Generic construction of threshold ring signatures and lattice-based instantiations

Lin, Hao; Wang, Mingqiang; Wen, Weiqiang; Sun, Shi Feng; Liang, Kaitai

DOI
10.1007/s10623-025-01660-6

Publication date
2025

Document Version
Final published version

Published in
Designs, Codes, and Cryptography

Designs, Codes and Cryptography
https://doi.org/10.1007/s10623-025-01660-6

Generic construction of threshold ring signatures and lattice-based instantiations

Hao Lin[1] · Mingqiang Wang[2] · Weiqiang Wen[3] · Shi-Feng Sun[4] · Kaitai Liang[5]

Received: 16 October 2024 / Revised: 20 April 2025 / Accepted: 28 May 2025

Abstract

A t-out-of-n threshold ring signature allows t parties to jointly sign a message on behalf of n parties without revealing the identities of the signers. In this paper, we introduce a new generic construction for threshold ring signature, called GC-TRS, which can be built on top of a selection on identification schemes, commitment schemes, and a new primitive called t-out-of-n proof protocol which is a special type of zero-knowledge proof. In general, our design enables a group of t signers to first generate an aggregated signature by interacting with each other; then they are able to compute a t-out-of-n proof to convince the verifier that the aggregated signature is indeed produced by t individuals among a particular set. The signature is succinct, as it contains only one aggregated signature and one proof in the final signature. We define all the properties required for the building blocks to capture the security of the GC-TRS and provide a detailed security proof. Furthermore, we propose two lattice-based instantiations for the GC-TRS, named LTRS and CTRS, respectively. Notably, the CTRS scheme is the first scheme that has a logarithmic signature size relative to the ring size. Additionally, during the instantiation process, we construct two t-out-of-n proof protocols, which may be of independent interest.

Communicated by C. Weinert.

✉ Hao Lin
haolinz6@qq.com

Mingqiang Wang
wangmingqiang@sdu.edu.cn

Weiqiang Wen
weiqiang.wen@telecom-paris.fr

Shi-Feng Sun
shifeng.sun@sjtu.edu.cn

Kaitai Liang
Kaitai.Liang@tudelft.nl

1 Department of Information Security, Naval University of Engineering, Wuhan 430033, China

2 School of Mathematics, Shandong University, Jinan 250100, Shandong, China

3 LTCI, Telecom Paris, Institut Polytechnique de Paris, Paris, France

4 School of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, China

5 Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft 2628 XE, Netherlands

Published online: 13 June 2025

Keywords Threshold ring signature · t-out-of-n proof · Zero-knowledge proof · Lattice-based cryptography

Mathematics Subject Classification 94A60

1 Introduction

Ring signatures (RS) introduced by Rivest et al. [47] enable a member of a set (ring) to anonymously sign messages on behalf of the ring. Verifiers can check if a signature is from one of the ring members without knowing who is the actual signer. Bresson et al. [16] generalized RS to a threshold variant. In this setting, t-out-of-n parties can interact together to produce a signature without giving any information about the set of signers. Compared with RS, the threshold variant can tolerate user corruption. This means that even if an adversary compromises some (fewer than t) secret keys of corrupted users, they still cannot generate a valid signature. Threshold ring signatures (TRS) can be applied to many scenarios where a statement must be endorsed by a quorum, while parties prefer to protect their identities. For example, one may use the technique in edge computing to protect the privacy of data owners [50] and in blockchain applications to provide strong supervision [33].

Following the concept of TRS, several schemes have been proposed in the literature based on traditional hard problems. Bresson et al. [16] and Liu et al. [34] constructed the schemes based on the RSA assumption; Yuen et al. [52], Okamoto et al. [44] and Aranha et al. [3] turned to the decisional Diffie–Hellman (DDH) assumption. There are a few potentially quantum-safe schemes that may not be practical and efficient. The state-of-the-art lattice-based TRS by Bettaieb et al. [13] (improved from Cayrel et al. [17]) takes a signature size with 18.1 MB, when #signers is 50 and the ring size is 1024. Some code-based schemes were proposed, such as [18, 40], but there is still a lack of efficiency, for example, Melchor et al. [40] proposed a TRS that requires 20 MB in a signature. Haque et al. [28, 29] recently designed two black-box constructions (which could be instantiated under post-quantum assumptions) but they did not provide concrete instantiations and parameters. All existing post-quantum secure TRS schemes (except [28]) require an additional leader in the signing process. The signers need to perform an additional negotiation to determine the leader. This centralized approach may cause an increase in the amount of communications.

We now review prior schemes [13, 17, 29, 40] that result in inefficient schemes. In [17, 29, 40], the final signature contains n different partial signatures, t of which are generated by the signers and the rest are by the leader. When the ring size n is relatively large, the size of the final signature inevitably increases. For example, in [40], the size of a partial signature is roughly 20 KB. Thus, if $n = 512$, the final signature size would be 10,240 KB. Bettaieb et al. [13] proposed a new construction to shorten the size of signature. The final signature only contains t different partial signatures and one proof, where the proof checks if these signatures are generated by t different users in the ring. Although this design greatly improves the efficiency as compare to the previous approach, the complexity is still restricted to $O(nt)$.

1.1 Our contribution

The main contribution of this paper is a new generic TRS construction (GC-TRS), which is obtained by cleverly combining selected identification schemes, commitment schemes and

t-out-of-n proof protocols. The t-out-of-n proof is a natural extension of 1-out-of-n proof [25, 37, 38] that proves the opening of a commitment is indeed the sum of t elements from a particular set. The GC-TRS is succinct, as it only contains one aggregated signature in the final signature, while prior schemes include at least t partial signatures.[1] Our approach also does not require a leader, i.e., non-centralized, and each signer has the same role. We also define all the required properties for the building blocks and present a detailed security proof for the GC-TRS.

We then develop two instantiations, called LTRS and CTRS, for the GC-TRS. These instantiations are based on the Module-LWE (MLWE) and Module-SIS (MSIS) problems. The LTRS has a signature size that is linear with ring size, while the CTRS has a logarithmic signature size at the expense of a linear factor in t. A brief comparison is shown in Table 1. In the table, we let n denote the size of the ring, t and λ denote the #signers and the security parameter, and N/A indicates non-applicable.

Recall that Haque et al. proposed two generic black-box constructions in [29] and [28] that may be possibly instantiated under various assumptions. Although logarithmic-sized signatures are generated in [28], as stated in their paper, their scheme is currently unable to be instantiated based on the standard lattice assumptions due to the lack of a secure NIWI scheme.

Except [28], the rest of schemes are in the random oracle model (ROM). The repetition generally only occurs in the lattice-based algorithms, so we do not consider this for other schemes. The scheme given in [13] should repeat the signature algorithm at least λ times to maintain its security, while ours only requires a constant number of repetitions, denoted as c, which is approximately 20. And the scheme given in [13] requires 6 interactions, while ours only requires 2 interactions. Thus, our instantiations are more efficient than [13].

Therefore, apart from the approach in [28], our method is the only one to achieve a signature size of $t \cdot \log(n)$. And since the method in [28] has other limitations which prevent it from being as favorable as our proposed solution in terms of overall performance.

To provide a comparison in terms of practical efficiency, we also provide parameters for our instantiations with 128-bit security. We present the comparisons on the concrete signature sizes and key sizes among our construction and [13, 40] in Table 2.

It is evident from Table 2 that our constructions exhibit significant improvements compared to the previous ones. Particularly, when n is considerably large, the CTRS scheme has a very small signature. Additionally, when n is not very large but t is slightly large, the LTRS scheme has the smallest signature. In this case, utilizing the LTRS scheme may be a favorable option.

1.2 Technical overview

To further reduce the size of the signature, our high-level idea is to compress the original t partial signatures into a single aggregate signature. Then, a succinct zero-knowledge proof is generated to prove that the aggregate signature was indeed created by t different users in the ring. In the following, we give an overview of our techniques. For simplicity, we will consider the case $t = 2$ here, and the general case will be described in Sect. 3. We first explain how to construct GC-TRS and then show how to instantiate the building blocks.

[1] In this paper, we mainly consider the (signature or proof) size for succinctness. For (threshold) ring signature, it is said to be succinct when the signature size is sublinear in the ring size. For ZK proof, the proof size is required to be sublinear in the witness size to achieve succinctness.

Table 1 Comparison of TRS schemes

Scheme	Hardness assumption	ROM	Without leader	Signature size	Repetition times	Interaction round
Bresson et al. [16]	RSA	✓	×	$n \log n$	N/A	t
Yuen et al. [52]	CDH	✓	×	n	N/A	2
Okamoto et al. [44]	Discrete Log	✓	×	n	N/A	4
Haque et al. [29]	N/A	✓	×	n	N/A	3
Haque et al. [28]	N/A	×	✓	$t \cdot \log n$	N/A	1
Melchor et al. [40]	Syndrome Decoding	✓	×	n	N/A	3
Bettaieb et al. [13]	SIS	✓	×	nt	λ	6
LTRS	MLWE MSIS	✓	✓	n	c	2
CTRS	MLWE MSIS	✓	✓	$t \cdot \log n$	c	2

Table 2 Concrete comparison of TRS scheme

Scheme	Signature size (MB)						Public key size (MB)
	$n = 2^{10}$ $t = 10$	$n = 2^{15}$ $t = 10$	$n = 2^{20}$ $t = 10$	$n = 2^{10}$ $t = 50$	$n = 2^{15}$ $t = 50$	$n = 2^{20}$ $t = 50$	
Melchor et al. [40]	20	640	20480	20	640	20480	24.5
Bettaieb et al. [13]	3.63	42.9	1263.5	18.11	214.5	6317.5	0.002
LTRS	1.75	41.3	1636.8	1.87	43.2	1701.9	0.0053
CTRS	1.37	1.73	2.07	5.53	7.47	9.56	0.0045

Table 3 Notations for the TRS

t	The number of actual signers
n	The total number of members in the ring
P_{i_1}, P_{i_2}	The actual signers
msg	The message to be signed
$R = (P_1, \ldots, P_n)$	Ring members in the threshold ring signature
pk_i, sk_i	The public key and secret key corresponding to the user P_i
$\mathsf{A}(\cdot)$	The commit function in the identification scheme
w	The output value of the commit function
$\mathcal{C}$	Challenge set of the identification scheme
c	The challenge
$\mathsf{Z}(\cdot)$	The response function in the identification scheme
z	The output value of the response function
$\mathsf{V}(\cdot)$	The verification function in the identification scheme
$H(\cdot)$	The hash function
$\mathsf{Com}(\cdot)$	The commitment scheme that outputs a message commitment
r	The randomness used in the commitment
$\vec{v}$	The indicator vectors corresponding to the actual signers
$\mathcal{R}_q$	The cyclotomic polynomial ring $\mathbb{Z}_q/(X^d+1)$
ϖ	Rank of decomposition of $\mathcal{R}_q : \mathcal{R}_q \cong \mathcal{R}_q^{(0)} \times \cdots \times \mathcal{R}_q^{(\varpi-1)}$
Encode	Transforms a ϖ-dimensional vector into an element in $\mathcal{R}_q$
$\mathsf{NTT}(\cdot)$	Number Theoretic Transform
$\mathsf{NTT}(\cdot)^{-1}$	The Inverse of the Number Theoretic Transform
$\otimes$	The tensor product of vectors

1.2.1 Compressing signature

Our starting point is the canonical identification scheme [1], which can be converted into a signature scheme using the Fiat–Shamir transformation [26, 36]. The canonical identification scheme is executed between a prover and a verifier. We use Dilithium signature as a running example, indicated inside [ ], with the prover holding a secret key $sk = s$ and the verifier having access to a public key $pk = B \cdot s$. The process is as follows: the prover first uses a commit function $\mathsf{A}(\cdot)$ to generate a commitment w [$\mathsf{A} : By \mapsto w$]. The resulting w is then sent to the verifier, who subsequently samples a challenge c from a special distribution [$c \leftarrow \mathcal{C}$]. Finally, the prover makes use of the response function $\mathsf{Z}(\cdot)$ to generate a response z [$\mathsf{Z} : y + cs \mapsto z$]. The verifier then can collect pk, z, c and use verification function $\mathsf{V}(\cdot)$ to compute w' [$\mathsf{V} : Bz - c \cdot pk \mapsto w'$], which is then compared with w. If $w' = w$, the transcript (w, c, z) is valid.[2]

In order to enable aggregate signatures, it is necessary for the canonical identification scheme to possess a specific homomorphic property:

$$\mathsf{V}(pk_1, c, z_1) + \mathsf{V}(pk_2, c, z_2) = \mathsf{V}(pk_1 + pk_2, c, z_1 + z_2).$$

[2] In fact, the real Dilithium signature process also encompasses a rejection sampling step and a process to verify the size of the vector z. However, to keep things simple, we'll set these aspects aside for now.

By this property, signers can compress signatures in the following way: Consider two signers P_{i_1} and P_{i_2}, who want to generate a threshold ring signature for a message msg and a ring $(P_1, \ldots, P_n)$. Each signer P_{i_j} first computes w_j by using the function $\mathsf{A}(\cdot)$. Next, they interact with each other to produce a challenge c based on the input message msg, ring R and the individually generated values of w_j (which can temporarily be imagined as $c = H(msg, R, w_1, w_2)$). Finally, each signer P_{i_j} makes use of the function $\mathsf{Z}(\cdot)$ to generate their own response z_j. The aggregate signature is (w, c, z), where $w = w_1 + w_2$ and $z = z_1 + z_2$. And the verification process is to check if $w = \mathsf{V}(pk_{i_1} + pk_{i_2}, c, z)$.

1.2.2 Generating challenge

Now, we describe how P_{i_1} and P_{i_2} collaborate to generate the challenge c. In scenarios where a leader is present in the scheme, this process is straightforward. The signers can simply send their respective w_j to the leader, who then generates c and sends it back to the signers. However, since we aim to avoid having a leader in the scheme, this method cannot be used.

On the other hand, it is not secure to simply exchange w_j with each other. For example, if P_{i_1} is a malicious user, they can adaptive choose w_1 after receiving w_2, which will lead to a Wagner-like attack [21, 49]. Of course, this vulnerability can be addressed by exchanging commitments first and then exchanging w_j after both parties received each other's commitment. Nevertheless, implementing this method introduces an additional round of interaction.

We observe that a technique used by Damgård et al. [19] for constructing an efficient multi-signature scheme comes close to what we need. So we adopt their method to achieve our goal. The process of generating challenge is as follows: each signer P_{i_j} first computes a commitment $\mathsf{Com}(w_j; r_j)$ for their respective w_j. Then, they exchange $\mathsf{Com}(w_j; r_j)$ with each other. To avoid introducing additional interactions, we require the used commitment scheme is a trapdoor homomorphic commitment scheme. The trapdoor plays a crucial role in providing provable security, while the homomorphism ensures the correctness of the scheme. The homomorphic property can be expressed as follows:

$$\mathsf{Com}(w_1; r_1) + \mathsf{Com}(w_2; r_2) = \mathsf{Com}(w_1 + w_2; r_1 + r_2).$$

After commitments are exchanged, the challenge is

$$c = H(\mathsf{Com}(w_1 + w_2; r_1 + r_2), m, R).$$

Finally, after respective partial signatures are calculated, the signers exchange (z_j, w_j, r_j) with each other. Thus, the aggregate signature is $(z, \mathsf{Com}(w; r), r)$, where $z = z_1 + z_2$, $w = w_1 + w_2$ and $r = r_1 + r_2$. And the verification process can be modified to check if the opening of the commitment $\mathsf{Com}(w; r)$ is $\mathsf{V}(pk_{i_1} + pk_{i_2}, c, z)$.

1.2.3 Achieving anonymity

There is an issue with the verification process mentioned above: it requires the verifier to use the public keys of the signers, which compromises anonymity. To address this issue, we can employ a zero-knowledge proof protocol. Specifically, we treat the verification process as a relation, where $(\mathsf{Com}(w; r), c, z)$ is the language, and the public keys of the signers (pk_{i_1}, pk_{i_2}) is the witness. Therefore, by using any non-interactive zero-knowledge (NIZK) proof scheme that works for arbitrary NP languages (such as [14, 46]), the signers can generate a proof π to convince the verifier that the signature is valid.

However, in general, proof schemes that work for arbitrary NP languages are significantly less efficient than proof schemes that designed for specific relations. Therefore, we will construct a specialized proof protocol for the above verification relation. To enable the design of an efficient proof scheme, we require the function $\mathsf{V}(\cdot)$ of the identification scheme can be expressed as:

$$\mathsf{V}(pk, c, z) = \mathsf{V}_1(c, z) \cdot pk + \mathsf{V}_2(c, z),$$

and the commitment scheme satisfies that if x is the opening of a commitment com, then $x + m$ is the opening of the commitment $com + m$ (see Definition 14 for the details). By taking advantage of these two properties, we can transform the above verification relation into the following form: prove that the opening of the commitment $\mathsf{Com}(w; r) - V_2(c, z)$ is the sum of some two elements in the public set $P = [V_1(c, z)pk_1, \ldots, V_1(c, z)pk_n]$, where pk_i is the public key of user P_i.

1.2.4 *t*-out-of-*n* proof

The task at hand is to prove that the opening of a commitment com is the sum of two elements in a public set $P = [p_1, \ldots, p_n]$. Suppose these two elements are p_i and p_j respectively, and their corresponding indicator vector is $\vec{v}$, which has a value of 1 at indices i and j, while the rest of the elements are zero. To accomplish this task, we leverage the commit-and-prove paradigm. Specifically, signers first compute a commitment com' for the vector $\vec{v}$. Then, they prove to the verifier that they have a witness $(p, \vec{v})$ that satisfies the following two conditions:

1. the opening of com is p and the opening of com' is $\vec{v}$;
2. $\vec{v} \in \{0, 1\}^n$, $\|\vec{v}\|_1 = 2$, $P\vec{v} = p$;

We refer to the protocol capable of accomplishing this task as the t-out-of-n proof protocol. When we put everything together, we obtain our GC-TRS scheme. For more details, please refer to Sect. 3.

1.2.5 Lattice-based instantiation

Now, we describe how to construct lattice-based instantiations. We obtain the canonical identification scheme by modifying the constructions in [35, 36], and the commitment scheme by modifying the constructions in [9, 19]. And in this subsection, we will mainly focus on describing the construction of the t-out-of-n proof protocol.

Our construction performs over the cyclotomic polynomial ring $\mathcal{R}_q$, which can be split into exactly ϖ factors (see Sect. 2.4 for the details). The t-out-of-n proof protocol is related to the used commitment scheme. Thus, we will provide a brief description of the commitment scheme employed. The commitment is of the form:

$$\vec{\boldsymbol{t}} = \bar{\boldsymbol{B}}\vec{\boldsymbol{r}} + \begin{bmatrix} \mathbf{0} \\ \vec{\boldsymbol{m}} \end{bmatrix},$$

where $\bar{\boldsymbol{B}} \in \mathcal{R}_q^{k \times m}$ is the commitment key, $\vec{\boldsymbol{r}} \in \mathcal{R}_q^m$ is the randomness, and $\vec{\boldsymbol{m}}$ is the message.

Note that the input message of the above commitment scheme needs to be vectors of ring elements. So, to compute a commitment of $\vec{v}$, we need to encode it to a ring element vector first. Different encoding methods will result in different proof protocols. Below, we will describe two encoding methods along with the corresponding proof method.

1.2.6 Linear *t*-out-of-*n* proof protocol

A straightforward encoding method is dividing the vector $\vec{v}$ into n' blocks and then converting each block into a ring element. Here, n' is defined as n/ϖ (assuming n' is an integer). Specifically, we divide $\vec{v}$ into n' blocks, with each block containing ϖ terms denoted as $\vec{v} = (\vec{v}_1|\ldots|\vec{v}_{n'})$. The encoding of $\vec{v}$ is defined as

$$\mathsf{Encode}(\vec{v}) := \vec{\boldsymbol{v}} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{n'}) \in \mathcal{R}_q^{n'},$$

where $\boldsymbol{v}_i = \mathsf{NTT}^{-1}(\vec{v}_i)$. Here, $\mathsf{NTT}(\cdot)$ operation is the Number Theoretic Transform defined in Sect. 2.4.

Let $\boldsymbol{P} = \{\vec{\boldsymbol{p}}_1, \ldots, \vec{\boldsymbol{p}}_n\} \subset \mathcal{R}_q^l$ be a public set, $ck = \bar{\boldsymbol{B}} \in \mathcal{R}_q^{k\times m}$ be a commitment key and $\vec{\boldsymbol{r}}$ be a randomness, then the commitment of $(\boldsymbol{P}\vec{v}, \vec{v})$ under the commitment key $\bar{\boldsymbol{B}}$ is:

$$\vec{\boldsymbol{t}} = \boldsymbol{B}\vec{\boldsymbol{r}} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{P}\vec{v} \\ \mathsf{Encode}(\vec{v}) \end{bmatrix}.$$

In this case, the goal of the prover is to convince the verifier that the opening $(\vec{\boldsymbol{p}}, \vec{\boldsymbol{v}})$ of the commitment $\vec{\boldsymbol{t}}$ satisfies the following three conditions:

1. $\mathsf{NTT}(\vec{\boldsymbol{v}}) \in \{0, 1\}^n$;
2. $\|\mathsf{NTT}(\vec{\boldsymbol{v}})\|_1 = t$;
3. $\vec{\boldsymbol{p}}$ and $\vec{\boldsymbol{v}}$ satisfies $\boldsymbol{P} \cdot \mathsf{NTT}(\vec{\boldsymbol{v}}) = \vec{\boldsymbol{p}}$.

We now explain how to prove the above conditions. Note that $\mathsf{NTT}(\vec{\boldsymbol{v}}) \in \{0, 1\}^n$ if and only if each element $\boldsymbol{v}_i$ of $\vec{\boldsymbol{v}}$ satisfies $\boldsymbol{v}_i(1 - \boldsymbol{v}_i) = 0$. The latter is a quadratic relation that can be proved using the protocol provided in [5]. And the 2nd condition can be expressed as $\langle \mathsf{NTT}(\vec{\boldsymbol{v}}), (1, 1, \ldots, 1)\rangle = t$. Thus,the 2nd and 3rd conditions can be regarded as an inner product relation. Consequently, they can be proved using the masking technique presented in [23]. Putting everything together, we can obtain a linear t-out-of-n proof protocol. For more details, please refer to Sect. 5.1.

1.2.7 Logarithmic *t*-out-of-*n* proof protocol

The above linear t-out-of-n proof protocol is efficient for small values of n. However, as n grows larger, the proof size also increases significantly. To enhance the efficiency of the t-out-of-n proof protocol, we present an alternative encoding method. Leveraging this encoding method, we present a t-out-of-n proof protocol with a logarithmic proof size.

It is worth noting that, the vector $\vec{v}$ is very sparse, with most positions being 0 and only t positions being 1. Our new encoding method takes advantage of this characteristic. Let I represent the set of coordinates corresponding to the elements in $\vec{v}$ that are equal to 1. Consequently, $\vec{v}$ can be expressed as $\vec{v} = \sum_{i\in I} \vec{e}_i$, where $\vec{e}_i \in \{0, 1\}^n$ has exactly one 1 in the ith coefficient. Therefore, we can adopt the technique used in [27, 38] to represent $\vec{e}_i$ in the form of a set of small vectors tensor products. Specifically, assuming $n = \varpi^{n'}$, any vector $\vec{e}_i$ can be uniquely decomposed into smaller vectors $\vec{v}_{i,1}, \ldots \vec{v}_{i,n'} \in \{0, 1\}^{\varpi}$, each having exactly one 1, and $\vec{e}_i = \vec{v}_{i,1} \otimes \cdots \otimes \vec{v}_{i,n'}$. After renumbering, we have $\vec{v} = \sum_{i\in[t]}(\vec{v}_{i,1} \otimes \cdots \otimes \vec{v}_{i,n'})$. Thus, the encoding of $\vec{v}$ can be defined as

$$\mathsf{Encode}(\vec{v}) := (\boldsymbol{v}_{1,1}, \ldots, \boldsymbol{v}_{1,n'}, \ldots, \boldsymbol{v}_{t,1}, \ldots, \boldsymbol{v}_{t,n'}) \in \mathcal{R}_q^{tn'},$$

where $\boldsymbol{v}_{i,j} = \mathsf{NTT}^{-1}(\vec{v}_{i,j})$. This encoding method uses only $O(t \log n)$ ring elements to uniquely represent the vector $\vec{v}$, thereby reducing the size of the commitment as well.

Let $\vec{\boldsymbol{t}}$ be the commitment of $(\boldsymbol{P}\vec{v}, \mathsf{Encode}(\vec{v}))$. In this case, the goal of the prover is also to convince the verifier that the opening $(\vec{\boldsymbol{p}}, \vec{\boldsymbol{v}})$ of the commitment $\vec{\boldsymbol{t}}$ satisfies the following three conditions:

- $\sum_{i\in[t]}(\mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'})) \in \{0,1\}^n$;
- $\|\sum_{i\in[t]}(\mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'}))\|_1 = t$;
- $\vec{\boldsymbol{p}}$ and $\vec{\boldsymbol{v}}$ satisfies $\boldsymbol{P} \cdot \sum_{i\in[t]}(\mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'})) = \vec{\boldsymbol{p}}$.

To prove the first two conditions, our actual approach is to convince the verifier that the opening $\vec{\boldsymbol{v}}$ satisfies a stronger relation:

1. Each element $\boldsymbol{v}_{i,j}$ of $\vec{\boldsymbol{v}}$ satisfies $\mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi}$;
2. Each element $\boldsymbol{v}_{i,j}$ of $\vec{\boldsymbol{v}}$ satisfies $\|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1$;
3. For any $i_1, i_2 \in [t]$, with $i_1 \neq i_2$, the vectors $\mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'})$ and $\mathsf{NTT}(\boldsymbol{v}_{j,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{j,n'})$ are different.

We briefly explain that it is sufficient to prove the above conditions. For all $i \in [t]$, define $\vec{v}_i = \mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'})$. Since $\mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi}$ and $\|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1$, it follows that $\vec{v}_1, \ldots, \vec{v}_t \in \{0,1\}^n$ and have exactly one 1 each. According to the third condition, the vectors $\vec{v}_i$ are distinct. Thus, the vector $\vec{v} = \vec{v}_1 + \cdots + \vec{v}_t$ satisfies $\vec{v} \in \{0,1\}^n$ and $\|\vec{v}\|_1 = t$.

We now elaborate on how to prove these conditions. Firstly, the 1st and 2nd conditions can be proved in the same way as in the linear t-out-of-n proof protocol. For the 3rd condition, since vectors $\mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'}) \in \{0,1\}^n$ for any $i \in [t]$, a straightforward idea is to prove that

$$\langle \mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'}), \mathsf{NTT}(\boldsymbol{v}_{j,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{j,n'}) \rangle = 0,$$

for any distinct $i, j \in [t]$. Based on some calculations, this is equivalent to proving that $\prod_{k=1}^{n'} \langle \mathsf{NTT}(\boldsymbol{v}_{i,k}), \mathsf{NTT}(\boldsymbol{v}_{j,k}) \rangle = 0$ for any distinct $i, j \in [t]$. While this is indeed a higher-order relationship, we can reduce it to a quadratic relationship by using the common technique of introducing intermediate commitments. Then, we can combine the quadratic proof protocol [5] to achieve our goal.

However, this approach would introduce a significant number of intermediate commitments, which will impact the final proof size. Therefore, we employ a more clever technique in this paper. We observe that since $\mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi}$ and $\|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1$, we have that $\langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle = 0$ is equivalent to $\boldsymbol{v}_{i_1,j} \neq \boldsymbol{v}_{i_2,j}$, and $\langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle = 1$ is equivalent to $\boldsymbol{v}_{i_1,j} = \boldsymbol{v}_{i_2,j}$. Thus, $\prod_{j=1}^{n'} \langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle = 0$ is equivalent to that there exists $j_0 \in [n']$, such that $\boldsymbol{v}_{i_1,j_0} \neq \boldsymbol{v}_{i_2,j_0}$ and furthermore, which is equivalent to $\sum_{j=1}^{n'} \langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle < n'$. Therefore, to prove the 3rd condition, we now only need to prove $\sum_{j=1}^{n'} \langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle < n'$ for any distinct $i_1, i_2 \in [t]$, which becomes a quadratic relation.

To prove this, we begin by committing to a vector $\vec{b}$, which is the binary representation of integer $n' - \sum_{j=1}^{n'} \langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle - 1$. We then prove that $\vec{b}$ is indeed non-zero binary vector and

$$\| \sum_{j=1}^{n'} \langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle + \langle \vec{b}, (1, 2, 2^2, \ldots, 0, \ldots, 0) \rangle = n' - 1.$$

Next, we adopt the recursive method proposed in [38] to prove the relation $\boldsymbol{P} \cdot \sum_{i\in[t]}(\mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'})) = \vec{\boldsymbol{p}}$. Finally, amortization techniques [8, 10] will also be employed to further compress the proof size. Putting everything together, we can obtain a logarithmic t-out-of-n proof protocol. For more details, please refer to Sect. 5.2.

1.3 Paper organization

The remaining of this paper is organized as follows. Section 2 covers the preliminaries including security assumptions and the used formal TRS model. In Sect. 3, we present the GC-TRS construction. And we present the lattice-based building blocks instantiation in Sects. 4.1 and 4.2, respectively. In Sects. 5.1 and 5.2, we presented our linear and logarithmic t-out-of-n proof protocols, respectively. Finally, the LTRS and CTRS schemes are given in Sects. 6.1 and 6.2, respectively.

2 Preliminaries

2.1 Notation

Let q be an odd modulus, and $\mathbb{Z}_q$ be the ring of integers modulo q. For $a < b$ and $n \in \mathbb{N}$, we write $[a, b] := \{a, a+1, \ldots, b\}$ and $[n] := [1, \ldots, n]$. For $x \in \mathbb{Z}$, we define $x \mod q$ to be the unique element in the interval $[-\frac{q-1}{2}, \frac{q-1}{2}]$ that is congruent to x modulo q. We write lower-case letters with arrows ($\vec{a}, \vec{b}$, etc.) to denote column vectors with coefficients in $\mathbb{Z}$ or $\mathbb{Z}_q$ and write upper-case letters (e.g., A, B) for matrices in $\mathbb{Z}$ or $\mathbb{Z}_q$.

Let d be a power of two, $\mathcal{R}$ and $\mathcal{R}_q$ be the rings $\mathbb{Z}[X]/(X^d+1)$ and $\mathbb{Z}_q[X]/(X^d+1)$, respectively. $\mathcal{R}_q^\times$ denotes for all invertible elements in $\mathcal{R}_q$. Bold lower-case letters (e.g., $\boldsymbol{a}, \boldsymbol{b}$) denote elements in $\mathcal{R}$ or $\mathcal{R}_q$ and bold lower-case letters with arrows (e.g., $\vec{\boldsymbol{a}}, \vec{\boldsymbol{b}}$) represent column vectors with coefficients in $\mathcal{R}$ or $\mathcal{R}_q$. We also write bold upper-case letters (e.g., $\boldsymbol{A}, \boldsymbol{B}$) for matrices in $\mathcal{R}$ or $\mathcal{R}_q$. For a polynomial denoted as a bold letter, we write its ith coefficient as the corresponding regular font letter subscript i, e.g. $x_0 \in \mathbb{Z}_q$ is a constant coefficient of $\boldsymbol{x} \in \mathcal{R}_q$.

For an element $x \in \mathbb{Z}_q$, we write $|x|$ to mean $|x \mod q|$. Define the ℓ^p norms for $\boldsymbol{x} \in \mathcal{R}_q$ as $\|\boldsymbol{x}\|_p = (|x_0|^p + \cdots + |x_{d-1}|^p)^{1/p}$ and the ℓ^p norms for $\vec{\boldsymbol{x}} \in \mathcal{R}_q^k$ as $\|\vec{\boldsymbol{x}}\|_p = (\|\boldsymbol{x}_1\|^p + \cdots + \|\boldsymbol{x}_k\|^p)^{1/p}$. For simplicity, when $p = 2$, we omit 2 and write as $\|\vec{\boldsymbol{x}}\|$.

By default, in this paper all vectors are column vectors, and we write $(\vec{v}\|\vec{w})$ for the concatenation of $\vec{v}$ and $\vec{w}$ which is still a column vector. More specifically, if $\vec{v} \in \mathbb{Z}^{k_1}$ and $\vec{w} \in \mathbb{Z}^{k_2}$, then $(\vec{v}\|\vec{w})$ is the vector $(\vec{v}^T, \vec{w}^T)^T \in \mathbb{Z}^{k_1+k_2}$, where $(\cdot)^T$ denotes the transpose of the vector.

We write $x \leftarrow S$ when $x \in S$ is sampled uniformly at random from the finite set S and similarly $x \leftarrow D$ when x is sampled according to the distribution D. We denote the security parameter by λ, say a function $f(\lambda)$ is negligible if for every polynomial $p(\lambda)$ there exists some n_0 such that for all $n > n_0$, $f(\lambda) < \frac{1}{p(\lambda)}$, and say a function $g(\lambda)$ is overwhelming if $1 - g(\lambda)$ is negligible.

2.2 Security assumptions

Our scheme is based on two well-known lattice problems, namely MLWE [32] and MSIS [45] problems, which are defined over the ring $\mathcal{R}_q$.

Definition 1 ($\mathsf{MSIS}_{\kappa,m,B}$) Given $\boldsymbol{A} \leftarrow \mathcal{R}_q^{\kappa \times m}$, the MSIS problem with parameters $\kappa, m > 0$ and $B > 0$ asks to find a short non-zero vector $\vec{\boldsymbol{z}} \in \mathcal{R}^m$ such that $\boldsymbol{A}\vec{\boldsymbol{z}} = \boldsymbol{0}$ over $\mathcal{R}_q$ and $0 < \|\vec{\boldsymbol{z}}\| \leq B$. We say that a PPT adversary $\mathcal{A}$ has advantage ϵ in solving $\mathsf{MSIS}_{\kappa,m,B}$ if

$$\Pr[0 < \|\vec{\boldsymbol{z}}\| \leq B \wedge \boldsymbol{A}\vec{\boldsymbol{z}} = \boldsymbol{0} \mid \boldsymbol{A} \leftarrow \mathcal{R}_q^{\kappa \times m}; \vec{\boldsymbol{z}} \leftarrow \mathcal{A}(\boldsymbol{A})] \geq \epsilon.$$

Definition 2 ($\mathsf{MLWE}_{m,\iota,\phi}$) The MLWE problem with parameters $\iota, m > 0$ and an error distribution ϕ over $\mathcal{R}$ asks the adversary $\mathcal{A}$ to distinguish between the following two cases: (1) $(\boldsymbol{A}, \boldsymbol{A}\vec{\boldsymbol{s}} + \vec{\boldsymbol{e}})$ for $\boldsymbol{A} \leftarrow \mathcal{R}_q^{m \times \iota}$, secret vector $\vec{\boldsymbol{s}} \leftarrow \phi^\iota$ and error vector $\vec{\boldsymbol{e}} \leftarrow \phi^m$; and (2) $(\boldsymbol{A}, \vec{\boldsymbol{b}}) \leftarrow \mathcal{R}^{m \times \iota} \times \mathcal{R}_q^m$. We say that a PPT adversary $\mathcal{A}$ has advantage ϵ in solving $\mathsf{MLWE}_{m,\iota,\phi}$ if

$$\begin{aligned} |\Pr[b = 1 \mid \boldsymbol{A} \leftarrow \mathcal{R}_q^{m \times \iota}; \vec{\boldsymbol{s}} \leftarrow \phi^\iota; \vec{\boldsymbol{e}} \leftarrow \phi^m; b \leftarrow \mathcal{A}(\boldsymbol{A}, \boldsymbol{A}\vec{\boldsymbol{s}} + \vec{\boldsymbol{e}})] \\ - \Pr[b = 1 \mid \boldsymbol{A} \leftarrow \mathcal{R}_q^{m \times \iota}; \vec{\boldsymbol{b}} \leftarrow \mathcal{R}_q^m; b \leftarrow \mathcal{A}(\boldsymbol{A}, \vec{\boldsymbol{b}})]| \geq \epsilon. \end{aligned}$$

In estimating the practical security of these two problems against known attacks, the parameter m may not be crucial. Therefore, we sometimes omit m and use the notations $\mathsf{MSIS}_{\kappa,B}$ and $\mathsf{MLWE}_{\iota,\phi}$. The parameters κ and ι denote the module ranks for MSIS and MLWE, respectively. Additionally, when ϕ is a uniform distribution over the set $\mathbb{S}_\eta$, we simply denote it as $\mathsf{MLWE}_{\iota,\eta}$, and when ϕ is a discrete Gaussian distribution $\mathbb{D}_\sigma^{dm}$, we denote it as $\mathsf{MLWE}_{\iota,\sigma}$.

2.3 Threshold ring signature

We review the definition of the TRS and its game-based security notion which is based on [28, 29].

Definition 3 A (t, n)-TRS scheme is a 4-tuple of PPT algorithms (Setup, KeyGen, Sign, Verify) that enable a set of signers $I \subset R$ sign a message msg with respect to the ring R, where $|I| = t$ and $|R| = n$. The syntax is as follows:

- $pp \leftarrow \mathsf{Setup}(1^\lambda)$: On input the security parameter λ, it generates public parameters pp. The public parameters are implicit input to all other algorithms and will be omitted when clear from the context.
- $(pk, sk) \leftarrow \mathsf{KeyGen}(pp)$: On input the public parameters pp, it generates a public key pk and a corresponding secret key sk for a signer.
- $\Sigma \leftarrow \mathsf{Sign}(msg, S, R)$: On input a message msg, a set of signer's secret keys S, and a set of public keys R, this potentially interactive procedure outputs a signature Σ.
- $b \leftarrow \mathsf{Verify}(msg, R, \Sigma, t)$: On input a message msg, a ring R, a signature Σ and a verification threshold t, this deterministic algorithm outputs a bit b. It outputs $b = 1$ if the signature Σ is valid, and $b = 0$ otherwise.

A secure TRS scheme satisfies correctness, t-unforgeability, and t-anonymity. Correctness guarantees that a signature generated by sufficiently honest users will always pass the verification algorithm. t-unforgeability requires that an adversary who has corrupted up to $t - 1$ signers will not be able to generate a valid signature for threshold t. And t-anonymity

says that for any non-signer (including non-signers in the ring), it is infeasible to infer from a valid signature which t honest users contributed to the signature generation. The formal definitions are as follows.

Definition 4 (*Correctness*) A TRS scheme has correctness if for any $pp \leftarrow \mathsf{Setup}(1^\lambda)$, any pair $L = \{(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp)\}_{i=1}^{q_{key}}$, any ring $R \subseteq L$, any set of signers $I \subseteq R$ with corresponding secret keys set S and $|I| = t$, and any message msg, we have

$$\Pr[\mathsf{Verify}(msg, R, \Sigma, t) = 1 \mid \Sigma \leftarrow \mathsf{Sign}(msg, S, R)] \geq 1 - \mathsf{negl}(\lambda).$$

In order to formally describe t-unforgeability and t-anonymity, we first give the following two oracles. In the definitions, the adversary may access to these oracles in arbitrary interleaf during the corresponding experiments.

2.3.1 Corruption Oracle $\mathcal{CO}(pk)$

On input a public key pk, the challenger $\mathcal{C}$ first checks whether the public key is generated by himself. If so, $\mathcal{C}$ returns the corresponding secret key sk; otherwise returns $\perp$. Then $\mathcal{C}$ adds pk to the query record set $\mathcal{Q}_{CO}$, which is initialized as an empty list.

2.3.2 Signing Oracle $\mathcal{SO}(msg, R, I)$

In our definition, the signing oracle allows for multiple signing sessions to be executed concurrently. We follow the approach in [19], continuously recording a signature query by letting the adversary to include a serial number (sid) when making the signing query. To produce a signature, the adversary needs to interact with challenger $\mathcal{C}$ and participate in the signing procedure. Here, we provide a more detailed description of signature oracles.

On input a message msg, a set of public keys R, a list of signers $I \subset R$, and a serial number sid, $\mathcal{C}$ proceeds as follows:

- If $\mathcal{C}$ is queried with sid for the first time:
  1. $\mathcal{C}$ checks whether the public keys in I are generated by himself. If not, $\mathcal{C}$ returns $\perp$.
  2. $\mathcal{C}$ decomposes I to $I = I_{corr} \sqcup I_{hon}$, where I_{corr} denotes corrupted signers and I_{hon} denotes honest signers and $\sqcup$ represents the disjoint union.
  3. $\mathcal{C}$ adds (msg, R, I) to the query record set $\mathcal{Q}_{SO}$, which is initialized as an empty list.
  4. $\mathcal{C}$ generates the 1st round signature message of the users in I_{hon} and sends it to $\mathcal{A}$, this message is the oracle reply.
  5. $\mathcal{C}$ generate a state $st = 1$ (to record the state of this signature query).
- If sid has queried before:
  1. $\mathcal{C}$ looks at the state st to determine which step the query has reached. If $st = \perp$, $\mathcal{C}$ halts with output $\perp$.
  2. $\mathcal{C}$ checks whether it has received the stth round of messages from the users in I_{corr} via adversary $\mathcal{A}$. If not, $\mathcal{C}$ halts with output $\perp$.
  3. $\mathcal{C}$ calculates the $st + 1$th round of messages and sends them to $\mathcal{A}$.
  4. If this is the final round of the signing procedure, $\mathcal{C}$ sets $st = \perp$, otherwise, it sets $st = st + 1$.

Definition 5 (*t-unforgeability*) A (t, n)-TRS scheme satisfies t-unforgeability w.r.t. insider corruption if for any polynomial time adversary $\mathcal{A}$, we have

$$\Pr\left[\begin{array}{l|l} \mathsf{Verify}(msg^*, R^*, \Sigma^*, t) = 1 & pp \leftarrow \mathsf{Setup}(1^\lambda), \\ \wedge R^* \subset L \wedge |\mathcal{Q}_{CO} \cap R^*| < t & \text{for } i \in [1, q_{key}]: \\ \wedge (msg^*, R^*, \cdot) \notin \mathcal{Q}_{SO} & (pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp), \\ & L = \{pk_i\}_{i=1}^{q_{key}}, \\ & (msg^*, R^*, \Sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}}(pp, L) \end{array}\right] \leq \mathrm{negl}(\lambda),$$

where $\mathcal{O} = (\mathcal{CO}, \mathcal{SO})$.

Definition 6 (*t-anonymity*) A (t, n)-TRS scheme satisfies t-anonymity w.r.t. adversarial keys if for any polynomial time adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we have

$$\Pr\left[\begin{array}{l|l} & pp \leftarrow \mathsf{Setup}(1^\lambda), \\ & \text{for } i \in [1, q_{key}]: \\ b = b' \wedge & (pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp), \\ |I_0| = |I_1| = t \wedge & L = \{pk_i\}_{i=1}^{q_{key}}, \\ I_0 \cup I_1 \subseteq R^* \wedge & (msg^*, R^*, I_0, I_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}}(pp, L) \\ (I_0 \cup I_1) \cap \mathcal{Q}_{CO} = \emptyset & b \leftarrow \{0, 1\} \\ & \Sigma^* \leftarrow \mathsf{Sign}(msg^*, S_b, R^*) \\ & b' \leftarrow \mathcal{A}_2^{\mathcal{O}}(st, \Sigma^*) \end{array}\right] \leq \frac{1}{2} + \mathrm{negl}(\lambda),$$

where $\mathcal{O} = (\mathcal{CO}, \mathcal{SO})$, st is the state of $\mathcal{A}$ and S_b denotes the secret keys set of I_b.

2.4 Cyclotomic rings

Let ϖ, d be a power of two with $\varpi \leq d, q-1 \equiv 2\varpi \pmod{4\varpi}$, and let us fix a primitive 2ϖth root of unity ζ in $\mathbb{Z}_q$. Then, the polynomial $X^d + 1$ factors into ϖ irreducible polynomials in $\mathbb{Z}_q$, i.e., $X^d + 1 \equiv \prod_{i=0}^{\varpi-1}(X^{\frac{d}{\varpi}} - \zeta^{2i+1}) \mod q$. By Chinese remainder theorem, we obtain $\mathcal{R}_q \cong \mathcal{R}_q^{(0)} \times \cdots \times \mathcal{R}_q^{(\varpi-1)}$ for $\mathcal{R}_q^i = \mathbb{Z}_q[X]/(X^{\frac{d}{\varpi}} - \zeta^{2i+1})$ (see Theorem 2.3 of [39]).

Let $\mathcal{M}_q := \{\boldsymbol{p} \in \mathbb{Z}_q[X] : \deg(\boldsymbol{p}) < d/\varpi\}$ be the $\mathbb{Z}_q$-module of polynomials of degree less than d/ϖ. Following [23, 38], we define the Number Theoretic Transform (NTT) of a polynomial $\boldsymbol{p} \in \mathcal{R}_q$ as follows:

$$\mathsf{NTT}(\boldsymbol{p}) := \left[\, \hat{\boldsymbol{p}}_0 \| \cdots \| \hat{\boldsymbol{p}}_{\varpi-1} \,\right] \in \mathcal{M}_q^{\varpi} \text{ where } \mathsf{NTT}(\boldsymbol{p})_i = \hat{\boldsymbol{p}}_i = \boldsymbol{p} \bmod (X^{\frac{d}{\varpi}} - \zeta^{2i+1}).$$

We expand the definition of NTT to vectors of polynomials $\vec{\boldsymbol{p}} \in \mathcal{R}_q^k$, where the NTT operation applies to each coefficient of $\vec{\boldsymbol{p}}$, resulting in a vector in $\mathcal{M}_q^{k\varpi}$. We also define the inverse NTT operation. Namely, for a vector $\vec{v} \in \mathcal{M}_q^{\varpi}$, $\mathsf{NTT}^{-1}(\vec{v})$ is the polynomial $\boldsymbol{p} \in \mathcal{R}_q$ such that $\mathsf{NTT}(\boldsymbol{p}) = \vec{v}$.

Let $\vec{v} = (v_0, \ldots, v_{\varpi-1}), \vec{w} = (w_0, \ldots, w_{\varpi-1}) \in \mathcal{M}_q^{\varpi}$. We define the component-wise product $\vec{v} \circ \vec{w}$ to be the vector $\vec{u} = (u_0, \ldots, u_{\varpi-1}) \in \mathcal{M}_q^{\varpi}$ such that $u_j = v_j w_j \bmod (X^{d/\varpi} - \zeta^{2i+1})$ for $j \in \mathbb{Z}_{\varpi}$. By definition, we have the following property of the inverse NTT operation:

$$\mathsf{NTT}^{-1}(\vec{v}) \cdot \mathsf{NTT}^{-1}(\vec{w}) = \mathsf{NTT}^{-1}(\vec{v} \circ \vec{w}).$$

Similarly, we define the inner product[3] over $\mathcal{M}_q$:

$$\langle \vec{v}, \vec{w} \rangle = \sum_{i=0}^{\varpi-1} (v_i w_i \bmod (X^{d/\varpi} - \zeta^{2i+1})).$$

We generalize this operations to work for vectors $\vec{v} = (\vec{v}_1 \| \ldots \| \vec{v}_k)$ and $\vec{w} = (\vec{w}_1 \| \ldots \| \vec{w}_k) \in \mathcal{M}_q^{k\varpi}$ of length being a multiple of ϖ in the usual way. In particular, $\langle \vec{v}, \vec{w} \rangle = \sum_{i=1}^{k} \langle \vec{v}_i, \vec{w}_i \rangle$.

Eventually, for a matrix $A \in \mathcal{M}_q^{n \times k\varpi}$ with rows $\vec{a}_1, \ldots, \vec{a}_n \in \mathcal{M}_q^{k\varpi}$ and a vector $\vec{v} \in \mathcal{M}_q^{k\varpi}$, we define the matrix-vector operation:

$$A\vec{v} = \big(\langle \vec{a}_1, \vec{v} \rangle \| \cdots \| \langle \vec{a}_n, \vec{v} \rangle \big) \in \mathcal{M}_q^n.$$

We also need the following lemmas, which will be used later.

Lemma 1 (Lemma 2.1 [38]) *Let $n, k \in \mathbb{N}$, for any $A \in \mathcal{M}_q^{n\varpi \times k\varpi}$, $\vec{v} \in \mathcal{M}_q^{n\varpi}$ and $\vec{s} \in \mathbb{Z}_q^{k\varpi}$, we have $\langle A\vec{s}, \vec{v} \rangle = \langle \vec{s}, A^T \vec{v} \rangle$.*

Lemma 2 (Lemma 2.1 [23]) *Let $\boldsymbol{p} = p_0 + p_1 X + \cdots + p_{d-1} X^{d-1} \in \mathcal{R}_q$, then we have*

$$\frac{1}{\varpi} \sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{p})_i = \sum_{i=0}^{d/\varpi-1} p_i X^i.$$

2.5 Probability distributions

We write $\mathbb{S}_\eta$ to denote the set of polynomials in $\mathcal{R}$ with infinity norm at most $\eta \in \mathbb{Z}^+$. We define $D(S_c) : \mathbb{S}_1 \mapsto [0, 1]$ to be the probability distribution on $\mathbb{S}_1$ such that the coefficients of a challenge $\boldsymbol{c} \leftarrow D(S_c)$ are independently identically distributed with $\Pr[0] = 1/2$ and $\Pr[1] = \Pr[-1] = 1/4$.

Consider the coefficients of the polynomial $\boldsymbol{c} \bmod (X^{d/\varpi} - \zeta^{2i+1})$ for $\boldsymbol{c} \leftarrow D(S_c)$. Then, all coefficients follow the same distribution over $\mathbb{Z}_q$. We write Y for the random variable over $\mathbb{Z}_q$ that follows this distribution. Following [5], we can give an upper bound on the maximum probability of Y.

Lemma 3 (Lemma 3.2 in [5]) *Let the random variable Y over $\mathbb{Z}_q$ be defined as above. Then for all $x \in \mathbb{Z}_q$, we have*

$$\Pr[Y = x] \le \frac{1}{q} + \frac{2\varpi}{q} \sum_{j=0}^{\varpi-1} \prod_{i=0}^{\varpi-1} \left| \frac{1}{2} + \frac{1}{2} \cos(2\pi(2j+1) y \zeta^i / q) \right|.$$

In particular, by numerical computing, we obtain that for $d = 128$, the maximum probability for each coefficient of $\boldsymbol{c} \bmod (X^{d/\varpi} - \zeta^{2i+1})$ is around q^{-1}. Thus, the polynomial $\boldsymbol{c} \leftarrow D(S_c)$ is invertible in $\mathcal{R}_q$ with overwhelming probability as long as parameters q, d, ϖ are selected so that $q^{-d/\varpi}$ is negligible.

[3] Note that this operation is not an inner product in the strictly mathematical sense. However, it has symmetry and distributive law which are characteristic of an inner product. We refer the reader to [38] for a further discussion.

2.5.1 Gaussian distribution

We now define the discrete Gaussian distribution as follows.

Definition 7 The discrete Gaussian distribution on $\mathcal{R}^k$ centered around $\mathbf{0}$ with standard deviation σ is given by

$$\mathbb{D}_{\sigma}^{dk}(\vec{z}) = \frac{e^{-\|\vec{z}\|^2/2\sigma^2}}{\sum_{\vec{y}\in\mathcal{R}^k} e^{-\|\vec{y}\|^2/2\sigma^2}}.$$

In our instantiation, we need to deal with the sum of independent vectors from a discrete Gaussian distribution and the tail bound of the discrete Gaussian distribution. Therefore, let us recall the following useful lemmas.

Lemma 4 (Lemma 9 in [24], Theorem 3.3 in [42]) *Let $\vec{y}_1, \ldots, \vec{y}_t$ be independent vectors with distribution $\mathbb{D}_{\sigma}^{dk}$. If we have $\sigma \geq \eta(\mathbb{Z}^{dk})/\sqrt{\pi}$ for the smoothing parameter $\eta(\mathbb{Z}^{dk})$ of $\mathbb{Z}^{dk}$, then the distribution of $\vec{y} = \vec{y}_1 + \cdots + \vec{y}_t$ is statistically close to $\mathbb{D}_{\sigma\sqrt{t}}^{dk}$. In particular, if we have $dk \leq 2^{45}$ and $\sigma \geq 11/\pi$, then the statistical distance between the distribution of $\vec{y} = \vec{y}_1 + \cdots + \vec{y}_t$ and $\mathbb{D}_{\sigma\sqrt{t}}^{dk}$ is at most 2^{-128}.*

Lemma 5 (Lemma 1.5 in [7]) *Let $\vec{z}$ be a vector with distribution $\mathbb{D}_{\sigma}^{dk}$, then we have*

$$\Pr[\|\vec{z}\| > \sigma\sqrt{2dk}] < 2^{-0.11\cdot dk}.$$

2.5.2 Rejection sampling

In our lattice-based instantiation, in order to make the prover's response independent of the secret information, we will use the rejection sampling technique from [15, 36].

Algorithm 1 $\mathsf{Rej}(\vec{z}, \vec{v}, \sigma)$:

1: $u \leftarrow [0, 1)$
2: **If** $u > \frac{1}{M} \cdot \exp(\frac{-2\langle\vec{z},\vec{v}\rangle + \|\vec{v}\|^2}{2\sigma^2})$, **then return 0**
3: **Else return 1**

Lemma 6 (Lemma 2.4 in [15]) *Let $V \subset \mathcal{R}^k$ be a set of polynomials with norm at most T and $\psi : V \mapsto [0, 1]$ be a probability distribution. Let M be the repetition rate parameter and σ be the standard deviation of discrete Gaussian distribution with $M \geq e^{\frac{24\sigma T + T^2}{2\sigma^2}}$. Now, sample $\vec{v} \leftarrow \psi$ and $\vec{y} \leftarrow \mathbb{D}_{\sigma}^{kd}$, set $\vec{z} = \vec{y} + \vec{v}$, and run $b \leftarrow \mathsf{Rej}(\vec{z}, \vec{v}, \sigma)$ as defined in Algorithm 1. Then, the probability that $b = 1$ occurs is at least $(1 - 2^{-100})/M$, and conditioned on the output being 1, the statistical distance between distribution of $(\vec{v}, \vec{z})$ and $\psi \times \mathbb{D}_{\sigma}^{kd}$ is at most $2^{-100}/M$.*

2.5.3 Trapdoor and preimage sampling

The following lemma demonstrates the properties of a lattice trapdoor and efficient preimage sampling algorithm. In pursuit of efficiency, we use the computational version of the trapdoor over the ring $\mathcal{R}_q$. The parameters in the lemma were selected based on the parameters selected in [12].

Lemma 7 (Adapted from [41, Theorem 5.1]) *There exists* PPT *algorithms* TrapGen *and* SampleD *satisfying the following:*

1. $\mathsf{TrapGen}(k, l, \sigma)$ *takes integers* k *and* l *as well as parameter* σ *as inputs, and outputs a matrix* $\bar{\boldsymbol{A}} = [\boldsymbol{A}|\boldsymbol{I}] \in \mathcal{R}_q^{k \times m}$ *and a trapdoor* $\boldsymbol{T}$*, such that the distribution of* $\boldsymbol{A}$ *is computationally indistinguishable from the uniform distribution under the* $\mathsf{MLWE}_{l,\sigma}$ *assumption, where* $m = l + k \cdot \lceil \log q \rceil + k$.
2. *Let* $(\bar{\boldsymbol{A}}, \boldsymbol{T})$ *be the outputs of* $\mathsf{TrapGen}(k, l, \sigma)$*, then for any* $\vec{\boldsymbol{u}} \in \mathcal{R}_q^k$*, there exists a randomized algorithm* $\mathsf{SampleD}(\bar{\boldsymbol{A}}, \boldsymbol{T}, \vec{\boldsymbol{u}})$ *that outputs a random vector* $\vec{\boldsymbol{s}} \in \mathcal{R}_q^m$ *such that* $\bar{\boldsymbol{A}}\vec{\boldsymbol{s}} = \vec{\boldsymbol{u}}$*, and the following two are statistically indistinguishable:*

$$\{(\vec{\boldsymbol{u}}, \vec{\boldsymbol{s}}) \mid \vec{\boldsymbol{u}} \leftarrow \mathcal{R}_q^k, \vec{\boldsymbol{s}} \leftarrow \mathsf{SampleD}(\bar{\boldsymbol{A}}, \boldsymbol{T}, \vec{\boldsymbol{u}})\}$$

and

$$\{(\vec{\boldsymbol{u}}, \vec{\boldsymbol{s}}) \mid \vec{\boldsymbol{s}} \leftarrow \mathbb{D}_{\sigma'}^{dm}, \vec{\boldsymbol{u}} = \bar{\boldsymbol{A}}\vec{\boldsymbol{s}}\},$$

where $\sigma' = \sqrt{\frac{5}{2\pi}} \cdot \sigma^2 \cdot (\sqrt{(k+l)d} + \sqrt{kd \cdot \lceil \log q \rceil} + \sqrt{\lambda})$.

3 A generic threshold ring signature construction

In this section, we will construct a generic TRS scheme, called GC-TRS, using a homomorphic canonical identification scheme, a trapdoor homomorphic commitment scheme, and a t-out-of-n proof protocol. Before we dive into the construction of our GC-TRS scheme, it is important to first understand the building blocks that we will be using. In the following subsection, we will provide definitions for each of these building blocks, along with their corresponding security properties.

3.1 Building blocks

Let us begin by discussing the homomorphic canonical identification scheme.

3.1.1 Homomorphic canonical identification scheme

The canonical identification scheme [1] is a three-move public-key identification protocol. The protocol is executed between a prover and a verifier, with the prover holding a secret key sk and the verifier having access to a public key pk. The framework present in Alg.2, which is based on previous work by [2, 51]. To make the scheme compatible with lattice-based constructions, we introduce a flag in the Setup. By default, the flag is set to False. However, if the scheme needs to perform an additional verification in the Proof-II, the flag is set to True. In some lattice-based schemes, such as [22], this additional check is used for rejection sampling [35, 36] to ensure the security of the scheme. It is important to note that the symbol $\perp$ is a special value that indicates failure and is not included in the set of valid responses S_z.

The homomorphic canonical identification scheme is a canonical identification scheme that possesses *homomorphic* and *linear* properties. We now define what the definition of these properties are.

Algorithm 2 Canonical Identification Scheme

Setup(1^λ):
1: Set flag and pp
2: **return** (pp, flag)

KeyGen(pp):
1: **return** (pk, sk)

Proof-I(sk):
1: $y \leftarrow D(S_y)$
2: $w = \mathsf{A}(sk; y)$
3: **return** (w, y)

Challenge():
1: **return** $c \leftarrow D(S_c)$

Proof-II(sk, y, c):
1: $z = \mathsf{Z}(sk, y, c)$
2: If $\mathsf{flag} = \mathsf{True}$, run extra verification on (y, c, z) and set $z = \perp$ if failed
3: **return** z

Verify(pk, z, c, w):
1: Check if $z \in S_z$ and $w = \mathsf{V}(pk, c, z)$
2: If failed **return** 0, otherwise **return** 1

Definition 8 (*Additive homomorphism*) A canonical identification scheme is said to be additive homomorphism if the function $\mathsf{V}(\cdot)$ is additive homomorphism:

$$\mathsf{V}(pk_1, c, z_1) + \mathsf{V}(pk_2, c, z_2) = \mathsf{V}(pk_1 + pk_2, c, z_1 + z_2).$$

In the discrete logarithm (DL-setting), the operation between public keys should be multiplication, while in the lattice-setting, the operation is addition. For simplicity, we represent both settings using addition.

Definition 9 (*Linear property*) A canonical identification scheme is said to have linear property if the function $\mathsf{V}(\cdot)$ can be expressed as:

$$\mathsf{V}(pk, c, z) = \mathsf{V}_1(c, z) \cdot pk + \mathsf{V}_2(c, z).$$

We also define t-correctness for the homomorphic canonical identification scheme, which is a generalization of the standard notion of correctness.

Definition 10 (*Correctness*) A homomorphic canonical identification scheme is said to have t-correctness with error δ if the following holds:

- Let $\{(pk_i, sk_i)\}_{i=1}^{l} \leftarrow \mathsf{KeyGen}(pp)$ with $l \in [t]$, if for all $(w_i, y_i) \leftarrow \mathsf{Proof\text{-}I}(sk_i)$, $c \leftarrow D(S_c)$, and $z_i \leftarrow \mathsf{Proof\text{-}II}(sk_i, y_i, c)$ with $z_i \neq \perp$, we have

$$\Pr[\mathsf{Verify}(\sum_{i\in[l]} pk_i, \sum_{i\in[l]} z_i, c, \sum_{i\in[l]} w_i) = 1] \geq 1 - \mathsf{negl}(\lambda).$$

- The probability that an honestly generated transcript (w, c, z) contains $z = \perp$ is bounded by δ, i.e., for all $(pk, sk) \leftarrow \mathsf{KeyGen}(pp)$, all $(w, y) \leftarrow \mathsf{Proof\text{-}I}(sk)$, all $c \leftarrow D(S_c)$ and all $z \leftarrow \mathsf{Proof\text{-}II}(sk, y, c)$, we have $\Pr[z = \perp] \leq \delta$.

We now define a new security notion for the homomorphic canonical identification scheme called "*secure against* $t-1$ *corruption attack*". This notion can be seen as a generalized version of special soundness in the multi-user model. Specifically, even if the adversary corrupts $t-1$ users and gains knowledge of their secret keys, they cannot produce two valid transcripts with the same w given an aggregated public key $pk = \sum_{i=1}^{t} pk_i$.

Definition 11 (*Secure against* $t-1$ *corruption attack*) A homomorphic canonical identification scheme is secure against $t-1$ corruption attack, if for any PPT $\mathcal{A}$, any integer q_{key}

polynomial in λ, we have the formula holds:

$$\Pr\left[\begin{array}{l|l} I^* \subset L, |I^*| = t & pp \leftarrow \mathsf{Setup}(1^\lambda), \\ \wedge |I^* \cap \mathcal{Q}_{CO}| \leq t-1 & \text{for } i \in [1, q_{key}]: \\ \wedge pk = \sum_{pk_i \in I^*} pk_i & (pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp), \\ \wedge \mathsf{Verify}(pk, z, c, w^*) = 1 & L = \{pk_i\}_{i=1}^{q_{key}}, \\ \wedge \mathsf{Verify}(pk, z', c', w^*) = 1 & (z, c, z', c', w^*, I^*) \leftarrow \mathcal{A}^{\mathcal{CO}}(pp, L) \\ \wedge c \neq c', c, c' \in S_c & \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where the oracle given to $\mathcal{A}$ is defined as: the corruption oracle $\mathcal{CO}(i)$ on input an index i, outputs corresponding secret key sk_i and the set $\mathcal{Q}_{CO}$ is the corrupted users queried in $\mathcal{CO}$.

In addition to satisfying the above security requirements, a homomorphic canonical identification scheme also must satisfies the property of honest-verifier zero-knowledge.

Definition 12 (*Strong honest-verifier zero-knowledge*) A canonical identification scheme is said to be strong honest-verifier zero-knowledge (SHVZK) if there exists a distribution $D(S_z)$, such that the distribution (w, c, z) which takes public key pk and $c \leftarrow D(S_c)$ as inputs, samples $z \leftarrow D(S_z)$ and then computes $w = \mathsf{V}(pk, c, z)$ is indistinguishable from an accepting protocol transcript generated by a real protocol run.

It is worth noting that many standard identification schemes, such as the Schnorr identification scheme [48], meet these requirements. Furthermore, in Sect. 4.1, we will show that the well-known lattice-based identification scheme [36] can also satisfy these properties with appropriate adjustments to its parameters.

3.1.2 Trapdoor homomorphic commitment scheme

We now formally define the trapdoor homomorphic commitment scheme based on the definition in [19].

Definition 13 A trapdoor homomorphic commitment scheme consists of the following algorithms:

- $pp \leftarrow \mathsf{Setup}(1^\lambda)$: On input the security parameter λ, the setup algorithm outputs a public parameter pp and defines sets S_{ck}, S_{msg}, S_r, S_{td} and the distribution $D(S_r)$ from which the randomness is sampled.
- $ck \leftarrow \mathsf{Gen}(pp)$: On input the public parameters pp, the key generation algorithm samples a commitment key ck from the commitment key set S_{ck}.
- $com \leftarrow \mathsf{Commit}_{ck}(msg; r)$: On input a commitment key ck, a message msg and a random string $r \in S_r$ as input, the committing algorithm generates a commitment com, where the randomness is sampled according to the distribution $D(S_r)$.
- $b \leftarrow \mathsf{Open}_{ck}(com, r, msg)$: On input a commitment key ck, a message msg, a random string r, and a commitment com as input, the opening algorithm outputs a bit b. If $com = \mathsf{Commit}_{ck}(msg; r)$ and $r \in S_r$ holds, it will output $b = 1$, and otherwise output $b = 0$.
- $(tck, td) \leftarrow \mathsf{TGen}(pp)$: On input the public parameters pp, the trapdoor key generation algorithm generates a commitment key $tck \in S_{ck}$ and a corresponding trapdoor $td \in S_{td}$.
- $com \leftarrow \mathsf{TCommit}_{tck}(td)$: On input a commitment key tck and its corresponding trapdoor td, the trapdoor committing algorithm outputs a commitment com.

– $r \leftarrow \mathsf{Eqv}_{tck}(td, com, msg)$: On input a commitment key tck, its corresponding trapdoor td, a commitment com and a message msg, the equivocation algorithm outputs a random string $r \in S_r$.

In this paper, we require the trapdoor homomorphic commitment scheme to satisfy correctness, hiding, binding, message homomorphism, additive homomorphism, equivocability, and uniform key.

Definition 14 (*Message homomorphism*) A commitment scheme is said to have message homomorphism if for any $ck \leftarrow \mathsf{Gen}(pp)$, any valid transcript (com, r, msg) $(i.e.\ \mathsf{Open}_{ck}(com, r, msg) = 1)$ and any message $msg' \in S_{msg}$, we have $\mathsf{Open}_{ck}(com + msg', r, msg + msg') = 1$.

Definition 15 (*Additive homomorphism*) A commitment scheme is said to have t-additive homomorphism if for any $l \in [t]$, any messages $\{msg_i\}_{i\in[l]} \in S_{msg}$, we have:

$$\Pr\left[\mathsf{Open}_{ck}\left(\begin{array}{l}\sum_{i\in[l]} com_i,\\ \sum_{i\in[l]} r_i,\\ \sum_{i\in[l]} msg_i\end{array}\right) = 1 \;\middle|\; \begin{array}{l} pp \leftarrow \mathsf{Setup}(1^\lambda),\\ ck \leftarrow \mathsf{Gen}(pp),\\ \text{for all } i \in [l] :\ r_i \leftarrow D(S_r),\\ com_i \leftarrow \mathsf{Commit}_{ck}(msg_i; r_i)\end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Definition 16 (*Equivocability*) A trapdoor commitment scheme is said to provide equivocability, if for any $pp \leftarrow \mathsf{Setup}(1^\lambda)$, any adversary $\mathcal{A}$, we have

$$\left|\Pr\left[b = 1 \;\middle|\; \begin{array}{c} ck \leftarrow \mathsf{Gen}(pp)\\ b \leftarrow \mathcal{A}(ck, \mathcal{O}_0)\end{array}\right] - \Pr\left[b = 1 \;\middle|\; \begin{array}{c}(tck, td) \leftarrow \mathsf{TGen}(pp)\\ b \leftarrow \mathcal{A}(tck, \mathcal{O}_1)\end{array}\right]\right| \leq \mathsf{negl}(\lambda),$$

where the oracles given to $\mathcal{A}$ are defined as:

– $\mathcal{O}_0(msg)$: sample $r \leftarrow D(S_r)$, compute $com \leftarrow \mathsf{Commit}_{ck}(msg; r)$ and return (com, r, msg);
– $\mathcal{O}_1(msg)$: compute $com \leftarrow \mathsf{TCommit}_{tck}(td)$, $r \leftarrow \mathsf{Eqv}_{tck}(td, com, msg)$ and return (com, r, msg).

Remark 1 It can be inferred from the property of equivocability that the tck generated by the TGen algorithm can also function as a regular commitment key, which satisfies correctness and all the properties that we have defined in this subsection.

In our TRS scheme, a commitment com can be the sum of t commitments. Therefore, we require that the resulting commitment com also satisfies hiding properties. We call this property *t-hiding* and provide the definition as follows.

Definition 17 (*t-hiding*) A commitment scheme is said to have t-hiding if for any $l \in [t]$ and any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we have

$$\Pr\left[b = b' \;\middle|\; \begin{array}{l} pp \leftarrow \mathsf{Setup}(1^\lambda), ck \leftarrow \mathsf{Gen}(pp),\\ (\{msg_i^0\}_{i\in[l]}, \{msg_i^1\}_{i\in[l]}, st) \leftarrow \mathcal{A}_1(ck, pp),\\ b \leftarrow \{0, 1\},\ \text{for all } i \in [l] :\ r_i \leftarrow D(S_r),\\ com_i \leftarrow \mathsf{Commit}_{ck}(msg_i^b; r_i),\\ com = \sum_{i\in[l]} com_i, b' \leftarrow \mathcal{A}_2(st, com)\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where st is the state of $\mathcal{A}$.

Note that, for a trapdoor homomorphic commitment scheme, equivocability implies t-hiding. Specifically, we have the following lemma.

Lemma 8 *Assuming the trapdoor homomorphic commitment scheme has equivocability property, then it also satisfies t-hiding property.*

Proof Assuming $\mathcal{A}$ is a PPT adversary breaking the t-hiding game, we aim to construct an algorithm $\mathcal{B}$ to break the equivocability game. Suppose $\mathcal{B}$ is given a commitment key ck and an oracle $\mathcal{O}$ from its challenger $\mathcal{C}$[4]. $\mathcal{B}$ then gives ck to $\mathcal{A}$.
`Challenge Stage.` $\mathcal{A}_1$ gives $\mathcal{B}$ two sets of messages $\{msg_i^0\}_{i\in[l]}$ and $\{msg_i^1\}_{i\in[l]}$. Then, $\mathcal{B}$ samples a uniform bit b and uses its own oracle $\mathcal{O}$ on messages $\{msg_i^b\}_{i\in[l]}$ respectively, obtaining commitments $\{com_i\}_{i\in[l]}$. Then $\mathcal{B}$ sets $com = \sum_{i\in[l]} com_i$ and return com to $\mathcal{A}$. Finally, $\mathcal{A}_2$ outputs a bit b' to $\mathcal{B}$. If $b = b'$, $\mathcal{B}$ returns 1 to $\mathcal{C}$; otherwise, $\mathcal{B}$ returns 0 to $\mathcal{C}$.
`Probability Analysis.` Assuming ϵ is the probability of $\mathcal{A}$ successfully guessing the challenge bit b. We now compute the advantage of $\mathcal{B}$ in attacking the equivocability game. If ck is generated by Gen, then $\mathcal{B}$ perfectly simulates the real t-hiding game. Thus, in this case, the probability that $\mathcal{A}$ can successfully guess the challenge bit b is $\frac{1}{2} + \epsilon$. On the other hand, when ck is generated by TGen, the generation process of $\{com_i\}_{i\in[l]}$ is completely irrelevant to the input $\{msg_i^b\}_{i\in[l]}$ by the definition of the oracle $\mathcal{O}$. Thus, in this case, the probability that the output of $\mathcal{A}$ is correct is exactly $\frac{1}{2}$. Hence, $\mathcal{B}$ also has an advantage of ϵ in attacking the equivocability game. As a result, we can break the equivocability of the trapdoor homomorphic commitment scheme, which is contradictory. □

In order to be compatible with lattice-based constructions, we introduce a new binding concept that is at least as strong as the standard one. We call it *binding for weak opening*. The reason for this is that efficient lattice-based zero-knowledge proofs usually do not satisfy the standard soundness definition [31]. This is due to the so-called "knowledge gap" [24, 36], which is an unavoidable phenomenon in efficient lattice-based zero-knowledge proofs. Specifically, in lattice-based proofs, the knowledge extractor can only recover an "approximate" witness for a relaxed relation. To ensure that the message is binding in this case, we need to use the generalized binding property.

Definition 18 (*Weak opening*) Let S_{fac} be a relaxed factor set and S'_r be a relaxed randomness set with $S_r \subseteq S'_r$. For any $ck \leftarrow \mathsf{Gen}(pp)$, let com be a commitment, we call (τ, r, msg) a weak opening for com, if it satisfies $\tau \in S_{fac}$, $\tau r \in S'_r$ and $com = \mathsf{Commit}_{ck}(msg; r)$.

The set of relaxed factors consists of elements that accommodate the extent of relaxation that is allowed in a weak opening. Given a randomness set S_r, the relaxed randomness set S'_r of S_r is defined as a superset of the randomness set S_r. For example, the relaxed set can be the same as the randomness set S_r for the discrete logarithm setting and the relaxed set could be strictly larger for the lattice case.

Definition 19 (*Binding for weak opening*) A commitment scheme is said to be binding for weak opening with respect to S_{fac}, S'_r, if for any PPT adversary $\mathcal{A}$, we have

$$\Pr\left[\begin{array}{l} msg \neq msg' \\ \wedge \tau, \tau' \in S_{fac} \\ \wedge \tau r, \tau' r' \in S'_r \\ \wedge com = \mathsf{Commit}_{ck}(msg; r) \\ \wedge com = \mathsf{Commit}_{ck}(msg'; r') \end{array} \middle| \begin{array}{l} pp \leftarrow \mathsf{Setup}(1^\lambda), \\ ck \leftarrow \mathsf{Gen}(pp), \\ \left(\begin{array}{c} com, (\tau, r, msg), \\ (\tau', r', msg') \end{array}\right) \leftarrow \mathcal{A}(ck) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

[4] The commitment key ck is randomly generated by either the Gen or TGen algorithm.

Remark 2 For some standard settings, such as the DL setting, zero-knowledge proofs do not suffer from the "knowledge gap" issue, meaning that the knowledge extractor can recover an "exact" witness, and the standard binding is sufficient. Fortunately, when we set $S_{fac} = \{1\}$ and $S'_r = S_r$, our definition of binding for weak opening is exactly the same as the standard binding. Thus, our definition is compatible with other settings, including the DL setting.

Definition 20 (*Correctness*) A commitment scheme has correctness if for any $msg \in S_{msg}$, we have

$$\Pr\left[\mathsf{Open}_{ck}(com, r, msg) = 1 \;\middle|\; \begin{array}{l} pp \leftarrow \mathsf{Setup}(1^\lambda), \\ ck \leftarrow \mathsf{Gen}(pp), \\ r \leftarrow D(S_r), \\ com \leftarrow \mathsf{Commit}_{ck}(msg; r) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Remark 3 The definition of additive homomorphism in Definition 15 already includes correctness. Specifically, when $l = 1$, it aligns with the definition of correctness presented here. Nevertheless, to ensure completeness, we also provide an independent definition of correctness.

Definition 21 (*Uniform key*) A commitment scheme is said to have uniform key if the output of $\mathsf{Gen}(pp)$ follows the uniform distribution over the key space S_{ck}.

It is worth noting that the equivocable commitment scheme presented in [6] meet these requirements. In Sect. 4.2, we will demonstrate that the lattice-based trapdoor commitment scheme proposed by Damgård et al. [19] also satisfies the required properties with appropriate adjustments to its parameters.

3.1.3 *t*-out-of-*n* proof protocol

A t-out-of-n proof protocol is a special type of *Sigma-protocol* that is used to prove that a commitment com opens to a partial sum of a publicly known set P. Specifically, given a commitment com and a public set $P = (p_1, \ldots, p_n)$ with n elements, a t-out-of-n proof protocol can demonstrate that there exists a binary vector $\vec{v} \in \{0, 1\}^n$ such that com is the commitment of $P\vec{v} = \sum_{i=1}^{n} p_i v_i$, where the Hamming weight of $\vec{v}$ is exactly t.

We present the definition of a Sigma-protocol based on [11, 25]. A Sigma-protocol is a type of zero-knowledge proof between two parties: the prover and the verifier. A language $\mathcal{L} \subset \{0, 1\}^*$ has a witness relationship $\mathfrak{R} \subset \{0, 1\}^* \times \{0, 1\}^*$ if $x \in \mathcal{L}$ if and only if there exists $w \in \{0, 1\}^*$ such that $(x, w) \in \mathfrak{R}$. We refer to x as a statement and w as a witness for x.

Definition 22 (*Sigma-protocol*) Let $(\mathcal{P}, \mathcal{V})$ be a two-party protocol, and $\mathcal{L}, \mathcal{L}'$ be languages with witness relations $\mathfrak{R}, \mathfrak{R}'$ with $\mathfrak{R} \subset \mathfrak{R}'$. Then, $(\mathcal{P}, \mathcal{V})$ is called a Sigma-protocol for $\mathfrak{R}, \mathfrak{R}'$ with a challenge set $\mathcal{C}$, public input x and private input w, if it satisfies the following conditions:

Three-move form: The protocol has the following form: on input (x, w), $\mathcal{P}$ computes initial commitment ω and sends it to $\mathcal{V}$; on input ω, $\mathcal{V}$ draws a challenge $c \leftarrow \mathcal{C}$ and sends it to $\mathcal{P}$; the prover sends a response z to $\mathcal{V}$; the verifier accepts or rejects depending on the transcript (ω, c, z). The transcript (ω, c, z) is called accepting if the verifier accepts the protocol run.

Completeness: Whenever $(x, w) \in \mathfrak{R}$, the honest verifier accepts with probability at least $1 - \delta$ when interacting with an honest prover. We call the protocol has a completeness with error δ.

k-**special soundness:** There exists a PPT algorithm $\mathcal{E}$ (called the extractor) which takes k accepting transcripts $(\omega, c_1, z_1), (\omega, c_2, z_2), \ldots, (\omega, c_k, z_k)$ as inputs. Here, $c_1, c_2, \ldots c_k$ are pairwise distinct. The algorithm outputs w' satisfying $(x, w') \in \mathfrak{R}'$. We call this procedure witness extraction, and say that the protocol has k-special soundness.

Special honest-verifier zero-knowledge: There exists a PPT algorithm $\mathcal{S}$ (called the simulator) that takes $x \in \mathcal{L}$ and $c \in \mathcal{C}$ as inputs, and outputs (ω, z) such that (ω, c, z) is indistinguishable from an accepting transcript generated by a real protocol run.

The t-out-of-n proof protocol is clearly associated with the underlying commitment scheme being used. In order to make it compatible with lattice-based schemes, we define two relations $\mathfrak{R}$ and $\mathfrak{R}'$ for the t-out-of-n proof. However, in some settings such as the DL setting, these two relations $\mathfrak{R}$ and $\mathfrak{R}'$ are identical.

Definition 23 Let S_r' and S_{fac} be the sets used in Definition 18, t, n be positive numbers, we define the following relation:

$$\mathfrak{R} = \{(P, t, ck, com), (\vec{v}, r) \mid \vec{v} \in \{0,1\}^n \wedge \|\vec{v}\|_1 = t \wedge \ \mathsf{Open}_{ck}(com, r, (P\vec{v}, \vec{v})) = 1\},$$

where the statement is (P, t, ck, com) and the witness is $(\vec{v}, r)$;
And we give the following relaxed relation:

$$\mathfrak{R}' = \left\{ (P, t, ck, com), (\vec{v}, r, \tau) \middle| \begin{array}{l} \vec{v} \in \{0,1\}^n \wedge \|\vec{v}\|_1 = t \wedge \tau \in S_{fac} \wedge \tau r \in S_r' \\ \wedge\ com = \mathsf{Commit}_{ck}((P\vec{v}, \vec{v}); r) \end{array} \right\},$$

where the statement is (P, t, ck, com) and the relaxed witness is $(\vec{v}, r, \tau)$.

Remark 4 The conditions on the randomness in the relations $\mathfrak{R}$ and $\mathfrak{R}'$ play a crucial role in the lattice-based setting, which is one of the main differences between a lattice-based protocol and its DL setting counterpart. Without these conditions, it may not be possible to establish the security of the efficient protocol using the lattice problem.

We present the framework of the t-out-of-n proof in Algorithm 3. Specifically, the proof process $\mathsf{TN.P} = (\mathsf{TN.P}_1, \mathsf{TN.P}_2)$ is split into two functions. Let $x = (P, t, ck, com)$ be the statement. Function $\mathsf{TN.P}_1$ takes input $(x, (\vec{v}, r))$ and returns a ω and a state st. Then, the verifier samples a challenge c from the distribution $D(\mathsf{CSet})$ over the set CSet. Finally, function $\mathsf{TN.P}_2$ takes input $(x, (\vec{v}, r), st, c)$ and returns a response $z \in \mathsf{ZSet}$. In the verification stage, the deterministic function Ver takes (x, ω, c, z) as input and outputs a decision, either 1 (acceptance) or 0 (rejection). We also add an extra check to support lattice-based constructions.

A secure t-out-of-n proof protocol should provide completeness and k-special soundness presented above. Besides, we also require t-out-of-n proof protocol satisfies one-time zero-knowledge defined as follows. This property guarantees that if an honest prover computes only one proof for any statement, then it is zero-knowledge.

Definition 24 (*One-time Zero-knowledge*) There exists a PPT algorithm $\mathcal{S}$ (called the simulator) that takes $x \in \mathcal{L}$ and $c \in \mathcal{C}$ as inputs, and outputs (ω, z) such that (ω, c, z) is indistinguishable from an accepting transcript generated by a real protocol run.

Algorithm 3 t-out-of-n proof

Prove Stage:
1: Set flag
2: $(\omega, st) \leftarrow \mathsf{TN.P}_1(x, (\vec{v}, r))$
3: $c \leftarrow D(\mathsf{CSet})$
4: $z \leftarrow \mathsf{TN.P}_2(x, (\vec{v}, r), st, c)$
5: If flag $=$ True, run extra verification, and set $z = \perp$ if failed

Verify Stage:
1: Check if $z \in \mathsf{ZSet}$
2: Check if $\mathsf{Ver}(x, \omega, c, z) = 1$
3: If failed **return** 0, otherwise **return** 1

Remark 5 We only require the protocol to satisfy one-time zero-knowledge is because it is sufficient for the security requirement in our TRS as we only use the commitment scheme in an auxiliary way to assist in proving that the value we concerned about satisfies certain relations. Hence the statement never needs to be reused. Furthermore, this relaxed security requirement allows us to reuse 'r' as randomness when we need to compute intermediate commitments in the t-out-of-n proof protocol, thereby reducing the proof size.

In Sect. 5.1, we will propose a linear lattice-based t-out-of-n proof protocol. And in Sect. 5.2, we present a logarithmic lattice-based t-out-of-n proof protocol.

3.2 Generic construction

Now we describe a generic construction for the TRS. Let HI denote a homomorphic canonical identification scheme, THC denote a trapdoor homomorphic commitment scheme and TN denote a t-out-of-n proof protocol. In the construction we apply the Fiat-Shamir [26] transformation to both HI and TN to convert them into a non-interactive form. We let $R = \{pk_1, \ldots, pk_n\}$ denote a set of public keys and $I \subset [n]$ denote the indicator set of signers. Let $\vec{v} \in \{0, 1\}^n$ be the indicator vector of I. More precisely, $\vec{v} = \sum_{i \in I} \vec{e}_i$, where $\vec{e}_i \in \{0, 1\}^n$ has exactly one 1 in the ith coefficient.

Our generic construction (GC-TRS) is described in Alg.4. First, we use the homomorphic property of the canonical identification scheme to compress and generate an aggregate signature. To enable the verifier to check the signature without knowing the public keys related to the aggregation signature, the signers generate a t-out-of-n proof. In the verify phase, the verifier only needs to check the validity of t-out-of-n proof. Unlike in previous works such as [13] and [29], our construction has no leader, and all signers share the same role. Therefore, we only describe the behavior of the ith signer. As our Sign algorithm is a two-round interaction protocol, we divide it into three stages for readability. We also require three random oracles, denoted as H_0, H_1, H_2. Specifically, the output of H_0 is an element in S_{ck}, and the outputs of H_1 and H_2 follow the distributions $D(S_c)$ and $D(\mathsf{CSet})$, respectively.

Remark 6 Strictly speaking, the parameters output by the Setup algorithm also depends on the values of t and n. For simplicity, we have omitted them here. The Setup algorithm must be executed honestly and not controlled by an adversary. This can be achieved by using a trusted third party or secure multi-party computation, which depend on specific scenarios and instantiations. And this issue is outside the scope of this paper.

Remark 7 In some constructions (including our later instantiation), flag is set to be True. At this time, the scheme has a probability of aborting. Fortunately, our scheme is suited for parallel executions. Thus, one can alternatively run sufficiently many parallel executions of the scheme at once to obtain a valid signature.

Algorithm 4 Generic Construction (GC-TRS)

$\underline{\mathsf{Setup}(1^\lambda)}$:
1: Set flag
2: $pp_1 \leftarrow \mathsf{HI.Setup}(1^\lambda)$
3: $pp_2 \leftarrow \mathsf{THC.Setup}(1^\lambda)$
4: Define H_0, H_1, H_2
5: **return** $(\mathsf{flag}, pp_1, pp_2, H_0, H_1, H_2)$

$\underline{\mathsf{KeyGen}(pp)}$:
1: $(pk, sk) \leftarrow \mathsf{HI.KeyGen}(pp_1)$
2: **return** (pk, sk)

$\underline{\mathsf{Sign}(msg, R = \{pk_1, \ldots, pk_n\}, I, sk_i)}$:

Phase 1:
1: $y_i \leftarrow D(S_y)$, $w_i = \mathsf{HI.A}(sk_i; y_i)$
2: $ck \leftarrow H_0(msg, R)$, $r_i \leftarrow D(S_r)$
3: $com_i = \mathsf{THC.Commit}_{ck}((w_i, \vec{e}_i); r_i)$
4: Send com_i to other signers

Phase 2: After receiving all signers' commitments, do:
5: $com = \sum_{i \in I} com_i$, $c_1 = H_1(ck, com)$
6: $z_i = \mathsf{HI.Z}(sk_i, y_i, c_1)$
7: If $\mathsf{flag} = \mathsf{True}$, run extra verification on (y_i, c_1, z_i) and set $z_i = \perp$ if failed
8: Send (z_i, r_i) to other signers

Phase 3: After receiving all signers' message, do:
9: If exists $j \in I$ and $z_j \notin S_z$, abort
10: $z = \sum_{i \in I} z_i$, $r = \sum_{i \in I} r_i$
11: $P = \mathsf{HI.V}_1(c_1, z)R$
12: $com' = com - (\mathsf{HI.V}_2(c_1, z), 0)$
13: $x = (P, t, ck, com')$
14: $(\omega, st) \leftarrow \mathsf{TN.P}_1(x, (\vec{v}, r))$
15: $c_2 = H_2(x, \omega)$
16: $z' \leftarrow \mathsf{TN.P}_2(x, (\vec{v}, r), st, c_2)$
17: If $\mathsf{flag} = \mathsf{True}$, run extra verification, and set $z' = \perp$ if failed
18: **return** $\Sigma = (com, z, \omega, z')$

$\underline{\mathsf{Verify}(msg, R = \{pk_1, \ldots, pk_n\}, \Sigma, t)}$:
1: Parse $\Sigma = (com, z, \omega, z')$
2: $ck \leftarrow H_0(msg, R)$, $c_1 = H_1(ck, com)$
3: $P = \mathsf{HI.V}_1(c_1, z)R$
4: $com' = com - (\mathsf{HI.V}_2(c_1, z), 0)$
5: $x = (P, t, ck, com')$, $c_2 = H_2(x, \omega)$
6: Check if $\mathsf{TN.Ver}(x, \omega, c_2, z') = 1$
7: Check if $z \in S_z$, $z' \in \mathsf{ZSet}$
8: If failed **return** 0, otherwise **return** 1

More specially, in each single execution of our signature protocol, each signer needs to send a commitment *com* in the 1st round and a pair of message (z, r) in the 2nd round, where the asymptotic size of these transcripts is $O(t \log(n))$ in CTRS. The number of repetitions (denoted by τ, which is around 20 in our instantiation) is then required to guarantee the generation of a valid signature. Consequently, the transcript complexity is $O(\tau t \log(n))$ and round complexity is $O(2\tau)$. Under parallelism, the communication complexity will remain the same but round complexity reduces to 2 (as repetition is not needed).

3.3 Security

We present the correctness, unforgeability and anonymity claims for GC-TRS. Note, we just give the security proof of GC-TRS in the ROM. We leave for future work a tighter security proof, as well as a proof in the QROM.

Theorem 1 (Correctness) *Suppose* HI *scheme has t-correctness with error* δ_1 *and linear properties,* THC *scheme has t-additive homomorphism and message homomorphism properties, and* TN *protocol has completeness with error* δ_2 *property, then our* GC-TRS *provides correctness. On average, the construction needs to be repeated* $\frac{1}{(1-\delta_1)^t(1-\delta_2)}$ *times to generate a valid signature.*

Proof Assuming the public key list $R = \{pk_1, \ldots, pk_n\}$ are generated by KeyGen, the signature $\Sigma = (com, z, \omega, z')$ is generated by the honest signers in I. We now show it holds $\mathsf{Verify}(msg, R, \Sigma, t) = 1$. First, by t-additive homomorphism property of THC, except for negligible probability we have

$$\mathsf{THC.Open}_{ck}(com, r, \left(\sum_{i \in I} w_i, \vec{v}\right)) = 1.$$

And by t-correctness and linear property of HI, we have

$$\sum_{i\in I} w_i = \mathsf{HI.V}_1(c_1, z) \sum_{i\in I} pk_i + \mathsf{HI.V}_2(c_1, z).$$

Then, by message homomorphism property of THC, it holds

$$\mathsf{THC.Open}_{ck}(com', r, (\mathsf{HI.V}_1(c_1, z) \cdot \sum_{i\in I} pk_i, \vec{v})) = 1,$$

where $com' = com - (\mathsf{HI.V}_2(c_1, z), 0)$. Let $\vec{v}$ be the indicator vector of I and $P = \mathsf{HI.V}_1(c_1, z)R$, then we have $(x, (\vec{v}, r)) \in \mathfrak{R}$, where $x = (P, t, ck, com')$. Thus, by completeness of TN, we have $\mathsf{Verify}(msg, R, \Sigma, t) = 1$. Therefore, the GC-TRS satisfies the correctness.

Now, we compute the required repetition times for $\mathsf{GC\text{-}TRS}$ to generate a valid signature. Since the HI scheme has correctness error δ_1, every honest signer in I has a success probability of at least $1 - \delta_1$ in the $\mathsf{Phase\ 2}$ ($\mathsf{Step\ 7}$). Thus, the probability that all signers succeed in the $\mathsf{Phase\ 2}$ is at least $(1 - \delta_1)^t$. And since the TN protocol has correctness error δ_2, the probability of success in the $\mathsf{Phase\ 3}$ ($\mathsf{Step\ 17}$) is at least $1 - \delta_2$. Hence, the construction should repeat $\frac{1}{(1-\delta_1)^t(1-\delta_2)}$ times on average to output a valid signature. □

Theorem 2 (t-unforgeability) *Suppose* HI *scheme has the strong honest-verifier zero-knowledge and secure against* $t - 1$ *corruption attack properties,* THC *scheme has the equivocability, uniform key, message homomorphism and binding for weak opening properties, and* TN *protocol has k-special soundness property. For any probabilistic polynomial-time adversary* $\mathcal{A}$ *that initiates* Q_s *signature generation protocols by querying signing oracle, and makes* Q_h *queries to the random oracle* H_0, H_1, H_2*, our* $\mathsf{GC\text{-}TRS}$ *has t-unforgeability w.r.t. insider corruption in the* ROM*. Concretely, the advantage of* $\mathcal{A}$ *is bounded as follows.*

$$\mathsf{Adv}^{\mathsf{t-unforgeability}}_{\mathsf{GC\text{-}TRS}}(\mathcal{A}) \le e(Q_h + Q_s + 1)\,[(Q_h + Q_s) \cdot \epsilon_{\mathrm{td}} + Q_s \cdot \epsilon_{\mathrm{SHVZK}} + (1 - \kappa)\epsilon_{\mathrm{IH}} + (Q_h + Q_s + 1)\kappa\,],$$

where $\kappa = 1 - (1 - \frac{1}{|S_c|})(1 - \frac{k}{|\mathsf{CSet}|})$.

Proof Denote $\mathcal{A}$ as a PPT adversary breaking the t-unforgeability game of the $\mathsf{GC\text{-}TRS}$. We are going to build an algorithm to break the *security against* $t - 1$ *corruption attack* of the HI scheme. Our proof is divided into the following three steps: we first construct the algorithm $\mathcal{B}$ around $\mathcal{A}$ that simulates the honest signers' behaviors without using their secret keys, then we use extractor algorithm $\mathcal{E}$ around $\mathcal{B}$ to obtain a set of tree structure transcripts, and finally we use these transcripts break the HI scheme. □

3.3.1 Step 1

Construct $\mathcal{B}$ around $\mathcal{A}$ that simulates the behaviors of honest signers without using secret keys.

G_0 This is the real t-unforgeability game. In this game, $\mathcal{B}$ first generates a list public-secret key pairs $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp)$ and sends $L = \{pk_i\}_{i\in q_{key}}$ to $\mathcal{A}$. And $\mathcal{B}$ responds to oracle queries as follows: **Random oracle simulation.** The table HT_i is initially empty.
$H_0(x)$: Given an x as input, $\mathcal{B}$ checks if $\mathsf{HT}_0(x) = \perp$. If $\mathsf{HT}_0(x) = \perp$, $\mathcal{B}$ samples $ck \leftarrow S_{ck}$ and sets $\mathrm{HT}_0(x) = ck$. Finally, $\mathcal{B}$ returns $\mathsf{HT}_0(x)$.
$H_1(x)$: Given an x as input, $\mathcal{B}$ checks if $\mathsf{HT}_1(x) = \perp$. If $\mathsf{HT}_1(x) = \perp$, $\mathcal{B}$ samples a

challenge $c \leftarrow D(S_c)$ and sets $\mathsf{HT}_1(x) = c$. Finally, $\mathcal{B}$ returns $\mathsf{HT}_1(x)$.
$H_2(x)$: Given an x as input, $\mathcal{B}$ checks if $\mathsf{HT}_2(x) = \perp$. If $\mathsf{HT}_2(x) = \perp$, $\mathcal{B}$ samples a challenge $c \leftarrow D(\mathsf{CSet})$ and sets $\mathsf{HT}_2(x) = c$. Finally, $\mathcal{B}$ returns $\mathsf{HT}_2(x)$.
Signing oracle simulation. In this game $\mathcal{B}$ behaves exactly like honest signers in GC-TRS. Specifically, on input a message msg, a set of public keys R, and a list of signers $I \subset R$, $\mathcal{B}$ first checks whether the public keys in I are generated by himself. If not, $\mathcal{B}$ returns $\perp$. Otherwise, $\mathcal{B}$ decomposes I to $I = I_{corr} \sqcup I_{hon}$, where I_{corr} denotes corrupted signers and I_{hon} denotes honest signers and $\sqcup$ represents the disjoint union. To produce a signature, $\mathcal{A}$ needs to cooperate with $\mathcal{B}$ and participate in the signing procedure. $\mathcal{B}$ simulates the behavior of users in I_{hon} by using their secret key according to the GC-TRS scheme. Then, $\mathcal{B}$ outputs a signature Σ. Finally, $\mathcal{B}$ adds (msg, R, I) to the query record set $\mathcal{Q}_{SO}$, which is initialized as an empty list.
Corruption oracle simulation. On input a public key pk, the challenger $\mathcal{B}$ first checks whether the public key is generated by himself. If so, $\mathcal{B}$ returns the corresponding secret key sk; otherwise returns $\perp$. Then $\mathcal{B}$ adds pk to the query record set $\mathcal{Q}_{CO}$, which is initialized as an empty list.
Forgery. When $\mathcal{A}$ output a forgery $(msg, R, \Sigma = (com, z, \omega, z'))$ at the end $\mathcal{B}$ proceeds as follows:

1. If there exists public key in R that does not generated by $\mathcal{B}$ or $|R \cap \mathcal{Q}_{CO}| \geq t$ or $(msg, R, \cdot) \in \mathcal{Q}_{SO}$, then $\mathcal{B}$ halts with output $(0, \perp)$.
2. If the output signature cannot pass verify, then $\mathcal{B}$ halts with output $(0, \perp)$.
3. $\mathcal{B}$ halts with output $(1, (msg, R, \Sigma = (com, z, \omega, z')))$.

Let $\Pr[G_i]$ denote a probability that $\mathcal{B}$ does not output $(0, \perp)$ at the game G_i. Then we have

$$\Pr[G_0] = \mathsf{Adv}^{\mathsf{t-unforgeability}}_{\mathsf{GC\text{-}TRS}}(\mathcal{A}).$$

G_1 This game is identical to G_0 except at the following points.
Random oracle simulation. Simulation of the random oracle H_0 is changed as follows: $\mathcal{B}$ first initializes two empty lists HT_0 and TDT. Given an x as input, $\mathcal{B}$ checks if $\mathsf{HT}_0(x) = \perp$. If $\mathrm{HT}_0(x) = \perp$, $\mathcal{B}$ samples a biased bit that returns 0 with probability θ and returns 1 with probability $1 - \theta$. When output is 0, $\mathcal{B}$ computes $(tck, td) \leftarrow \mathsf{THC.TGen}(pp_2)$, stores the trapdoor in $\mathrm{TDT}(x) = td$ and sets $\mathsf{HT}_0(x) = tck$; when output is 1, $\mathcal{B}$ samples $ck \leftarrow S_{ck}$ and sets $\mathrm{HT}_0(x) = ck$. Finally, $\mathcal{B}$ returns $\mathsf{HT}_0(x)$.
Signing oracle simulation. The $\mathcal{B}$ differs from the G_0 at the following steps in GC-TRS:
− **Step 2 in Phase 1**: Call $H_0(msg, R)$ to obtain tck. If $\mathrm{TDT}(msg, R) = \perp$ (i.e., ck is not generated by THC.TGen), then $\mathcal{B}$ sets bad_1 and halt with output $\perp$. Otherwise, it obtains the corresponding trapdoor $td \leftarrow \mathrm{TDT}(msg, R)$.
− **Step 3 in Phase 1**: Call $com_i \leftarrow \mathsf{THC.TCommit}_{ck}(td)$ instead of committing to $(w_i, \vec{e}_i)$.
− **Step 6 in Phase 2**: After computing $z_i = \mathsf{HI.Z}(sk_i, y_i, c_i)$ derive randomness $r_i \leftarrow \mathsf{THC.Eqv}_{ck}(td,$
$com_i, (\mathsf{HI.V}(pk_i, c_1, z_i), \vec{e}_i))$.
Forgery. When $\mathcal{A}$ outputs a successful forgery $(msg, R, \Sigma = (com, z, \omega, z'))$ at the end, we modify the step 2 of G_0 as follows: If the output signature cannot pass verify then $\mathcal{B}$ halts with output $(0, \perp)$; and if $\mathrm{TDT}(msg, R) \neq \perp$ (i.e., TGen was called for a query $H_0(msg, R)$) then set bad_2 and $\mathcal{B}$ halts with output $(0, \perp)$.
Note that due to the way H_0 is simulated, if $\mathcal{B}$ does not output $(0, \perp)$, it is now guaranteed that $ck = H_0(msg, R)$. Because the simulation is only successful if the random oracle H_0 internally uses TGen for all but one queries to H_0 (both directly and indirectly via

random oracle queries and signing oracle queries) and if H_0 uses a predefined ck for a single query (msg, R) associated with forgery. In other words, it is only successful if neither bad_1 nor bad_2 is set. Since THC satisfies the equivocability and uniform key properties, we have

$$\Pr[G_1] \geq \theta^{Q_h+Q_s} \cdot (1-\theta) \cdot \Pr[G_0] - (Q_h + Q_s) \cdot \epsilon_{\text{td}}.$$

By setting $\theta = (Q_h + Q_s)/(Q_h + Q_s + 1)$, we obtain

$$\Pr[G_1] \geq \frac{\Pr[G_0]}{e(Q_h + Q_s + 1)} - (Q_h + Q_s) \cdot \epsilon_{\text{td}}.$$

G_2 This game is identical to G_1 except at the following points.
Signing oracle simulation. The $\mathcal{B}$ does not honestly generate z_i anymore and instead simulates as follows:
− **Step 2 in Phase 1**: $\mathcal{B}$ does nothing here.
− **Step 6 in Phase 2**: Sample $z_i \leftarrow D(S_z)$ and gets randomness $r_i \leftarrow \mathsf{THC.Eqv}_{ck}(td, com_i, (\mathsf{HI.V}(pk_i, c_1, z_i), \vec{e}_i))$.
− **Step 7 in Phase 2**: If $\mathsf{flag} = \mathsf{True}$, $\mathcal{B}$ performs an extra verification on transcript, and if the verification failed set $z_i = \perp$.
The partial signature z_i simulated in this way is statistically indistinguishable from the real one because of strong honest-verifier zero-knowledge property of the underlying HI. Hence we have

$$\Pr[G_2] \geq \Pr[G_1] - Q_s \cdot \epsilon_{\text{SHVZK}}.$$

G_3 Note that, the response of signing queries in the G_2 does not rely on the signers' secret key. In this game, our goal is to change the public key set L as the inputs obtained by $\mathcal{B}$ from the $t-1$ corruption game of the HI scheme. Thus in this game after $\mathcal{B}$ receives a public key list $L = \{pk_i\}_{i \in q_{key}}$, it will sends L to $\mathcal{A}$.
Corruption oracle simulation. In this case, as $\mathcal{B}$ no longer possesses the secret keys, when $\mathcal{A}$ makes corruption oracle queries, $\mathcal{B}$ need to query its own corruption oracle to obtain answer. If $\mathcal{B}$ obtains $\perp$, $\mathcal{B}$ will also return $\perp$ to $\mathcal{A}$. If $\mathcal{B}$ obtains the corresponding secret key sk form its own oracle, $\mathcal{B}$ will return it to $\mathcal{A}$ and add pk to the query record set $\mathcal{Q}_{CO}$, which is initialized as an empty list.
Because the KeyGen of GC-TRS is the same as the KeyGen of HI, we have

$$\Pr[G_2] = \Pr[G_3].$$

3.3.2 Step 2

Construct extractor algorithm $\mathcal{E}$ around $\mathcal{B}$ to obtain a set of tree structure transcripts.

This extractor algorithm is analogous to that of Attema et al. [4]. Because our GC-TRS is essentially a multi-round protocol (specifically a 5-round protocol, using the Fiat-Shamir transformation twice), the core idea is to rewind $\mathcal{B}$ in G_3 multiple times in a layered manner, thereby generating a tree of transcripts. See Fig. 1 for a graphical illustration. Here, $\{c_1^i\}$ and $\{c_2^{i,j}\}$ correspond to c_1 and c_2 in the fifth and fifteenth lines of our GC-TRS, respectively. And transcripts a_1, $\{a_2^i\}$ and $\{a_3^{i,j}\}$ correspond to (ck, com), (x, ω) and z' of our scheme respectively. Note that, when $\mathcal{B}$ output $\perp$, we can obtain this transcript. Now we present our extractor algorithm $\mathcal{E}$ as follows:
Black-box access to: $\mathcal{B}$ in G_3
Random oracle queries: $Q_h + Q_s$

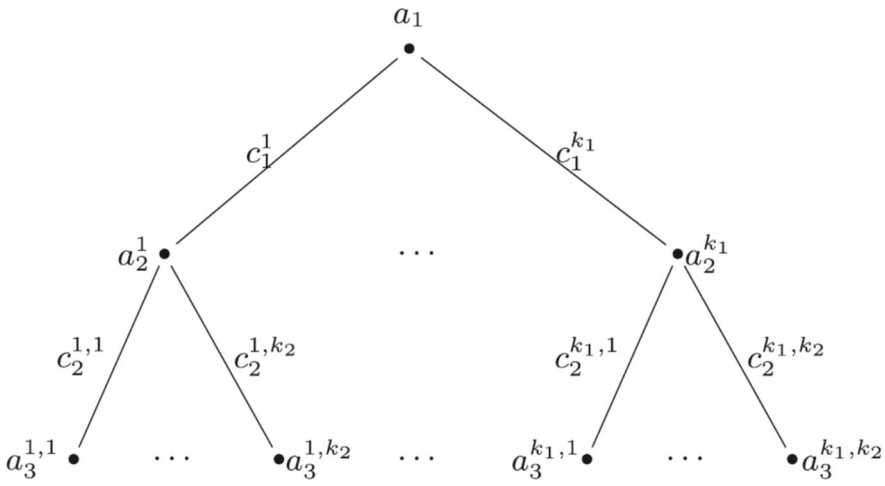

Fig. 1 Tree of transcripts

- Run $\mathcal{B}$ to obtain $(st, trans_1^1 = (a_1, a_2, a_3))$: relay the $Q_h + Q_s$ queries to the random oracle and record all query-response pairs. Let c_2 be the response to query (a_1, a_2). If $st = 0$, abort with output $st = 0$, else output transcript $trans_1^1$.
- Repeat the following process:
  - run $\mathcal{B}$ to obtain $(st', trans_1' = (a_1', a_2', a_3'))$, aborting right after the initial run of $\mathcal{B}$ if $(a_1, a_2) \neq (a_1', a_2')$: answer the query to (a_1, a_2) with a fresh random value c_2' in CSet, while answering all other queries consistently;

  until either $k-1$ additional c_2' with $st_1' = 1$ and $(a_1, a_2) = (a_1', a_2')$ have been found or until all challenges $c_2' \in \mathsf{CSet}$ have been tried. In the former case, output the $k-1$ accepting transcript $trans_1^2, \ldots, trans_1^k$, and set $st = 1$; in the latter case, abort with output $st = 0$.
- Repeat the following process:
  - run $\mathcal{B}$ to obtain $(st_2, trans_2^1 = (a_1^*, a_2^*, a_3^*))$, aborting right after the initial run of $\mathcal{B}$ if $a_1 \neq a_1^*$: answer the query to a_1 with a fresh random value c_1' in S_c, while answering all other queries consistently;

  until either an additional c_1' with $st_2' = 1$ and $a_1 = a_1^*$ have been found or until all challenges $c_1' \in S_c$ have been tried. In the former case, output the the accepting transcript $trans_2^1$, and set $st = 1$; in the latter case, abort with output $st = 0$.
- Repeat the following process:
  - run $\mathcal{B}$ to obtain $(st_2', trans_2^\# = (a_1^\#, a_2^\#, a_3^\#))$, aborting right after the initial run of $\mathcal{B}$ if $(a_1, a_2^*) \neq (a_1^\#, a_2^\#)$: answer the query to (a_1, a_2^*) with a fresh random value $c_2^\#$ in CSet, while answering all other queries consistently;

  until either $k-1$ additional $c_2^\#$ with $st_2' = 1$ and $(a_1, a_2^*) = (a_1^\#, a_2^\#)$ have been found or until all challenges $c_2^\# \in \mathsf{CSet}$ have been tried. In the former case, output the $k-1$ accepting transcript $trans_2^2, \ldots, trans_2^k$, and set $st = 1$; in the latter case, abort with output $st = 0$.

According to the analysis in [4], we can get the following lemma.

Lemma 9 (Adapted Proposition 2 in [4]) *The extractor $\mathcal{E}$ makes an expected number of at most $2k + (Q_h + Q_s) \cdot (2k - 1)$ queries to $\mathcal{B}$(and thus to $\mathcal{A}$) and successfully outputs $st = 1$ with probability at least*

$$\frac{\Pr[G_3] - (Q_h + Q_s + 1) \cdot \kappa}{1 - \kappa},$$

where $\kappa = 1 - (1 - \frac{1}{|S_c|})(1 - \frac{k}{|\mathsf{CSet}|})$.

3.3.3 Step 3

Break the HI scheme by using extracted transcripts.

Note that the transcripts extracted by $\mathcal{E}$ can be presented in the following form: $\{c_1^1, c_2^{1,1}, \Sigma_1^1 = (com, z_1^1, \omega^1, z_2^{1,1})\}, \ldots, \{c_1^1, c_2^{1,k}, \Sigma_1^k = (com, z_1^1, \omega^1, z_2^{1,k})\}$ and $\{c_1^2, c_2^{2,1}, \Sigma_2^1 = (com, z_1^2, \omega^2, z_2^{2,1})\}, \ldots, \{c_1^2, c_2^{2,k}, \Sigma_2^k = (com, z_1^2, \omega^2, z_2^{2,k})\}$. Let $x = (P, t, ck, com')$, where $ck = H_0(msg, R)$, $P = \mathsf{HI.V}_1(c_1^1, z_1^1)R$ and $com' = com - (\mathsf{HI.V}_2(c_1^1, z_1^1), 0)$. Since $\Sigma_1^1, \Sigma_1^2, \ldots, \Sigma_1^k$ are valid signatures, we have that $(\omega, c_2^{1,1}, z_2^{1,1})$, $\ldots$, $(\omega, c_2^{1,k}, z_2^{1,k})$ are k accepted transcripts of TN protocol. Thus, we can extract a witness $(\vec{v}, r, \tau)$ for the statement x of the relaxed relation $\mathfrak{R}'$ by k-special soundness of TN. Hence, we have $com' = \mathsf{THC.Commit}_{ck}((P\vec{v}, \vec{v}); r)$ by the definition of relation $\mathfrak{R}'$. Let $y = \mathsf{HI.V}_1(c_1^1, z_1^1)R\vec{v} + \mathsf{HI.V}_2(c_1^1, z_1^1)$, by message homomorphism of THC, we have that $(\tau, r, (y, \vec{v}))$ is a weak opening for com. Similarly, let $x^* = (P^*, t, ck, com^*)$, where $P^* = \mathsf{HI.V}_1(c_1^2, z_1^2)R$ and $com^* = com - (\mathsf{HI.V}_2(c_1^2, z_1^2), 0)$. Thus we can extract another witness $(\vec{v}^*, r^*, \tau^*)$ for the statement x^* of the relaxed relation $\mathfrak{R}'$ by using the left transcripts. Thus, let $y^* = \mathrm{HI}.V_1(c_1^2, z_1^2)R\vec{v}^* + \mathrm{HI}.V_2(c_1^2, z_1^2)$, by message homomorphism of THC, $(\tau^*, r^*, (y^*, \vec{v}^*))$ is another weak opening for com.

Hence, by binding for weak opening property of THC, we have $\vec{v} = \vec{v}^*$ and $y = y^*$. Therefore $\mathcal{B}$ can return $(z_1^1, c_1^1, z_1^2, c_1^2, y, I)$ to its challenger $\mathcal{C}$, where I is the signers' public keys (i.e., when $v_i = 1$, $pk_i \in I$, otherwise $pk_i \notin I$, where v_i is the ith coefficient of $\vec{v}$). As a result, we can break the security of HI, which is contradictory. □

Theorem 3 (t-Anonymity) *Suppose* HI *scheme has the strong honest-verifier zero-knowledge,* THC *scheme has the equivocability, uniform key, and* TN *protocol has one-time zero-knowledge. For any probabilistic polynomial-time adversary $\mathcal{A}$ that initiates Q_s signature generation protocols by querying signing oracle, and makes Q_h queries to the random oracle H_0, H_1, H_2, our* GC-TRS *has t-anonymity w.r.t. adversarial keys in the* ROM. *Concretely, the advantage of $\mathcal{A}$ is bounded as follows.*

$$\mathsf{Adv}^{\mathsf{t-anonymity}}_{\mathsf{GC\text{-}TRS}}(\mathcal{A}) \leq (\epsilon^{\mathsf{t-hiding}}_{\mathsf{THC}} + \mathsf{negl}(\lambda)) \cdot (Q_h + Q_s).$$

Proof Denote $\mathcal{A}$ as a PPT adversary breaking the t-anonymity game of GC-TRS, we are going to build an algorithm $\mathcal{B}$ to break the t-hiding game[5] of the THC scheme. We realize this through the following several intermediate games.

G_0 This is the real t-anonymity game. In this game, $\mathcal{B}$ first generates a list public-secret key pairs $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp)$ and sends $L = \{pk_i\}_{i \in q_{key}}$ to $\mathcal{A}$. And $\mathcal{B}$ responds to oracle queries as follows. In fact, it it exactly the same as that described in G_0 of t-unforgeability.

Random oracle simulation. The table HT_i is initially empty.

[5] Since equivocability implies t-hiding property, for the sake of brevity, we have not included it in the theorem.

$H_0(x)$: Given an x as input, $\mathcal{B}$ checks if $\mathsf{HT}_0(x) = \perp$. If $\mathsf{HT}_0(x) = \perp$, $\mathcal{B}$ samples $ck \leftarrow S_{ck}$ and sets $\mathrm{HT}_0(x) = ck$. Finally, $\mathcal{B}$ returns $\mathsf{HT}_0(x)$.

$H_1(x)$: Given an x as input, $\mathcal{B}$ checks if $\mathsf{HT}_1(x) = \perp$. If $\mathsf{HT}_1(x) = \perp$, $\mathcal{B}$ samples a challenge $c \leftarrow D(S_c)$ and sets $\mathsf{HT}_1(x) = c$. Finally, $\mathcal{B}$ returns $\mathsf{HT}_1(x)$.

$H_2(x)$: Given an x as input, $\mathcal{B}$ checks if $\mathsf{HT}_2(x) = \perp$. If $\mathsf{HT}_2(x) = \perp$, $\mathcal{B}$ samples a challenge $c \leftarrow D(\mathsf{CSet})$ and sets $\mathsf{HT}_2(x) = c$. Finally, $\mathcal{B}$ returns $\mathsf{HT}_2(x)$.

Signing oracle simulation. In this game $\mathcal{B}$ behaves exactly like honest signers in GC-TRS. Specifically, on input a message msg, a set of public keys R, and a list of signers $I \subset R$, $\mathcal{B}$ first checks whether the public keys in I are generated by himself. If not, $\mathcal{B}$ returns $\perp$. Otherwise, $\mathcal{B}$ decomposes I to $I = I_{corr} \sqcup I_{hon}$, where I_{corr} denotes corrupted signers and I_{hon} denotes honest signers and $\sqcup$ represents the disjoint union. To produce a signature, $\mathcal{A}$ needs to cooperate with $\mathcal{B}$ and participate in the signing procedure. $\mathcal{B}$ simulates the behavior of users in I_{hon} by using their secret key according to the GC-TRS scheme. Then, $\mathcal{B}$ outputs a signature Σ. Finally, $\mathcal{B}$ adds (msg, R, I) to the query record set $\mathcal{Q}_{SO}$, which is initialized as an empty list.

Corruption oracle simulation. On input a public key pk, the challenger $\mathcal{B}$ first checks whether the public key is generated by himself. If so, $\mathcal{B}$ returns the corresponding secret key sk; otherwise returns $\perp$. Then $\mathcal{B}$ adds pk to the query record set $\mathcal{Q}_{CO}$, which is initialized as an empty list.

Challenge Stage. $\mathcal{A}_1$ gives $\mathcal{B}$ a message msg^*, a public list R^* and two signer index sets I_0, I_1, with $|I_0| = |I_1| = t$. Then $\mathcal{B}$ sample a random bit $b \leftarrow \{0, 1\}$ and honestly invokes GC-TRS to compute $\Sigma^* \leftarrow \mathsf{Sign}(msg^*, I_b^* R^*)$. Next, Σ^* will be sent to $\mathcal{A}$.

Guess Stage. Finally $\mathcal{A}_2$ outputs a bit b'. Let $\Pr[G_0]$ denote a probability that $b = b'$. Then we have

$$\mathsf{Adv}^{\mathsf{t-anonymity}}_{\mathsf{GC\text{-}TRS}}(\mathcal{A}) = |\Pr[G_0] - \frac{1}{2}|.$$

G_1 In this game, our goal is to change the generation of challenge signature. We make it becomes the challenge message obtained by $\mathcal{B}$ from the t-hiding game of the THC scheme. In this game, $\mathcal{B}$ will first receive a commitment key ck from its challenger $\mathcal{C}$.

Random oracle simulation. Simulation of the random oracle H_0 is identical to G_0 except at the following point: $\mathcal{B}$ first sample an index $i^* \leftarrow [Q_h + Q_s]$. When $\mathcal{A}$ makes a query to H_0 for the i^*th time (directly and indirectly via random oracle queries and signing oracle queries), we set its value to be ck.

Challenge Stage. $\mathcal{A}_1$ gives $\mathcal{B}$ a message msg^*, a public list R^* and two signer index sets I_0, I_1, with $|I_0| = |I_1| = t$. $\mathcal{B}$ generates challenge signature as follows:

– If $H_0(msg^*, R^*) \neq ck$, then $\mathcal{B}$ sets bad_3 and halt with output a random bit.

– $\mathcal{B}$ first samples $c_1 \leftarrow D(S_c)$ and samples $z_1, \ldots, z_t \leftarrow D(S_z)$.

– For $j \in [t]$, $\mathcal{B}$ computes $w_j^0 = \mathsf{HI.V}(pk_{i_j}, c_1, z_j)$ where $i_j \in I_0$ and $w_j^1 = \mathsf{HI.V}(pk_{i_j}, c_1, z_i)$ where $i_j \in I_1$.

– Let $\vec{e}_j^0, \vec{e}_j^1$ be the indicator members in I_0, I_1 respectively. $\mathcal{B}$ returns messages $(\{(w_j^0, \vec{e}_j^0)\}_{j\in[t]}, \{(w_j^1, \vec{v}_j^1)\}_{j\in[t]})$ to its challenger $\mathcal{C}$.

– After $\mathcal{B}$ receives the commitment com from $\mathcal{C}$, $\mathcal{B}$ first sets $H_1(ck, com) = c_1$. If $H_1(ck, com)$ has been queried by $\mathcal{A}$, $\mathcal{B}$ sets bad_4, and halt with output a random bit.

– $\mathcal{B}$ computes $z = \sum_{i=1}^{t} z_i$, $P = \mathrm{HI}.V_1(c_1, z)R^*$ and $com' = com - (\mathsf{HI.V}_2(c_1, z), 0)$, and sets $x = (P, t, ck, com')$.

– $\mathcal{B}$ samples $c_2 \leftarrow D(\mathsf{CSet})$ and calls TN's simulator algorithm $\mathcal{S}$ to get transcript (ω, c_2, z'). Then $\mathcal{B}$ sets $H_2(x, \omega) = c_2$. If $H_2(x, \omega)$ has been queried by $\mathcal{A}$, $\mathcal{B}$ sets bad_5, and halt with output a random bit.

Finally, $\mathcal{B}$ sets $\Sigma^* = (com, z, \omega, z')$ and returns Σ^* to $\mathcal{A}$.

Guess Stage. Finally $\mathcal{A}_2$ outputs a bit b' to $\mathcal{B}$, and then $\mathcal{B}$ returns b' to its challenger $\mathcal{C}$.

Note that, when bad_3, bad_4 and bad_5 does not occur, by strong honest-verifier zero-knowledge property of HI and special honest-verifier zero-knowledge of TN, the challenge signature Σ^* is statistically indistinguishable from the real one. And Σ^* is equivalent to the signature generated by I_b, where b is the bit sampled by $\mathcal{C}$ in the hiding game. Thus, the probability that $\mathcal{A}$ wins the t-anonymity game is the same as the probability that $\mathcal{B}$ wins the t-hiding game.

We now analyse the probability that bad_3, bad_4 and bad_5 do not occur. Assume the adversary has queried the signing oracle at most Q_s times and the hash function oracle at most Q_h times, by randomness of i^*, we have the probability that bad_3 does not occur is exact $\frac{1}{Q_h+Q_s}$. Since com is not controlled by $\mathcal{A}$, it is random in the view of $\mathcal{A}$. Thus, the probability that bad_4 and bad_5 occurs are negligible. Hence, we have

$$\mathsf{Adv}^{\mathsf{t-hiding}}_{\mathsf{THC}}(\mathcal{B}) \geq \mathsf{Adv}^{\mathsf{t-anonymity}}_{\mathsf{GC\text{-}TRS}}(\mathcal{A})/(Q_h + Q_s) - \mathsf{negl}(\lambda).$$

□

4 Lattice-based HI and THC scheme

In this section, we will present the lattice-based instantiations for the homomorphic canonical identification scheme and trapdoor homomorphic commitment scheme, respectively. Additionally, we will prove these instantiation schemes satisfy all the properties required by GC-TRS.

4.1 Lattice-based homomorphic canonical identification scheme

We now present the lattice-based homomorphic canonical identification scheme in Algorithm 5, which is based on the scheme proposed by Lyubashevsky [36]. More precisely, our scheme is a variant of the scheme of [36], in a parameter range that ensures *t-correctness* and *secure against* $t-1$ *corruption attack* instead of just standard correctness and soundness.

Let q be an odd modulus and $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d+1)$ be a ring. The probability distributions $\mathbb{S}_\eta^{l+k}$, $\mathbb{D}_\sigma^{d(l+k)}$, and $D(S_c)$, as well as the algorithm Rej used in Algorithm 5 are described in Sect. 2.5.

The security and the constraints on parameters are obtained as follows.

Theorem 4 *The scheme in* Algorithm 5 *is a secure homomorphic canonical identification scheme under* MLWE *and* MSIS *assumptions. A more precise statement is as follows:*

Algorithm 5 *has additive homomorphism and linear properties for any parameter.*

Assuming parameters satisfy the following conditions: $2^{10} < d(l+k) < 2^{45}$, $11/\pi \leq \sigma$, *and* $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$ *is the repetition rate parameter of the algorithm* Rej, *where* $T = d\eta\sqrt{d(l+k)}$, *then* Algorithm 5 *has t-correctness with error* $\frac{M-1}{M}$.

Assuming the $\mathsf{MLWE}_{l,\eta}$ *and* $\mathsf{MSIS}_{k,\beta}$ *problems are hard, then Algorithm 5 is secure against* $t-1$ *corruption attack, where* $\beta = 2(\sigma\sqrt{2t} + (t-1)d\eta)\sqrt{d(l+k)} + 2\sqrt{d}$.

Assuming $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$, *where* $T = d\eta\sqrt{d(l+k)}$, *then Algorithm 5 has strong honest verifier zero-knowledge.*

Algorithm 5 Lattice-based HI Scheme

Setup(1^λ):
1: Choose parameters d, q, k, l, η, σ
2: Sample $\boldsymbol{A} \leftarrow \mathcal{R}_q^{k\times l}$, $\bar{\boldsymbol{A}} = [\boldsymbol{A}|\boldsymbol{I}]$
3: **return** $pp = ((d, q, k, l, \eta, \sigma), \bar{\boldsymbol{A}})$

KeyGen(pp):
1: $\vec{\boldsymbol{s}} \leftarrow \mathbb{S}_\eta^{l+k}$, $\vec{\boldsymbol{p}} = \bar{\boldsymbol{A}}\vec{\boldsymbol{s}}$
2: $pk = \vec{\boldsymbol{p}}$, $sk = \vec{\boldsymbol{s}}$
3: **return** (pk, sk)

Proof-I(sk):
1: $\vec{\boldsymbol{y}} \leftarrow \mathbb{D}_\sigma^{d(l+k)}$
2: $\vec{\boldsymbol{w}} = \mathsf{A}(sk, \vec{\boldsymbol{y}}) = \bar{\boldsymbol{A}}\vec{\boldsymbol{y}}$
3: **return** $(\vec{\boldsymbol{w}}, \vec{\boldsymbol{y}})$

Challenge():
1: Sample $\boldsymbol{c} \leftarrow D(S_c)$
2: **return** $\boldsymbol{c}$

Proof-II($sk, \vec{\boldsymbol{y}}, \boldsymbol{c}$):
1: $\vec{\boldsymbol{z}} = \mathsf{Z}(sk, \vec{\boldsymbol{y}}, \boldsymbol{c}) = \boldsymbol{c} \cdot sk + \vec{\boldsymbol{y}}$
2: If $\mathsf{Rej}(\vec{\boldsymbol{z}}, \vec{\boldsymbol{z}} - \vec{\boldsymbol{y}}, \sigma) = 0$, $\vec{\boldsymbol{z}} = \perp$
3: **return** $\vec{\boldsymbol{z}}$

Verify($pk, \vec{\boldsymbol{z}}, \boldsymbol{c}, \vec{\boldsymbol{w}}$):
1: Check if $\|\vec{\boldsymbol{z}}\| \le \sigma\sqrt{2dt(l+k)}$
2: Check if $\vec{\boldsymbol{w}} = \mathsf{V}(pk, \boldsymbol{c}, \vec{\boldsymbol{z}}) = \bar{\boldsymbol{A}}\vec{\boldsymbol{z}} - \boldsymbol{c} \cdot pk$
3: If failed **return** 0, otherwise **return** 1

We now prove that Algorithm 5 satisfies all the properties required by GC-TRS.

Lemma 10 (Additive homomorphism and linear property) *Algorithm 5 has additive homomorphism and linear properties under any parameter setting.*

Proof Note that the function $\mathsf{V}(\cdot)$ is additive homomorphism:

$$\begin{aligned}\mathsf{V}(pk_1, \boldsymbol{c}, \vec{\boldsymbol{z}}_1) + \mathsf{V}(pk_2, \boldsymbol{c}, \vec{\boldsymbol{z}}_2) &= \bar{\boldsymbol{A}}\vec{\boldsymbol{z}}_1 - \boldsymbol{c} \cdot pk_1 + \bar{\boldsymbol{A}}\vec{\boldsymbol{z}}_2 - \boldsymbol{c} \cdot pk_2 \\ &= \bar{\boldsymbol{A}}(\vec{\boldsymbol{z}}_1 + \vec{\boldsymbol{z}}_2) - \boldsymbol{c} \cdot (pk_1 + pk_2) = \mathsf{V}(pk_1 + pk_2, \boldsymbol{c}, \vec{\boldsymbol{z}}_1 + \vec{\boldsymbol{z}}_2).\end{aligned}$$

Let $\mathsf{V}_1(\boldsymbol{c}, \vec{\boldsymbol{z}}) = -\boldsymbol{c}$ and $\mathsf{V}_2(\boldsymbol{c}, \vec{\boldsymbol{z}}) = \bar{\boldsymbol{A}}\vec{\boldsymbol{z}}$, then we have

$$\mathsf{V}(pk, \boldsymbol{c}, \vec{\boldsymbol{z}}) = \mathsf{V}_1(\boldsymbol{c}, \vec{\boldsymbol{z}}) \cdot pk + \mathsf{V}_2(\boldsymbol{c}, \vec{\boldsymbol{z}}).$$

Thus, the function $\mathsf{V}(\cdot)$ has the linear property. □

Lemma 11 (Correctness) *Assuming parameters satisfy the following conditions:* $2^{10} < d(l+k) < 2^{45}$, $11/\pi \le \sigma$, $T = d\eta\sqrt{d(l+k)}$, *and* $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$ *is the repetition rate parameter of the algorithm* Rej, *then Algorithm 5 has t-correctness with error* $\frac{M-1}{M}$.

Proof First, we will prove that given a set of honestly generated transcripts $\{pk_i, \vec{\boldsymbol{z}}_i, \boldsymbol{c}, \vec{\boldsymbol{w}}_i\}_{i=1}^{t'}$ with $t' \in [t]$, the verification procedure always accepts if $\vec{\boldsymbol{z}}_i \neq \perp$, except for a negligible probability. Since we have

$$\bar{\boldsymbol{A}}\vec{\boldsymbol{z}}_i - \boldsymbol{c} \cdot pk_i = \bar{\boldsymbol{A}}(\boldsymbol{c} \cdot sk_i + \vec{\boldsymbol{y}}_i) - \boldsymbol{c} \cdot pk_i = \vec{\boldsymbol{w}}_i.$$

Thus, we can obtain $\mathsf{V}\left(\sum_{i=1}^{t'} pk_i, \boldsymbol{c}, \sum_{i=1}^{t'} \vec{\boldsymbol{z}}_i\right) = \sum_{i=1}^{t'} \vec{\boldsymbol{w}}_i$.

Since $sk_i \in \mathbb{S}_\eta^{l+k}$ and $\boldsymbol{c} \in \mathbb{S}_1$, we can obtain that $\|\boldsymbol{c} \cdot sk_i\| \le d\eta \cdot \sqrt{d(l+k)}$. It follows from Lemma 6 that, when $\vec{\boldsymbol{z}}_i \neq \perp$, $\vec{\boldsymbol{z}}_i$ follows the distribution $\mathbb{D}_\sigma^{d(l+k)}$, except for a negligible probability. Thus, by Lemma 4, $\vec{\boldsymbol{z}} = \sum_{i=1}^{t'} \vec{\boldsymbol{z}}_i$ follows the distribution $\mathbb{D}_{\sqrt{t'}\sigma}^{d(l+k)}$, except for a negligible probability. Then, according to Lemma 5, $\|\vec{\boldsymbol{z}}\|_2 \le \sigma\sqrt{2dt'(l+k)} \le \sigma\sqrt{2dt(l+k)}$ holds except for a negligible probability.

According to Lemma 6 again, we know that for an honestly generated transcript, the probability to have $\vec{\boldsymbol{z}} = \perp$ is at most $\frac{M-1}{M}$. □

Lemma 12 (Secure against $t-1$ corruption attack) *Assuming the* $\mathsf{MLWE}_{l,\eta}$ *and* $\mathsf{MSIS}_{k,\beta}$ *problems are hard, then Algorithm 5 is secure against* $t-1$ *corruption attack, where* $\beta = 2(\sigma\sqrt{2t} + (t-1)d\eta)\sqrt{d(l+k)} + 2\sqrt{d}$.

Proof We denote $\mathcal{A}$ as a PPT adversary breaking the security and build an algorithm $\mathcal{B}$ to break the MSIS problem. Suppose $\mathcal{B}$ is given a uniform matrix $[\boldsymbol{A}|\boldsymbol{I}|\vec{\boldsymbol{p}}] \in \mathcal{R}_q^{k\times(k+l+1)}$ from its challenger $\mathcal{C}$ [6].

$\mathcal{B}$ first sets public parameter $\bar{\boldsymbol{A}} = [\boldsymbol{A}|\boldsymbol{I}]$, and picks a random index $i^* \in [q_{key}]$. Then $\mathcal{B}$ runs $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(pp)$ for all $i \in [q_{key}]$ with $i \neq i^*$ and sets $pk_{i^*} = \vec{\boldsymbol{p}}$. Next, $\mathcal{B}$ gives public parameter pp and $L = \{pk_1, \ldots, pk_{q_{key}}\}$ to the adversary $\mathcal{A}$.

When $\mathcal{A}$ conducts corruption queries on input i, if $i = i^*$, $\mathcal{B}$ sets bad_1 and aborts. Otherwise, $\mathcal{B}$ returns the corresponding secret key sk_i.

Finally, $\mathcal{A}$ returns a forgery $(\vec{\boldsymbol{z}}, \boldsymbol{c}, \vec{\boldsymbol{z}}', \boldsymbol{c}', \vec{\boldsymbol{w}}, I^*)$. If $pk_{i^*} \notin I^*$, $\mathcal{B}$ sets bad_2 and aborts. Otherwise, denoting I^* as $\{pk_{i_1}, \ldots, pk_{i_{t-1}}, pk_{i^*}\}$ and the corresponding secret key of pk_{i_j} as sk_{i_j}, we have

$$\bar{\boldsymbol{A}}\vec{\boldsymbol{z}} - \boldsymbol{c}(pk_{i_1} + \cdots + pk_{i_{t-1}} + pk_{i^*}) = \bar{\boldsymbol{A}}\vec{\boldsymbol{z}}' - \boldsymbol{c}'(pk_{i_1} + \cdots + pk_{i_{t-1}} + pk_{i^*}).$$

Thus, let $\vec{\boldsymbol{x}} = \vec{\boldsymbol{z}} - \vec{\boldsymbol{z}}' + \tilde{\boldsymbol{c}}(sk_{i_1} + \cdots + sk_{i_{t-1}})$ and $\tilde{\boldsymbol{c}} = \boldsymbol{c}' - \boldsymbol{c}$, we have

$$[\boldsymbol{A}|\boldsymbol{I}|\vec{\boldsymbol{p}}]\begin{pmatrix}\vec{\boldsymbol{x}}\\ \tilde{\boldsymbol{c}}\end{pmatrix} = \boldsymbol{0},$$

and

$$\|\tilde{\boldsymbol{c}}\| = \|\boldsymbol{c}' - \boldsymbol{c}\|_2 \leq 2\sqrt{d},$$

$$\|\vec{\boldsymbol{x}}\| = \|\vec{\boldsymbol{z}} - \vec{\boldsymbol{z}}' + \tilde{\boldsymbol{c}}(sk_{i_1} + \cdots + sk_{i_{t-1}})\|_2 \leq 2\sigma\sqrt{2dt(l+k)} + 2(t-1)d\eta\sqrt{d(l+k)}.$$

Therefore, $\mathcal{B}$ can obtain a solution $(\vec{\boldsymbol{x}}||\tilde{\boldsymbol{c}})$ to the $\mathsf{MSIS}_{k,\beta}$ problem, where $\beta = 2\sigma\sqrt{2dt(l+k)} + 2(t-1)d\eta\sqrt{d(l+k)} + 2\sqrt{d}$.

`Probability analysis.` We analyze the probability that bad_1 and bad_2 do not occur. Note that, by $\mathsf{MLWE}_{l,\eta}$ assumption, $\mathcal{A}$ cannot distinguish which public keys are honestly generated by the KeyGen and which one is the uniform vector. Assuming $\mathcal{A}$ has queried the corrupted oracle at most Q_c times, according to the randomness of i^*, the probability that bad_1 does not occur is $\frac{q_{key}-Q_c}{q_{key}}$. Conditioned on bad_1 does not occur, I^* contains at least one honest public key, and by the randomness of i^* again, the probability that bad_2 does not occur is $\frac{1}{q_{key}-Q_c}$. Thus, we have the probability that bad_1 and bad_2 do not occur is at least $\frac{1}{q_{key}}$.

Let ϵ be the probability that $\mathcal{A}$ can break the security against $t-1$ corruption attack game, we have $\frac{\epsilon}{q_{key}}$ is the probability that $\mathcal{B}$ can successfully break the MSIS problem. Therefore, assuming the hardness of $\mathsf{MSIS}_{k,\beta}$ problem, Algorithm 5 provides the security against $t-1$ corruption attack. □

Since many literatures such as [20, 30, 36], have already proven the zero-knowledge properties of this identification scheme. Here we only give the corresponding lemma and ignore the proof.

[6] Note that, the challenge matrix that adversary $\mathcal{B}$ received is in Hermit normal form. And there exists reduction from the Hermit normal form to the standard form (for more details, see [43]).

Lemma 13 (SHVZK) *Assuming* $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$, *then Algorithm 5 has strong honest verifier zero-knowledge.*

4.2 Lattice-based trapdoor homomorphic commitment scheme

We now describe the lattice-based trapdoor homomorphic commitment scheme in Algorithm 6, which is based on the scheme proposed in [19]. More precisely, our scheme is a variant of the scheme of [19], in a parameter range that ensures *binding for weak opening* instead of just standard binding.

Let q be an odd modulus and $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ be a ring. The message is a ring element vector $\vec{\boldsymbol{m}} \in \mathcal{R}_q^n$. Let $\sigma' = \sqrt{\frac{5}{2\pi}} \cdot \sigma^2 \cdot (\sqrt{(k+l)d} + \sqrt{kd \cdot \lceil \log q \rceil} + \lambda)$, $m = l + k \cdot \lceil \log q \rceil + k$ and $S_r = \{\vec{\boldsymbol{r}} \in \mathcal{R}_q^m \| \|\vec{\boldsymbol{r}}\| \leq \sigma' \sqrt{2tdm}\}$. The probability distribution $\mathbb{D}_{\sigma'}^{dm}$ and algorithms TrapGen, SampleD are described in Sect. 2.5.

Algorithm 6 Lattice-based THC Scheme

$\underline{\mathsf{Setup}(1^\lambda)}$:
1: Choose d, q, k, l, σ
2: **return** $pp = (d, q, k, l, \sigma)$

$\underline{\mathsf{Gen}(pp)}$:
1: Sample $\boldsymbol{B} \leftarrow \mathcal{R}_q^{k \times (m-k)}$, $\bar{\boldsymbol{B}} = [\boldsymbol{B} | \boldsymbol{I}]$
2: **return** $ck = \bar{\boldsymbol{B}}$

$\underline{\mathsf{Commit}_{ck}(\vec{\boldsymbol{m}})}$:
1: $\vec{\boldsymbol{r}} \leftarrow \mathbb{D}_{\sigma'}^{dm}$
2: $\vec{\boldsymbol{t}} = \bar{\boldsymbol{B}}\vec{\boldsymbol{r}} + \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}} \end{bmatrix}$
3: **return** $com = \vec{\boldsymbol{t}}$

$\underline{\mathsf{Open}_{ck}(com, \vec{\boldsymbol{r}}, \vec{\boldsymbol{m}})}$:
1: Check if $\|\vec{\boldsymbol{r}}\| \leq \sigma' \sqrt{2tdm}$
2: Check if $\vec{\boldsymbol{t}} = \bar{\boldsymbol{B}}\vec{\boldsymbol{r}} + \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}} \end{bmatrix}$
3: If failed **return** 0, otherwise **return** 1

$\underline{\mathbf{TGen}(pp)}$:
1: $(\bar{\boldsymbol{B}}, \boldsymbol{T}) \leftarrow \mathsf{TrapGen}(k, l, \sigma)$
2: Set $tck = \bar{\boldsymbol{B}}$ and $td = \boldsymbol{T}$
3: **return** (tck, td)

$\underline{\mathbf{TCommit}_{tck}(td)}$:
1: Sample $\vec{\boldsymbol{t}} \leftarrow \mathcal{R}_q^k$,
2: **return** $com = \vec{\boldsymbol{t}}$

$\underline{\mathbf{Eqv}_{tck}(td, com, \vec{\boldsymbol{m}})}$:
1: Let $\vec{\boldsymbol{u}} = \vec{\boldsymbol{t}} - \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}} \end{bmatrix}$
2: $\vec{\boldsymbol{r}} \leftarrow \mathsf{SampleD}(\bar{\boldsymbol{B}}, \boldsymbol{T}, \vec{\boldsymbol{u}})$
3: **return** $\vec{\boldsymbol{r}}$

The security properties and the constraints on parameters are obtained as follows.

Theorem 5 *Assuming that* $\sigma' = \sqrt{\frac{5}{2\pi}} \cdot \sigma^2 \cdot (\sqrt{(k+l)d} + \sqrt{kd \cdot \lceil \log q \rceil} + \lambda)$ *and* $m = l + k \cdot \lceil \log q \rceil + k$, *then the scheme in Algorithm 6 is a secure trapdoor homomorphic commitment scheme under* MLWE *and* MSIS *assumptions. A more precise statement is as follows:*

Algorithm 6 has message homomorphism and uniform key properties for any parameter.

Assuming parameters satisfy the following conditions: $2^{10} < dm < 2^{45}$, $11/\pi \leq \sigma'$, *then Algorithm 6 has t-additive homomorphism.*

Assuming the $\mathsf{MLWE}_{l,\sigma}$ *problem is hard, then Algorithm 6 satisfies equivocability.*

Let $S_{fac} = \{\boldsymbol{c} \in \mathcal{R}_q^\times \mid \|\boldsymbol{c}\|_\infty \leq 2\}$ *and* $S_r' = \{\vec{\boldsymbol{r}} \in \mathcal{R}_q^m \mid \|\vec{\boldsymbol{r}}\|_2 \leq 2\beta\}$ *with* $\sigma' \sqrt{2tdm} \leq \beta$, *assuming the* $\mathsf{MSIS}_{k-n,8d\beta}$ *problem is hard, then Algorithm 6 satisfies binding for weak opening with respect to* S_{fac}, S_r'.

We now prove that Algorithm 6 satisfies all the properties required by GC-TRS.

Lemma 14 (Message homomorphism and uniform key) *The Algorithm 6 has message homomorphism and uniform key properties under any parameter setting.*

Proof Note that, by the definition of Gen algorithm, the Algorithm 6 obviously has uniform key.

Let $(com = \vec{t}, \vec{r}, \vec{m})$ be a valid transcript, for any message $\vec{m}'$, we have

$$\vec{t} + \begin{bmatrix} \mathbf{0} \\ \vec{m}' \end{bmatrix} = \boldsymbol{B}\vec{r} + \begin{bmatrix} \mathbf{0} \\ \vec{m} + \vec{m}' \end{bmatrix}.$$

Thus the THC scheme also has message homomorphism. □

Lemma 15 (Additive homomorphism) *Assuming parameters satisfy the following conditions:* $2^{10} < dm < 2^{45}$, $11/\pi \leq \sigma'$, *then Algorithm 6 has additive homomorphism.*

Proof Let $\{(com_i, \vec{r}_i, \vec{m}_i)\}_{i=1}^{t'}$ be valid transcripts, where $t' \in [t]$. Thus, by Lemma 4 and Lemma 5, we have

$$\Pr\left[\sum_{i=1}^{t'} \|\vec{r}_i\| \leq \sigma'\sqrt{2tdm}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Hence, by definition, we have

$$\Pr\left[\mathsf{Open}_{ck}\left(\sum_{i=1}^{t} com_i, \sum_{i=1}^{t'} \vec{r}_i, \sum_{i=1}^{t} \vec{m}_i\right) = 1\right] \geq 1 - \mathsf{negl}(\lambda).$$

Therefore, the commitment scheme has additive homomorphism. □

Lemma 16 (Equivocability) *Assuming the* $\mathsf{MLWE}_{l,\sigma}$ *problem is hard, then Algorithm 6 satisfies equivocability.*

Proof The proof is based on a sequence of hybrid games.

Game 0. The first game is the real equivocability security game with challenge bit $b = 0$ as presented in Fig.2.

Game 1. In this game, the commitment key is not computed by the Gen algorithm anymore, but generated by the TGen algorithm as presented in Fig.2.

Game 2. In this game, the oracle $\mathcal{O}$ is modified to generate commitments using the Eqv algorithm. This game exactly corresponds to the real equivocability security game with challenge bit $b = 1$ as presented in Fig.2.

According to Lemma 7, assuming the $\mathsf{MLWE}_{l,\sigma}$ problem is hard, the ck generated in Game 1 is computationally indistinguishable from the ck generated in Game 0. Since the difference between Game 0 and Game 1 is only in the method of ck generation, it follows that Game 0 and Game 1 are also computationally indistinguishable.

According to Lemma 7 again, we have that the $(\vec{t}, \vec{r}, \vec{m})$ generated in Game 1 is statistically indistinguishable from the $(\vec{t}, \vec{r}, \vec{m})$ generated in Game 2. Thus, Game 1 and Game 2 are also computationally indistinguishable.

Therefore, the Algorithm 6 satisfies equivocability. □

Lemma 17 (Binding for weak opening) *Let* $S_{fac} = \{\boldsymbol{c} \in \mathcal{R}_q^{\times} \mid \|\boldsymbol{c}\|_{\infty} \leq 2\}$ *and* $S'_r = \{\vec{\boldsymbol{r}} \in \mathcal{R}_q^m \mid \|\vec{\boldsymbol{r}}\|_2 \leq \beta\}$ *with* $\sigma'\sqrt{2tdm} \leq \beta$, *assuming the* $\mathsf{MSIS}_{k-n,4d\beta}$ *problem is hard, then Algorithm 6 satisfies binding for weak opening with respect to* S_{fac}, S'_r.

Game 0:
1: Sample $\boldsymbol{B} \leftarrow \mathcal{R}_q^{k\times(m-k)}$
2: $ck = \bar{\boldsymbol{B}} = [\boldsymbol{B}|\boldsymbol{I}]$
3: $b \leftarrow \mathcal{A}(ck, \mathcal{O}(\cdot))$
4: **return** b

$\mathcal{O}(\vec{\boldsymbol{m}})$:
1: $\vec{\boldsymbol{r}} \leftarrow \mathbb{D}_{\sigma'}^{dm}$
2: $\vec{\boldsymbol{t}} = \bar{\boldsymbol{B}}\vec{\boldsymbol{r}} + \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}} \end{bmatrix}$
3: **return** $(\vec{\boldsymbol{t}}, \vec{\boldsymbol{r}}, \vec{\boldsymbol{m}})$

Game 1:
1: $(\bar{\boldsymbol{B}}, \boldsymbol{T}) \leftarrow \mathsf{TrapGen}(k, l, \sigma)$
2: $ck = \bar{\boldsymbol{B}}$
3: $b \leftarrow \mathcal{A}(ck, \mathcal{O}(\cdot))$
4: **return** b

$\mathcal{O}(\vec{\boldsymbol{m}})$:
1: $\vec{\boldsymbol{r}} \leftarrow \mathbb{D}_{\sigma'}^{dm}$
2: $\vec{\boldsymbol{t}} = \bar{\boldsymbol{B}}\vec{\boldsymbol{r}} + \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}} \end{bmatrix}$
3: **return** $(\vec{\boldsymbol{t}}, \vec{\boldsymbol{r}}, \vec{\boldsymbol{m}})$

Game 2:
1: $(\bar{\boldsymbol{B}}, \boldsymbol{T}) \leftarrow \mathsf{TrapGen}(k, l, \sigma)$
2: $ck = \bar{\boldsymbol{B}}$
3: $b \leftarrow \mathcal{A}(ck, \mathcal{O}(\cdot))$
4: **return** b

$\mathcal{O}(\vec{\boldsymbol{m}})$:
1: $\vec{\boldsymbol{t}} \leftarrow \mathcal{R}_q^k$
2: $\vec{\boldsymbol{u}} = \vec{\boldsymbol{t}} - \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}} \end{bmatrix}$
3: $\vec{\boldsymbol{r}} \leftarrow \mathsf{SampleD}(\bar{\boldsymbol{B}}, \boldsymbol{T}, \vec{\boldsymbol{u}})$
4: **return** $(\vec{\boldsymbol{t}}, \vec{\boldsymbol{r}}, \vec{\boldsymbol{m}})$

Fig. 2 Hybrid games of equivocability

Proof Suppose that there is an adversary $\mathcal{A}$ that can outputs two weak openings $(\boldsymbol{c}_1, \vec{\boldsymbol{r}}_1, \vec{\boldsymbol{m}}_1)$ and $(\boldsymbol{c}_2, \vec{\boldsymbol{r}}_2, \vec{\boldsymbol{m}}_2)$ with $\vec{\boldsymbol{m}}_1 \neq \vec{\boldsymbol{m}}_2$. Then, we have

$$\bar{\boldsymbol{B}}\vec{\boldsymbol{r}}_1 + \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}}_1 \end{bmatrix} = \vec{\boldsymbol{t}} = \bar{\boldsymbol{B}}\vec{\boldsymbol{r}}_2 + \begin{bmatrix} \boldsymbol{0} \\ \vec{\boldsymbol{m}}_2 \end{bmatrix},$$

which implies $\vec{\boldsymbol{r}}_1 \neq \vec{\boldsymbol{r}}_2$. Since $\boldsymbol{c}_1, \boldsymbol{c}_2 \in S_{fac}$, we have $\boldsymbol{c}_1\boldsymbol{c}_2(\vec{\boldsymbol{r}}_1 - \vec{\boldsymbol{r}}_2) \neq 0$, and $\|\boldsymbol{c}_1\boldsymbol{c}_2(\vec{\boldsymbol{r}}_1 - \vec{\boldsymbol{r}}_2)\| \leq 4d\beta$. Let $\boldsymbol{B}_1$ be the matrix formed by the first $k-n$ rows of matrix $\boldsymbol{B}$, then we have $\boldsymbol{B}_1 \cdot \boldsymbol{c}_1\boldsymbol{c}_2(\vec{\boldsymbol{r}}_1 - \vec{\boldsymbol{r}}_2) = 0$. This implies that $\mathcal{A}$ can break the $\mathsf{MSIS}_{k-n,4d\beta}$ problem, which is contradictory. □

5 *t*-out-of-*n* proof protocol

5.1 Linear *t*-out-of-*n* proof protocol

In the following, we will present a linear lattice-based t-out-of-n proof protocol. By definition, the t-out-of-n proof protocol is related to the used commitment scheme. In this subsection, the applied commitment is described in Algorithm 6. The operations in our protocol are performed over the power-of-2 cyclotomic ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$. For soundness, we require that $\mathcal{R}_q$ splits into exactly ϖ factors so that $q^{-d/\varpi}$ is negligible.

Let t, n be two positive numbers and $\boldsymbol{P} = \{\vec{\boldsymbol{p}}_1, \ldots, \vec{\boldsymbol{p}}_n\} \subset \mathcal{R}_q^k$ be the public set. Let $\vec{v} \in \{0,1\}^n$ be a binary vector with Hamming weights of exactly t. Since the input message of Algorithm 6 needs to be vectors of ring elements, before delving into the construction of t-out-of-n protocol, it is essential to first define the encoding method of vector $\vec{v}$.

In this subsection, the encoding method involves dividing the vector $\vec{v}$ into n' blocks and then converting each block into a ring element. Here, n' is defined as n/ϖ (assuming n' is an integer without loss of generality). Specifically, we divide $\vec{v}$ into n' blocks, with each block containing ϖ terms denoted as $\vec{v} = (\vec{v}_1|\ldots|\vec{v}_{n'})$. The encoding of $\vec{v}$ is defined as $\vec{\boldsymbol{v}} = (\boldsymbol{v}_1\|\cdots\|\boldsymbol{v}_{n'}) \in \mathcal{R}_q^{n'}$, where $\boldsymbol{v}_i = \mathsf{NTT}^{-1}(\vec{v}_i)$.

Let $ck = (\boldsymbol{B}_0, \boldsymbol{B}_1, \boldsymbol{B}_2) \in \mathcal{R}_q^{\tilde{k}\times m}$ be the commitment key, where $\boldsymbol{B}_0 \in \mathcal{R}_q^{k_1\times m}$, $\boldsymbol{B}_1 \in \mathcal{R}_q^{k\times m}$ and $\boldsymbol{B}_2 \in \mathcal{R}_q^{n'\times m}$, and $\vec{\boldsymbol{r}} \in S_r$ be the randomness following the distribution $\mathbb{D}_{\sigma'}^{dm}$. Then the commitment of $(\boldsymbol{P}\vec{v}, \vec{v})$ is $com = (\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2)$, where $\vec{\boldsymbol{t}}_0 = \boldsymbol{B}_0\vec{\boldsymbol{r}}, \vec{\boldsymbol{t}}_1 = \boldsymbol{B}_1\vec{\boldsymbol{r}} + \boldsymbol{P}\cdot\vec{v}$,

$\vec{t}_2 = \boldsymbol{B}_2\vec{r} + \vec{v}$. Now, we define the following two sets:

$$S_{fac} = \{\boldsymbol{c} \in \mathcal{R}_q^{\times} \mid \|\boldsymbol{c}\|_\infty \leq 2\} \text{ and } S'_r = \{\vec{r} \in \mathcal{R}_q^m \mid \|\vec{r}\| \leq \beta\}.$$

Definition 25 Let S'_r, S_{fac} be the sets defined above, $com = (\vec{t}_0, \vec{t}_1, \vec{t}_2)$, we define the following two relations:

$$\mathfrak{R} = \left\{ (\boldsymbol{P}, t, ck, com), (\vec{v}, \vec{r}) \middle| \begin{array}{l} \vec{v} \in \{0,1\}^n \wedge \|\vec{v}\|_1 = t \wedge \vec{r} \in S_r \wedge \vec{t}_0 = \boldsymbol{B}_0\vec{r} \\ \wedge\, \vec{t}_1 = \boldsymbol{B}_1\vec{r} + \boldsymbol{P}\cdot\vec{v} \wedge \vec{t}_2 = \boldsymbol{B}_2\vec{r} + \vec{v} \end{array} \right\},$$

$$\mathfrak{R}' = \left\{ (\boldsymbol{P}, t, ck, com), (\vec{v}, \vec{r}, \boldsymbol{c}) \middle| \begin{array}{l} \vec{v} \in \{0,1\}^n \wedge \|\vec{v}\|_1 = t \wedge \boldsymbol{c} \in S_{fac} \wedge\ \boldsymbol{c}\vec{r} \in S'_r \\ \wedge \vec{t}_0 = \boldsymbol{B}_0\vec{r} \wedge \vec{t}_1 = \boldsymbol{B}_1\vec{r} + \boldsymbol{P}\cdot\vec{v} \wedge \vec{t}_2 = \boldsymbol{B}_2\vec{r} + \vec{v} \end{array} \right\}.$$

The proof protocol takes $\boldsymbol{P} = \{\vec{p}_1, \ldots, \vec{p}_n\}, t, ck = (\boldsymbol{B}_0, \boldsymbol{B}_1, \boldsymbol{B}_2)$ and $com = (\vec{t}_0, \vec{t}_1, \vec{t}_2)$ as statement, $\vec{v}$ and $\vec{r}$ as witness, and will output a proof. In order to convince the verifier that the prover has the witness $w = (\vec{v}, \vec{r})$, the prover should prove the committed message $(\vec{p}, \vec{v})$ satisfies the following conditions, where $\vec{p} = \boldsymbol{P}\cdot\vec{v}$:

1. $\vec{v} \circ (\vec{v} - \vec{1}) = \vec{0}$, where '$\circ$' denotes component-wise product of vector and $\vec{1} = (1, \ldots, 1) \in \mathcal{R}_q^{n'}$;
2. For any uniformly random vector $\vec{\gamma}_1 \in \mathcal{M}_q^{k\varpi}$, we let

$$F_1 = \mathsf{NTT}(\boldsymbol{P}) = (\mathsf{NTT}(\vec{\boldsymbol{p}}_1), \ldots, \mathsf{NTT}(\vec{\boldsymbol{p}}_n)) \in \mathcal{M}_q^{k\varpi \times n}, \tag{1}$$

   $\vec{x}_1 = F_1^T\vec{\gamma}_1$ and divide $\vec{x}_1$ into n' pieces, each of which has ϖ terms such that $\vec{x}_1 = (\vec{x}_{11} \| \ldots \| \vec{x}_{1n'})$ and let $\vec{\boldsymbol{x}}_1 = (\boldsymbol{x}_{11} \| \ldots \| \boldsymbol{x}_{1n'}) \in \mathcal{R}_q^{n'}$ with $\boldsymbol{x}_{1i} = \mathsf{NTT}^{-1}(\vec{x}_{1i})$. The prover will prove that the coefficients of the first d/ϖ terms of the polynomial

$$\langle \vec{v}, \vec{\boldsymbol{x}}_1 \rangle - \langle \vec{p}, \vec{\boldsymbol{\gamma}}_1 \rangle \in \mathcal{R}_q$$

   are all 0's, i.e., if $\langle \vec{v}, \vec{\boldsymbol{x}}_1 \rangle - \langle \vec{p}, \vec{\boldsymbol{\gamma}}_1 \rangle = a_0 + a_1 X + \cdots + a_{d-1}X^{d-1}$, then we have $a_0 = a_1 = \cdots = a_{d/\varpi - 1} = 0$, where $\vec{\boldsymbol{\gamma}}_1 = (\boldsymbol{\gamma}_{11} \| \cdots \| \boldsymbol{\gamma}_{1k}) \in \mathcal{R}_q^k$ with $\boldsymbol{\gamma}_{1i} = \mathsf{NTT}^{-1}(\vec{\gamma}_{1i})$.
3. For any uniformly random vector $\vec{\gamma}_2 \in \mathcal{M}_q^{\varpi}$, we let

$$F_2 = \begin{pmatrix} 1 & \cdots & 1 \\ 0 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & \cdots & 0 \end{pmatrix} \in \mathcal{M}_q^{\varpi \times n}, \tag{2}$$

   $\vec{x}_2 = F_2^T\vec{\gamma}_2$. Similarly, we let $\vec{\boldsymbol{x}}_2 = (\boldsymbol{x}_{21} \| \cdots \| \boldsymbol{x}_{2n'}) \in \mathcal{R}_q^{n'}$ with $\boldsymbol{x}_{2i} = \mathsf{NTT}^{-1}(\vec{x}_{2i})$. The prover will prove the coefficients of the first d/ϖ terms of the polynomial

$$\langle \vec{v}, \vec{\boldsymbol{x}}_2 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 \in \mathcal{R}_q$$

   are all 0's, where $\boldsymbol{\gamma}_2 = \mathsf{NTT}^{-1}(\vec{\gamma}_2)$, $\boldsymbol{e}_t = \mathsf{NTT}^{-1}(\vec{e}_t)$ with $\vec{e}_t = (t, 0, \ldots, 0)$.

We note that the 1st condition is to check $\vec{v} \in \{0,1\}^n$, the 2nd condition is to prove $\boldsymbol{P}\vec{v} = \vec{p}$ and the 3rd condition is to show $\|\vec{v}\|_1 = t$. We briefly show that the equivalence of proving the 2nd condition and proving $\boldsymbol{P}\vec{v} = \vec{p}$ more concretely. If the prover can prove that the coefficients of the first d/ϖ terms of the polynomial $\langle \vec{v}, \vec{\boldsymbol{x}}_1 \rangle - \langle \vec{p}, \vec{\boldsymbol{\gamma}}_1 \rangle$ are all 0's, then by Lemma 2, we have $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\langle \vec{v}, \vec{\boldsymbol{x}}_1 \rangle - \langle \vec{p}, \vec{\boldsymbol{\gamma}}_1 \rangle)_i = 0$. By definition, we have both $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\langle \vec{v}, \vec{\boldsymbol{x}}_1 \rangle)_i = \langle \vec{v}, \vec{x}_1 \rangle$ and $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\langle \vec{p}, \vec{\boldsymbol{\gamma}}_1 \rangle)_i = \langle \vec{p}, \vec{\gamma}_1 \rangle$

where $\vec{p} = (\vec{p}_1 \| \cdots \| \vec{p}_k) \in \mathcal{M}_q^{k\varpi}$ with $\boldsymbol{p}_i = \mathsf{NTT}^{-1}(\vec{p}_i)$. Then by Lemma 1, we have $\langle \vec{v}, \vec{x}_1 \rangle = \langle \vec{v}, F_1^T \vec{\gamma}_1 \rangle = \langle F_1 \vec{v}, \vec{\gamma}_1 \rangle$. Thus we have $\langle F_1 \vec{v} - \vec{p}, \vec{\gamma}_1 \rangle = 0$. Since $\vec{\gamma}_1$ is a uniformly random vector, if $F\vec{v} - \vec{p} \neq 0$, then the probability of $\langle F_1 \vec{v} - \vec{p}, \vec{\gamma}_1 \rangle = 0$ is exactly $q^{-d/\varpi}$, which is negligible under suitable parameters.

Because the 2nd condition and the 3rd condition are very similar, the prover is able to prove them together. More specifically, since $\langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}}_1 \rangle - \langle \vec{\boldsymbol{p}}, \vec{\boldsymbol{\gamma}}_1 \rangle$ and $\langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}}_2 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2$ are independent, the prover only needs to prove the first d/ϖ coefficients of $\langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}}_1 \rangle - \langle \vec{\boldsymbol{p}}, \vec{\boldsymbol{\gamma}}_1 \rangle + \langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}}_2 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2$ are zeros. To achieve this, the prover can use a uniformly random polynomial $\boldsymbol{g}$ that has the first d/ϖ coefficients equal to zero to mask it, i.e., compute $\boldsymbol{h} = \langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}}_1 \rangle - \langle \vec{\boldsymbol{p}}, \vec{\boldsymbol{\gamma}}_1 \rangle + \langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}}_2 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{g}$ and then send $\boldsymbol{h}$ to the verifier. Hence, the verifier can manually check the first d/ϖ coefficients of $\boldsymbol{h}$ are indeed zeros.

Next, the prover needs to convince the verifier that $\boldsymbol{h}$ is in a correct form. Let $\boldsymbol{t}_3 = \langle \vec{\boldsymbol{b}}_3, \vec{\boldsymbol{r}} \rangle + \boldsymbol{g}$ be the commitment of $\boldsymbol{g}$ and $\vec{\boldsymbol{x}} = \vec{\boldsymbol{x}}_1 + \vec{\boldsymbol{x}}_2$, then we have

$$\begin{aligned} &\langle \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{x}} \rangle - \langle \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{\gamma}}_1 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{t}_3 - \boldsymbol{h} \\ &\quad = \langle (\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{\gamma}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{r}} \rangle + (\langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}} \rangle - \langle \vec{\boldsymbol{p}}, \vec{\boldsymbol{\gamma}}_1 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{g} - \boldsymbol{h}). \end{aligned} \tag{3}$$

Thus, the prover only needs to guarantee that the above formula is the commitment of $\mathbf{0}$, which can be done using the approach by [9].

Finally, as for the 1st condition, the prover may prove for each $\boldsymbol{v}_i$ separately. It is able to sample a Gaussian vector $\vec{\boldsymbol{y}} \leftarrow \mathbb{D}_\sigma^{dm}$ and construct a quadratic polynomial:

$$\begin{aligned} \varphi_i(\boldsymbol{c}) &= (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle - \boldsymbol{c}\boldsymbol{v}_i) \cdot (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle - \boldsymbol{c}\boldsymbol{v}_i + \boldsymbol{c}) \\ &= \boldsymbol{v}_i(\boldsymbol{v}_i - 1) \cdot \boldsymbol{c}^2 - \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle (2\boldsymbol{v}_i - 1) \cdot \boldsymbol{c} + \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle^2, \end{aligned} \tag{4}$$

where $\vec{\boldsymbol{b}}_{2i}$ is the ith row vector of $\boldsymbol{B}_2$. Then the prover shows that the square term of the polynomial $\varphi_i(\boldsymbol{c})$ is zero. To do this, the prover computes a commitment $\boldsymbol{t}_4 = \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{r}} \rangle - \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle (2\boldsymbol{v}_i - 1)$, then sends $\boldsymbol{t}_4$ and $\langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{y}} \rangle + \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle^2$ to the verifier. The verifier randomly chooses a $\boldsymbol{c}$ for the prover.The prover responds by sending $\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \boldsymbol{c}\vec{\boldsymbol{r}}$. Since

$$\begin{aligned} \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}} \rangle - \boldsymbol{c}\boldsymbol{t}_{2i} &= \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle - \boldsymbol{c}\boldsymbol{v}_i, \\ (\langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{y}} \rangle + \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle^2) - \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{z}} \rangle + \boldsymbol{c}\boldsymbol{t}_4 &= -\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle (2\boldsymbol{v}_i - 1) \cdot \boldsymbol{c} + \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle^2, \end{aligned}$$

where $\boldsymbol{t}_{2i}$ is the ith element $\vec{\boldsymbol{t}}_2$, the verifier can verify whether the quadratic term of $\varphi_i(\boldsymbol{c})$ is zero according to the received messages. We note that the prover is able to aggregate all above messages and send them together at once to the verifier.

Below we describe the concrete protocol in Algorithm 7. Define three hash functions: $H_1 : \{0,1\}^* \mapsto \mathcal{R}_q^m \times \mathcal{R}_q^m$, $H_2 : \{0,1\}^* \mapsto \mathcal{M}_q^{k\varpi} \times \mathcal{M}_q^{\varpi}$ and $H_3 : \{0,1\}^* \mapsto \mathcal{R}_q^{n'+1}$. Let

$$\mathcal{G} := \{\boldsymbol{g} \in \mathcal{R}_q \mid g_0 = \cdots = g_{\frac{d}{\varpi}-1} = 0\} \tag{5}$$

be the polynomials with the first d/ϖ coefficients equal to 0. Let

$$\begin{aligned} \mathfrak{f}_0 &= \langle (\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{\gamma}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{z}} \rangle - \boldsymbol{c}(\langle \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{x}} \rangle - \langle \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{\gamma}}_1 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{t}_3 - \boldsymbol{h}), \\ \mathfrak{f}_1 &= \sum_{i=1}^{n'} \boldsymbol{\alpha}_i (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}} \rangle - \boldsymbol{c}\boldsymbol{t}_{2i}) \cdot (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}} \rangle - \boldsymbol{c}\boldsymbol{t}_{2i} + \boldsymbol{c}), \end{aligned} \tag{6}$$

be two elements in $\mathcal{R}_q$, where $\vec{\boldsymbol{b}}_{2i}$ is the ith row vector of $\boldsymbol{B}_2$ and $\boldsymbol{t}_{2i}$ is the ith element $\vec{\boldsymbol{t}}_2$. The algorithm Rej and distributions $\mathbb{D}_\sigma$, $D(S_c)$ are described in Sect. 2.5.

Algorithm 7 Lattice-based Linear t-out-of-n Proof Protocol

Prove$((\boldsymbol{P}, t, ck, com), (\vec{v}, \vec{\boldsymbol{r}}))$:
Prover:
1: $(\vec{\boldsymbol{b}}_3, \vec{\boldsymbol{b}}_4) = H_1(\boldsymbol{P}, t, ck, com)$
2: $\boldsymbol{g} \leftarrow \mathcal{G}, \boldsymbol{t}_3 = \langle \vec{\boldsymbol{b}}_3, \vec{\boldsymbol{r}} \rangle + \boldsymbol{g}$
3: $(\vec{\gamma}_1, \vec{\gamma}_2) = H_2(\vec{\boldsymbol{b}}_4, \boldsymbol{t}_3)$
4: $\vec{x} = F_1^T \vec{\gamma}_1 + F_2^T \vec{\gamma}_2, \vec{\boldsymbol{x}} = \mathsf{NTT}^{-1}(\vec{x})$
5: $\boldsymbol{h} = \langle \vec{v}, \vec{\boldsymbol{x}} \rangle - \langle \vec{\boldsymbol{p}}, \vec{\boldsymbol{\gamma}}_1 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{g}$
6: $\vec{\boldsymbol{y}} \leftarrow \mathbb{D}_\sigma^{dm}, \vec{\boldsymbol{w}} = \boldsymbol{B}_0 \vec{\boldsymbol{y}}$
7: $(\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{n'}) = H_3(\vec{\boldsymbol{x}}, \boldsymbol{h}, \vec{\boldsymbol{w}})$
8: $\boldsymbol{\Psi} = \boldsymbol{\alpha}_0 \langle (\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{\gamma}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{y}} \rangle + \sum_{i=1}^{n'} \boldsymbol{\alpha}_i \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle (2v_i - 1)$
9: $\boldsymbol{t}_4 = \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{r}} \rangle - \boldsymbol{\Psi}$
10: $\boldsymbol{\Omega} = \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{y}} \rangle + \sum_{i=1}^{n'} \boldsymbol{\alpha}_i \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}} \rangle^2$
11: The prover sends messages $\omega = \{\boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \vec{\boldsymbol{w}}, \boldsymbol{\Omega}\}$ to the verifier
Verifier:
12: $\boldsymbol{c} \leftarrow D(S_c)$
13: The verifier sends $\boldsymbol{c}$ to the prover
Prover:
14: $\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \boldsymbol{c}\vec{\boldsymbol{r}}$
15: If $\mathsf{Rej}(\vec{\boldsymbol{z}}, \boldsymbol{c}\vec{\boldsymbol{r}}, \sigma) = 0$ abort
16: Send $\vec{\boldsymbol{z}}$ to the verifier

Verify$((\boldsymbol{P}, t, ck, com), (\omega, \boldsymbol{c}, \vec{\boldsymbol{z}}))$:
1: $(\vec{\boldsymbol{b}}_3, \vec{\boldsymbol{b}}_4) = H_1(\boldsymbol{P}, t, ck, com)$
2: $(\vec{\gamma}_1, \vec{\gamma}_2) = H_2(\vec{\boldsymbol{b}}_4, \boldsymbol{t}_3)$
3: $\vec{x} = F_1^T \vec{\gamma}_1 + F_2^T \vec{\gamma}_2, \vec{\boldsymbol{x}} = \mathsf{NTT}^{-1}(\vec{x})$
4: $(\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{n'}) = H_3(\vec{\boldsymbol{x}}, \boldsymbol{h}, \vec{\boldsymbol{w}})$
5: Compute $\mathfrak{f}_0, \mathfrak{f}_1$ defined in (6)
6: Check if $\boldsymbol{h} \in \mathcal{G}$ and $\vec{\boldsymbol{w}} = \boldsymbol{B}_0 \vec{\boldsymbol{z}} - \boldsymbol{c}\vec{\boldsymbol{t}}_0$
7: Check if $\|\vec{\boldsymbol{z}}\|_2 \le \sigma\sqrt{2dm}$
8: Check if $\boldsymbol{\Omega} = \mathfrak{f}_1 - \boldsymbol{\alpha}_0 \boldsymbol{c} \cdot \mathfrak{f}_0 + \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{z}} \rangle - \boldsymbol{c}\boldsymbol{t}_4$
9: If failed **return** 0, else **return** 1

5.1.1 Security analysis

We now present the completeness, special soundness and special honest-verifier zero-knowledge claims for the protocol.

Theorem 6 *Let* $\|\boldsymbol{c}\vec{\boldsymbol{r}}\| \le T$ *and* $2^{10} < dm$, *then Algorithm 7 has a completeness with error* $1 - \frac{1}{M}$, *where* $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$.

Proof It follows directly from Lemma 6 that the honest prover does not abort in the Step 15 with probability at least $\frac{1-2^{-100}}{M} \approx \frac{1}{M}$. Next, we show that when the prover does not abort, the verifier will accept except with a negligible probability.

First, by Lemma 6, the statistical distance between $\vec{\boldsymbol{z}}$ and $\mathbb{D}_\sigma^{dm}$ is at most $2^{-100}/M$. Thus, by Lemma 5, we have $\|\vec{\boldsymbol{z}}\| \le \sigma\sqrt{2dm}$ with probability at lest $1 - \frac{2^{-100}}{M} - 2^{-0.11dm}$, which is overwhelming under property parameter selection.

Next, by Lemma 2 and Lemma 1, we have

$$\begin{aligned}\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}) &= \langle \vec{v}, \vec{x} \rangle - \langle \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1 \rangle - \langle \vec{e}_t, \vec{\gamma}_2 \rangle + \sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{g}) \\ &= \langle \vec{v}, F_1^T \vec{\gamma}_1 + F_2^T \vec{\gamma}_2 \rangle - \langle \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1 \rangle - \langle \vec{e}_t, \vec{\gamma}_2 \rangle \\ &= \langle F_1 \vec{v}, \vec{\gamma}_1 \rangle - \langle \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1 \rangle + \langle F_2 \vec{v}, \vec{\gamma}_2 \rangle - \langle \vec{e}_t, \vec{\gamma}_2 \rangle \\ &= \langle F_1 \vec{v} - \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1 \rangle + \langle F_2 \vec{v} - \vec{e}_t, \vec{\gamma}_2 \rangle = 0.\end{aligned}$$

Thus, by Lemma 1 again, we have $\boldsymbol{h} \in \mathcal{G}$. Since we have $\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \boldsymbol{c}\vec{\boldsymbol{r}}$, $\vec{\boldsymbol{t}}_0 = \boldsymbol{B}_0 \vec{\boldsymbol{r}}$ and $\vec{\boldsymbol{w}} = \boldsymbol{B}_0 \vec{\boldsymbol{y}}$, the equation $\vec{\boldsymbol{w}} = \boldsymbol{B}_0 \vec{\boldsymbol{z}} - \boldsymbol{c}\vec{\boldsymbol{t}}_0$ always holds.

Finally, we show that, the $\boldsymbol{\Omega}$ generated in the **Prove** stage is equal to that generated in the **Verify** stage. We observe that

$$\langle \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{x}} \rangle - \langle \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{\gamma}}_1 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{t}_3 - \boldsymbol{h} = \langle (\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{\gamma}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{r}} \rangle.$$

Thus, by calculation, we have

$$\begin{aligned}
\mathfrak{f}_0 &= \langle(\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{y}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}(\langle \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{x}}\rangle - \langle \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{y}}_1\rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{t}_3 - \boldsymbol{h}) \\
&= \langle(\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{y}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{y}}\rangle, \\
\mathfrak{f}_1 &= \sum_{i=1}^{n'} \boldsymbol{\alpha}_i (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_{2i}) \cdot (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_{2i} + \boldsymbol{c}) \\
&= \sum_{i=1}^{n'} \boldsymbol{\alpha}_i (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}\rangle^2 - \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}\rangle (2\boldsymbol{v}_i - 1) \cdot \boldsymbol{c}),
\end{aligned}$$

and

$$\langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_4 = \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{y}}\rangle - \boldsymbol{c}(\boldsymbol{\alpha}_0 \langle (\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{y}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{y}}\rangle + \sum_{i=1}^{n'} \boldsymbol{\alpha}_i \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}\rangle (2\boldsymbol{v}_i - 1)).$$

Thus, we have

$$\boldsymbol{\Omega} = \mathfrak{f}_1 - \boldsymbol{\alpha}_0 \boldsymbol{c} \mathfrak{f}_0 + \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_4 = \sum_{i=1}^{n'} \boldsymbol{\alpha}_i \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}\rangle^2 + \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{y}}\rangle.$$

Therefore, our protocol satisfies completeness. □

Theorem 7 (Soundness) *Let* $q^{-d/\varpi} \leq 2^{-128}$, $\beta = \sigma\sqrt{2dm}$, *and define sets* $S_{fac} = \{\boldsymbol{c} \in \mathcal{R}_q^\times \mid \|\boldsymbol{c}\|_\infty \leq 2\}$ *and* $S'_r = \{\vec{\boldsymbol{r}} \in \mathcal{R}_q^m \mid \|\vec{\boldsymbol{r}}\| \leq 2\beta\}$, *assuming the* $\mathsf{MSIS}_{k_1,8d\beta}$ *problem is hard, then Algorithm 7 has* 3*-special soundness.*

Proof Assuming that we have three valid transcripts $((\boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \vec{\boldsymbol{w}}, \boldsymbol{\Omega}), \boldsymbol{c}_0, \vec{\boldsymbol{z}}_0)$, $((\boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \vec{\boldsymbol{w}}, \boldsymbol{\Omega}), \boldsymbol{c}_1, \vec{\boldsymbol{z}}_1)$ and $((\boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \vec{\boldsymbol{w}}, \boldsymbol{\Omega}), \boldsymbol{c}_2, \vec{\boldsymbol{z}}_2)$. Here, $\boldsymbol{c}_0, \boldsymbol{c}_1, \boldsymbol{c}_2$ are pairwise distinct. We will construct an extractor to extract a relaxed witness w.

Let $\bar{\boldsymbol{c}}_1 = \boldsymbol{c}_1 - \boldsymbol{c}_0$ and $\bar{\boldsymbol{c}}_2 = \boldsymbol{c}_2 - \boldsymbol{c}_0$, from Lemma 3, we have that $\bar{\boldsymbol{c}}_1$ and $\bar{\boldsymbol{c}}_2$ are invertible over $\mathcal{R}_q$ with probability at least $1 - \varpi \cdot \mathbf{p}^{d/\varpi}$, where $\mathbf{p}$ is the maximum probability over $\mathbb{Z}_q$ of the coefficients $\boldsymbol{c} \mod (X^{d/\varpi} - \zeta^{2i+1})$, which is close to $\frac{1}{q}$. Thus, with overwhelming probability $\bar{\boldsymbol{c}}_1$ and $\bar{\boldsymbol{c}}_2$ are invertible. And by the definition of the distribution $D(S_c)$, we have $\|\bar{\boldsymbol{c}}_1\|_\infty \leq 2$ and $\|\bar{\boldsymbol{c}}_2\|_\infty \leq 2$. Thus, $\bar{\boldsymbol{c}}_1, \bar{\boldsymbol{c}}_2 \in S_{fac}$ clearly holds.

For $i = 1, 2$, define $\vec{\boldsymbol{z}}_i^* = \vec{\boldsymbol{z}}_i - \vec{\boldsymbol{z}}_0, \vec{\boldsymbol{r}}_i = \bar{\boldsymbol{c}}_i^{-1} \cdot \vec{\boldsymbol{z}}_i^*$, then we have $\|\bar{\boldsymbol{c}}_i \vec{\boldsymbol{r}}_i\| \leq 2\beta$. Let $\vec{\boldsymbol{p}}_i = \vec{\boldsymbol{t}}_1 - \boldsymbol{B}_1 \vec{\boldsymbol{r}}_i, \vec{\boldsymbol{v}}_i = \vec{\boldsymbol{t}}_2 - \boldsymbol{B}_2 \vec{\boldsymbol{r}}_i, \boldsymbol{g}_i = \boldsymbol{t}_3 - \langle \vec{\boldsymbol{b}}_3, \vec{\boldsymbol{r}}_i\rangle, \boldsymbol{\Psi}_i = \boldsymbol{t}_4 - \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{r}}_i\rangle$, then $(\boldsymbol{c}_i, \vec{\boldsymbol{r}}_i, (\vec{\boldsymbol{p}}_i, \vec{\boldsymbol{v}}_i, \vec{\boldsymbol{g}}_i, \vec{\boldsymbol{\Psi}}_i))$ are two weak opening of the commitment $(\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2, \boldsymbol{t}_3, \boldsymbol{t}_4)$. Thus, by Theorem 5, assuming the $\mathsf{MSIS}_{k_1,8d\beta}$ problem is hard, we have $(\vec{\boldsymbol{p}}_1, \vec{\boldsymbol{v}}_1, \vec{\boldsymbol{g}}_1, \vec{\boldsymbol{\Psi}}_1) = (\vec{\boldsymbol{p}}_2, \vec{\boldsymbol{v}}_2, \vec{\boldsymbol{g}}_2, \vec{\boldsymbol{\Psi}}_2)$.

For $i = 0, 1, 2$, define $\vec{\boldsymbol{y}}_i = \vec{\boldsymbol{z}}_i - \boldsymbol{c}_i \vec{\boldsymbol{r}}_1$, then $\boldsymbol{B}_0 \vec{\boldsymbol{y}}_i = \vec{\boldsymbol{w}}$. Below, we prove that they are all equal. If $\vec{\boldsymbol{y}}_i \neq \vec{\boldsymbol{y}}_j$, then we have $\boldsymbol{B}_0 \cdot \bar{\boldsymbol{c}}_1 (\vec{\boldsymbol{y}}_i - \vec{\boldsymbol{y}}_j) = 0$ and

$$\|\bar{\boldsymbol{c}}_1 (\vec{\boldsymbol{y}}_i - \vec{\boldsymbol{y}}_j)\| \leq \|\bar{\boldsymbol{c}}_1 (\vec{\boldsymbol{z}}_i - \vec{\boldsymbol{z}}_j)\| + \|(\boldsymbol{c}_i - \boldsymbol{c}_j)(\vec{\boldsymbol{z}}_1 - \vec{\boldsymbol{z}}_0)\| \leq 8\|\boldsymbol{c}_1 \vec{\boldsymbol{z}}_1\| \leq 8d\beta.$$

Thus, assuming the $\mathsf{MSIS}_{k_1,8d\beta}$ problem is hard, we have $\vec{\boldsymbol{y}}_0 = \vec{\boldsymbol{y}}_1 = \vec{\boldsymbol{y}}_2$. Since $\vec{\boldsymbol{p}}_i, \vec{\boldsymbol{v}}_i, \vec{\boldsymbol{g}}_i, \vec{\boldsymbol{\Psi}}_i$ and $\vec{\boldsymbol{y}}_i$ are equal, in the following calculations, we will directly omit their subscripts.

By substituting $\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{y}} + \boldsymbol{c}_i \vec{\boldsymbol{r}}_1$ into the following equation, we can obtain

$$\begin{aligned}
&\langle(\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{y}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{z}}_i\rangle \\
&= \langle(\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{y}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{y}}\rangle + \boldsymbol{c}_i (\langle \boldsymbol{B}_2 \vec{\boldsymbol{r}}_1, \vec{\boldsymbol{x}}\rangle - \langle \boldsymbol{B}_1 \vec{\boldsymbol{r}}_1, \vec{\boldsymbol{y}}_1\rangle + \langle \vec{\boldsymbol{b}}_3, \vec{\boldsymbol{r}}_1\rangle).
\end{aligned}$$

Thus, we can get:

$$\begin{aligned}\mathfrak{f}_0 &= \langle(\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{\gamma}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{z}}_i\rangle - \boldsymbol{c}_i(\langle \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{x}}\rangle - \langle \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{\gamma}}_1\rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{t}_3 - \boldsymbol{h}) \\ &= \langle(\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{\gamma}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{y}}\rangle - \boldsymbol{c}_i(\boldsymbol{f} - \boldsymbol{g} - \boldsymbol{h}),\end{aligned}$$

where $\boldsymbol{f} = \langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}}\rangle - \langle \vec{\boldsymbol{p}}, \vec{\boldsymbol{\gamma}}_1\rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2$, and

$$\begin{aligned}\mathfrak{f}_1 &= \sum_{i=1}^{n'} \boldsymbol{\alpha}_i(\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}}_i\rangle - \boldsymbol{c}_i \boldsymbol{t}_{2i}) \cdot (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}}_i\rangle - \boldsymbol{c}_i \boldsymbol{t}_{2i} + \boldsymbol{c}_i) \\ &= \sum_{i=1}^{n'} \boldsymbol{\alpha}_i(\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}\rangle - \boldsymbol{c}_i \boldsymbol{v}_i) \cdot (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}\rangle - \boldsymbol{c}_i(\boldsymbol{v}_i - 1)) \\ &= \left[\sum_{i=1}^{n'} \boldsymbol{\alpha}_i \boldsymbol{v}_i(\boldsymbol{v}_i - 1)\right] \cdot \boldsymbol{c}_i^2 + \left[\sum_{i=1}^{n'} \boldsymbol{\alpha}_i \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}(1 - 2\boldsymbol{v}_i)\right] \cdot \boldsymbol{c}_i + \sum_{i=1}^{n'} \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}\rangle^2.\end{aligned}$$

Let $\mathfrak{f} = \mathfrak{f}_1 - \boldsymbol{\alpha}_0 \boldsymbol{c} \cdot \mathfrak{f}_0 + (\langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_4) - \boldsymbol{\Omega}$, then $\mathfrak{f}$ can be viewed as a quadratic function in $\boldsymbol{c}$:

$$\mathfrak{f} = \mathfrak{f}(\boldsymbol{c}) = \boldsymbol{\mu}_2 \boldsymbol{c}^2 + \boldsymbol{\mu}_1 \boldsymbol{c} + \boldsymbol{\mu}_0, \tag{7}$$

where the coefficients of $\mathfrak{f}$ are determined by $\vec{\boldsymbol{p}}, \vec{\boldsymbol{v}}, \vec{\boldsymbol{g}}, \vec{\boldsymbol{\Psi}}, \vec{\boldsymbol{y}}$, which are independent of $\boldsymbol{c}$ and

$$\boldsymbol{\mu}_2 = \boldsymbol{\alpha}_0(\boldsymbol{f}^\sharp - \boldsymbol{g}^\sharp - \boldsymbol{h}) + \sum_{i=1}^{n'} \boldsymbol{\alpha}_i \boldsymbol{v}_i^\sharp(\boldsymbol{v}_i^\sharp - 1).$$

Note that, for all $\boldsymbol{c}_i$, we have $\mathfrak{f}(\boldsymbol{c}_i) = 0$. Thus, we can alternatively write these three equations as follows:

$$\begin{bmatrix} 1 & \boldsymbol{c}_0 & \boldsymbol{c}_0^2 \\ 1 & \boldsymbol{c}_1 & \boldsymbol{c}_1^2 \\ 1 & \boldsymbol{c}_2 & \boldsymbol{c}_2^2 \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\mu}_0 \\ \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

Since the difference of each two challenges in $\{\boldsymbol{c}_0, \boldsymbol{c}_1, \boldsymbol{c}_2\}$ is invertible over $\mathcal{R}_q$, we must have $\boldsymbol{\mu}_2 = 0$.

Next, in the random oracle model, $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{n'}$ are independent of $\boldsymbol{f} - \boldsymbol{g} - \boldsymbol{h}$ and $\boldsymbol{v}_i$. We show that $\boldsymbol{f} - \boldsymbol{g} - \boldsymbol{h} = 0$ and $\boldsymbol{v}_i(\boldsymbol{v}_i - 1) = 0$ for all $i \in [n']$. Assume that $\boldsymbol{f} - \boldsymbol{g} - \boldsymbol{h} \neq 0$, then for uniform $\boldsymbol{\alpha}_0$, the probability that $\boldsymbol{\mu}_2 = 0$ is at most $q^{d/\varpi}$, which is negligible. Hence, we have $\boldsymbol{f} - \boldsymbol{g} - \boldsymbol{h} = 0$. Similarly, we have $\boldsymbol{v}_i(\boldsymbol{v}_i - 1) = 0$ for all $i \in [n']$. And by this, we have $\mathsf{NTT}(\vec{\boldsymbol{v}}) \in \{0, 1\}^n$.

Finally, we show that $\|\mathsf{NTT}(\vec{\boldsymbol{v}})\|_1 = t$ and $\boldsymbol{P} \cdot \mathsf{NTT}(\vec{\boldsymbol{v}}) = \vec{\boldsymbol{p}}$. Since we have $\boldsymbol{f} - \boldsymbol{g} = \boldsymbol{h}$ and $\boldsymbol{h} \in \mathcal{G}$. Thus, by Lemma 2, we have $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{f}) = \sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{g})$. On the other hand, by Lemma 1 and Lemma 2, we have

$$\begin{aligned}&\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{f}) \\ &= \langle F_1 \cdot \mathsf{NTT}(\vec{\boldsymbol{v}}), \vec{\gamma}_1\rangle + \langle F_2 \cdot \mathsf{NTT}(\vec{\boldsymbol{v}}), \vec{\gamma}_2\rangle - \langle \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1\rangle - \langle \vec{e}_t, \vec{\gamma}\rangle \\ &= \langle F_1 \cdot \mathsf{NTT}(\vec{\boldsymbol{v}}) - \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1\rangle + \langle F_2 \cdot \mathsf{NTT}(\vec{\boldsymbol{v}}) - \vec{e}_t, \vec{\gamma}\rangle.\end{aligned}$$

If $F_1 \cdot \mathsf{NTT}(\vec{v}) - \mathsf{NTT}(\vec{p}) \neq 0$ or $F_2 \cdot \mathsf{NTT}(\vec{v}) - \vec{e}_t \neq 0$, then $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{f})$ is a uniformly random polynomial in $\mathcal{M}_q$ by randomness of $\vec{\gamma}_1, \vec{\gamma}_2$. In this time, the probability that $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{f}) = \sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{g})$ is $q^{-d/\varpi}$, which is negligible. Hence, we have $F_1 \cdot \mathsf{NTT}(\vec{v}) - \mathsf{NTT}(\vec{p}) = 0$ and $F_2 \cdot \mathsf{NTT}(\vec{v}) - \vec{e}_t = 0$.

In conclusion, $w = (\vec{v}, \vec{r}_1, \tilde{c}_1)$ is a relaxed witness. □

Theorem 8 (One-time Zero-knowledge) *Let* $\|\boldsymbol{c}\vec{\boldsymbol{r}}\| \leq T$, $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$ *and assuming the* $\mathsf{MLWE}_{m-\tilde{k}-2,\sigma'}$ *problem is hard, then Algorithm 7 satisfies one-time zero-knowledge.*

Proof Assuming the protocol is not aborted, we construct a simulator $\mathcal{S}$ that takes $x = (\boldsymbol{P}, t, ck, com)$ and challenge $\boldsymbol{c}$ as inputs and outputs a valid transcript.

The simulator $\mathcal{S}$ first sets $\boldsymbol{t}_3, \boldsymbol{t}_4 \leftarrow \mathcal{R}_q$ and $\boldsymbol{h} \leftarrow \mathcal{G}$. Then $\mathcal{S}$ samples $\vec{z} \leftarrow \mathbb{D}_{\sigma}^{dm}$. Next, $\mathcal{S}$ computes $\vec{\boldsymbol{w}} = \boldsymbol{B}_0\vec{z} - \boldsymbol{c}\vec{\boldsymbol{t}}_0$. Subsequently, $\mathcal{S}$ computes $(\vec{b}_3, \vec{b}_4) = H_1(\boldsymbol{P}, t, ck, com)$, $(\vec{\gamma}_1, \vec{\gamma}_2) = H_2(\vec{b}_4, \boldsymbol{t}_3)$, $\vec{x} = F_1^T\vec{\gamma}_1 + F_2^T\vec{\gamma}_2$, $\vec{\boldsymbol{x}} = \mathsf{NTT}^{-1}(\vec{x})$, $(\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{n'}) = H_3(\vec{\boldsymbol{x}}, \boldsymbol{h}, \boldsymbol{w})$, $\mathfrak{f}_1 = \sum_{i=1}^{n'} \boldsymbol{\alpha}_i(\langle \vec{\boldsymbol{b}}_{2i}, \vec{z}\rangle - \boldsymbol{c}\boldsymbol{t}_{2i}) \cdot (\langle \vec{\boldsymbol{b}}_{2i}, \vec{z}\rangle - \boldsymbol{c}\boldsymbol{t}_{2i} + \boldsymbol{c})$ and $\mathfrak{f}_0 = \langle (\vec{\boldsymbol{x}}^T\boldsymbol{B}_2 - \vec{\boldsymbol{\gamma}}_1^T\boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{z}\rangle - \boldsymbol{c}(\langle \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{x}}\rangle - \langle \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{\gamma}}_1\rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{t}_3 - \boldsymbol{h})$. Finally $\mathcal{S}$ computes $\boldsymbol{\Omega} = \mathfrak{f}_1 - \boldsymbol{\alpha}_0\boldsymbol{c} \cdot \mathfrak{f}_0 + (\langle \vec{\boldsymbol{b}}_4, \vec{z}\rangle - \boldsymbol{c}\boldsymbol{t}_4)$.

The distribution of simulated $\vec{z}$ is statistically close to the real distribution by Lemma 6. And the distribution of simulated $\boldsymbol{h}$ is the same as the real distribution. Conditioned on $(\vec{z}, \boldsymbol{h}, \boldsymbol{c})$ and $(\boldsymbol{t}_3, \boldsymbol{t}_4)$, variables $\vec{\boldsymbol{w}}$ and $\boldsymbol{\Omega}$ are uniquely determined from the verification equations. Thus, the distribution of simulated $(\vec{\boldsymbol{w}}, \boldsymbol{\Omega})$ is also the same as the real distribution. Finally, the distribution of simulated $(\boldsymbol{t}_3, \boldsymbol{t}_4)$ is computationally indistinguishable from the real one by the assumption of $\mathsf{MLWE}_{m-\tilde{k}-2,\sigma'}$ problem.

Therefore, our t-out-of-n proof protocol satisfies one-time zero-knowledge. □

5.2 Logarithmic *t*-out-of-*n* proof protocol

In this subsection, we will construct a succinct lattice-based t-out-of-n proof protocol, whose proof size is $O(t \log n)$. Operations are performed over a cyclotomic ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ in the protocol. For security, we require $\mathcal{R}_q$ can be split into exactly ϖ factors such that $q^{-d/\varpi}$ is negligible, see Sect. 2.4 for more details. According to the definition, the t-out-of-n proof protocol is related to the applied commitment scheme. In this subsection, the applied commitment scheme is also the scheme described in Algorithm 6.

Let t, n be two positive numbers, and $\vec{v} \in \{0, 1\}^n$ be a binary vector with a Hamming weight of exactly t. Since the input message of Algorithm 6 needs to be vectors of ring elements, we now describe how to encode $\vec{v}$. Let I represent the set of coordinates corresponding to the elements in $\vec{v}$ that are equal to 1. Then, $\vec{v}$ can be expressed as $\vec{v} = \sum_{i \in I} \vec{e}_i$, where $\vec{e}_i \in \{0, 1\}^n$ has exactly one 1 in the ith coefficient. Assuming that $n = \varpi^{n'}$, then any vector $\vec{e}_i$ can be uniquely decomposed into smaller vectors $\vec{v}_{i,1}, \ldots \vec{v}_{i,n'} \in \{0, 1\}^{\varpi}$, each having exactly one 1, and $\vec{e}_i = \vec{v}_{i,1} \otimes \cdots \otimes \vec{v}_{i,n'}$. After renumbering, we have $\vec{v} = \sum_{i \in [t]} (\vec{v}_{i,1} \otimes \cdots \otimes \vec{v}_{i,n'})$. Therefore, the encoding of $\vec{v}$ is defined as

$$\mathsf{Encode}(\vec{v}) := (\boldsymbol{v}_{1,1}, \ldots, \boldsymbol{v}_{1,n'}, \ldots, \boldsymbol{v}_{t,1}, \ldots, \boldsymbol{v}_{t,n'}) \in \mathcal{R}_q^{tn'},$$

where $\boldsymbol{v}_{i,j} = \mathsf{NTT}^{-1}(\vec{v}_{i,j})$. This encoding method uses only $O(t \log n)$ ring elements to uniquely represent the vector $\vec{v}$.

Let $\boldsymbol{P} = \{\vec{\boldsymbol{p}}_1, \ldots, \vec{\boldsymbol{p}}_n\} \subset \mathcal{R}_q^k$ be a public set, $ck = \bar{\boldsymbol{B}} \in \mathcal{R}_q^{k' \times m}$ be a commitment key and $\vec{\boldsymbol{r}} \in S_r$ be a randomness following the distribution $\mathbb{D}_{\sigma'}^{dm}$, then the commitment of $(\boldsymbol{P}\vec{v}, \vec{v})$ is:

$$\vec{\boldsymbol{t}} = \bar{\boldsymbol{B}}\vec{\boldsymbol{r}} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{P}\vec{v} \\ \mathsf{Encode}(\vec{v}) \end{bmatrix}.$$

For convenience, we divide $\bar{\boldsymbol{B}}$ into three parts by row: $\boldsymbol{B}_0$ represents the first $k_1 = k' - k - tn'$ rows, $\boldsymbol{B}_1$ represents the middle k rows, and $\boldsymbol{B}_2$ represents the last tn' row. Similarly, $\vec{\boldsymbol{t}}$ also be divided as $(\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2)$, such that

$$\vec{\boldsymbol{t}}_0 = \boldsymbol{B}_0\vec{\boldsymbol{r}},\ \vec{\boldsymbol{t}}_1 = \boldsymbol{B}_1\vec{\boldsymbol{r}} + \boldsymbol{P} \cdot \vec{v},\ \vec{\boldsymbol{t}}_2 = \boldsymbol{B}_2\vec{\boldsymbol{r}} + \mathsf{Encode}(\vec{v}).$$

We now instantiate the relations defined in Definition 23. For the convenience of subsequent t-out-of-n proof, we replace the vector $\vec{v}$ with its encoding form $\vec{\boldsymbol{v}} = (\boldsymbol{v}_{1,1}, \ldots, \boldsymbol{v}_{t,n'}) \in \mathcal{R}_q^{tn'}$ in the relation.

Definition 26 Let t, n be positive numbers, $S_{fac} = \{\boldsymbol{c} \in \mathcal{R}_q^{\times} \mid \|\boldsymbol{c}\|_\infty \leq 2\}$, $S_r = \{\vec{\boldsymbol{r}} \in \mathcal{R}_q^m \mid \|\vec{\boldsymbol{r}}\| \leq \beta_1\}$ and $S_r' = \{\vec{\boldsymbol{r}} \in \mathcal{R}_q^m \mid \|\vec{\boldsymbol{r}}\| \leq \beta_2\}$, we define the following two relations:

$$\mathfrak{R} = \left\{ (\boldsymbol{P}, t, \bar{\boldsymbol{B}}, \vec{\boldsymbol{t}}), (\vec{\boldsymbol{v}}, \vec{\boldsymbol{r}}) \;\middle|\; \begin{array}{l} \text{for all } i \in [t],\ j \in [n'] : \\ \quad \mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi},\ \|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1 \\ \wedge \text{ for all } i_1, i_2 \in [t] \text{ with } i_1 \neq i_2 : \\ \quad \|\mathsf{NTT}\left(\sum_{j \in [n']} \boldsymbol{v}_{i_1,j} \cdot \boldsymbol{v}_{i_2,j}\right)\|_1 < n' \\ \wedge\ \vec{\boldsymbol{p}} = \boldsymbol{P} \cdot \sum_{i \in [t]} (\mathsf{NTT}(\vec{\boldsymbol{v}}_{i1}) \otimes \cdots \otimes \mathsf{NTT}(\vec{\boldsymbol{v}}_{in'})) \\ \wedge\ \vec{\boldsymbol{r}} \in S_r \wedge \vec{\boldsymbol{t}}_0 = \boldsymbol{B}_0\vec{\boldsymbol{r}}, \vec{\boldsymbol{t}}_1 = \boldsymbol{B}_1\vec{\boldsymbol{r}} + \vec{\boldsymbol{p}}, \vec{\boldsymbol{t}}_2 = \boldsymbol{B}_2\vec{\boldsymbol{r}} + \vec{\boldsymbol{v}} \end{array} \right\},$$

and

$$\mathfrak{R}' = \left\{ (\boldsymbol{P}, t, \bar{\boldsymbol{B}}, \vec{\boldsymbol{t}}), (\vec{\boldsymbol{v}}, \vec{\boldsymbol{r}}, \boldsymbol{c}) \;\middle|\; \begin{array}{l} \text{for all } i \in [t],\ j \in [n'] : \\ \quad \mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi},\ \|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1 \\ \wedge \text{ for all } i_1, i_2 \in [t] \text{ with } i_1 \neq i_2 : \\ \quad \|\mathsf{NTT}\left(\sum_{j \in [n']} \boldsymbol{v}_{i_1,j} \cdot \boldsymbol{v}_{i_2,j}\right)\|_1 < n' \\ \wedge\ \vec{\boldsymbol{p}} = \boldsymbol{P} \cdot \sum_{i \in [t]} (\mathsf{NTT}(\vec{\boldsymbol{v}}_{i1}) \otimes \cdots \otimes \mathsf{NTT}(\vec{\boldsymbol{v}}_{in'})) \\ \wedge\ \boldsymbol{c} \in S_{fac} \wedge \boldsymbol{c}\vec{\boldsymbol{r}} \in S_r' \\ \wedge \vec{\boldsymbol{t}}_0 = \boldsymbol{B}_0\vec{\boldsymbol{r}}, \vec{\boldsymbol{t}}_1 = \boldsymbol{B}_1\vec{\boldsymbol{r}} + \vec{\boldsymbol{p}}, \vec{\boldsymbol{t}}_2 = \boldsymbol{B}_2\vec{\boldsymbol{r}} + \vec{\boldsymbol{v}} \end{array} \right\}.$$

At first glance, the requirement for $\vec{\boldsymbol{v}}$ in the above relation may seem to be different from the requirement in Definition 23. However, we will now demonstrate that they are actually the same.

Let $\vec{v}_i = \mathsf{NTT}(\vec{\boldsymbol{v}}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\vec{\boldsymbol{v}}_{i,n'})$ for all $i \in [t]$, then since for any $i \in [t]$ and $j \in [n']$, we have $\mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi}$ and $\|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1$, it follows that $\vec{v}_1, \ldots, \vec{v}_t \in \{0,1\}^n$ have exactly one 1 each. Now, consider the condition $\|\mathsf{NTT}\left(\sum_{j \in [n']} \boldsymbol{v}_{i_1,j} \cdot \boldsymbol{v}_{i_2,j}\right)\|_1 < n'$. From this condition, we can deduce that $\vec{v}_1, \ldots, \vec{v}_t$ are distinct. This is because if there exists i_1, i_2 such that $\vec{v}_{i_1} = \vec{v}_{i_2}$, then

$$\|\mathsf{NTT}\left(\sum_{j \in [n']} \boldsymbol{v}_{i_1,j} \cdot \boldsymbol{v}_{i_2,j}\right)\|_1 = \sum_{j \in [n']} \langle \mathsf{NTT}(\boldsymbol{v}_{i_1,j}), \mathsf{NTT}(\boldsymbol{v}_{i_2,j}) \rangle = n',$$

which is contradictory. Thus, the vector $\vec{v} = \vec{v}_1 + \cdots + \vec{v}_t$ satisfies all the requirements stated in Definition 23.

5.2.1 Overview

Now we present how to construct the protocol. Just as we discussed in the introduction, the goal of the prover is to convince the verifier that the opening $(\vec{\boldsymbol{p}}, \vec{\boldsymbol{v}})$ of the commitment $(\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2)$ satisfies the following four conditions:

1. Each element $\boldsymbol{v}_{i,j}$ of $\vec{\boldsymbol{v}}$ satisfies $\mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi}$;
2. Each element $\boldsymbol{v}_{i,j}$ of $\vec{\boldsymbol{v}}$ satisfies $\|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1$;
3. For any $i_1, i_2 \in [t]$, with $i_1 \neq i_2$, the elements of $\vec{\boldsymbol{v}}$ satisfies

$$\|\mathsf{NTT}\left(\sum_{j\in[n']} \boldsymbol{v}_{i_1,j} \cdot \boldsymbol{v}_{i_2,j}\right)\|_1 < n';$$

4. $\vec{\boldsymbol{p}}$ and $\vec{\boldsymbol{v}}$ satisfies $\boldsymbol{P} \cdot \sum_{i\in[t]}(\mathsf{NTT}(\boldsymbol{v}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\boldsymbol{v}_{i,n'})) = \vec{\boldsymbol{p}}$.

Since the method for proving the 1st and 2nd conditions are common, here we mainly focus on explaining how to prove the 3rd and 4th conditions. Let $\mathcal{M}_q := \{\boldsymbol{p} \in \mathbb{Z}_q[X] : \deg(\boldsymbol{p}) < d/\varpi\}$ be the $\mathbb{Z}_q$-module of polynomials of degree less than d/ϖ. Assuming parameters satisfies $n' < 2^b < \min\{q, 2^{\varpi}\}$, then define

$$\mathfrak{g} = (1, 2, \ldots, 2^{b-1}, 0, \ldots, 0) \in \mathcal{M}_q^{\varpi}. \tag{8}$$

For any $i_1, i_2 \in [t]$ with $i_1 \neq i_2$, let

$$u_{i_1,i_2} = n' - \sum_{j\in[n']} \langle \vec{v}_{i_1,j}, \vec{v}_{i_2,j} \rangle - 1,$$

then represent u_{i_1,i_2} as a binary representation and extend it to a ϖ-length vector $\vec{u}_{i_1,i_2} \in \{0,1\}^{\varpi}$ such that $u_{i_1,i_2} = \langle \vec{u}_{i_1,i_2}, \mathfrak{g} \rangle$. It is worth noting that, when $\sum_{j\in[n']} \langle \vec{v}_{i_1,j}, \vec{v}_{i_2,j} \rangle = n'$, we have $u_{i_1,i_2} = -1$.[7] In this case, there does not exist a binary vector $\vec{u}$ such that $\langle \vec{u}, \mathfrak{g} \rangle = u_{i_1,i_2}$. So we first compute a commitment $\boldsymbol{t}_3^{i_1,i_2} = \boldsymbol{b}_3^{i_1,i_2} \vec{\boldsymbol{r}} + \mathsf{NTT}^{-1}(\vec{u}_{i_1,i_2})$, where $\boldsymbol{b}_3^{i_1,i_2}$ is a uniform vector. Then, proving the 3rd condition is equivalent to proving that the opening of $\vec{\boldsymbol{t}}_3^{i_1,i_2}$ is $\boldsymbol{u}_{i_1,i_2}$ which satisfies

$$\mathsf{NTT}(\boldsymbol{u}_{i_1,i_2}) \in \{0,1\}^{\varpi} \quad \text{and} \quad \|\mathsf{NTT}(\boldsymbol{u}'_{i_1,i_2})\|_1 = n' - 1,$$

where $\boldsymbol{u}'_{i_1,i_2} = \sum_{j\in[n']} \boldsymbol{v}_{i_1,j} \cdot \boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2} \mathsf{NTT}^{-1}(\mathfrak{g})$. And these problems can be proved using the methods described in [5, 23].

We now show how to adapt the method from [38] to prove the 4th condition. Let $P_1 = \mathsf{NTT}(\boldsymbol{P}) \in \mathcal{M}_q^{k\varpi \times \varpi^{n'}}$ and $\vec{p} = \mathsf{NTT}(\vec{\boldsymbol{p}}) \in \mathcal{M}_q^{k\varpi}$. It is worth noting that, when $\boldsymbol{P} \cdot \sum_{i\in[t]}(\mathsf{NTT}(\vec{\boldsymbol{v}}_{i,1}) \otimes \cdots \otimes \mathsf{NTT}(\vec{\boldsymbol{v}}_{i,n'})) \neq \vec{\boldsymbol{p}}$, the probability that

$$\langle P_1 \cdot \sum_{i\in[t]} (\vec{v}_{i,1} \otimes \cdots \otimes \vec{v}_{i,n'}) - \vec{p}, \vec{\gamma} \rangle = 0$$

[7] In the modulo q setting, $-1 = q - 1$.

is exactly $q^{-d/\varpi}$, where $\vec{\gamma}$ is a uniformly random vector on $\mathcal{M}_q^{k\varpi}$. Under proper parameters, this probability is negligible. And by calculations and Lemma 1, we have

$$\begin{aligned}\langle P_1 \cdot \sum_{i\in[t]}(\vec{v}_{i,1}\otimes\cdots\otimes\vec{v}_{i,n'}) - \vec{p},\vec{\gamma}\rangle &= \sum_{i\in[t]}\langle \vec{v}_{i,1}\otimes\cdots\otimes\vec{v}_{i,n'}, P_1^T\vec{\gamma}\rangle - \langle\vec{p},\vec{\gamma}\rangle \\ &= \sum_{i\in[t]}(\vec{v}_{i,1}, P_2\cdot(\vec{v}_{i,2}\otimes\cdots\otimes\vec{v}_{i,n'})) - \langle\vec{p},\vec{\gamma}\rangle,\end{aligned}$$

where $P_2 = \begin{bmatrix}\vec{\gamma}_1^T P_{1,1}\\ \vdots \\ \vec{\gamma}_1^T P_{1,\varpi}\end{bmatrix} \in \mathcal{M}_q^{\varpi\times\varpi^{n'-1}}$ and $P_1 = [P_{1,1},\ldots,P_{1,\varpi}]$. Let us define $\vec{x}_{1,i} = P_2\cdot(\vec{v}_{i,2}\otimes\cdots\otimes\vec{v}_{i,n'})$ and $\boldsymbol{x}_{1,i} = \mathsf{NTT}^{-1}(\vec{x}_{1,i})$. Then we compute commitments $\boldsymbol{t}_4^{1i} = \vec{\boldsymbol{b}}_4^{1i}\vec{\boldsymbol{r}} + \boldsymbol{x}_{1,i}$. In this case, proving the 4rd condition is equivalent to proving that the opening of $\boldsymbol{t}_4^{1i}$ is $\boldsymbol{x}_{1,i}$, which satisfy

$$\mathsf{NTT}(\boldsymbol{x}_{1,i}) = P_2\cdot(\vec{v}_{i,2}\otimes\cdots\otimes\vec{v}_{i,n'}) \text{ and } \sum_{i=0}^{\varpi}\mathsf{NTT}(\boldsymbol{h})_i = 0,$$

where $\boldsymbol{h} = \sum_{i\in[t]}\vec{\boldsymbol{v}}_{i,1}\cdot\boldsymbol{x}_{1,i} - \langle\vec{\boldsymbol{p}}, \mathsf{NTT}^{-1}(\vec{\gamma})\rangle$. Note that the former is exactly the set membership relation addressed in [38], while the latter can also be proved using the masking technique [23] again. Finally, amortization techniques [8, 10] will be used to further compress the proof size.

5.2.2 Concrete construction

Now we describe the concrete lattice-based logarithmic t-out-of-n proof protocol, which is presented in Algorithm 8. Let $\vec{u}_{i_1,i_2}\in\{0,1\}^{\varpi}$ be the binary representation of integer $n'-1-\sum_{j\in[n']}\langle\vec{v}_{i_1,j},\vec{v}_{i_2,j}\rangle$, $\boldsymbol{u}_{i_1,i_2} = \mathsf{NTT}^{-1}(\vec{u}_{i_1,i_2})$ and

$$\boldsymbol{u}'_{i_1,i_2} = \sum_{j\in[n']}\boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g}),$$

where $\mathfrak{g}$ is the vector defined in Eq. (8). Next, we define

$$\vec{\boldsymbol{u}} = (\boldsymbol{u}_{1,2},\ldots,\boldsymbol{u}_{t-1,t}),\ \vec{\boldsymbol{u}}' = (\boldsymbol{u}'_{1,2},\ldots,\boldsymbol{u}'_{t-1,t}) \in \mathcal{R}_q^{\frac{t(t-1)}{2}}. \tag{9}$$

Let $\boldsymbol{P}\in\mathcal{R}_q^{k\times\varpi^{n'}}$ be the public key set, define $P_1 = \mathsf{NTT}(\boldsymbol{P})\in\mathcal{M}_q^{k\varpi\times\varpi^{n'}}$ and denoted as $P_1 = [P_{1,1},\ldots,P_{1,\varpi}]$. Similarly, for any $P_j^i\in\mathcal{M}_q^{\varpi\times\varpi^{n'-j+1}}$, we denote $P_j^i = [P_{j,1}^i,\ldots,P_{j,\varpi}^i]$. For any $i\in[n']$, denote $\vec{\boldsymbol{x}}_i = (\boldsymbol{x}_{i,1},\ldots,\boldsymbol{x}_{i,t})\in\mathcal{R}_q^t$. We define

$$F = \begin{pmatrix}1&\cdots&1\\0&\cdots&0\\\vdots&\vdots&\vdots\\0&\cdots&0\end{pmatrix}\in\mathcal{M}_q^{\varpi\times\varpi} \text{ and } \vec{e}_1 = \begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}\in\mathcal{M}_q^{\varpi}. \tag{10}$$

And let $\mathcal{G}$ be the polynomials with the first d/ϖ coefficients equal to 0, i.e.

$$\mathcal{G} := \{\boldsymbol{g}\in\mathcal{R}_q \mid g_0 = \cdots = g_{\frac{d}{\varpi}-1} = 0\}.$$

We require the following hash functions:

- $H_1 : \{0,1\}^* \mapsto \mathcal{R}_q^{[t(\frac{t-1}{2}+n'-1)+2]\times m}$, for convenience, we express the output as the form: $(\boldsymbol{B}_3, \boldsymbol{B}_4, \vec{\boldsymbol{b}}_5, \vec{\boldsymbol{b}}_6)$ such that $\boldsymbol{B}_3 \in \mathcal{R}_q^{\frac{t(t-1)}{2}\times m}$, $\boldsymbol{B}_4 \in \mathcal{R}_q^{t(n'-1)\times m}$ and $\vec{\boldsymbol{b}}_5, \vec{\boldsymbol{b}}_6 \in \mathcal{R}_q^m$;
- $H_2 : \{0,1\}^* \mapsto \mathcal{M}_q^{k\varpi}$;
- $H_3 : \{0,1\}^* \mapsto \mathcal{M}_q^{t\varpi}$;
- $H_4 : \{0,1\}^* \mapsto \mathcal{M}_q^{t(n'+\frac{t-1}{2})\varpi}$, for convenience, we express the output as the following form: $(\vec{\gamma}^2, \vec{\gamma}^3)$ such that $\vec{\gamma}^2 \in \mathcal{M}_q^{tn'\varpi}$ and denote as $\vec{\gamma}^2 = (\vec{\gamma}^2_{1,1}, \ldots, \vec{\gamma}^2_{t,n'})$, $\vec{\gamma}^3 \in \mathcal{M}_q^{\frac{t(t-1)\varpi}{2}}$ and denote as $\vec{\gamma}^3 = (\vec{\gamma}^3_{1,2}, \ldots, \vec{\gamma}^2_{t-1,t})$;
- $H_5 : \{0,1\}^* \mapsto \mathcal{R}_q^{1+t(n'+\frac{t-1}{2})}$, we express the output as the following form: $(\boldsymbol{\alpha}_0, \vec{\boldsymbol{\alpha}}^1, \vec{\boldsymbol{\alpha}}^2)$ such that $\vec{\boldsymbol{\alpha}}^1 \in \mathcal{R}_q^{tn'}$ and denote as $\vec{\boldsymbol{\alpha}}^1 = (\boldsymbol{\alpha}^1_{1,1}, \ldots, \boldsymbol{\alpha}^1_{t,n'})$, $\vec{\boldsymbol{\alpha}}^2 \in \mathcal{R}_q^{\frac{t(t-1)}{2}}$ and denote as $\vec{\boldsymbol{\alpha}}^2 = (\boldsymbol{\alpha}^1_{1,2}, \ldots, \boldsymbol{\alpha}^1_{t-1,t})$.

Let $\vec{\boldsymbol{b}}^2_{i,j}$ denote the vector in $\boldsymbol{B}_2$ used to calculate the commitment of $\boldsymbol{v}_{i,j}$ (specifically, the the $[(i-1)n'+j]$th row vector in $\boldsymbol{B}_2$), and denote this commitment as $\boldsymbol{t}^2_{i,j}$. Similarly, let $\vec{\boldsymbol{b}}^3_{i_1,i_2}$ denote the vector in $\boldsymbol{B}_3$ used to calculate the commitment $\boldsymbol{t}^3_{i_1,i_2}$ of $\boldsymbol{u}_{i_1,i_2}$ and let $\vec{\boldsymbol{b}}^4_{i,j}$ denote the vector in $\boldsymbol{B}_4$ used to calculate the commitment $\boldsymbol{t}^4_{i,j}$ of $\boldsymbol{x}_{i,j}$. Then, we define the following elements:

$$
\begin{aligned}
\boldsymbol{\Phi}_1 = {} & \langle \vec{\boldsymbol{b}}_5, \vec{\boldsymbol{y}} \rangle - \langle \boldsymbol{B}_1 \vec{\boldsymbol{y}}, \mathsf{NTT}^{-1}(\vec{\gamma}_1) \rangle \\
& + \sum_{i\in[t]} \sum_{j\in[n']} \langle \vec{\boldsymbol{b}}^2_{i,j}, \vec{\boldsymbol{y}} \rangle \cdot \mathsf{NTT}^{-1}(F^T \gamma^2_{i,j}) + \sum_{i\in[t]} \langle \vec{\boldsymbol{b}}^2_{i,n'}, \vec{\boldsymbol{y}} \rangle \mathsf{NTT}^{-1}(\vec{\gamma}^T_{n',i} \cdot P^i_{n'}) \\
& + \sum_{i\in[t]} \sum_{j\in[n'-1]} \left[\langle \vec{\boldsymbol{b}}^4_{j,i}, \vec{\boldsymbol{y}} \rangle \cdot (\mathsf{NTT}^{-1}(\boldsymbol{v}_{i,j} - \vec{\gamma}_{j+1,i})) + \langle \vec{\boldsymbol{b}}^2_{i,j}, \vec{\boldsymbol{y}} \rangle \cdot \boldsymbol{x}_{j,i} \right] \\
& + \sum_{i_1,i_2\in[t],i_1} \mathsf{NTT}^{-1}(F^T \gamma^3_{i_1,i_2}) \cdot \sum_{j\in[n']} \left[\langle \vec{\boldsymbol{b}}^2_{i_1,j}, \vec{\boldsymbol{y}} \rangle \boldsymbol{v}_{i_2,j} + \langle \vec{\boldsymbol{b}}^2_{i_2,j}, \vec{\boldsymbol{y}} \rangle \boldsymbol{v}_{i_1,j} \right] \\
& + \sum_{i_1,i_2\in[t],i_1} \mathsf{NTT}^{-1}(F^T \gamma^3_{i_1,i_2}) \cdot \langle \vec{\boldsymbol{b}}^3_{i_1,i_2}, \vec{\boldsymbol{y}} \rangle \cdot \mathsf{NTT}^{-1}(\mathfrak{g}), \\
\boldsymbol{\Omega}_1 = {} & \sum_{i\in[t]} \sum_{j\in[n'-1]} \langle \vec{\boldsymbol{b}}^2_{i,j}, \vec{\boldsymbol{y}} \rangle \cdot \langle \vec{\boldsymbol{b}}^4_{j,i}, \vec{\boldsymbol{y}} \rangle \\
& + \sum_{i_1,i_2\in[t],i_1\neq i_2} \mathsf{NTT}^{-1}(F^T \gamma^3_{i_1,i_2}) \cdot \left[\sum_{j\in[n']} \langle \vec{\boldsymbol{b}}^2_{i_1,j}, \vec{\boldsymbol{y}} \rangle \cdot \langle \vec{\boldsymbol{b}}^2_{i_2,j}, \vec{\boldsymbol{y}} \rangle \right].
\end{aligned}
$$

Thus we can compute the following two elements, which will be used in the proving stage:

$$
\begin{aligned}
\boldsymbol{\Phi} = {} & \boldsymbol{\alpha}_0 \boldsymbol{\Phi}_1 + \sum_{i\in[t]} \sum_{j\in[n']} \boldsymbol{\alpha}^1_{i,j} \cdot (2\boldsymbol{v}_{i,j} - 1) \cdot \langle \vec{\boldsymbol{b}}^2_{i,j}, \vec{\boldsymbol{y}} \rangle \\
& + \sum_{i_1,i_2\in[t],i_1\neq i_2} \boldsymbol{\alpha}^2_{i_1,i_2} \cdot (2\boldsymbol{u}_{i_1,i_2} - 1) \cdot \langle \vec{\boldsymbol{b}}^3_{i_1,i_2}, \vec{\boldsymbol{y}} \rangle, \\
\boldsymbol{\Omega}' = {} & \boldsymbol{\alpha}_0 \boldsymbol{\Omega}_1 + \sum_{i\in[t]} \sum_{j\in[n']} \boldsymbol{\alpha}^1_{i,j} \cdot \langle \vec{\boldsymbol{b}}^2_{i,j}, \vec{\boldsymbol{y}} \rangle^2 + \sum_{i_1,i_2\in[t],i_1\neq i_2} \boldsymbol{\alpha}^2_{i_1,i_2} \cdot \langle \vec{\boldsymbol{b}}^3_{i_1,i_2}, \vec{\boldsymbol{y}} \rangle^2. \qquad (11)
\end{aligned}
$$

Algorithm 8 Logarithmic t-out-of-n proof protocol.

$\mathsf{Prove}((\boldsymbol{P}, t, \vec{\boldsymbol{B}}, \vec{\boldsymbol{t}}), (\vec{v}, \vec{\boldsymbol{r}}))$:

Prover:

$(\boldsymbol{B}_3, \boldsymbol{B}_4, \vec{\boldsymbol{b}}_5, \vec{\boldsymbol{b}}_6) = H_1(\boldsymbol{P}, t, \vec{\boldsymbol{B}}, \vec{\boldsymbol{t}})$
Define $\vec{\boldsymbol{u}}, \vec{\boldsymbol{u}}'$ as in (9)
$\vec{\boldsymbol{t}}_3 = \boldsymbol{B}_3\vec{\boldsymbol{r}} + \vec{\boldsymbol{u}}$
Run $\mathsf{SP}(\cdot)$ algorithm defined in Algorithm9
$(\vec{\boldsymbol{t}}_4, \boldsymbol{h}^1) \leftarrow \mathsf{SP}(P_1, \vec{\boldsymbol{p}}, \vec{v}, \vec{\boldsymbol{t}}_3)$
$(\vec{\gamma}^2, \vec{\gamma}^3) = H_4(\vec{\boldsymbol{t}}_4)$
For all $i \in [t]$, $j \in [n']$, do:
$\boldsymbol{h}^2_{i,j} = \boldsymbol{v}_{i,j}\mathsf{NTT}^{-1}(F^T\gamma^2_{i,j}) - \boldsymbol{e}_1\mathsf{NTT}^{-1}(\gamma^2_{i,j})$
For all $i_1, i_2 \in [t]$, with $i_1 \neq i_2$, do:
$\boldsymbol{h}^3_{i_1,i_2} = \boldsymbol{u}'_{i_1,i_2}\mathsf{NTT}^{-1}(F^T\gamma^3_{i_1,i_2}) - (n'-1)\cdot\boldsymbol{e}_1\mathsf{NTT}^{-1}(\gamma^3_{i_1,i_2})$
$\boldsymbol{g} \leftarrow \mathcal{G}$, $\boldsymbol{t}_5 = \langle\vec{\boldsymbol{b}}_5, \vec{\boldsymbol{r}}\rangle + \boldsymbol{g}$
$\boldsymbol{h} = \boldsymbol{g} + \boldsymbol{h}^1 + \sum_{i\in[t], j\in[n']}\boldsymbol{h}^2_{i,j} + \sum_{i_1,i_2\in[t], i_1\neq i_2}\boldsymbol{h}^3_{i_1,i_2}$
$\vec{\boldsymbol{y}} \leftarrow \mathbb{D}^{dm}_\sigma$, $\vec{\boldsymbol{w}} = \boldsymbol{B}_0\vec{\boldsymbol{y}}$
$\vec{\boldsymbol{\alpha}} = H_5(\boldsymbol{h}, \vec{\boldsymbol{w}})$
Compute $\boldsymbol{\Phi}, \boldsymbol{\Omega}'$ as in (11)
$\boldsymbol{t}_6 = \langle\vec{\boldsymbol{b}}_6, \vec{\boldsymbol{r}}\rangle - \boldsymbol{\Psi}$
$\boldsymbol{\Omega} = \langle\vec{\boldsymbol{b}}_6, \vec{\boldsymbol{y}}\rangle + \boldsymbol{\Omega}'$
The prover sends messages $\omega = \{\vec{\boldsymbol{t}}_3, \vec{\boldsymbol{t}}_4, \boldsymbol{t}_5, \boldsymbol{t}_6, \boldsymbol{h}, \vec{\boldsymbol{w}}, \boldsymbol{\Omega}\}$ to the verifier

Verifier:

$\boldsymbol{c} \leftarrow D(S_c)$
The verifier sends $\boldsymbol{c}$ to the prover

Prover:

$\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \boldsymbol{c}\vec{\boldsymbol{r}}$
If $\mathsf{Rej}(\vec{\boldsymbol{z}}, \boldsymbol{c}\vec{\boldsymbol{r}}, \sigma) = 0$ abort
Send $\vec{\boldsymbol{z}}$ to the verifier

$\mathbf{Verify}((\boldsymbol{P}, t, ck, com), (\omega, \boldsymbol{c}, \vec{\boldsymbol{z}}))$:

Compute $\mathfrak{f}_1$ defined in (12)
Check if $\boldsymbol{h} \in \mathcal{G}$
Check if $\|\vec{\boldsymbol{z}}\|_2 \leq \sigma\sqrt{2dm}$
Check if $\vec{\boldsymbol{w}} = \boldsymbol{B}_0\vec{\boldsymbol{z}} - \boldsymbol{c}\vec{\boldsymbol{t}}_0$
Check if $\boldsymbol{\Omega} = \mathfrak{f}_1 + \langle\vec{\boldsymbol{b}}_6, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_6$
If failed **return** 0, else **return** 1

Algorithm 9 Sub-protocol (SP)

Input: $(P_1, \vec{\boldsymbol{p}}, \vec{v}, \vec{\boldsymbol{t}}_3)$; Output: $(\vec{\boldsymbol{t}}_4, \boldsymbol{h}^1)$

$\vec{\gamma}_1 = H_2(\vec{\boldsymbol{t}}_3)$
$P_2 = \begin{bmatrix}\vec{\gamma}_1^T P_{1,1}\\ \vdots \\ \vec{\gamma}_1^T P_{1,\varpi}\end{bmatrix} \in \mathcal{M}_q^{\varpi\times\varpi^{n'-1}}$
For all $i \in [t]$, do:
$\boldsymbol{x}_{1,i} = \mathsf{NTT}^{-1}(P_2\cdot(\vec{v}_{i,2}\otimes\cdots\otimes\vec{v}_{i,n'}))$
$\vec{\boldsymbol{t}}_{4,1} = \boldsymbol{B}_{4,1}\vec{\boldsymbol{r}} + \boldsymbol{x}_1$
$\boldsymbol{h}_1 = \sum_{i\in[t]}\boldsymbol{v}_{i,1}\cdot\boldsymbol{x}_{1,i} - \langle\vec{\boldsymbol{p}}, \mathsf{NTT}^{-1}(\vec{\gamma}_1)\rangle$
$(\vec{\gamma}_{2,1}, \ldots, \vec{\gamma}_{2,t}) = H_3(\vec{\boldsymbol{t}}_{4,1})$
For all $i \in [t]$, do:
$P_3^i = \begin{bmatrix}\vec{\gamma}_{2,i}^T P_{2,1}\\ \vdots \\ \vec{\gamma}_{2,i}^T P_{2,\varpi}\end{bmatrix} \in \mathcal{M}_q^{\varpi\times\varpi^{n'-2}}$,
$\boldsymbol{x}_{2,i} = \mathsf{NTT}^{-1}(P_3^i\cdot(\vec{v}_{i,3}\otimes\cdots\otimes\vec{v}_{i,n'}))$,
$\boldsymbol{h}_{2,i} = \boldsymbol{v}_{i,2}\cdot\boldsymbol{x}_{2,i} - \boldsymbol{x}_{1,i}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{2,i})$
$\vec{\boldsymbol{t}}_{4,2} = \boldsymbol{B}_{4,2}\vec{\boldsymbol{r}} + \boldsymbol{x}_2$
For $j = 3, \ldots n'-1$, do:
$(\vec{\gamma}_{j,1}\ldots, \vec{\gamma}_{j,t}) = H_3(\vec{\boldsymbol{t}}_{4,j-1})$
For all $i \in [t]$, do:
$P^i_{j+1} = \begin{bmatrix}\vec{\gamma}_{j,i}^T P^i_{j,1}\\ \vdots \\ \vec{\gamma}_{j,i}^T P^i_{j,\varpi}\end{bmatrix} \in \mathcal{M}_q^{\varpi\times\varpi^{n'-j}}$,
$\boldsymbol{x}_{j,i} = \mathsf{NTT}(P^i_{j+1}\cdot(\vec{v}_{i,j+1}\otimes\cdots\otimes\vec{v}_{i,n'}))$,
$\boldsymbol{h}_{j,i} = \boldsymbol{v}_{i,j}\cdot\boldsymbol{x}_{j,i} - \boldsymbol{x}_{j-1,i}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{j,i})$
$\vec{\boldsymbol{t}}_{4,j} = \boldsymbol{B}_{4,j}\vec{\boldsymbol{r}} + \boldsymbol{x}_j$
$(\vec{\gamma}_{n',1}\ldots, \vec{\gamma}_{n',t}) = H_3(\vec{\boldsymbol{t}}_{4,n'-1})$
For all $i \in [t]$, do:
$\boldsymbol{h}_{n',i} = \boldsymbol{v}_{i,n'}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}^T_{n',i}\cdot P^i_{n'}) - \boldsymbol{x}_{n'-1,i}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{n',i})$
$\vec{\boldsymbol{t}}_4 = (\vec{\boldsymbol{t}}_{4,1}, \ldots, \vec{\boldsymbol{t}}_{4,n'-1})$
$\boldsymbol{h}^1 = \boldsymbol{h}_1 + \sum_{i\in[t]}\sum_{j=2}^{n'}\boldsymbol{h}_{j,i}$
return $(\vec{\boldsymbol{t}}_4, \boldsymbol{h}^1)$

Now, we define another element, which will be used in the verification stage:

$$
\begin{aligned}
\mathfrak{f}_0 = {} & \boldsymbol{c}\cdot\langle\boldsymbol{B}_1\vec{\boldsymbol{z}} - \boldsymbol{c}\vec{\boldsymbol{t}}_1, \mathsf{NTT}^{-1}(\vec{\gamma}_1)\rangle - \boldsymbol{c}(\langle\vec{\boldsymbol{b}}_5, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_5) - \boldsymbol{c}^2\boldsymbol{h} \\
& - \sum_{i\in[t]}\boldsymbol{c}(\langle\vec{\boldsymbol{b}}^2_{i,n'}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}^2_{i,n'})\mathsf{NTT}^{-1}(\vec{\gamma}^T_{n',i}P^i_{n'}) \\
& + \sum_{i\in[t]}\sum_{j\in[n'-1]}\left[(\langle\vec{\boldsymbol{b}}^2_{i,j}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}^2_{i,j} + \boldsymbol{c}\mathsf{NTT}^{-1}(\vec{\gamma}_{j+1,i}))\cdot(\langle\vec{\boldsymbol{b}}^4_{j,i}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}^4_{j,i})\right] \\
& - \sum_{i\in[t]}\sum_{j\in[n']}\left[(\langle\vec{\boldsymbol{b}}^2_{i,j}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}^2_{i,j})\mathsf{NTT}^{-1}(F^T\gamma^2_{i,j})\cdot\boldsymbol{c} + \boldsymbol{e}_1\mathsf{NTT}^{-1}(\gamma^2_{i,j})\cdot\boldsymbol{c}^2\right] \\
& + \sum\ \mathsf{NTT}^{-1}(F^T\gamma^3_{i_1,i_2})\ \sum\left[(\langle\vec{\boldsymbol{b}}^2_{i_1,j}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}^2_{i_1,j})(\langle\vec{\boldsymbol{b}}^2_{i_2,j}, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}^2_{i_2,j})\right]
\end{aligned}
$$

Note the algorithm Rej and distributions $\mathbb{D}_\sigma$ are described in the Sect. 2.5. To make the Algorithm 8 clearer, we also define a sub-protocol Algorithm 9 that will be invoked by Algorithm 8 during the proving stage.

5.2.3 Security analysis

We now present the completeness, soundness and one-time zero-knowledge claims for our Algorithm 8.

Theorem 9 (Completeness) *Let* $\|\boldsymbol{c}\vec{\boldsymbol{r}}\| \leq T$ *and* $2^{10} < dm$, *then* Algorithm8 *has a completeness with error* $1 - \frac{1}{M}$, *where* $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$.

Proof It follows directly from Lemma 6 that the honest prover does not abort in the Step 20 with probability at least $\frac{1-2^{-100}}{M} \approx \frac{1}{M}$. Next, we show that when the prover does not abort, the verifier will accept except with a negligible probability.

First, by Lemma 6, the statistical distance between $\vec{z}$ and $\mathbb{D}_\sigma^{dm}$ is at most $2^{-100}/M$. Thus, by Lemma 5, we have $\|\vec{z}\| \leq \sigma\sqrt{2dm}$ with probability at lest $1 - \frac{2^{-100}}{M} - 2^{-0.11dm}$, which is overwhelming under property parameter selection.

Now, we prove $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}) = 0$. Since $\boldsymbol{g} \in \mathcal{G}$, we have

$$\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}) = \sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}^1) + \sum_{i\in[t], j\in[n']} [\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}^2_{i,j})] + \sum_{i_1,i_2\in[t], i_1\neq i_2} [\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}^3_{i_1,i_2})].$$

And by Lemma 1 and Lemma 2, we have

$$\begin{aligned}\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}_1) &= \sum_{i\in[t]} \langle \vec{v}_{i,1}, P_2 \cdot (\vec{v}_{i,2} \otimes \cdots \otimes \vec{v}_{i,n'}) \rangle - \langle \mathsf{NTT}(\vec{\boldsymbol{p}}), \gamma_1 \rangle \\ &= \sum_{i\in[t]} \langle \vec{v}_{i,1} \otimes \vec{v}_{i,2} \otimes \cdots \otimes \vec{v}_{i,n'}, P_1^T \gamma_1) \rangle - \langle \mathsf{NTT}(\vec{\boldsymbol{p}}), \gamma_1 \rangle \\ &= \langle P_1 \cdot \left(\sum_{i\in[t]} \vec{v}_{i,1} \otimes \cdots \otimes \vec{v}_{i,n'} \right) - \mathsf{NTT}(\vec{\boldsymbol{p}}), \gamma_1 \rangle = 0.\end{aligned}$$

Similarly, by Lemma 1 and Lemma 2, it can also be deduced that $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}^1)$, $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}^2_{i,j})$ and $\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}^3_{i_1,i_2})$ are 0. Thus, we have $\boldsymbol{h} \in \mathcal{G}$.

Since we have $\vec{z} = \vec{y} + \boldsymbol{c}\vec{r}$, $\vec{t}_0 = \boldsymbol{B}_0\vec{r}$ and $\vec{w} = \boldsymbol{B}_0\vec{y}$, the equation $\vec{w} = \boldsymbol{B}_0\vec{z} - \boldsymbol{c}\vec{t}_0$ always holds.

Finally, we show that, the $\boldsymbol{\Omega}$ generated in the **Prove** stage is equal to that generated in the **Verify** stage. By substituting $\vec{z} = \vec{y} + \boldsymbol{c}\vec{r}$ into $\mathfrak{f}_1$, we obtain $\mathfrak{f}_1 = -\boldsymbol{\Phi}\boldsymbol{c} + \boldsymbol{\Omega}'$. Thus, we have $\mathfrak{f}_1 + \langle \vec{\boldsymbol{b}}_6, \vec{z} \rangle - \boldsymbol{c}\boldsymbol{t}_6 = \langle \vec{\boldsymbol{b}}_6, \vec{y} \rangle + \boldsymbol{\Omega}' = \boldsymbol{\Omega}$. Therefore, our protocol satisfies completeness. □

Theorem 10 (Soundness) *Let* $q^{-d/\varpi} \leq 2^{-128}$, $\beta = \sigma\sqrt{2dm}$, *and define sets* $S_{fac} = \{\boldsymbol{c} \in \mathcal{R}_q^\times \mid \|\boldsymbol{c}\|_\infty \leq 2\}$ *and* $S'_r = \{\vec{\boldsymbol{r}} \in \mathcal{R}_q^m \mid \|\vec{\boldsymbol{r}}\| \leq 2\beta\}$, *assuming the* $\mathrm{MSIS}_{k_1,8d\beta}$ *problem is hard, then* Algorithm8 *has* 3*-special soundness.*

Proof Assuming that we have three valid transcripts, they are $(\omega, \boldsymbol{c}_0, \vec{\boldsymbol{z}}_0)$, $(\omega, \boldsymbol{c}_1, \vec{\boldsymbol{z}}_1)$ and $(\omega, \boldsymbol{c}_2, \vec{\boldsymbol{z}}_2)$, where $\omega = (\vec{\boldsymbol{t}}_3, \vec{\boldsymbol{t}}_4, \boldsymbol{t}_5, \boldsymbol{t}_6, \boldsymbol{h}, \vec{\boldsymbol{w}}, \boldsymbol{\Omega})$. Here, $\boldsymbol{c}_0, \boldsymbol{c}_1, \boldsymbol{c}_2$ are pairwise distinct. We will construct an extractor to extract a relaxed witness w.

Let $\bar{\boldsymbol{c}}_1 = \boldsymbol{c}_1 - \boldsymbol{c}_0$ and $\bar{\boldsymbol{c}}_2 = \boldsymbol{c}_2 - \boldsymbol{c}_0$, from Lemma 3, we have that $\bar{\boldsymbol{c}}_1$ and $\bar{\boldsymbol{c}}_2$ are invertible over $\mathcal{R}_q$ with probability at least $1 - \varpi \cdot \mathbf{p}^{d/\varpi}$, where $\mathbf{p}$ is the maximum probability over $\mathbb{Z}_q$ of the coefficients $\boldsymbol{c} \mod (X^{d/\varpi} - \zeta^{2i+1})$, which is close to $\frac{1}{q}$. Thus, with overwhelming probability $\bar{\boldsymbol{c}}_1$ and $\bar{\boldsymbol{c}}_2$ are invertible. And by the definition of the distribution $D(S_c)$, we have $\|\bar{\boldsymbol{c}}_1\|_\infty \leq 2$ and $\|\bar{\boldsymbol{c}}_2\|_\infty \leq 2$. Thus, $\bar{\boldsymbol{c}}_1, \bar{\boldsymbol{c}}_2 \in S_{fac}$ clearly holds.

For $i = 1, 2$, define $\vec{\boldsymbol{z}}_i^* = \vec{\boldsymbol{z}}_i - \vec{\boldsymbol{z}}_0$, $\vec{\boldsymbol{r}}_i = \bar{\boldsymbol{c}}_i^{-1} \cdot \vec{\boldsymbol{z}}_i^*$, then we have $\|\bar{\boldsymbol{c}}_i \vec{\boldsymbol{r}}_i\| \leq 2\beta$. Let $\vec{\boldsymbol{p}}_i = \vec{\boldsymbol{t}}_1 - \boldsymbol{B}_1 \vec{\boldsymbol{r}}_i, \vec{\boldsymbol{v}}_i = \vec{\boldsymbol{t}}_2 - \boldsymbol{B}_2 \vec{\boldsymbol{r}}_i, \vec{\boldsymbol{u}}_i = \vec{\boldsymbol{t}}_3 - \boldsymbol{B}_3 \vec{\boldsymbol{r}}_i, \vec{\boldsymbol{x}}_i = \vec{\boldsymbol{t}}_4 - \boldsymbol{B}_4 \vec{\boldsymbol{r}}_i, \boldsymbol{g}_i = \boldsymbol{t}_5 - \langle \vec{\boldsymbol{b}}_5, \vec{\boldsymbol{r}}_i \rangle, \boldsymbol{\Psi}_i = \boldsymbol{t}_6 - \langle \vec{\boldsymbol{b}}_6, \vec{\boldsymbol{r}}_i \rangle$, then $(\boldsymbol{c}_i, \vec{\boldsymbol{r}}_i, (\vec{\boldsymbol{p}}_i, \vec{\boldsymbol{v}}_i, \vec{\boldsymbol{u}}_i, \vec{\boldsymbol{x}}_i, \vec{\boldsymbol{g}}_i, \vec{\boldsymbol{\Psi}}_i))$ are two weak opening of the commitment $(\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{t}}_3, \vec{\boldsymbol{t}}_4, \boldsymbol{t}_5, \boldsymbol{t}_6)$. Thus, by Theorem 5, assuming the $\mathsf{MSIS}_{k_1, 8d\beta}$ problem is hard, we have

$$(\vec{\boldsymbol{p}}_1, \vec{\boldsymbol{v}}_1, \vec{\boldsymbol{u}}_1, \vec{\boldsymbol{x}}_1, \vec{\boldsymbol{g}}_1, \vec{\boldsymbol{\Psi}}_1) = (\vec{\boldsymbol{p}}_2, \vec{\boldsymbol{v}}_2, \vec{\boldsymbol{u}}_2, \vec{\boldsymbol{x}}_2, \vec{\boldsymbol{g}}_2, \vec{\boldsymbol{\Psi}}_2).$$

For $i = 0, 1, 2$, define $\vec{\boldsymbol{y}}_i = \vec{\boldsymbol{z}}_i - \boldsymbol{c}_i \vec{\boldsymbol{r}}_1$, then $\boldsymbol{B}_0 \vec{\boldsymbol{y}}_i = \vec{\boldsymbol{w}}$. Below, we prove that they are all equal. If $\vec{\boldsymbol{y}}_i \neq \vec{\boldsymbol{y}}_j$, then we have $\boldsymbol{B}_0 \cdot \bar{\boldsymbol{c}}_1 (\vec{\boldsymbol{y}}_i - \vec{\boldsymbol{y}}_j) = 0$ and

$$\|\bar{\boldsymbol{c}}_1 (\vec{\boldsymbol{y}}_i - \vec{\boldsymbol{y}}_j)\| \leq \|\bar{\boldsymbol{c}}_1 (\vec{\boldsymbol{z}}_i - \vec{\boldsymbol{z}}_j)\| + \|(\boldsymbol{c}_i - \boldsymbol{c}_j)(\vec{\boldsymbol{z}}_1 - \vec{\boldsymbol{z}}_0)\| \leq 8\|\boldsymbol{c}_1 \vec{\boldsymbol{z}}_1\| \leq 8d\beta.$$

Thus, assuming the $\mathsf{MSIS}_{k_1, 8d\beta}$ problem is hard, we have $\vec{\boldsymbol{y}}_0 = \vec{\boldsymbol{y}}_1 = \vec{\boldsymbol{y}}_2$. Since $\vec{\boldsymbol{p}}_i, \vec{\boldsymbol{v}}_i, \vec{\boldsymbol{u}}_i, \vec{\boldsymbol{x}}_i, \vec{\boldsymbol{g}}_i, \vec{\boldsymbol{\Psi}}_i$ and $\vec{\boldsymbol{y}}_i$ are equal, in the following calculations, we will directly omit their subscripts.

By substituting $\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{y}} + \boldsymbol{c}_i \vec{\boldsymbol{r}}_1$ into $\mathfrak{f} = \mathfrak{f}_1 + \langle \vec{\boldsymbol{b}}_6, \vec{\boldsymbol{z}} \rangle - \boldsymbol{c}_i \boldsymbol{t}_6$, where $\mathfrak{f}_1$ is defined in (12), we can obtain the following form equations:

$$\begin{bmatrix} 1 & \boldsymbol{c}_0 & \boldsymbol{c}_0^2 \\ 1 & \boldsymbol{c}_1 & \boldsymbol{c}_1^2 \\ 1 & \boldsymbol{c}_2 & \boldsymbol{c}_2^2 \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{\mu}_0 \\ \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

where

$$\boldsymbol{\mu}_2 = \boldsymbol{\alpha}_0(\boldsymbol{g} + \boldsymbol{h}' - \boldsymbol{h}) + \sum_{i \in [t]} \sum_{j \in [n']} \boldsymbol{\alpha}^1_{i,j} (\boldsymbol{v}_{i,j} - 1)\boldsymbol{v}_{i,j} + \sum_{i_1, i_2 \in [t], i_1 \neq i_2} \boldsymbol{\alpha}^2_{i_1, i_2} (\boldsymbol{u}_{i_1, i_2} - 1)\boldsymbol{u}_{i_1, i_2}$$

and

$$\begin{aligned}
\boldsymbol{h}' = & \sum_{i \in [t]} \boldsymbol{v}_{i,1} \cdot \boldsymbol{x}_{1,i} - \langle \vec{\boldsymbol{p}}, \mathsf{NTT}^{-1}(\vec{\gamma}_1) \rangle \\
& + \sum_{i \in [t]} \sum_{j=2}^{n'-1} \boldsymbol{v}_{i,j} \cdot \boldsymbol{x}_{j,i} - \boldsymbol{x}_{j-1,i} \cdot \mathsf{NTT}^{-1}(\vec{\gamma}_{j,i}) \\
& + \sum_{i \in [t]} \boldsymbol{v}_{i,n'} \cdot \mathsf{NTT}^{-1}(\vec{\gamma}^T_{n',i} \cdot P^i_{n'}) - \boldsymbol{x}_{n'-1,i} \cdot \mathsf{NTT}^{-1}(\vec{\gamma}_{n',i}) \\
& + \sum_{i \in [t], j \in [n']} \left[\boldsymbol{v}_{i,j} \mathsf{NTT}^{-1}(F^T \gamma^2_{i,j}) - \boldsymbol{e}_1 \mathsf{NTT}^{-1}(\gamma^2_{i,j}) \right] \\
& + \sum_{i_1, i_2 \in [t], i_1 \neq i_2} \left(\sum_{j \in [n']} \boldsymbol{v}_{i_1,j} \cdot \boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2} \mathsf{NTT}^{-1}(\mathfrak{g}) \right) \mathsf{NTT}^{-1}(F^T \gamma^3_{i_1,i_2})
\end{aligned}$$

$$- \sum_{i_1,i_2\in[t],i_1\neq i_2} (n'-1)\cdot \boldsymbol{e}_1\mathsf{NTT}^{-1}(\gamma^3_{i_1,i_2}).$$

Since the difference of each two challenges in $\{\boldsymbol{c}_0, \boldsymbol{c}_1, \boldsymbol{c}_2\}$ is invertible over $\mathcal{R}_q$, we must have $\boldsymbol{\mu}_2 = 0$. Next, in the random oracle model, $\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}^2_{t-1,t}$ are independent of $\boldsymbol{g}+\boldsymbol{h}'-\boldsymbol{h}$, $\boldsymbol{v}_{i,j}$ and $\boldsymbol{u}_{i_1,i_2}$.

Assume that $\boldsymbol{g}+\boldsymbol{h}'-\boldsymbol{h} \neq 0$, then for uniform $\boldsymbol{\alpha}_0$, the probability that $\boldsymbol{\mu}_2 = 0$ is at most $q^{d/\varpi}$, which is negligible. Hence, we have $\boldsymbol{g}+\boldsymbol{h}'-\boldsymbol{h} = 0$. Similarly, we have $\boldsymbol{v}_{i,j}(\boldsymbol{v}_{i,j}-1) = 0$ and $\boldsymbol{u}_{i_1,i_2}(\boldsymbol{u}_{i_1,i_2}-1) = 0$. And by this, we have $\mathsf{NTT}(\boldsymbol{v}_{i,j}) \in \{0,1\}^{\varpi}$, $\mathsf{NTT}(\boldsymbol{u}_{i_1,i_2}) \in \{0,1\}^{\varpi}$.

Since we have $\boldsymbol{g}+\boldsymbol{h}' = \boldsymbol{h}$ and $\boldsymbol{h} \in \mathcal{G}$. Thus, by Lemma 2, we have

$$\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{h}') = \sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{g}).$$

Note that, for any $i \in [t]$, $j \in [n']$, we have

$$\sum_{i=0}^{\varpi-1} \mathsf{NTT}(\boldsymbol{v}_{i,j}\mathsf{NTT}^{-1}(F^T\gamma^2_{i,j}) - \boldsymbol{e}_1\mathsf{NTT}^{-1}(\gamma^2_{i,j})) = \langle F\vec{v}_{i,j} - \vec{e}_1, \gamma^2_{i,j}\rangle.$$

Because all the other terms in $\boldsymbol{h}'$ are independent of $\gamma^2_{i,j}$, and when we have $F\cdot\mathsf{NTT}(\boldsymbol{v}_{i,j}) - \vec{e}_1 \neq 0$, then $\langle F\cdot\mathsf{NTT}(\boldsymbol{v}_{i,j}) - \vec{e}_1, \gamma^2_{i,j}\rangle$ is a uniformly random polynomial in $\mathcal{M}_q$, in this case $\sum_{i=0}^{\varpi-1}\mathsf{NTT}(\boldsymbol{h}')$ is also a uniformly random polynomial. In this time, the probability that $\sum_{i=0}^{\varpi-1}\mathsf{NTT}(\boldsymbol{h}') = \sum_{i=0}^{\varpi-1}\mathsf{NTT}(\boldsymbol{g})$ is $q^{-d/\varpi}$, which is negligible. Thus, we have $F\cdot\mathsf{NTT}(\boldsymbol{v}_{i,j}) = \vec{e}_1$, i.e. $\|\mathsf{NTT}(\boldsymbol{v}_{i,j})\|_1 = 1$.

Now we focus the on last two summands of $\boldsymbol{h}'$. For any i_1, i_2, we have

$$\begin{aligned}
&\sum_{i=0}^{\varpi}\mathsf{NTT}\left[\left(\sum_{j\in[n']}\boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g})\right)\cdot\mathsf{NTT}^{-1}(F^T\gamma^3_{i_1,i_2}) - (n'-1)\cdot\boldsymbol{e}_1\mathsf{NTT}^{-1}(\gamma^3_{i_1,i_2})\right]\\
&= \langle\mathsf{NTT}\left(\sum_{j\in[n']}\boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g})\right), F^T\gamma^3_{i_1,i_2}\rangle - \langle\mathsf{NTT}((n'-1)\boldsymbol{e}_1), \gamma^3_{i_1,i_2}\rangle\\
&= \langle F\cdot\mathsf{NTT}\left(\sum_{j\in[n']}\boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g})\right), \gamma^3_{i_1,i_2}\rangle - \langle\mathsf{NTT}((n'-1)\boldsymbol{e}_1), \gamma^3_{i_1,i_2}\rangle\\
&= \langle F\cdot\mathsf{NTT}\left(\sum_{j\in[n']}\boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g})\right) - \mathsf{NTT}((n'-1)\boldsymbol{e}_1), \gamma^3_{i_1,i_2}\rangle,
\end{aligned}$$

where the second equation is obtained according to Lemma 1. Thus, by the randomness of $\gamma^3_{i_1,i_2}$, we have

$$F\cdot\mathsf{NTT}\left(\sum_{j\in[n']}\boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g})\right) = \mathsf{NTT}((n'-1)\boldsymbol{e}_1).$$

Otherwise, the probability that the sum of the NTT coefficients of this term equals zero is negligible. Thus, by the definition of F and $\vec{e}_1$ (Eq. (10)), we have

$$\sum_{i=0}^{\varpi}\mathsf{NTT}\left(\sum_{j\in[n']}\boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g})\right) = (n'-1),$$

which implies that

$$\|\mathsf{NTT}\left(\sum_{j\in[n']} \boldsymbol{v}_{i_1,j}\cdot\boldsymbol{v}_{i_2,j} + \boldsymbol{u}_{i_1,i_2}\mathsf{NTT}^{-1}(\mathfrak{g})\right)\|_1 = (n'-1).$$

Next, let us consider $\boldsymbol{v}_{i,n'}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{n',i}^T\cdot P_{n'}^i) - \boldsymbol{x}_{n'-1,i}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{n',i})$, by calculation we have

$$\begin{aligned}
&\sum_{i=0}^{\varpi}\mathsf{NTT}(\boldsymbol{v}_{i,n'}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{n',i}^T\cdot P_{n'}^i) - \boldsymbol{x}_{n'-1,i}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{n',i}))\\
&\quad= \langle\mathsf{NTT}(\boldsymbol{v}_{i,n'}), (P_{n'}^i)^T\cdot\vec{\gamma}_{n',i}\rangle - \langle\mathsf{NTT}(\boldsymbol{x}_{n'-1,i}), \vec{\gamma}_{n',i}\rangle\\
&\quad= \langle P_{n'}^i\mathsf{NTT}(\boldsymbol{v}_{i,n'}) - \mathsf{NTT}(\boldsymbol{x}_{n'-1,i}), \vec{\gamma}_{n',i}\rangle. \qquad (13)
\end{aligned}$$

Thus, by the randomness of $\vec{\gamma}_{n',i}$, we obtain

$$P_{n'}^i\mathsf{NTT}(\boldsymbol{v}_{i,n'}) = \mathsf{NTT}(\boldsymbol{x}_{n'-1,i}).$$

Now, it remains to prove the 4th condition (in the overview of Sect. 5.2.1). We conduct an inductive argument and assume that for some $j\in[n'-1]$ we have

$$P_{j+1}^i\cdot(\mathsf{NTT}(\boldsymbol{v}_{i,j+1})\otimes\cdots\otimes\mathsf{NTT}(\boldsymbol{v}_{i,n'})) = \mathsf{NTT}(\boldsymbol{x}_{j,i}),$$

for every $i\in[t]$. Note that the base case $j=n'-1$ of induction (Eq. 13) is true.

Let us consider the term $\boldsymbol{v}_{i,j}\cdot\boldsymbol{x}_{j,i} - \boldsymbol{x}_{j-1,i}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{j,i})$. Since $P_{j+1}^i = \begin{bmatrix}\vec{\gamma}_{j,i}^T P_{j,1}^i\\ \vdots\\ \vec{\gamma}_{j,i}^T P_{j,\varpi}^i\end{bmatrix}$ and $P_j^i = [P_{j,1}^i,\ldots,P_{j,\varpi}^i]$, by calculation we have:

$$\begin{aligned}
&\sum_{i=0}^{\varpi-1}\mathsf{NTT}(\boldsymbol{v}_{i,j}\cdot\boldsymbol{x}_{j,i} - \boldsymbol{x}_{j-1,i}\cdot\mathsf{NTT}^{-1}(\vec{\gamma}_{j,i}))\\
&\quad= \langle\mathsf{NTT}(\boldsymbol{v}_{i,j}), P_{j+1}^i\cdot(\mathsf{NTT}(\boldsymbol{v}_{i,j+1})\otimes\cdots\otimes\mathsf{NTT}(\boldsymbol{v}_{i,n'}))\rangle - \langle\mathsf{NTT}(\boldsymbol{x}_{j-1,i}), \vec{\gamma}_{j,i}\rangle\\
&\quad= \langle P_j^i\cdot(\mathsf{NTT}(\boldsymbol{v}_{i,j})\otimes\cdots\otimes\mathsf{NTT}(\boldsymbol{v}_{i,n'})) - \mathsf{NTT}(\boldsymbol{x}_{j-1,i}), \vec{\gamma}_{j,i}\rangle.
\end{aligned}$$

Note that the second equation holds due to the definition of P_{j+1}^i. Thus, by the randomness of $\vec{\gamma}_{j,i}$, we can obtain

$$P_j^i\cdot(\mathsf{NTT}(\boldsymbol{v}_{i,j})\otimes\cdots\otimes\mathsf{NTT}(\boldsymbol{v}_{i,n'})) = \mathsf{NTT}(\boldsymbol{x}_{j-1,i}).$$

Note that, the P_2^i's are introduced for the convenience of the presentation. They all equal to P_2 defined in Algorithm 9, line 2. Hence, we have

$$P_2\cdot(\mathsf{NTT}(\boldsymbol{v}_{i,2})\otimes\cdots\otimes\mathsf{NTT}(\boldsymbol{v}_{i,n'})) = \mathsf{NTT}(\boldsymbol{x}_{1,i}).$$

Finally, let us consider the first item $\sum_{i\in[t]} \boldsymbol{v}_{i,1}\cdot \boldsymbol{x}_{1,i} - \langle \vec{\boldsymbol{p}}, \mathsf{NTT}^{-1}(\vec{\gamma}_1)\rangle$. Since $P_2 = \begin{bmatrix} \vec{\gamma}_1^T P_{1,1} \\ \vdots \\ \vec{\gamma}_1^T P_{1,\varpi} \end{bmatrix}$ and $P_1 = [P_{1,1}, \ldots, P_{1,\varpi}]$, by calculation we obtain

$$\begin{aligned}
&\sum_{i=0}^{\varpi-1} \mathsf{NTT}\left(\sum_{i\in[t]} \boldsymbol{v}_{i,1}\cdot \boldsymbol{x}_{1,i} - \langle \vec{\boldsymbol{p}}, \mathsf{NTT}^{-1}(\vec{\gamma}_1)\rangle\right) \\
&= \sum_{i\in[t]} \langle \mathsf{NTT}(\boldsymbol{v}_{i,1}), P_2\cdot(\mathsf{NTT}(\boldsymbol{v}_{i,2})\otimes\cdots\otimes\mathsf{NTT}(\boldsymbol{v}_{i,n'}))\rangle - \langle \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1\rangle \\
&= \langle P_1\cdot\sum_{i\in[t]}(\mathsf{NTT}(\boldsymbol{v}_{i,1})\otimes\cdots\otimes\mathsf{NTT}(\boldsymbol{v}_{i,n'})) - \mathsf{NTT}(\vec{\boldsymbol{p}}), \vec{\gamma}_1\rangle.
\end{aligned}$$

Therefore, by the randomness of $\vec{\gamma}_1$, we can obtain

$$\boldsymbol{P}\cdot\sum_{i\in[t]}(\mathsf{NTT}(\vec{\boldsymbol{v}}_{i,1})\otimes\cdots\otimes\mathsf{NTT}(\vec{\boldsymbol{v}}_{i,n'})) = \vec{\boldsymbol{p}}.$$

Otherwise, the probability that the sum of the NTT coefficients of $\boldsymbol{h}_1$ equals zero is negligible.

In conclusion, $w = (\vec{\boldsymbol{v}}, \vec{\boldsymbol{r}}_1, \tilde{\boldsymbol{c}}_1)$ is a relaxed witness. □

Theorem 11 (One-time Zero-knowledge) *Let* $\|\boldsymbol{c}\vec{\boldsymbol{r}}\| \le T$, $M = e^{\frac{24\sigma T + T^2}{2\sigma^2}}$ *and assuming the* $\mathsf{MLWE}_{m-\tilde{k},\sigma'}$ *problem is hard, where* $\tilde{k} = k' + t(n' - 1 + \frac{t-1}{2})$ *then* Algorithm8 *satisfies one-time zero-knowledge.*

Proof Assuming the protocol is not aborted, we construct a simulator $\mathcal{S}$ that takes $x = (\boldsymbol{P}, t, ck, com)$ and challenge $\boldsymbol{c}$ as inputs and outputs a valid transcript.

The simulator $\mathcal{S}$ first sets $\vec{\boldsymbol{t}}_3 \leftarrow \mathcal{R}_q^{\frac{t(t-1)}{2}}$, $\vec{\boldsymbol{t}}_4 \leftarrow \mathcal{R}_q^{t(n'-1)}$, $\boldsymbol{t}_5, \boldsymbol{t}_6 \leftarrow \mathcal{R}_q$ and $\boldsymbol{h} \leftarrow \mathcal{G}$. Then $\mathcal{S}$ samples $\vec{\boldsymbol{z}} \leftarrow \mathbb{D}_\sigma^{dm}$. Next, $\mathcal{S}$ computes $\vec{\boldsymbol{w}} = \boldsymbol{B}_0\vec{\boldsymbol{z}} - \boldsymbol{c}\vec{\boldsymbol{t}}_0$. Finally $\mathcal{S}$ computes $\boldsymbol{\Omega} = \mathfrak{f}_1 + \langle \vec{\boldsymbol{b}}_6, \vec{\boldsymbol{z}}\rangle - \boldsymbol{c}\boldsymbol{t}_6$, where $\mathfrak{f}_1$ defined in (12).

The distribution of simulated $\vec{\boldsymbol{z}}$ is statistically close to the real distribution by Lemma 6. And the distribution of simulated $\boldsymbol{h}$ is the same as the real distribution. Conditioned on $(\vec{\boldsymbol{z}}, \boldsymbol{h}, \boldsymbol{c})$ and $(\vec{\boldsymbol{t}}_3, \vec{\boldsymbol{t}}_4, \boldsymbol{t}_5, \boldsymbol{t}_6)$, variables $\vec{\boldsymbol{w}}$ and $\boldsymbol{\Omega}$ are uniquely determined from the verification equations. Thus, the distribution of simulated $(\vec{\boldsymbol{w}}, \boldsymbol{\Omega})$ is also the same as the real distribution. Finally, the distribution of simulated $(\vec{\boldsymbol{t}}_3, \vec{\boldsymbol{t}}_4, \boldsymbol{t}_5, \boldsymbol{t}_6)$ is computationally indistinguishable from the real one by the assumption of $\mathsf{MLWE}_{m-\tilde{k},\sigma'}$ problem.

Therefore, our t-out-of-n proof protocol satisfies one-time zero-knowledge. □

6 Threshold ring signature

6.1 Linear threshold ring signature

In this section, we present our LTRS scheme, by combining Algorithms 5, 6 and 7.

6.1.1 Specification and overview

Our LTRS is built over the power-of-2 cyclotomic ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ which splits into exactly ϖ factors.

The Setup (Algorithm 10) sets the system parameters, random oracles and sample random matrices. More concretely, it chooses public system parameters $d, q, \varpi, k, l, \eta, \kappa, \iota, \sigma, \sigma'$, n_{max} and t_{max}, where n_{max} is the maximum ring size and t_{max} is the maximum #signers. For simplicity, we require that ϖ is a divisor of n_{max}, i.e., $n'_{max} = n_{max}/\varpi \in \mathbb{Z}$. The Setup also defines the following hash functions:

- $H_1 : \{0,1\}^* \mapsto \mathcal{R}_q^{\kappa \times m}$ such that the rightmost k columns form an identity matrix $\boldsymbol{I}$, where $m = \iota + \kappa \lceil \log q \rceil + \kappa$;
- $H_2 : \{0,1\}^* \mapsto \mathbb{S}_1$, where the outputs follows the distribution $D(S_c)$ defined in the Sect. 2.5;
- $H_3 : \{0,1\}^* \mapsto \mathcal{R}_q^{2 \times m}$;
- $H_4 : \{0,1\}^* \mapsto \mathcal{G}$, where $\mathcal{G} := \{\boldsymbol{g} \in \mathcal{R}_q \mid g_0 = \cdots = g_{\frac{d}{\varpi}-1} = 0\}$;
- $H_5 : \{0,1\}^* \mapsto \mathcal{M}_q^{(k+1)\varpi}$;
- $H_6 : \{0,1\}^* \mapsto \mathcal{R}_q^m$, where the outputs follows distribution $\mathbb{D}_{\sigma_3}^{dm}$;
- $H_7 : \{0,1\}^* \mapsto \mathcal{R}_q^{n'_{max}+1}$.

Algorithm 10 LTRS: Setup & KeyGen

Setup(λ):
1: Choose parameters:
$d, q, \varpi, k, l, \eta, \kappa, \iota, \sigma_1, \sigma_2, \sigma_3, n_{max}, t_{max}$
2: Sample $\boldsymbol{A} \leftarrow \mathcal{R}_q^{k \times l}$, $\bar{\boldsymbol{A}} := [\boldsymbol{A} | \boldsymbol{I}]$
3: Define $H_1, H_2, H_3, H_4, H_5, H_6, H_7$
4: $pp = \left\{ \begin{array}{c} d, q, \varpi, k, l, \eta, \kappa, \iota, \sigma_1, \sigma_2, \sigma_3, \\ n_{max}, t_{max}, \bar{\boldsymbol{A}}, \\ H_1, H_2, H_3, H_4, H_5, H_6, H_7 \end{array} \right\}$
5: **return** pp

KeyGen(pp):
1: $\vec{\boldsymbol{s}} \leftarrow \mathbb{S}_\eta^{l+k}$, $\vec{\boldsymbol{p}} = \bar{\boldsymbol{A}}\vec{\boldsymbol{s}}$
2: **return** $pk = \vec{\boldsymbol{p}}$, $sk = \vec{\boldsymbol{s}}$

The KeyGen (Algorithm 10) is used to generate the public-secret key pair. Similarly as in the previous works [22, 25, 38], the secret key of a user is a vector $\vec{\boldsymbol{s}} \leftarrow \mathbb{S}_\eta^{l+k}$ of short polynomials over $\mathcal{R}_q$ and the public key is computed by $\vec{\boldsymbol{p}} = \bar{\boldsymbol{A}}\vec{\boldsymbol{s}}$.

We use $\boldsymbol{R}$ denote a public list (ring) with $n = n'\varpi \leq n_{max}$ users[8]. For $i \in [n]$, let pk_i, sk_i be the public key and secret key of the ith user in $\boldsymbol{R}$, respectively. We let I be the set of indices identifying the signers in $\boldsymbol{R}$ with $|I| = t \leq t_{max}$; S be the secret keys to the signers in I; and let $\vec{v} \in \{0,1\}^n$ be the vector of indices identifying the signers in $\boldsymbol{R}$, i.e., when $i \in I$ we have $v_i = 1$, otherwise $v_i = 0$, where v_i is the ith coefficient of $\vec{v}$.

The Sign (Alg.11) is an interactive procedure, which takes a message msg, a list of public key $\boldsymbol{R}$ and a list of secret key S as input and generates a signature Σ. Here we only present ith signer's behaviour (note that all signers behave the same) and divide the algorithm into three stages.

Let $\sigma' = \sqrt{\frac{5}{2\pi}} \cdot \sigma_2^2 \cdot (\sqrt{(\kappa+\iota)d} + \sqrt{\kappa d \cdot \lceil \log q \rceil} + \lambda)$ and $\vec{\boldsymbol{v}} = \mathsf{Encode}(\vec{v}) \in \mathcal{R}_q^{n'}$. We divide the output commitment key ck of H_1 into $(\boldsymbol{B}_0, \boldsymbol{B}_1, \boldsymbol{B}_2)$ by rows, where $\boldsymbol{B}_0 \in \mathcal{R}_q^{(\kappa-k-n'_{max}) \times m}$, $\boldsymbol{B}_1 \in \mathcal{R}_q^{k \times m}$ and $\boldsymbol{B}_2 \in \mathcal{R}_q^{n' \times m}$.[9] We define

$$F_1 = \mathsf{NTT}(-\boldsymbol{c}\boldsymbol{R}) = (\mathsf{NTT}(-\boldsymbol{c} \cdot pk_1), \ldots, \mathsf{NTT}(-\boldsymbol{c} \cdot pk_n)) \in \mathcal{M}_q^{k\varpi \times n}.$$

[8] If ϖ does not divide the ring size n, then we can simply add the redundant vectors as public keys.

[9] Here we directly replace n'_{max} with n', because the rest of row vector is not needed, we can just throw them away.

F_2 is the matrix defined in Eq. (2). Denote $\vec{\boldsymbol{y}}_1 = \mathsf{NTT}^{-1}(\vec{\gamma}_1) \in \mathcal{R}_q^k$, and $\boldsymbol{\gamma}_2 = \mathsf{NTT}^{-1}(\vec{\gamma}_2)$. Let $\boldsymbol{e}_t = \mathsf{NTT}^{-1}(\vec{e}_t)$ with $\vec{e}_t = (t, 0, \ldots, 0)$. Finally, let:

$$\begin{aligned}
\boldsymbol{\Omega} &= \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{y}}' \rangle + \sum_{i=1}^{n'} \boldsymbol{\alpha}_i \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}' \rangle^2, \\
\boldsymbol{\Psi} &= \boldsymbol{\alpha}_0 \langle (\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{y}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{y}}' \rangle + \sum_{i=1}^{n'} \boldsymbol{\alpha}_i \langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{y}}' \rangle (2\boldsymbol{v}_i - 1),
\end{aligned} \tag{14}$$

be two elements in $\mathcal{R}_q$, where $\vec{\boldsymbol{b}}_{2i}$ is the ith row vector of $\boldsymbol{B}_2$.

Algorithm 11 LTRS: Sign

Sign$_i(msg, \boldsymbol{R}, sk_i)$:

Phase 1:
1: $\vec{\boldsymbol{y}}_i \leftarrow \mathbb{D}_{\sigma_1}^{d(l+k)}$, $\vec{\boldsymbol{w}}_i = \bar{\boldsymbol{A}} \vec{\boldsymbol{y}}_i$
2: $ck \leftarrow H_1(msg, \boldsymbol{R})$, $\vec{\boldsymbol{r}}_i \leftarrow \mathbb{D}_{\sigma'}^{dm}$
3: $\vec{\boldsymbol{t}}_0^i = \boldsymbol{B}_0 \vec{\boldsymbol{r}}_i$, $\vec{\boldsymbol{t}}_1^i = \boldsymbol{B}_1 \vec{\boldsymbol{r}}_i + \vec{\boldsymbol{w}}_i$, $\vec{\boldsymbol{t}}_2^i = \boldsymbol{B}_2 \vec{\boldsymbol{r}}_i + \vec{\boldsymbol{e}}_i$
4: Send $(\vec{\boldsymbol{t}}_0^i, \vec{\boldsymbol{t}}_1^i, \vec{\boldsymbol{t}}_2^i)$ to other signers
Phase 2: After receiving all signers' commitments, do:
5: $\vec{\boldsymbol{t}}_0 = \sum_{j \in I} \vec{\boldsymbol{t}}_0^j, \vec{\boldsymbol{t}}_1 = \sum_{j \in I} \vec{\boldsymbol{t}}_1^j$, $\vec{\boldsymbol{t}}_2 = \sum_{j \in I} \vec{\boldsymbol{t}}_2^j, \boldsymbol{c}_1 = H_2(ck, \vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2)$
6: $\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{y}}_i + \boldsymbol{c}_1 \cdot sk_i$
7: If $\mathsf{Rej}(\vec{\boldsymbol{z}}_i, \boldsymbol{c}_1 \cdot sk_i, \sigma_1) = 0$, set $\vec{\boldsymbol{z}}_i = \perp$
8: Send $(\vec{\boldsymbol{z}}_i, \vec{\boldsymbol{r}}_i)$ to other signers
Phase 3: After receiving all signer's message, do:
9: If exists $j \in I$ and $\vec{\boldsymbol{z}}_j \notin S_z$, abort
10: $\vec{\boldsymbol{z}} = \sum_{j \in I} \vec{\boldsymbol{z}}_j, \vec{\boldsymbol{r}} = \sum_{j \in I} \vec{\boldsymbol{r}}_j$
11: $\boldsymbol{P} = -\boldsymbol{c}_1 \cdot \boldsymbol{R}, \vec{\boldsymbol{t}}_1' = \vec{\boldsymbol{t}}_1 - \bar{\boldsymbol{A}} \vec{\boldsymbol{z}}$
12: $x = (\boldsymbol{P}, t, ck, (\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1', \vec{\boldsymbol{t}}_2))$
13: $(\vec{\boldsymbol{b}}_3, \vec{\boldsymbol{b}}_4) = H_3(x)$
14: $\boldsymbol{g} = H_4(x, \vec{\boldsymbol{r}}), \boldsymbol{t}_3 = \langle \vec{\boldsymbol{b}}_3, \vec{\boldsymbol{r}} \rangle + \boldsymbol{g}$
15: $(\vec{\gamma}_1, \vec{\gamma}_2) = H_5(\vec{\boldsymbol{b}}_4, \boldsymbol{t}_3)$
16: $\vec{\boldsymbol{x}} = F_1^T \vec{\gamma}_1 + F_2^T \vec{\gamma}_2, \vec{\boldsymbol{x}} = \mathsf{NTT}^{-1}(\vec{x})$
17: $\boldsymbol{h} = \langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{x}} \rangle - \langle \vec{\boldsymbol{p}}, \vec{\boldsymbol{y}}_1 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{g}$
18: $\vec{\boldsymbol{y}}' = H_6(x, \vec{\boldsymbol{r}}), \vec{\boldsymbol{w}}' = \boldsymbol{B}_0 \vec{\boldsymbol{y}}'$,
19: $(\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{n'}) = H_7(\vec{\boldsymbol{x}}, \boldsymbol{h}, \vec{\boldsymbol{w}}')$
20: Compute $\boldsymbol{\Omega}, \boldsymbol{\Psi}$ defined in (14)
21: $\boldsymbol{t}_4 = \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{r}} \rangle - \boldsymbol{\Psi}$
22: $\boldsymbol{c}_2 = H_2(\boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \vec{\boldsymbol{w}}', \boldsymbol{\Omega})$
23: $\vec{\boldsymbol{z}}' = \vec{\boldsymbol{y}}' + \boldsymbol{c}_2 \cdot \vec{\boldsymbol{r}}$
24: If $\mathsf{Rej}(\vec{\boldsymbol{z}}', \boldsymbol{c}_2 \cdot \vec{\boldsymbol{r}}, \sigma_3) = 0$ abort
25: $\Sigma = (\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2, \boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \boldsymbol{c}_2, \vec{\boldsymbol{z}}, \vec{\boldsymbol{z}}')$
26: **return** signature Σ

The algorithm is presented in Alg.12, taking a message msg, a ring $\boldsymbol{R}$, a signature Σ and a verification threshold t and determining whether Σ is valid. Define

$$\begin{aligned}
\mathfrak{f}_1 &= \sum_{i=1}^{n'} \boldsymbol{\alpha}_i (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}}' \rangle - \boldsymbol{c}_2 \boldsymbol{t}_{2i}) \cdot (\langle \vec{\boldsymbol{b}}_{2i}, \vec{\boldsymbol{z}}' \rangle - \boldsymbol{c}_2 \boldsymbol{t}_{2i} + \boldsymbol{c}_2), \\
\mathfrak{f}_0 &= \langle (\vec{\boldsymbol{x}}^T \boldsymbol{B}_2 - \vec{\boldsymbol{y}}_1^T \boldsymbol{B}_1 + \vec{\boldsymbol{b}}_3), \vec{\boldsymbol{z}}' \rangle - \boldsymbol{c}_2 (\langle \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{x}} \rangle - \langle \vec{\boldsymbol{t}}_1', \vec{\boldsymbol{y}}_1 \rangle - \boldsymbol{e}_t \cdot \boldsymbol{\gamma}_2 + \boldsymbol{t}_3 - \boldsymbol{h}),
\end{aligned}$$

where $\vec{\boldsymbol{b}}_{2i}$ is the ith row vector of $\boldsymbol{B}_2$, $\boldsymbol{t}_{2i}$ is the ith element of $\vec{\boldsymbol{t}}_2$ and $\boldsymbol{e}_t = \mathsf{NTT}^{-1}(t, 0, \ldots, 0)$ that is determined by threshold t. Then, define

$$\boldsymbol{\Omega}' = \mathfrak{f}_1 - \boldsymbol{\alpha}_0 \boldsymbol{c}_2 \mathfrak{f}_0 + \langle \vec{\boldsymbol{b}}_4, \vec{\boldsymbol{z}}' \rangle - \boldsymbol{c}_2 \boldsymbol{t}_4. \tag{15}$$

Algorithm 12 LTRS: Verify

Verify$(msg, \boldsymbol{R}, \Sigma = (\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2, \boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \boldsymbol{c}_2, \vec{\boldsymbol{z}}, \vec{\boldsymbol{z}}'), t)$:

1: $ck \leftarrow H_1(msg, \boldsymbol{R})$
2: $\boldsymbol{c}_1 = H_2(ck, \vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2)$
3: $\boldsymbol{P} = -\boldsymbol{c}_1 \cdot \boldsymbol{R}, \vec{\boldsymbol{t}}'_1 = \vec{\boldsymbol{t}}_1 - \bar{\boldsymbol{A}}\vec{\boldsymbol{z}}$
4: $x = (\boldsymbol{P}, t, ck, (\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}'_1, \vec{\boldsymbol{t}}_2))$
5: $(\vec{\boldsymbol{b}}_3, \vec{\boldsymbol{b}}_4) = H_3(x), (\vec{\gamma}_1, \vec{\gamma}_2) = H_5(\vec{\boldsymbol{b}}_4, \boldsymbol{t}_3)$
6: $\vec{x} = F_1^T \vec{\gamma}_1 + F_2^T \vec{\gamma}_2, \vec{\boldsymbol{x}} = \mathrm{NTT}^{-1}(\vec{x})$
7: $\vec{\boldsymbol{w}}' = \boldsymbol{B}_0 \vec{\boldsymbol{z}}' - \boldsymbol{c}_2 \vec{\boldsymbol{t}}_0$
8: $(\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_{\eta'}) = H_7(\vec{\boldsymbol{x}}, \boldsymbol{h}, \vec{\boldsymbol{w}}')$
9: Compute $\boldsymbol{\Omega}'$ defined in (15)
10: Check if $\boldsymbol{h} \in \mathcal{G}$
11: Check if $\|\vec{\boldsymbol{z}}\|_2 \le \sigma\sqrt{2dt(l+k)}$,
12: Check if $\|\vec{\boldsymbol{z}}'\|_2 \le \sigma'\sqrt{2d(\kappa+\iota+\upsilon)}$
13: Check if $\boldsymbol{c}_2 = H_2(\boldsymbol{t}_3, \boldsymbol{t}_4, \boldsymbol{h}, \vec{\boldsymbol{w}}', \boldsymbol{\Omega}')$
14: If failed **return** 0, else **return** 1

6.1.2 Security analysis

We present the security claim for our LTRS scheme. Since the proof can be easily obtained by combining the security proof of GC-TRS and the lattice-based building blocks instantiation, we omit them.

Theorem 12 *Let* $T_1 = d\eta\sqrt{d(l+k)}$, $M_1 = e^{\frac{24\sigma_1 T_1 + T_1^2}{2\sigma_1^2}}$, $m = \iota + \kappa\lceil\log q\rceil + \kappa$, $\sigma' = \sqrt{\frac{5}{2\pi}} \cdot \sigma_2^2 \cdot (\sqrt{(\kappa+\iota)d} + \sqrt{\kappa d \cdot \lceil\log q\rceil} + \lambda)$, $T_2 = \sigma' d\sqrt{2t_{max}dm}$ *and* $M_2 = e^{\frac{24\sigma_3 T_2 + T_2^2}{2\sigma_3^2}}$, *and parameters satisfy the following conditions:* $2^{10} < d(l+k) < 2^{45}$, $2^{10} < dm < 2^{45}$, $11/\pi \le \sigma_1$, $11/\pi \le \sigma'$, *then our* LTRS *scheme has correctness. On average, the construction needs to be repeated* $M_1^t M_2$ *times to generate a valid signature.*

Let $\beta_1 = 2(\sigma_1\sqrt{2t_{max}} + (t-1)d\eta)\sqrt{d(l+k)} + 2\sqrt{d}$, $\beta_2 = \sigma_3\sqrt{2dm}$, *assuming the* $\mathsf{MLWE}_{l,\eta}$, $\mathsf{MLWE}_{\iota,\sigma_2}$, $\mathsf{MSIS}_{k,\beta_1}$ *and* $\mathsf{MSIS}_{(\kappa-k-n'_{max}),8d\beta_2}$ *problems are hard and* $q^{-d/\varpi} \le 2^{-128}$, *then our* LTRS *scheme has* t_{max}*-unforgeability w.r.t. insider corruption and* t_{max}*-anonymity w.r.t. adversarial keys in the* ROM.

Note the $\mathsf{MLWE}_{m-\kappa-2,\sigma'}$ problem is at least as hard as the $\mathsf{MLWE}_{\iota,\sigma_2}$ problem. Thus, in the above theorem, we only require the $\mathsf{MLWE}_{\iota,\sigma_2}$ problem to be hard. The same phenomenon also occurs in CTRS. It is also worth mentioning that our LTRS is suited for parallel executions. Hence, to obtain a valid signature, one can alternatively run sufficiently many parallel executions of the scheme at once.

6.1.3 Concrete parameters

In Table 4, we give concrete parameters for our LTRS scheme. With this parameter setting, the root Hermite factor (RHF), a standard metric to estimate MLWE/MSIS hardness in practice, is between 1.0043 and 1.00469 for all MLWE/MSIS assumptions. It is a common practice to choose a RHF around 1.0045 for 128-bit post-quantum security. By Theorem 12, when $t_{max} = 10$, our construction needs to be repeated on average 18.6 times to generate a valid signature; and when $t_{max} = 50$, it needs to be repeated 21.6 times.

We now calculate the signature size. Considering the "full-sized" elements of $\mathcal{R}_q$, we have $\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2, \boldsymbol{t}_3, \boldsymbol{t}_4$ and $\boldsymbol{h}$[10]. Therefore, we have in total $\kappa + 3$ full-sized elements, which altogether costs $(\kappa+3)d\log q$ bits. We further consider the vectors of short polynomials $\vec{\boldsymbol{z}}$

[10] In fact, $\boldsymbol{h}$ is missing d/ϖ coefficients, but that is a negligible consideration.

Table 4 Examples of parameters for LTRS (128-bit security)

n_{max}	2^{10}	2^{15}	2^{20}	2^{10}	2^{15}	2^{20}
t_{max}	10	10	10	50	50	50
d	128	128	128	128	128	128
ϖ	32	32	32	32	32	32
$q \approx$	2^{54}	2^{60}	2^{68}	2^{55}	2^{61}	2^{69}
η	1	1	1	1	1	1
l	17	19	22	18	20	22
k	5	5	4	5	5	4
σ_1	447063	466942	486009	457111	476571	486009
ι	17	19	22	18	20	22
κ	54	1049	32794	55	1049	32794
σ_2	2	2	2	2	2	2
σ'	2976	11904	68055	3022	11988	68496
$\sigma_3 \approx$	2^{33}	2^{37}	2^{42}	2^{34}	2^{38}	2^{43}
M_1	1.2	1.2	1.2	1.04	1.04	1.04
M_2	3	3	3	3	3	3
Repetition	18.6	18.6	18.6	21.6	21.6	21.6
Signature size (MB)	1.75	41.3	1636.8	1.87	43.2	1701.9

and $\vec{z}'$. Since they come from a Gaussian distribution with standard deviation $\sigma_1\sqrt{t}$ and σ_3 respectively, with a high probability we can upper bound their coefficients by $6\sigma_1\sqrt{t}$ and $6\sigma_3$ [36, 38]. Thus, they require at most

$$d(l+k)\log(13\sigma_1\sqrt{t}) + dm\log(13\sigma_3) \text{ bits.}$$

Finally, the challenge $\boldsymbol{c}_2$ costs at most $2 \cdot d = 256$ bits. Therefore, the signature size is:

$$d[(\kappa+3)\log q + (l+k)\log(13\sigma_1\sqrt{t}) + m\log(13\sigma_3) + 2] \text{ bits.}$$

6.2 Compact threshold ring signature

In this section, we present our CTRS scheme, which is constructed by combining Alg.5, $mathsf Algorithm6$ and Algorithm8.

6.2.1 Specification and overview

Our CTRS(Algorithm13) is over the power-of-2 cyclotomic ring $\mathcal{R}_q$ which splits into exactly ϖ factors. The public system parameters are d, q, ϖ, k, l, η, κ, ι, σ, σ', n_{max} and t_{max}, where n_{max} is the maximum ring size and t_{max} is the maximum #signers. For simplicity, we require $n_{max} = \varpi^{n'_{max}}$. The Setup will define the following hash functions:

- $H_1 : \{0,1\}^* \mapsto \mathcal{R}_q^{\kappa \times m}$ such that the rightmost k columns form an identity matrix $\boldsymbol{I}$, where $m = \iota + \kappa\lceil\log q\rceil + \kappa$;
- $H_2 : \{0,1\}^* \mapsto \mathbb{S}_1$, where the outputs follows the distribution $D(S_c)$ defined in Sect. 2.5;

We use $\boldsymbol{R}$ denote a public list (ring) with $n = \varpi^{n'} \leq n_{max}$ users. For $i \in [n]$, let pk_i, sk_i be the public key and secret key of the ith user in $\boldsymbol{R}$, respectively. We let I be the set of indices identifying the signers in $\boldsymbol{R}$ with $|I| = t \leq t_{max}$; S be the secret keys to the signers in I; and let $\vec{v} \in \{0,1\}^n$ be the vector of indices identifying the signers in $\boldsymbol{R}$, i.e., when $i \in I$ we have $v_i = 1$, otherwise $v_i = 0$, where v_i is the ith coefficient of $\vec{v}$.

The Sign algorithm is an interactive procedure, which takes a message msg, a list of public key $\boldsymbol{R}$ and a list of secret key S as input and generates a signature Σ. Here we only present ith signer's behaviour and also divide the algorithm into three stages. Let $\sigma' = \sqrt{\frac{5}{2\pi}} \cdot \sigma_2^2 \cdot (\sqrt{(\kappa+\iota)d} + \sqrt{\kappa d \cdot \lceil \log q \rceil} + \lambda)$ and $\vec{v} = \sum_{i\in[t]}(\vec{v}_{i,1} \otimes \cdots \otimes \vec{v}_{i,n'})$, then $\mathsf{Encode}(\vec{e}_i)$ is defined as

$$(0, \ldots, \mathsf{NTT}^{-1}(\vec{v}_{i,1}), \ldots, \mathsf{NTT}^{-1}(\vec{v}_{i,n'}), 0, \ldots, 0) \in \mathcal{R}_q^{tn'}.$$

In this case, we have $\mathsf{Encode}(\vec{e}_i) = \sum_{i\in[t]} \mathsf{Encode}(\vec{v})$. The algorithms Prove and Verify used in Algorithm13 are the algorithm described in Algorithm8. Note that, in Algorithm8, the Prove algorithm is an interactive algorithm. Thus, we need to use Fiat-Shamir transformation to it first and then invoke it.

Algorithm 13 Compact TRS (CTRS)

Setup(λ):
1: Choose parameters
2: Sample $\boldsymbol{A} \leftarrow \mathcal{R}_q^{k\times l}$, $\bar{\boldsymbol{A}} := [\boldsymbol{A}|\boldsymbol{I}]$
3: Define H_1, H_2

KeyGen(pp):
1: $\vec{s} \leftarrow \mathbb{S}_\eta^{l+k}$, $\vec{p} = \bar{\boldsymbol{A}}\vec{s}$
2: **return** $pk = \vec{p}, sk = \vec{s}$

Sign$_i$($msg, \boldsymbol{R}, sk_i$):

Phase 1:
1: $\vec{y}_i \leftarrow \mathbb{D}_{\sigma_1}^{d(l+k)}$, $\vec{w}_i = \bar{\boldsymbol{A}}\vec{y}_i$
2: $ck \leftarrow H_1(msg, \boldsymbol{R})$, $\vec{r}_i \leftarrow \mathbb{D}_{\sigma'}^{dm}$
3: $\vec{t}_0^i = \boldsymbol{B}_0\vec{r}_i$, $\vec{t}_1^i = \boldsymbol{B}_1\vec{r}_i + \vec{w}_i$, $\vec{t}_2^i = \boldsymbol{B}_2\vec{r}_i + \mathsf{Encode}(\vec{e}_i)$
4: Send $(\vec{t}_0^i, \vec{t}_1^i, \vec{t}_2^i)$ to other signers

Phase 2: After receiving all signers' commitments, do:
5: $\vec{t}_0 = \sum_{j\in I}\vec{t}_0^j, \vec{t}_1 = \sum_{j\in I}\vec{t}_1^j$, $\vec{t}_2 = \sum_{j\in I}\vec{t}_2^j, \boldsymbol{c}_1 = H_2(ck, \vec{t}_0, \vec{t}_1, \vec{t}_2)$
6: $\vec{z}_i = \vec{y}_i + \boldsymbol{c}_1 \cdot sk_i$
7: If $\mathsf{Rej}(\vec{z}_i, \boldsymbol{c}_1 \cdot sk_i, \sigma_1) = 0$, set $\vec{z}_i = \perp$
8: Send $(\vec{z}_i, \vec{r}_i)$ to other signers

Phase 3: After receiving all signer's message, do:
9: If exists $j \in I$ and $\vec{z}_j \notin S_z$, abort
10: $\vec{z} = \sum_{j\in I}\vec{z}_j, \vec{r} = \sum_{j\in I}\vec{r}_j$
11: $\boldsymbol{P} = -\boldsymbol{c}_1 \cdot \boldsymbol{R}, \vec{t}_1' = \vec{t}_1 - \bar{\boldsymbol{A}}\vec{z}$
12: $x = (\boldsymbol{P}, t, ck, (\vec{t}_0, \vec{t}_1', \vec{t}_2))$
13: $w = (\mathsf{Encode}(\vec{v}), \vec{r})$
14: $(\omega, c', z') \leftarrow \mathsf{Prove}(x, w)$
15: $\Sigma = (\vec{t}_0, \vec{t}_1, \vec{t}_2, \vec{z}, \omega, c', z')$
16: **return** signature Σ

Verify($msg, \boldsymbol{R}, \Sigma, t$):
1: $ck \leftarrow H_1(msg, \boldsymbol{R})$
2: $\boldsymbol{c}_1 = H_2(ck, \vec{t}_0, \vec{t}_1, \vec{t}_2)$
3: $\boldsymbol{P} = -\boldsymbol{c}_1 \cdot \boldsymbol{R}, \vec{t}_1' = \vec{t}_1 - \bar{\boldsymbol{A}}\vec{z}$
4: $x = (\boldsymbol{P}, t, ck, (\vec{t}_0, \vec{t}_1', \vec{t}_2))$
5: $\mathsf{Verify}(x, (\omega, c', z'))$
6: Check if $\|\vec{z}\|_2 \leq \sigma\sqrt{2dt(l+k)}$
7: If failed **return** 0, else **return** 1

6.2.2 Security analysis

We present the security claim for our CTRS scheme. Since the proof can be easily obtained by combining the security proof of GC-TRS and the lattice-based building blocks instantiation, we omit them.

Table 5 Examples of Parameters for CTRS (128-bit security)

n_{max}	2^{10}	2^{15}	2^{20}	2^{10}	2^{15}	2^{20}
t_{max}	10	10	10	50	50	50
d	128	128	128	128	128	128
ϖ	32	32	32	32	32	32
$q \approx$	2^{53}	2^{54}	2^{54}	2^{57}	2^{57}	2^{58}
η	1	1	1	1	1	1
l	17	17	17	18	18	19
k	5	5	5	5	5	5
σ_1	447063	447063	447063	457111	457111	466942
ι	17	17	17	18	18	19
κ	42	52	62	123	174	224
σ_2	2	2	2	2	2	2
σ'	2671	2931	3151	4316	5036	5687
$\sigma_3 \approx$	2^{33}	2^{33}	2^{33}	2^{36}	2^{36}	2^{36}
M_1	1.2	1.2	1.2	1.04	1.04	1.04
M_2	3	3	3	3	3	3
Repetition	18.6	18.6	18.6	21.6	21.6	21.6
Signature size (MB)	1.37	1.73	2.07	5.53	7.47	9.56

Theorem 13 *Let* $T_1 = d\eta\sqrt{d(l+k)}$, $M_1 = e^{\frac{24\sigma_1 T_1 + T_1^2}{2\sigma_1^2}}$, $m = \iota + \kappa\lceil \log q \rceil + \kappa$, $\sigma' = \sqrt{\frac{5}{2\pi}} \cdot \sigma_2^2 \cdot (\sqrt{(\kappa+\iota)d} + \sqrt{\kappa d \cdot \lceil \log q \rceil} + \lambda)$, $T_2 = \sigma' d\sqrt{2t_{max}dm}$ *and* $M_2 = e^{\frac{24\sigma_3 T_2 + T_2^2}{2\sigma_3^2}}$, *and parameters satisfy the following conditions:* $2^{10} < d(l+k) < 2^{45}$, $2^{10} < dm < 2^{45}$, $11/\pi \leq \sigma_1$, $11/\pi \leq \sigma'$, *then our* CTRS *scheme has correctness. On average, the construction needs to be repeated* $M_1^t M_2$ *times to generate a valid signature.*

Let $\beta_1 = 2(\sigma_1\sqrt{2t_{max}} + (t-1)d\eta)\sqrt{d(l+k)} + 2\sqrt{d}$, $\beta_2 = \sigma_3\sqrt{2dm}$, *assuming the* $\mathsf{MLWE}_{l,\eta}$, $\mathsf{MLWE}_{\iota,\sigma_2}$, $\mathsf{MSIS}_{k,\beta_1}$ *and* $\mathsf{MSIS}_{(\kappa-k-n'_{max}),8d\beta_2}$ *problems are hard and* $q^{-d/\varpi} \leq 2^{-128}$, *then our* CTRS *scheme has* t_{max}*-unforgeability w.r.t. insider corruption and* t_{max}*-anonymity w.r.t. adversarial keys in the* ROM.

Note that our CTRS is suited for parallel executions. Thus, to obtain a valid signature, one can alternatively run sufficiently many parallel executions of the scheme at once.

6.2.3 Concrete parameters

In Table 5, we give concrete parameters for our CTRS scheme. In the parameters setting, the root Hermite factor is less than 1.0042 for all MLWE/MSIS problems. It is a common practice to choose a root Hermite factor around 1.0042 for 128-bit post-quantum security. By Theorem 12, when $t_{max} = 10$, our construction needs to be repeated on average 18.6 times to generate a valid signature; and when $t_{max} = 50$, it needs to be repeated 21.6 times.

We now calculate the signature size. Considering the "full-sized" elements of $\mathcal{R}_q$, we have $\vec{\boldsymbol{t}}_0, \vec{\boldsymbol{t}}_1, \vec{\boldsymbol{t}}_2, \vec{\boldsymbol{t}}_3, \vec{\boldsymbol{t}}_4, \boldsymbol{t}_5, \boldsymbol{t}_6$ and $\boldsymbol{h}$.[11] So, we have $\kappa + t(\frac{t-1}{2} + n' - 1) + 2$ full-sized elements in total, which altogether costs $(\kappa + t(\frac{t-1}{2} + n' - 1) + 2)d \log q$ bits. We further consider the vectors of short polynomials $\vec{\boldsymbol{z}}$ and $\vec{\boldsymbol{z}}'$. Since they come from a Gaussian distribution with standard deviation $\sigma_1\sqrt{t}$ and σ_3 respectively, with a high probability we can upper bound their coefficients by $6\sigma_1\sqrt{t}$ and $6\sigma_3$ [36, 38]. Thus, they require at most

$$d(l+k)\log(13\sigma_1\sqrt{t}) + dm\log(13\sigma_3) \text{ bits.}$$

Finally, the challenge $\boldsymbol{c}_2$ costs at most $2 \cdot d = 256$ bits. Therefore, the signature size is:

$$d[(\kappa + t(\frac{t-1}{2} + n' - 1) + 2)\log q + (l+k)\log(13\sigma_1\sqrt{t}) + m\log(13\sigma_3) + 2] \text{ bits.}$$

6.2.4 Future work

Although our CTRS scheme achieves logarithmic size in terms of asymptotic efficiency, the actual signature size is not particularly ideal. Therefore, an important future task for us is to design a TRS scheme with a better actual signature size, which will make it meaningful in the real world.

Another important future work is to research and analyze the security proof of our scheme in the QROM model with the smallest possible reduction loss, which will ensure the security in the face of quantum threats.

Acknowledgements The authors would like to thank the reviewers for providing helpful suggestions. This work was supported by the Naval University of Engineering of China under Grant No. 2023508020, the National Key Research and Development Program of China under Grant No. 2021YFA1000600, the National Natural Science Foundation of China under Grant No. 12441104 and Grant No. 62272294, Shanghai Science and Technology Innovation Action Plan under Grant No. 23511101100 and the EU Horizon Europe Research and Innovation Program under Grant No. 101073920 (TENSOR), Grant No. 101070052 (TANGO), Grant No. 101070627 (REWIRE) and Grant No. 101092912 (MLSysOps).

References

1. Abdalla M., An J.H., Bellare M., Namprempre C.: From identification to signatures via the Fiat–Shamir transform: minimizing assumptions for security and forward-security. In: Knudsen L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 418–433. Springer (2002). https://doi.org/10.1007/3-540-46035-7_28.
2. Abe M., Ohkubo M., Suzuki K.: 1-out-of-n signatures from a variety of keys. In: Zheng Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 415–432. Springer (2002). https://doi.org/10.1007/3-540-36178-2_26.
3. Aranha D.F., Hall-Andersen M., Nitulescu A., Pagnin E., Yakoubov S.: Count me in! Extendability for threshold ring signatures. In: Hanaoka G., Shikata J., Watanabe Y. (eds.) PKC 2022. LNCS, vol. 13178, pp. 379–406. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_13.
4. Attema T., Fehr S., Klooß M.: Fiat–Shamir transformation of multi-round interactive proofs (extended version). J. Cryptol. **36**(4), 36 (2023). https://doi.org/10.1007/s00145-023-09478-y.
5. Attema T., Lyubashevsky V., Seiler G.: Practical product proofs for lattice commitments. In: Micciancio D., Ristenpart T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 470–499. Springer (2020). https://doi.org/10.1007/978-3-030-56880-1_17.
6. Bagherzandi A., Cheon J.H., Jarecki S.: Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In: Ning P., Syverson P.F., Jha S. (eds.) CCS 2008. pp. 449–458. ACM (2008). https://doi.org/10.1145/1455770.1455827.

[11] Note that, the value $\vec{\boldsymbol{w}}$ and $\boldsymbol{\Omega}$ generated in the Prove algorithm can not be included in the signature, using the same method used in LTRS.

7. Banaszczyk W.: New bounds in some transference theorems in the geometry of numbers. Math. Ann. **296**(1), 625–635 (1993). https://doi.org/10.1007/BF01445125.
8. Baum C., Bootle J., Cerulli A., del Pino R., Groth J., Lyubashevsky V.: Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In: Shacham H., Boldyreva A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 669–699. Springer (2018). https://doi.org/10.1007/978-3-319-96881-0_23.
9. Baum C., Damgård I., Lyubashevsky V., Oechsner S., Peikert C.: More efficient commitments from structured lattice assumptions. In: Catalano D., Prisco R.D. (eds.) SCN 2018. LNCS, vol. 11035, pp. 368–385. Springer (2018). https://doi.org/10.1007/978-3-319-98113-0_20.
10. Baum C., Lyubashevsky V.: Simple amortized proofs of shortness for linear relations over polynomial rings. IACR Cryptol. ePrint Arch. p. 759 (2017). http://eprint.iacr.org/2017/759.
11. Benhamouda F., Krenn S., Lyubashevsky V., Pietrzak K.: Efficient zero-knowledge proofs for commitments from learning with errors over rings. In: Pernul G., Ryan P.Y.A., Weippl E.R. (eds.) ESORICS 2015. LNCS, vol. 9326, pp. 305–325. Springer (2015). https://doi.org/10.1007/978-3-319-24174-6_16.
12. Bert P., Fouque P., Roux-Langlois A., Sabt M.: Practical implementation of ring-SIS/LWE based signature and IBE. In: Lange T., Steinwandt R. (eds.) PQCrypto 2018. LNCS, vol. 10786, pp. 271–291. Springer (2018). https://doi.org/10.1007/978-3-319-79063-3_13.
13. Bettaieb S., Schrek J.: Improved lattice-based threshold ring signature scheme. In: Gaborit P. (ed.) PQCrypto 2013. LNCS, vol. 7932, pp. 34–51. Springer (2013). https://doi.org/10.1007/978-3-642-38616-9_3.
14. Blum M., Feldman P., Micali S.: Non-interactive zero-knowledge and its applications (extended abstract). In: Simon J. (ed.) STOC 1988, pp. 103–112. ACM (1988). https://doi.org/10.1145/62212.62222.
15. Bootle J., Lyubashevsky V., Seiler G.: Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In: Boldyreva A., Micciancio D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 176–202. Springer (2019). https://doi.org/10.1007/978-3-030-26948-7_7.
16. Bresson E., Stern J., Szydlo M.: Threshold Ring signatures and applications to ad-hoc groups. In: Yung M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 465–480. Springer (2002). https://doi.org/10.1007/3-540-45708-9_30.
17. Cayrel P., Lindner R., Rückert M., Silva R.: A lattice-based threshold ring signature scheme. In: Abdalla M., Barreto P.S.L.M. (eds.) LATINCRYPT 2010. LNCS, vol. 6212, pp. 255–272. Springer (2010). https://doi.org/10.1007/978-3-642-14712-8_16.
18. Dallot L., Vergnaud D.: Provably secure code-based threshold ring signatures. In: Parker M.G. (ed.) IMA 2009. LNCS, vol. 5921, pp. 222–235. Springer (2009). https://doi.org/10.1007/978-3-642-10868-6_13.
19. Damgård I., Orlandi C., Takahashi A., Tibouchi M.: Two-round n-out-of-n and multi-signatures and Trapdoor commitment from lattices. In: Garay J.A. (ed.) PKC 2021. LNCS, vol. 12710, pp. 99–130. Springer (2021). https://doi.org/10.1007/978-3-030-75245-3_5.
20. Devevey J., Fallahpour P., Passelègue A., Stehlé D.: A detailed analysis of Fiat–Shamir with aborts. IACR Cryptol. ePrint Arch. p. 245 (2023). https://eprint.iacr.org/2023/245.
21. Drijvers M., Edalatnejad K., Ford B., Kiltz E., Loss J., Neven G., Stepanovs I.: On the security of two-round multi-signatures. In: IEEE Symposium on Security and Privacy, 2019, pp. 1084–1101. IEEE (2019). https://doi.org/10.1109/SP.2019.00050.
22. Ducas L., Kiltz E., Lepoint T., Lyubashevsky V., Schwabe P., Seiler G., Stehlé D.: CRYSTALS-Dilithium: a lattice-based digital signature scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(1), 238–268 (2018). https://doi.org/10.13154/tches.v2018.i1.238-268.
23. Esgin M.F., Nguyen N.K., Seiler G.: Practical exact proofs from lattices: new techniques to exploit fully-splitting rings. In: Moriai S., Wang H. (eds.) ASIACRYPT 2020. LNCS, vol. 12492, pp. 259–288. Springer (2020). https://doi.org/10.1007/978-3-030-64834-3_9.
24. Esgin M.F., Steinfeld R., Liu J.K., Liu D.: Lattice-based zero-knowledge proofs: new techniques for shorter and faster constructions and applications. In: Boldyreva A., Micciancio D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 115–146. Springer (2019). https://doi.org/10.1007/978-3-030-26948-7_5.
25. Esgin M.F., Steinfeld R., Sakzad A., Liu J.K., Liu D.: Short lattice-based one-out-of-many proofs and applications to ring signatures. In: Deng R.H., Gauthier-Umaña V., Ochoa M., Yung M. (eds.) ACNS 2019. LNCS, vol. 11464, pp. 67–88. Springer (2019). https://doi.org/10.1007/978-3-030-21568-2_4.
26. Fiat A., Shamir A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer (1986). https://doi.org/10.1007/3-540-47721-7_12.
27. Groth J., Kohlweiss M.: One-out-of-many proofs: or how to leak a secret and spend a coin. In: Oswald E., Fischlin M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 253–280. Springer (2015). https://doi.org/10.1007/978-3-662-46803-6_9.

28. Haque A., Krenn S., Slamanig D., Striecks C.: Logarithmic-size (linkable) threshold ring signatures in the plain model. In: Hanaoka G., Shikata J., Watanabe Y. (eds.) PKC 2022. LNCS, vol. 13178, pp. 437–467. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_15.
29. Haque A., Scafuro A.: Threshold ring signatures: new definitions and post-quantum security. In: Kiayias A., Kohlweiss M., Wallden P., Zikas V. (eds.) PKC 2020. LNCS, vol. 12111, pp. 423–452. Springer (2020). https://doi.org/10.1007/978-3-030-45388-6_15.
30. Kiltz E., Lyubashevsky V., Schaffner C.: A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In: Nielsen J.B., Rijmen V. (eds.) EUROCRYPT 2018. LNCS, vol. 10822, pp. 552–586. Springer (2018). https://doi.org/10.1007/978-3-319-78372-7_18.
31. Kiltz E., Masny D., Pan J.: Optimal security proofs for signatures from identification schemes. In: Robshaw M., Katz J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 33–61. Springer (2016). https://doi.org/10.1007/978-3-662-53008-5_2.
32. Langlois A., Stehlé D.: Worst-case to average-case reductions for module lattices. Des. Codes Cryptogr. **75**(3), 565–599 (2015). https://doi.org/10.1007/s10623-014-9938-4.
33. Lin S., Li J., Liang W.: Research on strong supervision algorithm model based on blockchain in e-government. In: IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC), pp. 345–349. IEEE (2020).
34. Liu J.K., Wei V.K., Wong D.S.: A separable threshold ring signature scheme. In: Lim J.I., Lee D.H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 12–26. Springer (2003). https://doi.org/10.1007/978-3-540-24691-6_2.
35. Lyubashevsky V.: Fiat–Shamir with aborts: applications to lattice and factoring-based signatures. In: Matsui M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer (2009). https://doi.org/10.1007/978-3-642-10366-7_35.
36. Lyubashevsky V.: Lattice signatures without trapdoors. In: Pointcheval D., Johansson T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer (2012). https://doi.org/10.1007/978-3-642-29011-4_43.
37. Lyubashevsky V., Nguyen N.K.: BLOOM: bimodal lattice one-out-of-many proofs and applications. IACR Cryptol. ePrint Arch., p. 1307 (2022). https://eprint.iacr.org/2022/1307.
38. Lyubashevsky V., Nguyen N.K., Seiler G.: SMILE: set membership from ideal lattices with applications to ring signatures and confidential transactions. In: Malkin T., Peikert C. (eds.) CRYPTO 2021. LNCS, vol. 12826, pp. 611–640. Springer (2021). https://doi.org/10.1007/978-3-030-84245-1_21.
39. Lyubashevsky V., Seiler G.: Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In: Nielsen J.B., Rijmen V. (eds.) EUROCRYPT 2018. LNCS, vol. 10820, pp. 204–224. Springer (2018). https://doi.org/10.1007/978-3-319-78381-9_8.
40. Melchor C.A., Cayrel P., Gaborit P., Laguillaumie F.: A new efficient threshold ring signature scheme based on coding theory. IEEE Trans. Inf. Theory **57**(7), 4833–4842 (2011). https://doi.org/10.1109/TIT.2011.2145950.
41. Micciancio D., Peikert C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval D., Johansson T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer (2012). https://doi.org/10.1007/978-3-642-29011-4_41.
42. Micciancio D., Peikert C.: Hardness of SIS and LWE with small parameters. In: Canetti R., Garay J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 21–39. Springer (2013). https://doi.org/10.1007/978-3-642-40041-4_2.
43. Micciancio D., Regev O.: Lattice-based cryptography. In: Post-quantum Cryptography, pp. 147–191. Springer (2009).
44. Okamoto T., Tso R., Yamaguchi M., Okamoto E.: A k-out-of-n ring signature with flexible participation for signers. IACR Cryptol. ePrint Arch., p. 728 (2018). https://eprint.iacr.org/2018/728.
45. Peikert C., Rosen A.: Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In: Halevi S., Rabin T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 145–166. Springer (2006). https://doi.org/10.1007/11681878_8.
46. Peikert C., Shiehian S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva A., Micciancio D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 89–114. Springer (2019). https://doi.org/10.1007/978-3-030-26948-7_4.
47. Rivest R.L., Shamir A., Tauman Y.: How to leak a secret. In: Boyd C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer (2001). https://doi.org/10.1007/3-540-45682-1_32.
48. Schnorr C.: Efficient identification and signatures for smart cards. In: Brassard G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer (1989). https://doi.org/10.1007/0-387-34805-0_22.
49. Wagner D.A.: A generalized birthday problem. In: Yung M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 288–303. Springer (2002). https://doi.org/10.1007/3-540-45708-9_19.

50. Wang Z., Fan J.: Flexible threshold ring signature in chronological order for privacy protection in edge computing. IEEE Trans. Cloud Comput. **10**(2), 1253–1261 (2022). https://doi.org/10.1109/TCC.2020.2974954.
51. Yuen T.H., Esgin M.F., Liu J.K., Au M.H., Ding Z.: DualRing: generic construction of ring signatures with efficient instantiations. In: Malkin T., Peikert C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 251–281. Springer (2021). https://doi.org/10.1007/978-3-030-84242-0_10.
52. Yuen T.H., Liu J.K., Au M.H., Susilo W., Zhou J.: Threshold ring signature without random oracles. In: Cheung B.S.N., Hui L.C.K., Sandhu R.S., Wong D.S. (eds.) ASIACCS 2011. pp. 261–267. ACM (2011). https://doi.org/10.1145/1966913.1966947.