Adaptively-Secure Proxy Re-Encryption with Tight Security

Chen Qian [1,2,3,✉], Shuo Chen [1,2,3], and Shuai Han [4]

[1] State Key Laboratory of Cryptography and Digital Economy Security, Shandong University, Qingdao, 266237, China
[2] Key Laboratory of Cryptologic Technology and Information Security of Ministry of Education, Shandong University, Qingdao, 266237, China
[3] School of Cyber Science and Technology, Shandong University, Qingdao, 266237, China
[4] School of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, China

Abstract. (Bi-Directional) Proxy Re-Encryption (PRE) is a public-key encryption scheme that allows a proxy, holding a re-encryption key from i to j, to transform a ciphertext intended for i into one intended for j. PRE has numerous applications, including secure data sharing and cloud computing. However, most existing PRE schemes experience significant security degradation when adversaries are allowed to adaptively corrupt re-encryption or secret keys. Prior to this work, only a few PRE schemes achieved quasi-polynomial security loss in the adaptive setting, and even those were limited to restricted re-encryption strategies.

In this paper, we propose four distinct PRE schemes with tight security guarantees in the adaptive setting, based on the MDDH assumption:

- PRE_1, PRE_2: Single- and multi-challenge HRA-secure PRE schemes with tight security focusing on efficient constructions.
- PRE_3, PRE_4: Single- and multi-challenge CCA-secure PRE schemes with (almost) tight security focusing on CCA-type security.

To obtain (almost) tight security for CCA-secure PRE schemes, we introduce a novel concept called tag-based language-malleable NIZK with special simulation soundness. This primitive provides simulation-sound NIZK while preserving a restricted form of malleability. We construct both one-time and unbounded versions of this primitive under the MDDH (Matrix Decisional Diffie-Hellman) assumption.

Keywords. Proxy Re-Encryption, Adaptive Security, NIZK, Simulation-Soundness, Malleability

1 Introduction

Proxy re-encryption (PRE) was first introduced by Blaze, Bleumer, and Strauss in 1998 [BBS98] as a mechanism for securely transforming ciphertexts between different users. Since then, PRE has been extensively studied in the context of secure data sharing, cloud computing, and access control scenarios.

Unidirectional vs. Bidirectional. There are two different notions of PRE schemes, called unidirectional PRE [LV08] and bidirectional PRE [BBS98]. In a unidirectional PRE scheme, the re-encryption key d can only be used to transform ciphertexts from Alice to Bob, but not vice versa. However, for many applications requiring mutual conversion or reversible updates, bidirectional PRE suffices: the re-encryption key d can be used to transform ciphertexts in both directions. In fact, as we will explain, bidirectional proxy re-encryption already implies updatable encryption schemes and public-key encryption schemes with adaptive key corruption, both of which have many real-life applications. Moreover, relaxing unidirectionality to bidirectionality leads to much more efficient PRE schemes, as shown by Canetti and Hohenberger [CH07]. Therefore, we focus on bidirectional PRE schemes in this paper and refer to them simply as PRE schemes.

Proxy Re-Encryption, Updatable Encryption and Encryption with Corruption. There are numerous notions related to encryption schemes, including proxy re-encryption [BBS98], updatable encryption [BLMR13], and public-key encryption in the multi-user setting with adaptive key corruption [BHJ+15]. In fact, these notions are closely related. An updatable encryption scheme can be seen as a special case of a proxy re-encryption scheme by treating the update key as the re-encryption key from epoch i to $i+1$. Furthermore, if we disallow the adversary from querying re-encryption or update tokens, then proxy re-encryption schemes can be seen as public-key encryption schemes in a multi-user setting with adaptive key corruption, where the adversary can corrupt users' secret keys adaptively.

Security Model. The security of proxy re-encryption (PRE) schemes is less well understood than that of public-key encryption schemes. Since its introduction in [BBS98], various security models have been proposed, including the chosen plaintext attack (CPA) model [BBS98], the honest re-encryption attack (HRA) model [Coh19], and the chosen ciphertext attack (CCA) model [CH07]. Among these, the HRA model, a strengthened variant of the CPA model, has gained prominence as the most widely accepted security framework for PRE schemes. In this model, the adversary is allowed to request the re-encryption of any ciphertext generated by the encryption oracle. While CCA security is a stronger notion, HRA-secure PRE schemes have garnered significant attention in the literature [AFGH05,ABH09,HRsV07], particularly in lattice-based constructions [Gen09,ABPW13,FL19].

Adaptive Security. However, none of the above security models (CPA, HRA, CCA) is fully adaptive: the adversary is required to preselect the subset of users to be corrupted. This requirement greatly facilitates the construction of a PRE scheme. Without it, if there are $\mathsf{N_U}$ users, the challenger may need to guess the corrupted subset, incurring an exponential security loss of $\mathcal{O}(2^{\mathsf{N_U}})$. Several subsequent works have sought to reduce this unrealistic loss. Canetti and

Hohenberger [CH07] allow adaptive corruption of re-encryption tokens between uncorrupted parties[5]; Weng et al. [WCY+10] constructed a standard-model, CCA-secure, single-hop unidirectional PRE scheme in an adaptive corruption model. Jafargholi et al. [JKK+17] proposed a generic transformation based on *graph-pebbling game techniques* to achieve adaptive security in certain settings. Following this approach, Fuchsbauer et al. [FKKP19] gave an adaptively secure unidirectional PRE scheme with polynomial security loss under restrictions on the re-encryption structure.

We ask the following question:

Can we construct efficient adaptively {HRA, CCA}-secure bi-directional multi-hop PRE schemes with tight security?

1.1 Our Contribution

In this paper, we affirmatively answer the above question by proposing four adaptively secure proxy re-encryption (PRE) schemes. We compare our solution with existing schemes in Table 1.

Our contributions are both practical and theoretical.

- **Efficient Constructions.** We propose four novel PRE schemes.
  - We introduce PRE_1 and PRE_2, the most efficient single- and multi-challenge adaptively HRA-secure (aHRA) schemes to date.
  - We present PRE_3 and PRE_4, the first single- and multi-challenge adaptively CCA-secure (aCCA) schemes with tight or almost-tight reductions.

  All four schemes feature tight or almost-tight security reductions, with PRE_4 achieving an almost-tight reduction. Our aHRA-secure schemes PRE_1 and PRE_2 have ciphertexts consisting of only three group elements. In contrast, PRE_3 and PRE_4 target the stronger combination of security, at the cost of substantially larger ciphertexts. The additional overhead mainly comes from the language-malleable simulation-sound NIZK proof and its authentication components. We therefore view PRE_3 and PRE_4 as security-oriented constructions rather than schemes optimized for immediate deployment. Reducing this CCA overhead while preserving the same functionality and reduction tightness is an important open problem.
- **New Techniques.** We develop new proof techniques and cryptographic tools that are of independent interest.
  - To achieve adaptive security with a tight reduction, we introduce a novel proof strategy that confines the guessing argument to a transition between two perfectly indistinguishable hybrid games.[6] We formalize this

[5] As noted in their paper, the adversary must still choose the subset of corrupted users before the protocol starts.

[6] The underlying static-to-adaptive intuition is related to prior equivalence results in perfect and information-theoretic settings [CDD+04]; our novelty lies in localizing this reasoning to a single perfect hybrid hop inside an otherwise computational security proof.

technique as a S2A core lemma (Lemma 3), which can be extended to a statistical version. We expect this to serve as a general tool for proving tight adaptive security for other cryptographic primitives.

- To reconcile the inherent malleability of PRE with CCA security, we introduce a new type of non-interactive zero-knowledge (NIZK) proof system. Termed language-malleable special simulation-sound NIZK, this system permits a specific, controlled form of malleability required for re-encryption while maintaining simulation soundness against all other forms of tampering.[7] We present the high-level ideas in the main body, with detailed constructions and proofs provided in the appendix.

Scheme	Hop	Dir.	Security	Loss	Ct Size	Assumption	PRE?
[BBS98]	multi	Bi	s-MC-CPA	loose	$2\mathbb{G}$	DDH	✓
[CH07, Π_{RO}, Π]	multi	Bi	s-SC-CCA	loose	$\ell + \sigma + 4\mathbb{G}$	DBDH	✓
[WCY+10]	single	Uni	a-SC-CCA	loose	$\ell + \lvert\mathbb{Z}_p\rvert + 3\mathbb{G}$	3-wDBDHI	✓
[Coh19, 5.4]	single	Uni	s-MC-HRA	loose	$2\mathbb{G}$	DDH	✓
[ZLH24]	multi	Uni	a-SC-HRA	loose	$\mathcal{O}(\lambda\sqrt{\lambda})$	LWE	✓
[LCB+25]	single	Uni	a-MC-HRA	tight	$3\mathbb{G}'$	SD$\{1,2,3\}$	✓
[HLG23]	–	–	a-MC-CCA	almost	$18\mathbb{G}$	MDDH	×
PRE_1	multi	Bi	a-SC-HRA	tight	$3\mathbb{G}$	MDDH	✓
PRE_2	multi	Bi	a-MC-HRA	tight	$3\mathbb{G}$	MDDH	✓
PRE_3	multi	Bi	a-SC-CCA	tight	$\ell + \sigma + 24\mathbb{G}$	MDDH	✓
PRE_4	multi	Bi	a-MC-CCA	almost	$\ell + \sigma + 51\mathbb{G}$	MDDH	✓

Table 1. Comparison of our schemes with existing ones. Security format: [corruption]-[challenge]-[security notion], where corruption is s (selective) or a (adaptive), challenge is SC (single-challenge) or MC (multi-challenge), and security notion is CPA, HRA, or CCA. Here "almost" denotes almost tight with a loss $\mathcal{O}(\log \lambda)$. For schemes with two ciphertext levels, **Ct Size** refers to the size of the re-encryptable ciphertext. $\mathbb{G}$ represents group elements from $\mathbb{G}_1$ and $\mathbb{G}_2$, $\mathbb{G}'$ represents elements from composite-order groups, σ is the signature size, and ℓ is the bit-string length; all these sizes are $\mathcal{O}(\lambda)$. [HLG23] is not a PRE scheme, but we include it for comparison as it is the only work that achieves tight adaptive multi-user multi-challenge CCA security for standard PKE.

1.2 Related Work

Proxy re-encryption was first introduced in [BBS98] and has since been studied under various security models. Early schemes [AFGH05,CH07] achieved CPA or CCA security but with loose reductions and selective corruption models. The HRA model [Coh19] provides a practical security notion between CPA and CCA, which has been widely adopted in lattice-based constructions [Gen09,ABPW13,FL19].

[7] Our language-malleable NIZK is inspired by the controlled-malleable NIZK framework of Chase et al. [CKLM12], detailed in Related Work.

Achieving tight adaptive security is challenging due to the impossibility result in [BJLS16], which shows that tightly secure adaptive PKE schemes cannot have secret keys that are both unique for each public key and efficiently re-randomizable. Previous adaptive PRE schemes [CH07,FKKP19,WCY+10] either require selective user corruption or suffer from polynomial security loss. Hash proof systems [CS02] provide a way to bypass this barrier by allowing exponentially many secret keys per public key that are not efficiently re-randomizable.

Canetti et al. [CDD+04] studied the relation between static and adaptive security of multi-party protocols under several definitions and models. In particular, they obtained equivalence results under an augmented information-theoretic formulation. Our use of this principle differs in both scope and purpose. We do not establish a general equivalence between selective and adaptive PRE security. Instead, our overall proof remains computational: we isolate the adaptive guessing argument to one carefully constructed hybrid hop, prove that this particular hop has zero distinguishing gap, and then combine it with the surrounding computational reductions.

Recent works [LLP20,HLG23] achieved tight multi-challenge security for standard PKE using random oracles or complex proof systems. For PRE schemes, [LCB+25] is the first to achieve tight adaptive multi-challenge HRA security for unidirectional single-hop schemes in composite-order groups under the subgroup decision assumptions SD3 [LCB+25] and SD1/SD2 [LW10]. Our work achieves both HRA and CCA security for bidirectional multi-hop PRE in prime-order groups under the MDDH assumption, providing the first tight adaptive multi-challenge CCA-secure schemes and the most efficient HRA constructions.

Existing tag-based simulation-sound NIZKs [AJO+19] are incompatible with PRE because their strong simulation soundness prevents re-encryption of challenge ciphertexts. We introduce language-malleable NIZKs that allow controlled malleability for re-encryption while maintaining security against other attacks. The controlled-malleable NIZKs of Chase et al. [CKLM12] are the closest conceptual predecessor to our language-malleable proof systems. Their framework considers an efficient relation $\mathcal{R}$ together with an efficiently testable class of allowable transformations $\mathcal{T}$ under which $\mathcal{R}$ is closed, and uses the resulting controlled-malleable NIZKs to obtain controlled-malleable CCA-secure encryption.

This framework does not, however, yield our CCA-secure PRE construction as a black-box consequence. One may syntactically represent a language family by the union relation

$$R^{\cup}((\mathsf{lpk}, \mathsf{x}), \mathsf{w}) \coloneqq R_{\mathsf{lpk}}(\mathsf{x}, \mathsf{w}).$$

In the PRE setting, a valid transformation must additionally map $(\mathsf{lpk}_i, \mathsf{x}_i, \mathsf{w}_i)$ to $(\mathsf{lpk}_j, \mathsf{x}_j, \mathsf{w}_j)$ consistently with a legitimately generated re-encryption token. It must also preserve these relations under adaptive secret-key corruption and token queries. These validity and linkage conditions do not follow merely from closure of $R^{\cup}$ under a publicly specified transformation: without them, an adversary could choose a target language public key whose span contains the transformed statement, even when the statement is outside the intended user's language.

Our membership-preserving language family and link predicate therefore bind the source and target language public keys, statements, and token-induced transformation jointly.

1.3 Technique Overview

In this section, we provide an overview of the techniques employed in our schemes, emphasizing the advancements made over previous approaches. We give an abstract view of the principles underlying our constructions. To avoid unnecessary structural details, we give a high-level description using a concrete MDDH instantiation as an example.

Starting Point: From Adaptively Secure PKE to PRE. We begin by observing that, in the absence of re-encryption and token generation queries, an adaptively secure PRE scheme is equivalent to a multi-user public-key encryption (PKE) scheme secure against adaptive key corruptions. However, constructing such a PKE scheme with a tight security reduction must circumvent the impossibility result of Bader et al. [BJLS16]. Their result precludes tightly secure adaptively secure PKE schemes where secret keys are unique for each public key or are efficiently re-randomizable.

To overcome this barrier, we employ hash proof systems (HPS) [CS02]. In an HPS-based construction, each public key corresponds to an exponential number of secret keys, which are not efficiently re-randomizable. Assuming $\mathbf{A} \in \mathbb{Z}_{\mathsf{p}}^{\ell \times k}$ is full rank, $[\mathbf{A}]$, the group representation of $\mathbf{A}$, serves as the public parameter. For any $\mathbf{k} \in \mathbb{Z}_{\mathsf{p}}^{\ell}$, $\mathsf{w} \in \mathbb{Z}_{\mathsf{p}}^{k}$, and $\mathsf{m} \in \mathbb{Z}_{\mathsf{p}}$, the ciphertext is generated as follows:

$$[\mathbf{c}_0] = [\mathbf{A}\mathsf{w}], \qquad [\mathbf{c}_1] = [\mathbf{x}^\intercal \mathsf{w} + \mathsf{m}],$$

The key pair $([\mathbf{x}^\intercal] = [\mathbf{k}^\intercal \mathbf{A}], \mathbf{k})$ naturally exhibits the key homomorphism required for PRE. This structure allows us to bypass the impossibility result of [BJLS16] and serves as the foundation for our first scheme, PRE_1, which is single-challenge and tightly aHRA-secure.

A Tight Selective-to-Adaptive Security Reduction. A key challenge in proving tight adaptive security is to simulate oracle queries, in particular for key corruptions and re-encryption tokens, without resorting to a guessing argument that incurs a security loss. Our core technical insight is a novel proof strategy that confines the guessing argument to a single transition between two hybrid games that are otherwise perfectly indistinguishable.

Specifically, we leverage the structure of hash proof systems. For any honestly generated language public key $\mathbf{k}^\intercal [\mathbf{A}]$, the space of corresponding secret keys is defined by $\ker(\mathbf{A}^\intercal) := \{\mathbf{k}' \mid \mathbf{k}'^\top \mathbf{A} = \mathbf{0}\}$ as $\mathbf{k} + \ker(\mathbf{A}^\intercal)$, which is information-theoretically independent of the public key. Moreover, shifting all user secret keys by a random element from $\ker(\mathbf{A}^\intercal)$ is perfectly undetectable by the adversary. We apply this key-shifting technique in a single hybrid step to re-randomize

the challenge ciphertext while leaving all other values unchanged. This allows us to bound the advantage of an adaptive adversary by that of a selective one within this specific transition, thereby avoiding any security loss. We formalize this technique as a selective-to-adaptive (S2A) core lemma (Lemma 3), which we believe is a general tool for proving tight adaptive security for other cryptographic primitives.

From Single-Challenge to Multi-Challenge To achieve multi-challenge security, we present an algebraic interpretation of the multi-key extraction techniques from [HLG23]. Intuitively, when the left kernel of $\mathbf{A}$ has dimension at least k, the MDDH assumption implies that shifting all secret keys by a uniformly random element of $\ker(\mathbf{A}^\intercal)$ is computationally indistinguishable (with a tight reduction) from shifting them by a uniformly random vector in the entire space.[8] This k-dimensional randomness from the kernel space precisely matches the dimension of the underlying language $\mathcal{L}_{\mathsf{lpk}}$[9], which also has dimension k.

The key insight is to leverage the *language linearity* property of our subset membership language. Starting from a language $\mathcal{L}_{\mathsf{lpk}}$ over dimension ℓ, we can construct an isomorphic language $\mathcal{L}_{\mathsf{lpk}'}$ over dimension $\ell + n$ (where $n = 1$ for aHRA schemes and $n = 2$ for aCCA schemes) through a linear transformation matrix $\mathbf{M} = (\mathbf{I}_\ell, \mathbf{r}_1, \ldots, \mathbf{r}_n)^\intercal \in \mathbb{F}^{(\ell+n)\times\ell}$, where each $\mathbf{r}_i$ is sampled uniformly from the orthogonal complement of $\mathcal{L}_{\mathsf{lpk}_0}$, which is independent of $\mathcal{L}_{\mathsf{lpk}}$. This transformation preserves the language structure: $\mathbf{M}$ maps $\mathcal{L}_{\mathsf{lpk}}$ isomorphically onto $\mathcal{L}_{\mathsf{lpk}'}$. Crucially, the hard subset membership (HSM) assumption for the expanded language $\mathcal{L}_{\mathsf{lpk}'}$ provides additional randomness that can be used to simultaneously randomize multiple challenge ciphertexts. By exploiting this expanded randomness space while maintaining tight security through the language isomorphism, we successfully extend the single-challenge scheme PRE_1 and PRE_3 to the multi-challenge scheme PRE_2 and PRE_4 with preserved tight security.

Simulation-Sound Proofs. In our aHRA-secure encryption schemes, the security relies on the perfect hiding of information in the left kernel $\ker(\mathbf{A}^\intercal)$ from the adversary. For the multi-challenge setting, the language linearity property provides additional randomness from the expanded language dimension, which together with the kernel space information enables simultaneous protection of multiple challenge ciphertexts. However, this condition holds true only in the absence of a decryption oracle. If a decryption oracle is available, the adversary could query the decryption of $[\mathsf{ct}_0]_1 \notin \mathsf{Span}(\mathbf{A})$, thereby leaking information about the $\ker(\mathbf{A}^\intercal)$ element used in the secret key $\mathbf{k}$. To address this issue, we incorporate zero-knowledge proofs for linear span languages.

One common challenge in using the zero-knowledge proof to achieve CCA security is that, in the security proof of hash proof systems, the challenger needs

[8] For a formal argument under the $\mathsf{Q_C}$-MDDH assumption, one must also incorporate the ct_0 component.

[9] The formal definition of the subset membership language is given in Section 4.1

to intentionally put $\mathsf{ct_0}$ of the challenge ciphertext outside the $\mathsf{Span}(\mathbf{A})$, then use the hidden randomness in $\ker(\mathbf{A}^\intercal)$ to hide the plaintext message. With a zero-knowledge proof for linear span, the adversary then needs to simulate the proof for false statements. Therefore, we need the simulation soundness property of the underlying zero-knowledge proof to guarantee that the adversary cannot prove false statements even after seeing the false proof in the challenge ciphertext.

A similar issue arises in existing multi-challenge tightly secure encryption schemes [LLP20,HLG23]. These schemes address the problem either by employing the random oracle model or by utilizing existing tag-based simulation-sound quasi-adaptive zero-knowledge proofs [AJO+19]. However, in the context of constructing PRE schemes, the key homomorphism property must be preserved, making the random oracle approach unsuitable. Additionally, the existing tag-based simulation-sound quasi-adaptive zero-knowledge proofs [AJO+19] lack the key homomorphism property. This limitation is inherent, as these proofs are designed to achieve strong simulation soundness, which prevents adversaries from generating any new false proofs. In contrast, for PRE schemes, the re-encryption algorithm must allow the generation of certain new false proofs using the re-encryption token, to re-encrypt the challenge ciphertext using the re-encryption token. To address this, we relax the strong simulation-soundness requirement by introducing a notion of special simulation soundness. We then construct new tag-based language-malleable NIZK proofs for linear span languages that satisfy this special simulation-soundness property. Specifically, we propose two such proofs, $\mathbf{\Pi}_{\mathsf{OT}}$ (one-time) and $\mathbf{\Pi}_{\mathsf{U}}$ (unbounded), which are subsequently employed in the CCA-secure PRE schemes PRE_3 and PRE_4. As $\mathbf{\Pi}_{\mathsf{OT}}$ and $\mathbf{\Pi}_{\mathsf{U}}$ are the first NIZK proof systems to combine language malleability with special simulation soundness, we believe they are of independent interest. The intuition behind the constructions $\mathbf{\Pi}_{\mathsf{OT}}$ and $\mathbf{\Pi}_{\mathsf{U}}$ is that, instead of directly proving the statement $\mathsf{x} \in \mathcal{L}$, we prove the following statement:

$$\{\mathsf{x}, \mathbf{C_s} \mid \mathsf{x} \in \mathcal{L} \vee (\mathbf{C_s} = \mathsf{Commit}(\mathsf{ck}, \sigma) \wedge \mathsf{Ver}(\mathsf{svk}, \mathsf{x}', \sigma) = 1 \wedge \mathsf{x}' = \mathbf{T}(\mathsf{x}))\},$$

where $\mathbf{T}$ is an allowed transformation of the statement, and σ is a signature on x' under the key pair $(\mathsf{ssk}, \mathsf{svk})$. In the context of PRE schemes, $\mathbf{T}$ corresponds to the re-encryption operation. The commitment scheme ensures that the adversary cannot adaptively choose the transformation after seeing the proof of the challenge ciphertext. The signature scheme ensures that the adversary cannot generate new valid transformations without knowing the signing key. This design allows us to achieve language malleability while maintaining special simulation soundness.

aCCA-secure PRE Schemes. We present two distinct constructions of PRE schemes: PRE_3 with single-challenge security and PRE_4 with multi-challenge security. For clarity, we focus on explaining the construction of PRE_3 (single-challenge CCA) from PRE_1 (single-challenge aHRA), as the construction of PRE_4 (multi-challenge CCA) from PRE_2 (multi-challenge aHRA) follows a similar approach.

The ciphertext structure of PRE_1 can be represented as:

$$\left[\mathbf{c}_0\right] = \left[\mathbf{A}\mathsf{w}\right], \qquad \left[\mathbf{c}_1\right] = \left[\mathbf{x}^\intercal \mathsf{w} + \mathsf{m}\right],$$

where $[\mathbf{A}]$ and $[\mathbf{x}]$ form the public key, and $\mathbf{k}$ is the secret key satisfying $\mathbf{x}^\intercal = \mathbf{k}^\intercal \mathbf{A}$. To achieve aCCA security, we must ensure that the plaintext message m cannot be modified. However, proxy re-encryption requires changing $[\mathbf{x}]$, which affects $[\mathbf{c}_1]$. This creates a conflict between aCCA security and proxy re-encryption functionality. To resolve this conflict, we modify the ciphertext structure as follows:

$$[\mathbf{c}_0] = [\mathbf{A}\mathsf{w}], \qquad [\mathbf{c}_1] = [(\mathbf{k}_i^\intercal \mathbf{A} + \mathbf{x}^\intercal)\mathsf{w}], \qquad [\mathbf{c}_2] = [\mathbf{x}^\intercal \mathsf{w} + \mathsf{m}],$$

where $\mathbf{k}_i$ is the secret key of user i with public key $[\mathbf{k}_i^\intercal \mathbf{A}]$, and $[\mathbf{A}], [\mathbf{x}]$ are common public parameters.

In this refined structure, $[\mathbf{A}]$ and $[\mathbf{x}]$ remain invariant across all users, while only $\mathbf{k}_i$ varies between users. This design enables proxy re-encryption to modify solely $[\mathbf{c}_1]$. To guarantee non-malleability, we incorporate a signature scheme for $[\mathbf{c}_0]$ and $[\mathbf{c}_2]$, along with a NIZK proof to verify that $[\mathbf{c}_0] \in \mathsf{Span}(\mathbf{A})$ and $[\mathbf{c}_1]$ is correctly formed with respect to the same witness w. Given that $[\mathbf{c}_0]$ uniquely determines w, this NIZK proof ensures the non-malleability of $[\mathbf{c}_1]$ as well.

2 Notations

Group Presentation. We write $[x]_1 \in \mathbb{G}_1$, $[x]_2 \in \mathbb{G}_2$, and $[x]_\mathsf{T} \in \mathbb{G}_\mathsf{T}$ for elements of the source and target groups. Matrices are denoted by uppercase letters (e.g., $\mathbf{A}$), vectors by bold lowercase letters (e.g., $\mathbf{w}$), and scalars by lowercase letters (e.g., x). For a matrix $\mathbf{A}$, the notation $[\mathbf{A}]_1$ denotes its component-wise encoding in $\mathbb{G}_1$, and similarly for $[\mathbf{B}]_2$. The bilinear pairing is written as $[\mathbf{A}]_1 \bullet [\mathbf{B}]_2 = [\mathbf{AB}]_\mathsf{T}$. We assume $\mathbb{G}_1$ and $\mathbb{G}_2$ are prime-order groups of order p, write $\mathbb{Z}_\mathsf{p}$ for the field of order p, and set $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{T}, e)$ with a type-3 bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_\mathsf{T}$.
Pseudo-codes. The notation $x \leftarrow\!\!\$\, \mathsf{U}(\mathbb{Z}_\mathsf{p})$ denotes sampling x uniformly from $\mathbb{Z}_\mathsf{p}$. The operator **check** verifies a condition and aborts (returning 0) if it fails. We write **if** / **then** /**else** and **for** for pseudocode control flow.
Game-based Proofs. We write pr_i for the probability that the adversary wins in $\mathbf{Game}_i$, and use Adv for advantage when convenient. The security parameter is denoted by λ, and its unary encoding by 1^λ.

3 Proxy Re-Encryption

We follow the classical definition of (bi-directional multi-hop) proxy re-encryption from [CH07]. To better illustrate our security notions, we then define corruption graph which updates dynamically as the adversary makes queries. We also define the security notions of aHRA and aCCA for our proxy re-encryption scheme.

Definition 1 (Proxy Re-Encryption). *A (bi-directional multi-hop) proxy re-encryption scheme* PRE *consists of six* PPT *algorithms* $\mathsf{PRE} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{RKGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{REnc})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$: *Takes the security parameter* λ *as input and returns the public parameter* pp.
- $\mathsf{KGen}(\mathsf{pp}) \to (\mathsf{pk}, \mathsf{sk})$: *Takes the public parameter* pp *as input and returns a public key* pk *and a secret key* sk.
- $\mathsf{RKGen}(\mathsf{sk}_i, \mathsf{sk}_j) \to \mathsf{d}_{i,j}$: *Takes the secret keys of users* i *and* j *as input and returns a re-encryption token* $\mathsf{d}_{i,j}$.
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{m}) \to \mathsf{ct}$: *Takes a public key* pk *and a message* m *as input and returns a ciphertext* ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to \mathsf{m}$: *Takes a secret key* sk *and a ciphertext* ct *as input and returns the decrypted message* m.
- $\mathsf{REnc}(\mathsf{d}_{i,j}, \mathsf{pk}_i, \mathsf{pk}_j, \mathsf{ct}) \to \mathsf{ct}'$: *Takes a re-encryption token* $\mathsf{d}_{i,j}$, *public keys* pk_i, pk_j, *and a ciphertext* ct *as input, and returns a new ciphertext* ct'.

Additionally, we require the following correctness property:

Correctness: *We require the multi-hop correctness of the* PRE *scheme. Namely, it requires that after any* L *rounds of re-encryption, the ciphertext can be decrypted to the original plaintext.*
More formally, for any security parameter λ, *any message* m *and any integer* L, *we have*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}_L, \mathsf{ct}_L) = \mathsf{m} \;\middle|\; \begin{array}{l} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ (\mathsf{pk}_1, \mathsf{sk}_1) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp}) \\ \mathsf{ct}_1 \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_1, \mathsf{m}) \\ \textbf{for } i = 2 \textbf{ to } L \\ \quad (\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp}) \\ \quad \mathsf{d}_{i-1,i} \xleftarrow{\$} \mathsf{RKGen}(\mathsf{sk}_{i-1}, \mathsf{sk}_i) \\ \quad \mathsf{ct}_i \xleftarrow{\$} \mathsf{REnc}(\mathsf{d}_{i-1,i}, \mathsf{pk}_{i-1}, \mathsf{pk}_i, \mathsf{ct}_{i-1}) \end{array}\right] = 1.$$

3.1 Security Model

In this section, we define the security notions of aHRA and aCCA for our proxy re-encryption scheme. The security definitions are based on the winning probability of the adversary while adhering to the constraints imposed by the corruption graph defined in Section 3.2.

(Un)linkability and Link Function. Unlinkability informally requires that an adversary cannot distinguish whether a ciphertext under a target key was obtained by re-encrypting some ciphertext of the same message under another key, or by an independent fresh encryption of that message. We formalize this notion below.

Definition 2 (Unlinkability). *A proxy re-encryption scheme is said to be unlinkable if, for any ciphertext* ct_i, *the distribution of a re-encrypted ciphertext* ct_j *derived from* ct_i *is computationally, statistically, or perfectly indistinguishable from that of a fresh encryption of the same message. Formally, we have*

$$\{\mathsf{ct}_j \xleftarrow{\$} \mathsf{REnc}(\mathsf{d}_{i,j}, \mathsf{pk}_i, \mathsf{pk}_j, \mathsf{Enc}(\mathsf{pk}_i, \mathsf{m}))\} \approx_{\mathsf{X}} \{\mathsf{ct}_j \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_j, \mathsf{m})\},$$

with $\mathsf{X} \in \{$*computational, statistical, perfect*$\}$.

In the aCCA experiment the adversary additionally obtains a decryption oracle, so we must prevent trivial wins in which a challenge ciphertext (or any of its re-encryptions) is submitted for decryption. To enforce this we use a publicly computable predicate $\mathsf{link}(\mathsf{ct}_i, \mathsf{ct}_j)$ that outputs 1 if ct_j is a (possibly multi-hop) re-encryption of ct_i, and 0 otherwise. This predicate is only required for the aCCA game, since in the aHRA game the adversary has no decryption oracle, so identifying such links yields no immediate advantage.

The presence of a public predicate $\mathsf{link}(\cdot, \cdot)$ already rules out perfect unlinkability: anyone can test whether two ciphertexts are (multi-hop) related by evaluating $\mathsf{link}(\mathsf{ct}_i, \mathsf{ct}_j)$. (Computational or statistical unlinkability may still be meaningful.)

In particular, no aCCA-secure scheme can satisfy perfect unlinkability. Otherwise, a re-encryption of a challenge ciphertext would be perfectly indistinguishable from a fresh encryption, yet the adversary could re-encrypt the challenge to another honest user and submit the result to the decryption oracle. The experiment must either reject directly using link (as in our definition), or run Dec and then reject all decryption queries that decrypt to a challenge ciphertext [CH07]. This attack is admissible because the adversary has re-encryption tokens between honest users.

aHRA and aCCA Security. We define the adaptive security of our proxy re-encryption scheme in the multi-challenge setting, where the adversary can issue a polynomial number of challenge queries. The adversary is allowed to corrupt users and re-encrypt challenge ciphertexts, but it must adhere to the constraints imposed by the corruption graph.

Definition 3 (Security). *We define the security experiment* $\mathsf{Exp}^{\mathsf{X},\mathsf{Q_C}}_{\mathsf{PRE},\mathcal{A}}(\lambda)$ *with* $\mathsf{X} \in \{\mathsf{aHRA}, \mathsf{aCCA}\}$ *in Figure 1. A* PRE *scheme is* X*-secure if for every security parameter* λ*, every* PPT *adversary* $\mathcal{A}$*, and any polynomial number of challenge queries* $\mathsf{Q_C} = \mathsf{poly}(\lambda)$*, the adversary's advantage defined as follows is negligible. Formally, we have*

$$\mathsf{Adv}^{\mathsf{X},\mathsf{Q_C}}_{\mathsf{PRE},\mathcal{A}}(\lambda) = \left| \Pr\left[\mathsf{Exp}^{\mathsf{X},\mathsf{Q_C}}_{\mathsf{PRE},\mathcal{A}}(\lambda) = 1\right] - \frac{1}{2} \right| = \mathsf{negl}(\lambda).$$

We define the X *security in the multi-user multi-challenge setting with corruptions, the single-challenge setting* (SC-X) *is a special case of* X*-security when* $\mathsf{Q_C} = 1$.

Trivial Attacks. The trivial attacks are those the adversary can easily win the game without any significant effort (e.g., by corrupting the target user or re-encrypting the challenge ciphertext to a corrupted user). We formalize the trivial attacks as a predicate $\mathsf{NonTrivial}(\mathcal{L}_{\mathsf{CorrT}}, \mathcal{L}_{\mathsf{CorrK}}, \mathcal{L}_{\mathsf{C}})$ that checks whether the adversary's queries lead to a non-trivial attack, which is defined in Figure 2.

$\underline{\mathsf{Exp}^{\mathsf{X},\mathsf{Q_C}}_{\mathsf{PRE},\mathcal{A}}(1^\lambda)}$:
01 $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
02 $b \hookleftarrow \mathsf{U}(\{0,1\})$
03 $\mathsf{N_K} := 0;\ \mathcal{L}_{\mathsf{CorrT}}, \mathcal{L}_{\mathsf{CorrK}}, \mathcal{L}_{\mathsf{C}}, \mathcal{L}_{\mathsf{HC}} := \emptyset$
04 $b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{X}}}(\mathsf{pp})$
05 **return** $[\![b = b']\!] \wedge \mathsf{NonTrivial}(\mathcal{L}_{\mathsf{CorrT}}, \mathcal{L}_{\mathsf{CorrK}}, \mathcal{L}_{\mathsf{C}})$

Oracle $\mathcal{O}_{\mathsf{KGen}}()$:
06 $\mathsf{N_K} := \mathsf{N_K} + 1$
07 $(\mathsf{pk}_{\mathsf{N_K}}, \mathsf{sk}_{\mathsf{N_K}}) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$
08 **for** $i = 1$ **to** $\mathsf{N_K} - 1$
09 $\quad \mathsf{d}_{i,\mathsf{N_K}} \xleftarrow{\$} \mathsf{RKGen}(\mathsf{sk}_i, \mathsf{sk}_{\mathsf{N_K}})$
10 **return** $\mathsf{pk}_{\mathsf{N_K}}$

Oracle $\mathcal{O}_{\mathsf{CorrT}}(i,j)$:
11 $\mathcal{L}_{\mathsf{CorrT}} := \mathcal{L}_{\mathsf{CorrT}} \cup \{(i,j)\}$
12 **return** $\mathsf{d}_{i,j}$

Oracle $\mathcal{O}_{\mathsf{CorrK}}(i)$:
13 $\mathcal{L}_{\mathsf{CorrK}} := \mathcal{L}_{\mathsf{CorrK}} \cup \{i\}$
14 **return** sk_i

Oracle $\mathcal{O}_{\mathsf{Enc}}(i,\mathsf{m})$:
15 $\mathsf{ct} := \mathsf{Enc}(\mathsf{pk}_i, \mathsf{m})$
16 $\mathcal{L}_{\mathsf{HC}} := \mathcal{L}_{\mathsf{HC}} \cup \{(i,\mathsf{ct})\}$
17 **return** ct

Oracle $\mathcal{O}_{\mathsf{Dec}}(i,\mathsf{ct})$: // CCA
18 $\mathsf{m} := \mathsf{Dec}(\mathsf{sk}_i, \mathsf{ct})$
19 **if** $\exists (i^*, \mathsf{ct}_{i^*}) \in \mathcal{L}_{\mathsf{C}} : \mathsf{link}(\mathsf{ct}_{i^*}, \mathsf{ct}) = 1$
20 $\quad$ **return** $\perp$
21 **return** m

Oracle $\mathcal{O}_{\mathsf{Chall,b}}(i,\mathsf{m}_0,\mathsf{m}_1)$:
22 **if** $|\mathsf{m}_0| \neq |\mathsf{m}_1|$ **return** $\perp$
23 $\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, \mathsf{m}_b)$
24 $\mathcal{L}_{\mathsf{C}} := \mathcal{L}_{\mathsf{C}} \cup \{(i,\mathsf{ct})\}$
25 **return** ct

Oracle $\mathcal{O}_{\mathsf{ReEnc}}(i,j,\mathsf{ct})$:
26 **check** $(i,\mathsf{ct}) \in \mathcal{L}_{\mathsf{HC}}$ // HRA
27 $\mathsf{ct}' \xleftarrow{\$} \mathsf{REnc}(\mathsf{d}_{i,j}, \mathsf{pk}_i, \mathsf{pk}_j, \mathsf{ct})$
28 **if** $(i,\mathsf{ct}) \in \mathcal{L}_{\mathsf{C}}$ **then**
29 $\quad \mathcal{L}_{\mathsf{C}} := \mathcal{L}_{\mathsf{C}} \cup \{(j,\mathsf{ct}')\}$
30 $\mathcal{L}_{\mathsf{HC}} := \mathcal{L}_{\mathsf{HC}} \cup \{(j,\mathsf{ct}')\}$
31 **return** ct'

Fig. 1. Security experiment $\mathsf{Exp}^{\mathsf{X},\mathsf{Q_C}}_{\mathsf{PRE},\mathcal{A}}(1^\lambda)$ in the multi-challenge, multi-user setting. The adversary may adaptively issue any polynomial number of oracle queries. For the aHRA security, where $\mathsf{X} = \mathsf{aHRA}$ we provide $\mathcal{O}_{\mathsf{aHRA}} = (\mathcal{O}_{\mathsf{KGen}}, \mathcal{O}_{\mathsf{CorrT}}, \mathcal{O}_{\mathsf{CorrK}}, \mathcal{O}_{\mathsf{Enc}}, \mathcal{O}_{\mathsf{Chall,b}}, \mathcal{O}_{\mathsf{ReEnc}})$. For the aCCA security, where $\mathsf{X} = \mathsf{aCCA}$ we additionally give the decryption oracle, i.e., $\mathcal{O}_{\mathsf{aCCA}} = (\mathcal{O}_{\mathsf{KGen}}, \mathcal{O}_{\mathsf{CorrT}}, \mathcal{O}_{\mathsf{CorrK}}, \mathcal{O}_{\mathsf{Enc}}, \mathcal{O}_{\mathsf{Dec}}, \mathcal{O}_{\mathsf{Chall,b}}, \mathcal{O}_{\mathsf{ReEnc}})$. The parameter $\mathsf{Q_C}$ denotes the number of $\mathcal{O}_{\mathsf{Chall,b}}$ queries. The challenger implicitly maintains all key pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ and all re-encryption tokens $\mathsf{d}_{i,j}$.

Alg $\underline{\mathsf{NonTrivial}(\mathcal{L}_{\mathsf{CorrT}}, \mathcal{L}_{\mathsf{CorrK}}, \mathcal{L}_{\mathsf{C}})}$:
01 if $\exists (i^\star, ct_{i^\star}) \in \mathcal{L}_{\mathsf{C}} : \mathcal{O}_{\mathsf{Dec}}(i^\star, ct_{i^\star})$ is queried // CCA
02 $\quad$ **return** 0 // CCA
03 $(\mathsf{V}, \mathsf{E}) := \mathsf{makeG}(\mathcal{L}_{\mathsf{CorrK}}, \mathcal{L}_{\mathsf{CorrT}})$
04 **parse** $(\mathsf{V_H}, \mathsf{V_C}) =: \mathsf{V}$
05 **if** $\exists (i^\star, \mathsf{ct}_{i^\star}) \in \mathcal{L}_{\mathsf{C}} : i^\star \in \mathsf{V_C}$ **return** 0
06 **return** 1

Fig. 2. Description of NonTrivial algorithm, where $\mathsf{makeG}(\mathcal{L}_{\mathsf{CorrK}}, \mathcal{L}_{\mathsf{CorrT}})$ generates the corruption graph.

3.2 Corruption Model

Following Canetti and Hohenberger [CH07], we exclude trivial attacks by maintaining a corruption graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ built by the procedure makeG. The vertex set V is partitioned into honest users $\mathsf{V_H}$ and corrupted users $\mathsf{V_C}$. An undirected edge $e_{i,j}$ is added if (i) the adversary can derive one user's secret key from the

other via a corrupted re-encryption token, or (ii) a challenge ciphertext has been re-encrypted between users i and j. The graph is updated after each oracle query. We formalize this below.

Definition 4 (Corruption Graph). *A corruption graph is a tuple* $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ *with the following syntax:*

- $\mathsf{V} = \mathsf{V_H} \cup \mathsf{V_C}$ *is the set of vertices, partitioned into honest vertices* $\mathsf{V_H}$ *and corrupted vertices* $\mathsf{V_C}$.
- $\mathsf{E} \subseteq \mathsf{V} \times \mathsf{V}$ *is the set of edges, representing the relationships between users.*
- $\mathsf{V_{CL}} \subseteq \mathsf{V_H}$ *is the set of challenge vertices, which includes all vertices that have challenge ciphertexts and all vertices that are connected to challenge vertices.*

The initial corruption graph consists only of the honest vertices $\mathsf{V} = \mathsf{V_H}$ and $\mathsf{E} = \emptyset$, and it evolves as the adversary makes queries to the oracles. Each query may reveal new relationships between users, leading to the addition of edges in the graph and the partition of vertices V into $\mathsf{V_H}$ and $\mathsf{V_C}$. We update the corruption graph with the following rules:

- If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{CorrK}}(i)$ with $i \in \mathsf{V_H}$, then move i from $\mathsf{V_H}$ to $\mathsf{V_C}$.
- If $\mathcal{A}$ queries the token corruption query $\mathcal{O}_{\mathsf{CorrT}}(i, j)$, add edge $e_{i,j}$ to E.
- If $\mathcal{A}$ queries the re-encryption oracle $\mathcal{O}_{\mathsf{ReEnc}}(i, j, \mathsf{ct})$ with a challenge ciphertext $(i, \mathsf{ct}) \in \mathcal{L}_{\mathsf{C}}$, add edge $e_{i,j}$.[10]
- For any $e_{i,j}$ added, if either i or j is in $\mathsf{V_C}$, then move both i and j into $\mathsf{V_C}$.
- For any i added to $\mathsf{V_C}$, add all vertices j connected to i into $\mathsf{V_C}$.

We use the algorithm makeG that takes all internal states and produces the corruption graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ as output.

4 Hash Proof System

We need a hash proof system with additional homomorphic properties and universality to construct our proxy re-encryption scheme. Before giving the formal definition of the hash proof system that we need, we first introduce the underlying subset membership language distribution.

4.1 Subset Membership Language Distribution

In this section, we define the subset membership language distribution SMP that is used in the construction of the hash proof system. The language distribution is defined as follows:

Definition 5 (Subset Membership Language). *A subset membership language* $\mathcal{L}$ *consists of four* PPT *algorithms* $(\mathsf{Setup}, \mathsf{Sample}_{\mathcal{L}}, \mathsf{Sample}_{\mathcal{X}}, \mathsf{CheckSMP}_{\mathcal{L}})$ *with the following syntax:*

[10] We emphasize that we only add edge $e_{i,j}$ for re-encryption of challenge ciphertexts, not for arbitrary ciphertexts. This makes a difference between CPA-type security and HRA-type security.

- $\mathsf{Setup}(1^\lambda, \ell) \to (\mathsf{lpk}, \mathsf{td})$*: Takes a dimension* ℓ *and a security parameter* λ *as input, and outputs a public key* lpk *and a trapdoor* td*. We denote by* $\mathcal{L}_{\mathsf{lpk}} \subseteq \mathcal{X}$ *the* **NP***-language defined by* lpk*, and a relation* $\mathcal{R}_{\mathsf{lpk}} \subseteq \mathcal{L}_{\mathsf{lpk}} \times \mathcal{W}$*, where* $\mathcal{X}$ *is a space associated with* ℓ *and* $\mathcal{W}$ *is the witness space. The relation* $\mathcal{R}_{\mathsf{lpk}}$ *specifies the valid statement-witness pairs for* $\mathcal{L}_{\mathsf{lpk}}$*.*
- $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}) \to (\mathsf{x}, \mathsf{w})$*: Takes the public key* lpk *as input, outputs a uniformly random statement* $\mathsf{x} \hookleftarrow \mathsf{U}(\mathcal{L}_{\mathsf{lpk}})$ *and a witness* w *such that* $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}_{\mathsf{lpk}}$*.*
- $\mathsf{Sample}_{\mathcal{X}}(1^\lambda) \to \mathsf{x}$*: Takes the security parameter* 1^λ *as input, and outputs a uniformly random statement* $\mathsf{x} \hookleftarrow \mathsf{U}(\mathcal{X})$ *from the ambient space.*
- $\mathsf{CheckSMP}_{\mathcal{L}}(\mathsf{lpk}, \mathsf{td}, \mathsf{x}) \to \{0, 1\}$*: Takes the public key* lpk*, trapdoor* td*, and a statement* x *as input, and outputs* 0 *or* 1 *depending on whether* x *is in the language* $\mathcal{L}_{\mathsf{lpk}}$ *defined by* lpk*.*

We require the following properties:

Correctness: *For all security parameters* λ*, all* $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \ell)$ *generated by the setup algorithm, and all statements* $\mathsf{x} \in \mathcal{X}$*, membership* $\mathsf{x} \in \mathcal{L}_{\mathsf{lpk}}$ *is equivalent to* $\mathsf{CheckSMP}_{\mathcal{L}}(\mathsf{lpk}, \mathsf{td}, \mathsf{x}) = 1$*. Namely,*

$$\mathsf{x} \in \mathcal{L}_{\mathsf{lpk}} \iff \mathsf{CheckSMP}_{\mathcal{L}}(\mathsf{lpk}, \mathsf{td}, \mathsf{x}) = 1.$$

Hard Subset Membership: *For any security parameter* λ*, any* $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \ell)$*, the distribution of statements* x *sampled from* $\mathcal{L}_{\mathsf{lpk}}$ *is computationally indistinguishable from the distribution of statements sampled from* $\mathcal{X}$*. Formally, for any* PPT *adversary* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathsf{HSM}}_{\mathsf{SMP},\mathcal{A}}(1^\lambda) = \left| \Pr\left[b' = b \;\middle|\; \begin{array}{l} (\mathsf{x}_0, \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}) \\ \mathsf{x}_1 \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda) \\ b \xleftarrow{\$} \{0,1\};\ b' \xleftarrow{\$} \mathcal{A}(\mathsf{x}_b, \mathsf{lpk}) \end{array} \right] - \frac{1}{2} \right| = \mathsf{negl}(\lambda).$$

Linear Homomorphism: *For any* $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \ell)$*, the statement set* $\mathcal{X}$ *forms a vector space over a finite field* $\mathbb{F}$*, and the language* $\mathcal{L}_{\mathsf{lpk}} \subset \mathcal{X}$ *is a subspace of* $\mathcal{X}$*. Moreover, we require that for all* $\alpha \in \mathbb{F}$*, and* $\mathsf{x}_1, \mathsf{x}_2 \in \mathcal{L}_{\mathsf{lpk}}$*, the following holds:*

$$(\mathsf{x}_1, \mathsf{w}_1) \in \mathcal{R}_{\mathsf{lpk}} \wedge (\mathsf{x}_2, \mathsf{w}_2) \in \mathcal{R}_{\mathsf{lpk}} \implies (\alpha \mathsf{x}_1 + \mathsf{x}_2, \alpha \mathsf{w}_1 + \mathsf{w}_2) \in \mathcal{R}_{\mathsf{lpk}}.$$

Language linearity: *For any security parameter* λ*, let* $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \ell)$ *and* $(\mathsf{lpk}', \mathsf{td}') \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \ell + n)$*, and let* $\mathcal{L}_{\mathsf{lpk}} \subseteq \mathcal{X}$*,* $\mathcal{L}_{\mathsf{lpk}'} \subseteq \mathcal{X}'$ *be the corresponding languages with* $\dim(\mathcal{L}_{\mathsf{lpk}}) = \dim(\mathcal{L}_{\mathsf{lpk}'}) = k$ *where* $k = \dim(\mathcal{W})$*. We require that there exists a linear map* $\mathbf{M} : \mathcal{X} \to \mathcal{X}'$ *of the form* $\mathbf{M} = (\mathbf{I}_\ell, \mathbf{r}_1, \ldots, \mathbf{r}_n)^\intercal \in \mathbb{F}^{(\ell+n)\times\ell}$*, where each* $\mathbf{r}_i \in \mathbb{F}^\ell$ *is sampled uniformly from a* d*-dimensional linear subspace that is independent of* lpk*, where* $d \geq k$*, such that the image of* $\mathcal{L}_{\mathsf{lpk}}$ *under* $\mathbf{M}$ *is statistically indistinguishable from the language produced by* $\mathsf{Setup}(1^\lambda, \ell + n)$*. That is,*

$$\mathbf{M}(\mathcal{L}_{\mathsf{lpk}}) \approx_s \mathcal{L}_{\mathsf{lpk}'}, \qquad (\mathsf{x}, \mathsf{w}) \in \mathcal{R}_{\mathsf{lpk}} \iff (\mathbf{M} \cdot \mathsf{x}, \mathsf{w}) \in \mathcal{R}_{\mathsf{lpk}'}.$$

4.2 Vector Space Hash Proof System

Let SMP be the subset membership language distribution of Definition 5, and $\mathcal{L}_{\mathsf{lpk}}$ be the language indexed by public key lpk. We now define the associated hash proof system VSHPS for $\mathcal{L}_{\mathsf{lpk}}$.

Definition 6 (Vector Space Hash Proof System). *A vector space hash proof system consists of four* PPT *algorithms* $\mathsf{VSHPS} = (\mathsf{Setup}, \mathsf{PKGen}, \mathsf{PEval}, \mathsf{SEval})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{hk}$*: Takes a security parameter* λ *as input, and outputs a hash key* hk*.*
- $\mathsf{PKGen}(\mathsf{hk}, \mathcal{L}_{\mathsf{lpk}}) \to \mathsf{projk}$*: Takes a hash key* hk *and a language* $\mathcal{L}_{\mathsf{lpk}}$ *as input, and outputs a projection key* projk*.*
- $\mathsf{PEval}(\mathsf{projk}, \mathsf{x}, \mathsf{w}) \to \mathsf{hv}$*: Takes a projection key* projk*, a statement* x*, and a witness* w *for* x*, and outputs a hash value* hv*.*
- $\mathsf{SEval}(\mathsf{hk}, \mathsf{x}) \to \mathsf{hv}$*: Takes a hash key* hk *and a statement* x *as input, and outputs a hash value* hv*.*

Note that $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$*, where* lpk *is the public parameter that defines the language* $\mathcal{L}_{\mathsf{lpk}}$*. We denote by* $\mathcal{R}_{\mathsf{lpk}}$ *the relation associated with* $\mathcal{L}_{\mathsf{lpk}}$*, which specifies the valid statement-witness pairs for* $\mathcal{L}_{\mathsf{lpk}}$*.*

We require the following properties:

Correctness: *For all security parameters* λ*, all* $\mathsf{hk} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, and all* (x, w) *such that* w *is a valid witness for* x*, it holds that*

$$\Pr[\mathsf{PEval}(\mathsf{PKGen}(\mathsf{hk}, \mathcal{L}_{\mathsf{lpk}}), \mathsf{x}, \mathsf{w}) = \mathsf{SEval}(\mathsf{hk}, \mathsf{x})] = 1.$$

Vector Space Structure: *Let* $\mathbb{F}$ *denote a finite field. The sets of hash keys* $\mathcal{HK}$*, projection keys* $\mathcal{PK}_{\mathsf{lpk}}$ *(which are determined by* lpk*), statements* $\mathcal{X}$*, and witnesses* $\mathcal{W}$ *each form an* $\mathbb{F}$*-vector space. The hash value space is* $\mathcal{HV} = \mathbb{F}$*. We also require that* $\dim(\mathcal{X}) = \dim(\mathcal{HK})$*.*

Homomorphism: *Due to the vector space structure, we require the following homomorphism*

- Projection-Homomorphism: *The projection keys satisfy the homomorphism property: For all security parameters* λ*, all* $\alpha \in \mathbb{F}$*, all* $\mathsf{hk}_1 \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ *and* $\mathsf{hk}_2 \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, it holds that*

$$\mathsf{PKGen}(\alpha \cdot \mathsf{hk}_1 + \mathsf{hk}_2, \mathcal{L}_{\mathsf{lpk}}) = \alpha \cdot \mathsf{PKGen}(\mathsf{hk}_1, \mathcal{L}_{\mathsf{lpk}}) + \mathsf{PKGen}(\mathsf{hk}_2, \mathcal{L}_{\mathsf{lpk}}).$$

- Message-Homomorphism: *For all security parameters* λ*, all* $\alpha \in \mathbb{F}$*, all* $\mathsf{hk} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, and all statements* $\mathsf{x}_1, \mathsf{x}_2 \in \mathcal{X}$*, it holds that*

$$\mathsf{SEval}(\mathsf{hk}, \alpha \cdot \mathsf{x}_1 + \mathsf{x}_2) = \alpha \cdot \mathsf{SEval}(\mathsf{hk}, \mathsf{x}_1) + \mathsf{SEval}(\mathsf{hk}, \mathsf{x}_2)$$

- Key-Homomorphism: *For all security parameters* λ*, all* $\alpha \in \mathbb{F}$*, all* $\mathsf{hk}_1 \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ *and* $\mathsf{hk}_2 \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, and all statements* $\mathsf{x} \in \mathcal{X}$*, it holds that*

$$\mathsf{SEval}(\alpha \cdot \mathsf{hk}_1 + \mathsf{hk}_2, \mathsf{x}) = \alpha \cdot \mathsf{SEval}(\mathsf{hk}_1, \mathsf{x}) + \mathsf{SEval}(\mathsf{hk}_2, \mathsf{x}).$$

Key Collision: *Given the language parameter* lpk*, for two independently sampled hash keys* $\mathsf{hk}_1 \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ *and* $\mathsf{hk}_2 \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, the probability that they produce the same projection key is statistically negligible. Formally, we have*

$$\mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{A}}(1^\lambda) := \Pr\left[\mathsf{projk}_1 = \mathsf{projk}_2 \;\middle|\; \begin{array}{l}\mathsf{hk}_1, \mathsf{hk}_2 \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ \mathsf{projk}_1 = \mathsf{PKGen}(\mathsf{hk}_1, \mathcal{L}_{\mathsf{lpk}}) \\ \mathsf{projk}_2 = \mathsf{PKGen}(\mathsf{hk}_2, \mathcal{L}_{\mathsf{lpk}})\end{array}\right] = \epsilon.$$

(ε, d)**-Universality:** *For any security parameter* λ*, let* $\mathsf{lpk} \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$*,* $\mathsf{hk} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, and* $\mathsf{projk} := \mathsf{PKGen}(\mathsf{hk}, \mathcal{L}_{\mathsf{lpk}})$*. Fix the projection key* projk*, the set of hash keys consistent with* projk *is defined as*

$$\mathcal{HK}' := \{\mathsf{hk}' \in \mathcal{HK} \mid \mathsf{PKGen}(\mathsf{hk}', \mathcal{L}_{\mathsf{lpk}}) = \mathsf{projk}\}.$$

We require that

- $\mathcal{HK}' \subseteq \mathcal{HK}$ *forms an affine* $\mathbb{F}$*-vector space of dimension* d*. We denote by* $\mathcal{HK}''$ *the direction space associated with* $\mathcal{HK}'$*:*

$$\mathcal{HK}'' := \{\mathsf{hk}'' \in \mathcal{HK} \mid \mathsf{PKGen}(\mathsf{hk}'', \mathcal{L}_{\mathsf{lpk}}) = 0\}.$$

- *Let* x *be a uniformly random statement from* $\mathcal{X}$*, then the probability that* $\mathsf{SEval}(\mathsf{hk}'', \mathsf{x}) = 0$ *for all* $\mathsf{hk}'' \in \mathcal{HK}''$ *is negligible. Formally, we have*

$$\mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS}}(1^\lambda) := \Pr\left[\forall \mathsf{hk}'' \in \mathcal{HK}'' : \mathsf{SEval}(\mathsf{hk}'', \mathsf{x}) = 0 \;\middle|\; \mathsf{x} \xleftarrow{\$} \mathsf{U}(\mathcal{X})\right] \leq \varepsilon.$$

d**-MinEntropy** *Following the definition of* (ε, d)*-universality, let* $\mathcal{HK}''$ *be the* d*-dimensional vector space corresponding to the affine space* $\mathcal{HK}'$*. For any set of statements* $\{\mathsf{x}_1, \ldots, \mathsf{x}_n\} \subseteq \mathcal{X}$ *(where* $n \geq d$*) such that the linear span of their projections onto the dual space of* $\mathcal{HK}''$ *has dimension exactly* d*, i.e.,*

$$\dim(\mathit{span}\{\langle \mathsf{x}_i, \mathsf{hk}'' \rangle : \mathsf{hk}'' \in \mathcal{HK}'', i = 1, \ldots, n\}) = d,$$

the distribution of hash values

$$(\mathsf{SEval}(\mathsf{hk}'', \mathsf{x}_1), \ldots, \mathsf{SEval}(\mathsf{hk}'', \mathsf{x}_n)), \quad \mathsf{hk}'' \xleftarrow{\$} \mathcal{HK}''$$

has min-entropy exactly $d \log_2 |\mathbb{F}|$*, due to the fact that only* d *out of the* n *hash values are truly independent.*

4.3 Instantiation with MDDH assumption

We give the instantiation of the subset membership language and the vector space hash proof system based on the MDDH assumption in bilinear groups.

Subset Membership Language Let $\mathbb{G}$ be a prime order group of order p, and let g be a generator of $\mathbb{G}$. We define a language public key as $\mathbf{A} \in \mathbb{G}^{\ell \times k}$, where k is an arbitrary integer and $\ell \geq k + 1$ varies depending on the properties we need. More specifically, for an (ε, d)-universal VSHPS instantiation, we will need $d = \ell - k \geq 1$. We give the full description of the subset membership language in Figure 3, where $(\mathbb{G}, \mathsf{p}, \mathsf{g}) \xleftarrow{\$} \mathsf{GGen}(1^\lambda)$ generates a group in which DDH is hard. The properties of this construction are established as follows.

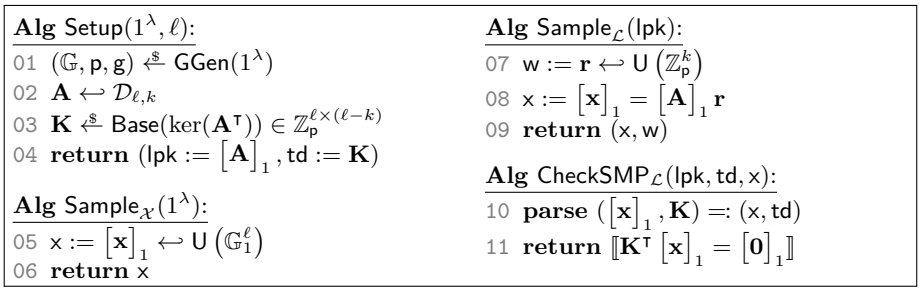

Fig. 3. Instantiation of the subset membership language $\mathcal{L}_{\mathsf{lpk}}$ based on the MDDH assumption in bilinear groups. $\mathsf{Base}(\ker(\mathbf{A}^\intercal))$ denotes the probabilistic algorithm that returns a basis for the kernel of the matrix $\mathbf{A}^\intercal$.

Correctness The correctness of the membership check follows from the definition of the language and the trapdoor. For any statement $\mathsf{x} = [\mathbf{A}]_1 \mathbf{r}$ in the language $\mathcal{L}_{\mathsf{lpk}}$, we have $\mathbf{K}^\intercal \mathsf{x} = \mathbf{K}^\intercal([\mathbf{A}]_1 \mathbf{r}) = \mathbf{0}$, since $\mathbf{K}$ is a basis for the left kernel of $\mathbf{A}$.

Hard Subset Membership The indistinguishability of a random statement $\mathsf{x} \in \mathbb{G}^\ell$ from a statement in the language $\mathcal{L}_{\mathsf{lpk}}$ is guaranteed by the Matrix Decisional Diffie-Hellman (MDDH) assumption. Moreover, due to the random self-reducibility [EHK+13], we have

$$\mathsf{Adv}^{\mathsf{MDDH},\mathsf{Q_C}}_{\mathbb{G},\mathcal{A}}(1^\lambda) = \mathsf{Adv}^{\mathsf{MDDH}}_{\mathbb{G},\mathcal{B}}(1^\lambda) + \mathsf{negl}(\lambda).$$

Linear Homomorphism The language is a k-dimensional vector space over $\mathbb{Z}_p$, and thus is linearly homomorphic by construction.

Language linearity This property is established as follows: Let $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \ell)$, and independently $(\mathsf{lpk}', \mathsf{td}') \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \ell + n)$. Write $\mathsf{lpk} = [\mathbf{A}]_1$ and $\mathsf{lpk}' = [\mathbf{A}']_1$, where $\mathbf{A} \in \mathbb{Z}_\mathsf{p}^{\ell \times k}$ and $\mathbf{A}' \in \mathbb{Z}_\mathsf{p}^{(\ell+n) \times k}$ are full-rank, so that $\mathcal{L}_{\mathsf{lpk}} = \{[\mathbf{A}]_1 \mathbf{w} : \mathbf{w} \in \mathbb{Z}_\mathsf{p}^k\}$ and $\mathcal{L}_{\mathsf{lpk}'} = \{[\mathbf{A}']_1 \mathbf{w} : \mathbf{w} \in \mathbb{Z}_\mathsf{p}^k\}$. The linear transformation matrix $\mathbf{M} = (\mathbf{I}_\ell, \mathbf{r}_1, \ldots, \mathbf{r}_n)^\intercal \in \mathbb{F}^{(\ell+n) \times \ell}$, where each $\mathbf{r}_i \in \mathbb{Z}_\mathsf{p}^\ell$ is sampled uniformly from a independent subspace with at least k dimension, defines a map $\mathbf{M} : \mathcal{X} \to \mathcal{X}'$ whose restriction is an isomorphism from $\mathcal{L}_{\mathsf{lpk}}$ onto the language output by $\mathsf{Setup}(1^\lambda, \ell + n)$, i.e., $\mathbf{M}(\mathcal{L}_{\mathsf{lpk}}) \approx_s \mathcal{L}_{\mathsf{lpk}'}$. Specifically, for any $\mathsf{x} = [\mathbf{A}]_1 \mathbf{w} \in \mathcal{L}_{\mathsf{lpk}}$, we have

$$\mathbf{M} \cdot \mathsf{x} = (\mathbf{I}_\ell, \mathbf{r}_1, \ldots, \mathbf{r}_n)^\intercal [\mathbf{A}]_1 \mathbf{w} \approx_s [\mathbf{A}']_1 \mathbf{w} \in \mathcal{L}_{\mathsf{lpk}'}$$

Vector Space Hash Proof System With respect to the subset membership language defined in Figure 3, we give the instantiation of the vector space hash proof system in Figure 4 based on the MDDH assumption. The properties of this construction are established as follows.

Correctness The correctness follows from the definitions of PEval and SEval. For any statement $\mathsf{x} \in \mathcal{L}_{\mathsf{lpk}}$ with witness w, we have $\mathsf{PEval}(\mathsf{projk}, \mathsf{x}, \mathsf{w}) = (\mathbf{k}^\intercal [\mathbf{A}]_1) \cdot \mathsf{w} = \mathbf{k}^\intercal \cdot ([\mathbf{A}]_1 \mathsf{w}) = \mathsf{SEval}(\mathsf{hk}, \mathsf{x})$.

Alg $\mathsf{Setup}(1^\lambda)$:
01 $\mathsf{hk} := \mathbf{k} \leftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^\ell\right)$
02 **return** hk

Alg $\mathsf{PKGen}(\mathsf{hk}, \mathcal{L}_\mathsf{lpk})$:
03 **parse** $\mathbf{k} =: \mathsf{hk}$
04 $\mathsf{projk} := \mathbf{k}^\intercal \left[\mathbf{A}\right]_1$
05 **return** projk

Alg $\mathsf{PEval}(\mathsf{projk}, \mathsf{x}, \mathsf{w})$:
06 **parse** $\mathbf{k}^\intercal \left[\mathbf{A}\right]_1 =: \mathsf{projk}$; $\mathbf{w} =: \mathsf{w}$
07 **return** $\mathsf{hv} := \left(\mathbf{k}^\intercal \left[\mathbf{A}\right]_1\right) \cdot \mathbf{w}$

Alg $\mathsf{SEval}(\mathsf{hk}, \mathsf{x})$:
08 **parse** $\mathbf{k} =: \mathsf{hk}$; $\left[\mathbf{x}\right]_1 =: \mathsf{x}$
09 **return** $\mathsf{hv} := \mathbf{k}^\intercal \left[\mathbf{x}\right]_1$

Fig. 4. Construction of VSHPS with respect to the language $\mathcal{L}_\mathsf{lpk} = \mathsf{Span}(\mathbf{A})$.

Vector Space Structure Let $\mathbb{F} = \mathbb{Z}_\mathsf{p}$, the vector space structure is preserved in the projection, as the projection of a linear combination of vectors is the same linear combination of their projections.

Homomorphism The projection operation is a homomorphism, meaning it respects the linear structure of the space.

Key Collision Resistance For distinct secret keys $\mathbf{k}, \mathbf{k}' \in \mathbb{Z}_\mathsf{p}^\ell$, a collision $\mathbf{k}^\intercal \left[\mathbf{A}\right]_1 = \mathbf{k}'^\intercal \left[\mathbf{A}\right]_1$ implies $\mathbf{k} - \mathbf{k}'$ lies in the left kernel of $\left[\mathbf{A}\right]_1$. Since $\mathbf{k} - \mathbf{k}'$ is non-zero and uniform, this happens with probability at most $1/\left|\mathbb{F}\right|^k$, which is negligible.

(ε, d)-Universality Fix a projection key projk, the set of consistent hash keys is

$$\mathcal{HK}' := \{\mathbf{k}' \in \mathbb{Z}_\mathsf{p}^\ell \mid \mathbf{k}'^\intercal \left[\mathbf{A}\right]_1 = \mathsf{projk}\}.$$

This set is an affine $\mathbb{F}$-space: if $\mathbf{k} \in \mathcal{HK}'$, then $\mathcal{HK}' = \mathbf{k} + \mathcal{HK}''$ where the direction space is the left kernel

$$\mathcal{HK}'' = \{\Delta \in \mathbb{Z}_\mathsf{p}^\ell \mid \Delta^\intercal \left[\mathbf{A}\right]_1 = 0\} = \ker(\left[\mathbf{A}\right]_1^\intercal).$$

Hence $\dim(\mathcal{HK}'') = d = \ell - k$. For any statement $\mathsf{x} \leftarrow\!\!\shortmid \mathsf{U}(\mathcal{X})$, we have

$$\Pr[\forall \mathsf{hk}'' \in \mathcal{HK}'' : \mathsf{SEval}(\mathsf{hk}'', \mathsf{x}) = 0] = \left|\mathbb{F}\right|^{-d},$$

so $\varepsilon = \left|\mathbb{F}\right|^{-d}$ and the construction is (ε, d)-universal.

d-MinEntropy We have $\mathsf{SEval}(\mathsf{hk}'', \mathsf{x}) = \mathsf{hk}'' \cdot \mathsf{x}$ where $\mathsf{hk}'' \xleftarrow{\$} \mathsf{U}\left(\mathcal{HK}''\right)$ and $\mathcal{HK}''$ is d-dimensional. For any n statements $\{\mathsf{x}_1, \ldots, \mathsf{x}_n\}$ (where $n \geq d$) with exactly d linearly independent among them, the hash values $(\mathsf{hk}'' \cdot \mathsf{x}_1, \ldots, \mathsf{hk}'' \cdot \mathsf{x}_n)$ have min-entropy $d \log_2 \left|\mathbb{F}\right|$ because the d-dimensional hk'' can extract d independent values through inner products.

5 aHRA-secure PRE with Tight Security

In this section, we present tightly aHRA-secure proxy re-encryption (PRE) schemes. Our constructions are based on the key-homomorphic properties of hash proof systems (HPS). Specifically, we adapt the HPS from the Cramer-Shoup cryptosystem [CS02] to build a PRE scheme. A limitation of this HPS is its single-use

nature for elements outside the language, which inherently restricts the resulting PRE scheme to single-challenge security. To achieve multi-challenge security, we incorporate a ciphertext randomization technique from [HLG23].

This approach yields two constructions: a single-challenge scheme, PRE_1, and a multi-challenge scheme, PRE_2. Both can be described within a unified framework, differing only in the underlying HPS. For PRE_1, we use a $(\varepsilon, 1)$-universal VSHPS. For PRE_2, we employ a (ε, k)-universal VSHPS over a linear language of dimension k.

5.1 Generic Construction from a Vector Space HPS

We begin with a generic construction of an aHRA-secure PRE scheme from a vector space hash proof system (VSHPS). The construction is detailed in Figure 5.

Alg $\mathsf{Setup}(1^\lambda)$:
01 $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$
02 **return** $\mathsf{pp} := \mathsf{lpk}$

Alg $\mathsf{KGen}(\mathsf{pp})$:
03 $\mathsf{hk} \xleftarrow{\$} \mathsf{VSHPS.Setup}(1^\lambda)$
04 $\mathsf{projk} := \mathsf{PKGen}(\mathsf{hk}, \mathcal{L}_{\mathsf{lpk}})$
05 $\mathsf{pk} := \mathsf{projk}$; $\mathsf{sk} := \mathsf{hk}$
06 **return** $(\mathsf{pk}, \mathsf{sk})$

Alg $\mathsf{Enc}(\mathsf{pk}, [\mathsf{m}])$:
07 $(\mathsf{x}, \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk})$
08 $\mathsf{ct}_0 := \mathsf{x}$
09 $\mathsf{ct}_1 := \mathsf{PEval}(\mathsf{pk}, \mathsf{x}, \mathsf{w}) + \mathsf{m}$
10 **return** $\mathsf{ct} := (\mathsf{ct}_0, \mathsf{ct}_1)$

Alg $\mathsf{RKGen}(\mathsf{sk}_i, \mathsf{sk}_j)$:
11 **parse** $\mathsf{hk}_i =: \mathsf{sk}_i$; $\mathsf{hk}_j =: \mathsf{sk}_j$
12 $\mathsf{d}_{i,j} := \mathsf{hk}_j - \mathsf{hk}_i$
13 **return** $\mathsf{d}_{i,j}$

Alg $\mathsf{REnc}(\mathsf{d}_{i,j}, \mathsf{pk}_i, \mathsf{pk}_j, \mathsf{ct}_i)$:
14 **parse** $(\mathsf{ct}_{i,0}, \mathsf{ct}_{i,1}) =: \mathsf{ct}_i$; $\mathsf{x}_i =: \mathsf{ct}_{i,0}$
15 $(\mathsf{x}', \mathsf{w}') \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk})$
16 $\mathsf{ct}_0 := \mathsf{x}_i + \mathsf{x}'$
17 $\mathsf{ct}_1 := \mathsf{ct}_{i,1} + \mathsf{SEval}(\mathsf{d}_{i,j}, \mathsf{x}_i) + \mathsf{PEval}(\mathsf{pk}_j, \mathsf{x}', \mathsf{w}')$
18 **return** $\mathsf{ct} := (\mathsf{ct}_0, \mathsf{ct}_1)$

Alg $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$:
19 $\mathsf{m} := \mathsf{ct}_1 - \mathsf{SEval}(\mathsf{sk}, \mathsf{ct}_0)$
20 **return** m

Fig. 5. Construction of aHRA-secure proxy re-encryption PRE with tight security from vector space hash proof system (VSHPS).

The correctness of the construction in Figure 5 is established by verifying the decryption of both original and re-encrypted ciphertexts. This relies on the correctness and linear properties of the underlying hash proof system (VSHPS).

Correctness of Decryption. For a fresh ciphertext $\mathsf{ct} := \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$, where $\mathsf{ct} = (\mathsf{x}, \mathsf{PEval}(\mathsf{pk}, \mathsf{x}, \mathsf{w}) + \mathsf{m})$, decryption with the corresponding secret key sk works as follows:

$$\begin{aligned}\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) &= \mathsf{ct}_1 - \mathsf{SEval}(\mathsf{sk}, \mathsf{ct}_0) \\ &= (\mathsf{PEval}(\mathsf{pk}, \mathsf{x}, \mathsf{w}) + \mathsf{m}) - \mathsf{SEval}(\mathsf{sk}, \mathsf{x}) = \mathsf{m}.\end{aligned}$$

Correctness of Re-encryption. Let ct_i be a ciphertext under key pk_i, and let $\mathsf{ct}_j := \mathsf{REnc}(\mathsf{d}_{i,j}, \mathsf{pk}_i, \mathsf{pk}_j, \mathsf{ct}_i)$ be the re-encrypted ciphertext. Decrypting ct_j

with secret key sk_j yields:

$$\begin{aligned}\mathsf{Dec}(\mathsf{sk}_j, \mathsf{ct}_j) &= \mathsf{ct}_{j,1} - \mathsf{SEval}(\mathsf{sk}_j, \mathsf{ct}_{j,0}) \\ &= \mathsf{ct}_{i,1} + \mathsf{SEval}(\mathsf{d}_{i,j}, \mathsf{x}_i) + \mathsf{PEval}(\mathsf{pk}_j, \mathsf{x}', \mathsf{w}') - \mathsf{SEval}(\mathsf{sk}_j, \mathsf{x}_i + \mathsf{x}') \\ &= \mathsf{m} + \mathsf{SEval}(\mathsf{hk}_i, \mathsf{x}_i) + \mathsf{SEval}(\mathsf{hk}_j - \mathsf{hk}_i, \mathsf{x}_i) + \mathsf{SEval}(\mathsf{hk}_j, \mathsf{x}') \\ &\quad - \mathsf{SEval}(\mathsf{hk}_j, \mathsf{x}_i) - \mathsf{SEval}(\mathsf{hk}_j, \mathsf{x}') = \mathsf{m}.\end{aligned}$$

In Section 5.2 and Section 5.3, we prove that the construction from Figure 5 achieves tight aHRA security in both single- and multi-challenge settings, depending on the universality of the underlying VSHPS. Specifically:

- An $(\varepsilon, 1)$-universal VSHPS yields a scheme with tight single-challenge aHRA security.
- An (ε, d)-universal VSHPS over a k-dimensional linear language, where $d \geq k$, yields a scheme with tight multi-challenge aHRA security.

5.2 Single-challenge aHRA security

Theorem 1. *Let* VSHPS *be a* $(\varepsilon, 1)$*-universal vector space hash proof system over a linear language of dimension* k *with* $d = 1$*, the proxy re-encryption scheme* PRE *described in Figure 5 is* SC-aHRA *secure. Formally, we have:*

$$\mathsf{Adv}^{\mathsf{SC\text{-}aHRA}}_{\mathsf{PRE},\mathcal{A}}(\lambda) \leq \binom{\mathsf{N_U}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{HSM}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).$$

$\mathsf{Exp}^{\mathsf{SC\text{-}aHRA}}_{\mathsf{PRE},\mathcal{A}}(1^\lambda)$:
Proof. 01 $v \hookleftarrow \mathsf{U}(\mathbb{F})$ // $\mathbf{G}_5$
02 $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
03 $b \hookleftarrow \mathsf{U}(\{0,1\})$
04 $\mathcal{O}_{\mathsf{aHra}} := (\mathcal{O}_{\mathsf{KGen}}, \mathcal{O}_{\mathsf{CorrT}}, \mathcal{O}_{\mathsf{CorrK}}, \mathcal{O}_{\mathsf{Enc}}, \mathcal{O}_{\mathsf{Chall,b}}, \mathcal{O}_{\mathsf{ReEnc}})$
05 **if** $\mathsf{Bad}_{\mathsf{Col}}$ **then** // $\mathbf{G}_1$-$\mathbf{G}_5$
06 **abort** // $\mathbf{G}_1$-$\mathbf{G}_5$
07 $b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{aHra}}}(\mathsf{pp})$
08 **return** $[\![b = b']\!] \wedge \mathsf{NonTrivial}(\mathcal{L}_{\mathsf{CorrT}}, \mathcal{L}_{\mathsf{CorrK}}, \mathcal{L}_{\mathsf{C}})$

Alg $\mathcal{O}_{\mathsf{Chall,b}}(i, \mathsf{m}_0, \mathsf{m}_1)$:
09 $(\mathsf{ct}_0, \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk})$ // $\mathbf{G}_0$-$\mathbf{G}_2$
10 $\mathsf{ct}_0 \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda)$ // $\mathbf{G}_3$-$\mathbf{G}_5$
11 $\mathsf{ct}_1 := \mathsf{PEval}(\mathsf{projk}_i, \mathsf{ct}_0, \mathsf{w}) + \mathsf{m}_b$ // $\mathbf{G}_0$-$\mathbf{G}_1$
12 $\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}_i, \mathsf{ct}_0) + \mathsf{m}_b$ // $\mathbf{G}_2$-$\mathbf{G}_3$
13 **if** $\mathsf{Bad}_{\mathsf{Univ}}$ **then abort** // $\mathbf{G}_4$-$\mathbf{G}_5$
14 $\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}_i, \mathsf{ct}_0) + \mathsf{m}_b + v \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0)$ // $\mathbf{G}_5$
15 $\mathsf{ct} := (\mathsf{ct}_0, \mathsf{ct}_1)$
16 $\mathcal{L}_{\mathsf{C}} := \mathcal{L}_{\mathsf{C}} \cup \{(i, \mathsf{ct})\}$
17 **return** ct

Fig. 6. Hybrid games for SC-aHRA security. We denote by $\mathsf{h} \in \mathcal{HK}$ a basis vector for the kernel space corresponding to the k-dimensional linear language over $\mathbb{F}^\ell$. Such a vector can be generated together with the linear language.

We prove the above theorem via the following hybrid games $\mathbf{G}_0, \ldots, \mathbf{G}_5$. The detailed security games are presented in Figure 6.

Game $\mathbf{G}_0$: This is the original SC-aHRA security game. By definition, the adversary's success probability is

$$\mathsf{pr}_0 \coloneqq \Pr\Big[\mathsf{Exp}^{\mathsf{SC\text{-}aHRA}}_{\mathsf{PRE},\mathcal{A}}(1^\lambda) = 1\Big].$$

Game $\mathbf{G}_1$: This game is identical to $\mathbf{G}_0$ except for an abort on key collisions. Let $\mathsf{Bad}_{\mathsf{Col}}$ be the event that there exist distinct $i, j \in [\mathsf{N}_\mathsf{U}]$ such that $\mathsf{pk}_i = \mathsf{pk}_j$; the challenger aborts if $\mathsf{Bad}_{\mathsf{Col}}$ occurs.

The difference in the adversary's success probability between these two games is bounded by $\Pr[\mathsf{Bad}_{\mathsf{Col}}]$, which is negligible because key collisions in the underlying VSHPS occur with statistically small probability. We formally have:

$$|\mathsf{pr}_0 - \mathsf{pr}_1| \leq \Pr[\mathsf{Bad}_{\mathsf{Col}}] \leq \binom{\mathsf{N}_\mathsf{U}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_2$: The only difference between $\mathbf{G}_2$ and $\mathbf{G}_1$ is that, the challenge ciphertext is generated using the secret key hk instead of the projection key projk. We generate the challenge ciphertext as

$$(\mathsf{ct}_0, \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}) \qquad\qquad \mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}, \mathsf{ct}_0) + \mathsf{m}_b$$

Since $\mathsf{ct}_0 \in \mathcal{L}_{\mathsf{lpk}}$ with witness w, by the perfect correctness of HPS, the challenge ciphertext in $\mathbf{G}_2$ and $\mathbf{G}_1$ is identical. Therefore, we have

$$\mathsf{pr}_1 = \mathsf{pr}_2.$$

Game $\mathbf{G}_3$: In this game, we change the way the challenge ciphertext is generated. Instead of sampling ct_0 from the language using $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk})$, we sample it from the statement set $\mathsf{ct}_0 \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda)$. Since the bad event $\mathsf{Bad}_{\mathsf{Col}}$ does not occur, the adversary cannot simply check if $\mathsf{ct}_0 \in \mathcal{L}_{\mathsf{lpk}}$ by checking if $\mathsf{SEval}(\mathsf{hk}_i - \mathsf{hk}_j, \mathsf{ct}_0) = 0$ where i, j are any two collision users. [11] By the hard subset membership property of the language, we have

$$|\mathsf{pr}_2 - \mathsf{pr}_3| \leq \mathsf{Adv}^{\mathsf{HSM}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_4$: In $\mathbf{G}_4$, we introduce a bad event $\mathsf{Bad}_{\mathsf{Univ}}$ that captures the failure of the universal hash property. Specifically, we define

$$\mathsf{Bad}_{\mathsf{Univ}} \coloneqq \{\forall \mathsf{hk}'' \in \mathcal{HK}'' : \mathsf{SEval}(\mathsf{hk}'', \mathsf{x}) = 0 \mid \mathsf{ct}_0 \xleftarrow{\$} \mathsf{U}(\mathcal{X})\}.$$

Using the universality of VSHPS, we have

$$|\mathsf{pr}_3 - \mathsf{pr}_4| \leq \Pr[\mathsf{Bad}_{\mathsf{Univ}}] \leq \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).$$

[11] If there is a collision between two public keys pk_i and pk_j, then the adversary can simply obtain $\mathsf{hk}_i - \mathsf{hk}_j \in \mathcal{HK}''$ by querying the oracle $\mathcal{O}_{\mathsf{CorrT}}(i, j)$, and check if $\mathsf{SEval}(\mathsf{hk}_i - \mathsf{hk}_j, \mathsf{ct}_0) = 0$. If yes, then $\mathsf{ct}_0 \in \mathcal{L}_{\mathsf{lpk}}$; otherwise, $\mathsf{ct}_0 \notin \mathcal{L}_{\mathsf{lpk}}$.

Game $\mathbf{G}_5$: In this game, we modify the generation of the challenge ciphertext. We sample $v \leftarrow \mathsf{U}(\mathbb{F})$ at the beginning of the experiment and compute the challenge ciphertext as $\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}_i + v \cdot \mathsf{h}, \mathsf{ct}_0) + \mathsf{m}_b = \mathsf{SEval}(\mathsf{hk}_i, \mathsf{ct}_0) + \mathsf{m}_b + v \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0)$, where i indexes the challenge user and h is a basis of the $\mathcal{HK}''$ with dimension $d = 1$ over the linear language $\mathcal{L}_{\mathsf{lpk}}$.

We claim that $\mathbf{G}_4$ and $\mathbf{G}_5$ are perfectly indistinguishable, which is formalized in Lemma 1. Assuming the claim holds, we have

$$|\mathsf{pr}_5 - \mathsf{pr}_4| = 0$$

In $\mathbf{G}_5$, since the game does not abort, we are guaranteed that $\mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) \neq 0$. As v is chosen uniformly at random from $\mathbb{F}$, the term $v \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0)$ acts as a one-time pad, perfectly masking the message m_b. Consequently, the adversary's advantage is zero, and its success probability is exactly $1/2$. Thus, we have

$$\mathsf{pr}_5 = \frac{1}{2}.$$

Taking union bound over all the hybrid games, we complete the proof. □

Lemma 1. *Games $\boldsymbol{G}_4$ and $\boldsymbol{G}_5$ are perfectly indistinguishable. That is, for any* PPT *adaptive adversary $\mathcal{A}$, we have*

$$\left|\Pr\left[\boldsymbol{G}_5^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\boldsymbol{G}_4^{\mathcal{A}} \Rightarrow 1\right]\right| = 0,$$

where $\Pr\left[\boldsymbol{G}_i^{\mathcal{A}} \Rightarrow 1\right]$ *denotes the probability that adversary $\mathcal{A}$ outputs* 1 *in $\boldsymbol{G}_i$.*

Proof. We prove the lemma via a reduction from adaptive to selective security. The argument is structured in two parts, formalized in Lemma 2 and Lemma 3.

Lemma 2 establishes that no PPT selective adversary can distinguish between $\mathbf{G}_4$ and $\mathbf{G}_5$ with any advantage. Then, by Lemma 3, the same conclusion extends to every PPT adaptive adversary, which completes the proof.

□

Lemma 2. *For any* PPT *selective adversary $\mathcal{B}$, which by definition needs to commit to all its oracle queries before the game begins, we have:*

$$\left|\Pr\left[\boldsymbol{G}_5^{\mathcal{B}} \Rightarrow 1\right] - \Pr\left[\boldsymbol{G}_4^{\mathcal{B}} \Rightarrow 1\right]\right| = 0.$$

Proof. We prove the lemma by introducing three intermediate games, $\mathbf{G}_{4.1}$, $\mathbf{G}_{4.2}$ and $\mathbf{G}_{4.3}$, and show that for any selective adversary $\mathcal{B}$, the following chain of perfect indistinguishability holds: $\mathbf{G}_4 \approx_p \mathbf{G}_{4.1} \approx_p \mathbf{G}_{4.2} \approx_p \mathbf{G}_{4.3} \approx_p \mathbf{G}_5$. This implies that $\mathbf{G}_4$ and $\mathbf{G}_5$ are perfectly indistinguishable for $\mathcal{B}$.

The simulation strategy for these games, detailed in Figure 7, leverages the selective nature of $\mathcal{B}$. Without loss of generality, we re-index users such that all challenge queries in $\mathcal{L}_{\mathsf{C}}$ are for users $i \in \{1, \ldots, |\mathsf{V}_{\mathsf{CL}}|\}$.

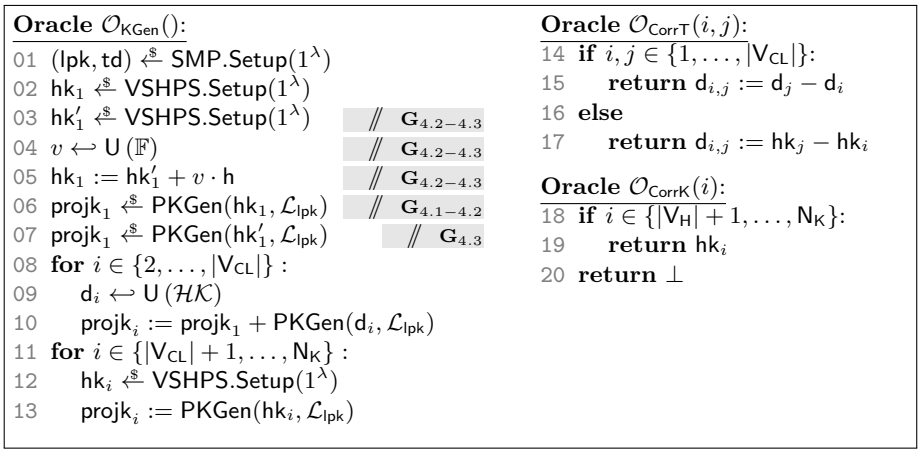

Fig. 7. Description of the hybrid game $\mathbf{G}_{4.1}$ for a selective adversary. The simulation of the oracles is based on the adversary's queries ($\mathcal{L}_{\mathsf{CorrT}}$, $\mathcal{L}_{\mathsf{CorrK}}$, $\mathcal{L}_{\mathsf{HC}}$, and $\mathcal{L}_{\mathsf{C}}$), which are known before the game begins. Without loss of generality, we re-index keys such that challenge queries in $\mathcal{L}_{\mathsf{C}}$ correspond to the first $|\mathsf{V_{CL}}|$ users.

Game $\mathbf{G}_{4.1}$: As detailed in Figure 7, this game is identical to $\mathbf{G}_4$ but uses a different simulation. The challenger first samples hk_1 and projk_1 for user 1, then samples d_i for each other challenged user $i \in \{2, \ldots, |\mathsf{V_{CL}}|\}$ and sets $\mathsf{hk}_i := \mathsf{hk}_1 + \mathsf{d}_i$ and $\mathsf{projk}_i := \mathsf{projk}_1 + \mathsf{PKGen}(\mathsf{d}_i, \mathcal{L}_{\mathsf{lpk}})$. For non-challenged users $j > |\mathsf{V_{CL}}|$, the challenger samples $(\mathsf{hk}_j, \mathsf{projk}_j)$ normally. Since d_i is uniformly random in $\mathcal{HK}$ and independent of hk_1, the distribution of hk_i remains uniform over $\mathcal{HK}$ for $i \in \{2, \ldots, |\mathsf{V_{CL}}|\}$. Thus, we have:

$$\mathsf{pr}_4 = \mathsf{pr}_{4.1}.$$

Game $\mathbf{G}_{4.2}$: This game modifies the generation of hk_1. Let $\mathsf{h} \in \mathcal{HK}''$ be a hash key such that $\mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) \neq 0$ for the challenge statement ct_0. Such an h is guaranteed to exist by the $(\varepsilon, 1)$-universality of the VSHPS. The challenger now computes $\mathsf{hk}_1 := \mathsf{hk}'_1 + v \cdot \mathsf{h}$, where $\mathsf{hk}'_1 \xleftarrow{\$} \mathcal{HK}$ and $v \hookleftarrow \mathsf{U}(\mathbb{F})$.

Since $\mathcal{HK}$ is a vector space over $\mathbb{F}$, the distribution of hk_1 remains uniform. Thus, the adversary's view is identical to that in $\mathbf{G}_{4.1}$, and we have

$$\mathsf{pr}_{4.1} = \mathsf{pr}_{4.2}.$$

Game $\mathbf{G}_{4.3}$: In this game, the challenger answers all oracle queries except $\mathcal{O}_{\mathsf{Chall}}$ using hk'_1 instead of hk_1 and generates the projection key projk_1 with hk'_1.

By the definition of $\mathcal{HK}''$, we have $\mathsf{PKGen}(\mathsf{hk}_1, \mathcal{L}_{\mathsf{lpk}}) = \mathsf{PKGen}(\mathsf{hk}'_1, \mathcal{L}_{\mathsf{lpk}})$, thus all the public keys and $\mathcal{O}_{\mathsf{Enc}}$ responses are unchanged. Since the adversary cannot query $\mathcal{O}_{\mathsf{CorrK}}(i)$ for user $i \in [|\mathsf{V_{CL}}|]$, the responses of $\mathcal{O}_{\mathsf{CorrK}}$ are identical. When answering $\mathcal{O}_{\mathsf{CorrT}}(i, j)$ queries for challenged users $i, j \in [|\mathsf{V_{CL}}|]$, we return

$\mathsf{d}_{i,j} = \mathsf{d}_j - \mathsf{d}_i$, assuming that $\mathsf{d}_1 = 0$. For non-challenged users $i, j > |\mathsf{V_{CL}}|$, we answer $\mathcal{O}_{\mathsf{CorrT}}(i,j)$ queries using their known secret keys; We note that $\mathcal{O}_{\mathsf{CorrT}}(i,j)$ for $i \in [|\mathsf{V_{CL}}|]$ and $j > |\mathsf{V_{CL}}|$ is not allowed. Thus, the answers of $\mathcal{O}_{\mathsf{CorrT}}$ are unchanged.

As for $\mathcal{O}_{\mathsf{ReEnc}}(i,j)$: if $i, j \in [|\mathsf{V_{CL}}|]$, we compute the re-encryption ciphertexts using $\mathsf{d}_{i,j} = \mathsf{d}_j - \mathsf{d}_i$; if $i \in [|\mathsf{V_{CL}}|]$ but $j > |\mathsf{V_{CL}}|$, the answer is unchanged because $\mathsf{SEval}(\mathsf{hk}_j - \mathsf{hk}_i, \mathsf{ct}_0) = \mathsf{SEval}(\mathsf{hk}_j - \mathsf{hk}'_i, \mathsf{ct}_0)$ for $(i, (\mathsf{ct}_0, \mathsf{ct}_1)) \in \mathcal{L}_{\mathsf{HC}}$;[12] If $i, j > |\mathsf{V_{CL}}|$, we compute the ciphertexts using their secret keys, which are also independent of both hk_1 and hk'_1. Thus, the answers of $\mathcal{O}_{\mathsf{ReEnc}}$ are unchanged.

Therefore, the adversary's view in $\mathbf{G}_{4.2}$ and $\mathbf{G}_{4.3}$ is identical, leading to

$$\mathsf{pr}_{4.2} = \mathsf{pr}_{4.3}.$$

By the changes in $\mathbf{G}_{4.3}$, hk'_1 is now the true secret key for user 1 in the simulation.

Since hk_1 and hk'_1 are identically distributed, the only difference between $\mathbf{G}_{4.3}$ and $\mathbf{G}_5$ lies in the generation of secret keys for users $i \in \{2, \cdots, |\mathsf{V_{CL}}|\}$. In $\mathbf{G}_{4.3}$, these keys are implicitly defined as $\mathsf{hk}_i = \mathsf{hk}'_1 + \mathsf{d}_i$, while in $\mathbf{G}_5$, they are sampled uniformly from $\mathcal{HK}$. By the same argument as in $\mathbf{G}_{4.1}$, the distribution of hk_i for $i \in \{2, \cdots, |\mathsf{V_{CL}}|\}$ is the same in both games, and we have:

$$\mathsf{pr}_{4.3} = \mathsf{pr}_5.$$

Combining the above results, we complete the proof. □

Lemma 3 (S2A - Core Lemma). *For any two games $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$, if for any* PPT *selective adversary $\mathcal{B}$ we have*

$$\left|\Pr\left[\boldsymbol{G}_2^{\mathcal{B}} \Rightarrow 1\right] - \Pr\left[\boldsymbol{G}_1^{\mathcal{B}} \Rightarrow 1\right]\right| = 0,$$

then for any PPT *adaptive adversary $\mathcal{A}$ we also have*

$$\left|\Pr\left[\boldsymbol{G}_2^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\boldsymbol{G}_1^{\mathcal{A}} \Rightarrow 1\right]\right| = 0.$$

Proof. Suppose an adaptive adversary $\mathcal{A}$ distinguishes $\mathbf{G}_1$ and $\mathbf{G}_2$ with advantage $\varepsilon > 0$. Let Q be a polynomial bound on the number of oracle queries made by $\mathcal{A}$.

We construct a selective adversary $\mathcal{B}$ as follows. Before the game starts, $\mathcal{B}$ guesses the entire query sequence that $\mathcal{A}$ will make; denote this guess by $\mathcal{Q}^\star$. Since $\mathcal{A}$ is a PPT algorithm and makes at most Q queries, the set of possible sequences is finite; therefore $\mathcal{B}$ guesses correctly with some probability $\gamma > 0$. It then commits to $\mathcal{Q}^\star$, as required in the selective setting.

Next $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine and answers queries. When $\mathcal{A}$ makes its t-th query, $\mathcal{B}$ checks whether it equals the t-th query in $\mathcal{Q}^\star$. If it matches, $\mathcal{B}$ forwards the query to its own challenger and returns the response to $\mathcal{A}$. If it does not match (or $\mathcal{A}$ makes more than Q queries), then $\mathcal{B}$ aborts and outputs a random bit. Finally, if $\mathcal{A}$ outputs a bit b', then $\mathcal{B}$ outputs the same bit.

[12] Since $(i, (\mathsf{ct}_0, \mathsf{ct}_1)) \in \mathcal{L}_{\mathsf{HC}}$, we have $\mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) = 0$, and hence $\mathsf{SEval}(\mathsf{hk}_i, \mathsf{ct}_0) = \mathsf{SEval}(\mathsf{hk}_1 + \mathsf{d}_i, \mathsf{ct}_0) = \mathsf{SEval}(\mathsf{hk}'_1 + \mathsf{d}_i, \mathsf{ct}_0) = \mathsf{SEval}(\mathsf{hk}'_i, \mathsf{ct}_0)$.

Conditioned on the event that $\mathcal{Q}^\star$ is correct, the entire interaction between $\mathcal{A}$ and $\mathcal{B}$ is a perfect simulation of the adaptive game, so $\mathcal{A}$'s advantage is preserved. Hence,

$$\begin{aligned}\left|\Pr\left[\mathbf{G}_2^{\mathcal{B}} \Rightarrow 1\right] - \Pr\left[\mathbf{G}_1^{\mathcal{B}} \Rightarrow 1\right]\right| &= \gamma \cdot \left|\Pr\left[\mathbf{G}_2^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathbf{G}_1^{\mathcal{A}} \Rightarrow 1\right]\right| \\ &= \gamma \cdot \varepsilon > 0,\end{aligned}$$

which contradicts the premise. Therefore no such adaptive adversary exists. □

Statistical extension. The same statement holds if perfect indistinguishability is replaced by statistical indistinguishability: if $\mathbf{G}_1$ and $\mathbf{G}_2$ are statistically indistinguishable for any selective adversary, then they are statistically indistinguishable for any adaptive adversary. The proof again proceeds by contradiction (assuming the existence of such an adaptive adversary), but now the selective adversary $\mathcal{B}$ commits to a random query order and answers $\mathcal{A}$'s oracle queries by direct computation using unbounded power (since the games are statistically close), rather than via its own oracle. Upon receiving the challenge bit from $\mathcal{A}$, it outputs the same bit, thereby transferring the advantage from $\mathcal{A}$ to $\mathcal{B}$. This contradicts the assumption that no selective adversary can distinguish the games with non-negligible advantage, thus proving the statistical extension of the lemma.

5.3 Multi-challenge aHRA security

Theorem 2. *Let* VSHPS *be a* (ε, d)*-universal vector space hash proof system over a linear language of dimension* k *with* $d = \ell - k \geq k$*, the proxy re-encryption scheme* PRE *described in Figure 5 is* aHRA *secure in the multi-challenge setting. Specifically, for any* PPT *adversary* $\mathcal{A}$ *making at most* $\mathsf{Q_C}$ *challenge queries, there exists a* PPT *algorithm* $\mathcal{B}$ *such that* $\mathbf{T}(\mathcal{B}) \approx \mathbf{T}(\mathcal{A})$*, and we have:*

$$\mathsf{Adv}^{\mathsf{aHRA},\mathsf{Q_C}}_{\mathsf{PRE}_2,\mathcal{A}}(\lambda) \leq \binom{\mathsf{N_U}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda) + 3 \cdot \mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).$$

Proof. The proof proceeds via a sequence of hybrid games, $\mathbf{G}_0, \ldots, \mathbf{G}_9$. The transitions from $\mathbf{G}_0$ to $\mathbf{G}_5$ mirror those in the proof of Theorem 1, but are adapted for the multi-challenge setting. The key differences are as follows:

- The transition from $\mathbf{G}_2$ to $\mathbf{G}_3$ relies on the $\mathsf{Q_C}$-Hard Subset Membership ($\mathsf{Q_C}$-HSM) property. All $\mathsf{Q_C}$ challenge statements are switched from being sampled in the language via $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk})$ to being sampled uniformly from the ambient space via $\mathsf{Sample}_{\mathcal{X}}(1^\lambda)$. This introduces a difference $\mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda)$.
- In $\mathbf{G}_4$, the bad event $\mathsf{Bad}_{\mathsf{Univ}}$ is defined with respect to the (ε, d)-universality of the VSHPS. The difference is bounded by $|\mathsf{pr}_4 - \mathsf{pr}_3| \leq \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda)$.
- In $\mathbf{G}_5$, the challenge ciphertexts are randomized. Let $\mathbf{H} \in \mathbb{F}^{\ell \times d}$ be a basis of the $\mathcal{HK}''$ over linear language $\mathcal{L}_{\mathsf{lpk}}$. We sample a random vector $\mathbf{v} \in \mathbb{F}^d$ and compute all the challenge ciphertexts as $\mathsf{ct}_1^j := \mathsf{SEval}(\mathsf{hk}_i + \mathbf{Hv}, \mathsf{ct}_0^j) + \mathsf{m}_b^j$ for

$j \in [\mathsf{Q_C}]$ and $i \in [\mathsf{N_U}]$ indexing the challenge users.[13] The proof of this step is similar to the single-challenge case, but requires a more detailed explanation.

- In $\mathbf{G}_{4.1}$, we index the first challenge user as 1.
- In $\mathbf{G}_{4.2}$, we do the same modification as in the single-challenge case, except that we now sample a random vector $\mathbf{v} \in \mathbb{F}^d$ and compute $\mathsf{hk}_1 := \mathsf{hk}'_1 + \mathbf{Hv}$, where $\mathsf{hk}'_1 \xleftarrow{\$} \mathcal{HK}$. We note that since all the challenge users share the same hash key hk_1, the random vector $\mathbf{Hv}$ is also shared among all challenge ciphertexts.

Alg $\mathcal{O}_{\mathsf{Chall},\mathsf{b}}(i, \mathsf{m}_0, \mathsf{m}_1)$:

01 $\mathsf{x} \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda)$ // $\mathbf{G}_5$
02 $(\mathsf{lpk}_1, \mathsf{td}_1) \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$ // $\mathbf{G}_6$-$\mathbf{G}_7$
03 $(\mathsf{x}, \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}_1)$ // $\mathbf{G}_6$-$\mathbf{G}_7$
04 $(\mathsf{lpk}'_1, \mathsf{td}'_1) \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell + 1)$ // $\mathbf{G}_8$
05 $(\mathsf{x}', \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}'_1)$ // $\mathbf{G}_8$
06 $\mathsf{x}' \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda)$ // $\mathbf{G}_9$
07 $\mathsf{ct}_0 := \mathsf{x}$ // $\mathbf{G}_5$-$\mathbf{G}_6$
08 $\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}_i, \mathsf{x}) + \mathsf{m}_b + \mathsf{SEval}(\mathbf{Hv}, \mathsf{x})$ // $\mathbf{G}_5$-$\mathbf{G}_6$
09 $\begin{pmatrix} \mathsf{ct}_0 \\ \mathsf{ct}_1 \end{pmatrix} := \begin{pmatrix} \mathbf{0} \\ \mathsf{SEval}(\mathsf{hk}_i, \mathsf{x}) + \mathsf{m}_b + \mathbf{R}_0^\intercal \cdot \mathsf{x} \end{pmatrix} + \begin{pmatrix} \mathsf{x} \\ \mathbf{r}^\intercal \mathbf{M} \cdot \mathsf{x} \end{pmatrix}$ // $\mathbf{G}_7$
10 $\begin{pmatrix} \mathsf{ct}_0 \\ \mathsf{ct}_1 \end{pmatrix} := \begin{pmatrix} \mathbf{0} \\ \mathsf{SEval}(\mathsf{hk}_i, \bar{\mathsf{x}}') + \mathsf{m}_b + \mathbf{R}_0^\intercal \cdot \bar{\mathsf{x}}' \end{pmatrix} + \mathsf{x}'$ // $\mathbf{G}_8$-$\mathbf{G}_9$
11 **return** $\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1)$

Fig. 8. Hybrid games for aHRA security from $\mathbf{G}_5$ onwards. Let lpk_1 define a linear language of dimension k over an ambient space of dimension ℓ. The original language lpk_0 has dimension k over an ambient space of dimension ℓ.

The subsequent games, $\mathbf{G}_6$ through $\mathbf{G}_9$, are detailed in Figure 8.

Game $\mathbf{G}_6$: In this game, we change the distribution of the statement component ct_0 for all challenge ciphertexts. Instead of sampling ct_0 uniformly from the ambient space via $\mathsf{Sample}_{\mathcal{X}}(1^\lambda)$, we sample it from the language $\mathcal{L}_{\mathsf{lpk}_1}$ via $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}_1)$, where $\mathsf{lpk}_1 \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$ is a new language public key. The difference between $\mathbf{G}_5$ and $\mathbf{G}_6$ is bounded by the $\mathsf{Q_C}$-HSM property.

$$|\mathsf{pr}_6 - \mathsf{pr}_5| \leq \mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_7$: This game is a conceptual rewrite of $\mathbf{G}_6$. We first express the challenge ciphertext generation in matrix form, which does not change the adversary's view. Then we rewrite the term $\mathsf{SEval}(\mathbf{Hv}, \mathsf{x})$ as $\mathbf{R}^\intercal \cdot \mathsf{x}$ where $\mathbf{R}^\intercal := \mathsf{SEval}(\mathbf{Hv}, \mathbf{u}_1) | \cdots | \mathsf{SEval}(\mathbf{Hv}, \mathbf{u}_\ell)$ is a row vector in $\mathbb{F}^{1 \times \ell}$ and $\mathbf{u}_1, \ldots, \mathbf{u}_\ell$ is

[13] We note that there are at most $\mathsf{Q_C}$ challenge ciphertexts which are distributed among $\mathsf{N_U}$ users and the exact allocation of ciphertexts to users is not fixed.

the standard basis of $\mathbb{F}^\ell$. Since SEval is linear in its second argument, we have $\mathsf{SEval}(\mathbf{Hv}, \mathsf{x}) = \mathbf{R}^\intercal \cdot \mathsf{x}$. Finally, we modify the vector $\mathbf{R}$ to be uniformly random in a subspace of dimension d of $\mathbb{F}^{\ell \times 1}$, where d is the dimension of the hash key space $\mathcal{HK}''$. This change is valid because $\mathbf{v} \in \mathbb{F}^d$ is uniformly random, $\mathbf{H} \in \mathbb{F}^{\ell \times d}$ has full column rank d, and the vectors $\mathbf{u}_1, \ldots, \mathbf{u}_\ell$ are linearly independent. By the d-MinEntropy property of VSHPS, $\mathbf{R}$ has full min-entropy $d \cdot \log |\mathbb{F}|$. Thus we can rewrite $\mathbf{R}^\intercal$ as $\mathbf{r}^\intercal \mathbf{M} + \mathbf{R}_0^\intercal$ for some fixed $\mathbf{R}_0 \in \mathbb{F}^{\ell \times 1}$, where $\mathbf{r} \xleftarrow{\$} \mathsf{U}\left(\mathbb{F}^d\right)$ and $\mathbf{M} \in \mathbb{F}^{d \times \ell}$ is a full-row-rank matrix; this change is purely conceptual.

$$\mathsf{pr}_7 = \mathsf{pr}_6.$$

Game $\mathbf{G}_8$: This game reinterprets the structure of the challenge ciphertext. In $\mathbf{G}_7$, the vector added to the core encryption is $\begin{pmatrix} \mathsf{x} \\ \mathbf{r}^\intercal \mathbf{M} \cdot \mathsf{x} \end{pmatrix} = \begin{pmatrix} \mathbf{I}_\ell \\ \mathbf{r}^\intercal \mathbf{M} \end{pmatrix} \cdot \mathsf{x}$. Since $\mathbf{I}_\ell$ is the $\ell \times \ell$ identity matrix, $\mathbf{r}$ is a random vector in $\mathbb{F}^d$ with $d \geq k = \dim(\mathcal{L}_{\mathsf{lpk}})$ and $\mathbf{M} \in \mathbb{F}^{d \times \ell}$ is a full-row-rank matrix independent of $\mathcal{L}_{\mathsf{lpk}_1}$, by the language linearity property, the matrix $\begin{pmatrix} \mathbf{I}_\ell \\ \mathbf{r}^\intercal \mathbf{M} \end{pmatrix}$ is an isomorphism from $\mathcal{L}_{\mathsf{lpk}_1}$ to a new linear space of dimension k over an ambient space of dimension $\ell + 1$. We denote this new linear space as $\mathcal{L}_{\mathsf{lpk}'_1}$, and rewrite the challenge ciphertexts as $\begin{pmatrix} \mathsf{ct}_0 \\ \mathsf{ct}_1 \end{pmatrix} := \begin{pmatrix} \mathbf{0} \\ \mathsf{SEval}(\mathsf{hk}_i, \bar{\mathsf{x}'}) + \mathsf{m}_b + \mathbf{R}_0^\intercal \cdot \bar{\mathsf{x}'} \end{pmatrix} + \mathsf{x}'$, where $(\mathsf{x}', \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}'_1)$ and $\bar{\mathsf{x}'}$ is the first ℓ components of x'.

$$\mathsf{pr}_8 = \mathsf{pr}_7.$$

Game $\mathbf{G}_9$: In this final game, we change how x' is generated for all challenge ciphertexts. Instead of sampling x' from the language $\mathcal{L}_{\mathsf{lpk}'_1}$ via $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}'_1)$, we sample it uniformly at random from the ambient space $\mathbb{F}^{\ell+1}$ via $\mathsf{Sample}_{\mathcal{X}}(1^\lambda)$. The indistinguishability between $\mathbf{G}_8$ and $\mathbf{G}_9$ is guaranteed by the $\mathsf{Q_C}$-HSM property of the language $\mathcal{L}_{\mathsf{lpk}'}$. [14]

$$|\mathsf{pr}_9 - \mathsf{pr}_8| \leq \mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda).$$

In $\mathbf{G}_9$, the challenge ciphertext is computed as $\begin{pmatrix} \mathsf{ct}_0 \\ \mathsf{ct}_1 \end{pmatrix} := \begin{pmatrix} \mathbf{0} \\ \mathsf{SEval}(\mathsf{hk}_i, \bar{\mathsf{x}'}) + \mathsf{m}_b \end{pmatrix} + \mathsf{x}'$, where x' is a uniformly random vector in $\mathbb{F}^{\ell+1}$. This vector acts as a one-time pad, making the resulting ciphertext $(\mathsf{ct}_0, \mathsf{ct}_1)$ uniformly random. Consequently, the adversary's success probability is the same as that of random guessing.

$$\mathsf{pr}_9 = \frac{1}{2}.$$

Combining the above results, we complete the proof. □

[14] The language dimension remains the same, but the ambient space dimension differs (from $\mathbb{F}^\ell$ to $\mathbb{F}^{\ell+1}$), which causes a negligible difference in the HSM advantage. For simplicity, we uniformly denote it by $\mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda)$.

6 CCA-secure PRE with (almost) tight security

In this section, we extend the aHRA-secure proxy re-encryption scheme from Figure 5 to achieve CCA-security.

6.1 General construction of aCCA schemes

Our construction builds upon the aHRA-secure proxy re-encryption scheme from Section 5. In that scheme, a ciphertext $\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1)$ consists of a statement ct_0 for a hard subset membership problem and a component $\mathsf{ct}_1 = \mathsf{PEval}(\mathsf{projk}_i, \mathsf{ct}_0, \mathsf{w}) + \mathsf{m}$ that binds the message m to an evaluation of ct_0 within a hash proof system.

The first obstacle in attaining CCA2-security is that it forbids any ciphertext malleability. A standard remedy, going back to [CHK03], is to append a one-time signature to the ciphertext. One might be tempted to sign the component ct_1, since it binds the message. However, re-encryption necessarily modifies ct_1, thereby invalidating the signature. This conflict precludes a direct extension of the aHRA-secure scheme to a aCCA-secure construction.

To resolve this issue, we restructure the ciphertext by introducing a third component, ct_2. We redefine ct_1 to be solely the hash proof system evaluation, i.e., $\mathsf{ct}_1 = \mathsf{PEval}(\mathsf{projk}_i, \mathsf{ct}_0, \mathsf{w})$, and move the message to ct_2, setting $\mathsf{ct}_2 = \mathsf{PEval}(\mathsf{projk}, \mathsf{ct}_0, \mathsf{w}) + \mathsf{m}$, where projk is a public parameter common to all users. Consequently, ct_2 remains invariant under re-encryption, which only affects ct_1. This design allows us to sign the pair $(\mathsf{ct}_0, \mathsf{ct}_2)$ to ensure integrity, providing the non-malleability required for aCCA-security while preserving the re-encryption functionality.

Furthermore, inspired by the Cramer-Shoup cryptosystem, we must ensure that the ciphertext components ct_0 and ct_1 are well-formed. Specifically, we need to prove that ct_1 is a valid evaluation of ct_0 under the hash proof system. We achieve this by incorporating a simulation-sound tag-based language-malleable NIZK proof system (tLM-NIZK). This system proves the relation between ct_0 and ct_1 using a tag derived from the one-time signature's verification key, i.e., $\mathsf{t} = \mathsf{H}(\mathsf{ovk})$, detailed in Appendix C. The final ciphertext is thus $\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1, \mathsf{ct}_2, \mathsf{ovk}, \sigma, \pi)$.

Our construction combines a language distribution SMP for a subset membership problem, a vector space hash proof system VSHPS, a one-time signature scheme OTS, and a special simulation-sound tag-based language-malleable NIZK proof system tLM-NIZK for the language sampled by SMP. We give our construction in Figure 9. Correctness is immediate from the construction.

6.2 (Almost) Tight aCCA security

The security proof for our aCCA-secure scheme builds upon the proof for the aHRA-secure scheme from Section 5. However, handling the decryption and re-encryption oracles requires additional arguments.

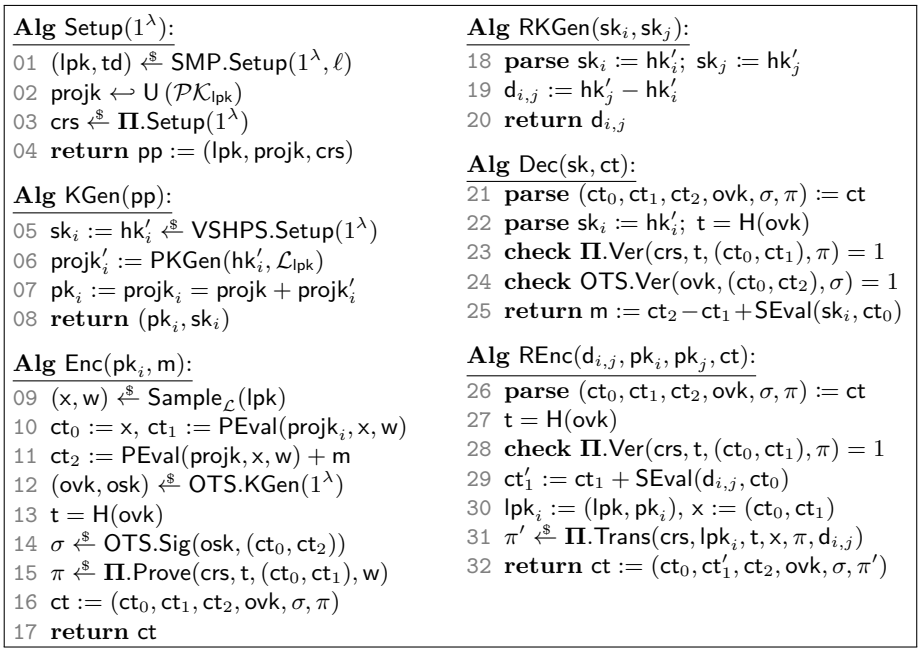

Fig. 9. General construction of aCCA-secure PRE scheme $\mathsf{PRE}_{\mathsf{aCCA}}$ with (almost) tight security. $\mathbf{\Pi}$ is a tLM-NIZK of the statement $\exists \mathsf{w} : \mathsf{x} \in_{\mathsf{w}} \mathcal{L}_{\mathsf{lpk}} \wedge \mathsf{ct}_1 = \mathsf{PEval}(\mathsf{projk}_i, \mathsf{x}, \mathsf{w})$ for the language lpk_i.

Decryption Oracle Queries We now detail the handling of decryption oracle queries. The core of the argument relies on the unforgeability of the one-time signature scheme and the special simulation-soundness of the tLM-NIZK proof system. When the oracle receives a ciphertext $\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1, \mathsf{ct}_2, \mathsf{ovk}, \sigma, \pi)$ for decryption under a key pk_i, we distinguish between the following cases. We denote components of the challenge ciphertext with a superscript star ($\star$).

The query uses the challenge verification key $\mathsf{ovk}^\star$ but is not a re-encryption of the challenge ciphertext. Here, $\mathsf{ovk} = \mathsf{ovk}^\star$, but $\sigma \neq \sigma^\star$ or $(\mathsf{ct}_0, \mathsf{ct}_2) \neq (\mathsf{ct}_0^\star, \mathsf{ct}_2^\star)$. For the decryption query to be valid, σ must be a valid signature on $(\mathsf{ct}_0, \mathsf{ct}_2)$ under $\mathsf{ovk}^\star$. This constitutes a forgery against the one-time signature scheme, an event that occurs with only negligible probability due to its unforgeability.

A collision has been found for ovk. Here, $\mathsf{ovk} \neq \mathsf{ovk}^\star$ but $\mathsf{t} = \mathsf{H}(\mathsf{ovk}) = \mathsf{H}(\mathsf{ovk}^\star) = \mathsf{t}^\star$, yielding a collision for the hash function.

The query uses a new verification key. Here, $\mathsf{ovk} \neq \mathsf{ovk}^\star$, and the challenger has not simulated a proof for the tag $\mathsf{t} = \mathsf{H}(\mathsf{ovk})$. By the special simulation-soundness of the tLM-NIZK system, an adversary cannot produce a valid proof π for a false statement. Thus, if the proof verifies, it must be that $\exists \mathsf{w} : \mathsf{x} \in_{\mathsf{w}} \mathcal{L}_{\mathsf{lpk}} \wedge \mathsf{ct}_1 = \mathsf{PEval}(\mathsf{projk}_i, \mathsf{x}, \mathsf{w})$. The information revealed by such a query does not compromise the (ε, d)-universality of the hash proof system, as

the set of possible hash keys consistent with all public information and oracle queries remains an affine subspace of dimension at least d.

The query is a re-encryption of the challenge ciphertext. Here, $\mathsf{ovk} = \mathsf{ovk}^\star$, $\sigma = \sigma^\star$, and $(\mathsf{ct}_0, \mathsf{ct}_2) = (\mathsf{ct}_0^\star, \mathsf{ct}_2^\star)$, so ct differs from $\mathsf{ct}^\star$ only in ct_1 and π. For the proof π to be valid, the special simulation-soundness of the tLM-NIZK implies that ct must be a valid re-encryption of some ciphertext. Since the other components match the challenge, ct must be a re-encryption of $\mathsf{ct}^\star$. Such queries are disallowed by the definition of CCA-security, so the oracle can reject them.

Re-Encryption Oracle Queries The primary difference in handling re-encryption oracle queries compared to the aHRA-secure scheme arises when the adversary requests the re-encryption of a self-generated ciphertext from an uncorrupted user to a corrupted user. In all other permitted cases, the simulation can use the re-encryption key, which is available for pairs of uncorrupted users or pairs of corrupted users.

To handle the challenging case, we add a check to the algorithm REnc that ensures the components ct_0 and ct_1 are well-formed, i.e., $\mathbf{\Pi}.\mathsf{Ver}(\mathsf{crs}, \mathsf{t}, (\mathsf{ct}_0, \mathsf{ct}_1), \pi) = 1$. This ensures that only well-formed ciphertexts—those for which there exists a witness w such that $\mathsf{x} \in_{\mathsf{w}} \mathcal{L}_{\mathsf{lpk}}$ and $\mathsf{ct}_1 = \mathsf{PEval}(\mathsf{projk}_i, \mathsf{x}, \mathsf{w})$—can be re-encrypted. If this check fails, the oracle rejects the query. In this case, all the information the adversary can learn from $\mathcal{O}_{\mathsf{ReEnc}}$ is through the evaluation of $\mathsf{SEval}(\mathsf{d}_{i,j}, \mathsf{ct}_0)$ for users i and j, which leaks no information beyond the public keys.[15]

Summarizing the above intuitions, we have the following theorem.

Theorem 3 (aCCA security). *Let* $\mathsf{PRE}_{\mathsf{aCCA}}$ *be the proxy re-encryption scheme from Figure 9. Assume that*

- SMP *is a hard subset membership language distribution.*
- *The vector space hash proof system* VSHPS *is* (ε, d)*-universal for negligible* ε *with* $d \geq 1$ *for single-challenge* aCCA *or* $d \geq k$ *for multi-challenge* aCCA.
- *The one-time signature scheme* OTS *is strongly unforgeable.*
- *The* tLM-NIZK *proof system* $\mathbf{\Pi}$ *is a (one-time/unbounded) extractable special simulation-sound tag-based language-malleable* NIZK *proof system.*

Then, $\mathsf{PRE}_{\mathsf{aCCA}}$ *is* aCCA*-secure. We will prove the theorem in Appendix F and G.*

Acknowledgements

Chen Qian was partially supported by the National Natural Science Foundation of China (Grant 62302272), the Shandong Provincial Outstanding Young Scientists Fund (Overseas) (Grant 2024HWYQ-013), the National Natural Science Foundation of China (Grant 62472255), and the National Key R&D Program of China (Grant 2023YFA1011200). Shuai Han was partially supported by the National Natural Science Foundation of China (62522212) and the National Key R&D Program of China (2025YFA1017100).

[15] We prove this result in the security proof of the aCCA-secure PRE scheme.

References

ABH09. Giuseppe Ateniese, Karyn Benson, and Susan Hohenberger. Key-private proxy re-encryption. In Marc Fischlin, editor, *CT-RSA 2009*, volume 5473 of *LNCS*, pages 279–294. Springer, Berlin, Heidelberg, April 2009. 2

ABPW13. Yoshinori Aono, Xavier Boyen, Le Trieu Phong, and Lihua Wang. Key-private proxy re-encryption under LWE. In Goutam Paul and Serge Vaudenay, editors, *INDOCRYPT 2013*, volume 8250 of *LNCS*, pages 1–18. Springer, Cham, December 2013. 2, 4

AFGH05. Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. In *NDSS 2005*. The Internet Society, February 2005. 2, 4

AJO+19. Masayuki Abe, Charanjit S. Jutla, Miyako Ohkubo, Jiaxin Pan, Arnab Roy, and Yuyu Wang. Shorter QA-NIZK and SPS with tighter security. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 669–699. Springer, Cham, December 2019. 5, 8, 37, 40

BBS98. Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 127–144. Springer, Berlin, Heidelberg, May / June 1998. 1, 2, 4

BHJ+15. Christoph Bader, Dennis Hofheinz, Tibor Jager, Eike Kiltz, and Yong Li. Tightly-secure authenticated key exchange. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part I*, volume 9014 of *LNCS*, pages 629–658. Springer, Berlin, Heidelberg, March 2015. 2

BJLS16. Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 273–304. Springer, Berlin, Heidelberg, May 2016. 5, 6

BLMR13. Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Berlin, Heidelberg, August 2013. 2

CDD+04. Ran Canetti, Ivan Damgård, Stefan Dziembowski, Yuval Ishai, and Tal Malkin. Adaptive versus non-adaptive security of multi-party protocols. *Journal of Cryptology*, 17(3):153–207, June 2004. 3, 5

CH07. Ran Canetti and Susan Hohenberger. Chosen-ciphertext secure proxy re-encryption. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM CCS 2007*, pages 185–194. ACM Press, October 2007. 2, 3, 4, 5, 9, 11, 12

CHK03. Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 255–271. Springer, Berlin, Heidelberg, May 2003. 28

CKLM12. Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. Malleable proof systems and applications. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 281–300. Springer, Berlin, Heidelberg, April 2012. 4, 5, 40

Coh19. Aloni Cohen. What about bob? The inadequacy of CPA security for proxy reencryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 287–316. Springer, Cham, April 2019. 2, 4

CS02. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, Berlin, Heidelberg, April / May 2002. 5, 6, 18

EHK+13. Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Berlin, Heidelberg, August 2013. 17, 34

EHK+17. Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Luis Villar. An algebraic framework for Diffie-Hellman assumptions. *Journal of Cryptology*, 30(1):242–288, January 2017. 33

FKKP19. Georg Fuchsbauer, Chethan Kamath, Karen Klein, and Krzysztof Pietrzak. Adaptively secure proxy re-encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 317–346. Springer, Cham, April 2019. 3, 5

FL19. Xiong Fan and Feng-Hao Liu. Proxy re-encryption and re-signatures from lattices. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 2019*, volume 11464 of *LNCS*, pages 363–382. Springer, Cham, June 2019. 2, 4

Gen09. Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, USA, 2009. 2, 4

GHKP18. Romain Gay, Dennis Hofheinz, Lisa Kohl, and Jiaxin Pan. More efficient (almost) tightly secure structure-preserving signatures. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 230–258. Springer, Cham, April / May 2018. 48

GS08. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Berlin, Heidelberg, April 2008. 36, 37, 41, 44, 50

HLG23. Shuai Han, Shengli Liu, and Dawu Gu. Almost tight multi-user security under adaptive corruptions & leakages in the standard model. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 132–162. Springer, Cham, April 2023. 4, 5, 7, 8, 19

HRsV07. Susan Hohenberger, Guy N. Rothblum, abhi shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 233–252. Springer, Berlin, Heidelberg, February 2007. 2

JKK+17. Zahra Jafargholi, Chethan Kamath, Karen Klein, Ilan Komargodski, Krzysztof Pietrzak, and Daniel Wichs. Be adaptive, avoid overcommitting. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 133–163. Springer, Cham, August 2017. 3

KPW15. Eike Kiltz, Jiaxin Pan, and Hoeteck Wee. Structure-preserving signatures from standard assumptions, revisited. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 275–295. Springer, Berlin, Heidelberg, August 2015. 33, 34, 44, 45

LCB+25. Yunhao Ling, Jie Chen, Zijian Bao, Man Ho Au, Luping Wang, and Haifeng Qian. Tightly, adaptively secure proxy re-encryption in multi-challenge setting. In *Advances in Cryptology – ASIACRYPT 2025*, volume 16250 of *Lecture Notes in Computer Science*, pages 174–204. Springer Singapore, 2025. 4, 5

LLP20. Youngkyung Lee, Dong Hoon Lee, and Jong Hwan Park. Tightly CCA-secure encryption scheme in a multi-user setting with corruptions. *DCC*, 88(11):2433–2452, 2020. 5, 8

LV08. Benoît Libert and Damien Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. In Ronald Cramer, editor, *PKC 2008*, volume 4939 of *LNCS*, pages 360–379. Springer, Berlin, Heidelberg, March 2008. 2

LW10. Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 455–479. Springer, Berlin, Heidelberg, February 2010. 5

Sah99. Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th FOCS*, pages 543–553. IEEE Computer Society Press, October 1999. 37

WCY+10. Jian Weng, Minrong Chen, Yanjiang Yang, Robert H. Deng, Kefei Chen, and Feng Bao. CCA-Secure unidirectional proxy re-encryption in the adaptive corruption model without random oracles. Cryptology ePrint Archive, Report 2010/265, 2010. 3, 4, 5

ZLH24. Yunxiao Zhou, Shengli Liu, and Shuai Han. Multi-hop fine-grained proxy re-encryption. In Qiang Tang and Vanessa Teague, editors, *PKC 2024, Part II*, volume 14604 of *LNCS*, pages 161–192. Springer, Cham, April 2024. 4

A Assumptions

We first describe the Matrix Diffie-Hellman assumption [EHK+17].

Definition 7 ((ℓ,k)-Full Rank Matrix Distribution). *Let $(\ell,k) \in \mathbb{N}$ be integers with $\ell > k$. The (ℓ,k)-full-rank matrix distribution $\mathcal{D}_{\ell,k}$ outputs uniformly random matrices in $\mathbb{Z}_{\mathsf{p}}^{\ell\times k}$ whose first k rows form a full-rank matrix in $\mathbb{Z}_{\mathsf{p}}^{k\times k}$.*

For simplicity of notation, we denote the distribution by $\mathcal{D}_k = \mathcal{D}_{k+1,k}$ when $\ell = k+1$.

For a group $\mathbb{G}$ of prime order p, we give the MDDH assumption over $\mathbb{G}$. Note that in pairing groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{T})$, variants of MDDH assumption can be defined over $\mathbb{G}_1, \mathbb{G}_2$ respectively.

Definition 8 ($\mathcal{D}_{\ell,k}$-MDDH Assumption). *The $\mathcal{D}_{\ell,k}$-MDDH assumption holds over $\mathbb{G}$ if, for all PPT adversaries $\mathcal{A}$, the following advantage $\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_{\ell,k},\mathbb{G},\mathcal{A}}(\lambda)$ is negligible:*

$$\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_{\ell,k},\mathbb{G},\mathcal{A}}(1^\lambda) := \left|\Pr\left[\mathcal{A}(\left[\mathbf{A}\right]_1, \left[\mathbf{u}\right]_1) = 1\right] - \Pr\left[\mathcal{A}(\left[\mathbf{A}\right]_1, \left[\mathbf{Aw}\right]_1) = 1\right]\right|.$$

where $\mathbf{A} \leftarrow\!\!\$\ \mathcal{D}_{\ell,k}$, $\mathbf{w} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^k\right)$, and $\mathbf{u} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^\ell\right)$.

For the one-time structure-preserving signature used in our proof system, we also recall the kernel Matrix Diffie–Hellman assumption of Kiltz, Pan, and Wee [KPW15].

Definition 9 ($\mathcal{D}_k$-KerMDH Assumption [KPW15]). *Let $\mathcal{D}_k$ be a matrix distribution over $\mathbb{Z}_{\mathsf{p}}^{(k+1)\times k}$ and let $s \in \{1,2\}$. The $\mathcal{D}_k$-KerMDH assumption holds if, for every PPT adversary $\mathcal{A}$, the following advantage is negligible:*

$$\mathsf{Adv}^{\mathsf{KerMDH}}_{\mathcal{D}_k,\mathsf{GGen},\mathcal{A}}(1^\lambda) := \Pr\left[\begin{array}{l} \mathbf{c} \neq \mathbf{0} \ \wedge \ \mathbf{c}^\intercal \mathbf{A} = \mathbf{0} : \\ \mathcal{PG} \xleftarrow{\$} \mathsf{GGen}(1^\lambda), \quad \mathbf{A} \xleftarrow{\$} \mathcal{D}_k, \\ [\mathbf{c}]_{3-s} \xleftarrow{\$} \mathcal{A}(\mathcal{PG}, [\mathbf{A}]_s) \end{array}\right].$$

Here $[\cdot]_s$ denotes component-wise encoding in the source group $\mathbb{G}_s$. Kiltz, Pan, and Wee show that the $\mathcal{D}_k$-MDDH assumption implies the $\mathcal{D}_k$-KerMDH assumption. Hence, the latter does not introduce an additional assumption beyond MDDH in our setting.

Similarly, we define the $\mathcal{U}_{\ell,k}$-MDDH assumption over $\mathbb{G}$ in the same way as the $\mathcal{D}_{\ell,k}$-MDDH assumption, except that $\mathbf{A} \hookleftarrow \mathcal{U}_{\ell,k}$. We then give two lemmas showing the relationship between the $\mathcal{D}$-MDDH and $\mathcal{U}$-MDDH assumptions.

For an integer $Q \geq 1$, we define the Q-$\mathcal{D}$-MDDH problem over $\mathbb{G}$ as follows.

Definition 10 (Q-$\mathcal{D}_{\ell,k}$-MDDH). *The Q-$\mathcal{D}_{\ell,k}$-MDDH assumption holds over $\mathbb{G}$ if, for all PPT adversaries $\mathcal{A}$, the following advantage $\mathsf{Adv}^{Q\text{-MDDH}}_{\mathcal{D}_{\ell,k},\mathbb{G},\mathcal{A}}(\lambda)$ is negligible:*

$$\mathsf{Adv}^{Q\text{-MDDH}}_{\mathcal{D}_{\ell,k},\mathbb{G},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathcal{A}([\mathbf{A}]_1, [\mathbf{AW}]_1) = 1\right] - \Pr\left[\mathcal{A}([\mathbf{A}]_1, [\mathbf{U}]_1) = 1\right]\right|.$$

where $\mathbf{A} \hookleftarrow \mathcal{D}_{\ell,k}$, $\mathbf{W} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{k\times Q}\right)$, and $\mathbf{U} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{\ell\times Q}\right)$.

The $\mathcal{D}_{\ell,k}$-MDDH problem is random self-reducible [EHK+13]; that is, the Q-$\mathcal{D}_{\ell,k}$-MDDH and $\mathcal{D}_{\ell,k}$-MDDH problems can be tightly reduced to each other. If the distribution is changed to $\mathcal{U}_{\ell,k}$, the reduction is even tighter.

Lemma 4 (Random Self-Reducibility of MDDH [EHK+13]). *Let $Q \geq \ell - k$. For any PPT adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ such that, for all security parameters λ, the following holds.*

$$\mathsf{Adv}^{Q\text{-MDDH}}_{\mathcal{D}_{\ell,k},\mathbb{G},\mathcal{A}}(\lambda) \leq (\ell - k) \cdot \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_{\ell,k},\mathbb{G},\mathcal{B}}(\lambda) + \tfrac{1}{p-1}.$$

Let $Q \geq \ell - k$. For any PPT adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ such that, for all security parameters λ, the following holds.

$$\mathsf{Adv}^{Q\text{-MDDH}}_{\mathcal{U}_{\ell,k},\mathbb{G},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{U}_{\ell,k},\mathbb{G},\mathcal{B}}(\lambda) + \tfrac{1}{p-1}.$$

Definition 11 (Signature Scheme). *A signature scheme $\Sigma = (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$ has the following syntax.*

- $\mathsf{KGen}(1^\lambda) \to (\mathsf{svk}, \mathsf{ssk})$*: The key generation algorithm takes as input a security parameter λ and outputs a verification key* svk *and a signing key* ssk*.*
- $\mathsf{Sig}(\mathsf{ssk}, \mathsf{m}) \to \sigma$*: The signing algorithm takes as input a signing key* ssk *and a message* m*, and outputs a signature σ.*
- $\mathsf{Ver}(\mathsf{svk}, \mathsf{m}, \sigma) \to \{0,1\}$*: The verification algorithm takes as input a verification key* svk*, a message* m*, and a signature σ, and outputs 1 if the signature is valid, and 0 otherwise.*

We require the following properties:

Correctness: *For every message* m *in the message space,*

$$\Pr\left[\mathsf{Ver}(\mathsf{svk},\mathsf{m},\sigma)=1 \;\middle|\; \begin{array}{l}(\mathsf{svk},\mathsf{ssk}) \xleftarrow{\$} \mathsf{KGen}(1^\lambda),\\ \sigma \xleftarrow{\$} \mathsf{Sig}(\mathsf{ssk},\mathsf{m})\end{array}\right]=1.$$

One-Time Unforgeability: *For any* PPT *adversary* $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$*, the probability of producing a valid signature on a new message after obtaining one signature on a message of its choice is negligible. Formally,*

$$\Pr\left[\mathsf{Ver}(\mathsf{svk},\mathsf{m}',\sigma')=1\wedge\mathsf{m}'\neq\mathsf{m} \;\middle|\; \begin{array}{l}(\mathsf{svk},\mathsf{ssk}) \xleftarrow{\$} \mathsf{KGen}(1^\lambda),\\ (\mathsf{m},\mathsf{st}) \xleftarrow{\$} \mathcal{A}_1(\mathsf{svk}),\\ \sigma \xleftarrow{\$} \mathsf{Sig}(\mathsf{ssk},\mathsf{m}),\\ (\mathsf{m}',\sigma') \xleftarrow{\$} \mathcal{A}_2(\mathsf{st},\sigma)\end{array}\right]\leq \mathsf{negl}(\lambda).$$

Strong One-Time Unforgeability: *For any* PPT *adversary* $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$*, the probability of producing a new valid message–signature pair after obtaining one signature on a message of its choice is negligible. Formally,*

$$\Pr\left[\begin{array}{l}\mathsf{Ver}(\mathsf{svk},\mathsf{m}',\sigma')=1\\ \wedge\ (\mathsf{m}',\sigma')\neq(\mathsf{m},\sigma)\end{array} \;\middle|\; \begin{array}{l}(\mathsf{svk},\mathsf{ssk}) \xleftarrow{\$} \mathsf{KGen}(1^\lambda),\\ (\mathsf{m},\mathsf{st}) \xleftarrow{\$} \mathcal{A}_1(\mathsf{svk}),\\ \sigma \xleftarrow{\$} \mathsf{Sig}(\mathsf{ssk},\mathsf{m}),\\ (\mathsf{m}',\sigma') \xleftarrow{\$} \mathcal{A}_2(\mathsf{st},\sigma)\end{array}\right]\leq \mathsf{negl}(\lambda).$$

In particular, a different valid signature on the already signed message is also considered a successful forgery; this is the distinction from ordinary one-time unforgeability.

Unforgeability against Chosen Message Attacks: *For any* PPT *adversary* $\mathcal{A}$*, the probability of successfully forging a signature on a new message after querying the signing oracle is negligible. Formally, for any* $\mathcal{A}$*, we have:*

$$\Pr\left[\begin{array}{l}\mathsf{Ver}(\mathsf{svk},\mathsf{m}',\sigma')=1\\ \wedge\ \mathsf{m}'\notin\mathcal{Q}\end{array} \;\middle|\; \begin{array}{l}(\mathsf{svk},\mathsf{ssk}) \xleftarrow{\$} \mathsf{KGen}(1^\lambda),\\ (\mathsf{m}',\sigma') \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sig}(\mathsf{ssk},\cdot)}}(\mathsf{svk})\end{array}\right]\leq \mathsf{negl}(\lambda).$$

where $\mathcal{Q}$ *is the set of messages submitted to the signing oracle* $\mathcal{O}_{\mathsf{Sig}(\mathsf{ssk},\cdot)}$*.*

Definition 12 (Commitment Scheme). *A commitment scheme* Com *is a tuple* $\mathsf{Com}=(\mathsf{Setup},\mathsf{Commit})$ *with the following syntax.*

- $\mathsf{Setup}(1^\lambda)\to\mathsf{ck}$*: Takes the security parameter as input, and outputs* ck*.*
- $\mathsf{Commit}(\mathsf{ck},\mathsf{m};\mathsf{open})\to\mathsf{c}$*: Takes a commitment key* ck*, a message* m*, and an opening value* open *as input, and outputs a commitment* c*.*

We require the following properties:

Hiding: *For any security parameter* λ*, any two messages* m_0 *and* m_1*, the commitments to these messages are indistinguishable. Formally, for any* PPT *adversary*

$\mathcal{A}$, we have:

$$\left| \Pr\left[b = b' \;\middle|\; \begin{array}{l} \mathsf{ck} \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ (\mathsf{m}_0, \mathsf{m}_1) \xleftarrow{\$} \mathcal{A}(\mathsf{ck}) \\ b \hookleftarrow \{0,1\};\ \mathsf{open} \hookleftarrow \mathsf{U}(\mathcal{O}) \\ \mathsf{c} := \mathsf{Commit}(\mathsf{ck}, \mathsf{m}_b; \mathsf{open}) \\ b' := \mathcal{A}(\mathsf{ck}, \mathsf{c}) \end{array} \right] - \tfrac{1}{2} \right| \leq \mathsf{negl}(\lambda).$$

Binding: *For any security parameter 1^λ, it is infeasible to find two different messages m_0 and m_1 and corresponding opening values open_0 and open_1 such that they produce the same commitment. Formally, for any probabilistic polynomial-time adversary $\mathcal{A}$, we have:*

$$\Pr\left[\begin{array}{l} \mathsf{m}_0 \neq \mathsf{m}_1 \wedge \\ \mathsf{Commit}(\mathsf{ck}, \mathsf{m}_0; \mathsf{open}_0) \\ = \mathsf{Commit}(\mathsf{ck}, \mathsf{m}_1; \mathsf{open}_1) \end{array} \;\middle|\; \begin{array}{l} \mathsf{ck} \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ (\mathsf{m}_0, \mathsf{m}_1, \mathsf{open}_0, \mathsf{open}_1) := \mathcal{A}(\mathsf{ck}) \end{array} \right] \leq \mathsf{negl}(\lambda).$$

where $\mathsf{negl}(\lambda)$ denotes a negligible function in λ.

B Groth–Sahai Non-interactive Zero-Knowledge Proofs

Groth and Sahai [GS08] introduced a unified methodology for constructing non-interactive witness-indistinguishable (NIWI) and non-interactive zero-knowledge (NIZK) proofs over bilinear groups $\mathcal{PG}$. It proves satisfiability of systems of mixed-type equations of the form

$$\sum_{j=1}^{n} f(a_j, y_j) + \sum_{i=1}^{m} f(x_i, b_i) + \sum_{i=1}^{m} \sum_{j=1}^{n} f(x_i, \gamma_{ij} y_j) = t,$$

where A_1, A_2, A_T are $\mathbb{Z}_\mathsf{p}$-modules; $\mathbf{x} \in A_1^m$ and $\mathbf{y} \in A_2^n$ are variables; $\mathbf{a} \in A_1^n$, $\mathbf{b} \in A_2^m$, $\Gamma = (\gamma_{ij}) \in \mathbb{Z}_\mathsf{p}^{m \times n}$, and $t \in A_T$ are constants; and $f : A_1 \times A_2 \to A_T$ is a bilinear map.

For compatible vectors, write

$$\mathbf{x} \cdot \mathbf{y} = \sum_i f(x_i, y_i).$$

The mixed equation can then be written as

$$\mathbf{a} \cdot \mathbf{y} + \mathbf{x} \cdot \mathbf{b} + \mathbf{x} \cdot \Gamma \mathbf{y} = t.$$

Depending on the modules and the bilinear map, this notation captures the following standard equation types:

- pairing-product equations, with $A_1 = \mathbb{G}_1$, $A_2 = \mathbb{G}_2$, $A_T = \mathbb{G}_\mathsf{T}$, and $f(X, Y) = X \bullet Y$;
- multi-scalar multiplication equations in $\mathbb{G}_1$, with $A_1 = \mathbb{G}_1$, $A_2 = \mathbb{Z}_\mathsf{p}$, and $f(X, y) = yX$;

– multi-scalar multiplication equations in $\mathbb{G}_2$, with $A_1 = \mathbb{Z}_\mathsf{p}$, $A_2 = \mathbb{G}_2$, and $f(x, Y) = xY$; and
– quadratic equations in $\mathbb{Z}_\mathsf{p}$, with $A_1 = A_2 = A_T = \mathbb{Z}_\mathsf{p}$ and $f(x, y) = xy \bmod p$.

The GS proof system commits to each witness element and substitutes the commitments for the variables. The homomorphic properties of the commitments then allow the prover to supply the additional proof elements required for verification [GS08]. We next recall the commitment forms used in this paper.

Commitments. The public parameters contain a matrix $\mathbf{D} \hookleftarrow \mathcal{D}_k$ and vectors $\mathbf{z}, \mathbf{t} \in \mathbb{Z}_\mathsf{p}^{k+1}$; set $\mathbf{U} := [\mathbf{D} \mid \mathbf{z}]$. We use two parameter regimes:

– **Perfectly binding mode:** for scalar vectors, $\mathbf{z} \notin \mathsf{Span}(\mathbf{D})$; for group vectors, $\mathbf{z} \in \mathsf{Span}(\mathbf{D})$ and $\mathbf{t} \notin \mathsf{Span}(\mathbf{D})$.
– **Perfectly hiding mode:** for scalar vectors, $\mathbf{z} \in \mathsf{Span}(\mathbf{D})$; for group vectors, $\mathbf{z} \notin \mathsf{Span}(\mathbf{D})$ and $\mathbf{t} \in \mathsf{Span}(\mathbf{D})$.

To commit to a scalar vector $\mathbf{w} \in \mathbb{Z}_\mathsf{p}^n$ with $\mathbf{R} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k \times n}\right)$, compute

$$\left[\mathbf{C_w}\right] := \left[\mathbf{D}\right] \mathbf{R} + \left[\mathbf{z}\right] \mathbf{w}^\intercal.$$

To commit to a group vector $\left[\mathbf{w}\right] \in \mathbb{G}^n$ with $\mathbf{R} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1) \times n}\right)$, compute

$$\left[\mathbf{C_w}\right] := \left[\mathbf{U}\right] \mathbf{R} + \mathbf{t} \left[\mathbf{w}^\intercal\right].$$

These commitments have the standard GS dual-mode behavior. For scalar commitments, $\mathbf{z} \notin \mathsf{Span}(\mathbf{D})$ makes $[\mathbf{D} \mid \mathbf{z}]$ full rank and hence binds the committed scalar, whereas $\mathbf{z} = \mathbf{Dr}$ allows the message term to be absorbed into the randomness as $\left[\mathbf{D}\right](\mathbf{R} + \mathbf{r}\mathbf{w}^\intercal)$. For group-element commitments, when $\mathbf{z} \in \mathsf{Span}(\mathbf{D})$ and $\mathbf{t} \notin \mathsf{Span}(\mathbf{D})$, the component outside $\mathsf{Span}(\mathbf{D})$ binds the committed group element; when $\mathbf{z} \notin \mathsf{Span}(\mathbf{D})$, the matrix $\mathbf{U}$ is full rank and fresh $\mathbf{R}$ perfectly hides it. The $\mathcal{D}_k$-MDDH assumption makes the corresponding parameter regimes computationally indistinguishable.

The standard Groth–Sahai compiler applies these commitments componentwise to the two source groups and derives proof elements and verification equations according to the type of each mixed equation. We therefore do not introduce a single overloaded operation on commitments from different modules. The concrete source-group assignments, proof elements, and pairing equations used by our constructions are given explicitly in Supplementary Material Section D.

C Language-Malleable Zero-Knowledge Proof with Special Simulation Soundness

We adapt the notion of simulation soundness from Sahai [Sah99] to the tag-based setting. Our definition differs from the existing notions of weak and strong unbounded simulation soundness from [AJO+19]. As noted in [AJO+19], weak

unbounded simulation soundness is insufficient for building CCA-secure protocols. This is because it allows an adversary to generate a new proof for the same statement contained in a challenge ciphertext, and then submit this modified ciphertext to a decryption oracle.

Conversely, the stronger notion of unbounded simulation soundness is overly restrictive for our application. This notion forbids an adversary from creating a valid proof for any false statement, thereby precluding any form of proof malleability. Such a strict requirement is incompatible with proxy re-encryption, where the core functionality relies on the ability to transform a proof for an original ciphertext into a new, valid proof for a re-encrypted ciphertext.

To bridge this gap, we introduce the notion of *special simulation soundness*. This notion permits a controlled form of malleability, allowing an adversary to transform a simulated proof for a given statement into a new, valid proof for a related, but otherwise false, statement. This controlled malleability is formalized through the concept of a trapdoor language distribution, which we define below.

Definition 13 (Membership-Preserving Language Family).
A membership-preserving language family MPL *with respect to a map family* $\mathcal{H} = \{\mathsf{H}_i\} = \{(\mathsf{H}_i^{\mathsf{lpk}}, \mathsf{H}_i^{\mathsf{x}}, \mathsf{H}_i^{\mathsf{w}})\}$ *consists of three* PPT *algorithms* $\mathsf{MPL} = (\mathsf{Setup}, \mathsf{Samp}, \mathsf{Test})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda) \to (\mathsf{lpk}, \mathsf{td})$*: Takes a security parameter* λ *as input, and outputs a language public key* lpk *and trapdoor* td*.*
- $\mathsf{Samp}(\mathsf{lpk}) \to (\mathsf{x}, \mathsf{w})$*: Takes a public key* lpk *as input, and outputs a pair* (x, w) *such that* $\mathsf{x} \in \mathcal{L}_{\mathsf{lpk}}$ *and* $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}_{\mathsf{lpk}}$*.*
- $\mathsf{Test}(\mathsf{lpk}, \mathsf{x}, \mathsf{w}, \mathsf{td}) \to \{0, 1\}$*: Takes a public key* lpk*, a statement* x*, a witness* w*, and a trapdoor* td *as input, and outputs a bit indicating whether* $\mathsf{x} \in \mathcal{L}_{\mathsf{lpk}}$*.*

We require the following properties.

Correctness. *For any security parameter* λ*, we have*

$$\Pr\left[\mathsf{Test}(\mathsf{lpk}, \mathsf{x}, \mathsf{w}, \mathsf{td}) = 1 \;\middle|\; \begin{array}{l}(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ (\mathsf{x}, \mathsf{w}) \xleftarrow{\$} \mathsf{Samp}(\mathsf{lpk})\end{array}\right] = 1.$$

Membership-Preserving. *We say that* MPL *is membership-preserving with respect to a function family* $\mathcal{H} = \{\mathsf{H}_i\} = \{(\mathsf{H}_i^{\mathsf{lpk}}, \mathsf{H}_i^{\mathsf{x}}, \mathsf{H}_i^{\mathsf{w}})\}$ *if, for any security parameter* λ*, any public key* lpk*, and any statement–witness pair* (x, w)*, let* $\mathsf{lpk}' := \mathsf{H}^{\mathsf{lpk}}(\mathsf{lpk})$*,* $\mathsf{x}' := \mathsf{H}^{\mathsf{x}}(\mathsf{x})$*, and* $\mathsf{w}' := \mathsf{H}^{\mathsf{w}}(\mathsf{w})$ *yields*

$$\mathsf{x} \in_{\mathsf{w}} \mathcal{L}_{\mathsf{lpk}} \iff \mathsf{x}' \in_{\mathsf{w}'} \mathcal{L}_{\mathsf{lpk}'}.$$

To formalize the notion of special simulation soundness, we introduce a symmetric link function, $\mathsf{link} : (\mathcal{X}, \mathcal{L})^2 \to \{0, 1\}$, which specifies when two statement-language pairs are considered linked. This function is designed to capture the controlled malleability of the proof system. In the context of our proxy re-encryption scheme, for instance, two statements (i.e., ciphertexts) are linked if one can be transformed into the other via re-encryption. Consequently,

the special simulation soundness property must permit this form of malleability. That is, the production of a new proof for a statement-language pair linked to a previously simulated one does not constitute a break of soundness. Taking a MPL as an example, we have

$$\mathsf{link}(\mathsf{x}, \mathsf{lpk}, \mathsf{x}', \mathsf{lpk}') = 1 \implies (\mathsf{x}' = \mathsf{H}^{\mathsf{x}}(\mathsf{x}) \wedge \mathsf{lpk}' = \mathsf{H}^{\mathsf{lpk}}(\mathsf{lpk})) \\ \vee (\mathsf{x} = \mathsf{H}^{\mathsf{x}}(\mathsf{x}') \wedge \mathsf{lpk} = \mathsf{H}^{\mathsf{lpk}}(\mathsf{lpk}')).$$

Definition 14 (Language Malleable Tag-based NIZK). *A language-malleable tag-based* NIZK *proof system for a membership-preserving language family* MPL *with respect to a function family* $\mathcal{H} = \{\mathsf{H}_i\} = \{(\mathsf{H}_i^{\mathsf{lpk}}, \mathsf{H}_i^{\mathsf{x}}, \mathsf{H}_i^{\mathsf{w}})\}$[16] *consists of four* PPT *algorithms* $\boldsymbol{\Pi} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Trans})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$*: Takes a security parameter* λ *as input, and outputs a common reference string* crs*.*
- $\mathsf{Prove}(\mathsf{crs}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}, \mathsf{w}) \to \pi$*: Takes a* crs*, a language public key* lpk*, a tag* t*, a statement* x*, and a witness* w *for the language* $\mathcal{L}_{\mathsf{lpk}}$ *as input, and outputs a proof* π*.*
- $\mathsf{Ver}(\mathsf{crs}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}, \pi) \to \{0, 1\}$*: Takes a* crs*, a language public key* lpk*, a tag* t*, a statement* x*, and a proof* π *as input, and outputs a bit.*
- $\mathsf{Trans}(\mathsf{crs}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}, \pi, \mathsf{H}) \to (\mathsf{lpk}', \mathsf{x}', \pi')$*: Takes a* crs*, a language public key* lpk*, a tag* t*, a statement–proof pair* (x, π) *for the language* $\mathcal{L}_{\mathsf{lpk}}$*, and a transformation* $\mathsf{H} = (\mathsf{H}^{\mathsf{lpk}}, \mathsf{H}^{\mathsf{x}}, \mathsf{H}^{\mathsf{w}}) \in \mathcal{H}$ *as input. It outputs* $\mathsf{lpk}' = \mathsf{H}^{\mathsf{lpk}}(\mathsf{lpk})$*,* $\mathsf{x}' = \mathsf{H}^{\mathsf{x}}(\mathsf{x})$*, and a transformed proof* π' *for* x' *in* $\mathcal{L}_{\mathsf{lpk}'}$ *under the same tag* t*.*[17]

We require the following properties.

Correctness. *For any security parameter* λ*, any language public key* lpk*, any tag* t*, and any statement–witness pair* $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}_{\mathsf{lpk}}$*, we have:*

$$\Pr\left[\mathsf{Ver}(\mathsf{crs}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}, \pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ \pi \xleftarrow{\$} \mathsf{Prove}(\mathsf{crs}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}, \mathsf{w}) \end{array}\right] = 1.$$

Zero-Knowledge. *There exists a* PPT *simulator* $\mathcal{S} = (\mathsf{SSetup}, \mathsf{Sim})$*, where* $\mathsf{SSetup}(1^\lambda) \to (\mathsf{crs}, \mathsf{simK})$ *and* $\mathsf{Sim}(\mathsf{crs}, \mathsf{simK}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}) \to \pi$*, such that for any* PPT *adversary* $\mathcal{A}$*, the following advantage is negligible:*

$$\mathsf{Adv}^{\mathsf{ZK}}_{\Pi,\mathcal{A}}(1^\lambda) := \Big|\Pr\Big[\mathcal{A}^{\mathcal{O}_{\mathsf{Prove}}(\mathsf{crs},\mathsf{lpk},\cdot)}(\mathsf{crs}) = 1\Big] \\ - \Pr\Big[\mathcal{A}^{\mathcal{O}_{\mathsf{Sim}}(\mathsf{crs},\mathsf{simK},\mathsf{lpk},\cdot)}(\mathsf{crs}) = 1\Big]\Big|,$$

where $(\mathsf{crs}, \mathsf{simK}) \xleftarrow{\$} \mathsf{SSetup}(1^\lambda)$ *in the second probability, and* $\mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$ *in the first. The oracles only answer queries for statements* $\mathsf{x} \in \mathcal{L}_{\mathsf{lpk}}$*. Note that* Prove_i *and* Sim_i *can detect invalid statements using* td *for the trapdoor language* $\mathcal{L}_{\mathsf{lpk}}$*.*

[16] We assume that the identity function is included in $\mathcal{H}$.

[17] In the concrete instantiations, the transformation is represented by a re-encryption token. Since lpk' and x' are publicly determined by the token and the original inputs, the displayed Trans algorithms return only π'. We also suppress lpk from the displayed Prove, Ver, Sim, and Trans interfaces when it is fixed by the user index.

Language Malleability. *We require that the proof transformation algorithm can efficiently transform a valid proof π for a statement x in $\mathcal{L}_{\mathsf{lpk}}$ into a valid proof π' for $\mathsf{x}' = \mathsf{H}^{\mathsf{x}}(\mathsf{x})$ in $\mathcal{L}_{\mathsf{lpk}'}$, where $\mathsf{lpk}' = \mathsf{H}^{\mathsf{lpk}}(\mathsf{lpk})$ and $\mathsf{H} = (\mathsf{H}^{\mathsf{lpk}}, \mathsf{H}^{\mathsf{x}}, \mathsf{H}^{\mathsf{w}}) \in \mathcal{H}$. Formally, for any tag t, any $(\mathsf{x}, \mathsf{w}) \in \mathcal{R}_{\mathsf{lpk}}$, and any $\mathsf{H} = (\mathsf{H}^{\mathsf{lpk}}, \mathsf{H}^{\mathsf{x}}, \mathsf{H}^{\mathsf{w}}) \in \mathcal{H}$, we have*

$$\Pr\left[\mathsf{Ver}(\mathsf{crs}, \mathsf{lpk}', \mathsf{t}, \mathsf{x}', \pi') = 1 \;\middle|\; \begin{array}{l} \mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda) \\ \pi \xleftarrow{\$} \mathsf{Prove}(\mathsf{crs}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}, \mathsf{w}) \\ (\mathsf{lpk}', \mathsf{x}', \pi') \xleftarrow{\$} \mathsf{Trans}(\mathsf{crs}, \mathsf{lpk}, \mathsf{t}, \mathsf{x}, \pi, \mathsf{H}) \end{array}\right] = 1.$$

Special Simulation Soundness. *For any* PPT *adversary $\mathcal{A}$, there exists a* PPT *simulator $\mathcal{S} = (\mathsf{SSetup}, \mathsf{Sim})$ such that the following probability is negligible.*

$$\Pr\left[\begin{array}{c} \mathsf{Ver}(\mathsf{crs}, \mathsf{lpk}^\star, \mathsf{t}^\star, \mathsf{x}^\star, \pi^\star) = 1 \\ \wedge \mathsf{x}^\star \notin \mathcal{L}_{\mathsf{lpk}^\star} \\ \wedge \nexists(\mathsf{x}, \mathsf{lpk}) \in \mathcal{T} : \mathsf{link}(\mathsf{x}, \mathsf{lpk}, \mathsf{x}^\star, \mathsf{lpk}^\star) = 1 \end{array} \;\middle|\; \begin{array}{l} (\mathsf{crs}, \mathsf{simK}) \xleftarrow{\$} \mathsf{SSetup}(1^\lambda) \\ (\mathsf{lpk}^\star, \mathsf{x}^\star, \mathsf{t}^\star, \pi^\star) \\ \quad \xleftarrow{\$} \mathcal{A}^{\mathsf{Sim}(\mathsf{crs}, \mathsf{simK}, \cdot)}(\mathsf{crs}) \end{array}\right],$$

where $\mathcal{T}$ records the statement–language pairs $(\mathsf{x}, \mathsf{lpk})$ submitted by $\mathcal{A}$ to the simulation oracle, and $\mathsf{link}(\mathsf{x}_1, \mathsf{lpk}_1, \mathsf{x}_2, \mathsf{lpk}_2)$ is the predicate defining whether two such pairs are linked via the function family $\mathcal{H}$.

Our notion of special simulation soundness permits a controlled form of malleability. Specifically, the proof transformation function, Trans, allows for the derivation of a new, valid proof for a statement that is legitimately related to the original one (e.g., a re-encrypted ciphertext in a PRE scheme), even when starting from a simulated proof. This property positions our notion between existing definitions: it is stronger than weak simulation soundness, which allows arbitrary proof modifications for a given tag, but weaker than strong simulation soundness, which prohibits any malleation of proofs for false statements, regardless of the tag.

Relation to other simulation soundness notions The literature presents two distinct definitions of simulation soundness for tag-based NIZK proof systems. The weak simulation soundness defined in [AJO+19] does not prevent the adversary from generating a new proof for false statements with the same tag used in the simulated proof but prohibits generating a valid proof for any false statement with other tags. Conversely, the strong simulation soundness in [AJO+19] prevents the adversary from generating any new valid proof for false statements. Our notion of special simulation soundness, which is inspired by [CKLM12], lies between these two definitions. It is stronger than weak simulation soundness because the adversary cannot generate a valid proof for any false statement with tags different from the one in the simulated proof. However, it is weaker than strong simulation soundness, as it allows the adversary to generate a new valid proof for some false statements with the same tag as the simulated proof. This relaxation is essential in the proxy re-encryption scheme, where the adversary can generate a new valid proof for re-encrypted ciphertexts with known tokens.

Generic Construction of Special Simulation Sound tLM-NIZK We now give the language family used in the aCCA proxy re-encryption scheme. In our case, a language public key consists of a matrix $\left[\mathbf{A}_i\right]_1$, and we need to prove membership in the linear span language $\mathcal{L}_{\mathbf{A}_i}$ defined as follows:

$$\mathcal{L}_{\mathbf{A}_i} := \{\mathsf{x} = \left[\mathbf{u}\right]_1 \in \mathbb{G}_1^{\ell+1} \mid \exists \mathbf{w} \in \mathbb{Z}_\mathsf{p}^k : \mathsf{x} = \left[\mathbf{A}_i\right]_1 \mathbf{w}\}.$$

For a fixed tag t, we define an extended language $\mathcal{L}_{\mathbf{A}_i}^{\mathsf{svk},\mathsf{ck}}$. We will show that a proof for $\mathcal{L}_{\mathbf{A}_i}^{\mathsf{svk},\mathsf{ck}}$ can be viewed as a special simulation-sound tLM-NIZK proof for the language $\mathcal{L}_{\mathbf{A}_i}$. Moreover, if the proof for $\mathcal{L}_{\mathbf{A}_i}^{\mathsf{svk},\mathsf{ck}}$ has language malleability over the choice of $\mathbf{A}_i$, then the corresponding tLM-NIZK proof for $\mathcal{L}_{\mathbf{A}_i}$ also has language malleability over that choice. The extended language is defined as follows:

$$\mathcal{L}_{\mathbf{A}_i}^{(\mathsf{svk},\mathsf{ck})} := \left\{ (\mathsf{x}, \mathbf{C_s}) \middle| \begin{array}{l} \exists \mathsf{w} \in \mathbb{Z}_\mathsf{p}^k : \mathsf{x} = \left[\mathbf{A}_i\right]_1 \mathsf{w} \\ \\ \vee \exists \mathbf{A}_j, \mathsf{x}', \sigma, \mathsf{open} : \left\{ \begin{array}{l} \mathbf{C_s} = \mathsf{Com.Commit}(\mathsf{ck}, \sigma; \mathsf{open}) \\ \wedge \mathsf{Ver}(\mathsf{svk}, \mathsf{H}(\mathbf{A}_j, \mathsf{x}', \mathsf{t}), \sigma) = 1 \\ \wedge \mathsf{link}(\mathsf{x}, \mathbf{A}_i, \mathsf{x}', \mathbf{A}_j) = 1 \end{array} \right. \end{array} \right\},$$

where $(\mathsf{svk}, \mathsf{ssk}) \xleftarrow{\$} \mathsf{SIG.KGen}(1^\lambda)$ and $\mathsf{ck} \xleftarrow{\$} \mathsf{COM.Setup}(1^\lambda)$.

Remark 1. In our instantiation using the Groth-Sahai proof system [GS08], the use of commitment $\mathbf{C_s}$ in the extended language definition is equivalent to the extractability property of GS08. Specifically, if an adversary can produce a valid proof for $(\mathsf{x}, \mathbf{C_s}) \in \mathcal{L}_{\mathbf{A}_i}^{(\mathsf{svk},\mathsf{ck})}$ where $\mathsf{x} \notin \mathcal{L}_{\mathbf{A}_i}$, then by the extractability of GS08, one can extract witnesses σ, open such that $\mathbf{C_s} = \mathsf{Com.Commit}(\mathsf{ck}, \sigma; \mathsf{open})$ and $\mathsf{Ver}(\mathsf{svk}, \mathsf{H}(\mathbf{A}_j, \mathsf{x}', \mathsf{t}), \sigma) = 1$ for some $\mathbf{A}_j, \mathsf{x}'$ with $\mathsf{link}(\mathsf{x}, \mathbf{A}_i, \mathsf{x}', \mathbf{A}_j) = 1$. The commitment mechanism thus transforms the existential quantification over signatures into a concrete extractable witness, which is essential for the security reduction to work properly in the GS framework.

Theorem 4. *Let* $\mathsf{SIG} = (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$ *be a signature scheme. Consider a language-malleable* tLM-NIZK *proof system* $\mathbf{\Pi}_{\mathsf{SIG}}$ *for the extended language family* $\{\mathcal{L}_{\mathbf{A}_i}^{\mathsf{svk},\mathsf{ck}}\}_i$ *with respect to the function family* $\mathcal{H}$*. If* $\mathbf{\Pi}_{\mathsf{SIG}}$ *is computationally sound, it can be interpreted as a special simulation-sound* tLM-NIZK *proof system, denoted* $\mathbf{\Pi}$*, for the linear language family* $\{\mathcal{L}_{\mathbf{A}_i}\}_i$*. More formally, for any* PPT *adversary* $\mathcal{A}$*, there exist* PPT *adversaries* $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4, \mathcal{B}_5$ *such that:*

$$\begin{aligned} \mathsf{Adv}_{\mathbf{\Pi},\mathcal{A}}^{\mathsf{ZK}}(1^\lambda) &= \mathsf{Adv}_{\mathbf{\Pi}_{\mathsf{SIG}},\mathcal{B}_1}^{\mathsf{ZK}}(1^\lambda) + \mathsf{Adv}_{\mathsf{COM},\mathcal{B}_4}^{\mathsf{Hiding}}(1^\lambda), \\ \mathsf{Adv}_{\mathbf{\Pi},\mathcal{A}}^{\mathsf{SSS}}(1^\lambda) &\leq \mathsf{Adv}_{\mathbf{\Pi}_{\mathsf{SIG}},\mathcal{B}_2}^{\mathsf{Sound}}(1^\lambda) + \mathsf{Adv}_{\mathsf{SIG},\mathcal{B}_3}^{\mathsf{UF\text{-}CMA}}(1^\lambda) + \mathsf{Adv}_{\mathsf{COM},\mathcal{B}_5}^{\mathsf{Binding}}(1^\lambda). \end{aligned}$$

Consequently, if $\mathbf{\Pi}_{\mathsf{SIG}}$ *is computationally sound and* SIG *is* UF-CMA *(resp., one-time* UF-CMA*), then* $\mathbf{\Pi}$ *is special simulation-sound (resp., one-time special simulation-sound).*

Proof. The correctness and language malleability of $\mathbf{\Pi}$ are direct consequences of the corresponding properties of $\mathbf{\Pi}_{\mathsf{SIG}}$, since a proof for the language $\mathcal{L}_{\mathbf{A}_i}$ is a specific instance of a proof for the extended language $\mathcal{L}^{\mathsf{svk},\mathsf{ck}}_{\mathbf{A}_i}$. Specifically, an honest proof for $\mathcal{L}_{\mathbf{A}_i}$ satisfies the first clause in the OR condition defining $\mathcal{L}^{\mathsf{svk},\mathsf{ck}}_{\mathbf{A}_i}$. Therefore, we omit the details for the correctness and language malleability.

We give more details for the zero-knowledge and special simulation-soundness property.

Zero-Knowledge. We construct a simulator for $\mathbf{\Pi}$. The simulation key for $\mathbf{\Pi}$ is the signing key ssk of the signature scheme SIG. To simulate a proof for a statement $\mathsf{x} \in \mathcal{L}_{\mathbf{A}_i}$ with tag t, the simulator first computes a signature σ on the hash value $\mathsf{hv} := \mathsf{H}(\mathbf{A}_i, \mathsf{x}, \mathsf{t})$ using its knowledge of ssk. It then commits to σ to obtain $\mathbf{C_s}$. Since the identity function is in $\mathcal{H}$, any statement is linked to itself, i.e., $\mathsf{link}(\mathsf{x}, \mathbf{A}_i, \mathsf{x}, \mathbf{A}_i) = 1$. Thus, the simulator can use the signature σ as a witness for the second clause of the extended language $\mathcal{L}^{\mathsf{svk},\mathsf{ck}}_{\mathbf{A}_i}$. It then invokes the honest prover of $\mathbf{\Pi}_{\mathsf{SIG}}$ to generate a proof for the extended statement $(\mathsf{x}, \mathbf{C_s})$. The indistinguishability of the simulated proof from a real one follows from the zero-knowledge property of $\mathbf{\Pi}_{\mathsf{SIG}}$ and the hiding property of the commitment scheme COM.

Special Simulation Soundness. The special simulation soundness of $\mathbf{\Pi}$ relies on the computational soundness of $\mathbf{\Pi}_{\mathsf{SIG}}$, the unforgeability of the signature scheme SIG, and the binding property of the commitment scheme COM. Suppose an adversary $\mathcal{A}$ breaks the special simulation soundness of $\mathbf{\Pi}$. It outputs a valid proof $(\mathsf{t}^\star, \pi^\star)$ for a false statement $\mathsf{x}^\star \notin \mathcal{L}_{\mathsf{lpk}^\star}$, where the pair $(\mathsf{lpk}^\star, \mathsf{x}^\star)$ is not linked to any pair $(\mathsf{lpk}, \mathsf{x})$ from the simulation oracle queries. Since $\mathsf{x}^\star$ is a false statement, the witness for the proof $\pi^\star$ must satisfy the second clause of the extended language $\mathcal{L}^{\mathsf{svk},\mathsf{ck}}_{\mathsf{lpk}^\star}$. This implies that the proof contains a commitment $\mathbf{C_s}$ to a valid signature σ on $\mathsf{hv}^\star = \mathsf{H}(\mathbf{A}_j, \mathsf{x}', \mathsf{t}^\star)$ for some linked pair $(\mathbf{A}_j, \mathsf{x}')$. Because $(\mathsf{lpk}^\star, \mathsf{x}^\star)$ is not linked to any prior query, the hash value $\mathsf{hv}^\star$ has not been signed by the simulation oracle. This constitutes a forgery against the signature scheme SIG. A formal reduction can be constructed where an adversary breaking the special simulation soundness of $\mathbf{\Pi}$ is used to break either the computational soundness of $\mathbf{\Pi}_{\mathsf{SIG}}$, the unforgeability of SIG, or the binding property of COM.

□

Language Family Specification To construct our aCCA-secure proxy re-encryption schemes, we first specify the language family $\{\mathcal{L}_{\mathbf{A}_i}\}_i$ and the corresponding transformation functions $\mathcal{H}$. Let $\mathsf{N_U}$ be a positive integer. The construction involves a set of matrices $\{\mathbf{A}_i\}_{i\in[\mathsf{N_U}]}$, where each $\mathbf{A}_i \in \mathbb{Z}_{\mathsf{p}}^{(\ell+1)\times k}$. Each matrix $\mathbf{A}_i$ is partitioned into its top ℓ rows, denoted by $\bar{\mathbf{A}}_i \in \mathbb{Z}_{\mathsf{p}}^{\ell\times k}$, and its bottom row, denoted by $\underline{\mathbf{A}}_i \in \mathbb{Z}_{\mathsf{p}}^{1\times k}$.

All matrices $\mathbf{A}_i$ for $i \in [\mathsf{N_U}]$ share a common sub-matrix $\bar{\mathbf{A}} \in \mathbb{Z}_{\mathsf{p}}^{\ell\times k}$ comprising their top ℓ rows. The bottom row $\underline{\mathbf{A}}_i$ of each matrix $\mathbf{A}_i$ is defined based on a

common vector $\mathbf{x} \leftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{k}\right)$ and a set of unique vectors $\{\bar{\mathbf{k}}_i \leftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{\ell}\right)\}_{i\in[\mathsf{N}_\mathsf{U}]}$. Specifically, for each $i \in [\mathsf{N}_\mathsf{U}]$, the bottom row is computed as $\underline{\mathbf{A}}_i := \mathbf{x}^\intercal + \bar{\mathbf{k}}_i^\intercal \bar{\mathbf{A}}$.

With these definitions, we can specify the language family and the transformation functions. For each $i \in [\mathsf{N}_\mathsf{U}]$, we define a vector $\mathbf{k}_i := (\bar{\mathbf{k}}_i^\intercal, 1)^\intercal \in \mathbb{Z}_{\mathsf{p}}^{\ell+1}$. The transformation from index i to j is determined by the difference vector $(d_1, \ldots, d_\ell, 0)^\intercal := \mathbf{k}_j - \mathbf{k}_i$. This vector is used to construct a transformation matrix $\boldsymbol{\Delta}_{i,j} \in \mathbb{Z}_{\mathsf{p}}^{(\ell+1)\times(\ell+1)}$ as follows:

$$\boldsymbol{\Delta}_{i,j} := \begin{pmatrix} 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \\ d_1 & d_2 & \ldots & d_\ell & 1 \end{pmatrix}.$$

By construction, this matrix maps $\mathbf{A}_i$ to $\mathbf{A}_j$, i.e., $\mathbf{A}_j = \boldsymbol{\Delta}_{i,j}\mathbf{A}_i$. The language family and transformation functions are then defined as follows:

- The language $\mathcal{L}_{\mathbf{A}_i}$ is the linear span language defined by

$$\mathcal{L}_{\mathbf{A}_i} := \{\mathsf{x} \in \mathbb{G}_1^{\ell+1} \mid \exists \mathbf{r} \in \mathbb{Z}_{\mathsf{p}}^{k} : \mathsf{x} = \left[\mathbf{A}_i\right]_1 \mathbf{r}\}.$$

- The transformation function family $\mathcal{H} = \{(\mathsf{H}_{i,j}^{\mathsf{lpk}}, \mathsf{H}_{i,j}^{\mathsf{x}}, \mathsf{H}_{i,j}^{\mathsf{w}})\}_{i,j\in[\mathsf{N}_\mathsf{U}]}$ consists of:
  - A language public key transformation $\mathsf{H}_{i,j}^{\mathsf{lpk}} : \mathbb{Z}_{\mathsf{p}}^{(\ell+1)\times k} \to \mathbb{Z}_{\mathsf{p}}^{(\ell+1)\times k}$ that maps $\mathbf{A}_i$ to $\mathbf{A}_j$ via left-multiplication by $\boldsymbol{\Delta}_{i,j}$:

$$\mathsf{H}_{i,j}^{\mathsf{lpk}}(\mathbf{A}_i) := \boldsymbol{\Delta}_{i,j}\mathbf{A}_i = \mathbf{A}_j.$$

  - A statement transformation $\mathsf{H}_{i,j}^{\mathsf{x}} : \mathbb{G}_1^{\ell+1} \to \mathbb{G}_1^{\ell+1}$ that maps a statement $\mathsf{x}_i \in \mathcal{L}_{\mathbf{A}_i}$ to a statement $\mathsf{x}_j \in \mathcal{L}_{\mathbf{A}_j}$ via left-multiplication by $\boldsymbol{\Delta}_{i,j}$:

$$\mathsf{H}_{i,j}^{\mathsf{x}}(\mathsf{x}_i) := \boldsymbol{\Delta}_{i,j}\mathsf{x}_i = \mathsf{x}_j.$$

  - A witness transformation $\mathsf{H}_{i,j}^{\mathsf{w}} : \mathbb{Z}_{\mathsf{p}}^{k} \to \mathbb{Z}_{\mathsf{p}}^{k}$ that maps a witness w_i to a witness w_j via identity function:

$$\mathsf{H}_{i,j}^{\mathsf{w}}(\mathsf{w}_i) := \mathsf{w}_i = \mathsf{w}_j.$$

We provide the full instantiation of our one-time and unbounded special simulation-sound tLM-NIZK proof system in Supplementary Material Section D. The construction is based on a Groth-Sahai proof for the language $\mathcal{L}_{\mathbf{A}_i}^{(\mathsf{svk},\mathsf{ck})}$. The language malleability property is a direct consequence of the linearity inherent in the Groth-Sahai framework.

D Instantiation of Special Simulation Sound tLM-NIZK

To instantiate the generic construction from C, we first fix the following public parameters and commitment algorithms.

To commit a vector $\mathbf{w} \in \mathbb{Z}_{\mathsf{p}}^{n}$ or $\left[\mathbf{w}\right] \in \mathbb{G}^{n}$, we proceed as follows:

Public parameters: Sample $\mathbf{D} \hookleftarrow \mathcal{D}_k$, choose $\mathbf{z}, \mathbf{y} \in \mathbb{Z}_{\mathsf{p}}^{k+1}$, and set $\mathbf{U} := \left[\mathbf{D} | \mathbf{z}\right]$. We consider two parameter regimes:

- Perfectly Binding Mode:
  Scalar vectors: $\mathbf{z} \notin \mathsf{Span}(\mathbf{D})$.
  Group vectors: $\mathbf{z} \in \mathsf{Span}(\mathbf{D})$ and $\mathbf{y} \notin \mathsf{Span}(\mathbf{D})$.
- Perfectly Hiding Mode:
  Scalar vectors: $\mathbf{z} \in \mathsf{Span}(\mathbf{D})$.
  Group vectors: $\mathbf{z} \notin \mathsf{Span}(\mathbf{D})$ and $\mathbf{y} \in \mathsf{Span}(\mathbf{D})$.

Commitments: Given the parameters above, commitments are computed as follows:

- Scalar $\mathbf{w} \in \mathbb{Z}_{\mathsf{p}}^{n}$: $\left[\mathbf{C}_{\mathbf{w}}\right] := \left[\mathbf{D}\right] \mathbf{R} + \left[\mathbf{z}\right] \mathbf{w}^{\intercal}$, where we sample $\mathbf{R} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{k \times n}\right)$.
- Group vector $\left[\mathbf{w}\right] \in \mathbb{G}^{n}$: $\left[\mathbf{C}_{\mathbf{w}}\right] := \left[\mathbf{U}\right] \mathbf{R} + \mathbf{y} \left[\mathbf{w}^{\intercal}\right]$, where we sample $\mathbf{R} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{(k+1) \times n}\right)$.

We use subscripts to indicate whether the public parameters are instantiated in $\mathbb{G}_1$ or $\mathbb{G}_2$.

D.1 One-Time Special Simulation Sound tLM-NIZK

To instantiate the one-time special simulation sound $\mathsf{tLM\text{-}NIZK}$ with tight security, we proceed by the following two steps:

- We use the tightly-secure one-time structure-preserving signature $\mathbf{\Sigma}_{\mathsf{OT}}$ given in [KPW15] to instantiate the underlying signature scheme.
- We use the Groth-Sahai proof system [GS08] to instantiate the language-malleable $\mathsf{tLM\text{-}NIZK}$ proof system $\mathbf{\Pi}_{\mathsf{SIG}}$ for the extended language family $\{\mathcal{L}_{\mathbf{A}_i}^{\mathsf{svk}}\}_i$ with respect to the function family $\mathcal{H}$.

We first recall the signature $\mathbf{\Sigma}_{\mathsf{OT}} = (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$ in Figure 10. To be compatible with the other part of the $\mathsf{tLM\text{-}NIZK}$, we exchange $\mathbb{G}_1$ and $\mathbb{G}_2$ in the original scheme. Due to the symmetric nature of pairing computation, this modification does not affect the correctness and security of the scheme.

Alg $\mathsf{KGen}(1^{\lambda})$:
01 $\mathbf{B} \hookleftarrow \mathcal{D}_k$
02 $\mathbf{K} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{(n+1) \times (k+1)}\right)$
03 $\mathbf{C} = \mathbf{K}\mathbf{B}$
04 $\mathsf{svk} := \left(\left[\mathbf{B}^{\intercal}\right]_1, \left[\mathbf{C}^{\intercal}\right]_1\right)$
05 $\mathsf{ssk} := \mathbf{K}$
06 **return** $(\mathsf{svk}, \mathsf{ssk})$

Alg $\mathsf{Sig}(\mathsf{ssk}, \left[\mathsf{m}\right]_2 \in \mathbb{G}_2^n)$:
07 $\left[\sigma\right]_2 := \left[\left(1 \ \mathsf{m}\right) \mathbf{K}\right]_2^{\intercal}$
08 **return** $\left[\sigma\right]_2 \in \mathbb{G}_2^{k+1}$

Alg $\mathsf{Ver}(\mathsf{svk}, \mathsf{m}, \left[\sigma\right]_2)$:
09 **check** $\left[\mathbf{B}^{\intercal}\right]_1 \bullet \left[\sigma\right]_2 = \left[\mathbf{C}^{\intercal}\right]_1 \bullet \left[\left(1 \ \mathsf{m}\right)^{\intercal}\right]_2$

Fig. 10. Construction of the one-time structure-preserving signature scheme $\mathbf{\Sigma}_{\mathsf{OT}} = (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Ver})$ [KPW15].

Theorem 5 ([KPW15]). *The signature scheme* $\mathbf{\Sigma}_{\mathsf{OT}}$ *in Figure 10 is one-time* UF-CMA. *Concretely, for all* PPT *adversaries* $\mathcal{A}$*, there exists a* PPT *adversary* $\mathcal{B}$ *with* $\mathbf{T}(\mathcal{A}) \approx \mathbf{T}(\mathcal{B})$ *and*

$$\mathsf{Adv}^{\mathsf{OT\text{-}CMA}}_{\mathbf{\Sigma}_{\mathsf{OT}},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{KerMDH}}_{\mathcal{D}_k,\mathsf{GGen},\mathcal{B}}(\lambda) + 1/\mathsf{p}.$$

We translate the statements of $\mathcal{L}^{\mathsf{svk}}_{\mathbf{A}_i}$ as follows:

- $\left[\mathbf{A}_i\right]_1 \bullet \boxed{\left[\mathbf{w}\right]_2} - \left[\mathsf{x}\right]_1 \bullet \left[1\right]_2 = \left[\mathbf{0}\right]_1$[18] // $\mathsf{x} \in \mathcal{L}_{\mathbf{A}_i}$
- $\left[\underline{\mathbf{A}}_i^{\intercal}\right]_1 - \left[(\underline{\mathbf{A}}^{\star})^{\intercal}\right]_1 - \left[\bar{\mathbf{A}}^{\intercal}\right]_1 \boxed{\bar{\mathbf{k}}^{\star}} = \left[0\right]_1$ // $\mathsf{link}(\mathbf{A}_i, \mathsf{x}, \mathbf{A}^{\star}, \mathsf{x}^{\star}) = 1$
- $\left[\underline{\mathsf{x}}^{\intercal}\right]_1 - \left[(\underline{\mathsf{x}}^{\star})^{\intercal}\right]_1 - \left[\bar{\mathsf{x}}^{\intercal}\right]_1 \boxed{\bar{\mathbf{k}}^{\star}} = \left[0\right]_1$ // $\mathsf{link}(\mathbf{A}_i, \mathsf{x}, \mathbf{A}^{\star}, \mathsf{x}^{\star}) = 1$
- $\left[\mathbf{B}^{\intercal}\right]_1 \bullet \boxed{\left[\sigma\right]_2} = \left[\mathbf{C}^{\intercal}\right]_1 \bullet \begin{bmatrix} 1 \\ \mathsf{hv} \end{bmatrix}_2$ // $\mathsf{Ver}(\mathsf{svk}, \mathsf{hv}, \sigma) = 1$

where $\mathsf{hv} = \mathsf{H}(\underline{\mathbf{A}}^{\star}, \underline{\mathsf{x}}^{\star}, \mathsf{t})$.

Remark 2. In our proxy re-encryption setting, a ciphertext consists of three components $(\mathsf{ct}_0, \mathsf{ct}_1, \mathsf{ct}_2)$, where $(\mathsf{ct}_0, \mathsf{ct}_1)$ together form the statement x used in this proof system, with ct_1 corresponding to the bottom row $\underline{\mathsf{x}}$. A complete link verification should ensure: (1) the transformational relationship of ct_1 (the bottom component), and (2) the invariance of ct_0 and ct_2 between linked ciphertexts.

The hash function and link verification equations above only address the bottom row relationships, corresponding to the ct_1 component. However, in our CCA construction, we employ an external signature scheme to authenticate ct_0 and ct_2, ensuring their invariance across linked ciphertexts. Since the hash value incorporates the tag t, any signature reuse necessarily implies tag reuse, which would be detected by the signature verification in the outer CCA protocol. Therefore, this tLM-NIZK system focuses solely on proving the correct transformation of ct_1, while the complete link property is jointly guaranteed by this proof system and the external signature scheme.

We prove the first equation using the vector $\mathbf{z}_{\mathbf{A}}$ to commit elements in $\mathbb{G}_2$, and the last three equations using $\mathbf{z}_{\sigma}$. Moreover, $\mathbf{z} = \mathbf{z}_{\mathbf{A}} + \mathbf{z}_{\sigma} \notin \mathsf{Span}(\mathbf{D})$, which ensures that either the first equation or the last three equations are proved in the soundness setting. All commitments in $\mathbb{G}_1$ are in the perfectly binding mode. The commitments are as follows:

- $\left[\mathbf{C}_{\mathbf{w}}\right]_2 := \left[\mathbf{D}\right]_2 \mathbf{R}_{\mathbf{w}}^{\intercal} + \left[\mathbf{z}_{\mathbf{A}}\right]_2 \mathbf{w}^{\intercal}$
- $\left[\mathbf{C}_{\mathbf{k}}\right]_2 := \left[\mathbf{D}\right]_2 \mathbf{R}_{\mathbf{k}}^{\intercal} + \left[\mathbf{z}_{\sigma}\right]_2 (\bar{\mathbf{k}}^{\star})^{\intercal}$
- $\left[\mathbf{C}_{\sigma}\right]_2 := \left[\mathbf{D}\right]_2 \mathbf{R}_{\sigma}^{\intercal} + \left[\mathbf{z}_{\sigma}\right]_2 \sigma^{\intercal}$

The verification equations are given as follows:

[18] Note that x inherently consists of group elements in $\mathbb{G}_1$, but we explicitly use the notation $[\cdot]_1$ here to emphasize its group properties for clarity. We adopt this consistent notation throughout the subsequent presentation.

- $\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}_{\mathbf{w}}^\intercal\right]_2 - \left[\mathsf{x}\right]_1 \bullet \left[\mathbf{z}_{\mathbf{A}}^\intercal\right]_2 = \left[\mathbf{P}_{\mathbf{A}}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$
- $\left[\underline{\mathbf{A}}_i^\intercal\right]_1 \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1 \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_{\mathbf{k}}^\intercal\right]_2 = \left[\mathbf{P}_{\mathbf{d}}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$
- $\left[\underline{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1 \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_{\mathbf{k}}^\intercal\right]_2 = \left[\mathbf{P}_{\mathsf{x}}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$
- $\left[\mathbf{B}^\intercal\right]_1 \bullet \left[\mathbf{C}_\sigma^\intercal\right]_2 - \left[\mathbf{C}^\intercal\right]_1 \bullet \left[\binom{1}{\mathsf{hv}} \mathbf{z}_\sigma^\intercal\right]_2 = \left[\mathbf{P}_\sigma\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$

We give the full construction of the one-time special simulation sound tLM-NIZK proof system in Figure 11. We show in Figure 12 that the construction is malleable with respect to the transformation set $\mathcal{H}$.

Alg $\mathsf{Setup}(1^\lambda)$:
01 $\mathbf{D} \leftarrow \mathcal{D}_k$
02 $\mathbf{z} \leftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D})\right)$
03 $\mathbf{B} \leftarrow \mathcal{D}_k$; $\mathbf{K} \leftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{2\times(k+1)}\right)$
04 $\mathbf{C} = \mathbf{K}\mathbf{B}$
05 $\mathsf{crs} := (\left[\mathbf{D}\right]_2, \left[\mathbf{z}\right]_2, \left[\mathbf{B}^\intercal\right]_1, \left[\mathbf{C}^\intercal\right]_1)$
06 **return** crs

Alg $\mathsf{Ver}(\mathsf{crs}, \mathsf{t}, \mathsf{x}, \pi)$:
07 $\mathsf{hv} := \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$
08 $\left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_{\mathbf{A}}\right]_2$
09 **check** $\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}_{\mathbf{w}}^\intercal\right]_2 - \left[\mathsf{x}\right]_1 \bullet \left[\mathbf{z}_{\mathbf{A}}^\intercal\right]_2 = \left[\mathbf{P}_{\mathbf{A}}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$
10 **check** $(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_{\mathbf{k}}^\intercal\right]_2 = \left[\mathbf{P}_{\mathbf{d}}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$
11 **check** $(\left[\underline{\mathsf{x}}^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_{\mathbf{k}}^\intercal\right]_2 = \left[\mathbf{P}_{\mathsf{x}}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$
12 **check** $\left[\mathbf{B}^\intercal\right]_1 \bullet \left[\mathbf{C}_\sigma^\intercal\right]_2 - \left[\mathbf{C}^\intercal\right]_1 \bullet \left[\binom{1}{\mathsf{hv}} \mathbf{z}_\sigma^\intercal\right]_2 = \left[\mathbf{P}_\sigma\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2$
13 **return** 1

Alg $\mathsf{Prove}(\mathsf{crs}, \mathsf{t}, \mathsf{x}, \mathbf{w})$:
14 $\left[\underline{\mathbf{A}}^\star\right]_1 \leftarrow \mathsf{U}\left(\mathbb{G}_1^{1\times k}\right)$
15 $\left[\underline{\mathsf{x}}^\star\right]_1 \leftarrow \mathsf{U}\left(\mathbb{G}_1^{1\times 1}\right)$
16 $\mathsf{hv} = \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$
17 $\mathbf{v}_\sigma \leftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^k\right)$; $\left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{D}\right]_2 \mathbf{v}_\sigma$
18 $\left[\mathbf{z}_{\mathbf{A}}\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\sigma\right]_2$
19 $\mathbf{R}_{\mathbf{w}} \leftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k\times k}\right)$
20 $\mathbf{R}_{\mathbf{k}}, \mathbf{R}_\sigma \leftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times k}\right)$
21 $\left[\mathbf{C}_{\mathbf{w}}^\intercal\right]_2 := \mathbf{R}_{\mathbf{w}} \left[\mathbf{D}^\intercal\right]_2 + \mathbf{w} \left[\mathbf{z}_{\mathbf{A}}^\intercal\right]_2$
22 $\left[\mathbf{C}_{\mathbf{k}}^\intercal\right]_2 := \mathbf{R}_{\mathbf{k}} \left[\mathbf{D}^\intercal\right]_2$
23 $\left[\mathbf{C}_\sigma^\intercal\right]_2 := \mathbf{R}_\sigma \left[\mathbf{D}^\intercal\right]_2$
24 $\left[\mathbf{P}_{\mathbf{A}}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_{\mathbf{w}}$
25 $\left[\mathbf{P}_{\mathbf{d}}\right]_1 := (\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1)\mathbf{v}_\sigma^\intercal - \left[\bar{\mathbf{A}}^\intercal\right]_1 \mathbf{R}_{\mathbf{k}}$
26 $\left[\mathbf{P}_{\mathsf{x}}\right]_1 := (\left[\underline{\mathsf{x}}^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1)\mathbf{v}_\sigma^\intercal - \left[\bar{\mathsf{x}}^\intercal\right]_1 \mathbf{R}_{\mathbf{k}}$
27 $\left[\mathbf{P}_\sigma\right]_1 := \left[\mathbf{B}^\intercal\right]_1 \mathbf{R}_\sigma - \left[\mathbf{C}^\intercal\right]_1 \binom{1}{\mathsf{hv}} \mathbf{v}_\sigma^\intercal$
28 $\pi := (\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \left[\mathbf{z}_{\mathbf{A}}\right]_2, \left[\mathbf{C}_{\mathbf{w}}\right]_2, \left[\mathbf{C}_{\mathbf{k}}\right]_2, \left[\mathbf{C}_\sigma\right]_2, \left[\mathbf{P}_{\mathbf{A}}\right]_1, \left[\mathbf{P}_{\mathbf{d}}\right]_1, \left[\mathbf{P}_{\mathsf{x}}\right]_1, \left[\mathbf{P}_\sigma\right]_1)$
29 **return** π

Fig. 11. Construction of one-time special simulation sound tLM-NIZK proof system.

We give the language malleability algorithm in Figure 12 and the construction of the simulator $\mathcal{S} = (\mathsf{SSetup}, \mathsf{Sim})$ for the $\mathbf{\Pi}_{\mathsf{OT}}$ scheme as detailed in Figure 13.

The correctness of the proof system $\mathbf{\Pi}_{\mathsf{OT}}$ is straightforward. We provide tight proofs of the zero-knowledge property in Theorem 6 and the special simulation soundness of $\mathbf{\Pi}_{\mathsf{OT}}$ in Theorem 7. Detailed proofs of both theorems can be found in Supplementary Material Section E.

Theorem 6 (Zero-Knowledge). *The one-time* tLM-NIZK *proof system in Figure 11 is zero-knowledge. Concretely, for all* PPT *adversaries* $\mathcal{A}$*, there exists a*

$\underline{\textbf{Alg } \mathsf{Trans}(\mathsf{crs}, \mathsf{t}, \mathsf{x}, \pi, \mathsf{d}_{i,j})}$:

$\mathbf{parse}\ (\left[\underline{\mathbf{A}}^{\star}\right]_1, \left[\underline{\mathbf{x}}^{\star}\right]_1, \left[\mathbf{z}_{\mathbf{A}}\right]_2, \left[\mathbf{C}_{\mathbf{w}}\right]_2, \left[\mathbf{C}_{\mathbf{k}}\right]_2, \left[\mathbf{C}_{\sigma}\right]_2, \left[\mathbf{P}_{\mathbf{A}}\right]_1, \left[\mathbf{P}_{\mathbf{d}}\right]_1, \left[\mathbf{P}_{\mathsf{x}}\right]_1, \left[\mathbf{P}_{\sigma}\right]_1) =: \pi$

$\left[\mathbf{z}_{\sigma}\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_{\mathbf{A}}\right]_2$

$\left[\mathbf{P}_{\mathbf{A}}\right]_1' := \boldsymbol{\Delta}_{i,j} \left[\mathbf{P}_{\mathbf{A}}\right]_1;\ \left[(\mathbf{C}_{\mathbf{k}}')^{\intercal}\right]_2 := \left[\mathbf{C}_{\mathbf{k}}^{\intercal}\right]_2 + \bar{\mathsf{d}}_{i,j} \left[\mathbf{z}_{\sigma}^{\intercal}\right]_2$

$\pi' = (\left[\underline{\mathbf{A}}^{\star}\right]_1, \left[\underline{\mathbf{x}}^{\star}\right]_1, \left[\mathbf{z}_{\mathbf{A}}\right]_2, \left[\mathbf{C}_{\mathbf{w}}\right]_2, \left[\mathbf{C}_{\mathbf{k}}\right]_2', \left[\mathbf{C}_{\sigma}\right]_2, \left[\mathbf{P}_{\mathbf{A}}\right]_1', \left[\mathbf{P}_{\mathbf{d}}\right]_1, \left[\mathbf{P}_{\mathsf{x}}\right]_1, \left[\mathbf{P}_{\sigma}\right]_1)$

$\mathbf{return}\ \pi'$

Fig. 12. Language malleability of the one-time special simulation sound tLM-NIZK proof system. Notice that $\boldsymbol{\Delta}_{i,j}$ can be efficiently computed from $\mathsf{d}_{i,j}$.

$\underline{\textbf{Alg } \mathsf{SSetup}(1^{\lambda})}$:

$\mathbf{D} \leftarrow\!\!\$\ \mathcal{D}_k$

$\mathbf{z} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{k+1} \setminus \mathsf{Span}(\mathbf{D})\right)$

$\mathbf{B} \leftarrow\!\!\$\ \mathcal{D}_k;\ \mathbf{K} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{2\times(k+1)}\right)$

$\mathbf{C} = \mathbf{K}\mathbf{B}$

$\mathsf{crs} := (\left[\mathbf{D}\right]_2, \left[\mathbf{z}\right]_2, \left[\mathbf{B}^{\intercal}\right]_1, \left[\mathbf{C}^{\intercal}\right]_1)$

$\mathsf{simK} := (\mathbf{K})$

$\mathbf{return}\ (\mathsf{crs}, \mathsf{simK})$

$\underline{\textbf{Alg } \mathsf{Sim}(\mathsf{crs}, \mathsf{simK}, \mathsf{t}, \mathsf{x})}$:

$\bar{\mathbf{k}}^{\star} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{k+1}\right)$

$\left[\underline{\mathbf{A}}^{\star}\right]_1 = \left[\underline{\mathbf{A}}_i\right]_1 - (\bar{\mathbf{k}}^{\star})^{\intercal} \left[\bar{\mathbf{A}}\right]_1$

$\left[\underline{\mathbf{x}}^{\star}\right]_1 = \left[\underline{\mathbf{x}}\right]_1 - (\bar{\mathbf{k}}^{\star})^{\intercal} \left[\bar{\mathbf{x}}\right]_1$

$\mathsf{hv} := \mathsf{H}(\left[\underline{\mathbf{A}}^{\star}\right]_1, \left[\underline{\mathbf{x}}^{\star}\right]_1, \mathsf{t})$

$\sigma := \mathbf{K}^{\intercal} \binom{1}{\mathsf{hv}}$

$\mathbf{v}_{\mathbf{A}} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{k}\right);\ \left[\mathbf{z}_{\mathbf{A}}\right]_2 := \left[\mathbf{D}\right]_2 \mathbf{v}_{\mathbf{A}}$

$\left[\mathbf{z}_{\sigma}\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_{\mathbf{A}}\right]_2$

$\mathbf{R}_{\mathbf{w}} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{k\times k}\right)$

$\mathbf{R}_{\mathbf{k}}, \mathbf{R}_{\sigma} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{(k+1)\times k}\right)$

$\left[\mathbf{C}_{\mathbf{w}}^{\intercal}\right]_2 := \mathbf{R}_{\mathbf{w}} \left[\mathbf{D}^{\intercal}\right]_2$

$\left[\mathbf{C}_{\mathbf{k}}^{\intercal}\right]_2 := \mathbf{R}_{\mathbf{k}} \left[\mathbf{D}^{\intercal}\right]_2 + \bar{\mathbf{k}}^{\star} \left[\mathbf{z}_{\sigma}^{\intercal}\right]_2$

$\left[\mathbf{C}_{\sigma}^{\intercal}\right]_2 := \mathbf{R}_{\sigma} \left[\mathbf{D}^{\intercal}\right]_2 + \sigma \left[\mathbf{z}_{\sigma}^{\intercal}\right]_2$

$\left[\mathbf{P}_{\mathbf{A}}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_{\mathbf{w}} - \left[\mathsf{x}\right]_1 \mathbf{v}_{\mathbf{A}}^{\intercal}$

$\left[\mathbf{P}_{\mathbf{d}}\right]_1 := -\left[\bar{\mathbf{A}}^{\intercal}\right]_1 \mathbf{R}_{\mathbf{k}}$

$\left[\mathbf{P}_{\mathsf{x}}\right]_1 := -\left[\bar{\mathbf{x}}^{\intercal}\right]_1 \mathbf{R}_{\mathbf{k}}$

$\left[\mathbf{P}_{\sigma}\right]_1 := \left[\mathbf{B}^{\intercal}\right]_1 \mathbf{R}_{\sigma}$

$\pi := (\left[\underline{\mathbf{A}}^{\star}\right]_1, \left[\underline{\mathbf{x}}^{\star}\right]_1, \left[\mathbf{z}_{\mathbf{A}}\right]_2, \left[\mathbf{C}_{\mathbf{w}}\right]_2, \left[\mathbf{C}_{\mathbf{k}}\right]_2, \left[\mathbf{C}_{\sigma}\right]_2, \left[\mathbf{P}_{\mathbf{A}}\right]_1, \left[\mathbf{P}_{\mathbf{d}}\right]_1, \left[\mathbf{P}_{\mathsf{x}}\right]_1, \left[\mathbf{P}_{\sigma}\right]_1)$

$\mathbf{return}\ \pi$

Fig. 13. Simulator of the one-time special simulation sound tLM-NIZK proof system.

PPT *adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{ZK}}_{\boldsymbol{\Pi}_{\mathsf{OT}},\mathcal{A}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(\lambda).$$

Theorem 7 (Special Simulation Sound). *The one-time* tLM-NIZK *proof system presented in Figure 11 satisfies special simulation soundness.*

D.2 Special Simulation Sound tLM-NIZK

This section details the construction of an unbounded special simulation sound tLM-NIZK scheme $\mathbf{\Pi}_{\mathsf{U}}$, employing a framework similar to that used for the one-time special simulation sound tLM-NIZK construction in Figure 13. We replace the one-time signature scheme with an almost tightly secure UF-CMA signature scheme and change the number of rows in matrix $\bar{\mathbf{A}}$ from $k+1$ to ℓ. The detailed construction of the proof system $\mathbf{\Pi}_{\mathsf{U}}$ is presented in Figure 16.

We first recall the almost tightly secure UF-CMA signature scheme SIG of Gay et al. [GHKP18]. Its construction appears in Figure 14. The signature scheme is almost tightly secure under the $\mathcal{D}_k$-MDDH assumption. We modify the signature scheme to sign a message $\mathsf{m} \in \mathbb{Z}_{\mathsf{p}}$; the signature is of the form $\sigma = (\left[\mathbf{t}\right]_1, \pi, \left[\mathbf{s}\right]_1)$.

Alg $\mathsf{KGen}(1^\lambda)$:
01 $\mathbf{B}_0, \mathbf{B}_1 \hookleftarrow \mathcal{D}_{2k,k}$
02 $\mathsf{crs} \xleftarrow{\$} \Pi^{\vee}_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Setup}(1^\lambda)$
03 $\mathbf{B} \hookleftarrow \mathcal{D}_k$
04 $\mathbf{K}_0, \mathbf{K}_1 \hookleftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^{2k\times(k+1)}\right)$
05 $\mathsf{svk} := (\mathsf{crs}, \left[\mathbf{B}_0\right]_1, \left[\mathbf{B}\right]_2, \left[\mathbf{K}_0\mathbf{B}\right]_2, \left[\mathbf{K}_1\mathbf{B}\right]_2)$
06 $\mathsf{ssk} := (\mathbf{K}_0, \mathbf{K}_1)$
07 **return** $(\mathsf{svk}, \mathsf{ssk})$

Alg $\mathsf{Sig}(\mathsf{ssk}, \mathsf{m} \in \mathbb{Z}_{\mathsf{p}})$:
08 $\mathbf{r} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_{\mathsf{p}}^k\right);\ \left[\mathbf{t}\right]_1 := \left[\mathbf{B}_0\right]_1 \mathbf{r}$
09 $\pi \xleftarrow{\$} \Pi^{\vee}_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Prove}(\mathsf{crs}, \left[\mathbf{t}\right]_1, \mathbf{r})$
10 $\left[\mathbf{s}\right]_1 := (\mathbf{K}_0 + \mathsf{m}\mathbf{K}_1)^\intercal \left[\mathbf{t}\right]_1$
11 **return** $\sigma := (\left[\mathbf{t}\right]_1, \pi, \left[\mathbf{s}\right]_1)$

Alg $\mathsf{Ver}(\mathsf{svk}, \mathsf{m}, \sigma)$:
12 **check** $\Pi^{\vee}_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Ver}(\mathsf{crs}, \left[\mathbf{t}\right]_1, \pi)$
13 **check** $\left[\mathbf{s}\right]_1 \neq \left[\mathbf{0}\right]_1$
14 **check** $\left[\mathbf{s}^\intercal\right]_1 \bullet \left[\mathbf{B}\right]_2 = \left[\mathbf{t}^\intercal\right]_1 \bullet \left[(\mathbf{K_0} + \mathsf{m}\mathbf{K_1})\mathbf{B}\right]_2$
15 **return** 1

Fig. 14. Construction of tightly-secure structure preserving signature scheme. [GHKP18]

For the sake of completeness, we also add the description of the OR proof system $\Pi^{\vee}_{\mathbf{B}_0,\mathbf{B}_1}$ in Figure 15. We define the language as

$$\mathcal{L}^{\vee}_{\mathbf{B}_0,\mathbf{B}_1} = \left\{ \left[\mathbf{t}\right]_1 \mid \exists \mathbf{r} \in \mathbb{Z}_{\mathsf{p}}^k.\ \left[\mathbf{t}\right]_1 = \left[\mathbf{B}_0\right]_1 \mathbf{r} \vee \left[\mathbf{t}\right]_1 = \left[\mathbf{B}_1\right]_1 \mathbf{r} \right\}.$$

Theorem 8 ([GHKP18, Theorem 2]). *Let $\Pi^{\vee}_{\mathbf{B}_0,\mathbf{B}_1}$ be a non-interactive zero-knowledge proof. The signature scheme in Figure 14 is almost tightly secure under the $\mathcal{D}_k$-MDDH and the $\mathcal{D}_{2k,k}$-MDDH assumptions.*

Concretely, for every PPT *adversary $\mathcal{A}$, there exist* PPT *adversaries $\mathcal{B}_1, \ldots, \mathcal{B}_4$ with $\mathbf{T}(\mathcal{B}_i) \approx \mathbf{T}(\mathcal{A}) + Q \cdot \mathsf{poly}(\lambda)$ for every $i \in [4]$ such that*

$$\mathsf{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{core}}_{\mathcal{B}_2}(\lambda) + \frac{Q}{p}.$$

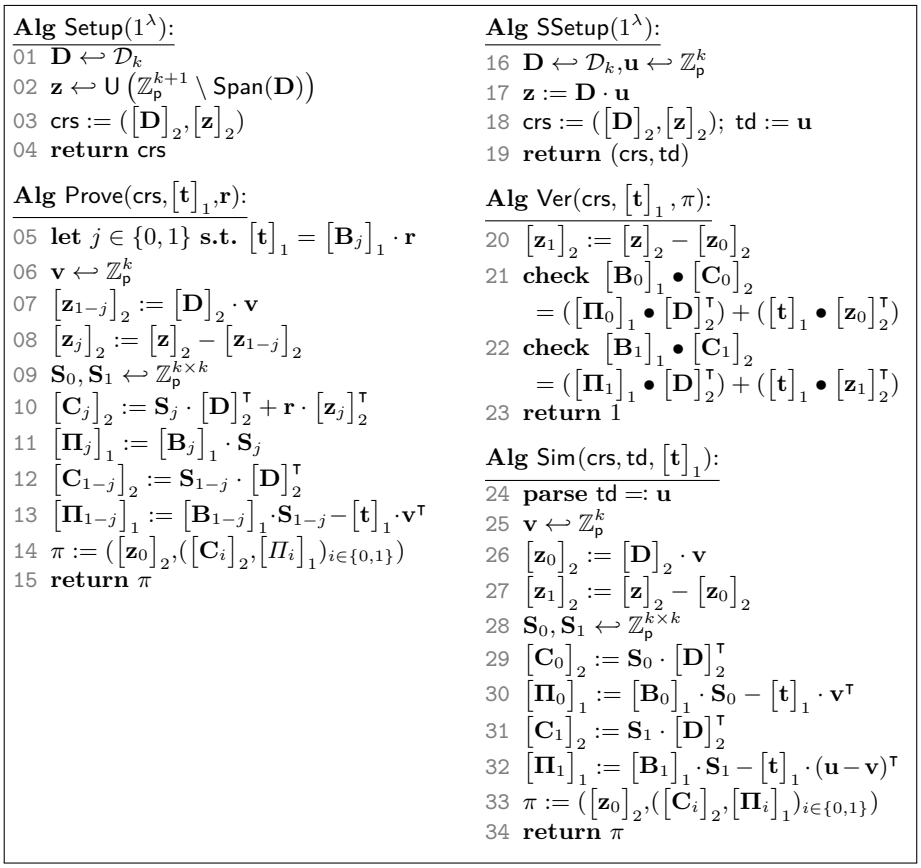

Fig. 15. NIZK argument for the OR language $\mathcal{L}^\vee_{\mathbf{B}_0,\mathbf{B}_1}$.

where

$$\begin{aligned}\mathsf{Adv}^{\mathsf{core}}_{\mathcal{B}_2}(\lambda) \coloneqq\ & (4k\lceil \log Q\rceil + 2)\cdot \mathsf{Adv}^{\mathsf{MDDH}}_{\mathbb{G}_1,\mathcal{D}_{2k,k},\mathcal{B}_3}(\lambda)\\ & + (2\lceil \log Q\rceil + 2)\cdot \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{PS},\mathcal{B}_4}(\lambda)\\ & + \lceil \log Q\rceil \cdot \varDelta_{\mathcal{D}_{2k,k}} + \tfrac{4\lceil \log Q\rceil+2}{p-1} + \tfrac{\lceil \log Q\rceil Q}{p}.\end{aligned}$$

We define $\mathbf{u}_i \in \mathbb{Z}_\mathsf{p}^{2k}$ as a vector with all entries equal to zero except for the i-th entry, which is 1. Let $\delta \in \{0,1\}$ and $\mathsf{hv} = \mathsf{H}(\underline{\mathbf{A}}^\star, \underline{\mathsf{x}}^\star, \mathsf{t}) \in \mathbb{Z}_\mathsf{p}$. To construct the special simulation-sound tLM-NIZK proof system, we establish the following relations:

- $\left[\mathbf{A}_i\right]_1 \ulcorner \bar{\mathbf{w}} \urcorner - \left[\mathsf{x}\right]_1 = \left[0\right]_\mathsf{T}$ // $\mathsf{x} \in \mathcal{L}_{\mathbf{A}_i}$
- $(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1) - \left[\bar{\mathbf{A}}^\intercal\right]_1 \ulcorner \bar{\mathbf{k}}^\star \urcorner = \left[0\right]_1$ // $\mathsf{link}(\mathbf{A}_i, \mathsf{x}, \mathbf{A}^\star, \mathsf{x}^\star)$
- $(\left[\underline{\mathsf{x}}^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1) - \left[\bar{\mathsf{x}}^\intercal\right]_1 \ulcorner \bar{\mathbf{k}}^\star \urcorner = \left[0\right]_1$ // $\mathsf{link}(\mathbf{A}_i, \mathsf{x}, \mathbf{A}^\star, \mathsf{x}^\star)$

– For $j \in [2k]$, $\left[\mathbf{t}^\intercal\right]_1 \bullet \left[\delta\mathbf{u}_j\right]_2 = \left[\delta\mathbf{t}^\intercal\right]_1 \bullet \left[\delta\mathbf{u}_j\right]_2$ // $\delta \in \{0,1\}$
– $\left[\mathbf{s}^\intercal\right]_1 \bullet \left[\mathbf{B}\right]_2 = \left[\delta\mathbf{t}^\intercal\right]_1 \bullet \left[(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2$ // $\mathsf{Ver}(\mathsf{svk}, \mathsf{hv}, \sigma) = 1$

To provide intuition for the last two verification equations, consider the goal of enabling the simulation of a proof that demonstrates knowledge of a signature for an honest user. As noted in [GS08], the Groth-Sahai proof for non-homogeneous bilinear equations requires modifications to facilitate simulation. Specifically, we introduce a binary variable $\delta \in \{0,1\}$ and its corresponding variable $\delta\mathbf{t}$ to restore homogeneity to the fourth equation. To ensure the well-formedness of the commitment to $\delta\mathbf{t}$, we incorporate the third equation. However, the quadratic nature of the third statement necessitates $2k$ distinct equations to complete the proof.

Analogous to the one-time special simulation sound tLM-NIZK proof system, we employ $\mathbf{z}_\mathbf{A}$ to prove the first statement and $\mathbf{z}_\sigma$ to prove the remaining four, ensuring that $\mathbf{z}_\mathbf{A} + \mathbf{z}_\sigma = \mathbf{z} \notin \mathsf{Span}(\mathbf{D})$. This condition enforces that either the first equation or the last four equations have perfect soundness. More specifically, we need to give the commitments $\left[\mathbf{C}_\mathbf{w}\right]_2, \left[\mathbf{C}_\mathbf{k}\right]_2, \left[\mathbf{C}_\mathbf{t}\right]_2, \left[\mathbf{C}_\mathbf{s}\right]_2$ and the proofs $\left[\mathbf{P}_\mathbf{A}\right]_1, \left[\mathbf{P}_\mathbf{d}\right]_1, \left[\mathbf{P}_\times\right]_1, \{\left[\mathbf{P}_\mathbf{t}^j\right]_1, \left[\mathbf{Q}_\mathbf{t}^j\right]_2\}_{j\in[2k]}, \left[\mathbf{Q}_\sigma\right]_2$ verifying the following equations:

– $\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}_\mathbf{w}^\intercal\right]_2 - \left[\underline{\mathsf{x}}\right]_1 \bullet \left[\mathbf{z}_\mathbf{A}^\intercal\right]_2 = \left[\mathbf{P}_\mathbf{A}\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2$
– $(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 = \left[\mathbf{P}_\mathbf{d}\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2$
– $(\left[\underline{\mathsf{x}}^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 = \left[\mathbf{P}_\times\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2$
– For $j \in [2k]$:
  $\left[\mathbf{y}_1\mathbf{t}^\intercal\right]_1 \bullet \left[\mathbf{u}_j\mathbf{z}_\sigma^\intercal\right]_2 - \left[\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[\mathbf{u}_j\mathbf{z}_\sigma^\intercal\right]_2 = \left[\mathbf{P}_\mathbf{t}^j\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2 + \left[\mathbf{U}_1\right]_1 \bullet \left[\mathbf{Q}_\mathbf{t}^j\right]_2$
– $\left[\mathbf{C}_\mathbf{s}\right]_1 \bullet \left[\mathbf{B}\right]_2 - \left[\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2 = \left[\mathbf{U}_1\right]_1 \bullet \left[\mathbf{Q}_\sigma\right]_2$

The construction of the special simulation-sound tLM-NIZK proof system is presented in Figure 16. This proof system achieves almost tight security under the $\mathcal{D}_k$-MDDH assumption. Additionally, the detailed simulator $\mathcal{S} = (\mathsf{SSetup}, \mathsf{Sim})$ for the proof system $\mathbf{\Pi}_\mathsf{U}$ is provided in Figure 17.

The language-malleability algorithm for $\mathbf{\Pi}_\mathsf{U}$ is identical to that for $\mathbf{\Pi}_\mathsf{OT}$ in Figure 12: given $\mathsf{d}_{i,j}$, it computes $\left[\mathbf{z}_\sigma\right]_2 = \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\mathbf{A}\right]_2$, sets $\left[\mathbf{P}_\mathbf{A}\right]_1' := \mathbf{\Delta}_{i,j} \left[\mathbf{P}_\mathbf{A}\right]_1$ and $\left[(\mathbf{C}_\mathbf{k}')^\intercal\right]_2 := \left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 + \bar{\mathsf{d}}_{i,j} \left[\mathbf{z}_\sigma^\intercal\right]_2$, and leaves all other proof components unchanged, since the additional components used by the unbounded signature are independent of this user-specific transformation.

The correctness of the proof system $\mathbf{\Pi}_\mathsf{U}$ is straightforward. We provide tight proofs of the zero-knowledge property in Theorem 9 and the special simulation soundness of $\mathbf{\Pi}_\mathsf{U}$ in Theorem 10. Detailed proofs of both theorems can be found in Supplementary Material Section E.

Theorem 9. *The* tLM-NIZK *proof system in Figure 16 is zero-knowledge. Concretely, for all* PPT *adversaries* $\mathcal{A}$*, there exists a* PPT *adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{ZK}}_{\mathbf{\Pi}_\mathsf{U},\mathcal{A}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(\lambda).$$

Alg $\mathsf{Setup}(1^\lambda)$:

$\mathbf{D}_1, \mathbf{D}_2 \hookleftarrow \mathcal{D}_k$
$\mathbf{z} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D}_2)\right)$
$\mathbf{y}_1 \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D}_1)\right)$
$\mathbf{v}_1 \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right)$; $\mathbf{z_1} = \mathbf{D}_1\mathbf{v}_1$
$\left[\mathbf{U}_1\right]_1 = \left[\mathbf{D}_1 \,|\, \mathbf{z}_1\right]_1$
$\mathbf{B}_0, \mathbf{B}_1 \hookleftarrow \mathcal{D}_{2k,k}$
$\mathsf{crs}^\vee \xleftarrow{\$} \mathbf{\Pi}^\vee_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Setup}(1^\lambda)$
$\mathbf{B} \hookleftarrow \mathcal{D}_k$
$\mathbf{K}_0, \mathbf{K}_1 \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{2k\times(k+1)}\right)$
$\mathsf{svk} := (\mathsf{crs}^\vee, \left[\mathbf{B}_0\right]_1, \left[\mathbf{B}\right]_2, \left[\mathbf{K}_0\mathbf{B}\right]_2, \left[\mathbf{K}_1\mathbf{B}\right]_2)$
$\mathsf{ssk} := (\mathbf{K}_0, \mathbf{K}_1)$
$\mathsf{crs}' := (\left[\mathbf{U}_1\right]_1, \left[\mathbf{D}_2\right]_2, \left[\mathbf{z}\right]_2, \mathbf{y}_1)$
return $\mathsf{crs} = (\mathsf{crs}', \mathsf{svk})$

Alg $\mathsf{Ver}(\mathsf{crs}, \mathsf{t}, \mathsf{x}, \pi)$:

$\mathsf{hv} := \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$
$\left[\mathbf{z}_\sigma\right]_2 = \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\mathbf{A}\right]_2$
check $\mathbf{\Pi}^\vee_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Ver}(\mathsf{crs}^\vee, \left[\mathbf{t}\right]_1, \pi_\mathbf{t})$
check $\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}^\intercal_\mathbf{w}\right]_2 - \left[\mathsf{x}\right]_1 \bullet \left[\mathbf{z}^\intercal_\mathbf{A}\right]_2 = \left[\mathbf{P}_\mathbf{A}\right]_1 \bullet \left[\mathbf{D}^\intercal_2\right]_2$
check $(\left[\underline{\mathbf{A}}^\intercal_i - (\underline{\mathbf{A}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}^\intercal_\sigma\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}^\intercal_\mathbf{k}\right]_2 = \left[\mathbf{P}_\mathbf{d}\right]_1 \bullet \left[\mathbf{D}^\intercal_2\right]_2$
check $(\left[\underline{\mathsf{x}}^\intercal - (\underline{\mathsf{x}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}^\intercal_\sigma\right]_2 - \left[\bar{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}^\intercal_\mathbf{k}\right]_2 = \left[\mathbf{P}_\mathsf{x}\right]_1 \bullet \left[\mathbf{D}^\intercal_2\right]_2$
for $j \in [2k]$:
  check $\mathbf{y}_1 \left[\mathbf{t}^\intercal\right]_1 \bullet \left[\mathbf{u}_j\mathbf{z}^\intercal_\sigma\right]_2 - \left[\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[\mathbf{u}_j\mathbf{z}^\intercal_\sigma\right]_2 = \left[\mathbf{P}^j_\mathbf{t}\right]_1 \bullet \left[\mathbf{D}^\intercal_2\right]_2 + \left[\mathbf{U}_1\right]_1 \bullet \left[\mathbf{Q}^j_\mathbf{t}\right]_2$
check $\left[\mathbf{C}_\mathbf{s}\right]_1 \bullet \left[\mathbf{B}\right]_2 - \left[\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2 = \left[\mathbf{U}_1\right]_1 \bullet \left[\mathbf{Q}_\sigma\right]_2$
return 1

Alg $\mathsf{Prove}(\mathsf{crs}, \mathsf{t}, \mathsf{x}, \mathbf{w})$:

$\underline{\mathbf{A}}^\star \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{1\times k}\right)$; $\underline{\mathsf{x}}^\star \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}\right)$
$\mathsf{hv} := \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$
$\mathbf{v}_\sigma \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right)$; $\left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{D_2}\right]_2 \mathbf{v}_\sigma$
$\left[\mathbf{z}_\mathbf{A}\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\sigma\right]_2$
$\mathbf{w_t} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right)$; $\left[\mathbf{t}\right]_1 := \left[\mathbf{B}_0\right]_1 \mathbf{w_t}$
$\pi_\mathbf{t} \xleftarrow{\$} \mathbf{\Pi}^\vee_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Prove}(\mathsf{crs}^\vee, \left[\mathbf{t}\right]_1, \mathbf{w_t})$
$\mathbf{R_w} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k\times k}\right), \mathbf{R_k} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{\ell\times k}\right)$
$\mathbf{R_t} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times 2k}\right)$
$\mathbf{R_s} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times(k+1)}\right)$
$\{\mathbf{\Gamma}_j\}_{j\in[2k]} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times k}\right)$
$\left[\mathbf{C}^\intercal_\mathbf{w}\right]_2 := \mathbf{R_w}\left[\mathbf{D}^\intercal_2\right]_2 + \mathbf{w} \cdot \left[\mathbf{z}^\intercal_\mathbf{A}\right]_2$
$\left[\mathbf{C}^\intercal_\mathbf{k}\right]_2 := \mathbf{R_k}\left[\mathbf{D}^\intercal_2\right]_2$
$\left[\mathbf{C_t}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R_t}$
$\left[\mathbf{C_s}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R_s}$
$\left[\mathbf{P_A}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R_w}$
$\left[\mathbf{P_d}\right]_1 := \left[(\underline{\mathbf{A}}^\intercal_i - (\underline{\mathbf{A}}^\star)^\intercal)\mathbf{v}^\intercal_\sigma - \bar{\mathbf{A}}^\intercal\mathbf{R_k}\right]_1$
$\left[\mathbf{P}_\mathsf{x}\right]_1 := \left[(\underline{\mathsf{x}}^\intercal - (\underline{\mathsf{x}}^\star)^\intercal)\mathbf{v}^\intercal_\sigma - \bar{\mathsf{x}}^\intercal\mathbf{R_k}\right]_1$
for $j \in [2k]$:
  $\left[\mathbf{P}^j_\mathbf{t}\right]_1 := \left[\mathbf{y}_1\mathbf{t}^\intercal\mathbf{u}_j\mathbf{v}^\intercal_\sigma\right]_1 - \left[\mathbf{U}_1\mathbf{R_t}\mathbf{u_j}\mathbf{v}^\intercal_\sigma\right]_1 - \left[\mathbf{U}_1\mathbf{\Gamma}_j\right]_1$
  $\left[\mathbf{Q}^j_\mathbf{t}\right]_2 := \left[\mathbf{\Gamma}_j\mathbf{D}^\intercal_2\right]_2$
$\left[\mathbf{Q}_\sigma\right]_2 := \left[\mathbf{R_s}\mathbf{B} - \mathbf{R_t}(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2$
$\pi' = (\mathbf{z_A}, \mathbf{C_w}, \mathbf{C_k}, \mathbf{C_t}, \mathbf{C_s}, \mathbf{P_A}, \mathbf{P_d}, \mathbf{P}_\mathsf{x}, \{\mathbf{P}^j_\mathbf{t}, \mathbf{Q}^j_\mathbf{t}\}_{j\in[2k]}, \mathbf{Q}_\sigma)$
return $\pi = (\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \pi_\mathbf{t}, \pi')$

Fig. 16. Construction of unbounded special simulation sound tLM-NIZK proof system

Theorem 10. *The* tLM-NIZK *proof system in Figure 16 is special simulation sound. Concretely, for all* PPT *adversaries* $\mathcal{A}$*, there exists a* PPT *adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{SSS}}_{\mathbf{\Pi}_\mathsf{U},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SIG},\mathcal{B}}(\lambda).$$

Alg $\mathsf{SSetup}(1^\lambda)$:

$\mathbf{D}_1, \mathbf{D}_2 \hookleftarrow \mathcal{D}_k$
$\mathbf{z} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D}_2)\right)$
$\mathbf{y}_1 \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D}_1)\right)$
$\mathbf{v}_1 \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \mathbf{z_1} = \mathbf{D}_1\mathbf{v}_1$
$\left[\mathbf{U}_1\right]_1 = \left[\mathbf{D}_1 \,|\, \mathbf{z}_1\right]_1$
$\mathbf{B}_0, \mathbf{B}_1 \hookleftarrow \mathcal{D}_{2k,k}$
$\mathsf{crs}^\vee \xleftarrow{\$} \Pi^\vee_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Setup}(1^\lambda)$
$\mathbf{B} \hookleftarrow \mathcal{D}_k$
$\mathbf{K}_0, \mathbf{K}_1 \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{2k\times(k+1)}\right)$
$\mathsf{svk} := (\mathsf{crs}^\vee, \left[\mathbf{B}_0\right]_1, \left[\mathbf{B}\right]_2, \left[\mathbf{K}_0\mathbf{B}\right]_2, \left[\mathbf{K}_1\mathbf{B}\right]_2)$
$\mathsf{ssk} := (\mathbf{K}_0, \mathbf{K}_1)$
$\mathsf{crs}' := (\left[\mathbf{U}_1\right]_1, \left[\mathbf{D}_2\right]_2, \left[\mathbf{z}\right]_2, \mathbf{y}_1)$
$\mathsf{crs} := (\mathsf{crs}', \mathsf{svk})$
$\mathsf{simK} := \mathsf{ssk}$
return $(\mathsf{crs}, \mathsf{simK})$

Alg $\mathsf{Sim}(\mathsf{crs}, \mathsf{simK}, \mathsf{t}, \mathsf{x})$:

$\bar{\mathbf{k}}^\star \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{\ell}\right)$
$\left[\underline{\mathbf{A}}^\star\right]_1 = \left[\underline{\mathbf{A}}_i\right]_1 - (\bar{\mathbf{k}}^\star)^\intercal \left[\bar{\mathbf{A}}\right]_1$
$\left[\underline{\mathsf{x}}^\star\right]_1 = \left[\underline{\mathsf{x}}\right]_1 - (\bar{\mathbf{k}}^\star)^\intercal \left[\bar{\mathsf{x}}\right]_1$
$\mathsf{hv} := \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$
$\mathbf{v}_\mathbf{A} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \left[\mathbf{z}_\mathbf{A}\right]_2 := \left[\mathbf{D}_2\right]_2 \mathbf{v}_\mathbf{A}$
$\left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\mathbf{A}\right]_2$
$\mathbf{w}_\mathbf{t} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \left[\mathbf{t}\right]_1 := \left[\mathbf{B}_0\right]_1 \mathbf{w}_\mathbf{t}$
$\pi_\mathbf{t} \xleftarrow{\$} \mathbf{\Pi}^\vee_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Prove}(\mathsf{crs}^\vee, \left[\mathbf{t}\right]_1, \mathbf{w}_\mathbf{t})$
$\left[\mathbf{s}\right]_1 := (\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)^\intercal \left[\mathbf{t}\right]_1$
$\mathbf{R}_\mathbf{w} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k\times k}\right), \mathbf{R}_\mathbf{k} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{\ell\times k}\right)$
$\mathbf{R}_\mathbf{t} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times 2k}\right)$
$\mathbf{R}_\mathbf{s} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times(k+1)}\right)$
$\{\mathbf{\Gamma}_j\}_{j\in[2k]} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times k}\right)$
$\left[\mathbf{C}^\intercal_\mathbf{w}\right]_2 := \mathbf{R}_\mathbf{w} \left[\mathbf{D}_2^\intercal\right]_2$
$\left[\mathbf{C}^\intercal_\mathbf{k}\right]_2 := \mathbf{R}_\mathbf{k} \left[\mathbf{D}_2^\intercal\right]_2 + \bar{\mathbf{k}}^\star \left[\mathbf{z}_\sigma^\intercal\right]_2$
$\left[\mathbf{C}_\mathbf{t}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R}_\mathbf{t} + \mathbf{y}_1 \left[\mathbf{t}^\intercal\right]_1$
$\left[\mathbf{C}_\mathbf{s}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R}_\mathbf{s} + \mathbf{y}_1 \left[\mathbf{s}^\intercal\right]_1$
$\left[\mathbf{P}_\mathbf{A}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_\mathbf{w} - \left[\mathsf{x}\right]_1 \mathbf{v}^\intercal_\mathbf{A}$
$\left[\mathbf{P}_\mathbf{d}\right]_1 := \left[-\bar{\mathbf{A}}^\intercal \mathbf{R}_\mathbf{k}\right]_1$
$\left[\mathbf{P}_\mathsf{x}\right]_1 := \left[-\bar{\mathsf{x}}^\intercal \mathbf{R}_\mathbf{k}\right]_1$
for $j \in [2k]$:
  $\left[\mathbf{P}^j_\mathbf{t}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{\Gamma}_j$
  $\left[\mathbf{Q}^j_\mathbf{t}\right]_2 := \mathbf{\Gamma}_j \left[\mathbf{D}_2^\intercal\right]_2 - \mathbf{R}_\mathbf{t}\mathbf{u}_j \left[\mathbf{z}_\sigma^\intercal\right]_2$
$\left[\mathbf{Q}_\sigma\right]_2 := \left[\mathbf{R}_\mathbf{s}\mathbf{B} - \mathbf{R}_\mathbf{t}(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2$
$\pi' := (\mathbf{z}_\mathbf{A}, \mathbf{C}_\mathbf{w}, \mathbf{C}_\mathbf{k}, \mathbf{C}_\mathbf{t}, \mathbf{C}_\mathbf{s}, \mathbf{P}_\mathbf{A}, \mathbf{P}_\mathbf{d}, \mathbf{P}_\mathsf{x}, \{\mathbf{P}^j_\mathbf{t}, \mathbf{Q}^j_\mathbf{t}\}_{j\in[2k]}, \mathbf{Q}_\sigma)$
return $\pi = (\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \pi_\mathbf{t}, \pi')$

Fig. 17. Construction of the simulator $\mathcal{S} = (\mathsf{SSetup}, \mathsf{Sim})$ for the unbounded special simulation sound tLM-NIZK proof system $\mathbf{\Pi}_\mathsf{U}$.

E Proof of tLM-NIZK

E.1 Proof of Theorem 6

Theorem 6. *The proof system* $\mathbf{\Pi}_\mathsf{OT}$ *in Figure 11 is zero-knowledge. Concretely, for all* PPT *adversaries* $\mathcal{A}$*, there exists a* PPT *adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{ZK}}_{\mathbf{\Pi}_\mathsf{OT},\mathcal{A}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(\lambda).$$

Proof. We present the construction of the simulator $\mathcal{S} = (\mathsf{SSetup}, \mathsf{Sim})$ in Figure 13. We first observe that the crs generated by SSetup is identical to the one generated by Setup in the real scheme.

We prove the zero-knowledge property of $\mathbf{\Pi}_{\mathsf{OT}}$ by introducing hybrid games $\mathbf{G}_0, \ldots, \mathbf{G}_4$, where in $\mathbf{G}_0$ the adversary interacts with the real proof system, and in $\mathbf{G}_4$ the adversary interacts with the simulated proof system. We show that the adversary's views in $\mathbf{G}_0$ and $\mathbf{G}_4$ are computationally indistinguishable. The pseudocode of the hybrid games is given in Figure 18.

Game $\mathbf{G}_0$: This is the original security game where $\mathcal{A}$ interacts with the Setup and Prove oracles.

$$\mathsf{pr}_0 = \Pr\left[\mathcal{A}^{\mathcal{O}_{\mathsf{Prove}}}(1^\lambda, \mathsf{crs}) = 1 \;\middle|\; \mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)\right].$$

Game $\mathbf{G}_1$: In this game, instead of sampling $\mathbf{z}$ uniformly at random from $\mathbb{Z}_{\mathsf{p}}^{k+1} \setminus \mathsf{Span}(\mathbf{D})$, we first sample $\mathbf{v}_0$ and set $\mathbf{z} = \mathbf{D}\mathbf{v}_0$. The only difference between $\mathbf{G}_1$ and $\mathbf{G}_0$ is the distribution of $\mathbf{z}$, which reduces to the $\mathcal{D}_k$-MDDH problem.

$$|\mathsf{pr}_1 - \mathsf{pr}_0| \leq \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k, \mathbb{G}_2, \mathcal{B}}(\lambda).$$

Game $\mathbf{G}_2$: In game $\mathbf{G}_2$, we change the way of computing $\left[\mathbf{C}_{\mathbf{w}}\right]_2$ and $\left[\mathbf{P}_{\mathbf{A}}\right]_1$. In the previous game, we have

$$\left[\mathbf{C}_{\mathbf{w}}^\intercal\right]_2 = \mathbf{R}_{\mathbf{w}} \left[\mathbf{D}^\intercal\right]_2 + \mathbf{w} \left[\mathbf{z}_{\mathbf{A}}^\intercal\right]_2, \qquad \left[\mathbf{P}_{\mathbf{A}}\right]_1 = \left[\mathbf{A}_i\right]_1 \mathbf{R}_{\mathbf{w}}.$$

We compute them in $\mathbf{G}_2$ as follows:

$$\left[\mathbf{C}_{\mathbf{w}}^\intercal\right]_2 = \mathbf{R}_{\mathbf{w}} \left[\mathbf{D}^\intercal\right]_2, \qquad \left[\mathbf{P}_{\mathbf{A}}\right]_1 = \left[\mathbf{A}_i\right]_1 \mathbf{R}_{\mathbf{w}} - \left[\mathsf{x}\right]_1 \mathbf{v}_{\mathbf{A}}^\intercal,$$

where $\mathbf{v}_{\mathbf{A}} = \mathbf{v}_0 - \mathbf{v}_\sigma$. Since $\mathbf{z}_{\mathbf{A}} = \mathbf{D}\mathbf{v}_{\mathbf{A}}$, the change in $\mathbf{G}_2$ can be seen as changing $\mathbf{R}_{\mathbf{w}}$ to $\mathbf{R}_{\mathbf{w}} - \mathbf{w}\mathbf{v}_{\mathbf{A}}^\intercal$. Therefore, we have

$$|\mathsf{pr}_2 - \mathsf{pr}_1| = 0.$$

Game $\mathbf{G}_3$: We change the way of computing $\left[\mathbf{C}_{\mathbf{k}}\right]_2, \left[\mathbf{C}_\sigma\right]_2, \left[\mathbf{P}_{\mathbf{d}}\right]_1, \left[\mathbf{P}_{\mathsf{x}}\right]_1, \left[\mathbf{P}_\sigma\right]_1$ in $\mathbf{G}_3$. We first compute two row vectors $\left[\underline{\mathbf{A}}^\star\right]_1$ and $\left[\underline{\mathsf{x}}^\star\right]_1$ and their hash value hv, then compute the one-time signature $\sigma = \bar{\mathbf{K}}^\intercal \binom{1}{\mathsf{hv}}$. We recall that in $\mathbf{G}_3$, we have

$$\begin{aligned}
&\left[\mathbf{C}_{\mathbf{k}}^\intercal\right]_2 = \mathbf{R}_{\mathbf{k}} \left[\mathbf{D}^\intercal\right]_2, && \left[\mathbf{C}_\sigma^\intercal\right]_2 = \mathbf{R}_\sigma \left[\mathbf{D}^\intercal\right]_2,\\
&\left[\mathbf{P}_{\mathbf{d}}\right]_1 = \left(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1\right)\mathbf{v}_\sigma^\intercal - \left[\bar{\mathbf{A}}^\intercal\right]_1 \mathbf{R}_{\mathbf{k}},\\
&\left[\mathbf{P}_{\mathsf{x}}\right]_1 = \left(\left[\underline{\mathsf{x}}_i^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1\right)\mathbf{v}_\sigma^\intercal - \left[\bar{\mathsf{x}}^\intercal\right]_1 \mathbf{R}_{\mathbf{k}},\\
&\left[\mathbf{P}_\sigma\right]_1 = \left[\mathbf{B}^\intercal\right]_1 \mathbf{R}_\sigma - \left[\mathbf{C}^\intercal\right]_1 \begin{pmatrix} 1 \\ \mathsf{hv} \end{pmatrix} \mathbf{v}_\sigma^\intercal.
\end{aligned}$$

Alg $\mathsf{Setup}(1^\lambda)$:

$\mathbf{D} \leftarrow\!\!\$\ \mathcal{D}_k$

$\mathbf{z} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D})\right)$ // $\mathbf{G}_0$-$\mathbf{G}_1$

$\mathbf{v}_0 \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \mathbf{z} := \mathbf{D}\mathbf{v}_0$ // $\mathbf{G}_1$-$\mathbf{G}_3$

$\mathbf{z} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D})\right)$ // $\mathbf{G}_4$

$\mathbf{B} \leftarrow\!\!\$\ \mathcal{D}_k;\ \mathbf{K} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{2\times(k+1)}\right);\ \mathbf{C} = \mathbf{K}\mathbf{B}$

$\mathsf{crs} := (\left[\mathbf{D}\right]_2, \left[\mathbf{z}\right]_2, \left[\mathbf{B}^\intercal\right]_1, \left[\mathbf{C}^\intercal\right]_1)$

$\mathsf{simK} := (\mathbf{K}, \mathbf{v}_0)$ // $\mathbf{G}_1$-$\mathbf{G}_3$

$\mathsf{simK} := (\mathbf{K})$ // $\mathbf{G}_4$

return crs

Alg $\mathsf{Prove}(\mathsf{crs}, \mathsf{t}, \mathsf{x}, \mathbf{w})$:

$\left[\underline{\mathbf{A}}^\star\right]_1 \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{G}_1^{1\times k}\right);\ \left[\underline{\mathsf{x}}^\star\right]_1 \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{G}_1\right)$ // $\mathbf{G}_0$-$\mathbf{G}_2$

$\bar{\mathbf{k}}^\star \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k+1}\right);\ \left[\underline{\mathbf{A}}^\star\right]_1 = \left[\underline{\mathbf{A}}_i\right]_1 - (\bar{\mathbf{k}}^\star)^\intercal \left[\bar{\mathbf{A}}\right]_1$ // $\mathbf{G}_3$-$\mathbf{G}_4$

$\left[\underline{\mathsf{x}}^\star\right]_1 = \left[\underline{\mathsf{x}}_i\right]_1 - (\bar{\mathbf{k}}^\star)^\intercal \left[\bar{\mathsf{x}}\right]_1$ // $\mathbf{G}_3$-$\mathbf{G}_4$

$\mathsf{hv} := \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$

$\sigma := \mathbf{K}^\intercal \binom{1}{\mathsf{hv}}$ // $\mathbf{G}_3$-$\mathbf{G}_4$

$\mathbf{v}_\sigma \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{D}\right]_2 \mathbf{v}_\sigma;\ \left[\mathbf{z}_\mathbf{A}\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\sigma\right]_2$ // $\mathbf{G}_0$

$\mathbf{v}_\mathbf{A} := \mathbf{v}_0 - \mathbf{v}_\sigma$ // $\mathbf{G}_1$-$\mathbf{G}_3$

$\mathbf{v}_\mathbf{A} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right)$ // $\mathbf{G}_4$

$\left[\mathbf{z}_\mathbf{A}\right]_2 := \left[\mathbf{D}\right]_2 \mathbf{v}_\mathbf{A};\ \left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\mathbf{A}\right]_2$ // $\mathbf{G}_1$-$\mathbf{G}_4$

$\mathbf{R}_\mathbf{w} \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k\times k}\right);\ \mathbf{R}_\mathbf{k}, \mathbf{R}_\sigma \leftarrow\!\!\$\ \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times k}\right)$

$\left[\mathbf{C}_\mathbf{w}^\intercal\right]_2 := \mathbf{R}_\mathbf{w} \left[\mathbf{D}^\intercal\right]_2 + \mathbf{w} \left[\mathbf{z}_\mathbf{A}^\intercal\right]_2$ // $\mathbf{G}_0$-$\mathbf{G}_1$

$\left[\mathbf{C}_\mathbf{w}^\intercal\right]_2 := \mathbf{R}_\mathbf{w} \left[\mathbf{D}^\intercal\right]_2$ // $\mathbf{G}_2$-$\mathbf{G}_4$

$\left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 := \mathbf{R}_\mathbf{k} \left[\mathbf{D}^\intercal\right]_2;\ \left[\mathbf{C}_\sigma^\intercal\right]_2 := \mathbf{R}_\sigma \left[\mathbf{D}^\intercal\right]_2$ // $\mathbf{G}_0$-$\mathbf{G}_2$

$\left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 := \mathbf{R}_\mathbf{k} \left[\mathbf{D}^\intercal\right]_2 + \bar{\mathbf{k}}^\star \left[\mathbf{z}_\sigma^\intercal\right]_2;\ \left[\mathbf{C}_\sigma^\intercal\right]_2 := \mathbf{R}_\sigma \left[\mathbf{D}^\intercal\right]_2 + \sigma \left[\mathbf{z}_\sigma^\intercal\right]_2$ // $\mathbf{G}_3$-$\mathbf{G}_4$

$\left[\mathbf{P}_\mathbf{A}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_\mathbf{w}$ // $\mathbf{G}_0$-$\mathbf{G}_1$

$\left[\mathbf{P}_\mathbf{A}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_\mathbf{w} - \left[\mathsf{x}\right]_1 \mathbf{v}_\mathbf{A}^\intercal$ // $\mathbf{G}_2$-$\mathbf{G}_4$

$\left[\mathbf{P}_\mathbf{d}\right]_1 := (\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1)\mathbf{v}_\sigma^\intercal - \left[\bar{\mathbf{A}}^\intercal\right]_1 \mathbf{R}_\mathbf{k}$ // $\mathbf{G}_0$-$\mathbf{G}_2$

$\left[\mathbf{P}_\mathbf{d}\right]_1 := -\left[\bar{\mathbf{A}}^\intercal\right]_1 \mathbf{R}_\mathbf{k}$ // $\mathbf{G}_3$-$\mathbf{G}_4$

$\left[\mathbf{P}_\mathsf{x}\right]_1 := (\left[\underline{\mathsf{x}}_i^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1)\mathbf{v}_\sigma^\intercal - \left[\bar{\mathsf{x}}^\intercal\right]_1 \mathbf{R}_\mathbf{k}$ // $\mathbf{G}_0$-$\mathbf{G}_2$

$\left[\mathbf{P}_\mathsf{x}\right]_1 := -\left[\bar{\mathsf{x}}^\intercal\right]_1 \mathbf{R}_\mathbf{k}$ // $\mathbf{G}_3$-$\mathbf{G}_4$

$\left[\mathbf{P}_\sigma\right]_1 := \left[\mathbf{B}^\intercal\right]_1 \mathbf{R}_\sigma - \left[\mathbf{C}^\intercal\right]_1 \binom{1}{\mathsf{hv}} \mathbf{v}_\sigma^\intercal$ // $\mathbf{G}_0$-$\mathbf{G}_2$

$\left[\mathbf{P}_\sigma\right]_1 := \left[\mathbf{B}^\intercal\right]_1 \mathbf{R}_\sigma$ // $\mathbf{G}_3$-$\mathbf{G}_4$

$\pi := (\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \left[\mathbf{z}_\mathbf{A}\right]_2, \left[\mathbf{C}_\mathbf{w}\right]_2, \left[\mathbf{C}_\mathbf{k}\right]_2, \left[\mathbf{C}_\sigma\right]_2, \left[\mathbf{P}_\mathbf{A}\right]_1, \left[\mathbf{P}_\mathbf{d}\right]_1, \left[\mathbf{P}_\mathsf{x}\right]_1, \left[\mathbf{P}_\sigma\right]_1)$

return π

Fig. 18. Hybrid games in proving the zero-knowledge property of one-time special simulation sound tLM-NIZK proof system.

We compute them in $\mathbf{G}_3$ as follows:

$$\left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 = \mathbf{R}_\mathbf{k} \left[\mathbf{D}^\intercal\right]_2 + \bar{\mathbf{k}}^\star \left[\mathbf{z}_\sigma^\intercal\right]_2, \qquad \left[\mathbf{C}_\sigma^\intercal\right]_2 = \mathbf{R}_\sigma \left[\mathbf{D}^\intercal\right]_2 + \sigma \left[\mathbf{z}_\sigma^\intercal\right]_2,$$
$$\left[\mathbf{P}_\mathbf{d}\right]_1 = -\left[\bar{\mathbf{A}}^\intercal\right]_1 \mathbf{R}_\mathbf{k}, \qquad \left[\mathbf{P}_\mathsf{x}\right]_1 = -\left[\bar{\mathsf{x}}^\intercal\right]_1 \mathbf{R}_\mathbf{k},$$
$$\left[\mathbf{P}_\sigma\right]_1 = \left[\mathbf{B}^\intercal\right]_1 \mathbf{R}_\sigma.$$

Since $\mathbf{z}_\sigma = \mathbf{D}\mathbf{v}_\sigma$, the above change is oblivious to the adversary.

Therefore, we have

$$|\mathsf{pr}_3 - \mathsf{pr}_2| = 0.$$

Game $\mathbf{G}_4$: We change again the way of computing $\mathbf{z}_\mathbf{A}, \mathbf{z}_\sigma$. We observe that in $\mathbf{G}_3$, we do not use the value of $\mathbf{v}_\sigma$ to compute $\left[\mathbf{C}_\mathbf{k}\right]_2, \left[\mathbf{C}_\sigma\right]_2, \left[\mathbf{P}_\mathbf{d}\right]_1, \left[\mathbf{P}_\mathsf{x}\right]_1, \left[\mathbf{P}_\sigma\right]_1$. Therefore, we can sample $\mathbf{v}_\mathbf{A}$ uniformly, set $\mathbf{z}_\mathbf{A} = \mathbf{D}\mathbf{v}_\mathbf{A}$ and $\mathbf{z}_\sigma = \mathbf{z} - \mathbf{z}_\mathbf{A}$, and sample $\mathbf{z}$ uniformly from $\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D})$. The only change in $\mathbf{G}_4$ is the distribution of $\mathbf{z}$; therefore, we have

$$|\mathsf{pr}_4 - \mathsf{pr}_3| \leq \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(\lambda).$$

Finally, we observe that the Setup and Prove algorithms in $\mathbf{G}_4$ are identical to the SSetup and Sim algorithms. Therefore, we have

$$\mathsf{pr}_4 = \Pr\left[\mathcal{A}^{\mathcal{O}_{\mathsf{Sim}}}(1^\lambda, \mathsf{crs}) = 1 \;\middle|\; (\mathsf{crs}, \mathsf{simK}) \xleftarrow{\$} \mathsf{SSetup}(1^\lambda)\right].$$

Using the triangle inequality over all above hybrids, we have

$$|\mathsf{pr}_0 - \mathsf{pr}_4| \leq 2\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(\lambda).$$

Therefore, we conclude the proof. □

E.2 Proof of Theorem 7

Theorem 7. *The one-time* tLM-NIZK *proof system presented in Figure 11 satisfies special simulation soundness.*

Proof. We begin by observing that the vector $\mathbf{z}$ within the common reference string (CRS), generated by the SSetup algorithm, is designed to lie outside the linear span of $\mathbf{D}$. Consequently, either $\mathbf{z}_\mathbf{A}$ or $\mathbf{z}_\sigma$ (or both) must also lie outside the linear span of $\mathbf{D}$. This implies the existence of a vector ξ, which the challenger can derive without affecting the adversary's view, such that $\xi^\intercal \mathbf{D} = 0$, and either $\xi^\intercal \mathbf{z}_\mathbf{A} = 1$ or $\xi^\intercal \mathbf{z}_\sigma = 1$. We consider these two scenarios separately:

- **Case $\mathbf{z}_\mathbf{A} \notin \mathsf{Span}(\mathbf{D})$:** The first verification equation is

$$\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}_\mathbf{w}^\intercal\right]_2 - \left[\mathsf{x}\right]_1 \bullet \left[\mathbf{z}_\mathbf{A}^\intercal\right]_2 = \left[\mathbf{P}_\mathbf{A}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2.$$

  Right-multiplying both sides by ξ yields

$$\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}_\mathbf{w}^\intercal \xi\right]_2 - \left[\mathsf{x}\right]_1 \bullet \left[1\right]_2 = \left[\mathbf{P}_\mathbf{A}\right]_1 \bullet \left[0\right]_2.$$

  Therefore, in this case, we conclude that $\mathsf{x} \in \mathsf{Span}(\mathbf{A}_i)$.

– **Case $\mathbf{z}_\sigma \notin \mathsf{Span}(\mathbf{D})$:** In this case, we examine the other three verification equations:

$$\begin{aligned}
\left(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1\right) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 &= \left[\mathbf{P}_\mathbf{d}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2, \\
\left(\left[\underline{\mathsf{x}}_i^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1\right) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 &= \left[\mathbf{P}_\mathsf{x}\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2, \\
\left[\mathbf{B}^\intercal\right]_1 \bullet \left[\mathbf{C}_\sigma^\intercal\right]_2 - \left[\mathbf{C}^\intercal\right]_1 \bullet \left[\begin{pmatrix} 1 \\ \mathsf{hv} \end{pmatrix} \mathbf{z}_\sigma^\intercal\right]_2 &= \left[\mathbf{P}_\sigma\right]_1 \bullet \left[\mathbf{D}^\intercal\right]_2.
\end{aligned}$$

Right-multiplying both equations by ξ, we obtain

$$\begin{aligned}
\left(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1\right) \bullet \left[1\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal \xi\right]_2 &= \left[\mathbf{P}_\mathbf{d}\right]_1 \bullet \left[0\right]_2, \\
\left(\left[\underline{\mathsf{x}}_i^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1\right) \bullet \left[1\right]_2 - \left[\bar{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal \xi\right]_2 &= \left[\mathbf{P}_\mathsf{x}\right]_1 \bullet \left[0\right]_2, \\
\left[\mathbf{B}^\intercal\right]_1 \bullet \left[\mathbf{C}_\sigma^\intercal \xi\right]_2 - \left[\mathbf{C}^\intercal\right]_1 \bullet \left[\begin{pmatrix} 1 \\ \mathsf{hv} \end{pmatrix}\right]_2 &= \left[\mathbf{P}_\sigma\right]_1 \bullet \left[0\right]_2.
\end{aligned}$$

From these equations, we can extract $\left[\mathbf{C}_\mathbf{k}^\intercal \xi\right]_2$ such that $\left(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1\right) \bullet \left[1\right]_2 = \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal \xi\right]_2$ and $\left(\left[\underline{\mathsf{x}}_i^\intercal\right]_1 - \left[(\underline{\mathsf{x}}^\star)^\intercal\right]_1\right) \bullet \left[1\right]_2 = \left[\bar{\mathsf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal \xi\right]_2$. Furthermore, $\left[\mathbf{C}_\sigma^\intercal \xi\right]_2$ constitutes a valid one-time signature for hv. We distinguish between two subcases:

1. If $\mathsf{hv} = \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$ where $\left[\underline{\mathbf{A}}^\star\right]_1$ and $\left[\underline{\mathsf{x}}^\star\right]_1$ differ from those generated in the challenge ciphertext, then the adversary has successfully broken the one-time unforgeability of the underlying signature scheme.
2. If $\left[\underline{\mathbf{A}}^\star\right]_1$ and $\left[\underline{\mathsf{x}}^\star\right]_1$ equal those generated in the challenge ciphertext, the challenger could have generated $\left[\underline{\mathbf{A}}^\star\right]_1$ and $\left[\underline{\mathsf{x}}^\star\right]_1$ as $\left[\underline{\mathbf{A}}^\star\right]_1 = (\mathbf{k}')^\intercal \left[\bar{\mathbf{A}}\right]_1$ and $\left[\underline{\mathsf{x}}^\star\right]_1 = (\mathbf{k}'')^\intercal \left[\bar{\mathsf{x}}\right]_1$. Consequently, $\left[\mathbf{C}_\mathbf{k}^\intercal \xi\right]_2 - \left[\mathbf{k}'\right]_2$ equals the $\mathbb{G}_2$ representation of the re-encryption token from the challenge user to user j.

Therefore, we conclude that the $\mathsf{tLM\text{-}NIZK}$ proof system in Figure 11 satisfies one-time special simulation soundness. □

E.3 Proof of Theorem 9

Theorem 9. *The* $\mathsf{tLM\text{-}NIZK}$ *proof system in Figure 16 is zero-knowledge. Concretely, for all* PPT *adversaries* $\mathcal{A}$*, there exists a* PPT *adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{ZK}}_{\mathbf{\Pi}_\mathsf{U},\mathcal{A}}(\lambda) \leq 2\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(\lambda).$$

Proof. We establish the zero-knowledge property of $\mathbf{\Pi}_\mathsf{U}$ through a sequence of hybrid games. As the verification algorithm remains consistent across these games, we present only the modifications to the Setup and Prove algorithms for conciseness. The pseudocode for these hybrid games is provided in Figure 19 and Figure 20.

Game $\mathbf{G}_0$: In this initial game, the adversary interacts directly with the original

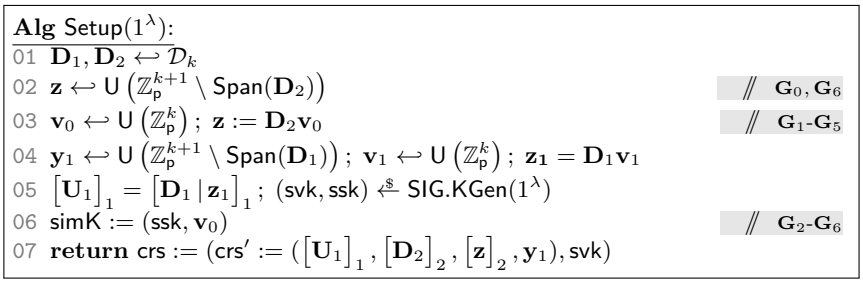

Fig. 19. Hybrid security games of unbounded special simulation sound tLM-NIZK proof system $\mathbf{\Pi}_\mathsf{U}$ in Setup.

Setup and Prove algorithms. The winning probability of the adversary in $\mathbf{G}_0$ is therefore:

$$\mathsf{pr}_0 = \Pr\left[\mathcal{A}^{\mathcal{O}_{\mathsf{Prove}}}(1^\lambda, \mathsf{crs}) = 1 \;\middle|\; \mathsf{crs} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)\right].$$

Game $\mathbf{G}_1$: In $\mathbf{G}_1$, we change the distribution of $\mathbf{z}$. Instead of being uniformly sampled from $\mathbb{Z}_\mathsf{p}^{k+1} \setminus \mathsf{Span}(\mathbf{D}_2)$, we sample $\mathbf{v}_0$ uniformly from $\mathbb{Z}_\mathsf{p}^k$ and set $\mathbf{z} = \mathbf{D}_2\mathbf{v}_0$. Under the $\mathcal{D}_k$-MDDH assumption, $\mathbf{G}_0$ and $\mathbf{G}_1$ are indistinguishable to the adversary, and we have:

$$|\mathsf{pr}_1 - \mathsf{pr}_0| \leq \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k, \mathbb{G}_2, \mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_2$: In this game, we introduce a simulation key simK that corresponds to the signing key of the underlying signature scheme and $\mathbf{v}_0$. Since this simulation key is generated internally and remains inaccessible to the adversary, the adversary's view remains unchanged. Consequently, we have:

$$|\mathsf{pr}_2 - \mathsf{pr}_1| = 0.$$

Game $\mathbf{G}_3$: In $\mathbf{G}_3$, we first compute $\mathbf{v}_\mathbf{A} := \mathbf{v}_0 - \mathbf{v}_\sigma$. We observe that $\mathbf{z}_\mathbf{A} = \mathbf{D}_2\mathbf{v}_\mathbf{A} \in \mathsf{Span}(\mathbf{D}_2)$. Then we generate $\left[\mathbf{C}_\mathbf{w}^\intercal\right]_2$ as $\mathbf{R}_\mathbf{w}\left[\mathbf{D}_2^\intercal\right]_2$ without using $\mathbf{w}$, and compute $\left[\mathbf{P}_\mathbf{A}\right]_1$ using the value of $\mathbf{v}_\mathbf{A}$ as $\left[\mathbf{P}_\mathbf{A}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_\mathbf{w} - \left[\mathsf{x}\right]_1 \mathbf{v}_\mathbf{A}^\intercal$. When $\mathbf{z}_\mathbf{A} \in \mathsf{Span}(\mathbf{D}_2)$, $\left[\mathbf{C}_\mathbf{w}^\intercal\right]_2 = \mathbf{R}_\mathbf{w}\left[\mathbf{D}_2^\intercal\right]_2 + \mathbf{w}\left[\mathbf{z}_\mathbf{A}^\intercal\right]_2$ is a perfectly hiding commitment of $\mathbf{w}$. Therefore, the adversary's views in $\mathbf{G}_2$ and $\mathbf{G}_3$ are identical. Thus, we have:

$$|\mathsf{pr}_3 - \mathsf{pr}_2| = 0.$$

Game $\mathbf{G}_4$: In this game, we generate the proof of the remaining four equations honestly using $\mathbf{z}_\sigma$ and $\mathbf{w}_\mathbf{t}$, without using the value of $\mathbf{v}_\sigma$. Similar to the argument

Alg $\mathsf{Prove}(\mathsf{crs}, \mathsf{t}, \mathsf{x}, \mathbf{w})$:

01 $\underline{\mathbf{A}}^\star \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{1\times k}\right)$ $\underline{\mathsf{x}}^\star \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}\right)$ // $\mathbf{G}_0$-$\mathbf{G}_3$
02 $\bar{\mathbf{k}}^\star \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{\ell}\right)$ // $\mathbf{G}_4$-$\mathbf{G}_6$
03 $\underline{\mathbf{A}}^\star := \underline{\mathbf{A}}_i - (\bar{\mathbf{k}}^\star)^\intercal \bar{\mathbf{A}}\ \underline{\mathsf{x}}^\star := \underline{\mathsf{x}}_i - (\bar{\mathbf{k}}^\star)^\intercal \bar{\mathsf{x}}$ // $\mathbf{G}_4$-$\mathbf{G}_6$
04 $\mathsf{hv} := \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \mathsf{t})$
05 $\mathbf{v}_\sigma \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{D}_2\right]_2 \mathbf{v}_\sigma$ // $\mathbf{G}_0$-$\mathbf{G}_4$
06 $\mathbf{v}_\mathbf{A} := \mathbf{v}_0 - \mathbf{v}_\sigma$ // $\mathbf{G}_3$-$\mathbf{G}_4$
07 $\mathbf{v}_\mathbf{A} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \left[\mathbf{z}_\mathbf{A}\right]_2 := \left[\mathbf{D}_2\right]_2 \mathbf{v}_\mathbf{A}$ // $\mathbf{G}_5$-$\mathbf{G}_6$
08 $\left[\mathbf{z}_\mathbf{A}\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\sigma\right]_2$ // $\mathbf{G}_0$-$\mathbf{G}_4$
09 $\left[\mathbf{z}_\sigma\right]_2 := \left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\mathbf{A}\right]_2$ // $\mathbf{G}_5$-$\mathbf{G}_6$
10 $\mathbf{w}_\mathbf{t} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k}\right);\ \left[\mathbf{t}\right]_1 := \left[\mathbf{B}_0\right]_1 \mathbf{w}_\mathbf{t};\ \pi_\mathbf{t} \xleftarrow{\$} \boldsymbol{\Pi}^\vee_{\mathbf{B}_0,\mathbf{B}_1}.\mathsf{Prove}(\mathsf{crs}^\vee, \left[\mathbf{t}\right]_1, \mathbf{w}_\mathbf{t})$
11 $\mathbf{R}_\mathbf{w} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{k\times k}\right), \mathbf{R}_\mathbf{k} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{\ell\times k}\right), \mathbf{R}_\mathbf{t} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times 2k}\right)$
12 $\mathbf{R}_\mathbf{s} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times(k+1)}\right), \{\boldsymbol{\Gamma}_i\}_{i\in[2k]} \hookleftarrow \mathsf{U}\left(\mathbb{Z}_\mathsf{p}^{(k+1)\times k}\right)$
13 $\left[\mathbf{C}^\intercal_\mathbf{w}\right]_2 := \mathbf{R}_\mathbf{w}\left[\mathbf{D}_2^\intercal\right]_2 + \mathbf{w}\cdot\left[\mathbf{z}_\mathbf{A}^\intercal\right]_2$ // $\mathbf{G}_0$-$\mathbf{G}_2$
14 $\left[\mathbf{C}^\intercal_\mathbf{w}\right]_2 := \mathbf{R}_\mathbf{w}\left[\mathbf{D}_2^\intercal\right]_2$ // $\mathbf{G}_3$-$\mathbf{G}_6$
15 $\left[\mathbf{C}^\intercal_\mathbf{k}\right]_2 := \mathbf{R}_\mathbf{k}\left[\mathbf{D}_2^\intercal\right]_2;\ \left[\mathbf{C}_\mathbf{t}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R}_\mathbf{t};\ \left[\mathbf{C}_\mathbf{s}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R}_\mathbf{s}$ // $\mathbf{G}_0$-$\mathbf{G}_3$
16 $\left[\mathbf{s}\right]_1 := (\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)^\intercal \left[\mathbf{t}\right]_1;\ \left[\mathbf{C}^\intercal_\mathbf{k}\right]_2 := \mathbf{R}_\mathbf{k}\left[\mathbf{D}_2^\intercal\right]_2 + \bar{\mathbf{k}}^\star\left[\mathbf{z}_\sigma^\intercal\right]_2$ // $\mathbf{G}_4$-$\mathbf{G}_6$
17 $\left[\mathbf{C}_\mathbf{t}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R}_\mathbf{t} + \mathbf{y}_1\left[\mathbf{t}^\intercal\right]_1;\ \left[\mathbf{C}_\mathbf{s}\right]_1 := \left[\mathbf{U}_1\right]_1 \mathbf{R}_\mathbf{s} + \mathbf{y}_1\left[\mathbf{s}^\intercal\right]_1$ // $\mathbf{G}_4$-$\mathbf{G}_6$
18 $\left[\mathbf{P}_\mathbf{A}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_\mathbf{w}$ // $\mathbf{G}_0$-$\mathbf{G}_2$
19 $\left[\mathbf{P}_\mathbf{A}\right]_1 := \left[\mathbf{A}_i\right]_1 \mathbf{R}_\mathbf{w} - \left[\mathsf{x}\right]_1 \mathbf{v}_\mathbf{A}^\intercal$ // $\mathbf{G}_3$-$\mathbf{G}_6$
20 $\left[\mathbf{P}_\mathbf{d}\right]_1 := \left[(\mathbf{A}_i - \underline{\mathbf{A}}^\star)^\intercal\right]_1 \mathbf{v}_\sigma^\intercal - \left[\bar{\mathbf{A}}^\intercal\right]_1 \mathbf{R}_\mathbf{k}$ // $\mathbf{G}_0$-$\mathbf{G}_3$
21 $\left[\mathbf{P}_\mathsf{x}\right]_1 := \left[(\underline{\mathsf{x}} - \underline{\mathsf{x}}^\star)^\intercal\right]_1 \mathbf{v}_\sigma^\intercal - \left[\bar{\mathsf{x}}^\intercal\right]_1 \mathbf{R}_\mathbf{k}$ // $\mathbf{G}_0$-$\mathbf{G}_3$
22 $\left[\mathbf{P}_\mathbf{d}\right]_1 := \left[-\bar{\mathbf{A}}^\intercal\mathbf{R}_\mathbf{k}\right]_1\ \left[\mathbf{P}_\mathsf{x}\right]_1 := \left[-\bar{\mathsf{x}}^\intercal\mathbf{R}_\mathbf{k}\right]_1$ // $\mathbf{G}_4$-$\mathbf{G}_6$
23 **for** $j \in [2k]$:
24 $\quad\left[\mathbf{P}^j_\mathbf{t}\right]_1 := \left[\mathbf{y}_1\mathbf{t}^\intercal\mathbf{u}_j\mathbf{v}_\sigma^\intercal\right]_1 - \left[\mathbf{U}_1\mathbf{R}_\mathbf{t}\mathbf{u}_j\mathbf{v}_\sigma^\intercal\right]_1 - \left[\mathbf{U}_1\boldsymbol{\Gamma}_j\right]_1$ // $\mathbf{G}_0$-$\mathbf{G}_3$
25 $\quad\left[\mathbf{Q}^j_\mathbf{t}\right]_2 := \left[\boldsymbol{\Gamma}_j\mathbf{D}_2^\intercal\right]_2$ // $\mathbf{G}_0$-$\mathbf{G}_3$
26 $\quad\left[\mathbf{P}^j_\mathbf{t}\right]_1 := \left[\mathbf{U}_1\boldsymbol{\Gamma}_j\right]_1;\ \left[\mathbf{Q}^j_\mathbf{t}\right]_2 := \left[\boldsymbol{\Gamma}_j\mathbf{D}_2^\intercal\right]_2 - \mathbf{R}_\mathbf{t}\mathbf{u}_j\left[\mathbf{z}_\sigma^\intercal\right]_2$ // $\mathbf{G}_4$-$\mathbf{G}_6$
27 $\left[\mathbf{Q}_\sigma\right]_2 := \left[\mathbf{R}_\mathbf{s}\mathbf{B} - \mathbf{R}_\mathbf{t}(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2$
28 $\pi' = (\mathbf{z}_\mathbf{A}, \mathbf{C}_\mathbf{w}, \mathbf{C}_\mathbf{k}, \mathbf{C}_\mathbf{t}, \mathbf{C}_\mathbf{s}, \mathbf{P}_\mathbf{A}, \mathbf{P}_\mathbf{d}, \mathbf{P}_\mathsf{x}, \{\mathbf{P}^j_\mathbf{t}, \mathbf{Q}^j_\mathbf{t}\}_{j\in[2k]}, \mathbf{Q}_\sigma)$
29 **return** $\pi = (\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathsf{x}}^\star\right]_1, \pi_\mathbf{t}, \pi')$

Fig. 20. Hybrid security games of unbounded special simulation sound tLM-NIZK proof system $\boldsymbol{\Pi}_\mathsf{U}$ in Prove.

in the previous game, the adversary's views in $\mathbf{G}_3$ and $\mathbf{G}_4$ are identical. Thus, we have:

$$|\mathsf{pr}_4 - \mathsf{pr}_3| = 0.$$

Game $\mathbf{G}_5$: In this game, we change the way of generating $\mathbf{z}_\mathbf{A}, \mathbf{z}_\sigma$. Instead of computing $\mathbf{z}_\mathbf{A}$ using $\mathbf{v}_\mathbf{A}$, we compute $\left[\mathbf{z}_\mathbf{A}\right]_2$ as $\left[\mathbf{z}\right]_2 - \left[\mathbf{z}_\sigma\right]_2$. Note that the

adversary's view is unchanged compared to $\mathbf{G}_4$. Thus, we have:

$$|\mathsf{pr}_5 - \mathsf{pr}_4| = 0.$$

Game $\mathbf{G}_6$: In this game, we change the distribution of $\mathbf{z}$ in the Setup algorithm. If an adversary distinguishes $\mathbf{G}_5$ and $\mathbf{G}_6$, we can construct an adversary $\mathcal{B}$ that breaks the $\mathcal{D}_k$-MDDH assumption. We have:

$$|\mathsf{pr}_6 - \mathsf{pr}_5| \leq \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(1^\lambda).$$

In $\mathbf{G}_6$, the adversary's view is identical to the one in interacting with the simulator $\mathcal{S}$. The simulator generates the same crs and simK as in $\mathbf{G}_6$, and the proof π is generated using the same method as in $\mathbf{G}_6$. Therefore, we have:

$$\mathsf{pr}_6 = \Pr\left[\mathcal{A}^{\mathcal{O}_{\mathsf{Sim}}}(1^\lambda, \mathsf{crs}) = 1 \;\middle|\; (\mathsf{crs}, \mathsf{simK}) \xleftarrow{\$} \mathsf{SSetup}(1^\lambda)\right].$$

Summarizing the above games, we have:

$$\begin{aligned} |\mathsf{pr}_6 - \mathsf{pr}_0| &\leq \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(1^\lambda) \\ &= 2\mathsf{Adv}^{\mathsf{MDDH}}_{\mathcal{D}_k,\mathbb{G}_2,\mathcal{B}}(1^\lambda). \end{aligned}$$

Thus, we conclude that the $\mathsf{tLM\text{-}NIZK}$ proof system $\mathbf{\Pi}_{\mathsf{U}}$ is zero-knowledge. □

E.4 Proof of Theorem 10

Theorem 10. *The* $\mathsf{tLM\text{-}NIZK}$ *proof system in Figure 16 is special simulation sound. Concretely, for all* PPT *adversaries* $\mathcal{A}$*, there exists a* PPT *adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{SSS}}_{\mathbf{\Pi}_{\mathsf{U}},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SIG},\mathcal{B}}(\lambda).$$

Proof. We begin by observing that the vector $\mathbf{z}$ is uniformly sampled from $\mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{D}_2)$. Consequently, at least one of $\mathbf{z}_{\mathbf{A}}$ or $\mathbf{z}_\sigma$ does not belong to $\mathsf{Span}(\mathbf{D}_2)$. This implies the existence of a vector ξ, which the challenger can derive without affecting the adversary's view, such that $\xi^\intercal \mathbf{D}_2 = 0$, and either $\xi^\intercal \mathbf{z}_{\mathbf{A}} = 1$ or $\xi^\intercal \mathbf{z}_\sigma = 1$. Similarly, there exists a vector ξ_2 such that $\xi_2^\intercal \mathbf{y}_1 = 1$ and $\xi_2^\intercal \mathbf{U}_1 = 0$. We analyze these two cases separately:

- **Case $\mathbf{z}_{\mathbf{A}} \notin \mathsf{Span}(\mathbf{D}_2)$:** In this scenario, the first verification equation is given by:

$$\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}_{\mathbf{w}}^\intercal\right]_2 - \left[\mathsf{x}\right]_1 \bullet \left[\mathbf{z}_{\mathbf{A}}^\intercal\right]_2 = \left[\mathbf{P}_{\mathbf{A}}\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2.$$

  Right-multiplying both sides by ξ yields:

$$\left[\mathbf{A}_i\right]_1 \bullet \left[\mathbf{C}_{\mathbf{w}}^\intercal \xi\right]_2 - \left[\mathsf{x}\right]_1 \bullet \left[1\right]_2 = \left[\mathbf{P}_{\mathbf{A}}\right]_1 \bullet \left[0\right]_2.$$

  Therefore, in this case, we conclude that $\mathsf{x} \in \mathsf{Span}(\mathbf{A}_i)$.

– **Case $\mathbf{z}_\sigma \notin \mathsf{Span}(\mathbf{D}_2)$:** In this scenario, we examine the remaining verification equations:
  1. $(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 = \left[\mathbf{P}_\mathbf{d}\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2$,
  2. $(\left[\underline{\mathbf{x}}^\intercal\right]_1 - \left[(\underline{\mathbf{x}}^\star)^\intercal\right]_1) \bullet \left[\mathbf{z}_\sigma^\intercal\right]_2 - \left[\bar{\mathbf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\right]_2 = \left[\mathbf{P}_\mathsf{x}\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2$,
  3. For $j \in [2k]$:
$$\left[\mathbf{y}_1\mathbf{t}^\intercal\right]_1 \bullet \left[\mathbf{u}_j\mathbf{z}_\sigma^\intercal\right]_2 - \left[\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[\mathbf{u}_j\mathbf{z}_\sigma^\intercal\right]_2 = \left[\mathbf{P}_\mathbf{t}^j\right]_1 \bullet \left[\mathbf{D}_2^\intercal\right]_2 + \left[\mathbf{U}_1\right]_1 \bullet \left[\mathbf{Q}_\mathbf{t}^j\right]_2,$$
  4. $\left[\mathbf{C}_\mathbf{s}\right]_1 \bullet \left[\mathbf{B}\right]_2 - \left[\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2 = \left[\mathbf{U}_1\right]_1 \bullet \left[\mathbf{Q}_\sigma\right]_2$.

  By right-multiplying these equations by ξ and applying $\xi_2^\intercal$ where appropriate, we obtain:
  1. $(\left[\underline{\mathbf{A}}_i^\intercal\right]_1 - \left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1) \bullet \left[1\right]_2 - \left[\bar{\mathbf{A}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\xi\right]_2 = \left[\mathbf{P}_\mathbf{d}\right]_1 \bullet \left[0\right]_2$,
  2. $(\left[\underline{\mathbf{x}}^\intercal\right]_1 - \left[(\underline{\mathbf{x}}^\star)^\intercal\right]_1) \bullet \left[1\right]_2 - \left[\bar{\mathbf{x}}^\intercal\right]_1 \bullet \left[\mathbf{C}_\mathbf{k}^\intercal\xi\right]_2 = \left[\mathbf{P}_\mathsf{x}\right]_1 \bullet \left[0\right]_2$,
  3. For $j \in [2k]$:
$$\left[\mathbf{t}^\intercal\right]_1 \bullet \left[\mathbf{u}_j\right]_2 - \left[\xi_2^\intercal\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[\mathbf{u}_j\right]_2 = \left[\mathbf{P}_\mathbf{t}^j\right]_1 \bullet \left[0\right]_2 + \left[0\right]_1 \bullet \left[\mathbf{Q}_\mathbf{t}^j\right]_2,$$
  4. $\left[\xi_2^\intercal\mathbf{C}_\mathbf{s}\right]_1 \bullet \left[\mathbf{B}\right]_2 - \left[\xi_2^\intercal\mathbf{C}_\mathbf{t}\right]_1 \bullet \left[(\mathbf{K}_0 + \mathsf{hv}\mathbf{K}_1)\mathbf{B}\right]_2 = \left[0\right]_1 \bullet \left[\mathbf{Q}_\sigma\right]_2$.

  From the above equations, the challenger can derive the vector $\left[\mathbf{C}_\mathbf{k}^\intercal\xi\right]_2$, which represents a re-encryption token from $\left[\underline{\mathbf{A}}_i\right]_1$ to $\left[(\underline{\mathbf{A}}^\star)^\intercal\right]_1$ and $\left[\underline{\mathsf{x}}_i\right]_1$ to $\left[(\underline{\mathbf{x}}^\star)^\intercal\right]_1$. Furthermore, the third set of equations implies that $\left[\mathbf{C}_\mathbf{t}\right]_1$ is a commitment to $\mathbf{t}$, and $\left[\xi_2^\intercal\mathbf{C}_\mathbf{s}\right]_1$ and $\left[\xi_2^\intercal\mathbf{C}_\mathbf{t}\right]_1$ constitute a valid signature for $\mathsf{hv} = \mathsf{H}(\left[\underline{\mathbf{A}}^\star\right]_1, \left[\underline{\mathbf{x}}^\star\right]_1, \mathsf{t})$. Therefore, unless the adversary can break the unforgeability of the underlying signature scheme, the adversary must have derived an invalid re-encryption token.

□

F Security Proof for Single-Challenge aCCA-Secure PRE

Theorem 15. *If we instantiate the general construction of* $\mathsf{PRE}_{\mathsf{aCCA}}$ *Figure 9 with*

– SMP*: a hard subset membership language distribution*
– VSHPS*: a vector space hash proof system* VSHPS *that is* (ε, d)*-universal, with* $d \geq 1$.
– OTS*: a one-time signature scheme* OTS *that is one-time strongly unforgeable.*
– tLM-NIZK*: a one-time special simulation-sound tag-based language-malleable* NIZK *proof system* tLM-NIZK *for the language associated with* SMP *and* VSHPS.

Then the resulting $\mathsf{PRE}_{\mathsf{aCCA}}$ *scheme is single-challenge* aCCA *secure against any* PPT *adversary* $\mathcal{A}$*, with advantage*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{SC\text{-}aCCA}}_{\mathsf{PRE},\mathcal{A}}(\lambda) \leq & \binom{\mathsf{N_U}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{StUnforg}}_{\mathsf{OTS},\mathcal{B}}(1^\lambda) \\ & + \mathsf{Adv}^{\mathsf{Collision}}_{\mathsf{H},\mathcal{B}}(1^\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{SSS}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{HSM}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda) \\ & + \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).\end{aligned}$$

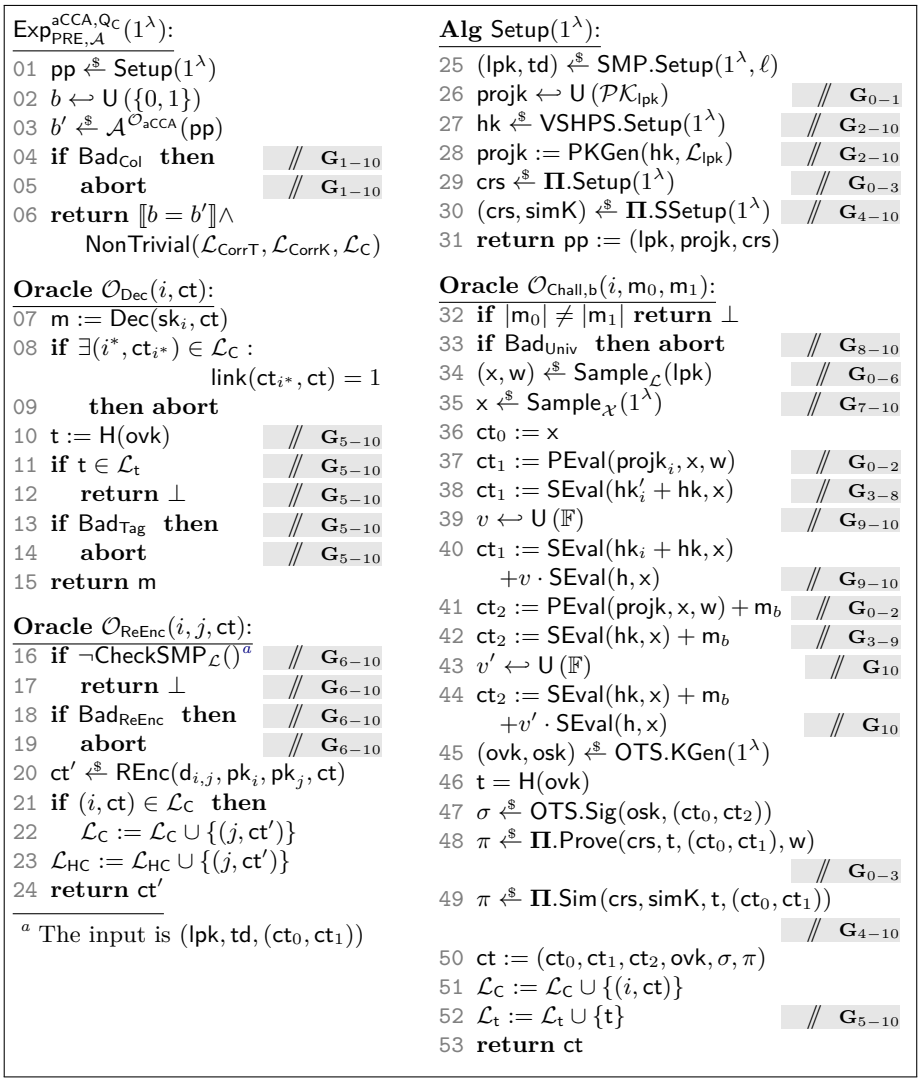

Fig. 21. Hybrid games for SC-aCCA security. We denote by $\mathsf{h} \in \mathcal{HK}$ the basis vector for the kernel space corresponding to the dimension-k linear language with support in $\mathbb{F}^\ell$. Such a vector can be generated together with the linear language.

Proof. We prove the above theorem via the following hybrid security games $\mathbf{G}_0, \ldots, \mathbf{G}_{10}$. The detailed security games are presented in Figure 21.

Game $\mathbf{G}_0$: This is the original SC-aCCA security game. By definition, the adversary's success probability is

$$\mathsf{pr}_0 := \Pr\left[\mathsf{Exp}^{\mathsf{SC\text{-}aCCA}}_{\mathsf{PRE},\mathcal{A}}(1^\lambda) = 1\right].$$

Game $\mathbf{G}_1$: This game is identical to $\mathbf{G}_0$, except that the challenger aborts if a collision occurs among the generated public keys. Let $\mathsf{Bad}_{\mathsf{Col}}$ be the event that there exist distinct $i, j \in [\mathsf{N}_{\mathsf{U}}]$ such that $\mathsf{pk}_i = \mathsf{pk}_j$. If $\mathsf{Bad}_{\mathsf{Col}}$ occurs, the game aborts.

The difference in the adversary's success probability between these two games is bounded by the probability of the event $\mathsf{Bad}_{\mathsf{Col}}$. Using a standard union bound argument, we have:

$$|\mathsf{pr}_0 - \mathsf{pr}_1| \leq \Pr[\mathsf{Bad}_{\mathsf{Col}}] \leq \binom{\mathsf{N}_{\mathsf{U}}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_2$: In this game, we modify the way of generating the public parameters. Instead of sampling the projection key projk uniformly from the set $\mathcal{PK}_{\mathsf{lpk}}$, we first sample the hash key hk using the hash proof system's setup algorithm, then compute the projection key as $\mathsf{projk} := \mathsf{PKGen}(\mathsf{hk}, \mathcal{L}_{\mathsf{lpk}})$. Since both methods of generating projk yield the same distribution, we have

$$\mathsf{pr}_2 = \mathsf{pr}_1.$$

Game $\mathbf{G}_3$: The only difference between $\mathbf{G}_3$ and $\mathbf{G}_2$ is that, the challenge ciphertext is encrypted using the hash key hk instead of using the public key projk. We generate the challenge ciphertext as

$$\mathsf{ct}_0 \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}) \quad \mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}'_i + \mathsf{hk}, \mathsf{ct}_0) \quad \mathsf{ct}_2 := \mathsf{SEval}(\mathsf{hk}, \mathsf{ct}_0) + \mathsf{m}_b$$

since $\mathsf{ct}_0 \in \mathcal{L}_{\mathsf{lpk}}$ with witness w, by the perfect correctness of HPS, the challenge ciphertext in $\mathbf{G}_3$ and $\mathbf{G}_2$ is identical. Therefore, we have

$$\mathsf{pr}_3 = \mathsf{pr}_2.$$

Game $\mathbf{G}_4$: In this game, we change the way of generating the zero-knowledge proof $\pi^\star$ in the challenge ciphertext. Instead of generating $\pi^\star$ using the witness w of ct_0, we generate $\pi^\star$ using the simulation trapdoor simK. By the zero-knowledge property of the proof system, we have

$$|\mathsf{pr}_4 - \mathsf{pr}_3| \leq \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_5$: In this game, we introduce a list $\mathcal{L}_{\mathsf{t}}$ in $\mathcal{O}_{\mathsf{Chall}}$ to store the Tag of the challenge ciphertexts, then we reject any decryption query that uses the same tag as the challenge ciphertext. Let $\mathsf{Bad}_{\mathsf{Tag}}$ be the event that the adversary submits a decryption query with the challenge tag that would otherwise be accepted by the decryption oracle. If $\mathsf{Bad}_{\mathsf{Tag}}$ occurs, the game aborts. As

in the general construction, this bad event occurs with probability at most $\mathsf{Adv}^{\mathsf{StUnforg}}_{\mathsf{OTS},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{Collision}}_{\mathsf{H},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{SSS}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda)$. Thus, we have

$$|\mathsf{pr}_5 - \mathsf{pr}_4| \leq \mathsf{Adv}^{\mathsf{StUnforg}}_{\mathsf{OTS},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{Collision}}_{\mathsf{H},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{SSS}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_6$: In this game, we reject any re-encryption query that ct_0 is not sampled from the $\mathcal{L}_{\mathsf{lpk}}$ with witness w or ct_1 is not computed as $\mathsf{ct}_1 := \mathsf{PEval}(\mathsf{projk}_i, \mathsf{ct}_0, \mathsf{w})$. Since the challenger knows the trapdoor td, it can always check the validity of the re-encryption queries. Let $\mathsf{Bad}_{\mathsf{ReEnc}}$ be the event that the adversary submits a re-encryption query that would otherwise be accepted by the re-encryption oracle. If $\mathsf{Bad}_{\mathsf{ReEnc}}$ occurs, the game aborts. By the special simulation-soundness of the proof system, we have

$$|\mathsf{pr}_6 - \mathsf{pr}_5| \leq \mathsf{Adv}^{\mathsf{SSS}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_7$: In this game, we change the way the challenge ciphertext is generated. Instead of sampling ct_0 from the language using $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk})$, we sample it from the statement set $\mathsf{ct}_0 \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda)$. Since the bad event $\mathsf{Bad}_{\mathsf{Col}}$ does not occur, the adversary cannot simply check if $\mathsf{ct}_0 \in \mathcal{L}_{\mathsf{lpk}}$ by checking if $\mathsf{SEval}(\mathsf{hk}_i - \mathsf{hk}_j, \mathsf{ct}_0) = 0$ where i, j are any two collision users.[19] By the hard subset membership property of the language, we have

$$|\mathsf{pr}_7 - \mathsf{pr}_6| \leq \mathsf{Adv}^{\mathsf{HSM}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_8$: In $\mathbf{G}_8$, we introduce a bad event $\mathsf{Bad}_{\mathsf{Univ}}$ that captures the failure of the universal hash property. Specifically, we define

$$\mathsf{Bad}_{\mathsf{Univ}} := \{\forall \mathsf{hk}'' \in \mathcal{HK}'' : \mathsf{SEval}(\mathsf{hk}'', \mathsf{x}) = 0 \mid \mathsf{ct}_0 \xleftarrow{\$} \mathsf{U}(\mathcal{X})\}.$$

We recall that $\mathcal{HK}' \subset \mathcal{HK}$ is the affine vector space, such that we can change hk to any element $\mathsf{hk}' \in \mathcal{HK}'$ without changing its projection over $\mathcal{L}_{\mathsf{lpk}}$. $\mathcal{HK}''$ is the vector space that corresponds to the affine vector space $\mathcal{HK}'$. Using the universality of VSHPS, we have

$$|\mathsf{pr}_8 - \mathsf{pr}_7| \leq \Pr[\mathsf{Bad}_{\mathsf{Univ}}] \leq \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_9$: In this game, we modify the generation of the challenge ciphertext. We sample $v \hookleftarrow \mathsf{U}(\mathbb{F})$ at the beginning of the experiment and compute the challenge ciphertext as $\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}_i, \mathsf{ct}_0) + v \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) + \mathsf{SEval}(\mathsf{hk}, \mathsf{ct}_0)$ where h is a basis of the $\mathcal{HK}''$ over the linear language $\mathcal{L}_{\mathsf{lpk}}$.

[19] If there is a collision between two public keys pk_i and pk_j, then the adversary can simply obtain $\mathsf{hk}_i - \mathsf{hk}_j \in \mathcal{HK}''$ by querying the oracle $\mathcal{O}_{\mathsf{CorrT}}(i, j)$, and check if $\mathsf{SEval}(\mathsf{hk}_i - \mathsf{hk}_j, \mathsf{ct}_0) = 0$. If yes, then $\mathsf{ct}_0 \in \mathcal{L}_{\mathsf{lpk}}$; otherwise, $\mathsf{ct}_0 \notin \mathcal{L}_{\mathsf{lpk}}$.

Since the game does not abort, we are guaranteed that $\mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) \neq 0$. As v is chosen uniformly at random from $\mathbb{F}$, the term $v \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0)$ acts as a one-time pad, perfectly masking the ciphertext component. Thus, we can also modify ct_1 to be $\mathsf{ct}_1 \leftarrow \mathsf{U}(\mathbb{F})$ without changing the adversary's view.

We first give the value of $|\mathsf{pr}_9 - \mathsf{pr}_8|$, then we explain why the above modification is valid in Lemma 5. We have

$$|\mathsf{pr}_9 - \mathsf{pr}_8| = 0.$$

Game $\mathbf{G}_{10}$: In this game, we modify the generation of the challenge ciphertext. We sample $v' \leftarrow \mathsf{U}(\mathbb{F})$ at the beginning of the experiment and compute the challenge ciphertext as $\mathsf{ct}_2 := \mathsf{SEval}(\mathsf{hk}, \mathsf{ct}_0) + v' \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) + \mathsf{m}_b$.

The proof is similar to that for $\mathbf{G}_9$: we first shift hk to $\mathsf{hk} + v' \cdot \mathsf{h}$, and then remove the $v' \cdot \mathsf{h}$ components from all oracles except $\mathcal{O}_{\mathsf{Chall}}$. Since hk is independent of the re-encryption keys $\mathsf{d}_{i,j}$, we can perform this change directly, without first considering a selective adversary; hence, the S2A lemma is not needed. Above all, we have

$$|\mathsf{pr}_{10} - \mathsf{pr}_9| = 0.$$

Finally, we observe that in $\mathbf{G}_{10}$, the challenge ciphertext is computed as

$$\begin{aligned}
&\mathsf{ct}_0 \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda) \\
&\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}_1 + \mathsf{hk}, \mathsf{ct}_0) + v \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) \\
&\mathsf{ct}_2 := \mathsf{SEval}(\mathsf{hk}, \mathsf{ct}_0) + v' \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) + \mathsf{m}_b
\end{aligned}$$

Since v' is chosen uniformly at random from $\mathbb{F}$ and $\mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) \neq 0$, the term $v' \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0)$ acts as a one-time pad, perfectly masking the message m_b. Therefore, the adversary's success probability is exactly $1/2$, and we have

$$\mathsf{pr}_{10} = \frac{1}{2}.$$

Taking union bound over all the hybrid games, we have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{SC\text{-}aCCA}}_{\mathsf{PRE},\mathcal{A}}(\lambda) \leq & \binom{\mathsf{N_U}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{StUnforg}}_{\mathsf{OTS},\mathcal{B}}(1^\lambda) \\
& + \mathsf{Adv}^{\mathsf{Collision}}_{\mathsf{H},\mathcal{B}}(1^\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{SSS}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{HSM}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda) \\
& + \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).
\end{aligned}$$

□

Lemma 5. *Games $\boldsymbol{G}_8$ and $\boldsymbol{G}_9$ are perfectly indistinguishable. That is, for any* PPT *adaptive adversary $\mathcal{A}$, we have*

$$\left| \Pr\left[\boldsymbol{G}_9^{\mathcal{A}} \Rightarrow 1 \right] - \Pr\left[\boldsymbol{G}_8^{\mathcal{A}} \Rightarrow 1 \right] \right| = 0,$$

Proof. We prove the lemma via a reduction from adaptive to selective security. The argument is structured in two parts, formalized in Lemma 6 and Lemma 3.

Lemma 6 establishes that no PPT selective adversary can distinguish between $\mathbf{G}_8$ and $\mathbf{G}_9$ with any advantage. Subsequently, Lemma 3 extends this result to every PPT adaptive adversary, which completes the proof.

□

Lemma 6. *For any* PPT *selective adversary* $\mathcal{B}$*, which by definition needs to commit to all its oracle queries before the game begins, we have:*

$$\left|\Pr\left[\boldsymbol{G}_9^{\mathcal{B}} \Rightarrow 1\right] - \Pr\left[\boldsymbol{G}_8^{\mathcal{B}} \Rightarrow 1\right]\right| = 0.$$

Proof. We prove the lemma by introducing three intermediate games, $\boldsymbol{G}_{8.1}$*,* $\boldsymbol{G}_{8.2}$ *and* $\boldsymbol{G}_{8.3}$*, and show that for any selective adversary* $\mathcal{B}$*, the following chain of perfect indistinguishability holds:* $\boldsymbol{G}_8 \approx_p \boldsymbol{G}_{8.1} \approx_p \boldsymbol{G}_{8.2} \approx_p \boldsymbol{G}_{8.3} \approx_p \boldsymbol{G}_9$*. This implies that* $\boldsymbol{G}_8$ *and* $\boldsymbol{G}_9$ *are perfectly indistinguishable for* $\mathcal{B}$*.*

The simulation strategy for these games, detailed in Figure 22, leverages the selective nature of $\mathcal{B}$*. Since all oracle queries are known beforehand, the challenger can simulate the environment without knowing most secret keys corresponding to challenge queries. Without loss of generality, we re-index users such that all challenge queries in* $\mathcal{L}_{\mathsf{C}}$ *are for users* $i \in \{1, \ldots, |\mathsf{V}_{\mathsf{CL}}|\}$*.*

Oracle $\mathcal{O}_{\mathsf{KGen}}()$:
01 $(\mathsf{lpk}, \mathsf{td}) \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$
02 $\mathsf{hk}_1 \xleftarrow{\$} \mathsf{VSHPS.Setup}(1^\lambda)$
03 $\mathsf{hk}'_1 \xleftarrow{\$} \mathsf{VSHPS.Setup}(1^\lambda)$ // $\mathbf{G}_{8.2-8.3}$
04 $v \hookleftarrow \mathsf{U}(\mathbb{F})$ // $\mathbf{G}_{8.2-8.3}$
05 $\mathsf{hk}_1 := \mathsf{hk}'_1 + v \cdot \mathsf{h}$ // $\mathbf{G}_{8.2-8.3}$
06 $\mathsf{projk}_1 \xleftarrow{\$} \mathsf{PKGen}(\mathsf{hk}_1, \mathcal{L}_{\mathsf{lpk}})$ // $\mathbf{G}_{8.1-8.2}$
07 $\mathsf{projk}_1 \xleftarrow{\$} \mathsf{PKGen}(\mathsf{hk}'_1, \mathcal{L}_{\mathsf{lpk}})$ // $\mathbf{G}_{8.3}$
08 **for** $i \in \{2, \ldots, |\mathsf{V}_{\mathsf{CL}}|\}$:
09 $\quad \mathsf{d}_i \hookleftarrow \mathsf{U}(\mathcal{HK})$
10 $\quad \mathsf{projk}_i := \mathsf{projk}_1 + \mathsf{PKGen}(\mathsf{d}_i, \mathcal{L}_{\mathsf{lpk}})$
11 $\mathsf{d}_1 = 0$
12 **for** $i \in \{|\mathsf{V}_{\mathsf{CL}}| + 1, \ldots, \mathsf{N}_{\mathsf{K}}\}$:
13 $\quad \mathsf{hk}_i \xleftarrow{\$} \mathsf{VSHPS.Setup}(1^\lambda)$
14 $\quad \mathsf{projk}_i := \mathsf{PKGen}(\mathsf{hk}_i, \mathcal{L}_{\mathsf{lpk}})$

Oracle $\mathcal{O}_{\mathsf{CorrT}}(i, j)$:
15 **if** $i, j \in \{1, \ldots, |\mathsf{V}_{\mathsf{CL}}|\}$:
16 $\quad$ **return** $\mathsf{d}_{i,j} := \mathsf{d}_j - \mathsf{d}_i$
17 **else**
18 $\quad$ **return** $\mathsf{d}_{i,j} := \mathsf{hk}_j - \mathsf{hk}_i$

Oracle $\mathcal{O}_{\mathsf{CorrK}}(i)$:
19 **if** $i \in \{|\mathsf{V}_{\mathsf{H}}| + 1, \ldots, \mathsf{N}_{\mathsf{K}}\}$:
20 $\quad$ **return** hk_i
21 **return** $\perp$

Fig. 22. Description of the hybrid games $\mathbf{G}_{8.1}$, $\mathbf{G}_{8.2}$ and $\mathbf{G}_{8.3}$ in a selective adversary. The figure first show the difference between $\mathbf{G}_{8.1}$ under selective adversary and $\mathbf{G}_8$ under adaptive adversary. Then it illustrates the changes in the key generation process from $\mathbf{G}_{8.2}$ and $\mathbf{G}_{8.3}$. Without loss of generality, we re-index keys such that challenge queries in $\mathcal{L}_{\mathsf{C}}$ correspond to the first $|\mathsf{V}_{\mathsf{CL}}|$ users. The key generation oracle $\mathcal{O}_{\mathsf{KGen}}$ is executed only once at the start.

Game $\boldsymbol{G}_{8.1}$***:*** *This game, detailed in Figure 22, is identical to* $\boldsymbol{G}_8$ *but uses a differ-*

ent simulation strategy. The challenger first generates a key pair $(\mathsf{hk}_1, \mathsf{projk}_1)$ *and for each user* $i \in \{2, \ldots, |\mathsf{V_{CL}}|\}$*, it generates* $\mathsf{projk}_i := \mathsf{projk}_1 + \mathsf{PKGen}(\mathsf{d}_{1,i}, \mathcal{L}_{\mathsf{lpk}})$ *via the re-encryption key* $\mathsf{d}_{1,i}$*. Keys for non-challenge users* $i > |\mathsf{V_{CL}}|$ *are generated independently as in the original game. The challenger can answer all oracle queries using the known secret keys of non-challenge users and the re-encryption keys for challenge users.*

The challenge ciphertext is generated using hk_1 *as in* $\boldsymbol{G}_8$*. Since the distribution of keys and ciphertexts are identical to* $\boldsymbol{G}_8$*, we have*

$$|\mathsf{pr}_{8.1} - \mathsf{pr}_8| = 0.$$

Game $\boldsymbol{G}_{8.2}$***:*** *In this game, we modify the generation of* hk_1*. Let* $\mathsf{h} \in \mathcal{HK}''$ *be a hash key such that* $\mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) \neq 0$ *for the challenge statement* ct_0*. Such an* h *is guaranteed to exist by the* $(\varepsilon, 1)$*-universality of the* VSHPS*. The challenger now computes* $\mathsf{hk}_1 := \mathsf{hk}'_1 + v \cdot \mathsf{h}$*, where* $\mathsf{hk}'_1 \xleftarrow{\$} \mathcal{HK}$ *and* $v \hookleftarrow \mathsf{U}(\mathbb{F})$*. Since* $\mathcal{HK}$ *is a vector space over* $\mathbb{F}$*, the distribution of* hk_1 *remains uniform over* $\mathcal{HK}$*. Furthermore, by the definition of* $\mathcal{HK}''$*, we have* $\mathsf{PKGen}(\mathsf{hk}_1, \mathcal{L}_{\mathsf{lpk}}) = \mathsf{PKGen}(\mathsf{hk}'_1, \mathcal{L}_{\mathsf{lpk}})$*. Thus, the adversary's view is identical to that in* $\boldsymbol{G}_{8.1}$*, and we have*

$$|\mathsf{pr}_{8.2} - \mathsf{pr}_{8.1}| = 0.$$

Game $\boldsymbol{G}_{8.3}$***:*** *This game modifies the generation of the projection key* projk_1 *by using the hash key* hk'_1 *instead of* hk_1*. Moreover, we answer all the oracle queries except* $\mathcal{O}_{\mathsf{Chall}}$ *without knowing* v*, which is only used in the challenge ciphertext generation.*

By the definition of $\mathcal{HK}''$*, we have* $\mathsf{PKGen}(\mathsf{hk}_1, \mathcal{L}_{\mathsf{lpk}}) = \mathsf{PKGen}(\mathsf{hk}'_1, \mathcal{L}_{\mathsf{lpk}})$*. Thus, the public key* projk_1 *remains unchanged. Since* $\mathcal{O}_{\mathsf{CorrK}}$ *is not queried for users* $i \in [|\mathsf{V_{CL}}|]$*,* $\mathcal{O}_{\mathsf{CorrT}}$ *queries for other challenged users* $i \in \{2, \ldots, |\mathsf{V_{CL}}|\}$ *are answered using* d_i*, and for non-challenged users* $j > |\mathsf{V_{CL}}|$*,* $\mathcal{O}_{\mathsf{CorrT}}$ *queries are answered using their known secret keys, the adversary gets no information about* hk_1 *or* hk'_1 *from* $\mathcal{O}_{\mathsf{CorrK}}$ *and* $\mathcal{O}_{\mathsf{CorrT}}$*. As for* $\mathcal{O}_{\mathsf{ReEnc}}$ *and* $\mathcal{O}_{\mathsf{Dec}}$*, by the changes in* $\boldsymbol{G}_5$ *and* $\boldsymbol{G}_6$*, the ciphertext components* ct_0 *and* ct_1 *in the re-encryption and decryption queries are guaranteed to be in the language* $\mathcal{L}_{\mathsf{lpk}}$ *with some witness. Therefore, all the information the adversary can learn from these oracles is through the evaluation of* $\mathsf{SEval}(\mathsf{hk}_i - \mathsf{hk}_1, \mathsf{ct}_0)$ *or* $\mathsf{SEval}(\mathsf{hk}_1, \mathsf{ct}_0)$ *for some users* i*. Since* $\mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) = 0$*, we have* $\mathsf{SEval}(\mathsf{hk}_1, \mathsf{ct}_0) = \mathsf{SEval}(\mathsf{hk}'_1, \mathsf{ct}_0) + v \cdot \mathsf{SEval}(\mathsf{h}, \mathsf{ct}_0) = \mathsf{SEval}(\mathsf{hk}'_1, \mathsf{ct}_0)$*, so the answer remains unchanged. Therefore, the adversary's view in* $\boldsymbol{G}_{8.2}$ *and* $\boldsymbol{G}_{8.3}$ *is identical, leading to*

$$\mathsf{pr}_{8.3} = \mathsf{pr}_{8.2}.$$

In $\boldsymbol{G}_{8.3}$*, the challenge ciphertext component* ct_1 *is computed as:*

$$\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}'_1 + v \cdot \mathsf{h}, \mathsf{ct}_0)$$

This is identical to the challenge ciphertext generation in $\boldsymbol{G}_9$*. The only difference lies in the generation of secret keys for users* $i \in \{2, \ldots, |\mathsf{V_{CL}}|\}$*. In* $\boldsymbol{G}_{8.3}$*, these keys*

are implicitly defined as $\mathsf{hk}_i = \mathsf{hk}'_1 + \mathsf{d}_i$*, while in* $\boldsymbol{G}_9$*, they are sampled uniformly from* $\mathcal{HK}$*. However, since* d_i *is uniformly random in* $\mathcal{HK}$ *and independent of* hk'_1*, the distribution of* hk'_i *remains uniform over* $\mathcal{HK}$*. Thus, the distribution of all secret keys in both games is identical. Therefore, we have*

$$\mathsf{pr}_9 = \mathsf{pr}_{8.3}.$$

Combining the above results, we conclude that

$$\left|\Pr\left[\boldsymbol{G}_9^{\mathcal{B}} \Rightarrow 1\right] - \Pr\left[\boldsymbol{G}_8^{\mathcal{B}} \Rightarrow 1\right]\right| = 0.$$

□

G Security Proof for Multi-Challenge aCCA-Secure PRE

Theorem 16. *If we instantiate the general construction of* $\mathsf{PRE}_{\mathsf{aCCA}}$ *Figure 9 with*

- SMP*: a hard subset membership language distribution*
- VSHPS*: a vector space hash proof system* VSHPS *that is* (ε, d)*-universal, with* $d \geq k$*.*
- OTS*: a one-time signature scheme* OTS *that is one-time strongly unforgeable.*
- tLM-NIZK*: an unbounded tag-based language-malleable* NIZK *proof system with special simulation soundness for the language associated with* SMP *and* VSHPS*.*

Then the resulting $\mathsf{PRE}_{\mathsf{aCCA}}$ *scheme is multi-challenge* aCCA *secure against any* PPT *adversary* $\mathcal{A}$*, with advantage*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{aCCA},\mathsf{Q_C}}_{\mathsf{PRE}_4,\mathcal{A}}(\lambda) = |\mathsf{pr}_0 - \mathsf{pr}_{14}| &\leq \sum_{i=0}^{13} \left|\mathsf{pr}_i - \mathsf{pr}_{i+1}\right| \\ &\leq \binom{\mathsf{N_U}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{StUnforg}}_{\mathsf{OTS},\mathcal{B}}(1^\lambda) \\ &\quad + \mathsf{Adv}^{\mathsf{Collision}}_{\mathsf{H},\mathcal{B}}(1^\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{SSS}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + 3 \cdot \mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda) \\ &\quad + \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).\end{aligned}$$

Proof. The proof proceeds via a sequence of hybrid games, $\mathbf{G}_0, \ldots, \mathbf{G}_{14}$. The transitions from $\mathbf{G}_0$ to $\mathbf{G}_{10}$ mirror those in the proof of Theorem 15, but are adapted for the multi-challenge setting. The key differences are as follows:

- The transition from $\mathbf{G}_6$ to $\mathbf{G}_7$ relies on the $\mathsf{Q_C}$-Hard Subset Membership ($\mathsf{Q_C}$-HSM) property. All $\mathsf{Q_C}$ challenge statements are switched from being sampled in the language via $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk})$ to being sampled uniformly from the ambient space via $\mathsf{Sample}_{\mathcal{X}}(1^\lambda)$. This introduces a difference of at most $\mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda)$.

- In $\mathbf{G}_8$, the bad event $\mathsf{Bad}_{\mathsf{Univ}}$ is defined with respect to the (ε, d)-universality of the VSHPS. The difference is bounded by $|\mathsf{pr}_8 - \mathsf{pr}_7| \leq \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda)$.
- In $\mathbf{G}_9$ and $\mathbf{G}_{10}$, the challenge ciphertext is randomized. Let $\mathbf{H} \in \mathbb{F}^{\ell \times d}$ be a basis of the $\mathcal{HK}''$ over linear language $\mathcal{L}_{\mathsf{lpk}}$. We sample random vectors $\mathbf{v}, \mathbf{v}' \in \mathbb{F}^d$ and compute the challenge ciphertext as $\mathsf{ct}_1^j := \mathsf{SEval}(\mathsf{hk}_i + \mathbf{Hv} + \mathsf{hk}, \mathsf{ct}_0^j)$ and $\mathsf{ct}_2^j := \mathsf{SEval}(\mathsf{hk} + \mathbf{Hv}', \mathsf{ct}_0^j) + \mathsf{m}_b^j$ for $j \in [\mathsf{Q_C}]$ and $i \in [\mathsf{N_U}]$ indexing the challenge users. The proof of this step is similar to the single-challenge case, but requires a more detailed explanation.
  - In $\mathbf{G}_{8.1}$, we index the first challenge user as 1 and generate its secret key hk_1 normally. We add a relative token d_i for each other challenge user i.[20] We can therefore simulate all oracle queries perfectly without knowing any other challenge user's secret key.
  - In $\mathbf{G}_{8.2}$ and $\mathbf{G}_{8.3}$, we do the same modification as in the single-challenge case, except that we now sample a random vector $\mathbf{v} \in \mathbb{F}^d$ and compute $\mathsf{hk}_1 := \mathsf{hk}'_1 + \mathbf{Hv}$, where $\mathsf{hk}'_1 \xleftarrow{\$} \mathcal{HK}$. We note that since all the challenge users share the same hash key hk_1, the random vector $\mathbf{Hv}$ is also shared among all challenge ciphertexts.
  - In $\mathbf{G}_{10}$, since all the challenge ciphertexts share the same hash key hk, the random vector $\mathbf{Hv}'$ is also shared among all challenge ciphertexts.

  As in the single-challenge case, this change is purely conceptual, so we have $\mathsf{pr}_9 = \mathsf{pr}_8$ and $\mathsf{pr}_{10} = \mathsf{pr}_9$.

The subsequent games, $\mathbf{G}_{11}$ through $\mathbf{G}_{14}$, are detailed in Figure 23 and complete the reduction.

Game $\mathbf{G}_{11}$: In this game, we change the distribution of the statement component ct_0 for all challenge ciphertexts. Instead of sampling ct_0 uniformly from the ambient space via $\mathsf{Sample}_{\mathcal{X}}(1^\lambda)$, we sample it from the language $\mathcal{L}_{\mathsf{lpk}_1}$ via $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}_1)$, where $\mathsf{lpk}_1 \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$ is a new language public key. The difference between $\mathbf{G}_{10}$ and $\mathbf{G}_{11}$ is bounded by the $\mathsf{Q_C}$-HSM property of the language.

$$|\mathsf{pr}_{11} - \mathsf{pr}_{10}| \leq \mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda).$$

Game $\mathbf{G}_{12}$: This game is a conceptual rewrite of $\mathbf{G}_{11}$. Since ct_2 is similar to ct_1, we only explain the change for ct_1.

We first express the challenge ciphertext generation in matrix form, which does not change the adversary's view. Then we rewrite the term $\mathsf{SEval}(\mathbf{Hv}, \mathsf{x})$ as $\mathbf{R}^\intercal \cdot \mathsf{x}$ where $\mathbf{R}^\intercal := \mathsf{SEval}(\mathbf{Hv}, \mathbf{u}_1) | \cdots | \mathsf{SEval}(\mathbf{Hv}, \mathbf{u}_\ell)$ is a row vector in $\mathbb{F}^{1 \times \ell}$ and $\mathbf{u}_1, \ldots, \mathbf{u}_\ell$ is the standard basis of $\mathbb{F}^\ell$. Since SEval is linear in its second argument, we have $\mathsf{SEval}(\mathbf{Hv}, \mathsf{x}) = \mathbf{R}^\intercal \cdot \mathsf{x}$. Finally, we modify the vector $\mathbf{R}$ to be uniformly random in a subspace of dimension d of $\mathbb{F}^{\ell \times 1}$, where d is the dimension of the hash key space $\mathcal{HK}''$. This change is valid because $\mathbf{v} \in \mathbb{F}^d$ is uniformly random, $\mathbf{H} \in \mathbb{F}^{\ell \times d}$ has full column rank d, and the vectors $\mathbf{u}_1, \ldots, \mathbf{u}_\ell$ are linearly

[20] The adversary need not issue $\mathcal{O}_{\mathsf{CorrT}}$ queries connecting all challenge users, so the graph induced by $\mathsf{V_{CL}}$ may be disconnected.

$\underline{\textbf{Alg } \mathcal{O}_{\mathsf{Chall},\mathsf{b}}(i, \mathsf{m}_0, \mathsf{m}_1):}$

$\mathsf{x} \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda)$ // $\mathbf{G}_{10}$

$(\mathsf{lpk}_1, \mathsf{td}_1) \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell)$ // $\mathbf{G}_{11-12}$

$(\mathsf{x}, \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}_1)$ // $\mathbf{G}_{11-12}$

$(\mathsf{lpk}'_1, \mathsf{td}'_1) \xleftarrow{\$} \mathsf{SMP.Setup}(1^\lambda, \ell + 2)$ // $\mathbf{G}_{13}$

$(\mathsf{x}', \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}'_1)$ // $\mathbf{G}_{13}$

$\mathsf{x}' \xleftarrow{\$} \mathsf{Sample}_{\mathcal{X}}(1^\lambda)$ // $\mathbf{G}_{14}$

$\mathsf{ct}_0 := \mathsf{x}$ // $\mathbf{G}_{10}$-$\mathbf{G}_{11}$

$\mathsf{ct}_1 := \mathsf{SEval}(\mathsf{hk}_i + \mathsf{hk}, \mathsf{x}) + \mathsf{SEval}(\mathbf{Hv}, \mathsf{x})$ // $\mathbf{G}_{10}$-$\mathbf{G}_{11}$

$\mathsf{ct}_2 := \mathsf{SEval}(\mathsf{hk}, \mathsf{x}) + \mathsf{m}_b + \mathsf{SEval}(\mathbf{Hv}', \mathsf{x})$ // $\mathbf{G}_{10}$-$\mathbf{G}_{11}$

$$\begin{pmatrix}\mathsf{ct}_0\\ \mathsf{ct}_1\\ \mathsf{ct}_2\end{pmatrix} := \begin{pmatrix}\mathbf{0}\\ \mathsf{SEval}(\mathsf{hk}_i + \mathsf{hk}, \mathsf{x}) + \mathbf{R}_0^\intercal \cdot \mathsf{x}\\ \mathsf{SEval}(\mathsf{hk}, \mathsf{x}) + \mathsf{m}_b + \mathbf{R}_0'^\intercal \cdot \mathsf{x}\end{pmatrix} + \begin{pmatrix}\mathsf{x}\\ \mathbf{r}^\intercal\mathbf{M} \cdot \mathsf{x}\\ \mathbf{r}'^\intercal\mathbf{M} \cdot \mathsf{x}\end{pmatrix}$$ // $\mathbf{G}_{12}$

$$\begin{pmatrix}\mathsf{ct}_0\\ \mathsf{ct}_1\\ \mathsf{ct}_2\end{pmatrix} := \begin{pmatrix}\mathbf{0}\\ \mathsf{SEval}(\mathsf{hk}_i + \mathsf{hk}, \bar{\mathsf{x}'}) + \mathbf{R}_0^\intercal \cdot \bar{\mathsf{x}'}\\ \mathsf{SEval}(\mathsf{hk}, \bar{\mathsf{x}'}) + \mathsf{m}_b + \mathbf{R}_0'^\intercal \cdot \bar{\mathsf{x}'}\end{pmatrix} + \mathsf{x}'$$ // $\mathbf{G}_{13}$-$\mathbf{G}_{14}$

$\mathsf{ct} = (\mathsf{ct}_0, \mathsf{ct}_1, \mathsf{ct}_2)$

Fig. 23. Hybrid games for aCCA security from $\mathbf{G}_{10}$ onwards.

independent. By the d-MinEntropy property of VSHPS, $\mathbf{R}$ has full min-entropy $d \cdot \log|\mathbb{F}|$. Thus we can rewrite $\mathbf{R}^\intercal$ as $\mathbf{r}^\intercal\mathbf{M} + \mathbf{R}_0^\intercal$ for some fixed $\mathbf{R}_0 \in \mathbb{F}^{\ell \times 1}$, where $\mathbf{r} \xleftarrow{\$} \mathsf{U}\left(\mathbb{F}^d\right)$ and $\mathbf{M} \in \mathbb{F}^{d \times \ell}$ is a full-row-rank matrix; this change is purely conceptual.

Thus, the games are perfectly indistinguishable, we have

$$|\mathsf{pr}_{12} - \mathsf{pr}_{11}| = 0$$

Game $\mathbf{G}_{13}$: This game reinterprets the structure of the challenge ciphertext. In $\mathbf{G}_{11}$, the vector added to the core encryption is $\begin{pmatrix}\mathsf{x}\\ \mathbf{r}^\intercal\mathbf{M} \cdot \mathsf{x}\\ \mathbf{r}'^\intercal\mathbf{M} \cdot \mathsf{x}\end{pmatrix} = \begin{pmatrix}\mathbf{I}_\ell\\ \mathbf{r}^\intercal\mathbf{M}\\ \mathbf{r}'^\intercal\mathbf{M}\end{pmatrix} \cdot \mathsf{x}$. Since $\mathbf{I}_\ell$ is the $\ell \times \ell$ identity matrix, $\mathbf{r}$ and $\mathbf{r}'$ are random vectors in $\mathbb{F}^d$ with $d \geq k = \dim(\mathcal{L}_{\mathsf{lpk}_1})$ and $\mathbf{M} \in \mathbb{F}^{d \times \ell}$ is a full-row-rank matrix independent of $\mathcal{L}_{\mathsf{lpk}_1}$, the matrix $\begin{pmatrix}\mathbf{I}_\ell\\ \mathbf{r}^\intercal\mathbf{M}\\ \mathbf{r}'^\intercal\mathbf{M}\end{pmatrix}$ is an isomorphism from $\mathcal{L}_{\mathsf{lpk}_1}$ to a new linear space of dimension k over an ambient space of dimension $\ell + 2$. We denote this new linear space as $\mathcal{L}_{\mathsf{lpk}'_1}$, and rewrite the challenge ciphertexts as $\begin{pmatrix}\mathsf{ct}_0\\ \mathsf{ct}_1\\ \mathsf{ct}_2\end{pmatrix} := \begin{pmatrix}\mathbf{0}\\ \mathsf{SEval}(\mathsf{hk}_i + \mathsf{hk}, \bar{\mathsf{x}'}) + \mathbf{R}_0^\intercal \cdot \bar{\mathsf{x}'}\\ \mathsf{SEval}(\mathsf{hk}, \bar{\mathsf{x}'}) + \mathsf{m}_b + \mathbf{R}_0'^\intercal \cdot \bar{\mathsf{x}'}\end{pmatrix} + \mathsf{x}'$, where $(\mathsf{x}', \mathsf{w}) \xleftarrow{\$} \mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}'_1)$ and $\bar{\mathsf{x}'}$ is the first ℓ components of x'.

This change is purely conceptual, so the games are perfectly indistinguishable, we have

$$\mathsf{pr}_{13} = \mathsf{pr}_{12}.$$

Game $\mathbf{G}_{14}$: In this final game, we change how x' is generated for all challenge ciphertexts. Instead of sampling x' from the language $\mathcal{L}_{\mathsf{lpk}'_1}$ via $\mathsf{Sample}_{\mathcal{L}}(\mathsf{lpk}'_1)$, we sample it uniformly at random from the ambient space $\mathbb{F}^{\ell+2}$ via $\mathsf{Sample}_{\mathcal{X}}(1^\lambda)$. The indistinguishability between $\mathbf{G}_{13}$ and $\mathbf{G}_{14}$ is guaranteed by the $\mathsf{Q_C}$-HSM property of the language $\mathcal{L}_{\mathsf{lpk}'_1}$.

$$|\mathsf{pr}_{14} - \mathsf{pr}_{13}| \leq \mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda).$$

In $\mathbf{G}_{14}$, by the above game transitions, the challenge ciphertexts are computed as $\begin{pmatrix}\mathsf{ct}_0\\ \mathsf{ct}_1\\ \mathsf{ct}_2\end{pmatrix} := \begin{pmatrix}\mathbf{0}\\ \mathsf{SEval}(\mathsf{hk}_i + \mathsf{hk}, \bar{\mathsf{x}'}) + \mathbf{R}_0^\intercal \cdot \bar{\mathsf{x}'}\\ \mathsf{SEval}(\mathsf{hk}, \bar{\mathsf{x}'}) + \mathsf{m}_b + \mathbf{R}_0'^\intercal \cdot \bar{\mathsf{x}'}\end{pmatrix} + \mathsf{x}'$, where x' is a uniformly random vector in $\mathbb{F}^{\ell+2}$. This vector acts as a one-time pad, making the resulting ciphertext $(\mathsf{ct}_0, \mathsf{ct}_1, \mathsf{ct}_2)$ uniformly random and statistically independent of the challenge bit b. Consequently, the adversary's advantage is zero.

$$\mathsf{pr}_{14} = \frac{1}{2}.$$

Combining the bounds from all game transitions yields the final security advantage:

$$\begin{aligned}\mathcal{A}^{\mathsf{aCCA},\mathsf{Q_C}}_{\mathsf{PRE}_4,\mathcal{A}}(\lambda) = |\mathsf{pr}_0 - \mathsf{pr}_{14}| &\leq \sum_{i=0}^{13} \left|\mathsf{pr}_i - \mathsf{pr}_{i+1}\right| \\ &\leq \binom{\mathsf{N_U}}{2} \cdot \mathsf{Adv}^{\mathsf{KCol}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + \mathsf{Adv}^{\mathsf{StUnforg}}_{\mathsf{OTS},\mathcal{B}}(1^\lambda) \\ &\quad + \mathsf{Adv}^{\mathsf{Collision}}_{\mathsf{H},\mathcal{B}}(1^\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{SSS}}_{\mathsf{tLM\text{-}NIZK},\mathcal{B}}(1^\lambda) + 3 \cdot \mathsf{Adv}^{\mathsf{HSM},\mathsf{Q_C}}_{\mathsf{SMP},\mathcal{B}}(1^\lambda) \\ &\quad + \mathsf{Adv}^{\mathsf{BadUniv}}_{\mathsf{VSHPS},\mathcal{B}}(1^\lambda).\end{aligned}$$

□