Journal of Computer and System Sciences 79 (2013) 1200–1213

Contents lists available at SciVerse ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

How to achieve non-repudiation of origin with privacy protection in cloud computing

Wei Wu[a], Jianying Zhou[b], Yang Xiang[c,*], Li Xu[a]

[a] School of Mathematics and Computer Science, Fujian Normal University, Fuzhou, China
[b] Institute for Infocomm Research (I^{2}R), Singapore
[c] School of Information Technology, Deakin University, Australia

ARTICLE INFO

Article history:
Received 16 December 2011
Received in revised form 11 April 2012
Accepted 22 February 2013
Available online 26 March 2013

Keywords:
Security
Non-repudiation
Privacy
Protocol

ABSTRACT

This paper studies a security issue in cloud computing: non-repudiation of origin (NRO) with privacy protection on message originator. We first define two concrete goals of NRO: NRO-I and NRO-II. Both notions are inspired by the non-repudiation service we can have by using traditional handwritten signatures as the evidence of origin. Then we show that existentially unforgeable digital signatures can provide NRO-I but not always NRO-II, by giving a counterexample. Another contribution of this paper is a communication protocol accommodating non-repudiation of origin and privacy of message originator. Our protocol satisfies NRO-I and NRO-II, and the recipient is unable to convince any other entities about the identity of message originator. The essence of our protocol is a designated verifier signature scheme with unforgeability against the designated verifier.

1. Introduction

Disputes could occur on cloud computing. A typical example is a message transfer from Alice to Bob, where Alice claims that she has sent the message M to Bob but Bob denies having received it, or Bob claims that he received M from Alice but Alice denies sending it. Another example is that one of the eligible (but malicious) clients modifies the data stored in the cloud but denies that fact. Non-repudiation is a security service supporting the settlement of those disputes by creating, collecting, validating, and maintaining cryptographic evidence in electronic transactions [27,29] and many other systems (e.g., [25,26]). It is a similar notion to authentication but has stronger proof requirements. Authentication only needs to convince the other party on the communication about the validity of an event, while non-repudiation should be able to prove to a third party the truth of an event. Taking the message handling system for example, there are three kinds of non-repudiation services that should be provided by the originator, the delivery agent(s) and the recipient(s), respectively [29]:

1. *Non-Repudiation of Origin (NRO)* provides the recipient(s) of a message with the proof of the origin of the message. It will protect against any attempt by the originator to falsely deny sending the message.
2. *Non-Repudiation of Delivery (NRD)* provides the originator of a message with the proof that the message has been delivered to the originally specified recipient(s).
3. *Non-Repudiation of Receipt (NRR)* provides the originator of a message with the proof of receipt of the message. It will protect against any attempt by the recipient(s) to falsely deny receiving the message.

* Corresponding author.
E-mail addresses: weiwu81@gmail.com (W. Wu), jyzhou@i2r.a-star.edu.sg (J. Zhou), yang.xiang@deakin.edu.au (Y. Xiang), xuli@fjnu.edu.cn (L. Xu).

0022-0000/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcss.2013.03.001

Related work. A non-repudiation service is made up of four phases [28]:

1. *Evidence Generation*: Depending on concrete designs and applications, the evidence could be generated by the originator, the recipient, the delivery agent, or the trusted third party.
2. *Evidence Transfer*: This phase provides fairness, efficiency and timeliness required in the communication.
3. *Evidence Verification and Storage*: Newly received evidence should be verified to gain confidence that the supplied evidence will indeed be adequate in the event of a dispute. Valid evidence must be stored safely.
4. *Dispute Resolution*: When a dispute occurs, an adjudicator will be invoked to settle the dispute according to the non-repudiation evidence provided by the disputing parties.

Below we only review the related work on evidence generation and verification, which are most relevant to this paper. A comprehensive review of all phases can be found in [28]. There are two types of security mechanisms for generating non-repudiation evidence: secure envelopes and digital signatures.

A secure envelope provides protection of the origin and the integrity of a message based on a shared secret key between communication parties. But such a protection alone is not able to provide non-repudiation service, since each party, with the shared key, can change the message content in the secure envelope or deny its origin. To make a secure envelope become irrefutable evidence, it must be generated by a trusted third party (TTP) using a secret key known only to TTP. Other parties are unable to verify such evidence directly but are assured of its validity through the mediation of TTP. Such a mechanism requires a full trust on TTP [28].

Another type of non-repudiation evidence is digital signature. Digital signature is one of the most important applications of public key cryptography (or, asymmetric key cryptography) [6], where each entity has a pair of keys, a public key and a private key. The binding between a public key and its owner is ensured by the certificate generated by a third party called certificate authority (CA) when necessary. One must use his/her private key to produce a valid digital signature, and the signature verification needs the associated public key. Like the traditional handwritten signature, a valid digital signature can assure the recipient the origin and the integrity of a message, and has received legal recognition [8]. For the purpose of non-repudiation, digital signatures must satisfy several security requirements, including the cryptographic strength of signature algorithms (e.g., existential unforgeability) and the management of signing keys (e.g., key revocation) and the device for signature generation [28]. Compared with non-repudiation evidence using secure envelopes, digital signatures require less trust on the third party and are widely accepted in situations need non-repudiation services [12].

Motivation and contribution. The motivation of this paper is to find a solution accommodating both non-repudiation and privacy, since privacy-aware security is increasingly recognized by academia and industry community (e.g., [11]). As a case study, our focus is on the non-repudiation of origin and the privacy of origin.

In Section 2, we define two security goals of the non-repudiation of origin.

1. NRO-I provides security assurance against recipient: It must be hard for the recipient to produce the evidence of origin which can prove another entity as the message sender.
2. NRO-II provides security assurance against message sender(s): It must be hard for two entities, even sharing all secret information, to produce the evidence which can prove each of them as the message sender. In other words, NRO-II defines the unambiguity of the evidence of origin against conspiring attacks.

In physical world, NRO-I and NRO-II can be satisfied using traditional handwritten signatures as the evidence of origin (under certain conditions).

Section 3 studies NRO-I and NRO-II in the digital world, using digital signatures as the evidence of origin. We present two communication protocols, both of which make use of strongly existentially unforgeable signatures as the evidence of origin. The first protocol satisfies both NRO-I and NRO-II, but the second one only provides NRO-I. This shows that the (strongly) existential unforgeability of digital signatures is not sufficient to provide NRO-II defined in this paper.

Section 4.1 defines the privacy of message sender in communication protocols with non-repudiation of origin. In a message transfer from Alice to Bob, our definition requires that:

1. Without the assistance of Alice, it is hard for any other entity to tell whether Alice is the message originator, even given the evidence generated by Alice. In other words, the validity of the evidence of origin is not publicly verifiable.
2. To deny her as the message originator, Alice must generate a publicly verifiable proof to prove that a claimed evidence (or more generally, a bit-string) is invalid, and vice versa.
3. Alice must convince Bob that the evidence of origin generated during the communication is valid and she cannot deny sending the message. However, such a conviction is non-transferable, even Bob shares all his secret information with others.

Section 4.3 describes a communication protocol where the non-repudiation of origin and the privacy of message originator co-exist, with formal proofs in the random oracle model. The design philosophy is allowing the recipient to generate a special kind of evidence of origin, which is similar to but different from the actual one generated by the originator. The difference is only computational, namely no one except the originator can distinguish those two kinds of evidence and the originator can issue a publicly verifiable proof to show whether the evidence is generated by the recipient or by the originator. In particular, for any evidence generated by the originator, it cannot produce a proof to deny that fact. As we will show shortly, the essence of our design is a designated verifier signature scheme with unforgeability against the designated verifier.

2. Definitions of non-repudiation of origin

This section defines two goals of non-repudiation of origin.

2.1. Case definition

For our purpose, the communication protocol studied in this paper only involves two entities, a message sender and a message recipient. They have a synchronized clock. The protocol is defined as follows.

1. *Recipient-Initialization*.
   Given a security number κ, the recipient generates his/her public parameter *pk-R* and (optionally) the relevant secret parameter *sk-R*.
2. *Sender-Initialization*.
   This phase generates the public/secret information, *pk-S*/*sk-S*, for the sender.
3. $\mathit{Sender}\{\mathit{pk}\text{-}R, \mathit{sk}\text{-}S, m\} \leftrightharpoons \mathit{Recipient}\{\mathit{pk}\text{-}S, \mathit{sk}\text{-}R, \mathit{time}\}$.
   An entity sends a message m to a recipient, together with the evidence of origin ϖ. The recipient verifies ϖ and stores the valid evidence in a secure place.
   The symbol "$\leftrightharpoons$" indicates that it may involve multi-round message exchanges between the sender and the recipient. The parameters in curly brackets $\{\dots\}$ are those needed during the communication.

According to ISO/IEC 13888-1 [12], evidence is information that either by itself or when used in conjunction with other information is used to establish proof about an event or action. If the evidence itself is sufficient to establish the proof of origin (e.g., handwritten signature), anyone with the evidence will find who sends the message and this may compromise the privacy of message sender. (Its formal definition shall be given shortly.) Thus, to protect the privacy of message sender, the evidence alone must not be able to generate the proof. In this case, an additional piece of information is necessary to verify the (in)validity of the evidence when disputes occur. This motivates the introduction of the following two additional phasesA non-repudiation service is made up.

4. $\mathit{Public\text{-}Proof\text{-}Generation}\{m, \mathit{pk}\text{-}R, \mathit{pk}\text{-}S, \mathit{sk}\text{-}S, \mathit{time}, \varpi\} \to \pi$.
   Given a triple $(m, \mathit{pk}\text{-}S, \mathit{time})$, *pk-R*, *sk-S* and a bit-string ϖ, one can generate another bit-string π which will be used to verify the validity of ϖ. This algorithm is only needed if the evidence ϖ is not publicly verifiable.
5. $\mathit{Public\text{-}Verification}\{m, \mathit{pk}\text{-}R, \mathit{pk}\text{-}S, \mathit{time}, \varpi, \pi\} \to \{0, 1\}$.
   Given a triple $(m, \mathit{pk}\text{-}S, \mathit{time})$, *pk-R*, a bit-string ϖ and (optional) another bit-string π generated by the *Public-Proof-Generation*, this algorithm outputs "1" if the owner of *pk-S* is the sender of m at *time*. Otherwise, this algorithm outputs "0". A triple $(m, \mathit{pk}\text{-}S, \mathit{time})$ is said to be valid if this algorithm outputs "1".
   To make it fair: (1) *Public-Verification* will never output "0" if ϖ is generated by the message sender and accepted by the recipient during the communication, and (2) To deny him/her as the message sender, one must produce a bit-string π such that *Public-Verification* outputs "0".

Remark 1. Our definition above requires that the message recipient should complete the initialization prior to the message sender. This is due to the concern that the production of *pk-S*/*sk-S* may need *pk-R* and/or *sk-R* if *Sender-Initialization* is an interactive process between the sender and the recipient.

Remark 2. *Public-Proof-Generation* requires *sk-S* as the input. The aim is to provide the privacy protection on message sender: The only information can prove/disprove Alice as the message sender is the proof generated by Alice.

2.2. NRO-I: untrusted recipient

The first type of non-repudiation of origin, which we call "NRO-I", prevents the recipient from producing any valid evidence of origin. The goal is to avoid disputes caused by inside threats, e.g., a malicious service provider or its staff forges a service request on behalf of its client such that it has a good reason to make a service charge. In other words, NRO-I addresses the issue of an untrusted recipient. Suppose there are N users $\{S_1, S_2, \dots, S_N\}$ that may send messages to a recipient R, NRO-I is defined as follows.

Game of NRO-I:

Setup. This phase consists of the following three steps:

1. The adversary first runs *Recipient-Initialization* to create a pair ($pk\text{-}R$, $sk\text{-}R$) for the recipient. $pk\text{-}R$ is sent to the challenger.
2. The challenger then runs *Sender-Initialization* N times to generate ($pk\text{-}S_1$, $sk\text{-}S_1$), ($pk\text{-}S_2$, $sk\text{-}S_2$), ..., and ($pk\text{-}S_N$, $sk\text{-}S_N$) for the senders. The adversary may be involved in this process by acting as the recipient if *Sender-Initialization* is an interactive protocol between the sender and the recipient.
3. Without loss of generality, the adversary is given $\{pk\text{-}S_1, (pk\text{-}S_2, sk\text{-}S_2), \ldots, (pk\text{-}S_N, sk\text{-}S_N)\}$.

Queries. The adversary can make up to q queries:

1. For a communication query m_i at $time_i$, the protocol $Sender\{pk\text{-}R, sk\text{-}S_1, m_i\} \leftrightharpoons Recipient\{pk\text{-}S_1, sk\text{-}R, time_i\}$ will be executed, where the challenger acts as the sender and the adversary acts as the recipient.
   Notice that (1) the adversary may not follow the protocol specifications during the execution, and (2) with $sk\text{-}S_2$, $sk\text{-}S_3$, ..., $sk\text{-}S_N$ and $sk\text{-}R$, the adversary is able to simulate all executions if the sender is any of $\{S_2, S_3, \ldots, S_N\}$.
2. For a query with the form $(m_i, time_i, \varpi_i)$, the challenger responds with a proof π by running the algorithm *Public-Proof-Generation*.

Output. The adversary outputs $(m^*, \varpi^*, time^*)$ and wins the game if $(m^*, time^*)$ has never appeared as a communication query and (m^*, $pk\text{-}S_1, time^*$) is a valid triple.

The success probability that the adversary has in the game is denoted by ϵ.

Definition 1. A communication protocol is said to provide (t, q, ϵ)-NRO-I if no adversary, by making q queries in time t, can win the game of NRO-I with success probability more than ϵ.

Remark 3. As one can see, a necessary condition for NRO-I is that the recipient does not have the secret information of the sender.

2.3. NRO-II: unambiguous evidence

The other kind of non-repudiation of origin, NRO-II, is about the unambiguity of the evidence, namely the evidence of origin can identify only one entity as the message originator, and it is never possible that the evidence is valid for two or more entities (even these entities share all secret information).

NRO-II is inspired by the handwritten signature, a traditional way to achieve non-repudiation. While one can forge a handwritten signature of someone else, it is difficult to produce a handwritten signature which belongs to two different persons, under the assumption that each person has a unique signature template at the registration phase. We believe such a property should be preserved in the cyber world. NRO-II is defined as follows.

Game of NRO-II:

Setup. The challenger generates ($pk\text{-}R$, $sk\text{-}R$) for the recipient R, and (pk_A, sk_A) and (pk_B, sk_B) for two potential message senders. The adversary is given all three key pairs, i.e., ($pk\text{-}R$, $sk\text{-}R$), (pk_A, sk_A) and (pk_B, sk_B), and thus does not need to make any queries.

Output.

1. The adversary selects a message m^* and executes the protocol $Sender\{pk\text{-}R, sk_A, m^*\} \leftrightharpoons Recipient\{pk_A, sk\text{-}R, time^*\}$ with the challenger at $time^*$. During the execution, the adversary acts as the message sender and the challenger acts as the recipient.
2. Let ϖ^* be the evidence generated during the communication.
3. The adversary wins the game if ϖ^* can prove that
   - (m^*, pk_A, $time^*$) is a valid triple; and
   - (m^*, pk_B, $time^*$) is a valid triple.

The success probability that the adversary has in the game is denoted by ϵ.

Definition 2. A communication protocol is said to provide (t, ϵ)-NRO-II if no adversary can win the game of NRO-II in time t with success probability more than ϵ.

Remark 4. In definition of NRO-II, the adversary is given the secret information of all entities. This is to simulate the non-repudiation we can have using handwritten signatures as the evidence of origin: Even given the ability to forge the evidence of origin, it is still difficult to create the evidence valid for two different senders.

3. Non-repudiation of origin and existentially unforgeable digital signatures

Like handwritten signatures in the physical world, digital signatures can be used as the non-repudiation evidence in the cyber world [12], as long as they are secure. The security of digital signatures relies not only on the cryptographic strength of signature algorithms, but also on the management of signature keys as well as the device for signature generation [28]. The focus of this section is on the cryptographic strength of signature algorithms. Please refer to [28] for the investigation of the other two issues.

This section first reviews the basic security requirement of digital signatures, i.e., the existential unforgeability. After that, we study the relationship between existential unforgeability and non-repudiation of origin. This is demonstrated by two communication protocols:

1. `Protocol I` and `Protocol II` are designed under the same framework but use different existentially unforgeable digital signatures as the evidence of origin.
2. `Protocol I` satisfies NRO-I and NRO-II defined in Section 2, but `Protocol II` only provides NRO-I.

Our conclusion is that there are existentially unforgeable digital signatures providing non-repudiation of origin defined in Section 2, but existential unforgeability cannot ensure the non-repudiation of origin.

3.1. Digital signature

In public key cryptography [6], a user is equipped with a pair of cryptographic keys: a private key and a public key. The private key is kept secret by the owner, but the public key can be widely distributed and published. One of the most important applications of public key cryptography is digital signature, which is defined as below.

Definition 3. A signature scheme Σ is made up of three essential algorithms: KeyGen, Sign and Verify. Given a system parameter param,[1] these algorithms work as follows.

KeyGen(param) $\to (sk, pk)$. On input param, this algorithm generates a private-public key pair (sk, pk). The public key includes the description of the message space $\mathcal{M}$ and signature space $\mathcal{S}$.
Sign(param, $m, sk) \to \sigma$. On input param, a message $m \in \mathcal{M}$ and a private key sk, this algorithm generates a signature $\sigma \in \mathcal{S}$.
Verify(param, $m, \sigma, pk) \to \{1, 0\}$. On input param, a message-signature pair $(m, \sigma) \in \mathcal{M} \times \mathcal{S}$ and a public key pk, this algorithm verifies the validity of the signature and outputs a decision "1" or "0". (m, σ, pk) is said to be a valid triple if Verify outputs "1".

Consistence. For any message $m \in \mathcal{M}$ and any key pair (sk, pk) generated by KeyGen, Verify(param, m, Sign(param, m, sk), $pk) = 1$.

Existential unforgeability of $\boldsymbol{\Sigma}$. The well established security notion of digital signature schemes is *existential unforgeability against adaptive chosen-message attacks* introduced by Goldwasser, Micali and Rivest [7]. It is defined by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup. The challenger $\mathcal{C}$ generates param and runs the algorithm KeyGen to obtain a key pair (sk, pk). The adversary $\mathcal{A}$ is given the description of Σ, param and pk.

Queries. Proceeding adaptively, the adversary $\mathcal{A}$ can request signatures of at most q_s messages. For each sign query $m_i \in \{m_1, \ldots, m_{q_s}\}$, the challenger $\mathcal{C}$ returns $\sigma_i =$ Sign(param, m_i, sk) as the response.

Output. After all queries, $\mathcal{A}$ outputs a pair (m^*, σ^*) and wins the game if: $m^* \notin \{m_1, \ldots, m_{q_s}\}$ and (m^*, σ^*, pk) is a valid triple.

Let Σ-Adv$_{\mathcal{A}}$ be the probability that the adversary $\mathcal{A}$ wins in the above game, taken over the coin tosses made by $\mathcal{A}$ and the challenger $\mathcal{C}$.

Definition 4. A forger $\mathcal{A}$ is said to (t, q_s, ε)-break a signature scheme Σ if $\mathcal{A}$ runs in time at most t, $\mathcal{A}$ makes at most q_s signature queries and Σ-Adv$_{\mathcal{A}}$ is at least ε. Σ is (t, q_s, ε)-existentially unforgeable against adaptive chosen-message attacks if there exists no forger that (t, q_s, ε)-breaks it.

Strong existential unforgeability. It is defined almost the same as the existential unforgeability, with the difference that the pair (m^*, σ^*) output by the adversary is not in the set $\{(m_1, \sigma_1), \ldots, (m_{q_s}, \sigma_{q_s})\}$.

[1] In this paper, we assume param is honestly generated and authenticated.

3.2. A generic framework

This subsection describes a generic communication protocol using digital signatures as the evidence of origin. For the sake of simplicity, we assume that the time at the two communication parties is synchronized. Let Σ be a signature scheme $\{\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}\}$ with a security parameter param and the message space $\mathcal{M}$. Let h be a collision-resistant hash: $\{0,1\}^* \rightarrow \mathcal{M}$. A generic framework of communication protocols can be designed as follows.

1. *Recipient-Initialization*. The recipient runs KeyGen to generate a pair $(pk\text{-}R, sk\text{-}R)$.
2. *Sender-Initialization*. The sender runs KeyGen to generate its own public/secret information $(pk\text{-}S, sk\text{-}S)$.
3. *Sender*$\{pk\text{-}R, sk\text{-}S, m\} \leftrightharpoons$ *Recipient*$\{pk\text{-}S, sk\text{-}R, time\}$. The message sender runs the signing algorithm $\mathsf{Sign}(\text{param}, h(m, time), sk\text{-}S)$ to generate a signature σ on "$h(m, time)$". Here, the signature σ is the evidence of origin ϖ. After that, (m, σ) is sent to the recipient. $(m, pk\text{-}S, time)$ is accepted as a valid triple if $(h(m, time), \sigma, pk\text{-}S)$ is a valid triple, i.e., $\mathsf{Verify}(\text{param}, h(m, time), \sigma, pk\text{-}S) = 1$.
4. *Public-Proof-Generation* is not needed since the evidence – the digital signature – is publicly verifiable.
5. *Public-Verification*$\{m, pk\text{-}R, pk\text{-}S, time, \varpi\}$ is the same as $\mathsf{Verify}(\text{param}, h(m, time), \varpi, pk\text{-}S)$.

We now show that the communication protocol described above satisfies NRO-I, if the underlying signature scheme Σ is existentially unforgeable.

Theorem 1. *The generic framework described above satisfies NRO-I, if Σ is existentially unforgeable.*

Proof. The proof is given in Appendix A. □

Theorem 1 shows that the NRO-I of the proposed framework can be reduced to the existential unforgeability of digital signatures. However, we cannot obtain the same conclusion on NRO-II, and actually this depends on the concrete digital signature schemes (as shown in the next two subsections).

3.3. `Protocol I`: with NRO-I and NRO-II

This subsection describes a concrete protocol, `Protocol I`, in the framework given in Section 3.2. `Protocol I` uses Boneh–Boyen signature [1,2] as the evidence of origin, and satisfies NRO-I and NRO-II. We first review the necessary facts about bilinear maps and groups.

- $(\mathbb{G}_1, *)$, $(\mathbb{G}_2, *)$ and $(\mathbb{G}_T, *)$ are three cyclic groups of prime order p;
- g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$;
- e is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, i.e., a map satisfying the following properties:

  Bilinearity: $\forall u \in \mathbb{G}_1, \forall v \in \mathbb{G}_2, \forall a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$;
  Non-degeneracy: $e(g_1, g_2) \neq 1$ and is thus a generator of $\mathbb{G}_T$.

Definition 5. We say that $(\mathbb{G}_1, \mathbb{G}_2)$ is a bilinear group pair if there exists a group $\mathbb{G}_T$ and a non-degenerate bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, such that the group order $p = |\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T|$ is prime, and the pairing e and the group operations in $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are all efficiently computable.

***q*-Strong Diffie–Hellman problem.** Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of prime order p, respectively generated by g_1 and g_2. In the bilinear group pair $(\mathbb{G}_1, \mathbb{G}_2)$, the q-SDH problem is stated as follows: Given as input a $(q+3)$-tuple of elements $(g_1, g_1^x, g_1^{x^2}, \ldots, g_1^{x^q}, g_2, g_2^x) \in \mathbb{G}_1^{q+1} \times \mathbb{G}_2^2$, output a pair $(c, g_1^{1/(x+c)}) \in \mathbb{Z}_p \times \mathbb{G}_1$.

An algorithm $\mathcal{B}$ solves the q-SDH problem in the bilinear group pair $(\mathbb{G}_1, \mathbb{G}_2)$ with advantage ε if $\mathsf{SDH\text{-}Adv}_{\mathcal{B}} = \Pr[\mathcal{B}(g_1, g_1^x, g_1^{x^2}, \ldots, g_1^{x^q}, g_2, g_2^x) = (c, g_1^{1/(x+c)})] \geqslant \varepsilon$, where the probability is over the random choice of generators $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, the random choice of $x \in \mathbb{Z}_p$, and the random bits consumed by $\mathcal{B}$.

Definition 6. We say that the (q, t, ε)-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ if no t-time algorithm has advantage more than ε in solving the q-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$.

Boneh–Boyen signature [1,2]. Let system parameter $\text{param} = \{\mathbb{G}_1, \mathbb{G}_2, e, p, \mathcal{M}, \mathcal{S}\}$. Here, $(\mathbb{G}_1, \mathbb{G}_2)$ is a bilinear group pair where $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ for some prime p. The message space $\mathcal{M}$ is $\mathbb{Z}_p$,[2] and the signature space $\mathcal{S}$ is $\mathbb{G}_1 \times \mathbb{Z}_p$. Boneh–Boyen signature [2] works as follows.

[2] The domain can be extended to all of $\{0,1\}^*$ using (target) collision-resistant hashing [2].

KeyGen. Select random generators $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, and random integers $x, y \in \mathbb{Z}_p^*$. Compute $u \leftarrow g_2^x \in \mathbb{G}_2$ and $v \leftarrow g_2^y \in \mathbb{G}_2$. Also compute $z \leftarrow e(g_1, g_2) \in \mathbb{G}_T$. The public key pk is the tuple (g_1, g_2, u, v, z). The secret key sk is the pair (x, y).

Sign. Given a secret key (x, y) and a message $m \in \mathbb{Z}_p$, pick a random $r \in \mathbb{Z}_p \backslash \{-\frac{x+m}{y}\}$ and compute $\sigma \leftarrow g_1^{1/(x+m+yr)} \in \mathbb{G}_1$. Here, $1/(x+m+yr)$ is computed modulo p. The signature is the pair (σ, r).

Verify. Given a public key (g_1, g_2, u, v, z) and a message m, this algorithm outputs "1" if $e(\sigma, u \cdot g_2^m \cdot v^r) = z$; otherwise, it outputs "0".

Boneh–Boyen signature is strongly existentially unforgeable under chosen-message attacks, provided that the q-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ [2].

Analysis of `Protocol I`. Using Boneh–Boyen signature as the evidence of origin, we obtain a concrete instance of the generic framework described in Section 3.2, which is called `Protocol I`. We will show that the NRO-I of `Protocol I` can be reduced to the q-SDH assumption and the NRO-II can be reduced to the DL assumption, which is defined as follows.

Discrete Logarithm (DL) in $\mathbb{G}_1$. Given $g_1, h_1 \in \mathbb{G}_1 \setminus \{1\}$, output $a \in \mathbb{Z}_p$ such that $h_1 = g_1^a$.

An algorithm $\mathcal{B}$ solves the DL problem in $\mathbb{G}_1$ with advantage at least ε if $\mathsf{DL} - \mathsf{Adv}_{\mathcal{B}} = \Pr[\mathcal{B}(g_1, h_1) = a : h_1 \in_R \mathbb{G}_1^*] \geqslant \varepsilon$. The probability is over the uniform random choice of h_1 from $\mathbb{G}_1$, and the random bits consumed by $\mathcal{B}$.

Definition 7. We say that the (t, ε)-DL assumption holds in $\mathbb{G}_1$ if no t-time algorithm has advantage more than ε in solving the DL problem in $\mathbb{G}_1$.

Theorem 2. *`Protocol I` satisfies the non-repudiation of origin defined in Section 2.*

Proof. The proof is given in Appendix B. □

3.4. *`Protocol II`: existential unforgeability $\nRightarrow$ NRO-II*

This subsection describes another instance of the generic framework proposed in Section 3.2, which we call `Protocol II`. Like `Protocol I`, `Protocol II` also uses existentially unforgeable signatures as the evidence of origin, but the difference is that `Protocol II` does not satisfy NRO-II. This serves as a good example to show that existentially unforgeable signatures cannot ensure the non-repudiation of origin defined in this paper, which we believe deserves special attention from protocol designers.

The unforgeable signatures used in this subsection is a variant of Boneh–Boyen signature, which we call Boneh–Boyen†. Although never been formally investigated, Boneh–Boyen† has been used as a building block in the design of many cryptographic protocols, such as those in [18,23,24], and is also strongly existentially unforgeable. Below is the description of Boneh–Boyen† signature.

Boneh–Boyen† signature. The system parameter is $\mathsf{param} = \{\mathbb{G}_1, \mathbb{G}_2, e, p, \mathcal{M}, \mathcal{S}, g_1, g_2\}$, where $\{\mathbb{G}_1, \mathbb{G}_2, e, p, \mathcal{M}, \mathcal{S}\}$ are the same as those in Boneh–Boyen signature, and g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$. The signature scheme works as follows.

KeyGen. Select two random integers $x, y \in \mathbb{Z}_p^*$, and compute $u \leftarrow g_2^x \in \mathbb{G}_2$ and $v \leftarrow g_2^y \in \mathbb{G}_2$. The public key pk is the pair (u, v) and the secret key sk is the pair (x, y).

Sign. Given a secret key (x, y) and a message $m \in \mathbb{Z}_p$, pick a random $r \in \mathbb{Z}_p \backslash \{-\frac{x+m}{y}\}$ and compute $\sigma \leftarrow g_1^{1/(x+m+yr)} \in \mathbb{G}_1$. Here, $1/(x+m+yr)$ is computed modulo p. The signature s is the pair (σ, r).

Verify. Given the system parameter param, a public key (u, v) and a message m, this algorithm outputs "1" if $e(\sigma, u \cdot g_2^m \cdot v^r) = e(g_1, g_2)$; otherwise, it outputs "0".

Existential unforgeability of Boneh–Boyen†. The only difference between Boneh–Boyen† signature and Boneh–Boyen signature [1,2] is the choice of generators in $\mathbb{G}_1$ and $\mathbb{G}_2$. Boneh–Boyen signature allows users to choose generators and include them as part of the public keys, but Boneh–Boyen† signature sets (g_1, g_2) in the system parameter. As a result, each user will use the same generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ in Boneh–Boyen†. This leads to a more efficient KeyGen algorithm than the original Boneh–Boyen scheme, but the efficiency of the other two algorithms (i.e., Sign and Verify) remains the same.

It is not hard to see that including (g_1, g_2) in param does not have any negative impact on the existential unforgeability of the signature. Like the original scheme, Boneh–Boyen† signature is also strongly existentially unforgeable under chosen-message attacks, provided that the q-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$. This can be proved using the same techniques in [1,2], since in the game of unforgeability it is the challenger who generates the system parameter and the challenging public key. (Actually, the security of [18,23,24], where Boneh–Boyen† signature serves as a building block, also requires the unforgeability of Boneh–Boyen†.)

Existential unforgeability $\nRightarrow$ NRO-II. Although Boneh–Boyen† is (strongly) existentially unforgeable, we will show that given two different public/private key pairs and any two messages (either identical or different), one can create a Boneh–Boyen† signature which is valid under both public keys. As a result, communication protocols using Boneh–Boyen† signatures as the evidence of origin will not provide NRO-II. The details are given as follows.

1. Let (sk_A, pk_A) be a key pair generated by KeyGen. Here, $sk_A = (x_A, y_A)$: x_A, y_A are randomly chosen in $\mathbb{Z}_p^*$; and $pk_A = (u_A, v_A)$: $u_A = g_2^{x_A} \in \mathbb{G}_2$ and $v_A = g_2^{y_A} \in \mathbb{G}_2$.
2. Let (sk_B, pk_B) be another key pair generated by KeyGen. Here, $sk_B = (x_B, y_B)$: x_B, y_B are randomly chosen in $\mathbb{Z}_p^*$; and $pk_B = (u_B, v_B)$: $u_B = g_2^{x_B} \in \mathbb{G}_2$ and $v_B = g_2^{y_B} \in \mathbb{G}_2$.
3. Given two messages m_A and m_B (either identical or different), find an integer $r \in \mathbb{Z}_p$ such that $x_A + m_A + ry_A = x_B + m_B + ry_B \mod p$. If $y_A \neq y_B$,
$$r = (y_A - y_B)^{-1}(x_B - x_A + m_B - m_A) \mod p.$$
Since y_A and y_B are randomly chosen from $\mathbb{Z}_p^*$, the probability that $y_A \neq y_B$ is $1 - \frac{1}{p-1}$.
4. Calculate $\sigma = g_1^{1/(x_A+m_A+ry_A)}$. It is evident that $s = (\sigma, r)$ is a valid signature on m_A under pk_A.
5. σ is also equal to $g_1^{1/(x_B+m_B+ry_B)}$, since $x_A + m_A + ry_A = x_B + m_B + ry_B \mod p$. It follows that $s = (\sigma, r)$ is a valid signature on m_B under pk_B.
6. Thus, one can successfully create a Boneh–Boyen† signature s such that (m_A, s, pk_A) and (m_B, s, pk_B) are two valid triples.

Remark 5. It is worth mentioning that a trivial approach does not fix the problem. Suppose that in signature generation, we prepend m by the public key pk in a non-ambiguous way and apply the algorithm Sign to the concatenation of pk and m, i.e., $pk \parallel m$. To verify the validity of a signature under pk, the algorithm Verify is modified by verifying a signature on $pk \parallel m$. Such a modification cannot solve the problem since the above algorithm works on any two messages. By viewing $pk_A \parallel m_A$ and $pk_B \parallel m_B$ as the messages to be signed, one can still create a signature s such that $(pk_A \parallel m_A, s, pk_A)$ and $(pk_B \parallel m_B, s, pk_B)$ are two valid triples.

Remark 6. Now we compare NRO-II and the security of digital signatures under key substitution attacks in [19]. Both notions consider the validity of a signature (or, a piece of evidence) under two different public keys, but there are two major differences: (1) The attacker defined in [19] is not given the private key of the challenging public key, and (2) Prepending the message by the public key pk in a non-ambiguous way, the method suggested in [19] to mitigate key substitution attacks, does not work on NRO-II (as shown in Remark 5).

4. Non-repudiation of origin with privacy protection

This section is devoted to the privacy issue in communication protocols with non-repudiation of origin. We will give the definition of privacy and describe a communication protocol accommodating both privacy and non-repudiation of origin.

4.1. Definition of privacy: non-transferable conviction

As shown in the previous section, existentially unforgeable digital signatures (but not all) can be used as non-repudiation evidence of origin in communication protocols. However, in ordinary digital signatures, the Verify algorithm only uses the signer's public key pk to check the validity of a message-signature pair. Thus, once given a valid message-signature pair under pk, anyone will be convinced that the message was signed by the owner of pk and thus find out who is the message sender. This property, together with the ease of copying and transmitting digital signatures, would be undesirable in personally or commercially sensitive applications where the message sender's privacy is a concern.

A representative scenario we are concerned about is as follows. The recipient is a service provider (say, a pay-TV server) and the message sender is the service requester. To make a correct fee charge and avoid future disputes, the service provider must keep a record of the evidence of origin, which shows the identity of service requester and the requested service. But such a record could also lead to the loss of custom privacy, e.g., the service provider can sell the information like "which TV program Alice is watching" to advertisers. If the evidence of origin is an ordinary digital signature, any third parties (including the advertisers) will believe the authenticity of that information.

To protect the privacy of message originator, the evidence generated by the originator should only convince the recipient.[3] To achieve that, we allow the recipient, by running a new algorithm called *Recipient-Simulation*, to generate a special

[3] Readers should bear in mind that privacy is a broad notion and has different meanings in different scenarios.

kind of evidence of origin which is different but (computationally) indistinguishable from the valid evidence generated by the originator.

Recipient-Simulation$\{m, pk\text{-}S, sk\text{-}R, time\}$.
This algorithm allows the recipient to simulate the protocol *Sender*$\{pk\text{-}R, sk\text{-}S, m\} \leftrightharpoons$ *Recipient*$\{pk\text{-}S, sk\text{-}R, time\}$ and generate a simulated evidence of origin.

To make it fair, for a simulated evidence of origin generated by the algorithm *Recipient-Simulation*, one (with $sk\text{-}S$) can run the algorithm *Public-Proof-Generation* to generate a bit-string π such that *Public-Verification* outputs "0".

We are now ready to define the privacy of message originator in communication protocols with non-repudiation of origin.

Game of privacy:

Setup. The challenger generates ($pk\text{-}R$, $sk\text{-}R$) for the recipient R, and ($pk\text{-}S$, $sk\text{-}S$) for the sender. The adversary is given ($pk\text{-}R$, $sk\text{-}R$) and $pk\text{-}S$.

Queries I. The adversary can make queries defined in the game of NRO-I (Section 2.2).

Challenge. The adversary chooses a message m^* and sends it to the challenger at $time^*$. The challenger responds with a transcript and an evidence ϖ^*, the generation of which depends on a random bit $b \in \{0, 1\}$ picked by the challenger. If $b = 0$, the transcript and the evidence are those generated using $sk\text{-}S$ during the execution of *Sender*$\{pk\text{-}R, sk\text{-}S, m^*\}$ $\leftrightharpoons$ *Recipient*$\{pk\text{-}S, sk\text{-}R, time^*\}$. Otherwise, they are the output of the algorithm *Recipient-Simulation*.

Queries II. The adversary can continue to make queries, with the restriction that $(m^*, time^*, \varpi^*)$ cannot be sent to the challenger, i.e., the adversary cannot ask the challenger to prove whether or not ϖ^* is a valid evidence of (m^*, $pk\text{-}S, time^*$).

Output. The output of the adversary is a bit b'. The adversary wins the game if $b' = b$.

Definition 8. A communication protocol is said to provide (t, q, ϵ)-privacy if no adversary, by making q queries in time t, can win the game of privacy with success probability more than $\epsilon + 1/2$.

As one can see, in a communication protocol satisfying our definition of privacy, no one (even with the secret information of the recipient) can tell whether the transcript and the evidence are generated by the claimed message sender or by the recipient. Such a property ensures that the evidence only convinces the intended recipient. This reminds us a closely relevant cryptographic primitive: Designated Verifier Signature.

4.2. Designated verifier signature: a brief review

Designated verifier signature (DVS) [14] is a variant of ordinary digital signatures with the following feature: Only the designated verifier can be convinced about the validity of the signature. When signing a message, the signer can choose whom will be convinced about the signer's commitment on a message, and the entity chosen by the signer is called designated verifier. A DVS on a message m proves to the designated verifier that the signer has signed the message, but this conviction is non-transferable, even the designated verifier shares all secret information with those who want to get convinced. A variant of DVS, strong designated verifier signature (SDVS), has an additional property: Given a designated verifier signature and two potential signing public keys, it is *computationally infeasible* for an eavesdropper to determine under which of the two corresponding private keys the signature was performed. SDVS prevents the adversaries, who obtain DVS during the transmission from the signer to the verifier, telling who the signature producer is. To date, a number of (strong) designated verifier signature schemes have been proposed [21,16,22,17,9,10].

It seems at the first glance that (S)DVS fits our need perfectly, by viewing the message sender as the signer, the recipient as the designated verifier and using (S)DVS as the evidence of origin. However, a closer look at existing (S)DVS schemes shows the gap between what we have and what we need. To the best of our knowledge, the existing designs of (S)DVS share the same approach: Given a signature σ between Alice and Bob, Alice and Bob have the equal chance to be the producer of σ. We would like to emphasize that the verifier of SDVS is also able to produce valid SDVS in the existing designs of SDVS. By taking this approach, no one except Alice and Bob can believe who is the actual signature producer. However, if we use such a signature as the evidence of origin, the resulting communication protocol will contradict NRO-I defined in Section 2.2, which requires that only one entity, i.e., the message sender, be able to generate the evidence of origin. Therefore, no existing designs of (S)DVS can satisfy our goals of non-repudiation and privacy protection.

To obtain non-repudiation of origin and protect the privacy of originator, the evidence should only convince the intended recipient, and meanwhile the recipient cannot forge that evidence. In other words, a DVS scheme with unforgeability against verifier will help us achieve the goal. Appendix C describes the design philosophy of such DVS schemes, which is used in our protocol in the next subsection.

4.3. A protocol with non-repudiation of origin and privacy protection

Following the approach described in the previous subsection, we give a concrete design of communication protocol with non-repudiation of origin and privacy protection on the message originator.

Let $\mathsf{param} = (G, p, g, H_1, H_2)$, where G is an Abelian group of prime order p, g is a generator of G, H_1 is a hash function: $\{0,1\}^* \to G$ and H_2 is another hash function: $\{0,1\}^* \to \mathbb{Z}_p$. Like the framework described in Section 3.2, we assume that the message sender and the recipient have a synchronized clock. The protocol is described as follows.

Recipient-Initialization. The recipient chooses a random number x_R from $\mathbb{Z}_p^*$ and calculates $y_R = g^{x_R}$. Let the pair $(pk\text{-}R, sk\text{-}R) = (x_R, y_R)$.

Sender-Initialization. The sender chooses a random number x_S from $\mathbb{Z}_p^*$ and calculates $y_S = g^{x_S}$. Let the pair $(pk\text{-}S, sk\text{-}S) = (x_S, y_S)$.

Sender$\{pk\text{-}R, sk\text{-}S, m\} \leftrightharpoons$ *Recipient*$\{pk\text{-}S, sk\text{-}R, time\}$.

1. The sender first calculates $\sigma = H_1(m, time)^{x_S}$, the FDH (full-domain hash) variant of Chaum's undeniable signature [3].
2. The sender then generates a DVP $\rho = (c_S, c_R, d_S, d_R)$, proving that he knows a secret sk:

$$\text{“}sk = \mathsf{DL}_g[y_S] = \mathsf{DL}_{H_1(m,time)}[\sigma]\text{”} \quad \text{or} \quad \text{“}sk = \mathsf{DL}_g[y_R]\text{”}.$$

   In other words, the sender proves that σ is a valid undeniable signature of $pk\text{-}S$ or he/she has the private key of the recipient. Notice that both the sender and the recipient are able to generate the proof ρ. For the sender, the secret sk is x_S, and the proof $\rho = (c_S, c_R, d_S, d_R)$ is generated as follows:
   - Choose three random integers r, c_R, d_R from $\mathbb{Z}_p^*$;
   - Calculate $c_S = H_2(g^r, H_1(m, time)^r, g^{d_R} y_R^{c_R}) - c_R$;
   - Calculate $d_S = (r - c_S x_S) \bmod p$.

   The generation of ρ is based on the non-interactive zero-knowledge proof for Chaum's scheme reviewed in [20], which ensures that the message sender is not able to generate the proof if σ is an invalid undeniable signature.
3. The sender sends the recipient the triple (m, σ, ρ), where the evidence of origin ϖ is σ.
4. Given (m, σ, ρ) and $\rho = (c_S, c_R, d_S, d_R)$, $(m, pk\text{-}S, time)$ is accepted by the recipient as a valid triple *iff*

$$H_2\big(g^{d_S} y_S^{c_S}, H_1(m, time)^{d_S} \sigma^{c_S}, g^{d_R} y_R^{c_R}\big) = (c_S + c_R) \mod p. \tag{1}$$

   For a valid triple, the recipient will keep a record of the associated evidence of origin, i.e., σ.

Recipient-Simulation$\{m, pk\text{-}S, sk\text{-}R, time\}$.

The recipient can use the secret $sk\text{-}R$ to generate a transcript of a normal communication, i.e., proving that he/she has a secret such that

$$\text{“}sk = \mathsf{DL}_g[y_S] = \mathsf{DL}_{H_1(m,time)}[\sigma]\text{”} \quad \text{or} \quad \text{“}sk = \mathsf{DL}_g[y_R]\text{”}.$$

The secret that the recipient has is $sk\text{-}R = x_S$, and the proof $\rho = (c_S, c_R, d_S, d_R)$ is generated as follows.
- Choose a random element $\sigma \in G$;
- Choose three random integers r, c_S, d_S from $\mathbb{Z}_p^*$;
- Calculate $c_R = H_2(g^{d_S} y_S^{c_S}, H_1(m, time)^{d_S} \sigma^{c_S}, g^r) - c_S$;
- Calculate $d_R = (r - c_R x_R) \bmod p$.

As one can see, the simulated transcript (m, σ, ρ) also satisfies Eq. (1).

Public-Proof-Generation$\{m, pk\text{-}R, pk\text{-}S, sk\text{-}S, time, \varpi\} \to \pi$.

1. If $\varpi = H_1(m, time)^{x_S}$, the sender generates a proof π showing that $\mathsf{DL}_g[y_S] = \mathsf{DL}_{H_1(m,time)}[\varpi]$. The proof consists of two parts, which are generated as follows:
   - Choose a random integer r from $\mathbb{Z}_p^*$;
   - Calculate $c = H_2(g^r, H_1(m, time)^r)$;
   - Calculate $d = (r - c x_S) \bmod p$.
2. Otherwise $\varpi \neq H_1(m, time)^{x_S}$, the sender generates a proof π showing that $\mathsf{DL}_g[y_S] \neq \mathsf{DL}_{H_1(m,time)}[\varpi]$:
   - Choose three random integers r_1, r_2, r_3 from $\mathbb{Z}_p^*$;
   - Calculate $c = (H_1(m, time)^{x_S} / \varpi)^{r_1}$;
   - Calculate $d_1 = H_2(c, g^{r_2}/Y_s^{r_3}, H_1(m, time)^{r_2}/\varpi^{r_3})$, $d_2 = (r_3 + r_1 d_1) \bmod p$, $d_3 = (r_2 + r_1 d_1 x_s) \bmod p$.

The proof above is based on the non-interactive zero-knowledge proof for Chaum's scheme reviewed in [20], which ensures that the message sender is not able to prove a valid evidence as invalid (or, an invalid evidence as valid).

Public-Verification$\{m, pk\text{-}R, pk\text{-}S, time, \varpi, \pi\}$.

1. Output "1" if $\pi = (c, d) \in \mathbb{Z}_p^2$ and $c = H_2(g^d y_S^c, H_1(m, time)^d \varpi^c)$.
2. Output "0" if $\pi = (c, d_1, d_2, d_3) \in G \times \mathbb{Z}_p^3$, $c \neq 1$ and

$$d_1 = H_2\big(c, g^{d_3}/Y_s^{d_2}, H_1(m, time)^{d_3}/\big(\varpi^{d_2} \cdot c^{d_1}\big)\big).$$

This completes the description of the proposed protocol.

4.4. Security analysis of the proposed protocol

We show that the proposed protocol satisfies NRO-I, NRO-II and provides privacy protection of message originator.

Computational Diffie–Hellman (CDH) in G. Given $g, g^a, g^b \in G^3$, compute g^{ab}.

Decisional Diffie–Hellman (DDH) in G. Given $g, g^a, g^b, h \in G^4$, output "1" if $h = g^{ab}$ and "0", otherwise.

NRO-I. Our protocol falls in the generic framework described in Section 3.2, by using the digital signature as the evidence of origin, i.e., the FDH variant of Chaum's undeniable signature [3]. As shown in [20], the existential unforgeability of Chaum's undeniable signature can be reduced to the hardness of computational Diffie–Hellman problem in the random oracle model. Thus, due to Theorem 1, our protocol can provide NRO-I in the random oracle model if computational Diffie–Hellman problem is hard on G.

NRO-II. We prove that the proposed protocol satisfies NRO-II defined in Section 2.3. Let (pk_A, sk_A) and (pk_B, sk_B) be two key pairs generated by the challenger. Suppose the adversary can create an evidence ϖ^* proving that

1. $(m^*, pk_A, time^*)$ is a valid triple, i.e., $\varpi^* = h_1(m^*, time^*)^{sk_A}$; and
2. $(m^*, pk_B, time^*)$ is a valid triple, i.e., $\varpi^* = h_1(m^*, time^*)^{sk_B}$.

It follows that $sk_A = sk_B \bmod p$, which occurs with probability $1/p$ since sk_A and sk_B are randomly chosen by the challenger in $\mathbb{Z}_p$. Therefore, the proposed protocol provides $(t, 1/p)$-NRO-II.

Privacy. We show that the privacy of the proposed protocol can be reduced to the invisibility of Chaum's undeniable signature in the random oracle model, and thus is based on the hardness of decisional Diffie–Hellman problem in G due to the analysis in [20].

Theorem 3. *The proposed protocol provides (t, q, ϵ)-privacy-protection if Chaum's undeniable signature is (t, q, ϵ)-invisible.*

Proof. Let $\mathcal{A}$ be the adversary who tries to break the privacy of our proposed protocol, and let $\mathcal{B}$ be the adversary who wishes to break the invisibility of Chaum's undeniable signature. (Please refer to [20] for the definition of the game of invisibility.) $\mathcal{B}$ is given a public key y and is allowed to make queries to the challenger. To make use of $\mathcal{A}$, $\mathcal{B}$ will answer all queries from $\mathcal{A}$.

Setup. $\mathcal{B}$ generates a key pair $(pk\text{-}R, sk\text{-}R)$ for the recipient (running *Recipient-Initialization*) and sets $pk\text{-}S = y$. After that, $(pk\text{-}R, sk\text{-}R, y)$ is sent to $\mathcal{A}$.

Queries I. $\mathcal{A}$ can issue two kinds of queries.

1. For a query m_i at $time_i$, $\mathcal{B}$ must simulate *Sender*$\{pk\text{-}R, sk\text{-}S, m_i\} \leftrightharpoons$ *Recipient*$\{pk\text{-}S, sk\text{-}R, time_i\}$. In our protocol, $\mathcal{B}$ must generate (σ_i, ρ_i). To do that, $\mathcal{B}$ asks its challenger to generate the undeniable signature σ on $H_1(m_i, time_i)$. After that, $\mathcal{B}$ generates the DVP ρ_i by treating H_2 as the random oracle. (m_i, σ_i, ρ_i) is returned as the response.
2. For a query $(m_i, time_i, \varpi_i)$, $\mathcal{B}$ first asks its challenger to verify the validity of the undeniable signature ϖ on the message $H_1(m_i, time_i)$. After that, $\mathcal{B}$ generates the proof π in the random oracle model, which is returned as the response.

Challenge. $\mathcal{A}$ chooses a message m^* and sends it to $\mathcal{B}$. Let $time^*$ be the associated time. $\mathcal{B}$ sends $H_1(m^*, time^*)$ as the challeng message to its own challenger, who will respond with a challenge signature σ^*. After that, $\mathcal{B}$ generates a DVP $\rho^* = (c_S^*, d_S^*, c_R^*, d_R^*)$ in the random oracle model, which actually is a random tuple among those satisfying Eq. (1). (σ^*, ρ^*) is given to $\mathcal{A}$.

Queries II. $\mathcal{A}$ can continue to make queries under the restriction described in Section 4.1. $\mathcal{B}$ generates the response in the same way as in **Queries I**.

Output. Finally, $\mathcal{A}$ outputs its guess b'. $\mathcal{B}$ will set b' as its own guess.

If $\mathcal{A}$ can output a correct guess, then $\mathcal{B}$ can successfully tell if σ^* is a valid undeniable signature. Thus, $\mathcal{B}$ can win the game of invisibility with the same success probability as $\mathcal{A}$. This completes the proof of Theorem 3. □

5. Conclusion

This paper demonstrated how to achieve a tradeoff between non-repudiation of origin and privacy of originator. We defined two concrete goals of non-repudiation of origin: NRO-I and NRO-II. NRO-I requires that the recipient should not be able to create any valid evidence of origin, and NRO-II requires the unambiguity of the evidence of origin. Our analysis showed that existentially unforgeable digital signatures can provide NRO-I but not always NRO-II. The privacy of originator is defined by enabling the recipient to produce a special kind of evidence of origin, which is invalid but computationally indistinguishable (to other entities) from valid evidence. We presented a concrete protocol accommodating both non-repudiation of origin and privacy of originator with formal proofs in the random oracle model.

Acknowledgments

This work is supported partially by National Natural Science Foundation of China (Grant No. 61202450, No. 61072080 and No. 61003232), Ph.D. Programs Foundation of Ministry of Education of China (Grant No. 20123503120001) and Department of Education, Fujian Province, A-Class Project (Grant No. JA12076).

Appendix A. Proof of Theorem 1

Proof. We will show that if there is an adversary $\mathcal{A}$ who can compromise NRO-I of the generic framework, then there is another adversary $\mathcal{B}$ who can break the existential unforgeability of Σ.

In the game of unforgeability, the adversary $\mathcal{B}$ is given a public key pk^* of a digital signature scheme $\Sigma = \{\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}\}$ with a security parameter param and the message space $\mathcal{M}$. $\mathcal{B}$ can make queries to its challenger, who will generate valid signatures on the messages chosen by $\mathcal{B}$. Finally, $\mathcal{B}$ must output a valid signature of a new message. To make use of $\mathcal{A}$ to achieve its goal, $\mathcal{B}$ acts as the challenger of $\mathcal{A}$ and works as follows.

Setup. The setup phase consists of the following steps:
1. $\mathcal{B}$ chooses a collision-resistant hash function $h : \{0,1\}^* \to \mathcal{M}$.
2. Let $pk\text{-}R$ be the public key of the recipient generated by the adversary.
3. $\mathcal{B}$ sets the public key of the first sender S_1 as $pk\text{-}S_1 = pk^*$.
4. $\mathcal{B}$ runs KeyGen to generate $(pk\text{-}S_2, sk\text{-}S_2)$ for $S_2, \ldots,$ and $(pk\text{-}S_N, sk\text{-}S_N)$ for S_N.

The adversary is given $\{pk\text{-}S_1, (pk\text{-}S_2, sk\text{-}S_2), \ldots, (pk\text{-}S_N, sk\text{-}S_N)\}$, together with the description of Σ, param and the hash function h.

Queries. For a query $(m_i,\ pk\text{-}S_1)$ at $time_i$, $\mathcal{B}$ sends a signature query "$h(m_i, time_i)$" to its challenger. Let the response be σ_i. $\mathcal{A}$ is given σ_i as the answer.

Output. $\mathcal{A}$ selects a message m^* and executes the protocol $Sender\{pk\text{-}R, sk\text{-}S_1, m^*\} \leftrightharpoons Recipient\{pk\text{-}S_1, sk\text{-}R, time^*\}$ with $\mathcal{B}$.

If $\mathcal{A}$ wins the game, then the evidence of origin ϖ^* generated during the execution must be a valid signature on $h(m^*, time^*)$. Since the time information $time^*$ never appears before and h is a collision-resistant hash function, $\mathcal{B}$ can output ϖ^* as a signature on the new message $h(m^*, time^*)$. In this way, $\mathcal{B}$ can break the existential unforgeability of Σ with the same success probability as $\mathcal{A}$. Therefore, the generic framework satisfies NRO-I if Σ is existentially unforgeable.

This completes the analysis on the NRO-I of the proposed framework. □

Appendix B. Proof of Theorem 2

Proof. Due to Theorem 1, `Protocol I` satisfies NRO-I under the q-SDH assumption. We will show that if there is an adversary $\mathcal{A}$ who can compromise the NRO-II of `Protocol I`, then there is another algorithm $\mathcal{B}$ who can solve DL problem. Thus, `Protocol I` also satisfies NRO-II under the DL assumption.

Let $(\mathbb{G}_1, \mathbb{G}_2)$ be bilinear groups where $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ for some prime p. Algorithm $\mathcal{B}$ is given (g_1, h_1) as an instance of the DL problem in $\mathbb{G}_1$, and its goal is to calculate a such that $h_1 = g_1^a$. Algorithm $\mathcal{B}$ simulates the challenger and interacts with $\mathcal{A}$ as follows.

Setup. Let $(\mathbb{G}_1, \mathbb{G}_2, e, p, \mathcal{M}, \mathcal{S})$ be the system parameter of Boneh–Boyen signature, where $\mathcal{M} = \mathbb{Z}_p$ and $\mathcal{S} = \mathbb{G}_1 \times \mathbb{Z}_p$.

1. $\mathcal{B}$ chooses a collision-resistant hash function $h : \{0, 1\}^* \to \mathcal{M}$.
2. $\mathcal{B}$ chooses two random integers $r_A, r_B \in \mathbb{Z}_p^*$, and calculates $a_1 \leftarrow g_1^{r_A}$ and $b_1 \leftarrow h_1^{r_B}$.
3. $\mathcal{B}$ chooses two random generators $a_2, b_2 \in \mathbb{G}_2$, and four random integers $x_A, y_A, x_B, y_B \in \mathbb{Z}_p^*$.
4. $\mathcal{B}$ calculates $u_A \leftarrow a_2^{x_A}$, $v_A \leftarrow a_2^{y_A}$, $u_B \leftarrow b_2^{x_B}$ and $v_B \leftarrow b_2^{y_B}$. $\mathcal{B}$ also computes $z_A \leftarrow e(a_1, a_2)$ and $z_B \leftarrow e(b_1, b_2)$.
5. The public key pk_A is the tuple $(a_1, a_2, u_A, v_A, z_A)$ and the corresponding secret key sk_A is (x_A, y_A).
6. The public key pk_B is the tuple $(b_1, b_2, u_B, v_B, z_B)$ and the corresponding secret key sk_B is (x_B, y_B).
7. The key pair $(pk\text{-}R, sk\text{-}R)$ is the output of a normal run of *Recipient-Initialization*.

The adversary $\mathcal{A}$ is given the hash h, (pk_A, sk_A), (pk_B, sk_B) and $(pk\text{-}R, sk\text{-}R)$.

Output. $\mathcal{A}$ selects a message m^* and executes the protocol $\mathit{Sender}\{pk\text{-}R, sk_A, m^*\} \leftrightharpoons \mathit{Recipient}\{pk_A, sk\text{-}R, \mathit{time}^*\}$ with $\mathcal{B}$ at time^*.

If $\mathcal{A}$ wins the game, it must output the evidence (σ, r) which is a valid signature on $h(m^*, \mathit{time}^*)$ under pk_A and pk_B, namely

1. $e(\sigma, u_A \cdot a_2^{h(m^*,\mathit{time}^*)} \cdot v_A^r) = z_A$, i.e., $\sigma = a_1^{1/(x_A + h(m^*,\mathit{time}^*) + y_A r)}$; and
2. $e(\sigma, u_B \cdot b_2^{h(m^*,\mathit{time}^*)} \cdot v_B^r) = z_B$, i.e., $\sigma = b_1^{1/(x_B + h(m^*,\mathit{time}^*) + y_B r)}$.

Thus, $a_1^{1/(x_A + h(m^*,\mathit{time}^*) + y_A r)} = b_1^{1/(x_B + h(m^*,\mathit{time}^*) + y_B r)}$. Replacing a_1 with $g_1^{r_A}$ and b_1 with $h_1^{r_B}$, we have $g_1^{r_A/(x_A + h(m^*,\mathit{time}^*) + y_A r)} = h_1^{r_B/(x_B + h(m^*,\mathit{time}^*) + y_B r)}$. $\mathcal{B}$ can calculate $a \leftarrow r_A r_B^{-1}(x_B + h(m^*, \mathit{time}^*) + y_B r)(x_A + h(m^*, \mathit{time}^*) + y_A r)^{-1}$.

In this way, $\mathcal{B}$ can solve the given instance of the DL problem with the same success probability as $\mathcal{A}$. Therefore, `Protocol I` satisfies NRO-II under the DL assumption. □

Appendix C. DVS with unforgeability against verifier: the design philosophy

This section reviews the original motivation of DVS in [14], which gives us a hint of how to preserve unforgeability in DVS.

The interactive version of DVS is originally called "Designated Verifier Proofs" (or, DVP for short). For many proofs of knowledge, it is important that only the verifier(s) designated by the prover can obtain any conviction of the correctness of the proof. A good example of such a situation is for undeniable signatures [3]. The most distinctive feature of undeniable signatures is that signature verification needs the assistance from the signer: An undeniable signature can only be verified by someone who knows the private key, or by a proof generated by the private key owner. This property allows the signer to decide when a signature can be verified, but not by whom (or even by how many), due to the blackmailing [5,13] and mafia attacks [4].[4] In those attacks, the conviction can be transferred to one or more hidden co-verifiers while the prover is not able to be aware of it. To defeat such attacks, the prover of an undeniable signature wants to make sure that only the intended verifier(s) in fact can be convinced by the proof. This is the motivation of DVP [14]. The design of DVP can be described in one sentence [14]: Instead of proving a statement Θ, Alice will prove the statement "Either Θ is true, or I am Bob". Bob will certainly trust that Θ is true upon seeing such a proof, but if Bob diverts the proof to Cindy, Cindy will have no reason at all to believe that Θ is true since Bob is capable of proving himself to be Bob (i.e., the second part of the statement). Depending on concrete designs, DVP can be interactive or non-interactive.

The review above gives us a hint of designing DVS with unforgeability, namely by combining together undeniable signatures and non-interactive DVP. This is called non-interactive undeniable signature in [14,15,20]. To designate a verifier Bob, the signer Alice generates an undeniable signature σ of m and a non-interactive DVP ρ, which proves that "Either σ is a valid undeniable signature of Alice, or I have Bob's private key". The resulting DVS of m is (σ, ρ). Given (σ, ρ), the designated verifier Bob will be convinced about Alice's commitment on m. But such a conviction is (computationally) non-transferable, since the undeniable signature σ can only be verified with the assistance from the private key owner and the proof ρ is DVP. The existential unforgeability (against the verifier) of (σ, ρ) is due to the unforgeability of the undeniable signature.

References

[1] D. Boneh, X. Boyen, Short signatures without random oracles, in: C. Cachin, J. Camenisch (Eds.), EUROCRYPT, in: Lecture Notes in Comput. Sci., vol. 3027, Springer, 2004, pp. 56–73.
[2] D. Boneh, X. Boyen, Short signatures without random oracles and the SDH assumption in bilinear groups, J. Cryptology 21 (2) (2008) 149–177.
[3] D. Chaum, H.V. Antwerpen, Undeniable signatures, in: G. Brassard (Ed.), CRYPTO, in: Lecture Notes in Comput. Sci., vol. 435, Springer, 1989, pp. 212–216.

[4] Also known as the man-in-the middle attack.

[4] Y. Desmedt, C. Goutier, S. Bengio, Special uses and abuses of the Fiat–Shamir passport protocol, in: C. Pomerance (Ed.), CRYPTO, in: Lecture Notes in Comput. Sci., vol. 293, Springer, 1987, pp. 21–39.
[5] Y. Desmedt, M. Yung, Weakness of undeniable signature schemes (extended abstract), in: D.W. Davies (Ed.), EUROCRYPT, in: Lecture Notes in Comput. Sci., vol. 547, Springer, 1991, pp. 205–220.
[6] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory 22 (6) (1976) 644–654.
[7] S. Goldwasser, S. Micali, R.L. Rivest, A digital signature scheme secure against adaptive chosen-message attacks, SIAM J. Comput. 17 (2) (1988) 281–308.
[8] L. Hollaar, A. Asay, Legal recognition of digital-signatures, J. IEEE Micro 16 (3) (1996) 44–45.
[9] X. Huang, W. Susilo, Y. Mu, F. Zhang, Short (identity-based) strong designated verifier signature schemes, in: K. Chen, R.H. Deng, X. Lai, J. Zhou (Eds.), ISPEC, in: Lecture Notes in Comput. Sci., vol. 3903, Springer, 2006, pp. 214–225.
[10] X. Huang, W. Susilo, Y. Mu, F. Zhang, Short designated verifier signature scheme and its identity-based variant, Int. J. Netw. Secur. 6 (1) (2008) 82–93.
[11] X. Huang, Y. Xiang, A. Chonka, J. Zhou, R.H. Deng, A generic framework for three-factor authentication: Preserving security and privacy in distributed systems, IEEE Trans. Parallel Distrib. Syst. 22 (8) (2011) 1390–1397.
[12] ISO/IEC, ISO/IEC 13888: Information Technology-Security Techniques-Non-repudiation-Part 3: Mechanisms using asymmetric techniques, 1997.
[13] M. Jakobsson, Blackmailing using undeniable signatures, in: A.D. Santis (Ed.), EUROCRYPT, in: Lecture Notes in Comput. Sci., vol. 950, Springer, 1994, pp. 425–427.
[14] M. Jakobsson, K. Sako, R. Impagliazzo, Designated verifier proofs and their applications, in: U.M. Maurer (Ed.), EUROCRYPT, in: Lecture Notes in Comput. Sci., vol. 1070, Springer, 1996, pp. 143–154.
[15] C. Kudla, K.G. Paterson, Non-interactive designated verifier proofs and undeniable signatures, in: N.P. Smart (Ed.), IMA Int. Conf., in: Lecture Notes in Comput. Sci., vol. 3796, Springer, 2005, pp. 136–154.
[16] F. Laguillaumie, D. Vergnaud, Designated verifier signatures: Anonymity and efficient construction from any bilinear map, in: C. Blundo, S. Cimato (Eds.), SCN, in: Lecture Notes in Comput. Sci., vol. 3352, Springer, 2004, pp. 105–119.
[17] F. Laguillaumie, D. Vergnaud, Multi-designated verifiers signatures, in: J. Lopez, S. Qing, E. Okamoto (Eds.), ICICS, in: Lecture Notes in Comput. Sci., vol. 3269, Springer, 2004, pp. 495–507.
[18] F. Laguillaumie, D. Vergnaud, Short undeniable signatures without random oracles: The missing link, in: S. Maitra, C.E.V. Madhavan, R. Venkatesan (Eds.), INDOCRYPT, in: Lecture Notes in Comput. Sci., vol. 3797, Springer, 2005, pp. 283–296.
[19] A. Menezes, N.P. Smart, Security of signature schemes in a multi-user setting, Des. Codes Cryptogr. 33 (3) (2004) 261–274.
[20] W. Ogata, K. Kurosawa, S.-H. Heng, The security of the FDH variant of Chaum's undeniable signature scheme, in: S. Vaudenay (Ed.), Public Key Cryptography, in: Lecture Notes in Comput. Sci., vol. 3386, Springer, 2005, pp. 328–345.
[21] S. Saeednia, S. Kremer, O. Markowitch, An efficient strong designated verifier signature scheme, in: J.I. Lim, D.H. Lee (Eds.), ICISC, in: Lecture Notes in Comput. Sci., vol. 2971, Springer, 2003, pp. 40–54.
[22] W. Susilo, F. Zhang, Y. Mu, Identity-based strong designated verifier signature schemes, in: H. Wang, J. Pieprzyk, V. Varadharajan (Eds.), ACISP, in: Lecture Notes in Comput. Sci., vol. 3108, Springer, 2004, pp. 313–324.
[23] C.H. Tan, Signcryption scheme in multi-user setting without random oracles, in: K. Matsuura, E. Fujisaki (Eds.), IWSEC, in: Lecture Notes in Comput. Sci., vol. 5312, Springer, 2008, pp. 64–82.
[24] D. Vergnaud, New extensions of pairing-based signatures into universal designated verifier signatures, in: M. Bugliesi, B. Preneel, V. Sassone, I. Wegener (Eds.), ICALP (2), in: Lecture Notes in Comput. Sci., vol. 4052, Springer, 2006, pp. 58–69.
[25] W. Xue, Y. Liu, K. Li, Z. Chi, G. Min, W. Qu, Dhtrust: a robust and distributed reputation system for trusted peer-to-peer networks, Concurr. Comput. Practice and Experience 24 (10) (2012) 1037–1051.
[26] Y. Yu, K. Li, W. Zhou, P. Li, Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures, J. Netw. Comput. Appl. 35 (3) (2012) 867–880.
[27] J. Zhou, Non-repudiation, PhD thesis, University of London, UK, 1996.
[28] J. Zhou, Non-Repudiation in Electronic Commerce, Artech House, 2001.
[29] J. Zhou, D. Gollmann, Evidence and non-repudiation, J. Netw. Comput. Appl. 20 (3) (1997) 267–281.