there exist $\epsilon' \in [0,1]$ and $\tau' \in \mathbb{N}$ as follows

$$\epsilon' \geq \epsilon; \quad \tau' \leq \tau + \Theta(.) \tag{24}$$

such that SDH2 problem can be solved with probability ϵ' within time τ', where $\Theta(.)$ is the time cost in the simulation.

Proof. We define a sequence of games Game_0, Game_1, $\cdots$ of modified attacks starting from the actual adversary $\mathcal{A}$ [18]. All the games operate on the same underlying probability space: the system parameters and master key, the coin tosses of $\mathcal{A}$. Let

$$\begin{pmatrix} g, g^x, g^y, g^{\frac{1}{x+y}} \\ c_1, g^{\frac{1}{x+c_1}}, c_2, g^{\frac{1}{x+c_2}}, \cdots, c_{l_1}, g^{\frac{1}{x+c_{q_K}}} \\ g^{(y^2)}, \cdots, g^{(y^{q_S})} \end{pmatrix}$$

be a random instance of SDH2, we will use these incremental games to reduce the SDH2 instance to the adversary $\mathcal{A}$ against the existential forgery under a weak chosen message attack in the proposed $\mathcal{SP}$ scheme.

Game_0: This is the real attack game. In the game, the SM chooses the master key (g^α, a, q) and the system public parameters $params = (\mathbb{G}, \mathbb{G}_T, e, n, g, u, h_1, h_2, e(g,g)^\alpha, A = g^a, h, H)$, and feeds the adversary $\mathcal{A}$ with these system public parameters. First, the adversary $\mathcal{A}$ outputs a list of q_S distinct message $m_1, m_2, \cdots, m_{q_S} \in \mathbb{Z}_n^*$, where $m_i = H(M_i)$ for $i = 1, 2, \cdots, q_S$, exactly makes q_K queries to $\mathcal{O}_K$ on group members' private keys (anonymous authentication keys), q_S queries to $\mathcal{O}_S$ on SP's signatures, and outputs a valid signature $(\sigma^\star, m^\star)$ of SP, where $m^\star = H(M^\star)$ is not queried to $\mathcal{O}_S$ before. In any Game_j, we denote by $\Pr[\mathsf{Forge}_j]$ the forgery success probability of $\mathcal{A}$ in Game_j. Then, by definition, we have

$$\Pr[\mathsf{Forge}_0] = \mathbf{Succ}_{\mathcal{SP},\mathcal{A}}^{\mathsf{ef\text{-}cma}} \geq \epsilon \tag{25}$$

Game_1: In this game, we modify the simulation by replacing the system parameter $(A = g^a, u)$ with (g^x, u'), where u' is generated by adopting the same simulating approach in [1], i.e., given $(g^y, g^{(y^2)}, \cdots, g^{(y^{q_S})}$ and $m_1, m_2, \cdots, m_{q_S} \in \mathbb{Z}_n^*$, we can generate $(u', (u')^y, (u')^{\frac{1}{y+m_1}}, \cdots, (u')^{\frac{1}{y+m_{q_S}}})$. In addition, we set the SP's public key pk_{sp} as g^y. Since the distribution of $(A = g^a, u, pk_{sp})$ is unchanged, we thus have

$$\Pr[\mathsf{Forge}_1] = \Pr[\mathsf{Forge}_0] \tag{26}$$

Game_2: In this game, we will simulate the key generation oracle $\mathcal{O}_K$ on q_K group members' anonymous authentication keys queries. Concretely, when a fresh query on user $U_i \in \mathcal{U}$ is queried, we randomly chooses an un-queried c_i, records (U_i, c_i), and return $(c_i, g^{\frac{1}{x+c_i}})$ as the answer to the oracle query. Since $(c_i, g^{\frac{1}{x+c_i}})$ is uniformly distributed, this game is therefore perfectly indistinguishable from the previous one. Hence,

$$\Pr[\mathsf{Forge}_2] = \Pr[\mathsf{Forge}_1]. \tag{27}$$

Game_3: In this game, we simulate total q_s times signing oracle $\mathcal{O}_S$. For any fresh query on m_i, we pick the corresponding $(u')^{\frac{1}{y+m_i}}$, and return $(g^y, g^{\frac{1}{x+y}}, (u')^{\frac{1}{y+m_i}})$ as the answer to the oracle query. In this game, $\mathcal{O}_S$ perfectly simulates the signature, we will have

$$\Pr[\mathsf{Forge}_3] = \Pr[\mathsf{Forge}_2]. \tag{28}$$

We consider the adversary $\mathcal{A}$ outputs a valid signature $\sigma_P^\star = (g^y, g^{\frac{1}{x+y}}, \sigma_{P3})$ on $m^\star$ at some point. Then, the forgery should satisfies the verification equation

$$e(g^{m^\star} g^y, \sigma_{P3}) = e(g, u') \tag{29}$$

and we obtain $\sigma_{P3} = (u')^{\frac{1}{y+m^\star}}$. Then, using the same method in [1], we convert $(u')^{\frac{1}{y+m^\star}}$ back to $g^{\frac{1}{y+m^\star}}$ and output $(m^\star, g^{\frac{1}{y+m^\star}})$ as the challenge of SDH2 problem. In the end, from Eqs. (25)-(29), we have

$$\epsilon' = \mathbf{Succ}_{\mathcal{A}}^{\mathrm{SDH2}} = \mathbf{Succ}_{\mathcal{SP},\mathcal{A}}^{\mathsf{ef\text{-}cma}} \geq \epsilon \tag{30}$$

In addition, we can obtain the claimed bound for $\tau' \leq \tau + \Theta(.)$, where $\Theta(.)$ is the time cost used for generating $(u', (u')^y, (u')^{\frac{1}{y+m_1}}, \cdots, (u')^{\frac{1}{y+m_{q_S}}})$ and $g^{\frac{1}{y+m^\star}}$. Thus, this completes the proof. □

In the following theorem, we will use the same approach from [2, 11] to prove the full anonymity of σ_A.

Theorem 3. (*Anonymity of* σ_A) Let $\mathcal{A}$ be an adversary against the proposed $\mathcal{SP}$ scheme in the standard model. Assume that $\mathcal{A}$ has the success probability $\mathbf{Succ}^{\mathsf{anony}}_{\mathcal{SP},\mathcal{A}} \geq \epsilon$ to break the full anonymity of σ_A within the running time τ. Then, there exist $\epsilon' \in [0,1]$ and $\tau' \in \mathbb{N}$ as follows

$$\epsilon' \geq \frac{\epsilon}{2}; \quad \tau' \leq \tau + \Theta(.) \tag{31}$$

such that SGD problem can be solved with probability ϵ' within time τ'.

Proof. We define a sequence of games Game_0, Game_1, $\cdots$ of modified attacks starting from the actual adversary $\mathcal{A}$ [18]. All the games operate on the same underlying probability space: the system parameters and master key, the coin tosses of $\mathcal{A}$. Let $(e, \mathbb{G}, \mathbb{G}_T, n, \tilde{h})$ be a random instance of SGD, we will use these incremental games to reduce the SGD instance to the adversary $\mathcal{A}$ against the full anonymity of σ_A in the proposed $\mathcal{SP}$ scheme.

Game_0: This is the real attack game. In the game, the SM chooses the master key (g^α, a, q) and the system public parameters $params = (\mathbb{G}, \mathbb{G}_T, e, n, g, u, h_1, h_2, e(g,g)^\alpha, A = g^a, h, H)$, and feeds the adversary $\mathcal{A}$ with these system public parameters. First, the adversary $\mathcal{A}$ exactly makes q_K queries to $\mathcal{O}_K$ on group members' private keys, q_S queries to $\mathcal{O}_S$ on anonymous signatures, and at some point, $\mathcal{A}$ chooses a message $M(=Y||\chi)$ and two groups (U_0, U_1) for challenge. Note that, the adversary $\mathcal{A}$ can know the private keys of (U_0, U_1) in the game. Then, we flip a coin $b \in \{0,1\}$ and produce U_b's anonymous signature $\sigma_A = (\sigma_1, \sigma_2, \sigma_3, \pi_1, \pi_2)$ as the challenge to the adversary $\mathcal{A}$. The anonymous signature comes from sk_i and three random numbers $z_1, z_2, z_3 \in \mathbb{Z}_n^*$, and $\sigma_1 = \rho_1 \cdot h^{z_1}$, $\sigma_2 = \rho_2 \cdot h^{z_2}$, $\sigma_3 = \rho_3 \cdot h^{z_3}$, $\pi_1 = \rho_2^{z_1}(A\rho_1)^{z_2}h^{z_1 z_2}$ and $\pi_2 = \rho_3^{z_1}(g^{H(M)}\rho_1)^{z_3}h^{z_1 z_3}$ with $(\rho_1, \rho_2, \rho_3) = \left(g^{s_b}, g^{\frac{1}{a+s_b}}, u^{\frac{1}{s_b+H(M)}}\right)$. Finally, the adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$. In any Game_j, we denote by Guess_j the event $b = b'$. Then, by definition, we have

$$\begin{aligned} \epsilon \leq \mathbf{Succ}^{\mathsf{anony}}_{\mathcal{SP},\mathcal{A}} = 2\Pr[b = b'] - 1 = 2\Pr[\mathsf{Guess}_0] - 1, \\ \Pr[\mathsf{Guess}_0] \geq \frac{\epsilon}{2} + \frac{1}{2}. \end{aligned} \tag{32}$$

Game_1: In this game, we modify the simulation by replacing the master key and system parameters with the SGD challenge $(e, \mathbb{G}, \mathbb{G}_T, n, \tilde{h})$. In specific, we choose $\alpha, a \in \mathbb{Z}_n^*$, compute the master key $(g^\alpha, a, \sqcup)$, and publish the systems parameters as $params = (\mathbb{G}, \mathbb{G}_T, e, n, g, u, h_1, h_2, e(g,g)^\alpha, A = g^a, \tilde{h}, H)$. Since the distribution of $(e(g,g)^\alpha, A = g^a, \tilde{h})$ is unchanged in the eye of the adversary, we thus have

$$\Pr[\mathsf{Guess}_1] = \Pr[\mathsf{Guess}_0] \tag{33}$$

Game_2: In this game, we will simulate the key generation oracle $\mathcal{O}_K$ on q_K group members' anonymous authentication keys queries and q_S anonymous signature oracle $\mathcal{O}_S$ queries. Because we have the master key $(g^\alpha, a, \sqcup)$, there is no difficult for us to simulate $\mathcal{O}_K$ and $\mathcal{O}_S$. Thus, this game is perfectly indistinguishable from the previous one, and we have

$$\Pr[\mathsf{Guess}_2] = \Pr[\mathsf{Guess}_1]. \tag{34}$$

Game_3: In this game, we consider, if $\tilde{h}$ in the SGD challenge $(e, \mathbb{G}, \mathbb{G}_T, n, \tilde{h})$ actually belongs to the subgroup $\mathbb{G}_q$, i.e., $\widetilde{b} = 0$ in experiment $\mathbf{Exp}^{\mathrm{SGD}}_{\mathcal{A}}$, we know this game is indistinguishable from the previous one, we have

$$\Pr[\mathsf{Guess}_3 | \widetilde{b} = 0] = \Pr[\mathsf{Guess}_2]. \tag{35}$$

and

$$\Pr\left[\mathbf{Exp}^{\mathrm{SGD}}_{\mathcal{A}} = 1 | \widetilde{b} = 0\right] = \Pr[\mathsf{Guess}_3 | \widetilde{b} = 0] \tag{36}$$

When $\tilde{h}$ doesn't belong to $\mathbb{G}_q$, i.e., $\tilde{h} \in \mathbb{G}$ and $\widetilde{b} = 1$ in experiment $\mathbf{Exp}^{\mathrm{SGD}}_{\mathcal{A}}$, $(\rho_1, \rho_2, \rho_3) = \left(g^{s_b}, g^{\frac{1}{a+s_b}}, u^{\frac{1}{s_b+H(M)}}\right)$ is masked by random numbers z_1, z_2, z_3 in $\mathbb{Z}_n^*$, i.e., $\sigma_1 = \rho_1 \cdot \widetilde{h}^{z_1}$, $\sigma_2 = \rho_2 \cdot \widetilde{h}^{z_2}$, $\sigma_3 = \rho_3 \cdot \widetilde{h}^{z_3}$, $(\sigma_1, \sigma_2, \sigma_3)$ reveals nothing about the challenge b. In this following, we show that (π_1, π_2) also doesn't reveal the challenge b.

For a specific $(\sigma_1, \sigma_2, \sigma_3)$, we know there exist random numbers $z_{1(b)}, z_{2(b)}, z_{3(b)}, z_{1(1-b)}, z_{2(1-b)}, z_{3(1-b)} \in \mathbb{Z}_n^*$ such that

$$\begin{cases} \sigma_1 = g^{s_b} \cdot \widetilde{h}^{z_{1(b)}} = g^{s_{1-b}} \cdot \widetilde{h}^{z_{1(1-b)}} \\ \sigma_2 = g^{\frac{1}{a+s_b}} \cdot \widetilde{h}^{z_{2(b)}} = g^{\frac{1}{a+s_{1-b}}} \cdot \widetilde{h}^{z_{2(1-b)}} \\ \sigma_3 = u^{\frac{1}{s_b+H(M)}} \cdot \widetilde{h}^{z_{3(b)}} = u^{\frac{1}{s_{1-b}+H(M)}} \cdot \widetilde{h}^{z_{3(1-b)}} \end{cases} \tag{37}$$

For example, suppose $\widetilde{h} = g^\eta = u^\xi$, $\varepsilon = \frac{a+s_b}{a+s_{1-b}}$, $\zeta = \frac{s_b+H(M)}{s_{1-b}+H(M)}$ for some unknown $\eta, \xi \in \mathbb{Z}_n^*$, we know

$$\begin{cases} z_{1(1-b)} = z_{1(b)} + \dfrac{s_b - s_{1-b}}{\eta} \\ z_{2(1-b)} = z_{2(b)} + \dfrac{1}{\eta}\left(\dfrac{1}{a+s_b} - \dfrac{1}{a+s_{1-b}}\right) \\ \qquad\quad = z_{2(b)} + \dfrac{1-\varepsilon}{\eta(a+s_b)} \\ z_{3(1-b)} = z_{3(b)} + \dfrac{1}{\xi}\left(\dfrac{1}{s_b+H(M)} - \dfrac{1}{s_{1-b}+H(M)}\right) \\ \qquad\quad = z_{3(b)} + \dfrac{1-\zeta}{\xi(s_b+H(M))} \end{cases} \tag{38}$$

Then,

$$\begin{cases} \pi_{1(b)} = g^{\frac{z_{1(b)}}{a+s_b}} g^{(a+s_b)z_{2(b)}} \widetilde{h}^{z_{1(b)}z_{2(b)}} \\ \pi_{1(1-b)} = g^{\frac{z_{1(1-b)}}{a+s_{1-b}}} g^{(a+s_{1-b})z_{2(1-b)}} \widetilde{h}^{z_{1(1-b)}z_{2(1-b)}} \end{cases} \tag{39}$$

$$\begin{aligned} &\log_g^{\pi_{1(1-b)}} \\ =&\frac{z_{1(b)} + \frac{s_b - s_{1-b}}{\eta}}{a+s_{1-b}} + (a+s_{1-b})\left(z_{2(b)} + \frac{1-\varepsilon}{\eta(a+s_b)}\right) \\ &+ \eta\left(z_{1(b)} + \frac{s_b - s_{1-b}}{\eta}\right)\left(z_{2(b)} + \frac{1-\varepsilon}{\eta(a+s_b)}\right) \\ =&\frac{z_{1(b)}}{a+s_{1-b}} + \frac{s_b - s_{1-b}}{\eta(a+s_{1-b})} + az_{2(b)} + s_{1-b}z_{2(b)} \\ &+ \frac{(1-\varepsilon)(a+s_b)}{\eta(a+s_b)} + \eta z_{1(b)}z_{2(b)} + s_b z_{2(b)} \\ &- s_{1-b}z_{2(b)} + \frac{z_{1(b)}(1-\varepsilon)}{a+s_b} + \frac{(1-\varepsilon)(s_b - s_{1-b})}{\eta(a+s_b)} \\ =&\frac{z_{1(b)}}{a+s_b} + (a+s_b)z_{2(b)} + \eta z_{1(b)}z_{2(b)} = \log_g^{\pi_{1(b)}} \\ \Rightarrow&\pi_{1(b)} = \pi_{1(1-b)} \end{aligned} \tag{40}$$

In addition,

$$\begin{cases} \pi_{2(b)} = u^{\frac{z_{1(b)}}{s_b+H(M)}} g^{(s_b+H(M))z_{3(b)}} \widetilde{h}^{z_{1(b)}z_{3(b)}} \\ \pi_{2(1-b)} = u^{\frac{z_{1(1-b)}}{s_{1-b}+H(M)}} g^{(s_{1-b}+H(M))z_{3(1-b)}} \widetilde{h}^{z_{1(1-b)}z_{3(1-b)}} \end{cases} \tag{41}$$

$$\begin{aligned} &\log_g^{\pi_{2(1-b)}} \\ &= \left(\frac{z_{1(b)} + \frac{s_b - s_{1-b}}{\eta}}{s_{1-b}+H(M)}\right)\frac{\eta}{\xi} + (s_{1-b}+H(M)) \\ &\quad \cdot \left(z_{3(b)} + \frac{1-\zeta}{\xi(s_b+H(M))}\right) + \eta\left(z_{1(b)} + \frac{s_b - s_{1-b}}{\eta}\right) \\ &\quad \cdot \left(z_{3(b)} + \frac{1-\zeta}{\xi(s_b+H(M))}\right) \\ &= \left(\frac{z_{1(b)}}{s_{1-b}+H(M)}\right)\frac{\eta}{\xi} + \left(\frac{s_b - s_{1-b}}{s_{1-b}+H(M)}\right)\frac{1}{\xi} \\ &\quad + s_{1-b}\cdot z_{3(b)} + H(M)\cdot z_{3(b)} + \left(\frac{s_{1-b}+H(M)}{s_b+H(M)}\right)\frac{1-\zeta}{\xi} \\ &\quad + \eta z_{1(b)}z_{3(b)} + s_b z_{3(b)} + s_{1-b}z_{3(b)} \\ &\quad + \left(\frac{z_{1(b)}(1-\zeta)}{s_b+H(M)}\right)\frac{\eta}{\xi} + \left(\frac{s_b - s_{1-b}}{s_b+H(M)}\right)\frac{1-\zeta}{\xi} \\ &= \left(\frac{z_{1(b)}}{s_b+H(M)}\right)\frac{\eta}{\xi} + (s_b+H(M))z_{3(b)} + \eta z_{1(b)}z_{3(b)} \\ &= \log_g \pi_2(b) \\ &\Rightarrow \pi_2(b) = \pi_2(1-b) \end{aligned} \tag{42}$$

Since $\pi_1(b) = \pi_1(1-b)$ and $\pi_2(b) = \pi_2(1-b)$ are independent of b, we have

$$\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{SGD}} = 1|\widetilde{b} = 1\right] = \Pr[\mathsf{Guess}_3|\widetilde{b} = 1] = \frac{1}{2} \tag{43}$$

As a result, from Eqs. (32)-(43), we have

$$\begin{aligned} \epsilon' &= \mathbf{Adv}_{\mathcal{A}}^{\mathrm{SGD}} \\ &= \left|\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{SGD}} = 1|\widetilde{b} = 0\right] - \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{SGD}} = 1|\widetilde{b} = 1\right]\right| \\ &\geq \left|\frac{\epsilon}{2} + \frac{1}{2} - \frac{1}{2}\right| = \frac{\epsilon}{2} \end{aligned} \tag{44}$$

In addition, we can obtain the claimed bound for $\tau' \leq \tau + \Theta(.)$, where $\Theta(.)$ is the time cost used for queries on $\mathcal{O}_K$ and $\mathcal{O}_S$ in the simulation. Thus, the proof is completed. □

In the anonymous signature $\sigma_A = (\sigma_1, \sigma_2, \sigma_3, \pi_1, \pi_2)$, the elements $(\sigma_1, \sigma_2, \sigma_3)$ are derived from $\rho = (\rho_1, \rho_2, \rho_3)$ which has the same form of $\sigma_P = (\sigma_{P1}, \sigma_{P2}, \sigma_{P3})$. In the following, we use the same approach in *Theorem 2* to prove the full traceability of σ_A under the SDH2 assumption, the only difference is the adversary can also have the master key q for tracking in the game. Note that the following theorem, like *Theorem 2*, is also carried out under a weak chosen message attack [1].

Theorem 4. (*Traceability of* σ_A) Let $\mathcal{U} = \{U_1, U_2, \cdots\}$ be a group of registered users and SP be the service provider. Let $\mathcal{A}$ be an ef-cma adversary against the proposed $\mathcal{SP}$ scheme in the standard model. Assume that $\mathcal{A}$ has the success probability $\mathbf{Succ}_{\mathcal{SP},\mathcal{A}}^{\mathsf{trace}} \geq \epsilon$ to existentially forge the anonymous signature σ_A (in a weak chosen message attack), within the running time τ, after making q_K, q_S queries to the key generation oracle $\mathcal{O}_K$ and the signature oracle $\mathcal{O}_S$, respectively. Then, there exist $\epsilon' \in [0,1]$ and $\tau' \in \mathbb{N}$ as follows

$$\epsilon' \geq \epsilon; \quad \tau' \leq \tau + \Theta(*) \tag{45}$$

such that SDH2 problem can be solved with probability ϵ' within time τ', where $\Theta(*)$ is the time cost in the simulation.

Proof. We define a sequence of games Game_0, Game_1, $\cdots$ of modified attacks starting from the actual adversary $\mathcal{A}$ [18]. All the games operate on the same underlying probability space: the system parameters and master key, the coin tosses of $\mathcal{A}$. Let

$$\begin{pmatrix} g, g^x, g^y, g^{\frac{1}{x+y}} \\ c_1, g^{\frac{1}{x+c_1}}, c_2, g^{\frac{1}{x+c_2}}, \cdots, c_{l_1}, g^{\frac{1}{x+c_{q_K}}} \\ g^{(y^2)}, \cdots, g^{(y^{q_S})} \end{pmatrix}$$

be a random instance of SDH2, where g be an element of the subgroup $\mathbb{G}_p$ of $\mathbb{G}$, we will use these incremental games to reduce the SDH2 instance to the adversary $\mathcal{A}$ against the existential forgery under a weak chosen message attack in the proposed $\mathcal{SP}$ scheme.

Game_0: This is the real attack game. In the game, the SM chooses the master key (g^α, a, q) and the system public parameters $params = (\mathbb{G}, \mathbb{G}_T, e, n, g, u, h_1, h_2, e(g,g)^\alpha, A = g^a, h, H)$, and feeds the adversary $\mathcal{A}$ with these system public parameters and the master key q for tracking. First, the adversary $\mathcal{A}$ outputs a list of q_S distinct messages $m_1, m_2, \cdots, m_{q_S} \in \mathbb{Z}_n^*$, where $m_i = H(M_i)$ for $i = 1, 2, \cdots, q_S$, exactly makes q_K queries to $\mathcal{O}_K$ on group members' private keys (anonymous authentication keys), q_S queries to $\mathcal{O}_S$ on a specific user $U^\star$'s signatures, where $U^\star$ has not been queried on $\mathcal{O}_K$. Note that the signatures queried from $\mathcal{O}_S$ can be traced by $\mathcal{A}$ with q and tracking list. In the end, the adversary $\mathcal{A}$ outputs a valid signature $(\sigma_A^\star, m^\star)$ of $U^\star$, where $m^\star$ is not queried to $\mathcal{O}_S$ before. In any Game_j, we denote by $\Pr[\mathsf{Forge}_j]$ the forgery success probability of $\mathcal{A}$ in Game_j. Then, by definition, we have

$$\Pr[\mathsf{Forge}_0] = \mathbf{Succ}_{\mathcal{SP},\mathcal{A}}^{\mathsf{trace}} \geq \epsilon \tag{46}$$

Game_1: In this game, we confine the elements (g, u) in the subgroup $\mathbb{G}_p$, i.e., $g^p = 1$, $u^p = 1$. Since $g, u \in \mathbb{G}_p$ still belong to $\mathbb{G}$ in accord with the scheme design, the adversary $\mathcal{A}$ can't detect this trick. Therefore, we have

$$\Pr[\mathsf{Forge}_1] = \Pr[\mathsf{Forge}_0] \tag{47}$$

Game_2: In this game, we modify the simulation by replacing the system parameter $(A = g^a, u)$ with (g^x, u'), where u' is generated by adopting the same simulating approach in [1], i.e., given $(g^y, g^{(y^2)}, \cdots, g^{(y^{q_S})}$ and $m_1, m_2, \cdots, m_{q_S} \in \mathbb{Z}_n^*$, we can generate $(u', (u')^y, (u')^{\frac{1}{y+m_1}}, \cdots, (u')^{\frac{1}{y+m_{q_S}}})$. Since the distribution of $(A = g^a, u)$ is unchanged, we thus have

$$\Pr[\mathsf{Forge}_2] = \Pr[\mathsf{Forge}_1] \tag{48}$$

Game_3: In this game, we will simulate the key generation oracle $\mathcal{O}_K$ on q_K group members' anonymous authentication keys queries. Concretely, when a fresh query on user $U_i \in \mathcal{U}$ is queried, we randomly chooses an un-queried c_i, records $(U_i, c_i, g^{c_i q})$, and return $(c_i, g^{\frac{1}{x+c_i}})$ as the answer to

the oracle query. Since $(c_i, g^{\frac{1}{x+c_i}})$ is uniformly distributed, this game is therefore perfectly indistinguishable from the previous one. Hence,

$$\Pr[\mathsf{Forge}_3] = \Pr[\mathsf{Forge}_2]. \tag{49}$$

Game_4: In this game, we simulate total q_s times signing oracle $\mathcal{O}_S$. Note that, since the adversary $\mathcal{A}$ has already gotten other group members' private keys, $\mathcal{A}$ only queries the signing oracle $\mathcal{O}_S$ on the challenged U^*. For any fresh query on m_i, we first pick the corresponding $(u')^{\frac{1}{y+m_i}}$, set $\rho = (\rho_1, \rho_2, \rho_3) = (g^y, g^{\frac{1}{x+y}}, (u')^{\frac{1}{y+m_i}})$. Then, we choose three random numbers $z_1, z_2, z_3 \in \mathbb{Z}_n^*$, and compute $\sigma_1 = \rho_1 \cdot h^{z_1}$, $\sigma_2 = \rho_2 \cdot h^{z_2}$, $\sigma_3 = \rho_3 \cdot h^{z_3}$; and $\pi_1 = \rho_2^{z_1}(A\rho_1)^{z_2} h^{z_1 z_2}$, $\pi_2 = \rho_3^{z_1}(g^{m_i}\rho_1)^{z_3} h^{z_1 z_3}$. In the end, we return $\sigma_A = (\sigma_1, \sigma_2, \sigma_3, \pi_1, \pi_2)$ as the answer to the oracle query. In this game, $\mathcal{O}_S$ perfectly simulates the anonymous signature, we will have

$$\Pr[\mathsf{Forge}_4] = \Pr[\mathsf{Forge}_3]. \tag{50}$$

We consider the adversary $\mathcal{A}$ outputs a valid anonymous signature $\sigma_A^\star = (\sigma_1^\star, \sigma_2^\star, \sigma_3^\star, \pi_1^\star, \pi_2^\star)$ on $(m^\star, U^\star)$ at some point. Then, the forgery should satisfies the verification equations

$$\begin{aligned} e(\sigma_1^\star A, \sigma_2^\star) &= e(g,g)e(\pi_1^\star, h) \\ e(\sigma_1^\star g^{m^\star}, \sigma_3^\star) &= e(g,u)e(\pi_2^\star, h) \end{aligned} \tag{51}$$

We generate a random number λ such that $\lambda \equiv 1 \bmod p$ and $\lambda \equiv 0 \bmod q$. Then, we know $g^\lambda = g$, $u^\lambda = u$ and $h^\lambda = 1$. Thus, from Eq. (51), we have

$$\begin{aligned} & e(\sigma_1^\star A, \sigma_2^\star)^{\lambda^2} = (e(g,g)e(\pi_1^\star, h))^{\lambda^2} \\ \Rightarrow & e(\sigma_1^{\star\lambda} A^\lambda, \sigma_2^{\star\lambda}) = e(g^\lambda, g^\lambda) e(\pi_1^{\star\lambda}, h^\lambda) \\ \Rightarrow & e(\sigma_1^{\star\lambda} A, \sigma_2^{\star\lambda}) = e(g,g) \end{aligned} \tag{52}$$

$$\begin{aligned} & e(\sigma_1^\star g^{m^\star}, \sigma_3^\star)^{\lambda^2} = (e(g,u)e(\pi_2^\star, h))^{\lambda^2} \\ \Rightarrow & e(\sigma_1^{\star\lambda} g^{m^\star\lambda}, \sigma_3^{\star\lambda}) = e(g^\lambda, u^\lambda) e(\pi_2^{\star\lambda}, h^\lambda) \\ \Rightarrow & e(\sigma_1^{\star\lambda} g^{m^\star}, \sigma_3^{\star\lambda}) = e(g,u) \end{aligned} \tag{53}$$

and can get $\sigma_1^{\star\lambda} = g^y$, $\sigma_2^{\star\lambda} = g^{\frac{1}{x+y}}$, and $\sigma_3^{\star\lambda} = (u)^{\frac{1}{y+m^\star}} = (u')^{\frac{1}{y+m^\star}}$. Then, using the same approach in [1], we convert $(u')^{\frac{1}{y+m^\star}}$ back to $g^{\frac{1}{y+m^\star}}$ and output $(m^\star, g^{\frac{1}{y+m^\star}})$ as the challenge of SDH2 problem. In the end, from Eqs. (46)-(53), we have

$$\epsilon' = \mathbf{Succ}_{\mathcal{A}}^{\mathrm{SDH2}} = \mathbf{Succ}_{\mathcal{SP},\mathcal{A}}^{\mathsf{trace}} \geq \epsilon \tag{54}$$

In addition, we can obtain the claimed bound for $\tau' \leq \tau + \Theta(*)$, where $\Theta(*)$ is the time cost used for generating $(u', (u')^y, (u')^{\frac{1}{y+m_1}}, \cdots, (u')^{\frac{1}{y+m_{q_S}}})$ and $g^{\frac{1}{y+m^\star}}$. Thus, this completes the proof. □

Data Forensics in Cloud Computing. Based on the above security proofs, the proposed secure provenance $\mathcal{SP}$ scheme can guarantee the information confidentiality, anonymous authentication, authorized access and provenance tracking. Thus, it can be used for data forensics in cloud computing. As shown in Figure 2, if a document is normally stored in cloud, the ownership and process history of the document is privacy. However, if it is in dispute, $\mathcal{SP}$ can provide all provenance information related to all versions of the document to SM. Then, SM can use the provenance tracking algorithm to plot a visible provenance chain to track the specific user identity.

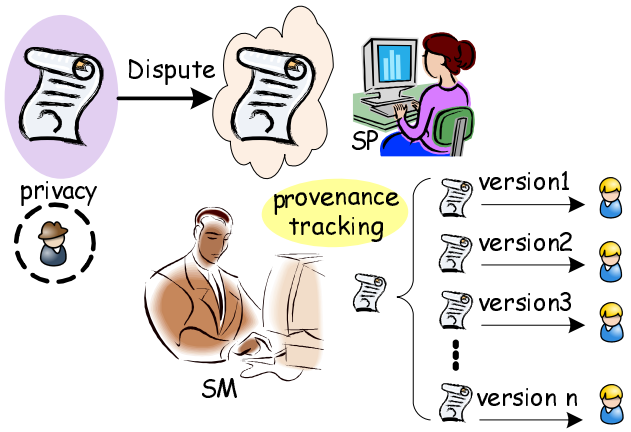

Figure 2: Data forensics in cloud computing

6. CONCLUSIONS

Secure provenance is of paramount importance to the flourish of cloud computing, yet it is still challenging today. In this paper, we formally defined the secure provenance and the corresponding security model in cloud computing. Then, we proposed a concrete secure provenance $\mathcal{SP}$ scheme based on the bilinear pairings, and used the provable security technique to prove its security in the standard model. Due to its comprehensive security features, the proposed $\mathcal{SP}$ scheme provides trusted evidences for data forensics in cloud computing and thus pushes the cloud computing for wide acceptance to the public.

7. REFERENCES

[1] Boneh, D., and Boyen, X. Short signatures without random oracles and the sdh assumption in bilinear groups. *Journal of Cryptology 21*, 2 (2008), 149–177.

[2] Boyen, X., and Waters, B. Full-domain subgroup hiding and constant-size group signatures. In *Lecture Notes in Computer Science, PKC 2007* (2007), pp. 1–15.

[3] Chai, Z., Cao, Z., and Lu, R. Efficient password-based authentication and key exchange scheme preserving user privacy. In *Lecture Notes in Computer Science, Wireless Algorithms, Systems, and Applications* (2006), vol. 4138, pp. 467–477.

[4] Erdogmus, H. Cloud computing: Does nirvana hide behind the nebula? *IEEE Software 11*, 2 (March-April 2009), 4–6.

[5] Foster, I., Zhao, Y., Raicu, I., and Lu, S. Cloud computing and grid computing 360-degree compared. In *Proceedings of Grid Computing Environments Workshop, GCE'08* (Austin, TX, 2008), pp. 1–10.

[6] Gellman, R. Privacy in the clouds: Risks to privacy and confidentiality from cloud computing. Tech. rep., February 2009. http://www.worldprivacyforum.org.

[7] Goldwasser, S., Micali, S., and Rivest, R. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing 17*, 2 (1988), 281–308.

[8] Hartig, K. What is cloud computing? website, April 2009. http://kevinhartig.sys-con.com/.

[9] Hasan, R., Sion, R., and Winslett, M. Introducing secure provenance: problems and challenges. In *Proceedings of ACM workshop on Storage security and survivabilit, StorageSS'07* (Alexandria, Virginia, USA, October 2007), pp. 13–18.

[10] Kaufman, L. M. Data security in the world of cloud computing. *IEEE Security & Privacy 7*, 4 (July-Aug. 2009), 61–64.

[11] Liang, X., Cao, Z., Shao, J., and Lin, H. Short group signature without random oracles. In *Lecture Notes in Computer Science, ICICS 2007* (2007), pp. 69–82.

[12] Lin, X., Sun, X., Ho, P.-H., and Shen, X. GSIS: a secure and privacy-preserving protocol for vehicular communication. *IEEE Transactions on Vehicular Technology 56*, 6 (2007), 3442–3456.

[13] Lu, R., Lin, X., and Shen, X. Spring: A social-based privacy-preserving packet forwarding protocol for vehicular delay tolerant networks. In *The 29th IEEE International Conference on Computer Communications (INFOCOM 2010)* (San Diego, California, USA, March 2010).

[14] Lu, R., Lin, X., Zhu, H., Ho, P.-H., and Shen, X. ECPP: Efficient conditional privacy preservation protocol for secure vehicular communications. In *The 27th Conference on Computer Communications (INFOCOM 2008)* (Phoenix, Arizona, USA, April 2008), pp. 1229–1237.

[15] Lu, R., Lin, X., Zhu, H., and Shen, X. Spark: a new vanet-based smart parking scheme for large parking lots. In *The 28th Conference on Computer Communications (INFOCOM 2009)* (Rio de Janeiro, Brazil, April 2009).

[16] Lynch, C. A. When documents deceive: Trust and provenance as new factors for information retrieval in a tangled web. *Journal of the American Society for Information Sciene and Technology 52*, 1 (2001), 12–17.

[17] Mei, L., Chan, W., and Tse, T. A tale of clouds: Paradigm comparisons and some thoughts on research issues. In *Proceedings of Asia-Pacific Services Computing Conference, APSCC'08* (Yilan, Dec. 2008), pp. 464–469.

[18] Shoup, V. Oaep reconsidered. *Journal of Cryptology 15*, 4 (2002), 223–249.

[19] Sterling, T., and Stark, D. A high-performance computing forecast: Partly cloudy. *Computing in Science & Engineering 11*, 4 (July-Aug. 2009), 42–49.

[20] Voas, J., and Zhang, J. Cloud computing: New wine or just a new bottle? *IT Professional 11*, 2 (March-April 2009), 15–17.

[21] Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. Cryptology ePrint Archive, Report 2008/290, 2008.