Threshold Signatures with Private Accountability via Secretly Designated Witnesses

Meng Li[1,2,3], Hanni Ding[1,2,3], Qing Wang[1,2,3], Zijian Zhang[4(✉)], and Mauro Conti[5,6]

[1] Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology,Hefei, China
mengli@hfut.edu.cn, {hanniding,qingwang}@mail.hfut.edu.cn
[2] Ministry of Education; School of Computer Science and Information Engineering, Hefei University of Technology,Hefei, China
[3] Anhui Province Key Laboratory of Industry Safety and Emergency Technology; and Intelligent Interconnected Systems Laboratory of Anhui Province, Hefei University of Technology,Hefei, China
[4] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China
zhangzijian@bit.edu.cn
[5] Department of Mathematics and HIT Center, University of Padua, Padua, Italy
mauro.conti@math.unipd.it
[6] Department of Intelligent Systems, CyberSecurity Group, Delft University of Technology, Delft, The Netherlands

Abstract. Threshold signature is a powerful cryptographic technique with a large number of real-life applications. As designed by Boneh and Komlo (CRYPTO'22), TAPS is a new threshold signature integrating privacy and accountability. It allows a combiner to combine t signature shares while protecting t and the signing group from the public. It also enables a tracer to trace a threshold signature to its original signing group. Despite being valuable, TAPS neglects the witnessing of tracing, i.e., leaves the tracing activity unrestrained.

In this paper, we introduce Accountable and Private Threshold Signature with Hidden Witnesses (HiTAPS) that not only provides privacy and accountability, but also incorporates witnessed tracing. In specific, we first utilize Dynamic Threshold Public-Key Encryption (DTPKE) and ElGamal encryption to designate a set of t' witnesses for endorsing the tracing activity. We then compute a keyed-hash tag for the t' witnesses to initiate the tracing activity secretly. Moreover, we present an optimized protocol HiTAPS2 to reduce communication overhead of the combiner. We formalize the definitions, security, and privacy for HiTAPS. We formally prove its security and privacy. To evaluate the performance of HiTAPS and HiTAPS2, we build a prototype based on pypbc. Experimental results show that HiTAPS takes 217 (370) ms to combine (track) a threshold signature of 5 signers (witnesses). The optimized HiTAPS2 only takes 137 ms to combine a threshold signature of 5 signers.

Keywords: Threshold Signatures · Privacy · Accountability · Witness

T. Zhu and Y. Li (Eds.): ACISP 2024, LNCS 14895, pp. 389–407, 2024.
https://doi.org/10.1007/978-981-97-5025-2_20

1 Introduction

1.1 Background

A Threshold Signature Scheme (TSS) [1] allows a set of n people to sign a message via a combiner when no less than t people join in the signing process. The system model is sketched in Fig. 1. Among its multiple variants, Private Threshold Signature (PTS) [2–4] and Accountable Threshold Signature (ATS) [5–8] stand out. A PTS signature σ on a message m tells nothing about the group of t signers who generated σ, which is useful since security and privacy are increasingly gaining importance [9,10]. An ATS signature σ on a message m can disclose the identity of all t people who co-generate σ via a tracer. For this reason, ATS is also considered as traceable secret sharing [11]. ATS can be used in real-world applications where accountability is required. For instance, if five of nine cooperative manufacturers prepare to authorize a product transfer, and all of them expect accountability in case a fraudulent transfer is consented. By using an ATS scheme, a Threshold Signature (TS) on a fraudulent transfer can disclose the five manufacturers who approved of it.

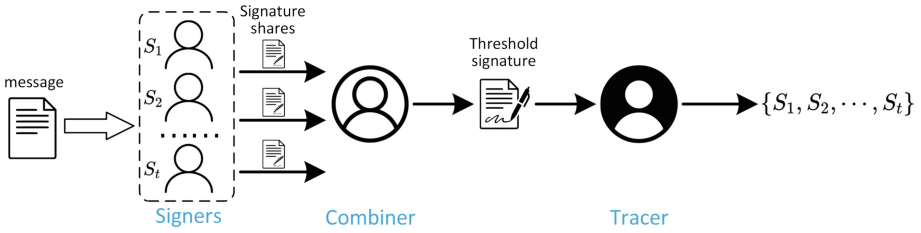

Fig. 1. System Model of ATS.

1.2 Motivation and Goal

Recently, Boneh and Komlo proposed a Threshold, Accountable, and Private Signature (TAPS) [12] to achieve both accountability and privacy for threshold signatures. It works as below: It generates a public key pk and n private keys $\{sk_1, sk_2, \cdots, sk_n\}$ for n signers; t signers $\mathcal{S} = \{S_1, S_2, \cdots, S_t\}$ generate t signature shares for a combiner with a combining key ck to generate a complete signature; and a tracer with a tracing key tk can identify the signing group of a complete signature. Here, accountability guarantees that a TS that is related to a misbehavior can be traced to its t signers. Meanwhile, privacy refers to the fact that any signing group and t are kept secret from the public.

Based on the observations on TAPS, our motivation arises from two aspects. **M1. Unwitnessed Tracing.** Tracing a TS σ to its signing quorum is a formidable capability that should be "kept in a box" at ordinary times. If not

properly restrained, the "almighty" tracer can trace any σ to its t signers. Followed by the motivation above, we are driven to achieve **M2. Secretly Designated Witnessing**, i.e., designate another group of witnesses to sanction the tracing while keeping their identities secret.

1.3 Possible Solutions and Technical Challenges

Intuitively, one can first ask the combiner to encrypt the TS σ by using the t' witnesses' public keys and then ask each of them to decrypt the ciphertext right before tracing. However, this will expose σ to all witnesses and increase the risk of leakage. In this work, we resort to the Dynamic Threshold Public-Key Encryption (DTPKE) [13], which allows (1) a sender (combiner) to dynamically choose the authorized group of recipients (witnesses) for a ciphertext (TS), and (2) a set of witnesses to decrypt a ciphertext when a threshold of authorized witnesses collaborate. Such two properties shed light on a promising approach. Still, we have to tackle two technical challenges:

C1. How to awake the t' witnesses to share-decrypt the encrypted TS without exposing their identities? We assume that neither the combiner nor the tracer is aware of the t' witnesses at any phase of the protocol given the identity privacy of witnesses. It is not feasible to ask the combiner or the tracer to "contract" the pertinent witnesses during tracing. Thus, it leaves us to design a method for the ($\leq t$) signers to designate the t' witnesses secretly.

C2. How to prove the validity of a TS that is already encrypted by the combiner via layered encryption? In TAPS, the combiner generates a Non-Interactive Zero Knowledge Proof (NIZKP) π that the output signature σ_m is a valid ATS signature on m. However, we intend to protect σ_m from public including the tracer via DTPKE as well as a second layer of encryption due to designation. This in turn makes it challenging for the combiner to prove the signature validity.

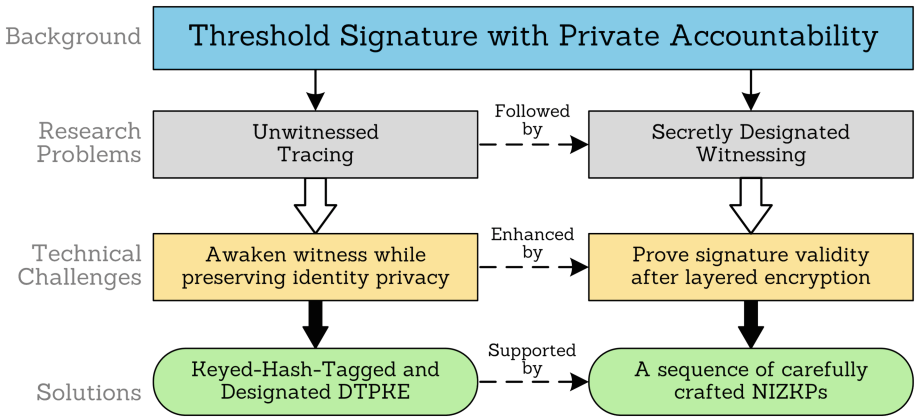

Fig. 2. Technical Roadmap of HiTAPS.

1.4 Our Solutions

To solve the first challenge, we design a Keyed-Hash-Tagged and Designated DTPKE using three carefully structured cryptographic primitives. Specifically, we first designate t' witnesses during signing, encrypt the TS σ_m with DTPKE to obtain $\hat{\sigma}_m$ during combining to protect σ_m from arbitrary tracing, encrypt $\hat{\sigma}_m$ with PKE to obtain t' versions of encrypted $\hat{\sigma}_m$ for t' witnesses, and compte t' keyed-hash tags of the message and witness identities. In this way, we lay the groundwork for awakening secret awakening while preserving the identity privacy of the witnesses.

To solve the second challenge, we carefully craft a sequence of NIZKPs for the combiner to prove the validity of t' ciphertexts sent by the combiner. Specifically, the combiner has to prove the validity of σ_m, the encryption of σ_m, the encryption of $\hat{\sigma}_m$, and the possession of the combining key. By doing so, we pay the way for systematically proving the validity of σ_m after σ_m is encrypted.

We portray our technical roadmap in Fig. 2. Note that, given the high communication overhead of the t' ciphertexts, we further design an optimized version of HiTAPS by computing an aggregated ciphertext at the combiner.

2 Related Work

In this section, we first revisit some related work, including PTS, ATS, and TAPS. Then we discuss how our work advances the state-of-the-art.

Shoup [2] proposed an RSA threshold signature scheme that provided unforgeability and robustness in the random oracle model and made the generation and verification non-interactive. Stern et al. [3] presented new techniques to fully distribute RSA, i.e., generated RSA moduli for Shoup's scheme without a trusted dealer. Koprowski et al. [4] designed a threshold RSA scheme as efficient as Shoup's scheme while not depending on two previously used assumptions and building its robustness on an intractability assumption.

Micali et al. [5] formalized Accountable-Subgroup Multisignatures (ASM) where a subgroup $\mathcal{S}$ of signers was enabled to sign a message m and the resulting signature σ provably discloses the identities of the signers in $\mathcal{S}$ to any verifier without the help of any trusted third parties. Boneh et al. [6] proposed a short signature scheme building upon Gap-Diffie-Hellman groups with small representations. Bellare et al. [7] removed the key-setup requirements (it is not necessary for a signer to have a secret key) and proposed a multi-signature scheme in the plain public-key model that is secure in the random-oracle model. Nick et al. [8] presented a two-round multi-signature scheme that is secure under concurrent signing sessions, supports key aggregation, produces Schnorr signatures, and needs two communication rounds.

TAPS [12] is a novel TS that achieves both privacy and accountability. It protects not only the threshold t, but also the t signers via keeping the tracking key tk to the sole tracer. Based on TAPS, our proposed HiTAPS concentrates on the *secure authorization of tracing for TSs*. It constitutes an important piece

of the puzzle. To the best of our knowledge, this problem is not very understood yet, and we aim to fill the gap.

3 Problem Formulation

In this section, we formalize the notion of Accountable and Private Threshold Signature with Hidden Witnesses (HiTAPS), including the system model, definition, unforgeability and accountability, and privacy. We use n for the total number of signers, t for the threshold number of required signers in combining, t' for the threshold number of required witnesses in tracing.

3.1 System Model

The system model of HiTAPS is depicted in Fig. 3. It consists of n signers $\{S_1, S_2, \cdots, S_n\}$, t' witnesses $\{W_1, W_2, \cdots, W_{t'}\}$, a combiner C, and a tracer T. Next, we describe how they work in the system.

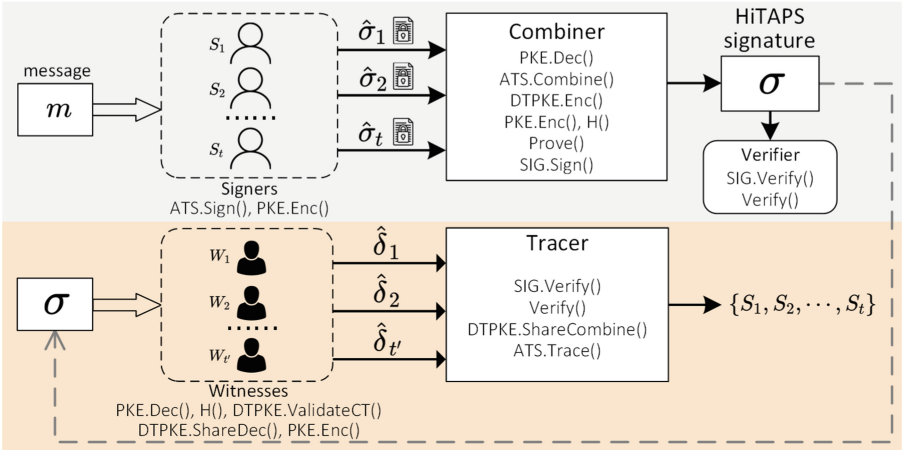

Fig. 3. System Model of HiTAPS.

Signer. A set of t signers $\mathcal{S} = \{S_1, S_2, \cdots, S_t\}$ belonging to a bigger set of signers $\{S_1, S_2, \cdots, S_n\}$ $(t \leq n)$ decides to cooperatively generate a signature on a message m. Each signer S_i has a private key sk_i^s. First, they select a set of t' witness $\mathcal{W} = \{W_1, W_2, \cdots, W_{t'}\}$. Then, each S_i computes a signature share σ_i on m and encrypts σ_i with the combiner C's public key pk^c to obtain $\widehat{\sigma}_i$. Next, each S_i sends $\widehat{\sigma}_i$ and an encrypted list $\widehat{\mathcal{W}}$ of $\mathcal{W}$ to C.

Witness. The selected set $\mathcal{W}$ belongs to a bigger set $\{W_1, W_2, \cdots, W_w\}$ $(t' \leq w)$. Each witness W_i has a public key (pk_i^w, upk_i) as his identity. Upon the call of tracing, W_i computes a keyed hash tag and compares it with the ones

on a public bulletin. If a match is found, W_i will decrypt it to have a HiTAPS signature σ, verifies it, and share-decrypts it to obtain a decryption share δ_i. Next, W_i sends $\widehat{\delta}_i$, i.e., the encryption of δ_i under the tracer T's public key pk^t, to C.

Combiner. The combiner C receives the encrypted signature shares and encrypted lists from the signers. C decrypts them to obtain the signature shares $\{\sigma_i\}_{i=1}^{t}$ in plaintext. C combines t signature shares to compute an ATS signature σ_m and encrypts it via DTPKE and an encryption key ek^c. C also decrypts the encrypted list $\widehat{\mathcal{W}}$ to know which witnesses to send to. After checking all the lists, C encrypts σ_m with the pk_i^w of t' witnesses to get a triad (C_{i1}, C_{i2}, C_{i3}) and computed a keyed hash tag ht_i. We assume that C and each W_i shares a secret key ssk_i. Furthermore, C generates a proof π and a signature $\hat{\sigma}$, and submits them to the public bulletin.

Tracer. The tracer T receives the encrypted decryption shares from witnesses. T decrypts them to get decryption shares and then combines them to obtain σ_m. Lastly, T traces σ_m to its original signing group if σ_m is a valid ATS signature.

3.2 Definition

Definition 1. An accountable and private threshold signature with hidden witness, or HiTAPS, is a tuple of five polynomial time algorithms $\Pi = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Combine}, \mathsf{Verify}, \mathsf{Trace})$ where

- $\mathsf{Setup}(1^\lambda, n, w, t, t') \rightarrow (PK, \{sk_i^s\}_{i=1}^{n}, \{wsk_j, ssk_j, sk_j^{\text{wit}}\}_{j=1}^{w}, ck, tk, SK)$ is a probabilistic algorithm that takes as input a security parameter λ, the number of signers n, the number of witnesses w, a threshold t, and another threshold t' to output a public key PK, a set of n signing keys $\{sk_i^s\}_{i=1}^{n}$, a set of witness private keys $\{wsk_j\}_{j=1}^{w}$, a set of shared secret keys $\{ssk_j\}_{j=1}^{w}$, a set of private keys $\{sk_j^{\text{wit}}\}_{j=1}^{w}$, three private keys sk_c^{enc}, sk_c^{sig}, sk_t^{enc}, a combining key ck, a tracing key tk, and a secret key SK, an encryption key EK, and a combining key CK.
- $\mathsf{Sign}(m, \mathcal{S}, \{sk_i^s\}_{i=1}^{t}, \mathcal{W}) \rightarrow (\{\widehat{\sigma}_i, \widehat{\mathcal{W}}_i\})$ is a probabilistic algorithm run by a set of signers that takes as input a set of signers $\mathcal{S}$ and a corresponding set of t signing keys $\{sk_i^s\}_{i=1}^{t}$, a message m in message space $\mathcal{M}$, and a set of t' witnesses, to output a set of encrypted signature shares $\{\widehat{\sigma}_i\}$ on m and a set of encrypted list $\{\widehat{\mathcal{W}}_i\}$. For simplicity, we denote $\{sk_i^s\}_{i=1}^{t}$ as the set of t secret keys from any set of t signers in $\{S_1, S_2, \cdots, S_n\}$.
- $\mathsf{Combine}(m, t, t', PK, \mathcal{S}, \{\widehat{\sigma}_i\}, \{\widehat{\mathcal{W}}_i\}, ck, EK) \rightarrow \sigma$ is a probabilistic algorithm run by the combiner that takes as input a message m, two thresholds t and t', the public key PK, a set of signers $\mathcal{S}$, a set of encrypted signature shares $\{\widehat{\sigma}_i\}$ on m and a set of encrypted list $\{\widehat{\mathcal{W}}_i\}$, a combining key ck, and an encryption key EK to output a HiTAPS signature σ.
- $\mathsf{Verify}(PK, m, \sigma) \rightarrow \{0, 1\}$ is a deterministic algorithm run by the public and the tracer that verifies a HiTPAS signature σ on a message m with respect to the public key PK.

– $\mathsf{Trace}(m, PK, tk, \sigma) \rightarrow \mathcal{S}$ is a deterministic algorithm run by the tracer that takes as input a message m, the public key PK, a tracing key tk, and a HiTAPS signature σ to output a set $\mathcal{S}$ who previously generated σ or a failure symbol $\perp$.

For *correctness*, we require that for all $t \in [n]$, all t-size sets $\mathcal{S}$, all $t' \in [w]$, all t'-size set $\mathcal{W}$, all $m \in \mathcal{M}$, and all the $(PK, \{sk_i^s\}_{i=1}^n, \{ssk_j\}_{j=1}^w, \{sk_j\}_{i=1}^w, ck$ $tk, SK) \leftarrow \mathsf{Setup}(1^\lambda, n, t)$, the following two conditions hold:

$$\Pr[\mathsf{Verify}(m, PK, \mathsf{Combine}(ck, sk^s, m, \mathcal{S}, \mathsf{Sign}(m, \mathcal{S}, \{sk_i^s\}_{i=1}^t, \mathcal{W}))) = 1] = 1,$$
$$\Pr[\mathsf{Trace}(m, tk, \mathsf{Combine}(ck, sk^s, m, \mathcal{S}, \mathsf{Sign}(m, \mathcal{S}, \{sk_i^s\}_{i=1}^t, \mathcal{W}))) = \mathcal{S}] = 1.$$

3.3 Unforgeability and Accountability

Before we dive into privacy, HiTAPS must satisfy the standard notion of existential unforgeability against a chosen message attack (EUF-CMA) [14,15] like any signature scheme, i.e., an adversary compromising fewer than t signers cannot generate a valid message-signature pair. Meanwhile, HiTAPS should be accountable, i.e., a tracer holding a tracing key tk can output the correct set of signers for a correct message-signature pair. We formalize these two properties in the adversarial experiment in Fig. 4. Let $\mathbf{Adv}^{\text{forg}}_{\mathcal{A},\Pi}(\lambda)$ be the probability that $\mathcal{A}$ wins the experiment $\mathbf{Exp}^{\text{unf-acc}}$ against the HiTAPS scheme Π.

Definition 2 (Unforgeability and Accountability). A HiTAPS scheme Π is **unforgeable and accountable** if for all Probabilistic Polynomial Time (PPT) adversaries $\mathcal{A}$, there is a negligible function negl such that $\mathbf{Adv}^{\text{forg}}_{\mathcal{A},\Pi}(\lambda) \leq \mathsf{negl}(\lambda)$.

$\mathbf{Exp}^{\text{unf-acc}}$

1. $(n, w, t, t', \mathcal{S}, \mathsf{state}) \xleftarrow{\$} \mathcal{A}(1^\lambda)$ where $t \in [n]$, $t' \in [w]$, $\mathcal{S} \subseteq [n]$
2. $(PK, \{sk_i^s\}_{i=1}^n, \{wsk_j, ssk_j, sk_j^{\text{wit}}\}_{i=1}^w, ck, tk, SK) \leftarrow \mathsf{Setup}(1^\lambda, n, w, t, t')$
3. $(m', \sigma') \xleftarrow{\$} \mathcal{A}^{\mathcal{O}(\cdot,\cdot)}(PK, \{sk_i^s\}_{S_i \in \mathcal{S}}, \mathcal{W}, \{wsk_j, ssk_j, sk_j^{\text{wit}}\}_{i=1}^w, ck, tk, SK, \mathsf{state})$

where $\mathcal{O}(\mathcal{S}_j, m_i)$ returns $\mathsf{Sign}(m_i, \mathcal{S}_j, \{sk_i^s\}_{S_i \in \mathcal{S}_j}, \mathcal{W}_j)$

$\mathcal{A}$'s winning conditions:

Let $(\mathcal{S}_1, m_1), (\mathcal{S}_2, m_2), \cdots$ be $\mathcal{A}$'s queries to $\mathcal{O}$

Let $\mathcal{S} \leftarrow \cup \mathcal{S}_i$, union over all queries to $\mathcal{O}(\mathcal{S}_j, m')$,
if no such queries, set $\mathcal{S}_j = \emptyset$

let $\mathcal{S}_t \leftarrow \mathsf{Trace}(m', tk, PK, \sigma', \hat{\sigma}')$

Output 1 if $\mathsf{Verify}(m', PK, \sigma', \hat{\sigma}') = 1$ and either $\mathcal{S}_t \nsubseteq \mathcal{S} \cup \mathcal{S}'$ or if $\mathcal{S}_t = \mathsf{fail}$

Fig. 4. Experiment of Unforgeability and Accountability.

3.4 Privacy

Now we define privacy for HiTAPS. Usually, the privacy for a TSS is formalized by requiring that a TS on a message m be computationally indistinguishable

from a standard signature on m [16]. It ensures that a TS tells nothing about the threshold t or the set of signers that generated the TS.

We give three informal privacy requirements as below and they are further captured by the experiments in Fig. 5 and Fig. 6, respectively.

- **Privacy against public:** A party who only has (PK, n, w) and observes a series of $(m, \sigma, \hat{\sigma})$ triads, learns nothing about t, t' or the set of signers that generated the observed $(\sigma, \hat{\sigma})$.
- **Privacy against signer:** A set of all signers who only has (PK, n, w, t, t') and observes a series of $(m, \sigma, \hat{\sigma})$ triads, learns nothing about the set of signers that generated the observed $(\sigma, \hat{\sigma})$.

Let E be the event that the experiment $\mathbf{Exp}^{\text{privP}}$ in Fig. 5 outputs 1 and E' be the event that the experiment $\mathbf{Exp}^{\text{privS}}$ in Fig. 6 outputs 1. We define the two advantage functions for an adversary $\mathcal{A}$ against the HiTAPS scheme Π, as a function of the security parameter λ:

$$\mathbf{Adv}^{\text{privP}}_{\mathcal{A},\Pi}(\lambda) = |2\Pr[E] - 1|, \ \mathbf{Adv}^{\text{privS}}_{\mathcal{A},\Pi}(\lambda) = |2\Pr[E'] - 1|.$$

Definition 3 (Privacy). A HiTAPS scheme is **private** if for all PPT adversaries $\mathcal{A}$, $\mathbf{Adv}^{\text{privP}}_{\mathcal{A},\Pi}(\lambda)$ and $\mathbf{Adv}^{\text{privS}}_{\mathcal{A},\Pi}(\lambda)$ are negligible functions of λ.

$b_0 \xleftarrow{\$} \{0,1\}$, $b_1 \xleftarrow{\$} \{0,1\}$, $(t_0, t_1, t'_0, t'_1, \mathsf{state}) \xleftarrow{\$} \mathcal{A}(1^\lambda)$ where $t_0, t_1 \in [n], t'_0, t'_1 \in [w]$
$(PK, \{sk_i^s\}_{i=1}^n, \{wsk_j, ssk_j, sk_j^{\text{wit}}\}_{i=1}^w, ck, tk, SK) \leftarrow \mathsf{Setup}(1^\lambda, n, w, t_{b_0}, t'_{b_1})$
$(b'_0, b'_1) \leftarrow \mathcal{A}^{\mathcal{O}_1(\cdot,\cdot,\cdot,\cdot,\cdot),\mathcal{O}_2(\cdot,\cdot,\cdot)}(PK, \mathsf{state})$
Output $(b'_0 = b_0) \wedge (b'_1 = b_1)$.
where $\mathcal{O}_1(m, \mathcal{S}_0, \mathcal{S}_1, \mathcal{W}_0, \mathcal{W}_1)$ returns
  $(\sigma, \hat{\sigma}) \leftarrow \mathsf{Combine}(m, \mathcal{S}_{b_0}, \mathsf{Sign}(m, \mathcal{S}_{b_0}, \{sk_i^s\}_{S_i \in S_{b_0}}, \mathcal{W}_{b_1}), ck, EK)$
  for $\mathcal{S}_0, \mathcal{S}_1 \subseteq [n]$, $|\mathcal{S}_0| = t_0$ and $|\mathcal{S}_1| = t_1$, $\mathcal{W}_0, \mathcal{W}_1 \subseteq [w]$, $|\mathcal{W}_0| = t'_0$ and $|\mathcal{W}_1| = t'_1$
$\mathcal{O}_2(m, \sigma, \hat{\sigma})$ returns $\mathsf{Trace}(m, tk, PK, \sigma, \hat{\sigma})$.
Restriction: if $(\sigma, \hat{\sigma})$ is returned by $\mathcal{O}_2$, then $(m, \sigma, \hat{\sigma})$ will not be sent to $\mathcal{O}_3$.

Fig. 5. $\mathbf{Exp}^{\text{privP}}$: Experiment of Privacy against the Public.

In $\text{Exp}^{\text{privP}}$, $\mathcal{A}$ generates four thresholds t_0, t_1, t'_0 and t'_1 in $[n]$ and is given PK. $\mathcal{A}$ submits a string of signature queries to a signing oracle $\mathcal{O}_1$, where each query contains a message m and four sets $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{W}_0$, and $\mathcal{W}_1$. Then, $\mathcal{A}$ receives a signature generated using either $\mathcal{S}_0$ or $\mathcal{S}_1$ (same for $\mathcal{W}_0$ or $\mathcal{W}_1$). $\mathcal{A}$ can access a tracing oracle $\mathcal{O}_2$ while not being able to determine whether the string of signatures it observed related to the left or the right sequence of sets.

In $\text{Exp}^{\text{privS}}$, $\mathcal{A}$ generates (t, t'), and is given all the signing keys. Same as $\text{Exp}^{\text{privP}}$, $\mathcal{A}$ cannot determine whether $\mathcal{O}_1$ that takes four sets $\mathcal{S}_0$, $\mathcal{S}_1$, $\mathcal{W}_0$, and $\mathcal{W}_1$ responds using either $\mathcal{S}_0$ or $\mathcal{S}_1$ (same for $\mathcal{W}_0$ or $\mathcal{W}_1$).

$b_0 \xleftarrow{\$} \{0,1\}$, $b_1 \xleftarrow{\$} \{0,1\}$, $(t, t', \mathsf{state}) \xleftarrow{\$} \mathcal{A}(1^\lambda)$ where $t_0, t_1 \in [n], t'_0, t'_1 \in [w]$
$(PK, \{sk_i\}_{i=1}^n, \{wsk_j, ssk_j, sk_j^{\mathrm{wit}}\}_{i=1}^w, ck, tk, SK) \leftarrow \mathsf{Setup}(1^\lambda, n, w, t, t')$
$(b'_0, b'_1) \leftarrow \mathcal{A}^{\mathcal{O}_1(\cdot,\cdot,\cdot,\cdot,\cdot),\mathcal{O}_2(\cdot,\cdot,\cdot,\cdot,\cdot),\mathcal{O}_3(\cdot,\cdot,\cdot)}(PK, \{sk_i^s\}_{i=1}^n, \mathsf{state})$
Output $(b'_0 = b_0) \wedge (b'_1 = b_1)$.
where $\mathcal{O}_1(m, \mathcal{S}_0, \mathcal{S}_1, \mathcal{W}_0, \mathcal{W}_1)$ returns $\widehat{\sigma}_i \leftarrow \mathsf{PKE.Enc}(pk_c^{\mathsf{enc}}, m||\sigma_i)$ and $\widehat{\mathcal{W}_i} \leftarrow \mathsf{PKE.Enc}(pk_c^{\mathsf{enc}}, \mathcal{W}_{b_1})$ for $S_i \in \mathcal{S}_{b_0}$ where $\sigma_i \leftarrow \mathsf{ATS.Sign}(sk_i^s, m)$
$\mathcal{O}_2(m, \mathcal{S}_0, \mathcal{S}_1, \mathcal{W}_0, \mathcal{W}_1)$ returns
  $(\sigma, \hat{\sigma}) \leftarrow \mathsf{Combine}(m, \mathcal{S}_{b_0}, \mathsf{Sign}(m, \mathcal{S}_{b_0}, \{sk_i^s\}_{S_i \in S_{b_0}}, \mathcal{W}_{b_1}), ck, EK)$
  for $\mathcal{S}_0, \mathcal{S}_1 \subseteq [n]$, $|\mathcal{S}_0| = |\mathcal{S}_1| = t$, $\mathcal{W}_0, \mathcal{W}_1 \subseteq [w]$, $|\mathcal{W}_0| = |\mathcal{W}_1| = t'$
$\mathcal{O}_3(m, \sigma, \hat{\sigma})$ returns $\mathsf{Trace}(m, tk, PK, \sigma, \hat{\sigma})$.
Restriction: if $(\sigma, \hat{\sigma})$ is returned by $\mathcal{O}_1$, then $(m, \sigma, \hat{\sigma})$ will not be sent to $\mathcal{O}_2$.

Fig. 6. $\mathbf{Exp}^{\mathrm{privS}}$: Experiment of Privacy against the Signers.

4 Preliminaries

In this section, we revisit and review some preliminaries, including notations, ATS, DTPKE, ElGamal encryption, and hash function.

4.1 Notations

Since there are many notations used in this work, we list the key notations in Table 1, including number of all signers n, number of required signers t, etc.

4.2 ATS

An accountable threshold signature is a tuple of five polynomial time algorithms $(\mathsf{Setup}, \mathsf{Sign}, \mathsf{Combine}, \mathsf{Verify}, \mathsf{Trace})$ invoked as

$$(\{pk_i^s, sk_i^s\}_{i=1}^n) \leftarrow \mathsf{ATS.Setup}(1^\lambda, n, t),\ \sigma_i \leftarrow \mathsf{ATS.Sign}(sk_i^s, m),$$
$$\sigma_m \leftarrow \mathsf{ATS.Combine}(t, \{pk_i^s\}_{i=1}^n, m, \mathcal{S}, \{\sigma_i\}_{i\in\mathcal{S}}),$$
$$\{0,1\} \leftarrow \mathsf{ATS.Verify}(t, \{pk_i^s\}_{i=1}^n, m, \sigma_m),\ \mathcal{S} \leftarrow \mathsf{ATS.Trace}(t, \{pk_i^s\}_{i=1}^n, m, \sigma_m).$$

An ATS is secure if it is unforgeable and accountable, i.e., if for every PPT adversary $\mathcal{A}$, the function $\mathbf{Adv}_{A,\mathsf{ATS}}^{\mathrm{forg}}$ of winning an unforgeability and accountability attack game is a negligible function of λ [12,17].

4.3 DTPKE

A dynamic threshold public-key encryption is a tuple of six polynomial algorithms $(\mathsf{Setup}, \mathsf{Join}, \mathsf{Encrypt}, \mathsf{ValidateCT}, \mathsf{ShareDecrypt}, \mathsf{Combine})$ invoked as

$$(SK, EK, CK) \leftarrow \mathsf{Setup}(1^\lambda),\ (wpk, wsk) \leftarrow \mathsf{Join}(SK, id),$$
$$(C_0, C_1, C_2, C_3) \leftarrow \mathsf{Enc}(Ek, t', m),\ \{0,1\} \leftarrow \mathsf{ValidateCT}(EK, t', C_0, C_1),$$
$$\sigma_{id} \leftarrow \mathsf{ShareDecrypt}(id, wsk, C_0, C_1),$$
$$\sigma_m \leftarrow \mathsf{Combine}(CK, t', C_0, C_1, C_2, C_3, \mathcal{W}, \{\sigma_m^j\}_{W_j\in\mathcal{W}}).$$

Table 1. Key Notations

Notation	Meaning	Notation	Meaning
TSS	Threshold signature scheme	TS	Threshold signature
PTS	Private threshold signature	ATS	Accountable threshold signature
n	Number of all signers	t	Number of required signers
w	Number of all witnesses	t'	Number of required witnesses
S_i	Signer i	W_j	Witness j
$\mathcal{S}$	Set of required signers	$\mathcal{W}$	Set of required witnesses
C	Combiner	T	Tracer
λ	Security parameter	PK	Public key
pk_j^w, wpk_j	W_j's public key	sk_j^{wit}, wsk_j	W_j's private key
$pk_c^{\text{enc}}, pk_c^{\text{sig}}$	C's Public key	$sk_c^{\text{enc}}, sk_c^{\text{sig}}$	C's private key
pk_t^{enc}	T's public key	sk_t^{enc}	T's private key
(sk_i^s, pk_i^s)	S_i's signing key pair	ssk	Shared secret key
SK	Secret key	EK	Encryption key
CK	Combining key	m	Message
ck	Combining key	tk	Tracing key
σ_i	S_i's signature	$\widehat{\sigma}_i$	Encrypted signature
σ_m	ATS signature	$\widehat{\sigma}_m$	Encrypted ATS signature
$\widehat{\mathcal{W}}$	Encrypted witness list	ht	Keyed hash tag
δ_i	W_i's share decryption	$\widehat{\delta}_i$	Encrypted share decryption
π	NIZKP	σ	HiTAPS signature

Its non-adaptive adversary, non-adaptive corruption, chosen-plaintext attacks (IND-NAA-NAC-CPA) security is based on the Multi-sequence of Exponents Diffie-Hellman (MSE-DDH) assumption, where $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{A},\mathsf{DTPKE}}(l, m, t') \leq \mathbf{Adv}^{\text{mse-ddh}}(l, m, t')$ [13,18,19].

4.4 ElGamal Encryption

The ElGamal encryption scheme PKE is a tuple of three polynomial algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ invoked as

$$(sk, pk) \leftarrow \mathsf{KeyGen}(1^\lambda),\ (C_0, C_1) \leftarrow \mathsf{Enc}(pk, m),\ m \leftarrow \mathsf{Dec}(sk, (C_0, C_1)).$$

The ElGamal encryption scheme is secure against Chosen Plaintext Attack (CPA) if the decisional Diffie-Hellman (DDH) problem [17].

4.5 Hash Function

A cryptographic hash function with output length $l(\lambda)$ is a tuple of two polynomial algorithms $(\mathsf{Gen}, \mathsf{H})$ invoked as

$$k \leftarrow \mathsf{Gen}(\lambda), \{0,1\}^{l(\lambda)} \leftarrow \mathsf{H}(k, m),\ m \in \{0,1\}^*.$$

A hash function H is preimage resistant if for every PPT adversary $\mathcal{A}$, there is a negligible function negl such that $\Pr[\mathsf{Hash\text{-}pre}_{\mathcal{A},\mathsf{H}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$ [17].

4.6 PKE, COM, and SIG

A commitment scheme COM is a pair of algorithms $(\mathsf{Comm}, \mathsf{Verify})$ invoked as

$$com \leftarrow \mathsf{Comm}(x, r),\ \{0,1\} \leftarrow \mathsf{Verify}(x, r, com).$$

A COM scheme is secure if it is unconditionally hiding and computationally binding, i.e., for every PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\text{blind}}_{\mathcal{A},\text{COM}}(\lambda)$ is negligible.

A signature scheme SIG is a triple of algorithms $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ invoked as

$$(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda),\ \sigma \leftarrow \mathsf{Sign}(sk, m),\ \{0,1\} \leftarrow \mathsf{Verify}(pk, m, \sigma).$$

A SIG scheme is strongly unforgeable if for every PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{A},\text{SIG}}(\lambda)$ is negligible.

5 The Proposed Scheme HiTAPS

We show the construction of HiTAPS in Fig. 7. In Setup, all system parameters are generated for the ATS, DTPKE, commitment scheme COM, ElGamal encryption PKE, signature scheme SIG, hash function H, including the secret keys, public keys, shared secret keys, and a commitment.

In Sign, t signers from one signer set $\mathcal{S}$ generate t signature shares $\{\sigma_i\}_{S_i \in \mathcal{S}}$ on a message m. The signature shares are further encrypted into $\{\widehat{\sigma}_i\}$ with C's public key pk_c^{enc} and then sent to C along with encrypted witness lists $\widehat{\mathcal{W}}$.

In Combine, C first decrypts $\widehat{\mathcal{W}}$ to collect encrypted signature shares belonging to some signer set $\mathcal{S}$ and then decrypts them to recover $\{\sigma_i\}_{S_i \in \mathcal{S}}$. Next, C combines them to obtain an ATS signature σ_m and encrypts it into $\widehat{\sigma}_m$ with DTPKE. Then, C encrypts (C_0, C_1) in $\widehat{\sigma}_m$ with the public keys of witnesses in $\mathcal{W}$. Here, (C_0, C_1) is encrypted because they can be used to recover $C_2 = k$ to decrypt C_3. For instance, $\mathsf{ElGamal.Enc}(pk_1^w, C_0) = (C_{10}, C_{11}) = (C_0 g^{sk_1^w r_1}, g^{r_1})$ where $pk_1^w = g^{sk_1^w}$. C also computes t' hash tag $ht_j = \mathsf{H}(ssk_j, m||W_j)$ $(W_j \in \mathcal{W})$ as a secret token to awake W_j into share-decrypt $\widehat{\sigma}_m$. Here, the t' witnesses do not actively participate in the process. C generates a NIZPK π to prove the validity of a TS. For example, to prove $C_{10} = C_0 g^{sk_1^w r_1}$ while keeping C_0, sk_1^w, r_1 secret, we first transfer it into $C_{10} = g^x g^{sk_1^w r_1} = g^{x+sk_1^w r_1}$ for simplicity. Now the combiner (Prover) and a verifier proceed as follows. *Prover*: (1) set $A = C_0 g^{sk_1^w r_1}$ where $C_0 = g^x$; (2) compute $B = g$, $\delta = \mathsf{H}(g||A)$, $C = g^\delta$, and $D = g^{(x+sk_1^w r_1)/\delta}$. *Verifier*: check $e(A, B) \stackrel{?}{=} e(C, D)$.

In Verify, a verifier verifies the validity of σ by checking the signature η and the proof π. If they are both valid, then the validity of σ is guaranteed.

In Trace, each W_j computes a hash tag ht'_j and compares it with the ones in $\{ht_j\}_{W_j \in \mathcal{S}}$. If one tag is found to be the same, W_j continues to verify the validity of σ to decide whether this signature is worthy of tracing. If so, W_j share-decrypts (C_0, C_1) and sends an encrypted decryption share $\widehat{\delta}_j$ to T. After decrypting $\{\widehat{\delta}_j\} \in \mathcal{S}$, T verifies whether (C_0, C_1) is a valid ciphertext w.r.t. EK and t'. If so, T combines all decryption shares to obtain σ_m and traces $\mathcal{S}$.

$\underline{\mathsf{Setup}(1^\lambda, n, w, t, t') \to (PK, \{sk_i^s\}_{i=1}^n, \{wsk_j, ssk_j, sk_j^{\mathrm{wit}}\}_{j=1}^w, ck, tk, SK)}$

1. $(\{pk_i^s, sk_i^s\}_{i=1}^n) \leftarrow \mathsf{ATS.KeyGen}(1^\lambda, n, t)$, set $pk = (t, \{pk_i^s\}_{i=1}^n)$
2. $r_{pk} \leftarrow \mathcal{R}_\lambda$, $\mathsf{com}_{pk} \leftarrow \mathsf{COM.Comm}(\{pk_i^s\}_{i=1}^n, r_{pk})$
3. $(SK, EK, CK) \leftarrow \mathsf{DTPKE.Setup}(1^\lambda)$
4. $(pk_c^{\mathsf{enc}}, sk_c^{\mathsf{enc}}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$, $(pk_t^{\mathsf{enc}}, sk_t^{\mathsf{enc}}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$
5. $(pk_c^{\mathsf{sig}}, sk_c^{\mathsf{sig}}) \leftarrow \mathsf{SIG.KeyGen}(1^\lambda)$, $(pk_j^{\mathrm{wit}}, sk_j^{\mathrm{wit}}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$, $j \in [w]$
6. $(wpk_j, wsk_j) \leftarrow \mathsf{DTPKE.Join}(SK, id_j)$, $id_j \in [w]$
7. $ck \leftarrow (\{pk_i^s\}_{i=1}^n, sk_c^{\mathsf{enc}}, sk_c^{\mathsf{sig}}, t', \mathsf{com}_{pk}, r_{pk})$, $tk \leftarrow (\{pk_i^s\}_{i=1}^n, sk_t^{\mathsf{enc}}, pk_c^{\mathsf{sig}})$
8. $ssk_j \leftarrow \mathsf{Gen}(\lambda)$, for $W_j \in \mathcal{W}$
9. $PK \leftarrow (\mathsf{com}_{pk}, pk_c^{\mathsf{sig}}, pk_c^{\mathsf{enc}}, pk_t^{\mathsf{enc}}, \{wpk_j, pk_j^{\mathrm{wit}}\}_{j=1}^w)$
10. Output $(PK, \{sk_i^s\}_{i=1}^n, \{wsk_j, ssk_j, sk_j^{\mathrm{wit}}\}_{i=1}^w, ck, tk, SK)$

$\underline{\mathsf{Sign}(m, \mathcal{S}, \{sk_i^s\}_{S_i \in \mathcal{S}}, \mathcal{W}) \to (\widehat{\sigma_i}, \widehat{\mathcal{W}_i}) :}$

1. $\sigma_i \leftarrow \mathsf{ATS.Sign}(sk_i^s, m)$ for $S_i \in \mathcal{S}$
2. $\widehat{\sigma}_i \leftarrow \mathsf{PKE.Enc}(pk_c^{\mathsf{enc}}, m||\sigma_i)$, $\widehat{\mathcal{W}_i} \leftarrow \mathsf{PKE.Enc}(pk_c^{\mathsf{enc}}, \mathcal{W})$ for $S_i \in \mathcal{S}$

$\underline{\mathsf{Combine}(m, t, t', \mathcal{S}, PK, \{\widehat{\sigma}_i\}_{S_i \in \mathcal{S}}, \{\widehat{\mathcal{W}_i}\}_{S_i \in \mathcal{S}}, ck, EK) \to \sigma}$

1. $\mathcal{W} \leftarrow \mathcal{W}_i \leftarrow \mathsf{PKE.Dec}(\widehat{\mathcal{W}_i}, sk_c^{\mathsf{enc}})$ for $S_i \in \mathcal{S}$
2. $(m||\sigma_i) \leftarrow \mathsf{PKE.Dec}(\widehat{\sigma}_i, sk_c^{\mathsf{enc}})$ for $S_i \in \mathcal{S}$
3. $\sigma_m \leftarrow \mathsf{ATS.Combine}(\{pk_i^s\}_{i=1}^n, m, \mathcal{S}, \{\sigma_i\}_{S_i \in \mathcal{S}})$
4. $(C_0, C_1, k, \mathsf{AES.Enc}(k, \sigma_m)) \leftarrow \mathsf{DTPKE.Enc}(EK, \{wpk_j\}_{W_j \in \mathcal{W}}, \sigma_m)$
5. $(C_{j0}, C_{j1}) = \mathsf{ElGamal.Enc}(pk_j^{\mathrm{wit}}, C_0)$, $(C_{j2}, C_{j3}) = \mathsf{ElGamal.Enc}(pk_j^{\mathrm{wit}}, C_1)$, $ht_j = \mathsf{H}(ssk_j, m||W_j))$ for $W_j \in \mathcal{W}$
6. Generate a proof by using Prove for the relation:

$$\mathcal{R}((\mathsf{com}_{pk}, m, C_0, C_1); (pk, r_{pk}, \sigma_m, \{r_j\}_{j=1}^{t'})) = 1 \text{ iff}$$

$$\left\{ \begin{array}{l} \mathsf{COM.Verify}(pk, r_{pk}, \mathsf{com}_{pk}) = 1,\ \mathsf{ATS.Verify}(pk, m, \sigma_m) = 1 \\ \quad \mathsf{ElGamal.Enc}(pk_j^{\mathrm{wit}}, C_0) = (C_{j0}, C_{j1}),\ \text{for } W_j \in \mathcal{W} \\ \quad \mathsf{ElGamal.Enc}(pk_j^{\mathrm{wit}}, C_1) = (C_{j2}, C_{j3}),\ \text{for } W_j \in \mathcal{W} \end{array} \right\}$$

7. $\widehat{\sigma}_m = (\{C_{j0}, C_{j1}, C_{j2}, C_{j3}, ht_j\}, \mathsf{AES.Enc}(k, \sigma_m))$ for $W_j \in \mathcal{W}$
8. $\eta \leftarrow \mathsf{SIG.Sign}(sk_c^{\mathsf{sig}}, (m, \widehat{\sigma}_m, \pi))$
9. Output $\sigma \leftarrow (\widehat{\sigma}_m, \pi, \eta)$

$\underline{\mathsf{Verify}(PK, m, \sigma = (\widehat{\sigma}_m, \pi, \eta)) \to \{0, 1\}}$

1. Accept σ if $\mathsf{SIG.Verify}(pk_c^{\mathsf{sig}}, m, \widehat{\sigma}_m, \pi, \eta) = 1$ and $\mathsf{Verify}(\pi) = 1$; reject otherwise.

$\underline{\mathsf{Trace}(m, PK, tk = (\{pk_i^s\}_{i=1}^n, sk_t^{\mathsf{enc}}, pk_c^{\mathsf{sig}}), \sigma = (\widehat{\sigma}_m, \pi, \eta)) \to \mathcal{S}}$

1. If $\mathsf{Verify}(PK, m, \sigma) \neq 1$, output fail and return.
2. $ht_j' = \mathsf{H}(ssk_j, m||W_j)$ by $W_j \in \mathcal{W}$ and compare with $\{ht_j\}_{W_j \in \mathcal{W}}$
3. $(C_0, C_1) = (\mathsf{ElGamal.Dec}(sk_j^{\mathrm{wit}}, C_{j0}, C_{j1}), \mathsf{ElGamal.Dec}(sk_j^{\mathrm{wit}}, C_{j2}, C_{j3}))$ for each j
4. $\delta_j \leftarrow \mathsf{DTPKE.ShareDecrypt}(W_j, wsk_j, C_0, C_1)$ for $W_j \in \mathcal{W}$
5. $\widehat{\delta}_j \leftarrow \mathsf{PKE.Enc}(pk_t^{\mathsf{enc}}, \delta_j)$ for $W_j \in \mathcal{W}$

- -

6. $\delta_j \leftarrow \mathsf{PKE.Dec}(sk_t^{\mathsf{enc}}, \widehat{\delta}_j)$ for $W_j \in \mathcal{W}$
7. $\{0, 1\} \leftarrow \mathsf{DTPKE.ValidateCT}(EK, t', C_0, C_1)$
8. $\sigma_m \leftarrow \mathsf{DTPKE.Combine}(CK, t', C_0, C_1, C_2, C_3, \mathcal{W}, \{\sigma_m^j\}_{W_j \in \mathcal{W}})$
9. $\mathcal{S} \leftarrow \mathsf{ATS.Trace}(\{pk_i^s\}_{i=1}^n, m, \sigma_m)$.

Fig. 7. The HiTAPS scheme

6 The Optimized Scheme HiTAPS2

For each encrypted ATS signature $\widehat{\sigma}_m$, C has to encrypt it $2t'$ times via ElGamal encryption, resulting in much time on encryption and produces $4t'$ ciphertexts. Intuitively, such a process is optimizable and we improve it by leveraging the homomorphic feature of ElGamal encryption. Specifically, C encrypts $\widehat{\sigma}_m$ with the public keys of all t' witnesses and $2t'$ random numbers:

$$\begin{aligned}\mathsf{ElGamal.Enc}(pk_j^{\mathrm{wit}}, C_0||C_1) =&(C_0 g^{pk_1^{\mathrm{wit}} r_{11}} \cdots g^{pk_{t'}^{\mathrm{wit}} r_{1t'}}, C_1 g^{pk_1^{\mathrm{wit}} r_{21}} \cdots g^{pk_{t'}^{\mathrm{wit}} r_{2t'}},\\ & g^{r_{11}}, \cdots, g^{r_{1t'}}, g^{r_{21}}, \cdots, g^{r_{2t'}}).\end{aligned}$$

In this way, the number of ciphertexts is reduced to $2t' + 2$. For tracing, each witness W_j is awakened by the hash tag, but partially decrypts the first two ciphertexts in the above equation by sending $((g^{r_{1j}})^{-pk_j^{\mathrm{wit}}}, (g^{r_{2j}})^{-pk_j^{\mathrm{wit}}}) = (g^{-r_j pk_j^{\mathrm{wit}}}, g^{-r_{2j} pk_j^{\mathrm{wit}}})$ to T. Next, T recovers (C_0, C_1) by removing $g^{sk_1^{\mathrm{wit}} r_{11}} \cdots g^{pk_{t'}^{\mathrm{wit}} r_{1t'}}$ and $g^{pk_1^{\mathrm{wit}} r_{21}} \cdots g^{pk_w^{\mathrm{wit}} r_{2t'}}$ from $C_0 g^{pk_1^{\mathrm{wit}} r_{11}} \cdots g^{pk_{t'}^{\mathrm{wit}} r_{1t'}}$ and $C_1 g^{pk_1^{\mathrm{wit}} r_{21}} \cdots g^{pk_w^{\mathrm{wit}} r_{2t'}}$ step by step, respectively.

7 Security and Privacy Analysis

Theorem 1. The HiTAPS scheme Π in Fig. 7 is unforgeable, accountable, and private, assuming that the underlying ATS is secure, the DTPKE is IND-NAA-NAC-CPA secure, the PKE is semantically secure, the hash function is preimage resistant, the $(\mathsf{Prove}, \mathsf{Verify})$ is an argument of knowledge and Honest Verifier Zero Knowledge (HVZK), the COM is hiding and binding, and the SIG is strongly unforgeable.

The proof of Theorem 1 is captured in the following three lemmas.

Lemma 1. The HiTAPS scheme Π is unforgeable and accountable if the ATS is secure, the $(\mathsf{Prove}, \mathsf{Verify})$ is an argument of knowledge, and COM is blinding, i.e., for all PPT adversaries $\mathcal{A}$, $\mathcal{A}_1$, and $\mathcal{A}_2$, such that

$$\mathbf{Adv}^{\mathrm{forg}}_{\mathcal{A},\Pi}(\lambda) \leq \left(\mathbf{Adv}^{\mathrm{forg}}_{\mathcal{A}_1,\mathsf{ATS}}(\lambda) + \mathbf{Adv}^{\mathrm{bind}}_{\mathcal{A}_2,\mathsf{COM}}(\lambda)\right) \cdot \alpha(\lambda) + \beta(\lambda), \tag{1}$$

where α and β are the knowledge error and tightness of the proof system.

Proof. Proof. We prove Lemma 1 by defining three experiments.

Ept 0. It is the unforgeability and accountability experiment in Fig. 4 applied to Π. If Evt_0 is the event that $\mathcal{A}$ wins in Ept 0, then

$$\mathbf{Adv}^{\mathrm{forg}}_{\mathcal{A},\Pi}(\lambda) = \Pr[Evt_0]. \tag{2}$$

Ept 1. Let $PK = (\mathsf{com}_{pk}, pk_c^{\mathsf{sig}}, pk_c^{\mathsf{enc}}, pk_t^{\mathsf{enc}}, \{wpk_j, pk_j^{\mathrm{wit}}\}_{j=1}^{w})$ be the public key and let $tk \leftarrow (\{pk_i^s\}_{i=1}^{n}, sk_t^{\mathsf{enc}}, pk_c^{\mathsf{sig}})$ be he tracing key that are given to $\mathcal{A}$.

Ept 1 is identical to Ept 0 except that we strengthen the winning condition over Exp 0: to win Exp 1, $\mathcal{A}$ has to produce a valid forgery (m', σ''_m) where $\sigma'_m = (\widehat{\sigma}'_m, \{ht'_j\}, \pi', \eta')$, along with a witness $(pk'', r''_{pk}, \sigma''_m, \{r''_j\}_{j=1}^{t'})$ such that

$$\mathcal{R}((\mathsf{com}_{pk}, m', C'_1, C'_2); (pk'', r''_{pk}, \sigma''_m, \{r''_j\}_{j=1}^{t'})) = 1,$$

We construct an adversary $\mathcal{A}'$ from $\mathcal{A}$ in Exp 0. It invokes $\mathcal{A}$ and answers to all $\mathcal{A}$'s queries until it receives from $\mathcal{A}$ the $(m', \widehat{\sigma}'_m)$ to provide a statement $(\mathsf{com}_{pk}, m', C'_1, C'_2)$. $\mathcal{A}'$ executes an extractor Ext for (P, V) on $\mathcal{A}$'s remaining execution. Ext produces a witness $wt = (pk'', r''_{pk}, \sigma''_m, \{r''_j\}_{j=1}^{t'})$. $\mathcal{A}'$ uses wt and sk_c^{Sig} to generate π' and η' such that $\sigma'_m = (\widehat{\sigma}'_m, \{ht'_j\}, \pi', \eta')$ is a valid signature on m'. $\mathcal{A}'$ outputs (m', σ'_m) and wt. If Evt_1 stands for $\mathcal{A}'$ wins Exp 1, then

$$\Pr[Evt_1] \geq (\Pr[Evt_0] - \alpha(\lambda))/\beta(\lambda). \tag{3}$$

Ept 2. We strengthen the winning condition by requiring $pk'' = (t, \{pk_i^s\}_{i=1}^n)$. Given the binding property of COM, we have $\mathsf{COM.Verify}(pk, r_{pk}, \mathsf{com}_{pk}) = \mathsf{COM.Verify}(pk'', r''_{pk}, \mathsf{com}_{pk}) = 1$. If $pk'' \neq pk$, we find an attack on the binding property of COM. Specifically, let Evt_2 be the event that $\mathcal{A}'$ wins Ept 2 and E be the event that $pk \neq pk''$, then $\Pr[Evt_2] = \Pr[Evt_1 \wedge \neg E] \geq \Pr[Evt_1] - \Pr[E]$. We assume that there is an adversary $\mathcal{A}_2$ such that $\Pr[E] = \mathbf{Adv}^{\mathrm{forg}}_{\mathcal{A}_2,\mathsf{COM}}(\lambda)$

We construct an adversary $\mathcal{A}_1$ that invokes $\mathcal{A}$ and answers to $\mathcal{A}$'s queries. When $\mathcal{A}$ outputs a forgery (m', σ'_m) and a witness $(pk'', r''_{pk}, \sigma''_m, \{r''_j\}_{j=1}^{t'})$ that meet the winning condition of Exp 1 and Exp 2, $\mathcal{A}_1$ outputs (m', σ''^m). By $\mathcal{R}$, we have σ''_m is a valid signature on m' with respect to pk''. By Exp 2, we have $pk = pk''$. Therefore, if $\mathcal{A}$ wins Exp 2, then (m', σ''_m) is a valid forgery for the ATS scheme. Since the ATS is secure, we have

$$\mathbf{Adv}^{\mathrm{forg}}_{\mathcal{A}_1,\mathsf{ATS}}(\lambda) \geq \Pr[Evt_2]. \tag{4}$$

Conclusively, combining (2), (3), (4), and (5) proves (1). □

Lemma 2. The HiTAPS scheme Π is private against the public if the PKE is semantically secure, the SIG is strongly unforgeable the $(\mathsf{Prove}, \mathsf{Veriy})$ is an argument of knowledge and HVZK, the COM is hiding, the $\mathcal{H}$ is preimage resistant, and the DTPKE is IND-NAA-NAC-CPA secure, i.e., for all PPT adversaries $\mathcal{A}$, there exists adversaries $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$, $\mathcal{A}_4$, $\mathcal{A}_5$, and $\mathcal{A}_6$ such that

$$\begin{aligned}\mathbf{Adv}^{\mathrm{priP}}_{\mathcal{A},\Pi}(\lambda) \leq & 2\Big(3\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{A}_1,\mathsf{PKE}}(\lambda) + \mathbf{Adv}^{\text{euf-cma}}_{\mathcal{A}_2,\mathsf{SIG}}(\lambda) + Q \cdot \mathbf{Adv}^{\mathrm{hvzk}}_{\mathcal{A}_3,(\mathsf{P},\mathsf{V})}(\lambda) \\ & + \epsilon_{\mathcal{A}_4}(\lambda) + \mathbf{Adv}^{\text{hash-pre}}_{\mathcal{A}_5,\mathsf{H}}(\lambda) + \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{A}_6,\mathsf{DTPKE}}(\lambda)\Big)\end{aligned} \tag{5}$$

where $\epsilon(\lambda)_{\mathcal{A}_4}$ is hiding statistical distance of COM and Q is query number.

Proof. We prove Lemma 2 by defining seven experiments.

Exp 0. It is the experiment of privacy against the public $\mathbf{Exp}^{\text{priP}}$ defined in Fig. 2 applied to Π. If Evt_0 stands for $\mathcal{A}$ wins Exp 0, then

$$\mathbf{Adv}^{\text{priP}}_{\mathcal{A},\Pi}(\lambda) = |2\Pr[Evt_0] - 1|. \tag{6}$$

Exp 1. It is identical to Exp 0 except that the signing oracle $\mathcal{O}_1(\mathcal{S}_0, \mathcal{S}_1, \mathcal{N}_0, \mathcal{N}_1, m)$ is modified such that step 2 of Sign in Fig. 5 now returns $\widehat{\sigma}_i \leftarrow \mathsf{PKE.Enc}(pk^e_j, 0)$, where 0 is encrypted instead of $(m||\sigma_i)$. Since PKE is semantically secure, $\mathcal{A}_1$'s $\mathbf{Adv}$ in Exp 1 is indistinguishable from its $\mathbf{Adv}$ in Exp 0, i.e., say Evt_1 stands for $\mathcal{A}_1$ wins Exp 1,

$$|\Pr[Evt_1] - \Pr[Evt_0]| \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{A}_1,\mathsf{PKE}}(\lambda). \tag{7}$$

Exp 2. It is identical to Exp 0 except that responses to $\mathcal{O}_2(m, \sigma, \hat{\sigma})$ are fail. If SIG is strongly unforgeable, $\mathcal{A}_1$'s $\mathbf{Adv}$ in Exp 2 is indistinguishable from its $\mathbf{Adv}$ in Exp 1, i.e., say Evt_2 stands for $\mathcal{A}_2$ wins Exp 2,

$$|\Pr[Evt_2] - \Pr[Evt_1]| \leq \mathbf{Adv}^{\text{euf-cma}}_{\mathcal{A}_2,\mathsf{SIG}}(\lambda). \tag{8}$$

Exp 3. It is identical to Exp 2 except that $\mathcal{O}_1(m, \mathcal{S}_0, \mathcal{S}_1, \mathcal{N}_0, \mathcal{N}_1)$ is modified such that step 6 of $\mathsf{Combine}$ now generates a proof π by using the simulator, which is given $(\mathsf{com}_{pk}, m, C_1, C_2)$ as input. Since the simulated proofs are computationally indistinguishable from real proofs, $\mathcal{A}_3$'s $\mathbf{Adv}$ in Exp 3 is indistinguishable from its $\mathbf{Adv}$ in Exp 2, i.e., say Evt_3 stands for $\mathcal{A}_2$ wins Exp 3,

$$|\Pr[Evt_3] - \Pr[Evt_2]| \leq Q \cdot \mathbf{Adv}^{\text{hvzk}}_{\mathcal{A}_3,(\mathsf{P},\mathsf{V})}(\lambda). \tag{9}$$

Exp 4. It is identical to Exp 3 except that step 2 of Setup in Fig. 5 is modified such that $r_{pk} \leftarrow \mathcal{R}_\lambda$, $\mathsf{com}_{pk} \leftarrow \mathsf{COM.Comm}(0, r_{pk})$, where 0 is committed instead of pk. Since COM is hiding, the adversary's $\mathbf{Adv}$ in Exp 4 is indistinguishable from its $\mathbf{Adv}$ in Exp 3, i.e., say Evt_3 stands for $\mathcal{A}_4$ wins Exp 4,

$$|\Pr[Evt_4] - \Pr[Evt_3]| \leq \epsilon_{\mathcal{A}_4}(\lambda). \tag{10}$$

Exp 5. This step resembles Exp 1, but the first part step 5 of $\mathsf{Combine}$ now returns $\mathsf{ElGamal.Enc}(pk^{\text{wit}}_j, 0)$, $\mathsf{ElGamal.Enc}(pk^{\text{wit}}_j, 0)$, and

$$|\Pr[Evt_5] - \Pr[Evt_4]| \leq 2\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{A}_5,\mathsf{PKE}}(\lambda). \tag{11}$$

Exp 6. It is identical to Exp 5 except that the signing oracle $\mathcal{O}_1(m, \mathcal{S}_0, \mathcal{S}_1, \mathcal{N}_0, \mathcal{N}_1)$ is modified such that the second part of step 5 of $\mathsf{Combine}$ now returns $\mathcal{H}(0)$. Since the $\mathcal{H}$ is preimage resistant, $\mathcal{A}_6$'s $\mathbf{Adv}$ in Exp 6 is indistinguishable from its $\mathbf{Adv}$ in Exp 5, i.e., say Evt_6 stands for $\mathcal{A}_6$ wins Exp 6,

$$|\Pr[Evt_6] - \Pr[Evt_5]| \leq \mathbf{Adv}^{\text{hash-pre}}_{\mathcal{A}_6,\mathsf{H}}(\lambda). \tag{12}$$

Exp 7. It is identical to Exp 6 except that $\mathcal{O}_1(m, \mathcal{S}_0, \mathcal{S}_1, \mathcal{N}_0, \mathcal{N}_1)$ is modified such that step 4 of $\mathsf{Combine}$ now returns $\hat{\sigma} \leftarrow \mathsf{DTPKE.Enc}(EK, 0, \sigma_m)$. Since

DTPKE is secure, $\mathcal{A}_7$'s **Adv** in Exp 7 is indistinguishable from its **Adv** in Exp 6, i.e., say Evt_7 stands for $\mathcal{A}_7$ wins Exp 7,

$$|\Pr[Evt_7] - \Pr[Evt_6]| \leq \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{A}_7,\mathsf{DTPKE}}(\lambda). \tag{13}$$

In Exp 7, $\mathcal{A}_7$'s view is independent of b, i.e.,

$$\Pr[Evt_7] = 1/2. \tag{14}$$

Lastly, combining (6)-(14) proves (5). This completes the proof of lemma 2. □

Lemma 3. The HiTAPS scheme Π is private against the signer.

The proof of Lemma 3 is almost identical to the proof of Lemma 2 and is omitted. Combining Lemma 1, Lemma 2, and Lemma 3, we prove Theorem 1.

Theorem 2. The HiTAPS2 scheme is unforgeable, accountable, and private.

The proof is Theorem 2 is almost identical to the proof of Theorem 1 except that the step 5 and step 6 of Combine are altered, leading to changes in corresponding experiments.

8 Performance Analysis

8.1 Experiment Settings

Dataset and Parameters. Considering the validity and observability of the results, we comprehensively provide the parameters. We change the number of signers n and the maximum number of witnesses w from 20 to 100, the length of the signer message m from 50 KB to 400 KB, the threshold t from 5 to 15, and the number of participating witnesses t' from 5 to 40, and finally change t and t' from 3 to 5.
Setup. We implemented HiTAPS using Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz on a Linux server running Ubuntu 18.04. We use HMAC-SHA256 as the pseudo-random function to implement the hash function and AES as the symmetric encryption.

8.2 Computational Cost

HiTAPS consists of (Setup, Sign, Combine, Verify, Trace). We measured the time required for each phase under the given parameters.

In Setup, HiTAPS generates all keys. As shown in Fig. 8(a), Setup takes 314 ms when $n = 100$ and $w = 100$. In Sign, signers calculate signature shares, and in Fig. 8(b) we can see that it takes 154 ms to sign a message with a length of 400 KB. In Combine, the combiner combines signatures from t signature shares and constructs zero-knowledge proofs. By observing Fig. 8(c), we can find that as t increases, the time required for Combine also increases. When $t' = 5$ and $t = 5$, 100 groups of signers takes 22.3 s, while HiTAPS2 only takes 13.6 s. In

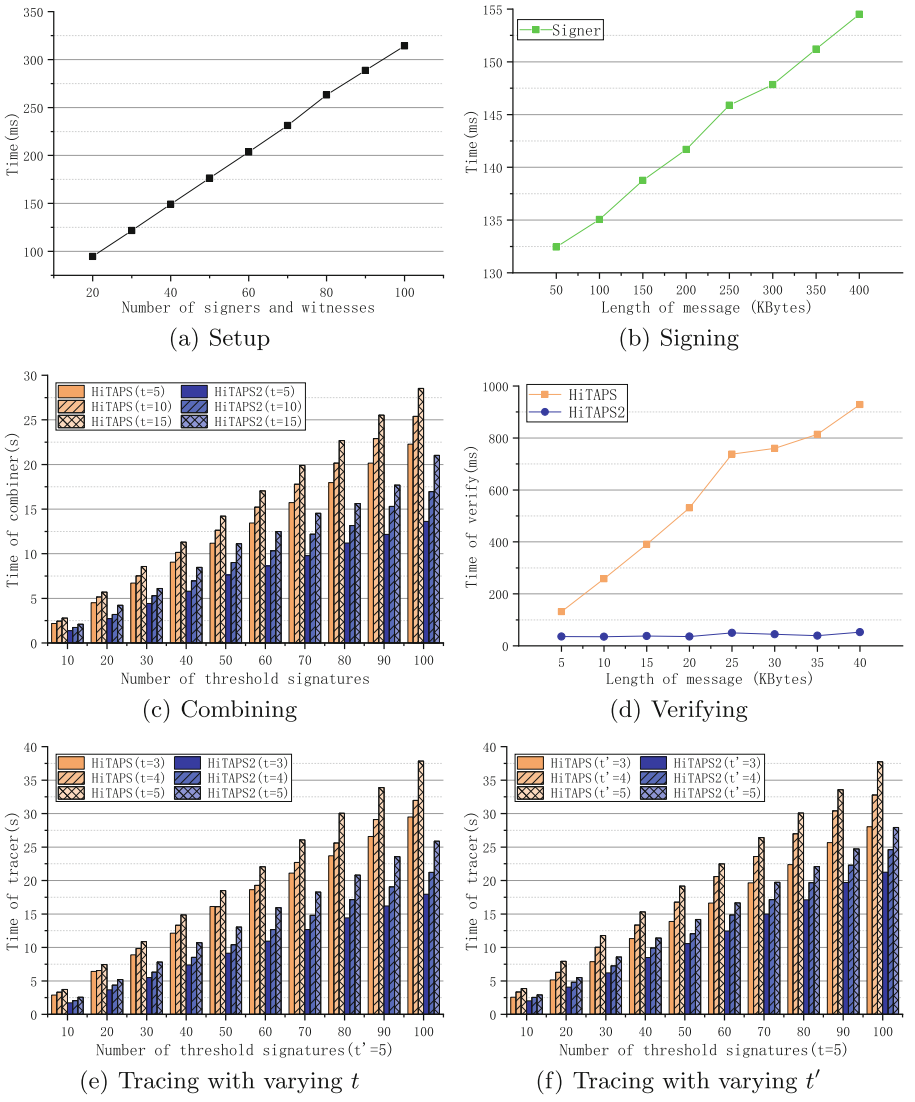

Fig. 8. Computational Costs.

Verify, the verifier verifies the threshold signature and the zero-knowledge proof given by combiner. As shown in Fig. 8(d), as t' increases, the time required for Verify changes from 131 ms to 928 ms, while the verification time of HiTAPS2 stabilizes at 41 ms. In Trace, the tracer traces a signing group. In Fig. 8(e) and Fig. 8(f), we can see that tracking 100 groups of signers at $t = 5$ and $t' = 5$ takes 37.8 s (HiTAPS) and 25.9 s (HiTAPS2), respectively.

8.3 Communication Overhead

We analyze communication overhead by calculating the length of messages transmitted by each party in a signing group.

In Sign, the signer sends plaintext m, signature $\widehat{\sigma}_i$, and encrypted witness sequence $\widehat{\mathcal{W}}_i$. In Combine, the combiner outputs a message m, an encrypted threshold signature σ, and the threshold signature contains $(\widehat{\sigma}_m, \{ht_j\}, \pi, \eta)$. In Verify, the verifier outputs 1 bit. In Trace, the witness gives his partial decryption key $\widehat{\delta}_j$, and the tracer combines the keys to trace and gives the signer sequence S. We record the communication overhead in Table 2.

Table 2. Communication Overhead

Phase	Signing	Combining	Verifying	Tracing	
Party	Signer	Combiner	Verifier	Witness	Tracer
Theory	$\lvert m\rvert, \lvert\widehat{\sigma}_i\rvert, \lvert\widehat{\mathcal{W}}_i\rvert$	$\lvert m\rvert, \lvert\sigma\rvert$	$\lvert b\rvert$	$\lvert\widehat{\delta}_j\rvert$	$\lvert\mathcal{S}\rvert$
Practice	1.969 KB	1.165 KB	1 bit	0.461 KB	0.445 KB

9 Conclusions

In this work, we have proposed HiTAPS, a new threshold signature scheme that achieves unforgeability, accountability, and privacy (against the public and signers). HiTAPS builds on TAPS and moves forward by facilitating witnessed tracing and securely designating witnesses. We formally prove the security and privacy of HiTAPS. Experimental results obtained from running a prototype show that HiTAPS is practical and efficient, HiTAPS2 is even better in Combine and Verify, e.g., HiTAPS2 stabilizes Verify at around 41 ms.

Acknowledgment. This work is supported by National Natural Science Foundation of China (NSFC) under the grant No. 62372149, No. U23A20303, and National Natural Science Foundation of China (NSFC) under the grant No. 62172040. It is partially supported by EU LOCARD Project under Grant H2020-SU-SEC-2018-832735.

References

1. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Proceedings of 6th Annual International Cryptology Conference (CRYPTO), pp. 307–315. Santa Barbara, USA (1989)
2. Shoup, V.: Practical threshold signatures. In: International Conference on the Theory and Application of Cryptographic Techniques(EUROCRYPT), pp. 14–18. Bruges, Belgium (2000)
3. Fouque, P., Stern, J.: Fully distributed threshold RSA under standard assumptions. International Conference on the Theory and Application of Cryptology and Information Security(ASIACRYPT), pp. 310–330. Gold Coast, Australia (2001)

4. Damgård, I., Koprowski, M.: Practical threshold RSA signatures without a trusted dealer. In: International Conference on the Theory and Applications of Cryptographic Techniques(EUROCRYPT), pp. 152–165. Innsbruck, Austria (2001)
5. Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multi signatures: extended abstract. In: Proceedings of 8th ACM Conference on Computer and Communications Security (CCS), pp. 245–254. Philadelphia, USA (2001)
6. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Proceedings of 7th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), pp. 514–532. Gold Coast, Australia (2001)
7. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: Proceedings of 13th ACM Conference on Computer and Communications Security (CCS), pp. 390–399. Alexandria, USA (2006)
8. Nick, J., Ruffing, T., Seurin, Y.: MuSig2: simple two-round Schnorr multi-signatures. In: Proceedings of 41st Annual International Cryptology Conference (CRYPTO), pp. 189–221, Virtual (2021)
9. Tang, G.: On tightly-secure (linkable) ring signatures. In: Proceedings of 23rd International Conference on Information and Communications Security (ICICS), pp. 375–393. Chongqing, China (2021)
10. Ji, Y., Tao, Y., Rui, Z.: More efficient construction of anonymous signatures. In: Proceedings of 23rd International Conference on Information and Communications Security (ICICS), pp. 394–411. Chongqing, China (2021)
11. Goyal, V., Song, Y., Srinivasan, A.: Traceable secret sharing and applications. In: Proceedings of 41st Annual International Cryptology Conference (CRYPTO), pp. 718–747, Virtual (2021)
12. Boneh, D., Komlo, C.: Threshold signatures with private accountability. In: Proceedings of 42nd Annual International Cryptology Conference (CRYPTO), pp. 551–581. Santa Barbara, USA (2022)
13. Delerablée, C., Pointcheval, D.: Dynamic threshold public-key encryption. In: Proceedings of 28th Annual International Cryptology Conference (CRYPTO), pp. 317–334. Santa Barbara, USA (2008)
14. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attack. SIAM J. Comput. **17**(2), 281–308 (1988)
15. Loh, J.-C., Guo, F., Susilo, W., Yang, G.: A tightly secure ID-based signature scheme under DL assumption in AGM. In: Proceedings of 28th Australasian Conference on Information Security and Privacy (ACISP), pp. 199–219. Brisbane, Australia (2023)
16. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Robust threshold DSS signatures. Inf. Comput. **164**(1), 54–84 (2001)
17. Katz, J., Lindell, Y.: Introduction to Modern Cryptography (Third edition), pp. 1–598. CRC Press (2021)
18. Boneh, D., Boyen, X.: Hierarchical identity based encryption with constant size ciphertext. Proceedings of 24th International Conference on the Theory and Application of Cryptographic Technique (EUROCRYPT), pp. 440–456 (2005)
19. Delerablé, C., Paillier, P., Pointcheval, D.: Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In: Proceedings of first International Conference on Pairing-Based Cryptography (Pairing), pp. 39–59 (2007)