Tagged Chameleon Hash from Lattices and Application to Redactable Blockchain

Yiming Li[1,2] and Shengli Liu[1,2](✉)

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
{lym_sjtu,slliu}@sjtu.edu.cn

[2] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China

Abstract. Chameleon hash (CH) is a trapdoor hash function. Generally it is hard to find collisions, but with the help of a trapdoor, finding collisions becomes easy. CH plays an important role in converting a conventional blockchain to a redactable one. However, most of existing CH schemes are too weak to support redactable blockchains. The currently known CH schemes serving for redactable blockchains have the best security of so-called "full collision resistance (f-CR)", but they are built either in the random oracle model or rely on heavy tools like the simulation-sound extractable non-interactive zero-knowledge (SSE-NIZK) proof system. Moreover, up to now there is no CH scheme with post-quantum f-CR security in the standard model. Therefore, no CH can support redactable blockchains in a post-quantum way without relying on random oracles.

In this paper, we introduce a variant of CH, namely tagged chameleon hash (tCH). Tagged chameleon hash takes a tag into hash evaluations and collision finding algorithms. We define two security notions for tCH, restricted collision resistance (r-CR) and full collision resistance (f-CR), and prove the equivalence between r-CR and f-CR when tCH works in the one-time tag mode. We propose a tCH scheme from lattices without using any NIZK proof, and prove that its restricted collision resistance is reduced to the Short Integer Solution (SIS) assumption in the standard model. We also show how to apply tCH to a blockchain in one-time tag mode so that the blockchain can be compiled to a redactable one. Our tCH scheme provides the first post-quantum solution for redactable blockchains, without resorting to random oracles or NIZK proofs. Besides, we also construct a more efficient tCH scheme with r-CR tightly reduced to SIS in the random oracle model, which may be of independent interest.

Keywords: Tagged chameleon hash · Lattice-based cryptography · Redactable blockchain

1 Introduction

The chameleon hash (CH) was first introduced by Krawczyk and Rabin [25] and it can be seen as a trapdoor collision resistant hash function. Informally,

Q. Tang and V. Teague (Eds.): PKC 2024, LNCS 14603, pp. 288–320, 2024.
https://doi.org/10.1007/978-3-031-57725-3_10

a CH is associated with a public parameter pp and a trapdoor td. With pp, one can efficiently evaluate the hash value for any given message, and with td, one can efficiently find collisions for any target hash value. The fundamental security requirement of a chameleon hash, namely the collision resistance, assures that any adversary cannot find collisions without the knowledge of td. Since its introduction, chameleon hash has developed different security notions, serving for a wide range of applications. There are mainly four kinds of security notions for CH, as we summarized below.

Weak Collision Resistance. The weak collision resistance (w-CR) for CH is the basic security requirement formalized in [25] and it assures the infeasibility of finding a collision $(h^*, m^*, r^*, m'^*, r'^*)$ s.t. $m^* \neq m'^*$ but $h^* = \mathsf{Hash}(m^*; r^*) = \mathsf{Hash}(m'^*; r'^*)$ without the trapdoor. The w-CR CH is often used to construct chameleon signatures [25] and lift non-adaptively secure signatures to adaptively secure ones [23,28,31]. However, most of CH schemes with w-CR suffer from a so-called key-exposure problem, that is, anyone can recover the trapdoor after seeing only one collision with two different messages. A sequence of works [4,10,11] have identified this problem and proposed different CHs with key-exposure freeness. However, such CHs are still insufficient for the security requirements asked from more complicated applications.

Enhanced Collision Resistance. The enhanced collision resistance (e-CR) was first proposed by Ateniese et al. [3] as a strengthening of the weak collision resistance. It assures the infeasibility of finding a collision $(h^*, m^*, r^*, m'^*, r'^*)$ if no collision for this specific h^* has ever been revealed to the adversary before. A chameleon hash with e-CR was suggested to construct a redactable blockchain [3,24,38], but in fact, e-CR is still not strong enough to deal with attacks on a redactable blockchain system as we will discuss later.

Standard Collision Resistance. The standard collision resistance (s-CR) was introduced by Camenisch et al. [9] and it assures the infeasibility of finding a collision $(h^*, m^*, r^*, m'^*, r'^*)$ if no collision involving the target message m^* has ever been revealed to the adversary before. A CH with s-CR can be used to construct sanitizable signatures [9] and redactable blockchains. However, s-CR is still insufficient for the security requirements asked by a redactable blockchain.

Full Collision Resistance. The full collision resistance (f-CR) was introduced by Derler, Samelin and Slamanig [13] as a combination of e-CR and s-CR[1]. It assures the infeasibility of finding a collision $(h^*, m^*, r^*, m'^*, r'^*)$ if no collision for the target hash-message pair (h^*, m^*) has ever been revealed to the adversary before. To the best of our knowledge, f-CR is the strongest one among all security notions of a chameleon hash, and it is adequate for most of the applications of chameleon hash, especially for redactable blockchain.

Redactable blockchain is an important application of a chameleon hash and it has high requirements for CH. Recall that blockchain was originally designed

[1] According to [13], e-CR and s-CR are incomparable, which means that neither e-CR implies s-CR nor s-CR implies e-CR.

to satisfy immutability, i.e., the infeasibility of tampering the messages stored in the blocks. However, rigid immutability might not be friendly for healthy developments of blockchains. For example, once some illegal or malicious information is stored in blocks, it is hardly to be erased any more. In fact, the European General Data Protection Regulation (GDPR) has suggested the "right to be forgotten". Therefore, researches on technical tools for changing or deleting sensitive information stored in blocks draw more attentions in the academic society. This yields the so-called *redactable* blockchain. In a redactable blockchain, immutability becomes flexible in the sense that a trusted regulation party (or multi-parties) can use a trapdoor to redact the chain by rewriting blocks in the chain according to the well-accepted regulation rules. We refer readers to [3] for more discussions about the necessity of a redactable blockchain.

Given the concept of redactable blockchain, how to do the redactions in a secure and controlled way has become a critical problem to be solved. As summarized by [39], there are four mechanisms to achieve redactable blockchains, that is, the consensus-based, chameleon hash-based, mate-transaction-based, and pruning-based. For the consensus-based mechanisms, redactions are performed by on-chain voting, like the hard fork and [14,36]; for the mate-transaction-based mechanisms, redactions are triggered by a special transaction called the mate-transaction, like [16,17,34]; for the pruning-based mechanisms, redactions are made by pruning transactions or blocks when some conditions are satisfied, like [27,35]. Ateniese et al. [3] suggested to construct redactable blockchains with the help of CH, that is, the chameleon hash-based redactable blockchains. In this paper, we focus on this type of redaction mechanism.

Below we briefly describe the suggestion of constructing a redactable blockchain from a chameleon hash in [3] and show the security requirements of CH.

Redactable Blockchain from CH. A conventional blockchain can be converted to a redactable one by replacing one of the hash functions used to construct blocks with a chameleon one [3]. Let $\mathcal{H} = (\mathsf{Setup}, \mathsf{Hash}, \mathsf{Adapt})$ be a chameleon hash, where the setup algorithm is used to generate the public parameter and trapdoor, i.e., $(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$, the hash algorithm is used to evaluate the hash value for a given message with some randomness, i.e., $h \leftarrow \mathsf{Hash}(m; r)$, and the adaptation algorithm is used to find a collision with td, i.e., $r' \leftarrow \mathsf{Adapt}(td, h, m, r, m')$ s.t. $h = \mathsf{Hash}(m; r) = \mathsf{Hash}(m'; r')$.

For a CH-based redactable blockchain, a trusted regulation party is granted to generate $(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$ and then publish pp. A miner collects the message m, evaluates $h \leftarrow \mathsf{Hash}(m; r)$, constructs a valid block B containing the triple (h, m, r) as well as other information required, and finally appends it to the blockchain. When an adaptation is required from m to m' in some block B[2], the trusted authority computes $r' \leftarrow \mathsf{Adapt}(td, h, m, r, m')$, replaces (m, r) stored in B with (m', r') while keeping other information unchanged, and finally publishes the redacted block. In this way, we obtain a redactable blockchain.

[2] Here, adaptations are only allowed for blocks considered to be settled in the redactable blockchain system.

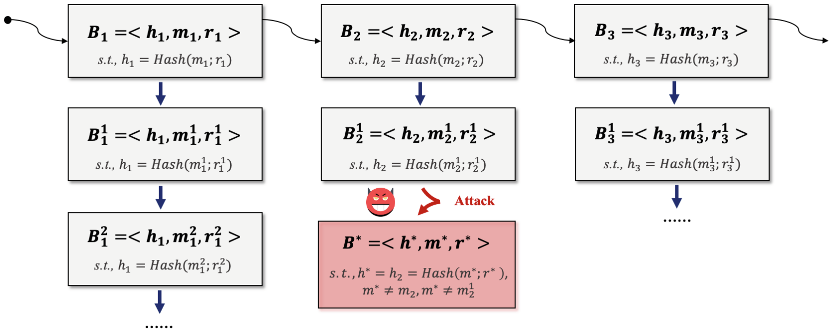

$B_1 =< h_1, m_1, r_1 >$
s.t., $h_1 = Hash(m_1; r_1)$
$B_2 =< h_2, m_2, r_2 >$
s.t., $h_2 = Hash(m_2; r_2)$
$B_3 =< h_3, m_3, r_3 >$
s.t., $h_3 = Hash(m_3; r_3)$
$B_1^1 =< h_1, m_1^1, r_1^1 >$
s.t., $h_1 = Hash(m_1^1; r_1^1)$
$B_2^1 =< h_2, m_2^1, r_2^1 >$
s.t., $h_2 = Hash(m_2^1; r_2^1)$
$B_3^1 =< h_3, m_3^1, r_3^1 >$
s.t., $h_3 = Hash(m_3^1; r_3^1)$
Attack
$B_1^2 =< h_1, m_1^2, r_1^2 >$
s.t., $h_1 = Hash(m_1^2; r_1^2)$
$B^* =< h^*, m^*, r^* >$
s.t., $h^* = h_2 = Hash(m^*; r^*)$,
$m^* \neq m_2, m^* \neq m_2^1$
......
......

Fig. 1. Possible attack on redactions in a redactable blockchain. The adversary sees all grey blocks and tries to create the red one. The blocks link to a chain in the way that previous hash value h_i constitutes a part of message m_{i+1} in the next block. The down-arrows in dark-blue denote authorized adaptations done by the trusted regulation party and the arrow in red denotes an attack. (Color figure online)

Security Requirements of CH in Redactable Blockchain. In a redactable blockchain, each block B_j records information of (h_j, m_j, r_j) and we denote it by $B_j = \langle h_j, m_j, r_j \rangle$. Adaptations for block B_j result in multiple new adapted blocks $\{B_j^i = \langle h_j, m_j^i, r_j^i \rangle\}_{i \in [n_j]}$ s.t. $h_j = \mathsf{Hash}(m_j; r_j) = \mathsf{Hash}(m_j^i; r_j^i)$ for $i \in [n_j]$. Now we consider the adversary's attack on redactions in a redactable blockchain. The adversary sees all original blocks $B_1, B_2, B_3, \cdots$ and all corresponding adapted blocks $\{B_1^i\}_{i \in [n_1]}, \{B_2^i\}_{i \in [n_2]}, \{B_3^i\}_{i \in [n_3]}, \cdots$, where each $n_j = \mathsf{poly}(\kappa)$ denotes the number of adaptations for block B_j. The aim of an adversary is to redact the chain by adapting some block $B_j = \langle h_j, m_j, r_j \rangle$ to a new one $B^* = \langle h^*, m^*, r^* \rangle$ s.t. $h^* = h_j = \mathsf{Hash}(m_j; r_j) = \mathsf{Hash}(m^*; r^*)$, where m^* is the adapted message satisfying $m^* \notin \{m_j\} \cup \{m_j^i\}_{i \in [n_j]}$, in other words, (h^*, m^*) is fresh w.r.t. B_j and $\{B_j^i\}_{i \in [n_j]}$. Note that we do not exclude the possibility that m^* belongs to $\{m_{j'}\} \cup \{m_{j'}^i\}_{i \in [n_{j'}]}$ with $j \neq j'$. See Fig. 1.

Obviously, to make sure that the adversary succeeds in redacting blocks with negligible probability, it suffices for a chameleon hash to be full collision resistant. In contrast, e-CR and s-CR are not sufficient. Firstly, the adversary can obtain multiple adapted blocks $\{B_j^i = \langle h_j, m_j^i, r_j^i \rangle\}_{i \in [n_j]}$ for the target hash value $h_j = h^*$, so, e-CR is not enough for CH. Secondly, the adversary may obtain some adapted block $B_{j'}^i = \langle h_{j'}, m^*, r_{j'}^i \rangle$ with $j \neq j'$, and hence s-CR is not enough either.

To the best of our knowledge, only a CH with f-CR security is sufficient to the security requirements of a redactable blockchain. However, existing CH schemes with f-CR [12,13] are all generic constructions relying on some heavy building blocks like the simulation-sound extractable non-interactive zero knowledge (SSE-NIZK) proof system [13]. Besides, almost all instantiations of CH with f-CR security are based on pairings or the discrete logarithm (DL) assumption, and hence are not secure against quantum adversaries. The only known post-quantum instantiation is based on the learning parity with noise (LPN) assumption in the random oracle (RO) model [12]. Then a natural question arises:

"*Can we construct a post-quantum chameleon hash function serving for a redactable blockchain in the standard model (especially without relying on a NIZK proof system)?*"

In this paper, we provide a new approach to this problem. We take into considerations some nice properties of a redactable blockchain so that the security requirements for CH can be weakened. That makes possible simpler constructions of CH serving for a secure redactable blockchain. In more details, we have the following three observations for a CH-based redactable blockchain.

- **Observation 1.** Each settled block can be uniquely indexed by a unique identifier τ (like the timestamp, the hash value of its previous block or its position in the chain). Taking τ into account results in blocks of the form $B = \langle \tau, h, m, r \rangle$.
- **Observation 2.** Each block has a chameleon hash value h and identifier τ, and adaptations towards that block keep h and τ unchanged. Together with observation 1, we know that each tag τ is uniquely bound with one block (and hence the chameleon hash value h).
- **Observation 3.** All adaptations towards a specific block are made with fresh messages. In a redactable blockchain, this can be easily accomplished by appending a unique (e.g. increasing) counter value to the adapted message.

Now we additionally take τ as input for chameleon hash evaluations and adaptations, and this results in a new variant of CH, namely the *tagged CH* (tCH). Next we consider the full collision resistance for a tagged CH. The adversary can see many tuples (τ, h, m, r) as well as their adaptations (τ, h, m', r'), where m is the original message and m' is the adapted message s.t. $h = \mathsf{Hash}(\tau, m; r) = \mathsf{Hash}(\tau, m'; r')$. Let $\mathcal{Q}$ record tuples (τ, h, m) and the adapted tuples (τ, h, m'). The adversary wins if it finally comes up with a forgery $(\tau^*, h^*, m^*, r^*, m'^*, r'^*)$ such that

$$h^* = \mathsf{Hash}(\tau^*, m^*; r^*) = \mathsf{Hash}(\tau^*, m'^*; r'^*),\ m^* \neq m'^*,\ (\tau^*, h^*, m^*) \notin \mathcal{Q}. \quad (1)$$

Obviously, the full collision resistance of tCH is sufficient for a redactable blockchain. But actually, the three observations can help to change the security requirements of tCH to a weaker variant. Note that in (1.1), we have

$$\begin{aligned} & (\tau^*, h^*, m^*) \notin \mathcal{Q} \\ \Leftrightarrow & \big((\tau^*, h^*, m^*) \notin \mathcal{Q} \wedge (\tau^*, \cdot, m'^*) \notin \mathcal{Q}\big) \vee \big((\tau^*, h^*, m^*) \notin \mathcal{Q} \wedge (\tau^*, \cdot, m'^*) \in \mathcal{Q}\big) \\ \Leftrightarrow & \big((\tau^*, \cdot, m^*) \notin \mathcal{Q} \wedge (\tau^*, \cdot, m'^*) \notin \mathcal{Q}\big) \vee \big((\tau^*, \cdot, m^*) \notin \mathcal{Q} \wedge (\tau^*, h^*, m'^*) \in \mathcal{Q}\big), \end{aligned}$$

where $(\tau^*, \cdot, m'^*) \notin \mathcal{Q}$ means that there exists no h such that $(\tau^*, h, m'^*) \in \mathcal{Q}$, while $(\tau^*, \cdot, m'^*) \in \mathcal{Q}$ means that there exists an h such that $(\tau^*, h, m'^*) \in \mathcal{Q}$. Here "$\Leftarrow$" holds obviously, and "$\Rightarrow$" holds due to the observation 2. By observation 2, for any adapted blocks with $(\tau^*, h^*, \cdot, \cdot)$ we know that τ^* is uniquely bound with h^*, so $(\tau^*, h^*, m^*) \notin \mathcal{Q} \Rightarrow (\tau^*, \cdot, m^*) \notin \mathcal{Q}$ and $(\tau^*, \cdot, m'^*) \in \mathcal{Q} \Rightarrow (\tau^*, h^*, m'^*) \in \mathcal{Q}$ (otherwise, τ^* corresponds to both h^* and some $h \neq h^*$ in the blockchain system, which is impossible).

Define a predicate Valid as $\mathsf{Valid}(\tau^*, h^*, m^*, m'^*) = 1$ if $\big((\tau^*, \cdot, m^*) \notin \mathcal{Q} \wedge (\tau^*, \cdot, m'^*) \notin \mathcal{Q}\big) \vee \big((\tau^*, \cdot, m^*) \notin \mathcal{Q} \wedge (\tau^*, h^*, m'^*) \in \mathcal{Q}\big)$. Now (1.1) becomes:

$$h^* = \mathsf{Hash}(\tau^*, m^*; r^*) = \mathsf{Hash}(\tau^*, m'^*; r'^*),\ m^* \neq m'^*,\ \mathsf{Valid}(\tau^*, h^*, m^*, m'^*) = 1.$$

According to observation 2 again, it is reasonable to assume that there do not exist $(\tau, h, \cdot)$ and $(\tau, h'', \cdot)$ with $h \neq h''$ among those tuples and adapted tuples contained in $\mathcal{Q}$. Furthermore, according to observation 3, we can require that all adapted messages w.r.t. a block (and hence a unique τ) are distinct.

Hence for redactable blockchain, we arrive at a security requirement for tCH which is weaker than the full collision resistance. We call such a security requirement *restricted collision resistance* since it has more restrictions on adversaries compared with the full one (see Fig. 3 for their formal definitions). Now the problem can be simplified as follows.

"*Can we construct a post-quantum tagged chameleon hash function with restricted collision resistance in the standard model (especially without relying on a NIZK proof system)?*"

1.1 Our Contributions

In this paper, we answer the above question in the affirmative and have made the following three contributions.

New Concept of Tagged Chameleon Hash (tCH). We introduce a new primitive, named tagged chameleon hash (tCH), which additionally takes as input a tag τ for hash evaluations and adaptations. We provide two CR security notions for our tCH. One is the full collision resistance (f-CR) and the other is the restricted collision resistance (r-CR). The full collision resistance is defined similar to that of a tag-free CH [13]. That is, it is infeasible to find $(\tau^*, h^*, m^*, r^*, m'^*, r'^*)$ s.t. $m^* \neq m'^*$ and $h^* = \mathsf{Hash}(\tau^*, m^*; r^*) = \mathsf{Hash}(\tau^*, m'^*; r'^*)$ even if the adversary sees many adaptation outputs r' by issuing queries (τ, h, m, r, m') of its choice. The only limitation is that (τ^*, h^*, m^*) does not appear in its queries. Restricted collision resistance is weaker than the full one in the sense that the adversary's behaviors and winning conditions are further restricted. Meanwhile, we also require *statistical indistinguishability* from tCH which asks that the hash value and randomness are statistically close to the adapted ones.

We show that if tCH works in the one-time tag mode, the two CR security notions are equivalent to each other. Here the one-time tag mode requires that each invocation of hash evaluation takes a fresh and distinct tag as input.

Constructions of tCH from Lattices. We provide two constructions of tCH from lattices and prove their r-CR security.

- Our first tCH construction achieves the restricted collision resistance in the standard model. The restricted collision resistance of our tCH is tightly reduced to the SIS assumption and the pseudorandomness of a pseudorandom function (PRF). Given the LWE-based PRFs like [5], our construction yields the first r-CR secure tCH from LWE and SIS in the standard model.
- Our second tCH construction achieves the restricted collision resistance in the random oracle model. It is more efficient than the first one and is tightly reduced to the SIS assumption.

According to the relation between f-CR and r-CR, both of our two tCHs can provide security guarantee as good as f-CR when working in the one-time tag mode. We stress that our tCH schemes are free of NIZK proof systems.

Application of tCH in Redactable Blockchain. Each settled block can be uniquely indexed by a unique identifier τ in the redactable blockchain. So different blocks have distinct identifiers τ. When a tCH is applied to the blockchain, we can take τ as the tag of tCH to compute hash values for messages stored in blocks, and hence each hash value (for a settled block) is computed from a distinct tag. Note that, adaptations are made only for those settled blocks. In this way, the tCH already works in the one-time tag mode for the redactable blockchain. Therefore, our tCH schemes with restricted collision resistance serve for redactable blockchains perfectly.

1.2 Related Works

Chameleon Hash. Krawczyk and Rabin [25] proposed two CH constructions with w-CR based on the claw-free trapdoor permutations [22] and the Pedersen's commitment scheme [32] respectively. Chen, Zhang and Kim [11] proposed the first key-exposure free CH from the computational Diffie-Hellman (CDH) assumption based on the gap Diffie-Hellman (GDH) group. Ateniese and de Medeiros [4] also proposed several key-exposure free CHs from various assumptions like the RSA and the discrete logarithm (DL) assumptions. Later in 2017, Ateniese et al. [3] proposed a generic way to lift a CH from w-CR to e-CR with helps of a CPA secure public key encryption (PKE) and a true-simulation extractable non-interactive zero knowledge (tSE-NIZK) proof system. Ateniese et al. [3] instantiated the generic construction from the decisional Diffie-Hellman assumption in the random oracle model, and from k-linear assumption in the standard model, respectively. Since then, several efficient CH schemes with e-CR have been proposed from various assumptions. Khalili, Dakhilalian and Susilo [24] proposed two CHs with e-CR: one is constructed by combining a weak CH with Groth-Sahai NIZK proof and Cramer-Shoup PKE, and the other is constructed with the ZK-SNARKs. Wu, Ke and Du [38] gave two CH schemes from the lattice-based assumptions in the generic group model (GGM) and in the random oracle model (ROM), respectively. As for s-CR, Camenisch et al. [9] proposed an s-CR secure CH based on the one-more RSA assumption in the random oracle model. Recently, Derler, Samelin and Slamanig [13] suggested f-CR as a more desirable security notion for a CH, and proposed a generic construction of a f-CR secure CH with building blocks a CPA secure PKE and a simulation-sound extractable non-interactive zero knowledge (SSE-NIZK) proof. Derler, Samelin and Slamanig [13] provided instantiations of the generic construction based on the DDH assumption in ROM, and based on the symmetric external Diffie-Hellman (SXDH) assumption in the standard model, respectively. Later, Deler et al. [12] proposed a relatively simpler generic f-CR secure CH construction with building blocks a non-interactive commitment scheme and also an SSE-NIZK. Deler et al. [12] instantiated the generic construction from the DL assumption in ROM, and from the LPN assumption in ROM, respectively.

1.3 Technique Overview

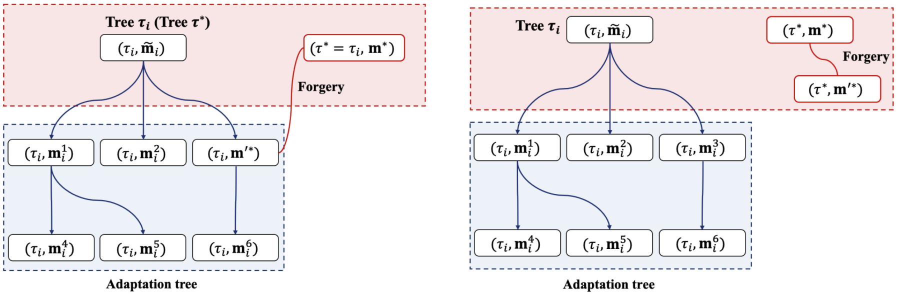
In this subsection, we provide high-level ideas of our tCH constructions from lattices. We propose two tCH schemes: one is in the standard model and the other is in the random oracle model. Both of our tCHs are constructed and proved following a partitioning proof strategy, which has been used in designing advanced signatures and public-key encryptions [6,7,37]. To do the "partitioning", our tCH in the standard model uses a pseudorandom function (PRF) and homomorphic evaluation techniques [6–8,20], while our tCH in the ROM relies on the re-programmable property of random oracles.

Here we provide a brief description of our tCH in the standard model. The public parameter pp consists of a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a PRF's secret key $\mathbf{k}$ (only used for the security proof), and random matrices $\mathbf{A}_1, \ldots, \mathbf{A}_k, \hat{\mathbf{A}}_1, \ldots, \hat{\mathbf{A}}_h \in \mathbb{Z}_q^{n\times w}$ (which will be used for embedding $\mathbf{k}$ and messages to be hashed in the security proof). The master trapdoor mtd is set as a trapdoor $\mathbf{T}_\mathbf{A}$ of $\mathbf{A}$ s.t. $\mathbf{T}_\mathbf{A}$ is small and $\mathbf{A} \cdot \mathbf{T}_\mathbf{A} = 0^{n\times m}$.

To hash a message $\mathbf{m} = (m_1, \ldots, m_h) \in \{0,1\}^h$ w.r.t. a tag τ, we first sample $\mathbf{y}$ uniformly at random, and then construct a circuit $C[\tau\|\mathbf{m}, \mathbf{y}](\cdot)$ s.t. $C[\tau\|\mathbf{m}, \mathbf{y}](\mathbf{k})$ returns 1 if $\mathsf{PRF}(\mathbf{k}, \tau\|\mathbf{m}) = \mathbf{y}$, and returns 0 otherwise. We further construct a matrix $\mathbf{F} := [\mathbf{A}|\mathbf{A}_{\mathsf{prf}}] \in \mathbb{Z}_q^{n\times(m+w)}$ from pp, $\mathbf{m}$ and τ, where $\mathbf{A}_{\mathsf{prf}}$ is generated through homomorphic evaluations on $C[\tau\|\mathbf{m}, \mathbf{y}](\cdot)$ with $\mathbf{A}_1, \ldots, \mathbf{A}_k$. The hash value is computed as $\mathbf{h} := \mathbf{F} \cdot \mathbf{e}$ with $\mathbf{e} \in \mathbb{Z}^{m+w}$ a short integer vector sampled from the discrete Gaussian distribution; the randomness r includes $\mathbf{e}$, $\mathbf{y}$ and other randomnesses used to generate $\mathbf{F}$.

To find a collision r' towards $(\tau, \mathbf{h}, \mathbf{m}, r, \mathbf{m}')$ so that $(\tau, \mathbf{m}, r)$ and $(\tau, \mathbf{m}', r')$ both hash to $\mathbf{h}$, we first construct $\mathbf{F}' := [\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}]$ from pp, τ and $\mathbf{m}'$. Then we can find a short integer vector $\mathbf{e}'$ s.t. $\mathbf{h} = \mathbf{F}' \cdot \mathbf{e}'$ with the help of $\mathbf{T}_\mathbf{A}$ through trapdoor delegation [33] and preimage sampling [19].

Now we are ready to sketch the security proof. In the security experiment of r-CR, adversary $\mathcal{A}$ can make multiple adaptation queries and for each query $(\tau_i, \mathbf{h}_i, \mathbf{m}_i, r_i, \mathbf{m}'_i)$, the challenger responds $\mathcal{A}$ with a randomness r'_i s.t. $\mathbf{h}_i = \mathsf{Hash}(\tau_i, \mathbf{m}_i; r_i) = \mathsf{Hash}(\tau_i, \mathbf{m}'_i; r'_i)$. Then in the challenge phase, $\mathcal{A}$ submits its forgery $(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*, \mathbf{m}'^*, r'^*)$ and it wins if $\mathbf{h}^* = \mathsf{Hash}(\tau^*, \mathbf{m}^*; r^*) = \mathsf{Hash}(\tau^*, \mathbf{m}'^*; r'^*)$, $\mathbf{m}'^* \neq \mathbf{m}^*$ and $\mathsf{Valid}(\tau^*, \mathbf{h}^*, \mathbf{m}^*, \mathbf{m}'^*) = 1$. The reduction algorithm


Tree τ_i (Tree τ^*)
$(\tau_i, \tilde{\mathbf{m}}_i)$
$(\tau^* = \tau_i, \mathbf{m}^*)$
Forgery
$(\tau_i, \mathbf{m}_i^1)$
$(\tau_i, \mathbf{m}_i^2)$
$(\tau_i, \mathbf{m}'^*)$
$(\tau_i, \mathbf{m}_i^4)$
$(\tau_i, \mathbf{m}_i^5)$
$(\tau_i, \mathbf{m}_i^6)$
Adaptation tree
Tree τ_i
$(\tau_i, \tilde{\mathbf{m}}_i)$
$(\tau^*, \mathbf{m}^*)$
Forgery
$(\tau^*, \mathbf{m}'^*)$
$(\tau_i, \mathbf{m}_i^1)$
$(\tau_i, \mathbf{m}_i^2)$
$(\tau_i, \mathbf{m}_i^3)$
$(\tau_i, \mathbf{m}_i^4)$
$(\tau_i, \mathbf{m}_i^5)$
$(\tau_i, \mathbf{m}_i^6)$
Adaptation tree

(a) Case I

(b) Case II

Fig. 2. A partition on tag-message pairs: those in blue dashed boxes with $\mathsf{PRF}(\mathbf{k}, \tau\|\mathbf{m}) = \mathbf{y}$, and those in red dashed boxes with $\mathsf{PRF}(\mathbf{k}, \tau\|\mathbf{m}) \neq \mathbf{y}$. Here "$(\tau, \mathbf{m}_i) \rightarrow (\tau, \mathbf{m}_j)$" with arrows in dark-blue means an adaptation from tuple $(\tau, \mathbf{m}_i)$ to $(\tau, \mathbf{m}_j)$ made by the challenger during the adaptation query phase; "$(\tau^*, \mathbf{m}^*) \sim (\tau^*, \mathbf{m}'^*)$" means the forgery tuples submitted by the adversary. (Color figure online)

can embed a SIS problem instance into the random matrix $\mathbf{A}$, but then there are two problems to be solved.

- **Problem I:** Since $\mathbf{A}$ is a SIS instance now, the trapdoor $\mathbf{T_A}$ of $\mathbf{A}$ is unknown to the reduction algorithm. In this case, how to find a collision for $(\tau_i, \mathbf{h}_i, \mathbf{m}_i, r_i, \mathbf{m}_i')$ without $\mathbf{T_A}$ upon the adversary's adaptation queries?
- **Problem II:** How does the reduction algorithm derive a valid solution to the SIS problem when $\mathcal{A}$ successfully finds a valid collision?

For expression simplicity, let's introduce some facts for tCH first. Consider all valid adaptation queries $\{(\tau_i, \mathbf{h}_i, \mathbf{m}_i, r_i, \mathbf{m}_i')\}$ submitted by $\mathcal{A}$ in the r-CR security experiment, where τ_i is bound to a unique $\mathbf{h}_i$ and $\mathsf{Hash}(\tau_i, \mathbf{m}_i; r_i) = \mathbf{h}_i$. Then all valid adaptation queries constitute a sequence of trees. Let τ_i index the trees. Tree τ_i has a root $(\tau_i, \tilde{\mathbf{m}}_i)$ which is NOT an adapted tuple, and all non-root nodes $\{(\tau_i, \mathbf{m}_i')\}$ in the tree are adapted from their parent nodes. For $\mathcal{A}$'s forgery $(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*, \mathbf{m}'^*, r'^*)$, it requires that $(\tau^*, \mathbf{m}^*)$ never appears in adaptation queries, so $(\tau^*, \mathbf{m}^*)$ does not belong to any adaptation tree. The other tuple $(\tau^*, \mathbf{m}'^*)$ either lies in some adaptation tree τ_i (Case I), or does not belong to any adaptation tree (Case II). See Fig. 2 for a demonstration.

Now let us see how to solve the above two problems. We give an adaptive partition of all tag-and-message tuples $\{(\tau_i, \mathbf{m}_i), (\tau_i, \mathbf{m}_i')\}$ in adaptation queries and tuples $(\tau^*, \mathbf{m}^*), (\tau^*, \mathbf{m}'^*)$ in the forgery according to whether $\mathsf{PRF}(\mathbf{k}, \tau\|\mathbf{m}) = \mathbf{y}$, where $\mathbf{y}$ is a randomness included in r.

- For the root node $(\tau_i, \tilde{\mathbf{m}}_i)$ in each tree (say tree τ_i), its corresponding $\tilde{\mathbf{y}}_i$ is chosen by the adversary who knows nothing about $\mathsf{PRF}(\mathbf{k}, \tau_i\|\tilde{\mathbf{m}}_i)$. Then $\mathsf{PRF}(\mathbf{k}, \tau_i\|\tilde{\mathbf{m}}_i) \neq \tilde{\mathbf{y}}_i$ due to the pseudorandomness of PRF, and hence $C[\tau_i\|\tilde{\mathbf{m}}_i,\, \tilde{\mathbf{y}}_i](\mathbf{k}) = 0$.

- For those non-root nodes in tree τ_i, they must be adapted tuples $(\tau_i, \mathbf{m}'_i)$. We choose $\mathbf{y}'_i$ s.t. $\mathbf{y}'_i = \mathsf{PRF}(\mathbf{k}, \tau_i \| \mathbf{m}'_i)$ and hence $C[\tau_i \| \mathbf{m}'_i, \mathbf{y}'_i](\mathbf{k}) = 1$.
- For the node $(\tau^*, \mathbf{m}^*)$ in the forgery, it is submitted by the adversary and does not belong to any adaptation tree, so $\mathsf{PRF}(\mathbf{k}, \tau^* \| \mathbf{m}^*) \neq \mathbf{y}^*$ and hence $C[\tau^* \| \mathbf{m}^*, \mathbf{y}^*](\mathbf{k}) = 0$ due to the pseudorandomness of PRF.
- For the node $(\tau^*, \mathbf{m}'^*)$ in the forgery, we consider two cases.
  - **Case I:** $(\tau^*, \mathbf{m}'^*)$ lies in some tree τ_i. Then it can be a root with $\mathsf{PRF}(\mathbf{k}, \tau^* \| \mathbf{m}'^*) \neq \mathbf{y}'^*$ and $C[\tau^* \| \mathbf{m}'^*, \mathbf{y}'^*](\mathbf{k}) = 0$, or an adapted tuple with $\mathsf{PRF}(\mathbf{k}, \tau^* \| \mathbf{m}'^*) = \mathbf{y}'^*$ and $C[\tau^* \| \mathbf{m}'^*, \mathbf{y}'^*](\mathbf{k}) = 1$.
  - **Case II:** $(\tau^*, \mathbf{m}'^*)$ does not belong to any adaptation tree. Then $\mathsf{PRF}(\mathbf{k}, \tau^* \| \mathbf{m}'^*) \neq \mathbf{y}'^*$ and $C[\tau^* \| \mathbf{m}'^*, \mathbf{y}'^*](\mathbf{k}) = 0$ due to the pseudorandomness of PRF.

In conclusion, for those adapted tuples $\{(\tau_i, \mathbf{m}'_i)\}$, they all satisfy $C[\tau_i \| \mathbf{m}'_i, \mathbf{y}'_i](\mathbf{k}) = 1$, see nodes in blue dashed boxes in Fig. 2. In Case I, we have $(\tau^*, \mathbf{m}^*)$ and $(\tau^* = \tau_i, \tilde{\mathbf{m}}_i)$ s.t. $C[\tau^* \| \mathbf{m}^*, \mathbf{y}^*](\mathbf{k}) = C[\tau^* \| \tilde{\mathbf{m}}_i, \tilde{\mathbf{y}}_i](\mathbf{k}) = 0$. In Case II, we have $(\tau^*, \mathbf{m}^*)$ and $(\tau^*, \mathbf{m}'^*)$ s.t. $C[\tau^* \| \mathbf{m}^*, \mathbf{y}^*](\mathbf{k}) = C[\tau^* \| \mathbf{m}'^*, \mathbf{y}'^*](\mathbf{k}) = 0$. See nodes in red dashed boxes in Fig. 2.

To implement the partitioning strategy, we embed the PRF's key $\mathbf{k}$ in $\mathbf{A}_i$, that is, we generate $\mathbf{A}_i := \mathbf{A}\mathbf{R}_i + k_i \mathbf{G}$ instead of $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times w}$, where $\mathbf{R}_i \in \mathbb{Z}_q^{m \times w}$ is a randomly chosen short matrix and $\mathbf{G} \in \mathbb{Z}_q^{n \times w}$ is the gadget matrix [28]. This change is statistically indistinguishable to $\mathcal{A}$ due to the leftover hash lemma. For each adaptation query $(\tau_i, \mathbf{h}_i, \mathbf{m}_i, r_i, \mathbf{m}'_i)$, we compute $\mathbf{y}'_i := \mathsf{PRF}(\mathbf{k}, \tau_i \| \mathbf{m}'_i)$ instead of $\mathbf{y}'_i \xleftarrow{\$} \{0,1\}^y$, and these two ways of generating $\mathbf{y}'_i$ are computationally indistinguishable due to the pseudorandomness of PRF. Then $C[\tau_i \| \mathbf{m}'_i, \mathbf{y}'_i](\mathbf{k}) = 1$ and we have $\mathbf{F}'_i := [\mathbf{A} | \mathbf{A}'_{\mathsf{prf},i}] = [\mathbf{A} | \mathbf{A}\mathbf{R}'_{\mathsf{prf},i} + C[\tau_i \| \mathbf{m}'_i, \mathbf{y}'_i](\mathbf{k}) \cdot \mathbf{G}] = [\mathbf{A} | \mathbf{A}\mathbf{R}'_{\mathsf{prf},i} + \mathbf{G}]$ through homomorphic evaluations. Note that $\mathbf{F}'_i \cdot [-\mathbf{R}'^{\top}_{\mathsf{prf},i} | \mathbf{I}^{\top}]^{\top} = \mathbf{G}$, and hence $\mathbf{R}'_{\mathsf{prf},i}$ is a gadget trapdoor [28] for $\mathbf{F}'_i$. Given the gadget trapdoor $\mathbf{R}'_{\mathsf{prf},i}$, the reduction can also generate a delegated trapdoor for $\mathbf{F}'_i$ efficiently [28], and then find a collision for $(\tau_i, \mathbf{h}_i, \mathbf{m}_i, r_i, \mathbf{m}'_i)$ with the help of the delegated trapdoor. This solves the problem I. Then for the problem II, as we analyzed before, there exist $(\tau^*, \mathbf{m}^*)$ (the forgery tuple) and some $(\tau^*, \bar{\mathbf{m}})$ s.t. $C[\tau^* \| \mathbf{m}^*, \mathbf{y}^*](\mathbf{k}) = 0$ and $C[\tau^* \| \bar{\mathbf{m}}, \bar{\mathbf{y}}](\mathbf{k}) = 0$. Hence we have $\mathbf{F}^* = [\mathbf{A} | \mathbf{A}^*_{\mathsf{prf}}] = [\mathbf{A} | \mathbf{A}\mathbf{R}^*_{\mathsf{prf}} + C[\tau^* \| \mathbf{m}^*, \mathbf{y}^*](\mathbf{k}) \cdot \mathbf{G}] = [\mathbf{A} | \mathbf{A}\mathbf{R}^*_{\mathsf{prf}} + 0 \cdot \mathbf{G}]$ and $\bar{\mathbf{F}} = [\mathbf{A} | \bar{\mathbf{A}}_{\mathsf{prf}}] = [\mathbf{A} | \mathbf{A}\bar{\mathbf{R}}_{\mathsf{prf}} + C[\tau^* \| \bar{\mathbf{m}}, \bar{\mathbf{y}}](\mathbf{k}) \cdot \mathbf{G}] = [\mathbf{A} | \mathbf{A}\bar{\mathbf{R}}_{\mathsf{prf}} + 0 \cdot \mathbf{G}]$ due to homomorphic evaluations. If $\mathcal{A}$ wins, it holds that $\mathbf{h}^* = [\mathbf{A} | \mathbf{A}\mathbf{R}^*_{\mathsf{prf}}] \cdot \mathbf{e}^* = [\mathbf{A} | \mathbf{A}\bar{\mathbf{R}}_{\mathsf{prf}}] \cdot \bar{\mathbf{e}}$, and then $\mathbf{A} \cdot ([\mathbf{I} | \mathbf{R}^*_{\mathsf{prf}}]\mathbf{e}^* - [\mathbf{I} | \bar{\mathbf{R}}_{\mathsf{prf}}]\bar{\mathbf{e}}) = 0^{n \times m}$. The short vector $\mathbf{v} := ([\mathbf{I} | \mathbf{R}^*_{\mathsf{prf}}]\mathbf{e}^* - [\mathbf{I} | \bar{\mathbf{R}}_{\mathsf{prf}}]\bar{\mathbf{e}})$ serves as a solution to the SIS problem.

There is a subtlety in above SIS solution $\mathbf{v}$ in the reduction. For valid solution, we have to make sure that $\mathbf{v} \neq 0^m$. To this end, we construct those $\mathbf{F}$ as $[\mathbf{A} | \mathbf{A}_{\mathsf{prf}} + \sum_i m_i \hat{\mathbf{A}}_i]$ with public parameters $\hat{\mathbf{A}}_i = \mathbf{A}\hat{\mathbf{R}}_i$. This change does not influence the correctness and the partitioning strategy. We refer readers to Subsect. 4.1 for a more detailed description.

We note that, by replacing the homomorphic evaluations related algorithms with random oracles, namely $\mathbf{F} := [\mathbf{A} | \mathsf{H}(\mathbf{A}, \tau \| \mathbf{m})]$ and H is a hash function modeled as a random oracle, we obtain a tCH in the ROM. To see this, the re-programmable properties of random oracles can also play the role of implementing the partition strategy, and hence the above reduction still holds.

2 Preliminaries

Notations. In this paper, column vectors are denoted by bold lower-case letters like $\mathbf{x}$ and the i-th component of $\mathbf{x}$ is denoted by x_i. Specifically, let 0^k denote the k-dimensional zero vector $(0, 0, \ldots, 0)^\top \in \mathbb{Z}_q^k$. For two bit strings $\mathbf{x}_1 \in \{0,1\}^n$ and $\mathbf{x}_2 \in \{0,1\}^m$, let $\mathbf{x}_1 \| \mathbf{x}_2 \in \{0,1\}^{n+m}$ denote the concatenation of $\mathbf{x}_1$ and $\mathbf{x}_2$. Matrices are denoted by bold upper-case letters like $\mathbf{A}$ and the i-th column of $\mathbf{A}$ is denoted by $\mathbf{a}_i$. The transpose of $\mathbf{A}$ is denoted by $\mathbf{A}^\top$. Let $\mathbf{I}_k \in \{0,1\}^{k\times k}$ denote the k-dimensional identity matrix. For matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{k\times s}$, denote by $\mathbf{A} \otimes \mathbf{B}$ the Kronecker product of $\mathbf{A}$ and $\mathbf{B}$. For a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)^\top \in \mathbb{Z}^n$, let $\|\mathbf{x}\| := (\sum_{i\in[n]} x_i^2)^{\frac{1}{2}}$ denote the ℓ_2 norm of $\mathbf{x}$. For a matrix $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m) \in \mathbb{Z}^{n\times m}$ with $\mathbf{a}_i \in \mathbb{Z}^n$, let $\|\mathbf{A}\| := \max_{i\in[m]} \|\mathbf{a}_i\|$ denote the ℓ_2 norm of $\mathbf{A}$, $\tilde{\mathbf{A}}$ denote the Gram-Schmidt orthogonalization of $\mathbf{A}$, and $s_1(\mathbf{A}) := \max_{\|\mathbf{x}\|=1} \|\mathbf{A}\mathbf{x}\|$ the largest singular value of $\mathbf{A}$.

For an integer $n \in \mathbb{N}$, let $[n]$ denote the finite set $\{1, 2, \ldots, n\}$. For a distribution (or a random variable) X, let $x \leftarrow X$ denote the process of sampling x according to X. For a finite set $\mathcal{X}$, let $x \xleftarrow{\$} \mathcal{X}$ denote the process of sampling x from $\mathcal{X}$ uniformly at random.

Let κ denote the security parameter and $\mathsf{poly}(\kappa)$ denote the polynomial function. An algorithm is efficient if it runs in $\mathsf{poly}(\kappa)$-time. Let $\mathsf{negl} : \mathbb{N} \to \mathbb{R}$ denote the negligible function, i.e., for any polynomial $\mathsf{poly}(n)$, there exists an $n' \in \mathbb{N}$ s.t. for all $n > n'$, $\mathsf{negl}(n) < 1/\mathsf{poly}(n)$. For a primitive XX and a security notion YY, we denote by $\mathsf{Exp}^{\mathsf{YY}}_{\mathsf{XX},\mathcal{A}}(\kappa) \Rightarrow b$ a security experiment interacting with adversary $\mathcal{A}$ and returning a bit b. Furthermore, we denote by $\mathsf{Adv}^{\mathsf{YY}}_{\mathsf{XX},\mathcal{A}}(\kappa)$ the advantage of $\mathcal{A}$ in $\mathsf{Exp}^{\mathsf{YY}}_{\mathsf{XX},\mathcal{A}}(\kappa)$, and define $\mathsf{Adv}^{\mathsf{YY}}_{\mathsf{XX}}(\kappa) := \max_{\mathrm{PPT}\mathcal{A}} \mathsf{Adv}^{\mathsf{YY}}_{\mathsf{XX},\mathcal{A}}(\kappa)$.

Let X and Y be two random variables over support $\mathcal{S}$, then the statistical distance between X and Y is defined by $\mathsf{SD}(X, Y) = 1/2 \cdot \sum_{s\in\mathcal{S}} |\Pr[X = s] - \Pr[Y = s]|$. We say that X and Y are statistically indistinguishable and denote it by $X \approx_s Y$ if $\mathsf{SD}(X, Y) \le \mathsf{negl}(\kappa)$. If $\mathsf{SD}(X, Y) = 0$, then X and Y has the same distribution and we denote it by $X \equiv Y$.

Definition 1 (Average min-entropy [15]). *Let X and Y be two random variables. The min-entropy of X is defined as $\mathsf{H}_\infty(X) := -\log(\max_x \Pr[X = x])$. The average min-entropy of X given Y is defined as $\tilde{\mathsf{H}}_\infty(X \mid Y) := -\log[\mathbb{E}_{y\leftarrow Y} (\max_x \Pr[X = x \mid Y = y])]$.*

Lemma 1 ([15]). *Let X, Y be two random variables and Y has at most 2^ℓ possible values, then $\tilde{\mathsf{H}}_\infty(X|Y) \ge \mathsf{H}_\infty(X) - \ell$.*

2.1 Lattice Background

Let k, n, m, q be positive integers. Given n ($n \leq m$) linearly independent basis vectors $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathbb{R}^m$, construct a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ as $\mathbf{A}^\top := (\mathbf{a}_1, \ldots, \mathbf{a}_n)$. Define the m-dimensional lattice generated by $\mathbf{A}$ as $\Lambda(\mathbf{A}) := \{\mathbf{y} \in \mathbb{R}^m \mid \mathbf{y} = \mathbf{A}^\top \mathbf{x}, \mathbf{x} \in \mathbb{Z}^n\}$. We also define the following m-dimensional q-ary integer lattices: $\Lambda_q(\mathbf{A}) := \{\mathbf{y} \in \mathbb{Z}^m \mid \mathbf{y} = \mathbf{A}^\top \mathbf{x} \mod q, \mathbf{x} \in \mathbb{Z}_q^n\}$; $\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x} = 0^n \mod q\}$. For any vector $\mathbf{u} \in \mathbb{Z}_q^n$, define the coset (or shifted lattice) $\Lambda_q^{\mathbf{u}}(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x} = \mathbf{u} \mod q\}$.

Definition 2 (Discrete Gaussian distribution). *The Gaussian function with parameter s and center $\mathbf{c} \in \mathbb{R}^n$ is defined as $\rho_{s,\mathbf{c}} : \mathbb{R}^n \to \mathbb{R}$, $\rho_{s,\mathbf{c}}(\mathbf{x}) := \exp(-\pi\|\mathbf{x} - \mathbf{c}\|^2/s^2)$. For a countable set $\mathcal{S} \subset \mathbb{R}^n$, the discrete Gaussian distribution $D_{\mathcal{S},s,\mathbf{c}}$ parameterized with s and $\mathbf{c}$ is defined as $D_{\mathcal{S},s,\mathbf{c}}(\mathbf{x}) := \rho_{s,\mathbf{c}}(\mathbf{x})/\sum_{\mathbf{x}\in\mathcal{S}} \rho_{s,\mathbf{c}}(\mathbf{x})$ for $\mathbf{x} \in \mathcal{S}$ and $D_{\mathcal{S},s,\mathbf{c}}(\mathbf{x}) := 0$ for $\mathbf{x} \notin \mathcal{S}$. Usually, s is omitted when $s = 1$ and $\mathbf{c}$ is omitted if $\mathbf{c} = \mathbf{0}$.*

Lemma 2 (Randomness extraction [1,19]). *Let q, n, m be positive integers s.t. q is a prime and $m \geq 3n \log q$. Then:*

- *If $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \xleftarrow{\$} \{1, -1\}^m$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$, then $\mathsf{SD}((\mathbf{A}, \mathbf{A}\mathbf{s}), (\mathbf{A}, \mathbf{u})) \leq 2^{-n}$.*
- *If $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \leftarrow D_{\mathbb{Z}^m,\gamma}$ with Gaussian parameter $\gamma \geq \omega(\sqrt{\log m})$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$, then $\mathsf{SD}((\mathbf{A}, \mathbf{A}\mathbf{s}), (\mathbf{A}, \mathbf{u})) \leq 2^{-n}$.*

In this paper, we consider two types of lattice trapdoors. Let q, n, m be integers and define $w := n\lceil\log q\rceil$. Firstly, for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, we consider a non-singular square matrix $\mathbf{T}_\mathbf{A} \in \mathbb{Z}_q^{m\times m}$ of short integer vectors such that $\mathbf{A}\mathbf{T}_\mathbf{A} = 0^{n\times m} \mod q$, and call it a trapdoor of $\mathbf{A}$. We also consider the G-trapdoor (gadget trapdoor) proposed by Micciancio and Peikert [28]. A G-trapdoor for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is a matrix $\mathbf{R} \in \mathbb{Z}^{(m-w)\times w}$ s.t. $\mathbf{A} \cdot [-\mathbf{R}^\top | \mathbf{I}_w^\top]^\top = \mathbf{G}$, where $\mathbf{G} \in \mathbb{Z}_q^{n\times w}$ is the gadget matrix (see Definition 3). Clearly, if $\mathbf{A} = [\bar{\mathbf{A}} | \bar{\mathbf{A}}\mathbf{R} + \mathbf{G}]$, then $\mathbf{R}$ is the G-trapdoor for $\mathbf{A}$. Below we recall some definitions and lemmas related to afore-mentioned two trapdoors.

Lemma 3 (Trapdoor generation [2]). *Let q, n, m be positive parameters s.t. q is odd, $q \geq 3$ and $m = O(n \log q)$. There exists a PPT algorithm* $\mathsf{TrapGen}(1^n, 1^m, q)$ *that outputs matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m\times m}$ s.t. the distribution of $\mathbf{A}$ is statistically close to a uniform rank n matrix in $\mathbb{Z}_q^{n\times m}$ and matrix $\mathbf{T}_\mathbf{A}$ is a trapdoor for $\mathbf{A}$ satisfying $\mathbf{A}\mathbf{T}_\mathbf{A} = 0^{n\times m}$, $\|\tilde{\mathbf{T}}_\mathbf{A}\| \leq O(\sqrt{n \log q})$ and $\|\mathbf{T}_\mathbf{A}\| \leq O(n \log q)$ with all but 2^{-n} probability.*

Lemma 4 (Preimage sampling [19]). *Let q, n, m, γ be positive parameters s.t. $q \geq 2$. Let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ be a matrix with a trapdoor $\mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m\times m}$. Let $\gamma \geq \|\tilde{\mathbf{T}}_\mathbf{A}\| \cdot \omega(\sqrt{\log m})$. For any $\mathbf{u} \in \mathbb{Z}_q^n$, there exists a PPT algorithm* $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{u}, \gamma)$ *that outputs $\mathbf{s} \in \mathbb{Z}_q^m$ with distribution statistically close to $D_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\gamma}$.*

Lemma 5 (Trapdoor delegation [33]). *Let $q, n, m, m', \bar{m}$ be positive parameters and $\bar{m} = m + m'$. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{A}' \in \mathbb{Z}_q^{n \times m'}$ be matrices and $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m \times m}$ be a trapdoor for $\mathbf{A}$. There exists a deterministic polynomial-time algorithm $\mathsf{TrapDel}([\mathbf{A}|\mathbf{A}'], \mathbf{T}_{\mathbf{A}})$ that outputs a trapdoor $\mathbf{T}_{\mathbf{A}|\mathbf{A}'} \in \mathbb{Z}_q^{\bar{m} \times \bar{m}}$ for the matrix $[\mathbf{A}|\mathbf{A}']$. Besides, it holds that $\|\tilde{\mathbf{T}}_{\mathbf{A}|\mathbf{A}'}\| = \|\tilde{\mathbf{T}}_{\mathbf{A}}\|$.*

Definition 3 (Gadget matrix [28]). *For any integer modulus q, the gadget vector over $\mathbb{Z}_q$ is defined as $\mathbf{g}^\top := (1, 2, 4, \ldots, 2^{\lceil \log q \rceil - 1}) \in \mathbb{Z}_q^{1 \times \lceil \log q \rceil}$. Let $w := n\lceil \log q \rceil$, the gadget matrix $\mathbf{G}$ with full row rank is defined as:*

$$\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^\top = \begin{pmatrix} \mathbf{g}^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{g}^\top & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{g}^\top \end{pmatrix} \in \mathbb{Z}_q^{n \times w}.$$

Lemma 6 (G-to-Basis [28]). *Let n, m, q be positive integers and define $w := n\lceil \log q \rceil$. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be a matrix with a G-trapdoor $\mathbf{R} \in \mathbb{Z}^{(m-w) \times w}$. There exists a PPT algorithm $\mathsf{GtoBasis}(\mathbf{R})$ that returns a trapdoor $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m \times m}$ of $\mathbf{A}$. Moreover, the trapdoor $\mathbf{T}_{\mathbf{A}}$ satisfies $\|\widetilde{\mathbf{T}}_{\mathbf{A}}\| \le \sqrt{5}(s_1(\mathbf{R}) + 1)$.*

We recall in Lemma 7 the results of homomorphic evaluations established by a sequence of works [6–8,20] . Lemma 8 provides two statistically indistinguishable methods to generate $(\mathbf{A}, \mathbf{h}, \mathbf{e})$ s.t. $\mathbf{h} = \mathbf{A}\mathbf{e}$, where $\mathbf{h}$ follows the uniform distribution and $\mathbf{e}$ is short.

Lemma 7 (Homomorphic evaluation [6–8,20]). *Let q, n, m, ℓ and k be positive integers and define $w := n\lceil \log q \rceil$. Let $\mathbf{G} \in \mathbb{Z}_q^{n \times w}$ be the gadget matrix. Given a NAND boolean circuit $C : \{0,1\}^\ell \to \{0,1\}^k$ with circuit depth d, vector $\mathbf{x} = (x_1, \ldots, x_\ell)^\top \in \{0,1\}^\ell$, and matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $(\mathbf{A}_i \in \mathbb{Z}_q^{n \times w})_{i \in [\ell]}$ and $(\mathbf{R}_i \in \{\pm 1\}^{m \times w})_{i \in [\ell]}$, there exist two efficient deterministic algorithms.*

- *Algorithm $\mathsf{Eval}_{pub}(C, \mathbf{A}, (\mathbf{A}_i)_{i \in [\ell]})$ takes as inputs the circuit C and matrices $\mathbf{A}$, $(\mathbf{A}_i)_{i \in [\ell]}$, and outputs a matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times kw}$.*
- *Algorithm $\mathsf{Eval}_{prv}(C, \mathbf{A}, \mathbf{x}, (\mathbf{R}_i)_{i \in [\ell]})$ takes as inputs the circuit C, matrix $\mathbf{A}$, vector $\mathbf{x}$ and matrices $(\mathbf{R}_i)_{i \in [\ell]}$, and outputs a matrix $\mathbf{R}_C \in \mathbb{Z}^{m \times kw}$.*

Homomorphism. *If $\mathbf{A}_i = \mathbf{A}\mathbf{R}_i + x_i \cdot \mathbf{G} \in \mathbb{Z}_q^{n \times w}$ for all $i \in [\ell]$, $\mathbf{A}_C \leftarrow \mathsf{Eval}_{pub}(C, \mathbf{A}, (\mathbf{A}_i)_{i \in [\ell]})$ and $\mathbf{R}_C \leftarrow \mathsf{Eval}_{prv}(C, \mathbf{A}, \mathbf{x}, (\mathbf{R}_i)_{i \in [\ell]})$, then we have $\mathbf{A}_C = \mathbf{A}\mathbf{R}_C + C(\mathbf{x}) \otimes \mathbf{G}$, where $s_1(\mathbf{R}_C) \le O(4^d \cdot m^{\frac{3}{2}})$. Particularly, when C is in the circuit class NC^1, i.e., C is of depth $d = c \log \ell$ for some constant c, we have $s_1(\mathbf{R}_C) \le O(\ell^{2c} \cdot m^{\frac{3}{2}})$.*

Lemma 8 ([19]). *Let n, m, q be integers and $\gamma > 2\sqrt{n \log q}$, then for all but negligible probability over $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, it holds that*

$$\begin{aligned} &\{(\mathbf{A}, \mathbf{h}, \mathbf{e}) \mid \mathbf{h} \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{h}, \gamma)\} \approx_s \\ &\qquad\qquad\qquad\qquad \{(\mathbf{A}, \mathbf{h}, \mathbf{e}) \mid \mathbf{e} \leftarrow D_{\mathbb{Z}^m, \gamma}, \mathbf{h} := \mathbf{A}\mathbf{e}\}. \end{aligned}$$

2.2 Computational Assumption

Definition 4 (The SIS assumption). *Let q, n, m be positive integers and β be a positive real. The (homogeneous) short integer solution (SIS) assumption* $\mathsf{SIS}_{n,q,\beta,m}$ *states that for any PPT adversary $\mathcal{A}$, its advantage satisfies:*

$$\mathsf{Adv}^{\mathsf{SIS}}_{[n,q,\beta,m],\mathcal{A}}(\kappa) := \Pr\left[\mathcal{A}(\mathbf{A}) \to \mathbf{e} : \mathbf{A}\mathbf{e} = 0^n \wedge \mathbf{e} \neq 0^m \wedge \|\mathbf{e}\| \leq \beta\right] \leq \mathsf{negl}(\kappa),$$

where $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n \times m}$ *and* $0^n = (0, \ldots, 0)^\top \in \mathbb{Z}_q^n$.

Lemma 9 (The hardness of SIS [19,29,30]). *For any $m = \mathsf{poly}(n)$ and any sufficiently large $q \geq \beta \cdot \mathsf{poly}(n)$, solving $\mathsf{SIS}_{n,q,\beta,m}$ with non-negligible probability is at least as hard as solving the decisional approximate shortest vector problem GapSVP_γ and the approximate shortest independent vector problem SIVP_γ in the worst case with overwhelming probability, for some $\gamma = \beta \cdot \mathsf{poly}(n)$.*

Since GapSVP and SIVP are well-studied worst-case hard problems on lattices, the reduction from GapSVP and SIVP to SIS in Lemma 9 makes the SIS assumption a widely-accepted post-quantum assumption.

2.3 Pseudorandom Function

Definition 5 (Pseudorandom function family [21]). *A pseudorandom function family $\mathsf{PRF} := \{F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}\}$ is equipped with two polynomial time algorithms $(\mathsf{Setup}, \mathsf{PRF})$ defined below.*

- $\mathsf{Setup}(1^\kappa)$ *takes as input the security parameter $\kappa \in \mathbb{N}$ and outputs a public parameter pp.*
- $\mathsf{PRF}(pp, k, x)$ *takes as inputs the public parameter pp, key $k \in \mathcal{K}$ and message $x \in \mathcal{X}$, and outputs $y \in \mathcal{Y}$. For simplicity, we will omit pp and just write it as $\mathsf{PRF}(k, x)$ when the context is clear.*

Pseudorandomness. *Let $\mathsf{RF} : \mathcal{X} \to \mathcal{Y}$ be a truly random function. For any PPT adversary $\mathcal{A}$, its advantage satisfies $\mathsf{Adv}^{pse}_{\mathsf{PRF},\mathcal{A}}(\kappa) := \left|\Pr[\mathcal{A}^{\mathcal{O}_{\mathsf{PRF}}(\cdot)}(pp) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\mathsf{RF}}(\cdot)}(pp) \Rightarrow 1]\right| \leq \mathsf{negl}(\kappa)$, where $pp \leftarrow \mathsf{Setup}(1^\kappa)$, $k \overset{\$}{\leftarrow} \mathcal{K}$, oracle $\mathcal{O}_{\mathsf{PRF}}(x)$ returns $\mathsf{PRF}(pp, k, x)$ and oracle $\mathcal{O}_{\mathsf{RF}}(x)$ returns $\mathsf{RF}(x)$.*

3 Tagged Chameleon Hash

In this section, we propose a new primitive named tagged chameleon hash (tCH), which is characterized by four algorithms, the setup algorithm, hash algorithm, adapt algorithm and check algorithm. The setup algorithm generates public parameters *pp* along with a trapdoor *td*. The hash algorithm is a randomized one used for evaluating the hash value of a message m w.r.t. a tag τ and it outputs a randomness r serving as the witness of hashing relation among h, m and τ. For simplicity, we just call h the hash value of (τ, m, r). Given (τ, h, m, r, m'), where

```
$\mathsf{Exp}^{fcr}_{\mathsf{tCH},\mathcal{A}}(\kappa)$:
    $(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$, $\mathcal{Q}_{\mathsf{Adapt}} := \emptyset$
    $(\tau^*, h^*, m^*, r^*, m'^*, r'^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Adapt}}(\cdot,\cdot,\cdot,\cdot,\cdot)}(pp)$
    If $\mathsf{Check}(\tau^*, h^*, m^*, r^*) = \mathsf{Check}(\tau^*, h^*, m'^*, r'^*) = 1$
        $\wedge m^* \neq m'^* \wedge (\tau^*, h^*, m^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$, return 1
    Otherwise, return 0

$\mathcal{O}_{\mathsf{Adapt}}(\tau, h, m, r, m')$:   // $m'$ is the adapted message
    If $\mathsf{Check}(pp, \tau, h, m, r) = 0$, return $\perp$
    $r' \leftarrow \mathsf{Adapt}(td, \tau, h, m, r, m')$
    If $r' \neq \perp$, $\mathcal{Q}_{\mathsf{Adapt}} := \mathcal{Q}_{\mathsf{Adapt}} \cup \{(\tau, h, m), (\tau, h, m')\}$
    Return $r'$
```

```
$\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa)$:
    $(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$, $\mathcal{Q}_{\mathsf{Adapt}} := \emptyset$
    $(\tau^*, h^*, m^*, r^*, m'^*, r'^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Adapt}}(\cdot,\cdot,\cdot,\cdot,\cdot)}(pp)$
    If $\mathsf{Check}(\tau^*, h^*, m^*, r^*) = \mathsf{Check}(\tau^*, h^*, m'^*, r'^*) = 1$
        $\wedge m^* \neq m'^* \wedge \mathsf{Valid}(\tau^*, h^*, m^*, m'^*) = 1$, return 1
    Otherwise, return 0

$\mathsf{Valid}(\tau^*, h^*, m^*, m'^*)$:
    If $(\tau^*, \cdot, m^*) \notin \mathcal{Q}_{\mathsf{Adapt}} \wedge (\tau^*, h^*, m'^*) \in \mathcal{Q}_{\mathsf{Adapt}}$, return 1
    If $(\tau^*, \cdot, m^*) \notin \mathcal{Q}_{\mathsf{Adapt}} \wedge (\tau^*, \cdot, m'^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$, return 1
    Otherwise, return 0

$\mathcal{O}_{\mathsf{Adapt}}(\tau, h, m, r, m')$:  // $m'$ is the adapted message
    If $\mathsf{Check}(pp, \tau, h, m, r) = 0$, return $\perp$
    If $\exists\ (\tau, h'', m) \in \mathcal{Q}_{\mathsf{Adapt}} \wedge h'' \neq h$, return $\perp$
        // $\tau$ is uniquely bound with hash value $h$
    If $\exists\ (\tau, \cdot, m') \in \mathcal{Q}_{\mathsf{Adapt}}$, return $\perp$
        // $m'$ is a fresh message w.r.t $\tau$
    $r' \leftarrow \mathsf{Adapt}(td, \tau, h, m, r, m')$
    If $r' \neq \perp$, $\mathcal{Q}_{\mathsf{Adapt}} := \mathcal{Q}_{\mathsf{Adapt}} \cup \{(\tau, h, m), (\tau, h, m')\}$
    Return $r'$
```

Fig. 3. Experiments $\mathsf{Exp}^{fcr}_{\mathsf{tCH},\mathcal{A}}$ and $\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}$ defining f-CR and r-CR for tCH.

h is the hash value of (τ, m, r) and m' is a new message, the adapt algorithm uses the trapdoor td to find a randomness r' so that (τ, m, r) and (τ, m', r') collide at the same hash value h. The check algorithm is used to decide whether h is the hash value of a tag-message-randomness triple (τ, m, r). For a tCH, we define the statistical indistinguishability, and provide two security notions: one is the full collision resistance (f-CR) defined following [13], and the other is a weaker one named the restricted collision resistance (r-CR). We show that when tCH works in the one-time tag mode, f-CR and r-CR are equivalent.

Definition 6 (Tagged chameleon hash). *Let $\mathcal{M}$ be the message space and $\mathcal{T}$ be the tag space. A tagged chameleon hash (tCH)* tCH *consists of four polynomial time algorithms* $\mathsf{tCH} = (\mathsf{Setup}, \mathsf{Hash}, \mathsf{Adapt}, \mathsf{Check})$ *defined as follows.*

- $\mathsf{Setup}(1^\kappa)$ *takes as input the security parameter $\kappa \in \mathbb{N}$ and returns a public parameter pp and a trapdoor td.*
- $\mathsf{Hash}(pp, \tau, m)$ *takes as inputs the public parameter pp, a tag $\tau \in \mathcal{T}$ and a message $m \in \mathcal{M}$, and returns a hash value h and a randomness r.*
- $\mathsf{Adapt}(td, \tau, h, m, r, m')$ *takes as inputs the trapdoor td, a tag $\tau \in \mathcal{T}$, a hash value h, a message $m \in \mathcal{M}$, a randomness r and a fresh target message $m' \in \mathcal{M}$, and returns a new randomness r'.*
- $\mathsf{Check}(pp, \tau, h, m, r)$ *takes as inputs the public parameter pp, a tag $\tau \in \mathcal{T}$, a hash value h, a message $m \in \mathcal{M}$ and a randomness r, and returns a decision bit $b \in \{0, 1\}$.*

For expression simplicity, we will sometimes omit the "pp" part in the inputs of Hash *and* Check*, and just write them as* $\mathsf{Hash}(\tau, m)$ *and* $\mathsf{Check}(\tau, h, m, r)$ *respectively when the context is clear.*

- ***Correctness.*** *For all tag* $\tau \in \mathcal{T}$ *and messages* $m, m' \in \mathcal{M}$*, for all* $(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$*,* $(h, r) \leftarrow \mathsf{Hash}(pp, \tau, m)$ *and* $r' \leftarrow \mathsf{Adapt}(td, \tau, h, m, r, m')$*, we have*

$$\Pr\left[\mathsf{Check}(pp, \tau, h, m, r) = \mathsf{Check}(pp, \tau, h, m', r') = 1\right] \geq 1 - \mathsf{negl}(\kappa).$$

- ***Statistical Indistinguishability.*** *For all tag* $\tau \in \mathcal{T}$ *and messages* $m, m' \in \mathcal{M}$*, and for* $(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$*, it holds that*

$$\left\{(h, r) \mid (h, r) \leftarrow \mathsf{Hash}(pp, \tau, m)\right\} \\ \approx_s \left\{(h, r) \mid (h, r') \leftarrow \mathsf{Hash}(pp, \tau, m'), r \leftarrow \mathsf{Adapt}(td, \tau, h, m', r', m)\right\}.$$

- ***Full collision resistance (f-CR).*** *For any PPT adversary* $\mathcal{A}$*, its advantage satisfies* $\mathsf{Adv}^{fcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) := \Pr[\mathsf{Exp}^{fcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) \Rightarrow 1] \leq \mathsf{negl}(\kappa)$*, where the experiment* $\mathsf{Exp}^{fcr}_{\mathsf{tCH},\mathcal{A}}$ *is described in Fig. 3 (left).*
- ***Restricted collision resistance (r-CR).*** *For any PPT adversary* $\mathcal{A}$*, its advantage satisfies* $\mathsf{Adv}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) := \Pr[\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) \Rightarrow 1] \leq \mathsf{negl}(\kappa)$*, where the experiment* $\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}$ *is described in Fig. 3 (right).*

One-time tag mode for tCH. In this paper, we consider a special working mode for tagged chameleon hash, where every invocation of hash evaluation takes as input a distinct tag. The special working mode is named *one-time tag mode.* Note that in a tCH-based redactable blockchain, tCH just works in this mode when setting the unique identifier of the block (like the timestamp, hash value of its previous block, or its position in the chain) as its tag.

Definition 7 (One-time tag mode). *A tCH scheme* $\mathsf{tCH} = (\mathsf{Setup}, \mathsf{Hash}, \mathsf{Adapt}, \mathsf{Check})$ *works in the one-time tag mode if any* $Q = \mathsf{poly}(\kappa)$ *invocations of* $\mathsf{Hash}(pp, \tau_i, m_i)$ *with* $i \in [Q]$*, we have* $\tau_k \neq \tau_j$ *for any* $k, j \in [Q]$ *and* $k \neq j$*.*

Next we show that f-CR is equivalent to r-CR in the one-time tag mode. It is easy to see that f-CR implies r-CR. As for the other direction, we show in Theorem 1 that r-CR implies f-CR when a tCH works in the one-time tag mode.

Theorem 1. *If a tagged chameleon hash* tCH *satisfies the restricted collision resistance, then it also satisfies the full collision resistance when it is used in the one-time tag mode. More precisely, for any PPT adversary* $\mathcal{A}$*, it holds that*

$$\mathsf{Adv}^{fcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) \leq \mathsf{Adv}^{rcr}_{\mathsf{tCH}}(\kappa).$$

A high-level idea of proof for Theorem 1 has been described in the introduction, and see our full version [26] for the detailed proof.

Remark. Ateniese and de Medeiros considered a chameleon hash with labels (abbrv., labeled CH) in [4]. Our tCH and labeled CH both take an extra tag/label as input, but they have different syntax, security notions and applications.

- Syntax difference. Labeled CH involves an additional algorithm lForge, which generates (m'', r'') given a collision pair (τ, h, m, r, m', r') s.t. $h = \mathsf{Hash}(\tau, m''; r'') = \mathsf{Hash}(\tau, m'; r') = \mathsf{Hash}(\tau, m; r)$. In other words, anyone who obtains a collision for (τ, h) can freely generate a new collision for the same (τ, h). In contrast, our tCH can find a collision only with a secret trapdoor.

- Security difference. Labeled CH requires a weaker security named the key-exposure freeness, which assures the infeasibility of finding a collision ($\tau^*, h^*, m^*, r^*, m'^*, r'^*$) when no collision for the specific τ^* has been revealed. In contrast, our CR/fCR allows the adversary to see polynomial collisions for the same target tag.
- Application difference. Labeled CH is usually used to construct chameleon signature and it is not secure enough to be used in a redactable blockchain. Note that adversaries in a redactable blockchain may obtain multiple collisions towards one (τ, h). With labeled CH, any one is able to create collisions for (τ, h) using algorithm lForge, and then redactable blockchain becomes insecure. In contrast, our tCH with f-CR security (or r-CR security in one-time tag mode) serves for the security requirement from a redactable blockchain.

4 Lattice-Based Tagged Chameleon Hash

In this section, we propose two tCH constructions satisfying the restricted collision resistance based on the SIS assumption. In Subsect. 4.1, we propose a tCH construction in the standard model. In Subsect. 4.2, we provide another tCH scheme with tight security in the random oracle model.

4.1 tCH in the Standard Model

In this subsection, we propose a tCH construction from lattices, namely tCH, in the standard model.

First we introduce the building blocks and some notations used in our tCH construction. Let n, q, m be positive integers, and define $w := n\lceil \log q \rceil$.

- A pseudorandom function $\mathsf{PRF} = (\mathsf{PRF.Setup}, \mathsf{PRF})$ with key space $\{0,1\}^k$, input space $\{0,1\}^x$ and output space $\{0,1\}^y$.
- Define a circuit $C[\mathbf{x}, \mathbf{y}] : \{0,1\}^k \rightarrow \{0,1\}$ w.r.t. PRF as below, where $\mathbf{x} \in \{0,1\}^x$ and $\mathbf{y} \in \{0,1\}^y$ are hard-wired to the circuit.

$$C[\mathbf{x}, \mathbf{y}](\mathbf{k}) = \begin{cases} 1 & \text{if } \mathsf{PRF}(\mathbf{k}, \mathbf{x}) = \mathbf{y}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Our tCH construction tCH is given in Fig. 4.

Parameter setting. Parameters of our tCH construction include the security parameter κ, the dimension parameters k, x, y, t, h, the SIS parameters n, m, q, β and the Gaussian parameter γ. Define $w := n\lceil \log q \rceil$. The afore-mentioned parameters are required to satisfy the following conditions simultaneously.

- Let $k, x, y, t, h = \mathsf{poly}(\kappa)$ be positive integers and $x = t + h + k$.
- Let n, q, m, β be positive parameters, $n, m, \beta, q = \mathsf{poly}(\kappa)$ and $\beta \cdot \mathsf{poly}(n) \le q$ so that the SIS problem is hard according to Lemma 9.
- Let $\gamma \ge O(\kappa^c) \cdot \omega(\sqrt{m+w})$ with some constant c and $\gamma \ge O(n \log q) \cdot \omega(\sqrt{m+w})$ so that Lemma 4 can be applied.

```
$(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$.
  1. Generate $pp_{\mathsf{prf}} \leftarrow \mathsf{PRF.Setup}(1^\kappa)$.
  2. Generate $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ with $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$.
  3. For $i \in [k]$, sample $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times w}$. For $i \in [h]$, sample $\hat{\mathbf{A}}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times w}$.
  4. Return $pp := (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]}, \{\hat{\mathbf{A}}_i\}_{i \in [h]})$ and $td := (pp, \mathbf{T_A})$.

$(\mathbf{h}, r) \leftarrow \mathsf{Hash}(pp, \tau \in \{0,1\}^t, \mathbf{m} \in \{0,1\}^h)$.
  1. Parse $pp = (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]}, \{\hat{\mathbf{A}}_i\}_{i \in [h]})$ and $\mathbf{m} = (m_1, \ldots, m_h)$.
  2. Sample $\mathbf{z} \xleftarrow{\$} \{0,1\}^\kappa$ and set $\mathbf{x} := \tau \| \mathbf{m} \| \mathbf{z} \in \{0,1\}^x$.
  3. Sample $\mathbf{y} \xleftarrow{\$} \{0,1\}^y$ and construct the circuit $C[\mathbf{x}, \mathbf{y}]$ as defined by (2).
  4. Compute $\mathbf{C}_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}, \mathbf{y}](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]})$ and $\mathbf{B}_{\mathsf{prf}} := \sum_{i \in [h]} m_i \hat{\mathbf{A}}_i$.
  5. Compute $\mathbf{A}_{\mathsf{prf}} := \mathbf{C}_{\mathsf{prf}} + \mathbf{B}_{\mathsf{prf}}$.
  6. Sample $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2) \leftarrow D_{\mathbb{Z}^{m+w}, \gamma}$ with $\mathbf{e}_1 \in \mathbb{Z}_q^m$ and $\mathbf{e}_2 \in \mathbb{Z}_q^w$ s.t. $\mathbf{e}_2 \neq 0^w$.
  7. Return $\mathbf{h} := [\mathbf{A} | \mathbf{A}_{\mathsf{prf}}] \cdot \mathbf{e} \in \mathbb{Z}_q^n$ and $r := (\mathbf{z}, \mathbf{y}, \mathbf{e})$.

$r' \leftarrow \mathsf{Adapt}(td, \tau \in \{0,1\}^t, \mathbf{h}, \mathbf{m} \in \{0,1\}^h, r, \mathbf{m}' \in \{0,1\}^h)$.
  1. Parse $td = (pp, \mathbf{T_A})$, $pp = (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]}, \{\hat{\mathbf{A}}_i\}_{i \in [h]})$, $\mathbf{m}' = (m'_1, \ldots, m'_h)$.
  2. If $\mathsf{Check}(pp, \tau, \mathbf{h}, \mathbf{m}, r) = 0$, return $\perp$. Otherwise, continue.
  3. Sample $\mathbf{z}' \xleftarrow{\$} \{0,1\}^\kappa$ and $\mathbf{y}' \xleftarrow{\$} \{0,1\}^y$.
  4. Set $\mathbf{x}' := \tau \| \mathbf{m}' \| \mathbf{z}' \in \{0,1\}^x$ and construct $C[\mathbf{x}', \mathbf{y}']$ as defined by (2).
  5. Compute $\mathbf{C}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]})$ and $\mathbf{B}'_{\mathsf{prf}} := \sum_{i \in [h]} m'_i \hat{\mathbf{A}}_i$.
  6. Compute $\mathbf{A}'_{\mathsf{prf}} := \mathbf{C}'_{\mathsf{prf}} + \mathbf{B}'_{\mathsf{prf}}$. Delegate $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}} \leftarrow \mathsf{TrapDel}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T_A})$.
  7. Sample $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2) \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}, \mathbf{h}, \gamma)$ with $\mathbf{e}'_1 \in \mathbb{Z}_q^m$ and
     $\mathbf{e}'_2 \in \mathbb{Z}_q^w$ s.t. $\mathbf{e}'_2 \neq 0^w$.
  8. Return $r' := (\mathbf{z}', \mathbf{y}', \mathbf{e}')$.

$0/1 \leftarrow \mathsf{Check}(pp, \tau \in \{0,1\}^t, \mathbf{h}, \mathbf{m} \in \{0,1\}^h, r)$.
  1. Parse $pp = (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]}, \{\hat{\mathbf{A}}_i\}_{i \in [h]})$, $\mathbf{m} = (m_1, \ldots, m_h)$ and $r = (\mathbf{z}, \mathbf{y}, \mathbf{e})$.
  2. Set $\mathbf{x} := \tau \| \mathbf{m} \| \mathbf{z} \in \{0,1\}^x$ and construct $C[\mathbf{x}, \mathbf{y}]$ as defined by (2).
  3. Compute $\mathbf{C}_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}, \mathbf{y}](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]})$ and $\mathbf{B}_{\mathsf{prf}} := \sum_{i \in [h]} m_i \hat{\mathbf{A}}_i$.
  4. Compute $\mathbf{A}_{\mathsf{prf}} := \mathbf{C}_{\mathsf{prf}} + \mathbf{B}_{\mathsf{prf}}$.
  5. Parse $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2)$ with $\mathbf{e}_1 \in \mathbb{Z}_q^m$ and $\mathbf{e}_2 \in \mathbb{Z}_q^w$. If $\mathbf{h} = [\mathbf{A}|\mathbf{A}_{\mathsf{prf}}] \cdot \mathbf{e}$, $\|\mathbf{e}\| \leq \gamma\sqrt{m+w}$
     and $\mathbf{e}_2 \neq 0^w$, return 1; otherwise, return 0.
```

Fig. 4. Tagged chameleon hash tCH in the standard model.

- Let $m = O(n \log q)$ and $\gamma \cdot O(\kappa^c) \cdot \sqrt{m+w} \leq \beta$ with some constant c to serve for our security proof.

Theorem 2. *Let* PRF *be a pseudorandom function. Given parameters described above, construction* tCH *in Fig. 4 is a tagged chameleon hash if the* $\mathsf{SIS}_{n,q,\beta,m}$ *assumption holds. Furthermore, restricted collision resistance of* tCH *is tightly reduced to the SIS assumption and the pseudorandomness of* PRF*:*

$$\Pr[\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) \Rightarrow 1] \leq \mathsf{Adv}^{\mathsf{SIS}}_{[n,q,\beta,m]}(\kappa) + 2\mathsf{Adv}^{pse}_{\mathsf{PRF}}(\kappa) + 2^{-O(\kappa)}.$$

Correctness of tCH. It follows directly from Lemma 5 (trapdoor delegation), Lemma 4 (preimage sampling) and Lemma 7 (homomorphic evaluation), and we omit the proof of it here.

Proof of statistical indistinguishability for tCH. We prove that, given tag τ and messages $\mathbf{m}, \mathbf{m}'$, the distribution of $(\mathbf{h}, r)$ generated by Hash is statistically close to that generated by Hash-then-Adapt.

First consider the distribution of $(\mathbf{h}, r)$ generated by Hash, i.e., $(\mathbf{h}, r) \leftarrow \mathsf{Hash}(\tau, \mathbf{m})$. It follows the distribution D_{H} defined below:

$$D_{\mathsf{H}} := \left\{ (\mathbf{h}, r = (\mathbf{z}, \mathbf{y}, \mathbf{e})) \;\middle|\; \begin{array}{l} \mathbf{z} \xleftarrow{\$} \{0,1\}^{\kappa}, \mathbf{y} \xleftarrow{\$} \{0,1\}^{y}, \\ \mathbf{e} \leftarrow D_{\mathbb{Z}^{m+w},\gamma}, \mathbf{h} = [\mathbf{A}|\mathbf{A}_{\mathsf{prf}}] \cdot \mathbf{e} \end{array} \right\},$$

where $\mathbf{A}_{\mathsf{prf}}$ is deterministically computed from $\tau, \mathbf{m}$, the public parameters $(\mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]}, \{\hat{\mathbf{A}}_i\}_{i\in[h]})$ and uniformly chosen $\mathbf{z}, \mathbf{y}$ (see algorithm Hash in Fig. 4).

Next consider the distribution of $(\mathbf{h}, r)$ generated by Hash-then-Adapt, i.e., first $(\mathbf{h}, r' = (\mathbf{z}', \mathbf{y}', \mathbf{e}')) \leftarrow \mathsf{Hash}(\tau, \mathbf{m}')$ and then $r \leftarrow \mathsf{Adapt}(td, \tau, \mathbf{h}, \mathbf{m}', r', \mathbf{m})$. It follows the distribution $D_{\mathsf{H\&A}}$ defined below:

$$D_{\mathsf{H\&A}} := \left\{ (\mathbf{h}, r = (\mathbf{z}, \mathbf{y}, \mathbf{e})) \;\middle|\; \begin{array}{l} \mathbf{z}, \mathbf{z}' \xleftarrow{\$} \{0,1\}^{\kappa} \;,\; \mathbf{y}, \mathbf{y}' \xleftarrow{\$} \{0,1\}^{y} \;,\; \mathbf{e}' \leftarrow D_{\mathbb{Z}^{m+\omega},\gamma} \;, \\ \mathbf{h} := [\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}] \cdot \mathbf{e}', \mathbf{e} \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}_{\mathsf{prf}}}, \mathbf{h}, \gamma) \end{array} \right\}$$

where $\mathbf{A}_{\mathsf{prf}}$ is computed in the same way as above, and $\mathbf{A}'_{\mathsf{prf}}$ is generated similar to $\mathbf{A}_{\mathsf{prf}}$ but with $\mathbf{m}', \mathbf{z}'$ and $\mathbf{y}'$.

First we show that $\mathbf{h} := [\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}] \cdot \mathbf{e}'$ in $D_{\mathsf{H\&A}}$ is statistically close to the uniform distribution over $\mathbb{Z}_q^n$. Note that

$$\mathbf{h} := [\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}] \cdot \mathbf{e}' = \mathbf{A}\mathbf{e}'_1 + \mathbf{A}'_{\mathsf{prf}}\mathbf{e}'_2 \approx_s \mathbf{u}' + \mathbf{A}'_{\mathsf{prf}}\mathbf{e}'_2 \equiv \mathbf{u},$$

where $\mathbf{u}', \mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}' = (\mathbf{e}'_1 \| \mathbf{e}'_2) \leftarrow D_{\mathbb{Z}^{m+w},\sigma}$, $\mathbf{e}'_1 \in \mathbb{Z}_q^m$ and $\mathbf{e}'_2 \in \mathbb{Z}_q^w$. The "$\approx_s$" follows from Lemma 2 and "$\equiv$" follows from the uniformity of $\mathbf{u}'$. Therefore, it holds that $D_{\mathsf{H\&A}} \approx_s D'_{\mathsf{H\&A}}$, where

$$D'_{\mathsf{H\&A}} := \left\{ (\mathbf{h}, r = (\mathbf{z}, \mathbf{y}, \mathbf{e})) \;\middle|\; \begin{array}{l} \mathbf{z} \xleftarrow{\$} \{0,1\}^{\kappa}, \mathbf{y} \xleftarrow{\$} \{0,1\}^{y}, \mathbf{h} \xleftarrow{\$} \mathbb{Z}_q^n, \\ \mathbf{e} \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}_{\mathsf{prf}}}, \mathbf{h}, \gamma) \end{array} \right\}$$

Then according to Lemma 8, $D_{\mathsf{H}} \approx_s D'_{\mathsf{H\&A}}$. Therefore, $D_{\mathsf{H}} \approx_s D_{\mathsf{H\&A}}$ by triangle inequality and this proves the statistical indistinguishability of tCH. □

Proof of restricted collision resistance for tCH. We define a sequence of hybrid games $\mathsf{G}_0 \sim \mathsf{G}_4$, where G_0 is identical to $\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa)$ defined in Fig. 3. We show that G_i and G_{i-1} are indistinguishable for all $i \in [4]$, and in G_4, the adversary wins with negligible probability. The differences between adjacent games are highlighted in blue. Assume that $\mathcal{A}$ makes at most Q adaptation queries.

Game G_0. Game G_0 is identical to $\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa)$ defined by Fig. 3.

0. The challenger $\mathcal{C}$ initializes set $\mathcal{Q}_{\mathsf{Adapt}} := \emptyset$.
1. During the setup phase, the challenger $\mathcal{C}$ proceeds as follows.
   - Generate $pp_{\mathsf{prf}} \leftarrow \mathsf{PRF.Setup}(1^{\kappa})$.
   - Generate $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$.
   - Sample $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$ for $i \in [k]$. Sample $\hat{\mathbf{A}}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$ for $i \in [h]$.

   - $pp := (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]}, \{\hat{\mathbf{A}}_i\}_{i\in[h]})$, $td := (pp, \mathbf{T_A})$ and send pp to $\mathcal{A}$.
2. Upon an adaptation query $(\tau, \mathbf{h}, \mathbf{m}, r, \mathbf{m}')$ from $\mathcal{A}$, $\mathcal{C}$ proceeds as follows.
   - If $\exists\ (\tau, \mathbf{h}'', \mathbf{m}) \in \mathcal{Q}_{\mathsf{Adapt}} \wedge \mathbf{h}'' \neq \mathbf{h}$, or $\exists\ (\tau, \cdot, \mathbf{m}') \in \mathcal{Q}_{\mathsf{Adapt}}$, or $\mathsf{Check}(\tau, \mathbf{h}, \mathbf{m}, r) = 0$ holds, return $\perp$; otherwise, continue.
   - Sample $\mathbf{z}' \xleftarrow{\$} \{0,1\}^\kappa$ and set $\mathbf{x}' := \tau\|\mathbf{m}'\|\mathbf{z}'$.
   - Sample $\mathbf{y}' \xleftarrow{\$} \{0,1\}^y$ and construct $C[\mathbf{x}', \mathbf{y}']$ as defined by (2).
   - $\mathbf{C}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]})$ and $\mathbf{B}'_{\mathsf{prf}} := \sum_{i\in[h]} m'_i \hat{\mathbf{A}}_i$.
   - Set $\mathbf{A}'_{\mathsf{prf}} := \mathbf{C}'_{\mathsf{prf}} + \mathbf{B}'_{\mathsf{prf}}$. Delegate $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}} \leftarrow \mathsf{TrapDel}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T_A})$.
   - $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2) \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}, \mathbf{h}, \gamma)$ s.t. $\mathbf{e}'_2 \neq 0^w$.
   - Send $r' := (\mathbf{z}', \mathbf{y}', \mathbf{e}')$ to $\mathcal{A}$ and $\mathcal{Q}_{\mathsf{Adapt}} := \mathcal{Q}_{\mathsf{Adapt}} \cup \{(\tau, \mathbf{h}, \mathbf{m}), (\tau, \mathbf{h}, \mathbf{m}')\}$.
3. On receiving the forgery $(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*, \mathbf{m}'^*, r'^*)$, $\mathcal{C}$ makes the following checks, and returns 0 if any of them fails. Otherwise, $\mathcal{C}$ returns 1.
   - Check if $\mathsf{Check}(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*) = \mathsf{Check}(\tau^*, \mathbf{h}^*, \mathbf{m}'^*, r'^*) = 1$.
   - Check if $\mathbf{m}^* \neq \mathbf{m}'^*$ and $\mathsf{Valid}(\tau^*, \mathbf{h}^*, \mathbf{m}^*, \mathbf{m}'^*) = 1$.

By definition, we have $\Pr[\mathsf{G}_0 \Rightarrow 1] = \Pr[\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) \Rightarrow 1]$.

Game G_1. Game G_1 is similar to G_0 except for the generation of $\mathbf{y}'$ in the adaptation query phase. In G_0, $\mathbf{y}'$ is sampled uniformly at random for each adaptation query. In G_1, $\mathbf{y}'$ is computed by PRF, i.e., $\mathbf{y}' \leftarrow \mathsf{PRF}(\mathbf{k}, \mathbf{x}')$, where key $\mathbf{k} \xleftarrow{\$} \{0,1\}^k$ is sampled in the setup phase.

1'. During the setup phase, the challenger $\mathcal{C}$ proceeds as follows.

   - Generate $pp_{\mathsf{prf}} \leftarrow \mathsf{PRF.Setup}(1^\kappa)$ and sample $\mathbf{k} \xleftarrow{\$} \{0,1\}^k$.
   - Generate $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$.
   - Sample $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$ for $i \in [k]$. Sample $\hat{\mathbf{A}}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$ for $i \in [h]$.
   - $pp := (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]}, \{\hat{\mathbf{A}}_i\}_{i\in[h]})$, $td := (pp, \mathbf{T_A})$ and send pp to $\mathcal{A}$.

2'. Upon an adaptation query $(\tau, \mathbf{h}, \mathbf{m}, r, \mathbf{m}')$ from $\mathcal{A}$, $\mathcal{C}$ proceeds as follows.

   - If $\exists\ (\tau, \mathbf{h}'', \mathbf{m}) \in \mathcal{Q}_{\mathsf{Adapt}} \wedge \mathbf{h}'' \neq \mathbf{h}$, or $\exists\ (\tau, \cdot, \mathbf{m}') \in \mathcal{Q}_{\mathsf{Adapt}}$, or $\mathsf{Check}(\tau, \mathbf{h}, \mathbf{m}, r) = 0$ holds, return $\perp$; otherwise, continue.
   - Sample $\mathbf{z}' \xleftarrow{\$} \{0,1\}^\kappa$ and set $\mathbf{x}' := \tau\|\mathbf{m}'\|\mathbf{z}'$.
   - Compute $\mathbf{y}' \leftarrow \mathsf{PRF}(\mathbf{k}, \mathbf{x}')$ and construct $C[\mathbf{x}', \mathbf{y}']$ as defined by (2).
   - $\mathbf{C}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]})$ and $\mathbf{B}'_{\mathsf{prf}} := \sum_{i\in[h]} m'_i \hat{\mathbf{A}}_i$.
   - Set $\mathbf{A}'_{\mathsf{prf}} := \mathbf{C}'_{\mathsf{prf}} + \mathbf{B}'_{\mathsf{prf}}$. Delegate $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}} \leftarrow \mathsf{TrapDel}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T_A})$.
   - $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2) \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}, \mathbf{h}, \gamma)$ s.t. $\mathbf{e}'_2 \neq 0^w$.
   - Send $r' := (\mathbf{z}', \mathbf{y}', \mathbf{e}')$ to $\mathcal{A}$ and $\mathcal{Q}_{\mathsf{Adapt}} := \mathcal{Q}_{\mathsf{Adapt}} \cup \{(\tau, \mathbf{h}, \mathbf{m}), (\tau, \mathbf{h}, \mathbf{m}')\}$.

Lemma 10. *Games G_1 and G_2 are computationally indistinguishable due to the pseudorandomness of* PRF*, i.e.,* $|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_2 \Rightarrow 1]| \leq \mathsf{Adv}^{pse}_{\mathsf{PRF}}(\kappa) + 2^{-O(\kappa)}$.

Proof of Lemma 10 (sketch). Note that $\mathbf{z}' \xleftarrow{\$} \{0,1\}^\kappa$ is sampled uniformly at random for each adaptation query, then all $\mathbf{x}' = \tau\|\mathbf{m}'\|\mathbf{z}'$ constructed for the adaptation queries are different from each other with probability $1 - 2^{-O(\kappa)}$. Now according to the pseudorandomness of PRF, we know that the distribution of

$\mathbf{y}' \xleftarrow{\$} \{0,1\}^y$ is computationally indistinguishable from that of $\mathbf{y}' \leftarrow \mathsf{PRF}(\mathbf{k}, \mathbf{x}')$ and this proves Lemma 10. $\square$

Game G_2. Game G_2 is similar to G_1 except for the generations of $\{\mathbf{A}_i\}_{i\in[k]}$ and $\{\hat{\mathbf{A}}_i\}_{i\in[h]}$ in the setup phase, and the computations of $\mathbf{A}'_{\mathsf{prf}}$ in the adaptation query phase. In G_1, $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$ and $\hat{\mathbf{A}}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$ are sampled uniformly at random in the setup phase, and $\mathbf{A}'_{\mathsf{prf}}$ is computed by Eval_{pub} from $\mathbf{A}_i$'s and $\hat{\mathbf{A}}_i$'s when answering each adaptation query. In G_2, $\mathbf{A}_i$ and $\hat{\mathbf{A}}_i$ are computed by $\mathbf{A}_i := \mathbf{A}\mathbf{R}_i + k_i\mathbf{G}$ and $\hat{\mathbf{A}}_i := \mathbf{A}\hat{\mathbf{R}}_i$ with $\mathbf{R}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and $\hat{\mathbf{R}}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ in the setup phase, and $\mathbf{A}'_{\mathsf{prf}} := \mathbf{A}\mathbf{R}'_{\mathsf{prf}} + \mathbf{G}$ when answering each adaptation query with $\mathbf{R}'_{\mathsf{prf}}$ computed by Eval_{prv} from $\mathbf{R}_i$'s and $\hat{\mathbf{R}}_i$'s.

1″. During the setup phase, the challenger $\mathcal{C}$ proceeds as follows.

- Generate $pp_{\mathsf{prf}} \leftarrow \mathsf{PRF.Setup}(1^\kappa)$ and sample $\mathbf{k} \xleftarrow{\$} \{0,1\}^k$.
- Generate $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$.
- Sample $\mathbf{R}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and set $\mathbf{A}_i := \mathbf{A}\mathbf{R}_i + k_i\mathbf{G}$ for $i \in [k]$. Sample $\hat{\mathbf{R}}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and set $\hat{\mathbf{A}}_i := \mathbf{A}\hat{\mathbf{R}}_i$ for $i \in [h]$.
- $pp := (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]}, \{\hat{\mathbf{A}}_i\}_{i\in[h]})$, $td := (pp, \mathbf{T}_\mathbf{A})$ and send pp to $\mathcal{A}$.

2″. Upon an adaptation query $(\tau, \mathbf{h}, \mathbf{m}, r, \mathbf{m}')$ from $\mathcal{A}$, $\mathcal{C}$ proceeds as follows.

- If $\exists\ (\tau, \mathbf{h}'', \mathbf{m}) \in \mathcal{Q}_{\mathsf{Adapt}} \wedge \mathbf{h}'' \neq \mathbf{h}$, or $\exists\ (\tau, \cdot, \mathbf{m}') \in \mathcal{Q}_{\mathsf{Adapt}}$, or $\mathsf{Check}(\tau, \mathbf{h}, \mathbf{m}, r) = 0$ holds, return $\perp$; otherwise, continue.
- Sample $\mathbf{z}' \xleftarrow{\$} \{0,1\}^\kappa$ and set $\mathbf{x}' := \tau \| \mathbf{m}' \| \mathbf{z}'$.
- Compute $\mathbf{y}' \leftarrow \mathsf{PRF}(\mathbf{k}, \mathbf{x}')$ and construct $C[\mathbf{x}', \mathbf{y}']$ as defined by (2).
- $\mathbf{S}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{prv}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i\in[k]})$ and $\mathbf{P}'_{\mathsf{prf}} := \sum_{i\in[h]} m'_i \hat{\mathbf{R}}_i$.
- Set $\mathbf{R}'_{\mathsf{prf}} := \mathbf{S}'_{\mathsf{prf}} + \mathbf{P}'_{\mathsf{prf}}$ and $\mathbf{A}'_{\mathsf{prf}} := \mathbf{A}\mathbf{R}'_{\mathsf{prf}} + \mathbf{G}$. Delegate $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}} \leftarrow \mathsf{TrapDel}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T}_\mathbf{A})$.
- $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2) \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}, \mathbf{h}, \gamma)$ s.t. $\mathbf{e}'_2 \neq 0^w$.
- Send $r' := (\mathbf{z}', \mathbf{y}', \mathbf{e}')$ to $\mathcal{A}$ and $\mathcal{Q}_{\mathsf{Adapt}} := \mathcal{Q}_{\mathsf{Adapt}} \cup \{(\tau, \mathbf{h}, \mathbf{m}), (\tau, \mathbf{h}, \mathbf{m}')\}$.

Lemma 11. *Games G_1 and G_2 are statistically indistinguishable and* $|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_2 \Rightarrow 1]| \leq 2^{-O(\kappa)}$.

Proof of Lemma 11. For each $i \in [k]$, we have

$$\mathbf{A}_i := \mathbf{A}\mathbf{R}_i + k_i\mathbf{G}\ (\text{in } \mathsf{G}_2)\ \approx_s \mathbf{U}_i + k_i\mathbf{G}\ \equiv\ \mathbf{U}'_i =: \mathbf{A}_i\ (\text{in } \mathsf{G}_1)\ ,$$

where $\mathbf{R}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and $\mathbf{U}_i, \mathbf{U}'_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$. The "$\approx_s$" follows from Lemma 2 (randomness extraction) and the triangle inequality. The "$\equiv$" holds due to the uniformity of $\mathbf{U}_i$. Similarly, we can prove that the distribution of $\{\hat{\mathbf{A}}_i\}_{i\in[h]}$ in G_1 is statistically indistinguishable from that of $\{\hat{\mathbf{A}}_i\}_{i\in[h]}$ in G_2 by

$$\hat{\mathbf{A}}_i := \mathbf{A}\hat{\mathbf{R}}_i\ (\text{in } \mathsf{G}_2)\ \ \approx_s\ \mathbf{U}_i =: \hat{\mathbf{A}}\ (\text{in } \mathsf{G}_1)\ ,$$

where $\mathbf{R}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and $\mathbf{U}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times w}$.

Next we show that $\mathbf{A}'_{\mathsf{prf}}$ computed by Eval_{pub} from $\mathbf{A}_i$'s and $\hat{\mathbf{A}}_i$'s in G_1 is identical to that computed by Eval_{prv} from $\mathbf{R}_i$'s and $\hat{\mathbf{R}}_i$'s in G_2. Given $\mathbf{A}_i = \mathbf{A}\mathbf{R}_i + k_i\mathbf{G}$ for $i \in [k]$, we have $\mathbf{C}'_{\mathsf{prf}} := \mathbf{A}\mathbf{S}'_{\mathsf{prf}} + C[\mathbf{x}', \mathbf{y}'](\mathbf{k})\cdot\mathbf{G} = \mathbf{A}\mathbf{S}'_{\mathsf{prf}} + \mathbf{G}$ with $\mathbf{C}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]})$ and $\mathbf{S}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{prv}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i\in[k]})$ due to Lemma 7 (homomorphic evaluation) and the fact that $\mathbf{y}' = \mathsf{PRF}(\mathbf{k}, \mathbf{x}')$. Besides, given $\hat{\mathbf{A}}_i = \mathbf{A}\hat{\mathbf{R}}_i$, we have $\mathbf{B}'_{\mathsf{prf}} := \sum_{i\in[h]} m'_i\hat{\mathbf{A}}_i = \mathbf{A}\sum_{i\in[h]} m'_i\hat{\mathbf{R}}_i = \mathbf{A}\mathbf{P}'_{\mathsf{prf}}$. Then it holds that $\mathbf{A}'_{\mathsf{prf}} := \mathbf{C}'_{\mathsf{prf}} + \mathbf{B}'_{\mathsf{prf}} = \mathbf{A}\mathbf{S}'_{\mathsf{prf}} + \mathbf{G} + \mathbf{A}\mathbf{P}'_{\mathsf{prf}} = \mathbf{A}\mathbf{R}'_{\mathsf{prf}} + \mathbf{G}$ with $\mathbf{R}'_{\mathsf{prf}} = \mathbf{S}'_{\mathsf{prf}} + \mathbf{P}'_{\mathsf{prf}}$. This completes the proof. □

Game G_3. Game G_3 is similar to G_2 except for the generation of the trapdoor $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}$ in the adaptation query phase. In G_2, $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}$ is delegated from $\mathbf{T}_{\mathbf{A}}$. In G_3, $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}$ is generated from a G-trapdoor of $[\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}]$.

$2'''$. Upon an adaptation query $(\tau, \mathbf{h}, \mathbf{m}, r, \mathbf{m}')$ from $\mathcal{A}$, $\mathcal{C}$ proceeds as follows.

- If $\exists\ (\tau, \mathbf{h}'', \mathbf{m}) \in \mathcal{Q}_{\mathsf{Adapt}} \wedge \mathbf{h}'' \neq \mathbf{h}$, or $\exists\ (\tau, \cdot, \mathbf{m}') \in \mathcal{Q}_{\mathsf{Adapt}}$, or $\mathsf{Check}(\tau, \mathbf{h}, \mathbf{m}, r) = 0$ holds, return $\perp$; otherwise, continue.
- Sample $\mathbf{z}' \xleftarrow{\$} \{0,1\}^{\kappa}$ and set $\mathbf{x}' := \tau\|\mathbf{m}'\|\mathbf{z}'$.
- Compute $\mathbf{y}' \leftarrow \mathsf{PRF}(\mathbf{k}, \mathbf{x}')$ and construct $C[\mathbf{x}', \mathbf{y}']$ as defined by (2).
- $\mathbf{S}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{prv}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i\in[k]})$ and $\mathbf{P}'_{\mathsf{prf}} := \sum_{i\in[h]} m'_i\hat{\mathbf{R}}_i$.
- Set $\mathbf{R}'_{\mathsf{prf}} := \mathbf{S}'_{\mathsf{prf}} + \mathbf{P}'_{\mathsf{prf}}$ and $\mathbf{A}'_{\mathsf{prf}} := \mathbf{A}\mathbf{R}'_{\mathsf{prf}} + \mathbf{G}$. Generate $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}} \leftarrow \mathsf{GtoBasis}(\mathbf{R}'_{\mathsf{prf}})$.
- $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2) \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}, \mathbf{h}, \gamma)$ s.t. $\mathbf{e}'_2 \neq 0^w$.
- Send $r' := (\mathbf{z}', \mathbf{y}', \mathbf{e}')$ to $\mathcal{A}$ and $\mathcal{Q}_{\mathsf{Adapt}} := \mathcal{Q}_{\mathsf{Adapt}} \cup \{(\tau, \mathbf{h}, \mathbf{m}), (\tau, \mathbf{h}, \mathbf{m}')\}$.

Lemma 12. *Games G_2 and G_3 are statistically indistinguishable and* $|\Pr[\mathsf{G}_2 \Rightarrow 1] - \Pr[\mathsf{G}_3 \Rightarrow 1]| \leq 2^{-\kappa}$.

Proof of Lemma 12. Note that the changes in G_3 only influence the sampling of $\mathbf{e}'$ during the adaptation query phase, then it suffices to show that the distribution of $\mathbf{e}'$ in G_3 is identical to that in G_2. In G_2, $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}$ is delegated from $\mathbf{T}_{\mathbf{A}}$ and of norm $\|\tilde{\mathbf{T}}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}\| = \|\tilde{\mathbf{T}}_{\mathbf{A}}\| \leq O(\sqrt{n\log q})$ according to Lemma 5 (trapdoor delegation). Together with Lemma 4 (preimage sampling) and the parameter setting that $\gamma > O(\sqrt{n\log q})\cdot\omega(\sqrt{m+w})$, the vector $\mathbf{e}'$ sampled in G_2 follows the distribution $D_{\Lambda_q^{\mathbf{h}}(\mathbf{A}),\gamma}$. In G_3, we have $\mathbf{A}'_{\mathsf{prf}} = \mathbf{A}\mathbf{R}'_{\mathsf{prf}} + \mathbf{G}$, and hence $\mathbf{R}'_{\mathsf{prf}}$ is a G-trapdoor for $[\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}]$ according to [28]. Then according to Lemma 6 (G-to-basis), $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}$ generated from the G-trapdoor $\mathbf{R}'_{\mathsf{prf}}$ is also a trapdoor for $[\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}]$ with norm $\|\tilde{\mathbf{T}}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}\| = \sqrt{5}(s_1(\mathbf{R}'_{\mathsf{prf}}) + 1) \leq O(\kappa^c)$ for some constant c. Together with Lemma 4 (preimage sampling) and the parameter setting that $\gamma \geq O(\kappa^c)\cdot\omega(\sqrt{m+w})$, the vector $\mathbf{e}'$ sampled in G_3 also follows the distribution $D_{\Lambda_q^{\mathbf{h}}(\mathbf{A}),\gamma}$. This completes the proof. □

Game G_4. Game G_4 is similar to G_3 except for the generation of $\mathbf{A}$ in the setup phase. In G_3, $\mathbf{A}$ is generated by algorithm $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$. In G_4, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ is sampled uniformly at random.

1‴. During the setup phase, the challenger $\mathcal{C}$ proceeds as follows.

  - Generate $pp_{\mathsf{prf}} \leftarrow \mathsf{PRF.Setup}(1^\kappa)$ and sample $\mathbf{k} \xleftarrow{\$} \{0,1\}^k$.
  - Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
  - Sample $\mathbf{R}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and set $\mathbf{A}_i := \mathbf{A}\mathbf{R}_i + k_i\mathbf{G}$ for $i \in [k]$. Sample $\hat{\mathbf{R}}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and set $\hat{\mathbf{A}}_i := \mathbf{A}\hat{\mathbf{R}}_i$ for $i \in [h]$.
  - $pp := (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]}, \{\hat{\mathbf{A}}_i\}_{i\in[h]})$, $td := (pp, \perp)$ and send pp to $\mathcal{A}$.

Lemma 13. *Games* G_3 *and* G_4 *are statistically indistinguishable and* $|\Pr[\mathsf{G}_3 \Rightarrow 1] - \Pr[\mathsf{G}_4 \Rightarrow 1]| \le 2^{-\kappa}$.

Lemma 13 holds directly from Lemma 3 (trapdoor generation).

Next we show that any PPT adversary $\mathcal{A}$ wins in G_4 with negligible probability. To do this, we classify the adversaries into two types, $\mathcal{A}^{(I)}$ and $\mathcal{A}^{(II)}$.

- **Type I:** $\mathcal{A}^{(I)}$ finally submits a forgery $(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*, \mathbf{m}'^*, r'^*)$ satisfying the first Valid condition, i.e., $(\tau^*, \cdot, \mathbf{m}^*) \notin \mathcal{Q}_{\mathsf{Adapt}} \wedge (\tau^*, \mathbf{h}^*, \mathbf{m}'^*) \in \mathcal{Q}_{\mathsf{Adapt}}$.
- **Type II:** $\mathcal{A}^{(II)}$ finally submits a forgery $(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*, \mathbf{m}'^*, r'^*)$ satisfying the second Valid condition, i.e., $(\tau^*, \cdot, \mathbf{m}^*) \notin \mathcal{Q}_{\mathsf{Adapt}} \wedge (\tau^*, \cdot, \mathbf{m}'^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$.

Next we show in Lemma 14 that $\mathcal{A}^{(I)}$ and $\mathcal{A}^{(II)}$ hardly win in G_4.

Lemma 14. *For any PPT adversary* $\mathcal{A}^{(T)}$ *with* $T \in \{I, II\}$*, it holds that* $\Pr[\mathsf{G}_4 \Rightarrow 1] \le \mathsf{Adv}^{pse}_{\mathsf{PRF}}(\kappa) + \mathsf{Adv}^{\mathsf{SIS}}_{[n,q,\beta,m]}(\kappa) + 2^{-\kappa}$.

Proof of Lemma 14. We consider $\mathcal{A}^{(I)}$ and $\mathcal{A}^{(II)}$ separately.

First, we prove that if there exists a PPT $\mathcal{A}^{(I)}$ that wins in G_4, then we construct a PPT algorithm $\mathcal{B}^{(I)}$ to solve the SIS problem.

Algorithm $\mathcal{B}^{(I)}$. Given an SIS instance $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathcal{B}^{(I)}$ aims to obtain a non-zero short vector $\mathbf{v} \in \mathbb{Z}_q^m$ s.t. $\mathbf{A}\mathbf{v} = 0^n$. It proceeds as follows.

0. The algorithm $\mathcal{B}^{(I)}$ initializes sets $\mathcal{Q}_{\mathsf{Adapt}} := \emptyset$ and $\mathcal{Q}_r := \emptyset$.
1. During the setup phase, the challenger $\mathcal{B}^{(I)}$ proceeds as follows.
   - Generate $pp_{\mathsf{prf}} \leftarrow \mathsf{PRF.Setup}(1^\kappa)$ and sample $\mathbf{k} \xleftarrow{\$} \{0,1\}^k$.
   - Sample $\mathbf{R}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and set $\mathbf{A}_i := \mathbf{A}\mathbf{R}_i + k_i\mathbf{G}$ for $i \in [k]$. Sample $\hat{\mathbf{R}}_i \xleftarrow{\$} \{\pm 1\}^{m\times w}$ and set $\hat{\mathbf{A}}_i := \mathbf{A}\hat{\mathbf{R}}_i$ for $i \in [h]$. (Note that $\mathbf{A}$ is the SIS instance.)
   - Send $pp = (pp_{\mathsf{prf}}, \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]}, \{\hat{\mathbf{A}}_i\}_{i\in[h]})$ to $\mathcal{A}^{(I)}$.
2. Upon an adaptation query $(\tau, \mathbf{h}, \mathbf{m}, r, \mathbf{m}')$ from $\mathcal{A}^{(I)}$, $\mathcal{B}^{(I)}$ proceeds as follows.
   - If $\exists\ (\tau, \mathbf{h}'', \mathbf{m}) \in \mathcal{Q}_{\mathsf{Adapt}} \wedge \mathbf{h}'' \ne \mathbf{h}$, or $\exists\ (\tau, \cdot, \mathbf{m}') \in \mathcal{Q}_{\mathsf{Adapt}}$, or $\mathsf{Check}(\tau, \mathbf{h}, \mathbf{m}, r) = 0$ holds, return $\perp$; otherwise, continue.
   - Sample $\mathbf{z}' \xleftarrow{\$} \{0,1\}^\kappa$ and set $\mathbf{x}' := \tau \| \mathbf{m}' \| \mathbf{z}'$.
   - Compute $\mathbf{y}' \leftarrow \mathsf{PRF}(\mathbf{k}, \mathbf{x}')$ and construct $C[\mathbf{x}', \mathbf{y}']$ as defined by (2).
   - $\mathbf{S}'_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{prv}(C[\mathbf{x}', \mathbf{y}'](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i\in[k]})$ and $\mathbf{P}'_{\mathsf{prf}} := \sum_{i\in[h]} m'_i \hat{\mathbf{R}}_i$.

- Set $\mathbf{R}'_{\mathsf{prf}} := \mathbf{S}'_{\mathsf{prf}} + \mathbf{P}'_{\mathsf{prf}}$ and $\mathbf{A}'_{\mathsf{prf}} := \mathbf{A}\mathbf{R}'_{\mathsf{prf}} + \mathbf{G}$. Generate $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}} \leftarrow \mathsf{GtoBasis}(\mathbf{R}'_{\mathsf{prf}})$.
- $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2) \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}], \mathbf{T}_{\mathbf{A}|\mathbf{A}'_{\mathsf{prf}}}, \mathbf{h}, \gamma)$ s.t. $\mathbf{e}'_2 \neq 0^w$.
- Send $r' := (\mathbf{z}', \mathbf{y}', \mathbf{e}')$ to $\mathcal{A}^{(I)}$. Set $\mathcal{Q}_{\mathsf{Adapt}} := \mathcal{Q}_{\mathsf{Adapt}} \cup \{(\tau, \mathbf{h}, \mathbf{m}), (\tau, \mathbf{h}, \mathbf{m}')\}$ and $\mathcal{Q}_r := \mathcal{Q}_r \cup \{(\tau, \mathbf{h}, \mathbf{m}, r), (\tau, \mathbf{h}, \mathbf{m}', r')\}$.

3. Upon a forgery tuple $(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*, \mathbf{m}'^*, r'^*)$, if $\mathcal{A}^{(I)}$ wins, it holds that $\mathbf{m}^* \neq \mathbf{m}'^*$, $\mathsf{Check}(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*) = \mathsf{Check}(\tau^*, \mathbf{h}^*, \mathbf{m}'^*, r'^*) = 1$ and $(\tau^*, \cdot, \mathbf{m}^*) \notin \mathcal{Q}_{\mathsf{Adapt}} \wedge (\tau^*, \mathbf{h}^*, \mathbf{m}'^*) \in \mathcal{Q}_{\mathsf{Adapt}}$. Find $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}) \in \mathcal{Q}_{\mathsf{Adapt}}$ s.t. $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{r}) \in \mathcal{Q}_r$, $\mathsf{Check}(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{r}) = 1$ and $\bar{\mathbf{m}}$ is never queried to $\mathcal{O}_{\mathsf{Adapt}}$ as the adapted message w.r.t. tag τ^*. Then $\mathcal{B}^{(I)}$ computes a SIS solution as follows.
   - Parse $\bar{r} = (\bar{\mathbf{z}}, \bar{\mathbf{y}}, \bar{\mathbf{e}})$ and $r^* = (\mathbf{z}^*, \mathbf{y}^*, \mathbf{e}^*)$.
   - Set $\bar{\mathbf{x}} := \tau^* \| \bar{\mathbf{m}} \| \bar{\mathbf{z}}$ and $\mathbf{x}^* := \tau^* \| \mathbf{m}^* \| \mathbf{z}^*$. Construct $C[\bar{\mathbf{x}}, \bar{\mathbf{y}}]$ and $C[\mathbf{x}^*, \mathbf{y}^*]$.
   - $\bar{\mathbf{C}}_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\bar{\mathbf{x}}, \bar{\mathbf{y}}](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]})$ and $\bar{\mathbf{B}}_{\mathsf{prf}} := \sum_{i \in [h]} \bar{m}_i \hat{\mathbf{A}}_i$.
   - $\bar{\mathbf{S}}_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{prv}(C[\bar{\mathbf{x}}, \bar{\mathbf{y}}](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i \in [k]})$ and $\bar{\mathbf{P}}_{\mathsf{prf}} := \sum_{i \in [h]} \bar{m}_i \hat{\mathbf{R}}_i$.
   - Set $\bar{\mathbf{A}}_{\mathsf{prf}} := \bar{\mathbf{C}}_{\mathsf{prf}} + \bar{\mathbf{B}}_{\mathsf{prf}}$ and $\bar{\mathbf{R}}_{\mathsf{prf}} := \bar{\mathbf{S}}_{\mathsf{prf}} + \bar{\mathbf{P}}_{\mathsf{prf}}$.
   - $\mathbf{C}^*_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}^*, \mathbf{y}^*](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i \in [k]})$ and $\mathbf{B}^*_{\mathsf{prf}} := \sum_{i \in [h]} m^*_i \hat{\mathbf{A}}_i$.
   - $\mathbf{S}^*_{\mathsf{prf}} \leftarrow \mathsf{Eval}_{prv}(C[\mathbf{x}^*, \mathbf{y}^*](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i \in [k]})$ and $\mathbf{P}^*_{\mathsf{prf}} := \sum_{i \in [h]} m^*_i \hat{\mathbf{R}}_i$.
   - Set $\mathbf{A}^*_{\mathsf{prf}} := \mathbf{C}^*_{\mathsf{prf}} + \mathbf{B}^*_{\mathsf{prf}}$ and $\mathbf{R}^*_{\mathsf{prf}} := \mathbf{S}^*_{\mathsf{prf}} + \mathbf{P}^*_{\mathsf{prf}}$.
   - Compute and return $\mathbf{v} := [\mathbf{I}_m | \mathbf{R}^*_{\mathsf{prf}}] \cdot \mathbf{e}^* - [\mathbf{I}_m | \bar{\mathbf{R}}_{\mathsf{prf}}] \cdot \bar{\mathbf{e}}$ to its own challenger.

We show the existence of tuple $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}) \in \mathcal{Q}_{\mathsf{Adapt}}$ in step 3. The adversary may issue multiple adaptation queries centered around $(\tau^*, \mathbf{h}^*)$, but there must be a root tuple $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{\mathbf{r}})$ such that all other tuples $(\tau^*, \mathbf{h}^*, \cdot, \cdot)$ are adapted from it directly or indirectly. According to the specification of $\mathcal{O}_{\mathsf{Adapt}}$, all the target new messages w.r.t. τ^* are different from the root message $\bar{\mathbf{m}}$. Consequently, tuple $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{\mathbf{r}})$ satisfies $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{r}) \in \mathcal{Q}_r$, $\mathsf{Check}(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{r}) = 1$ and $\bar{\mathbf{m}}$ is never queried to $\mathcal{O}_{\mathsf{Adapt}}$ as the adapted message w.r.t. τ^*.

Next we show that $\mathbf{v}$ is a valid solution to the SIS problem. Note that $(\tau^*, \cdot, \mathbf{m}^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$ is never queried to the adaptation oracle, then nothing about $\mathsf{PRF}(\mathbf{k}, \mathbf{x}^*)$ with $\mathbf{x}^* = \tau^* \| \mathbf{m}^* \| \mathbf{z}^*$ is revealed to $\mathcal{A}^{(I)}$. For $\mathbf{y}^*$ chosen by $\mathcal{A}^{(I)}$, $\mathbf{y}^* = \mathsf{PRF}(\mathbf{k}, \mathbf{x}^*)$ hardly holds due to the pseudorandomness of PRF. Then with overwhelming probability, $C[\mathbf{x}^*, \mathbf{y}^*](\mathbf{k}) = 0$ and

$$\mathbf{A}^*_{\mathsf{prf}} = \mathbf{A}\mathbf{R}^*_{\mathsf{prf}} + C[\mathbf{x}^*, \mathbf{y}^*](\mathbf{k}) \cdot \mathbf{G} = \mathbf{A}\mathbf{R}^*_{\mathsf{prf}} + 0 \cdot \mathbf{G} = \mathbf{A}\mathbf{R}^*_{\mathsf{prf}}.$$

Besides, since $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}) \in \mathcal{Q}_{\mathsf{Adapt}}$ and $\bar{\mathbf{m}}$ is never queried to $\mathcal{Q}_{\mathsf{Adapt}}$ as a target new message under tag τ^* before, we know that $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{r})$ is generated by $\mathcal{A}^{(I)}$ itself and $\mathsf{PRF}(\mathbf{k}, \bar{\mathbf{x}})$ with $\bar{\mathbf{x}} = \tau^* \| \bar{\mathbf{m}} \| \bar{\mathbf{z}}$ is never obtained by $\mathcal{A}^{(I)}$. Through an analogous analysis, we know that with overwhelming probability,

$$\bar{\mathbf{A}}_{\mathsf{prf}} = \mathbf{A}\bar{\mathbf{R}}_{\mathsf{prf}} + C[\bar{\mathbf{x}}, \bar{\mathbf{y}}](\mathbf{k}) \cdot \mathbf{G} = \mathbf{A}\bar{\mathbf{R}}_{\mathsf{prf}} + 0 \cdot \mathbf{G} = \mathbf{A}\bar{\mathbf{R}}_{\mathsf{prf}}.$$

Furthermore, since $\mathsf{Check}(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*) = \mathsf{Check}(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}, \bar{r}) = 1$, we have

$$\begin{aligned} [\mathbf{A}|\mathbf{A}^*_{\mathsf{prf}}] \cdot \mathbf{e}^* = \mathbf{h}^* = [\mathbf{A}|\bar{\mathbf{A}}_{\mathsf{prf}}] \cdot \bar{\mathbf{e}} \;\Leftrightarrow\; & [\mathbf{A}|\mathbf{A}\mathbf{R}^*_{\mathsf{prf}}] \cdot \mathbf{e}^* - [\mathbf{A}|\mathbf{A}\bar{\mathbf{R}}_{\mathsf{prf}}] \cdot \bar{\mathbf{e}} = 0^n \\ \Leftrightarrow\; & \mathbf{A}\,\underbrace{\left([\mathbf{I}_m|\mathbf{R}^*_{\mathsf{prf}}] \cdot \mathbf{e}^* - [\mathbf{I}_m|\bar{\mathbf{R}}_{\mathsf{prf}}] \cdot \bar{\mathbf{e}}\right)}_{=:\mathbf{v} \in \mathbb{Z}_q^m} = 0^n, \end{aligned}$$

where $\mathbf{e}^* = (\mathbf{e}_1^*, \mathbf{e}_2^*)$, $\|\mathbf{e}^*\| \le \gamma\sqrt{m+w}$, $\mathbf{e}_2^* \ne 0^w$, $\bar{\mathbf{e}} = (\bar{\mathbf{e}}_1, \bar{\mathbf{e}}_2)$, $\|\bar{\mathbf{e}}\| \le \gamma\sqrt{m+w}$ and $\bar{\mathbf{e}}_2 \ne 0^w$. Together with our parameter setting that $\gamma \cdot O(\kappa^c) \cdot \sqrt{m+w} \le \beta$, we have $\|\mathbf{v}\| \le O(\kappa^c) \cdot \gamma\sqrt{m+w} \le \beta$ for some constant c.

It remains to show that $\mathbf{v} = ([\mathbf{I}_m|\mathbf{R}_{\mathsf{prf}}^*] \cdot \mathbf{e}^* - [\mathbf{I}_m|\bar{\mathbf{R}}_{\mathsf{prf}}] \cdot \bar{\mathbf{e}}) \ne 0^m$. Denote by $\mathbf{r}_i^*$ (resp., $\bar{\mathbf{r}}_i$, $\mathbf{s}_i^*$, $\mathbf{p}_i^*$ and $\{\hat{\mathbf{r}}_{j,i}\}_{j\in[h]}$) the i-th column of $\mathbf{R}_{\mathsf{prf}}^*$ (resp., $\bar{\mathbf{R}}_{\mathsf{prf}}$, $\mathbf{S}_{\mathsf{prf}}^*$, $\mathbf{P}_{\mathsf{prf}}^*$ and $\{\hat{\mathbf{R}}_j\}_{j\in[h]}$), and $e_{2,i}^*$ the i-th item of $\mathbf{e}_2^*$. Recall that $\mathbf{r}_i^* = \mathbf{s}_i^* + \mathbf{p}_i^* = \mathbf{s}_i^* + \sum_{j\in[h]} m_j^* \hat{\mathbf{r}}_{j,i}$. Since $(\tau^*, \cdot, \mathbf{m}^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$ and $(\tau^*, \mathbf{h}^*, \bar{\mathbf{m}}) \in \mathcal{Q}_{\mathsf{Adapt}}$, we know that $\bar{\mathbf{m}} \ne \mathbf{m}^*$ and hence there must exist some index $\iota \in [h]$ s.t. $\bar{m}_\iota \ne m_\iota^*$. W.l.o.g., let $\bar{m}_\iota = 0$ and $m_\iota^* = 1$. Besides, since $\mathbf{e}_2^* \ne 0^w$, there must exist some index $\nu \in [w]$ s.t. $e_{2,\nu}^* \ne 0$. Now we show that $\mathbf{v} = 0^m$ holds with negligible probability. Note that

$$\begin{aligned} &\mathbf{v} = \mathbf{e}_1^* + \mathbf{R}_{\mathsf{prf}}^* \mathbf{e}_2^* - \bar{\mathbf{e}}_1 - \bar{\mathbf{R}}_{\mathsf{prf}} \bar{\mathbf{e}}_2 = 0^m \\ \Leftrightarrow\ &\hat{\mathbf{r}}_{\iota,\nu} = \underbrace{(\bar{\mathbf{e}}_1 + \bar{\mathbf{R}}_{\mathsf{prf}} \bar{\mathbf{e}}_2 - \mathbf{e}_1^* - \sum\nolimits_{i\ne\nu} \mathbf{r}_i^* e_{2,i}^*)/e_{2,\nu}^* - \mathbf{s}_\nu^* - \sum\nolimits_{j\ne\iota} m_j^* \hat{\mathbf{r}}_{j,\nu}}_{=:W}. \end{aligned} \quad (3)$$

Recall that $\hat{\mathbf{r}}_{\iota,\nu}$ is sampled uniformly from $\{1,-1\}^m$ and the only information of $\hat{\mathbf{r}}_{\iota,\nu}$ revealed to $\mathcal{A}^{(I)}$ is $\mathbf{u} = \mathbf{A}\hat{\mathbf{r}}_{\iota,\nu} \in \mathbb{Z}_q^n$. Together with Lemma 1 and the parameter setting that $m \ge O(n\log q)$, $\tilde{\mathbf{H}}_\infty(\hat{\mathbf{r}}_{\iota,\nu}|\mathbf{u}) \ge \mathbf{H}_\infty(\hat{\mathbf{r}}_{\iota,\nu}) - n\log q = m - n\log q \ge \kappa$ and $\hat{\mathbf{r}}_{\iota,\nu}$ still has high entropy. Further since "W" in Eq. (3) is independent of $\hat{\mathbf{r}}_{\iota,\nu}$, we have $\hat{\mathbf{r}}_{\iota,\nu} = W$ with probability $2^{-\kappa}$. Then Eq. (3) holds with a negligible probability and $\mathbf{v} = 0^m$ holds with negligible probability.

Now we have proved that $\mathbf{v}$ is a valid solution for SIS and $\Pr[\mathsf{G}_4 \Rightarrow 1|\mathcal{A}^{(I)}] \le \mathsf{Adv}_{\mathsf{PRF}}^{pse}(\kappa) + \mathsf{Adv}_{[n,q,\beta,m]}^{\mathsf{SIS}}(\kappa) + 2^{-\kappa}$.

Next, we prove that if there exists a PPT $\mathcal{A}^{(II)}$ that wins in G_4, then we construct a PPT algorithm $\mathcal{B}^{(II)}$ to solve the SIS problem. The algorithm $\mathcal{B}^{(II)}$ is similar to $\mathcal{B}^{(I)}$ except for the step 3, as described below.

3. Upon a forgery tuple $(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*, \mathbf{m}'^*, r'^*)$, if $\mathcal{A}^{(II)}$ wins, it holds that $\mathbf{m}^* \ne \mathbf{m}'^*$, $\mathsf{Check}(\tau^*, \mathbf{h}^*, \mathbf{m}^*, r^*) = \mathsf{Check}(\tau^*, \mathbf{h}^*, \mathbf{m}'^*, r'^*) = 1$ and $(\tau^*, \cdot, \mathbf{m}^*) \notin \mathcal{Q}_{\mathsf{Adapt}} \wedge (\tau^*, \cdot, \mathbf{m}'^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$. Then $\mathcal{B}^{(II)}$ computes a SIS solution as follows.
   - Parse $r^* = (\mathbf{z}^*, \mathbf{y}^*, \mathbf{e}^*)$ and $r'^* = (\mathbf{z}'^*, \mathbf{y}'^*, \mathbf{e}'^*)$. Set $\mathbf{x}^* := \tau^*\|\mathbf{m}^*\|\mathbf{z}^*$ and $\mathbf{x}'^* := \tau^*\|\mathbf{m}'^*\|\mathbf{z}'^*$. Construct $C[\mathbf{x}^*, \mathbf{y}^*]$ and $C[\mathbf{x}'^*, \mathbf{y}'^*]$.
   - $\mathbf{C}_{\mathsf{prf}}^* \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}^*, \mathbf{y}^*](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]})$ and $\mathbf{B}_{\mathsf{prf}}^* := \sum_{i\in[h]} m_i^* \hat{\mathbf{A}}_i$.
   - $\mathbf{S}_{\mathsf{prf}}^* \leftarrow \mathsf{Eval}_{prv}(C[\mathbf{x}^*, \mathbf{y}^*](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i\in[k]})$ and $\mathbf{P}_{\mathsf{prf}}^* := \sum_{i\in[h]} m_i^* \hat{\mathbf{R}}_i$.
   - Set $\mathbf{A}_{\mathsf{prf}}^* := \mathbf{C}_{\mathsf{prf}}^* + \mathbf{B}_{\mathsf{prf}}^*$ and $\mathbf{R}_{\mathsf{prf}}^* := \mathbf{S}_{\mathsf{prf}}^* + \mathbf{P}_{\mathsf{prf}}^*$.
   - $\mathbf{C}_{\mathsf{prf}}'^* \leftarrow \mathsf{Eval}_{pub}(C[\mathbf{x}'^*, \mathbf{y}'^*](\cdot), \mathbf{A}, \{\mathbf{A}_i\}_{i\in[k]})$ and $\mathbf{B}_{\mathsf{prf}}'^* := \sum_{i\in[h]} m_i'^* \hat{\mathbf{A}}_i$.
   - $\mathbf{S}_{\mathsf{prf}}'^* \leftarrow \mathsf{Eval}_{prv}(C[\mathbf{x}'^*, \mathbf{y}'^*](\cdot), \mathbf{A}, \mathbf{k}, \{\mathbf{R}_i\}_{i\in[k]})$ and $\mathbf{P}_{\mathsf{prf}}'^* := \sum_{i\in[h]} m_i'^* \hat{\mathbf{R}}_i$.
   - Set $\mathbf{A}_{\mathsf{prf}}'^* := \mathbf{C}_{\mathsf{prf}}'^* + \mathbf{B}_{\mathsf{prf}}'^*$ and $\mathbf{R}_{\mathsf{prf}}'^* := \mathbf{S}_{\mathsf{prf}}'^* + \mathbf{P}_{\mathsf{prf}}'^*$.
   - Compute and return $\mathbf{v} := [\mathbf{I}_m|\mathbf{R}_{\mathsf{prf}}^*] \cdot \mathbf{e}^* - [\mathbf{I}_m|\mathbf{R}_{\mathsf{prf}}'^*] \cdot \mathbf{e}'^*$ to its challenger.

Then we show that $\mathbf{v}$ is a valid solution to the SIS problem. Note that $(\tau^*, \cdot, \mathbf{m}^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$ and $(\tau^*, \cdot, \mathbf{m}'^*) \notin \mathcal{Q}_{\mathsf{Adapt}}$ are not queried to the adaptation oracle, so nothing about $\mathsf{PRF}(\mathbf{k}, \mathbf{x}^*)$ and $\mathsf{PRF}(\mathbf{k}, \mathbf{x}'^*)$ with $\mathbf{x}^* = \tau^*\|\mathbf{m}^*\|\mathbf{z}^*$

and $\mathbf{x}'^* = \tau^* \| \mathbf{m}'^* \| \mathbf{z}'^*$, has ever been revealed to $\mathcal{A}^{(II)}$, and hence $\mathsf{PRF}(\mathbf{k}, \mathbf{x}^*)$ and $\mathsf{PRF}(\mathbf{k}, \mathbf{x}'^*)$ are pseudorandom due to the pseudorandomness of PRF. As a consequence, neither $\mathbf{y}^* = \mathsf{PRF}(\mathbf{k}, \mathbf{x}^*)$ nor $\mathbf{y}'^* = \mathsf{PRF}(\mathbf{k}, \mathbf{x}'^*)$ holds except for negligible probability, where $\mathbf{y}^*$ and $\mathbf{y}'^*$ are chosen by $\mathcal{A}^{(II)}$, and this leads to $C[\mathbf{x}^*, \mathbf{y}^*](\mathbf{k}) = 0$ and $C[\mathbf{x}'^*, \mathbf{y}'^*](\mathbf{k}) = 0$. Therefore,

$$\mathbf{A}^*_{\mathsf{prf}} = \mathbf{A}\mathbf{R}^*_{\mathsf{prf}} + C[\mathbf{x}^*, \mathbf{y}^*](\mathbf{k}) \cdot \mathbf{G} = \mathbf{A}\mathbf{R}^*_{\mathsf{prf}} + 0 \cdot \mathbf{G} = \mathbf{A}\mathbf{R}^*_{\mathsf{prf}}$$
$$\mathbf{A}'^*_{\mathsf{prf}} = \mathbf{A}\mathbf{R}'^*_{\mathsf{prf}} + C[\mathbf{x}'^*, \mathbf{y}'^*](\mathbf{k}) \cdot \mathbf{G} = \mathbf{A}\mathbf{R}'^*_{\mathsf{prf}} + 0 \cdot \mathbf{G} = \mathbf{A}\mathbf{R}'^*_{\mathsf{prf}}.$$

Through an analogous analysis as before, $\mathbf{v} := [\mathbf{I}_m | \mathbf{R}^*_{\mathsf{prf}}] \cdot \mathbf{e}^* - [\mathbf{I}_m | \mathbf{R}'^*_{\mathsf{prf}}] \cdot \mathbf{e}'^*$ is a valid SIS solution with overwhelming probability. Now we obtain $\Pr[\mathsf{G}_4 \Rightarrow 1 | \mathcal{A}^{(II)}] \leq \mathsf{Adv}^{pse}_{\mathsf{PRF}}(\kappa) + \mathsf{Adv}^{\mathsf{SIS}}_{[n,q,\beta,m]}(\kappa) + 2^{-\kappa}$. □

From Lemma 14, we have

$$\begin{aligned} \Pr[\mathsf{G}_4 \Rightarrow 1] &= \Pr[\mathsf{G}_4 \Rightarrow 1 | \mathcal{A}^{(I)}] \Pr[\mathcal{A}^{(I)}] + \Pr[\mathsf{G}_4 \Rightarrow 1 | \mathcal{A}^{(II)}] \Pr[\mathcal{A}^{(II)}] \\ &\leq \mathsf{Adv}^{pse}_{\mathsf{PRF}}(\kappa) + \mathsf{Adv}^{\mathsf{SIS}}_{[n,q,\beta,m]}(\kappa) + 2^{-O(\kappa)}. \end{aligned} \tag{4}$$

Finally combining Lemmas 10, 11, 12, 13 and (4), it holds that

$$\begin{aligned} &\Pr[\mathsf{Exp}^{rcr}_{\mathsf{tCH},\mathcal{A}}(\kappa) \Rightarrow 1] \\ \leq &\big|\Pr[\mathsf{G}_0 \Rightarrow 1] - \Pr[\mathsf{G}_1 \Rightarrow 1]\big| + \big|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_2 \Rightarrow 1]\big| \\ &+ \big|\Pr[\mathsf{G}_2 \Rightarrow 1] - \Pr[\mathsf{G}_3 \Rightarrow 1]\big| + |\Pr[\mathsf{G}_3 \Rightarrow 1] - \Pr[\mathsf{G}_4 \Rightarrow 1]| + \Pr[\mathsf{G}_4 \Rightarrow 1] \\ \leq &\mathsf{Adv}^{\mathsf{SIS}}_{[n,q,\beta,m]}(\kappa) + 2\mathsf{Adv}^{pse}_{\mathsf{PRF}}(\kappa) + 2^{-O(\kappa)}. \end{aligned} \tag{5}$$

By (5), it is easy to see that the r-CR security of tCH can be tightly reduced to the SIS assumption and the pseudorandomness of PRF. Given the concrete PRF schemes [5], our tCH enjoys r-CR based on the SIS and LWE assumptions. ■

4.2 tCH with Tight Security in ROM

In this subsection, we provide another lattice-based tCH construction, namely tCH', with r-CR security proved in the random oracle model. Compared with tCH in Fig. 4, our second tCH construction replaces the underlying homomorphic evaluations and PRF with a hash function (which is modeled as a random oracle), and hence achieves better efficiency and tightness. Let $\mathsf{H} : \mathbb{Z}_q^{n \times m} \times \{0,1\}^x \rightarrow \mathbb{Z}_q^{n \times w}$ be a hash function. Our tCH scheme tCH' is given in Fig. 5.

Parameter setting. Parameters of tCH' include the security parameter κ, the dimension parameters x, t, h, the SIS parameters n, m, q, β and the Gaussian parameter γ. Define $w = n\lceil \log q \rceil$. The afore-mentioned parameters are required to satisfy the following restrictions simultaneously.

- Let $x, t, h = \mathsf{poly}(\kappa)$ be positive integers and $x = t + h + \kappa$.
- Let n, q, m, β be positive parameters, $n, m, \beta, q = \mathsf{poly}(\kappa)$ and $\beta \cdot \mathsf{poly}(n) \leq q$ so that the SIS problem is hard according to Lemma 9.

```
$(pp, td) \leftarrow \mathsf{Setup}(1^\kappa)$.
  1. Generate $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ with $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$.
  2. Return $pp := \mathbf{A}$ and $td := (pp, \mathbf{T_A})$.

$(\mathbf{h}, r) \leftarrow \mathsf{Hash}(pp, \tau \in \{0,1\}^t, \mathbf{m} \in \{0,1\}^h)$.
  1. Parse $pp = \mathbf{A}$.
  2. Sample $\mathbf{z} \xleftarrow{\$} \{0,1\}^\kappa$ and set $\mathbf{x} := \tau \| \mathbf{m} \| \mathbf{z} \in \{0,1\}^x$.
  3. Compute $\mathbf{A}_h = \mathsf{H}(\mathbf{A}, \mathbf{x})$ with $\mathbf{A}_h \in \mathbb{Z}_q^{n\times w}$.
  4. Sample $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2) \leftarrow D_{\mathbb{Z}^{m+w},\gamma}$ with $\mathbf{e}_1 \in \mathbb{Z}_q^m$ and $\mathbf{e}_2 \in \mathbb{Z}_q^w$ s.t. $\mathbf{e}_2 \neq 0^w$.
  5. Return $\mathbf{h} = [\mathbf{A}|\mathbf{A}_h] \cdot \mathbf{e} \in \mathbb{Z}_q^n$ and $r = (\mathbf{z}, \mathbf{e})$.

$r' \leftarrow \mathsf{Adapt}(td, \tau \in \{0,1\}^t, \mathbf{h}, \mathbf{m} \in \{0,1\}^h, r, \mathbf{m}' \in \{0,1\}^h)$.
  1. Parse $td = (pp, \mathbf{T_A})$ and $pp = \mathbf{A}$.
  2. If $\mathsf{Check}(pp, \tau, \mathbf{h}, \mathbf{m}, r) = 0$, return $\perp$. Otherwise, continue.
  3. Sample $\mathbf{z}' \xleftarrow{\$} \{0,1\}^\kappa$ and set $\mathbf{x}' := \tau \| \mathbf{m}' \| \mathbf{z}' \in \{0,1\}^x$.
  4. Compute $\mathbf{A}'_h = \mathsf{H}(\mathbf{A}, \mathbf{x}')$ with $\mathbf{A}'_h \in \mathbb{Z}_q^{n\times w}$.
  5. Delegate $\mathbf{T}_{\mathbf{A}|\mathbf{A}'_h} \leftarrow \mathsf{TrapDel}([\mathbf{A}|\mathbf{A}'_h], \mathbf{T_A})$.
  6. Sample $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2) \leftarrow \mathsf{SamplePre}([\mathbf{A}|\mathbf{A}'_h], \mathbf{T}_{\mathbf{A}|\mathbf{A}'_h}, \mathbf{h}, \gamma)$ with $\mathbf{e}'_1 \in \mathbb{Z}_q^m$ and $\mathbf{e}'_2 \in \mathbb{Z}_q^w$ s.t. $\mathbf{e}'_2 \neq 0^w$.
  7. Return $r' = (\mathbf{z}', \mathbf{e}')$.

$0/1 \leftarrow \mathsf{Check}(pp, \tau \in \{0,1\}^t, \mathbf{h}, \mathbf{m} \in \{0,1\}^h, r)$.
  1. Parse $pp = \mathbf{A}$ and $r = (\mathbf{z}, \mathbf{e})$. Set $\mathbf{x} := \tau \| \mathbf{m} \| \mathbf{z} \in \{0,1\}^x$.
  2. Compute $\mathbf{A}_h = \mathsf{H}(\mathbf{A}, \mathbf{x})$ with $\mathbf{A}_h \in \mathbb{Z}_q^{n\times w}$.
  3. Parse $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2)$ with $\mathbf{e}_1 \in \mathbb{Z}_q^m$ and $\mathbf{e}_2 \in \mathbb{Z}_q^w$. If $\mathbf{h} = [\mathbf{A}|\mathbf{A}_h] \cdot \mathbf{e}$, $\|\mathbf{e}\| \le \gamma\sqrt{m+w}$ and $\mathbf{e}_2 \neq 0^w$, return 1; otherwise, return 0.
```

Fig. 5. Tagged chameleon hash tCH' in the random oracle model.

- Let $\gamma \ge \omega(\sqrt{m(m+w)})$ so that Lemma 4 can be applied.
- Let $m = O(n \log q)$ and $2\gamma\sqrt{m(m+w)} \le \beta$ to serve for our security proof.

Theorem 3. *Let* $\mathsf{H} : \mathbb{Z}_q^{n\times m} \times \{0,1\}^x \to \mathbb{Z}_q^{n\times w}$ *be a hash function modeled as a random oracle. Given parameters described above, construction* tCH' *in Fig. 5 is a tagged chameleon hash if the* $\mathsf{SIS}_{n,q,\beta,m}$ *assumption holds. Furthermore, restricted collision resistance of* tCH' *enjoys tight security:*

$$\Pr[\mathsf{Exp}^{rcr}_{\mathsf{tCH}',\mathcal{A}}(\kappa) \Rightarrow 1] \le \mathsf{Adv}^{\mathsf{SIS}}_{[n,q,\beta,m]}(\kappa) + 2^{-O(\kappa)}.$$

The correctness of tCH' follows directly from Lemma 5 (trapdoor delegation) and Lemma 4 (preimage sampling). Proofs of statistical indistinguishability and restricted collision resistance for tCH' are similar to those for tCH, and we provide them in our full version [26].

5 Application of tCH to the Redactable Blockchain

In this section, we show how to apply our tCH in constructing redactable blockchain. In Subsect. 5.1, we introduce some notations of a redactable blockchain. In Subsect. 5.2, we show how to redact a blockchain with a tCH. In Subsect. 5.3, we provide a security analysis of our redactable blockchain.

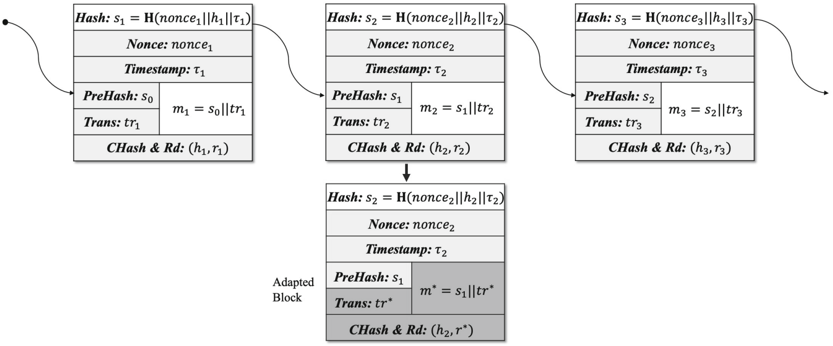


Fig. 6. An illustration of a redactable blockchain from tCH. All parts with light-grey background constitutes a block. Parts with white background are conceptual and shown for better presentation. Blocks link to a chain in a way that a previous hash value s_{i-1} for block B_{i-1} is stored in the "PreHash" part of block B_i. Take an adaptation from tr_2 to tr^* as an example, the corresponding hash-randomness pair (h_2, r_2) will be changed accordingly to (h_2, r^*), where r^* is computed by Adapt of tCH. The changed parts in the adapted block are decorated with dark-grey background. The down-arrow in dark-blue denotes the authorized adaptation done by the trusted regulation party. (Color figure online)

5.1 Redactable Blockchain

We follow notations of blockchain and redactable blockchain introduced in [3,18]. According to [3], a redactable block uses two hash functions, one is a cryptographic hash and the other is a chameleon hash. Now we replace the chameleon hash with our tCH, and additionally introduce a unique identifier (like the timestamp, previous hash or position of the block in the chain) into each block to serve as the "tag τ" of tCH. See Fig. 6 for a pictorial presentation.

Let $\mathsf{H} : \{0,1\}^* \to \mathbb{N}$ be a cryptographic hash function and $\mathsf{tCH} = (\mathsf{Setup}, \mathsf{Hash}, \mathsf{Adapt}, \mathsf{Check})$ be a tCH. In a redactable blockchain, each block B is of the form

$$B = \langle nonce, \tau, \underbrace{s, tr}_{m}, (h, r)\rangle,$$

where $nonce \in \mathbb{N}$ denotes the nonce value, $\tau \in \{0,1\}^t$ denotes a unique identifier, $s \in \mathbb{N}$ is a hash value computed by H, $tr \in \{0,1\}^x$ denotes the information stored in a block, (h, r) is a hash-randomness pair computed by Hash from $m := (s\|tr) \in \{0,1\}^h$ w.r.t. τ, i.e., $(h, r) \leftarrow \mathsf{Hash}(\tau, m)$. We say that a block B is valid if $\mathsf{ValidRB}_q^D(B) = 1$ with

$$\mathsf{ValidRB}_q^D(B) := \big(\mathsf{H}(nonce\|h\|\tau) < D\big) \wedge \big(\mathsf{Check}(\tau, h, m, r) = 1\big) \wedge \big(nonce < q\big),$$

where $D \in \mathbb{N}$ is the block's difficulty level and $q \in \mathbb{N}$ denotes the maximum allowed number of hash queries in a round.

```
Algorithm 1: Blockchain Redacting Algorithm
Input: The public parameter and trapdoor $(pp, td)$ of $\mathsf{tCH}$.
       A blockchain $\mathcal{C}$ with $\mathsf{len}(\mathcal{C}) = n$.
       A sequence of target indices $\mathcal{I} = (\iota_1, \ldots, \iota_k) \subseteq [n]$.
       A sequence of target messages $\mathcal{M} = (tr^*_{\iota_1}, \ldots, tr^*_{\iota_k})$.
Output: A redacted blockchain $\mathcal{C}'$ with $\mathsf{len}(\mathcal{C}') = n$
for $i = 1, \ldots, n$ do
    if $i \in \mathcal{I}$ then
        Parse the $i$-th block $B_i$ of $\mathcal{C}$ as $B_i = \langle nonce_i, \tau_i, s_i, tr_i, (h_i, r_i)\rangle$;
        Compute $r^*_i \leftarrow \mathsf{Adapt}(td, \tau_i, h_i, s_i \| tr_i, r_i, s_i \| tr^*_i)$;
        Set $B^*_i := \langle nonce_i, \tau_i, s_i, tr^*_i, (h_i, r^*_i)\rangle$;
        Set $\mathcal{C} := \mathcal{C}^{\lceil n-i+1} \parallel B^*_i \parallel {}^{i\rceil}\mathcal{C}$;
return $\mathcal{C}$
```

A redactable blockchain $\mathcal{C}$ is a sequence of valid blocks. The head of chain $\mathcal{C}$, denoted by $\mathsf{head}(\mathcal{C})$, is the rightmost block in it. The length of chain $\mathcal{C}$, denoted by $\mathsf{len}(\mathcal{C})$, is the number of blocks contained in it. Let $\mathcal{C} = \varepsilon$ if chain $\mathcal{C}$ is empty.

Any chain $\mathcal{C}'$ with head $\mathsf{head}(\mathcal{C}') = \langle nonce', \tau', s', tr', (h', r')\rangle$ can be extended to a longer one by appending a new valid block $B = \langle nonce, \tau, s, tr, (h, r)\rangle$ satisfying $s = \mathsf{H}(nonce' \| h' \| \tau')$, and then the head of the extended chain $\mathcal{C} = \mathcal{C}' \| B$ is changed to $\mathsf{head}(\mathcal{C}) = B$. In case $\mathcal{C}' = \varepsilon$, any valid block B can append to it.

For a chain $\mathcal{C}$ with length $\mathsf{len}(\mathcal{C}) = n$ and any nonnegative integer $k \leq n$, we denote by $\mathcal{C}^{\lceil k}$ the chain resulting from removing the k rightmost blocks of $\mathcal{C}$, and denote by ${}^{k\rceil}\mathcal{C}$ the chain resulting from removing the k leftmost blocks of $\mathcal{C}$.

5.2 Redacting Blocks

In this subsection, we provide a blockchain redacting algorithm to redact blocks. Let n, k be positive integers s.t. $k \leq n$. The algorithm takes as inputs the public parameter and trapdoor of a tagged chameleon hash tCH, a blockchain $\mathcal{C}$ of length n, k target indices that represent the positions of blocks in $\mathcal{C}$ to be redacted, and k corresponding adapted messages for blocks to be redacted, and finally returns a redacted blockchain $\mathcal{C}'$. The detailed description is given in Algorithm 1.

5.3 Security Analysis

In this subsection, we provide a security analysis for the resulting redactable blockchain given a tCH with r-CR security. Note that tCH works in the one-time tag mode in the redactable blockchain since the tCH hash value w.r.t. each settled block is computed with a unique tag and authorized adaptations are made only for those settled blocks. Then f-CR of tCH is equivalent to r-CR according to Theorem 1. Therefore, all we need to do is to prove that the redactable blockchain is secure as long as tCH has f-CR security.

As we described in Subsect. 5.1, each block B in the chain is of the form $B = \langle nonce, \tau, s, tr, (h, r)\rangle$. For expression simplicity, we only consider the tCH-related parts of each block and briefly write B as $B = \langle \tau, h, m, r\rangle$ (note that $m = s\|tr$). Recall that in a redactable blockchain system, the adversary sees all original blocks $B_1, B_2, B_3, \cdots$ and adapted blocks $\{B_1^i\}_{i\in[n_1]}, \{B_2^i\}_{i\in[n_2]}, \{B_3^i\}_{i\in[n_3]}, \cdots$, where each $n_j = \mathsf{poly}(\kappa)$ denotes the number of adaptations for block B_j. The aim of an adversary is to redact the chain by adapting some block $B_j = \langle \tau_j, h_j, m_j, r_j\rangle$ to a new one $B^* = \langle \tau_j, h_j, m^*, r^*\rangle$, such that $h_j = \mathsf{Hash}(\tau_j, m_j; r_j) = \mathsf{Hash}(\tau_j, m^*; r^*)$ and $m^* \notin \{m_j\} \cup \{m_j^i\}_{i\in[n_j]}$ w.r.t. those B_j and $\{B_j^i\}_{i\in[n_j]}$.

We show that if there exists an adversary $\mathcal{A}$ performing the above attack successfully, then we can break the full collision resistance of tCH. If $\mathcal{A}$ wins, it must hold that $h_j = \mathsf{Hash}(\tau_j, m_j; r_j) = \mathsf{Hash}(\tau_j, m^*; r^*)$ and m^* is fresh w.r.t. (τ_j, h_j). In this case, we find a tuple $(\tau_j, h_j, m^*, r^*, m_j, r_j)$ s.t. $h_j = \mathsf{Hash}(\tau_j, m_j; r_j) = \mathsf{Hash}(\tau_j, m^*; r^*)$ and (τ_j, h_j, m^*) is fresh, and hence break the f-CR of tCH.

Therefore, the security of the resulting redactable blockchain is reduced to the f-CR security of tCH. Given the equivalence of f-CR and r-CR in the scenario of redactable blockchain, we know that, the redactable blockchain is secure as long as the underlying tCH has r-CR security.

Finally with Theorems 2 and 3, both our tCHs in Subsect. 4.1 and Subsect. 4.2 can serve as secure compilers converting a conventional blockchain to a redactable one.

Acknowledgements. We would like to thank the reviewers for their valuable comments. The authors are partially sponsored by Guangdong Major Project of Basic and Applied Basic Research under Grant No. 2019B030302008, National Natural Science Foundation of China under Grant No. 61925207 and the National Key R&D Program of China under Grant No. 2022YFB2701500.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
2. Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. In: STACS, pp. 75–86 (2009)
3. Ateniese, G., Magri, B., Venturi, D., Andrade, E.R.: Redactable blockchain - or - rewriting history in bitcoin and friends. In: EuroS&P, pp. 111–126 (2017)
4. Ateniese, G., de Medeiros, B.: On the key exposure problem in chameleon hashes. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 165–179. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30598-9_12
5. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_42

6. Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30
7. Boyen, X., Li, Q.: Towards tightly secure lattice short signature and ID-based encryption. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10032, pp. 404–434. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_14
8. Brakerski, Z., Vaikuntanathan, V.: Lattice-based FHE as secure as PKE. In: ITCS (2014). https://doi.org/10.1145/2554797.2554799
9. Camenisch, J., Derler, D., Krenn, S., Pöhls, H.C., Samelin, K., Slamanig, D.: Chameleon-hashes with ephemeral trapdoors – and applications to invisible sanitizable signatures. In: Fehr, S. (ed.) PKC 2017. LNCS, vol. 10175, pp. 152–182. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54388-7_6
10. Chen, X., Tian, H., Zhang, F., Ding, Y.: Comments and improvements on key-exposure free chameleon hashing based on factoring. In: Lai, X., Yung, M., Lin, D. (eds.) Inscrypt 2010. LNCS, vol. 6584, pp. 415–426. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21518-6_29
11. Chen, X., Zhang, F., Kim, K.: Chameleon hashing without key exposure. In: Zhang, K., Zheng, Y. (eds.) ISC 2004. LNCS, vol. 3225, pp. 87–98. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30144-8_8
12. Derler, D., Krenn, S., Samelin, K., Slamanig, D.: Fully collision-resistant chameleon-hashes from simpler and post-quantum assumptions. In: Galdi, C., Kolesnikov, V. (eds.) SCN 2020. LNCS, vol. 12238, pp. 427–447. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57990-6_21
13. Derler, D., Samelin, K., Slamanig, D.: Bringing order to chaos: the case of collision-resistant chameleon-hashes. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020. LNCS, vol. 12110, pp. 462–492. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45374-9_16
14. Deuber, D., Magri, B., Thyagarajan, S.A.K.: Redactable blockchain in the permissionless setting. In: IEEE Symposium on Security and Privacy, pp. 124–138 (2019). https://doi.org/10.1109/SP.2019.00039
15. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 523–540. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_31
16. Dorri, A., Kanhere, S.S., Jurdak, R.: MOF-BC: a memory optimized and flexible blockchain for large scale networks. Future Gener. Comput. Syst. **92**, 357–373 (2019). https://doi.org/10.1016/J.FUTURE.2018.10.002
17. Florian, M., Henningsen, S.A., Beaucamp, S., Scheuermann, B.: Erasing data from blockchain nodes. In: EuroS&P. pp. 367–376 (2019). https://doi.org/10.1109/EUROSPW.2019.00047
18. Garay, J., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: analysis and applications. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 281–310. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_10
19. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC, pp. 197–206 (2008)

20. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_5
21. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: FOCS, pp. 464–479. IEEE Computer Society (1984)
22. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. (2) (1988). https://doi.org/10.1137/0217017
23. Hohenberger, S., Waters, B.: Short and stateless signatures from the RSA assumption. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 654–670. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_38
24. Khalili, M., Dakhilalian, M., Susilo, W.: Efficient chameleon hash functions in the enhanced collision resistant model. Inf. Sci. 155–164 (2020)
25. Krawczyk, H., Rabin, T.: Chameleon signatures. In: Proceedings of the Network and Distributed System Security Symposium. NDSS 2000, San Diego, California, USA (2000). https://www.ndss-symposium.org/ndss2000/chameleon-signatures/
26. Li, Y., Liu, S.: Tagged chameleon hash from lattice and application to redactable blockchain. ePrint (2023). https://eprint.iacr.org/2023/774
27. Matzutt, R., Kalde, B., Pennekamp, J., Drichel, A., Henze, M., Wehrle, K.: How to securely prune bitcoin's blockchain. In: IFIP Networking Conference, pp. 298–306 (2020). https://ieeexplore.ieee.org/document/9142720
28. 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, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41
29. 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, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_2
30. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: FOCS. IEEE Computer Society (2004). https://doi.org/10.1109/FOCS.2004.72
31. Pan, J., Wagner, B.: Short identity-based signatures with tight security from lattices. In: Cheon, J.H., Tillich, J.-P. (eds.) PQCrypto 2021 2021. LNCS, vol. 12841, pp. 360–379. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-81293-5_19
32. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-46766-1_9
33. Peikert, C.: Bonsai trees (or, arboriculture in lattice-based cryptography). IACR Cryptology ePrint Archive (2009). http://eprint.iacr.org/2009/359
34. Puddu, I., Dmitrienko, A., Capkun, S.: μchain: how to forget without hard forks. IACR Cryptology ePrint Archive, p. 106 (2017). http://eprint.iacr.org/2017/106
35. Pyoung, C.K., Baek, S.J.: Blockchain of finite-lifetime blocks with applications to edge-based IoT. IEEE Internet Things J. **7**(3), 2102–2116 (2020). https://doi.org/10.1109/JIOT.2019.2959599
36. Thyagarajan, S.A.K., Bhat, A., Magri, B., Tschudi, D., Kate, A.: Reparo: Publicly verifiable layer to repair blockchains. In: Borisov, N., Diaz, C. (eds.) FC 2021. LNCS, vol. 12675, pp. 37–56. Springer, Heidelberg (2021). https://doi.org/10.1007/978-3-662-64331-0_2

37. Tsabary, R.: Fully secure attribute-based encryption for t-CNF from LWE. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 62–85. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_3
38. Wu, C., Ke, L., Du, Y.: Quantum resistant key-exposure free chameleon hash and applications in redactable blockchain. Inf. Sci. 438–449 (2021)
39. Zhang, D., Le, J., Lei, X., Xiang, T., Liao, X.: Exploring the redaction mechanisms of mutable blockchains: a comprehensive survey. Int. J. Intell. Syst. **36**(9), 5051–5084 (2021). https://doi.org/10.1002/INT.22502