Registered ABE via Predicate Encodings

Ziqi Zhu[1], Kai Zhang[2], Junqing Gong[1,3], and Haifeng Qian[1]

[1] East China Normal University
[2] Shanghai University of Electric Power
[3] Shanghai Qi Zhi Institute

Abstract. This paper presents the first generic black-box construction of registered attribute-based encryption (Reg-ABE) via predicate encoding [TCC'14]. The generic scheme is based on k-Lin assumption in the prime-order bilinear group and implies the following concrete schemes that improve existing results:

- the first Reg-ABE scheme for span program in the *prime-order group*; prior work uses *composite-order group*;
- the first Reg-ABE scheme for zero inner-product predicate from *k-Lin assumption*; prior work relies on *generic group model (GGM)*;
- the first Reg-ABE scheme for *arithmetic branching program (ABP)* which has not been achieved previously.

Technically, we follow the blueprint of Hohenberger *et al.* [EUROCRYPT'23] but start from the prime-order dual-system ABE by Chen et al. [EUROCRYPT'15], which transforms a predicate encoding into an ABE. The proof follows the dual-system method in the context of Reg-ABE: we conceptually consider helper keys as secret keys; furthermore, malicious public keys are handled via pairing-based quasi-adaptive non-interactive zero-knowledge argument by Kiltz and Wee [EUROCRYPT'15].

1 Introduction

Registered attribute-based encryption (Reg-ABE) [HLWW23] is an emerging primitive that extends attribute-based encryption (ABE) [SW05,GPSW06] to avoid key escrow issue. Conceptually, this is an extension of registration-based encryption (RBE) [GHMR18]. A Reg-ABE for predicate $P : X \times Y \to \{0,1\}$ is established by publishing a common reference string crs. A user can generate his/her own key pair $(\mathsf{pk}, \mathsf{sk})$ locally and register (pk, y) for some $y \in Y$ into the system. The registration is carried out by the curator in a public and deterministic manner, and will produce a master public key mpk for encryption as traditional ABE. The user can decrypt a ciphertext for $x \in X$ using his/her sk when $P(x, y) = 1$ along with so-called helper key hsk obtained from the curator during registration phase. Furthermore, each registration might trigger an update to all users' helper keys.

Existing Reg-ABE can be classified into two classes: (1) Early work [GHMR18,GHM+19,GV20,CES21] uses non-black-box technique based on garbling scheme [Yao82,BHR12] or indistinguishable obfuscation (iO) [GGH+13,JLS22]; while (2) recent work [GKMR22,DKL+23,HLWW23,FFM+23] uses black-box technique based on concrete assumptions in bilinear group or integral lattice.

This work explores a *systematic* way to build pairing-based Reg-ABE in a black-box fashion: we want to cover a large set of functionalities in a *unified* framework. All prior work [GKMR22,HLWW23,FFM+23] focused on a single *specific* predicate. See Figure 1 for more details.

1.1 Results

In this work, we propose a generic Reg-ABE scheme via predicate encoding [Wee14,CGW15]. It works with prime-order bilinear group and the security is based on the well-known k-Lin assumption for $k \geq 1$. Given our knowledge of existing predicate encoding [Wee14,CGW15], this implies:

- the first Reg-ABE scheme for span program in the *prime-order* group; this improves the result of [HLWW23] which supports the same predicate in *composite-order* groups;

– the first Reg-ABE scheme for zero inner-product predicate from *standard assumption* (k-Lin); this *partially* resolved the open problem posted in [FFM+23]: the RIPE in [FFM+23] relies on generic group model (GGM) but achieves attribute-hiding; note that, even without attribute-hiding, the RIPE [FFM+23] does not seem to get rid of GGM;
– the first Reg-ABE scheme for arithmetic branching program (ABP) that goes *beyond span program*.

See Figure 1 for more details. We also highlight more implications thanks to the result in [ABS17] and more subsequent work on predicate encodings: we are able to come up with different variants of all Reg-ABE schemes mentioned above, such as dual of policy (i.e., "key-policy vs ciphertext-policy" transformation) and composition of policies (i.e., disjunction, conjunction and negation of predicates).

reference	**functionality**	**assumption**
[GKMR22]	Equality Check (IBE)	prime, q-type/DBDH
[HLWW23]	Span Program	composite, static
[FFM+23]	Inner-Product Predicate †	prime, GGM
§ D.1	Span Program	prime ✓, k-Lin ✓
§ D.2	Inner-Product Predicate	prime, k-Lin ✓
§ 4	Arithmetic Branching Program ✓	prime ✓, k-Lin ✓

Fig. 1. Summary of black-box construction of pairing-based Reg-ABE. In the column **assumption**, "composite" and "prime" indicate composite- and prime-order bilinear groups respectively; "static" means a specific set of static assumptions, "GGM" stands for generic group model; for k-Lin assumption, we allow $k \geq 1$. We use ✓to highlight the advantage of our scheme over prior ones supporting the same predicate.
† [FFM+23] also achieves attribute-hiding while ours in Appendix D.2 does not; we note that, without considering attribute-hiding, their scheme does not seem to be provably secure under standard assumption.

Strategy. We follow the blueprint by [HLWW23] and focus on a weaker primitive called *slotted Reg-ABE*. A slotted Reg-ABE scheme for $L \in \mathbb{N}$ slots (L-slot Reg-ABE for short) is similar to the standard Reg-ABE except that the curator is replaced by an *aggregator* who simply collects all L public keys and generate mpk and hsk's *once for all*. Here, the aggregator is stateless while the curator is stateful which allows us to register the L public keys in a one-by-one fashion. By this, we do not worry about update operations for now which can be handled by so-called "powers-of-two" approach by [HLWW23]. In particular, [HLWW23] shows that one can use the approach to generically transform any slotted Reg-ABE to a (full-fledged) Reg-ABE while preserving basic features such as predicates, assumptions, etc. In this work, we give a pairing-based slotted Reg-ABE via predicate encodings from k-Lin assumption. We provide a detailed technical overview of our slotted Reg-ABE scheme in the next two subsections.

Remarks. Before we proceed, we remark that our Reg-ABE inherits several restrictions from [HLWW23], compared with prior RBE [GHMR18,GHM+19,GV20,CES21,DKL+23]. We highlight two of them:

– Our Reg-ABE only accommodates *bounded number of users*, the size of crs depends on the number of users. Note that, almost all known RBE schemes supporting unbounded number of users [GHMR18,GHM+19,GV20,CES21] require non-black-box techniques; the only exception is the recent LWE-based scheme by Döttling *et al.* [DKL+23].
– Our Reg-ABE requires an explicit verification of public key before registration, only those "valid" public keys can be registered to the system, see Section 2.2. This is introduced by [HLWW23] to handle malicious public keys, see Section 1.3, paragraph **Handle Malicious pk**; however, this is not needed in prior RBE schemes.

It is an interesting open problem to explore whether these restrictions or relaxations are necessary to support expressive predicates. See Section 1.4 for more discussions and open problems.

1.2 Overview of Slotted ABE

In this overview, we explain our construction of slotted Reg-ABE from predicate encodings. A L-slotted Reg-ABE for $P : X \times Y \to \{0,1\}$ is governed by a crs; given $(\mathsf{pk}_1, y_1), \ldots, (\mathsf{pk}_L, y_L)$ and crs, an aggregator can generate a master public key mpk for encryption. For correctness, we require that one can use sk_i, the corresponding secret key of pk_i, to decrypt when $P(x, y_i) = 1$ where x is associated with the ciphertext. For security, when sk_i is leaked, we require that $P(x, y_i) = 0$; when sk_i is secret, it is allowed to have $P(x, y_i) = 1$; here we neglect the case where pk_i is malicious for now and handle this case later on.

Starting Point: Predicate Encoding & Dual-system ABE. Let lower-case boldface denote *row* vectors and upper-case boldface denote matrices. We first review the notion of predicate encoding and dual-system ABE [Wee14,CGW15] with the notation in [ABS17,ACGU20]. A predicate $P : X \times Y \to \{0,1\}$ has an (n, n_c, n_k)-predicate encoding (PE) if: For all $x \in X$, $y \in Y$, one can efficiently and deterministically find

$$\mathbf{C}_x \in \mathbb{Z}_p^{n \times n_c},\ \mathbf{K}_y \in \mathbb{Z}_p^{n \times n_k},\ \mathbf{a}_y \in \mathbb{Z}_p^{1 \times n_k},\ \mathbf{d}_{x,y} \in \mathbb{Z}_p^{n_c + n_k}$$

that forms $\mathbf{M}_{x,y} = \begin{pmatrix} \mathbf{a}_y & \mathbf{0}_{n_c} \\ \mathbf{K}_y & \mathbf{C}_x \end{pmatrix}$ such that

- when $P(x, y) = 1$, we have $\mathbf{M}_{x,y}\mathbf{d}_{x,y}^\top = \mathbf{e}_1^\top$;
- when $P(x, y) = 0$, we have $\{x, y, \alpha, (\alpha \| \mathbf{w})\mathbf{M}_{x,y}\} \approx_s \{x, y, \alpha, (0 \| \mathbf{w})\mathbf{M}_{x,y}\}$ where $\mathbf{w} \leftarrow \mathbb{Z}_p^n$.

In the literature, they are called α-reconstruction and α-privacy which are used to ensure correctness and security of ABE, respectively. (For the reader who is familiar with the notations in [CGW15], $\mathbf{C}_x, \mathbf{K}_y, \mathbf{a}_y$ correspond to $\mathsf{sE}, \mathsf{rE}, \mathsf{kE}$, and $\mathbf{d}_{x,y}$ corresponds to sD, rD.) Let $\mathbb{G}$ be a finite cyclic group with generator g and denote $[x] = g^x$, we will start from the following one-key ABE scheme:

$$\begin{aligned} \mathsf{mpk} &: [\mathbf{w}, \alpha]; \\ \mathsf{ct}_x &: [s, s\mathbf{w}\mathbf{C}_x], [s\alpha] \cdot \mathsf{m}; \\ \mathsf{sk}_y &: \alpha\mathbf{a}_y + \mathbf{w}\mathbf{K}_y. \end{aligned} \tag{1}$$

Decryption relies on the following equation:

$$(s \cdot (\alpha\mathbf{a}_y + \mathbf{w}\mathbf{K}_y) \| s\mathbf{w}\mathbf{C}_x)\mathbf{d}_{x,y}^\top = (s\alpha \| s\mathbf{w})\mathbf{M}_{x,y}\mathbf{d}_{x,y}^\top = (s\alpha \| s\mathbf{w})\mathbf{e}_1^\top = s\alpha \tag{2}$$

where the second equation uses the α-reconstruction of PE; security follows from the α-privacy of PE. The actual proof needs a composite-order group with subgroup decision assumption; we omit the details.

Zero-slot Scheme. The left-hand side of equation (2) immediately inspires the following (oversimplified) Reg-ABE scheme where we can embed y to mpk so that an encryption under x reveals m if and only $P(x, y) = 1$. We call this *zero-slot* scheme since there is no user to register at all.

$$\begin{aligned} \mathsf{crs} &: [\mathbf{w}, \alpha]; \\ \mathsf{mpk}_y &: [\alpha\mathbf{a}_y + \mathbf{w}\mathbf{K}_y, \mathbf{w}, \alpha]; \\ \mathsf{ct}_x &: [s\alpha\mathbf{a}_y + s\mathbf{w}\mathbf{K}_y, s\mathbf{w}\mathbf{C}_x], [s\alpha] \cdot \mathsf{m}. \end{aligned} \tag{3}$$

Observe that the structure of ct_x is quite similar to the left-hand side of (2); conceptually, we embed the *decryption procedure* (not just the functional key sk_y in scheme (1)) into mpk. Decryption uses the same equation as in scheme (1), i.e., equation (2). The security follows from the α-privacy as well as DDH assumption. In particular, the proof works in two steps: DDH assumption allows us to change the ciphertext ct_x to

$$[\tilde{\alpha}\mathbf{a}_y + \tilde{\mathbf{w}}\mathbf{K}_y, \tilde{\mathbf{w}}\mathbf{C}_x], [\tilde{\alpha}] \cdot \mathsf{m}$$

where $\tilde{\alpha}, \tilde{\mathbf{w}}$ are uniform and independent of $\alpha, \mathbf{w}$; then privacy applies w.r.t. $\tilde{\alpha}$ and $\tilde{\mathbf{w}}$. The proof is quite simple due to the fact that we actually work in the one-key setting.

From Zero to One. We proceed to modify the zero-slot scheme to allow user registration. As [HLWW23], the user will generate an ElGamal key pair: $\mathsf{pk} = [u]$ and $\mathsf{sk} = u$ where u is uniformly sampled by the user himself/herself. To register this user, we simply replace α with $\alpha + u$ in mpk_y and ct_x. This means that, in ct_x, we actually encrypt $[s\alpha]$ by ElGamal encryption under pk; the user who holds $\mathsf{sk} = u$ can recover the ciphertext in zero-slot scheme (3). In more details, the one-slot scheme is

$$\begin{aligned} \mathsf{crs} &: [\mathbf{w}, \alpha]; \\ \mathsf{pk}, \mathsf{sk} &: [u], u; \\ \mathsf{mpk}_{\mathsf{pk},y} &: [(\alpha+u)\mathbf{a}_y + \mathbf{w}\mathbf{K}_y, \mathbf{w}, \alpha]; \\ \mathsf{ct}_x &: [s, s(\alpha+u)\mathbf{a}_y + s\mathbf{w}\mathbf{K}_y, s\mathbf{w}\mathbf{C}_x], [s\alpha] \cdot \mathsf{m}. \end{aligned} \tag{4}$$

Here we add $[s]$ for correctness. Clearly, one can publicly and deterministically compute $\mathsf{mpk}_{\mathsf{pk},y}$ from crs, pk and y; this is an important feature for Reg-ABE. For security, we consider two cases:

- when u is leaked, we require that $P(x, y) = 0$, the security reduced to that for zero-slot scheme (3);
- when u is secret, we allow that $P(x, y) = 1$, the security relies on the fact that $[s\alpha]$ is hidden by $[su]$ which is basically the security of ElGamal encryption.

A caveat is that we should also allow pk to be maliciously generated by the adversary; this is a stronger attack than the first case and *cannot* be captured by the current scheme; we will defer the solution to the end of this overview. Before we proceed, we mention that an alternative way to implement our strategy is to embed $[u]$ as follows:

$$\begin{aligned} \mathsf{mpk}_{\mathsf{pk},y} &: [\alpha\mathbf{a}_y + \mathbf{w}\mathbf{K}_y, \mathbf{w}, \alpha+u]; \\ \mathsf{ct}_x &: [s, s\alpha\mathbf{a}_y + s\mathbf{w}\mathbf{K}_y, s\mathbf{w}\mathbf{C}_x], [s(\alpha+u)] \cdot \mathsf{m}. \end{aligned}$$

They are basically equivalent. We will work with (4) that makes the follow-up discussion simpler.

From One to Many: Observation. We follow the strategy of [HLWW23,FFM+23] to build L-slot scheme based on one-slot scheme that allows us to register $(\mathsf{pk}_1, y_1), \ldots, (\mathsf{pk}_L, y_L)$ for a priori known $L \in \mathbb{N}$: we generate L parallel one-slot schemes, register (pk_j, y_j) to j-th instance of one-slot scheme (or slot j for short) and "add" the corresponding $\mathsf{mpk}_{\mathsf{pk}_j,y_j}$ and ciphertext in a "component-wise" way. In particular, the scheme is as follows:

$$\begin{aligned} \mathsf{crs} &: [\mathbf{w}_j, \alpha_j], \forall j; \\ \mathsf{pk}_i &: [u_i]; \\ \mathsf{sk}_i &: u_i; \\ \mathsf{mpk} &: [\textstyle\sum_j((\alpha_j+u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), \sum_j \mathbf{w}_j, \sum_j \alpha_j]; \\ \mathsf{ct}_x &: [s, s\textstyle\sum_j((\alpha_j+u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), s\sum_j \mathbf{w}_j\mathbf{C}_x], [s\sum_j \alpha_j] \cdot \mathsf{m}; \end{aligned} \tag{5}$$

where j ranges over $1, \ldots, L$ and those terms with subscript j correspond to slot j. We encounter the same issue as in [HLWW23]: even with $\mathsf{sk}_i = u_i$ and $P(x, y_i) = 1$ for some i, we still cannot decrypt successfully as before due to the "add" operation and the solution is to issue an extra helper key hsk_i for each slot $i \in [L]$. Omitting the term with message m and fixing $i \in [L]$, the ciphertext is the "sum" of two parts:

$$\begin{aligned} &[s, s((\alpha_i+u_i)\mathbf{a}_{y_i} + \mathbf{w}_i\mathbf{K}_{y_i}), s\mathbf{w}_i\mathbf{C}_x], && \text{// local part;} \\ &[s, s\textstyle\sum_{j\neq i}((\alpha_j+u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), s\sum_{j\neq i}\mathbf{w}_j\mathbf{C}_x], && \text{// mixed part.} \end{aligned}$$

The local part corresponds to one-slot scheme for slot i and can be handled via sk_i as before, i.e., scheme (4); the mixed part involves terms from all other slots. The helper key hsk_i is designed to remove the mixed part.

From One to Many: Helper Keys via Pairing. A naive solution is to set

$$\mathsf{hsk}_i : \sum_{j\neq i}((\alpha_j + u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), \sum_{j\neq i}\mathbf{w}_j.$$

This definitely works but may suffer from "mix-and-match" attack. As an example, for $L = 3$, we have:

$$\mathsf{hsk}_2 - \mathsf{hsk}_1 + \mathsf{hsk}_3 = 2((\alpha_1 + u_1)\mathbf{a}_{y_1} + \mathbf{w}_1\mathbf{K}_{y_1}, \mathbf{w}_1)$$

this allows user in slot 1 to recover α_1 since u_1 is known to this user and $\mathsf{hsk}_1, \mathsf{hsk}_2, \mathsf{hsk}_3$ should be public. Therefore, the scheme is entirely broken. We fix the issue using the idea of achieving collusion resistance in ABE: we introduce different random coins into different hsk_i which avoids the above attack; this requires bilinear group. Let $\mathbb{G}_1 = \langle g_1\rangle, \mathbb{G}_2 = \langle g_2\rangle$ be finite cyclic source groups of bilinear maps e and $\mathbb{G}_T$ be the target group. Write $[x]_1 = g_1^x, [x]_2 = g_2^x$. We embed mpk,ct_x in $\mathbb{G}_1$ and set hsk_i over $\mathbb{G}_2$ with random coin r_i:

$$\mathsf{hsk}_i : [r_i, r_i\sum_{j\neq i}((\alpha_j + u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), r_i\sum_{j\neq i}\mathbf{w}_j]_2.$$

This is analogous to the secret key in ABE and helps to recover the local part of ct_x in the same form as before but over $\mathbb{G}_T$ with random coin sr_i instead of s:

$$[sr_i((\alpha_i + u_i)\mathbf{a}_{y_i} + \mathbf{w}_i\mathbf{K}_{y_i}), sr_i\mathbf{w}_i\mathbf{C}_x]_T$$

Then, decryption of one-slot scheme gives $[sr_i\alpha_i]_T$ when $P(x, y_i) = 1$. However, one cannot use this to carry message m: since α_i and r_i are fresh for each $i \in [L]$, we have to include terms $[sr_1\alpha_1]_T\cdot\mathsf{m}, \ldots, [sr_L\alpha_L]_T\cdot\mathsf{m}$ in ct_x for correctness, this further requires us to publish $[r_1\alpha_1]_T, \ldots, [r_L\alpha_L]_T$ in mpk, i.e., we have $|\mathsf{mpk}| = O(L)$, which is disallowed in Reg-ABE. A common trick in the context of ABE is sufficient to fix this: we will include term $[s\alpha]_T\cdot\mathsf{m}$ in ct_x as usual and connect α_i and α via term $[r_i\alpha_i + \alpha]_2$ in hsk_i. By this, we do not make any change to ct and user in slot i can compute

$$e([s]_1, [r_i\alpha_i + \alpha]_2) = [sr_i\alpha_i]_T\cdot[s\alpha]_T$$

which recovers m given $[sr_i\alpha_i]_T$ we computed before and $[s\alpha]_T\cdot\mathsf{m}$ in ct_x.

Summary. Putting all these together and writing α_j as v_j, we have the following scheme:

$$\begin{aligned}
\mathsf{crs} &= [\alpha]_T, [v_j, \mathbf{w}_j]_1, \quad \forall j;\\
&\quad\ [r_i, r_iv_j, r_i\mathbf{w}_j, r_iv_i + \alpha]_2, \quad \forall i \neq j;\\
\mathsf{pk}_i &= [u_i]_1, [u_ir_j]_2, \quad \forall j \neq i;\\
\mathsf{sk}_i &= u_i;\\
\mathsf{mpk} &= [\textstyle\sum_j((v_j + u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), \sum_j\mathbf{w}_j]_1, [\alpha]_T;\\
\mathsf{hsk}_i &= [r_i, r_i\textstyle\sum_{j\neq i}((v_j + u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), r_i\sum_{j\neq i}\mathbf{w}_j, r_iv_i + \alpha]_2;\\
\mathsf{ct}_x &= [s, s\textstyle\sum_j((v_j + u_j)\mathbf{a}_{y_j} + \mathbf{w}_j\mathbf{K}_{y_j}), s\sum_j\mathbf{w}_j\mathbf{C}_x]_1, [s\alpha]_T\cdot\mathsf{m}.
\end{aligned} \tag{6}$$

Here crs is constructed so that one can use it to generate mpk and $\mathsf{hsk}_1, \ldots, \mathsf{hsk}_L$ in a public way. To prove the security, we will need to embed (6) into composite-order group. We decide not to dive into details in the composite-order group and focus on prime-order scheme where we will handle malicious public key. Before that, we quickly mention the connect to broadcast encryption (BE) by Gentry and Waters [GW09]: neglecting all terms involving $\mathbf{w}_1, u_1, \ldots, \mathbf{w}_L, u_L$, the first row of crs is the master public key of BE, the second row of crs gives the secret keys for users $1, \ldots, L$ and ct_x is the BE ciphertext for set $[L]$. In another words, by introducing term $[r_i\alpha_i + \alpha]_2$ in hsk_i and crs in the previous paragraph, we actually employ Gentry-Waters BE [GW09] to reduce the size of ct_x and mpk from $O(L)$ to $O(1)$. Two recent results formally clarify the connection, see Section 1.4, paragraph **Concurrent Work**.

1.3 Final Slotted Reg-ABE in Prime-Order Group

Our final scheme is based on the prime-order version of scheme (6). We first explain how to get this prime-order scheme and then reach the final slotted Reg-ABE scheme with an additional concern on malicious public keys.

Prime-order Scheme. Applying the "composite-order-to-prime-order" transformation in [CGW15], we can get our scheme in the prime-order group. In more details, discarding all subscripts i and j, we do the following substitution with $\mathbf{A} \in \mathbb{Z}_p^{k\times(k+1)}$ and $\mathbf{B} \in \mathbb{Z}_p^{(k+1)\times k}$:

$$\alpha \in \mathbb{Z}_N,\ v \in \mathbb{Z}_N,\ \mathbf{w} \in \mathbb{Z}_N^n \mapsto \mathbf{k} \in \mathbb{Z}_p^{k+1},\ \mathbf{V} \in \mathbb{Z}_p^{(k+1)\times(k+1)},\ \mathbf{W} \in \mathbb{Z}_p^{(k+1)\times(k+1)n};$$

and

$$\begin{aligned}
[s]_1 \in \mathbb{G}_1, [r]_2 \in \mathbb{G}_2, [\alpha]_2 \in \mathbb{G}_2 &\mapsto [\mathbf{sA}]_1 \in \mathbb{G}_1^{1\times(k+1)}, [\mathbf{Br}^\top]_2 \in \mathbb{G}_2^{k+1}, [\mathbf{k}]_2 \in \mathbb{G}_2^{k+1}\\
[\alpha]_T \in \mathbb{G}_T, [s\alpha]_T \in \mathbb{G}_T &\mapsto [\mathbf{Ak}^\top]_T \in \mathbb{G}_T^k, [\mathbf{sAk}^\top]_T \in \mathbb{G}_T\\
[v]_1 \in \mathbb{G}_1, [\mathbf{w}]_1 \in \mathbb{G}_1^n &\mapsto [\mathbf{AV}]_1 \in \mathbb{G}_1^{k\times(k+1)}, [\mathbf{AW}]_1 \in \mathbb{G}_1^{k\times(k+1)n}\\
[sv]_1 \in \mathbb{G}_1, [s\mathbf{w}]_1 \in \mathbb{G}_1^n &\mapsto [\mathbf{sAV}]_1 \in \mathbb{G}_1^{1\times(k+1)}, [\mathbf{sAW}]_1 \in \mathbb{G}_1^{1\times(k+1)n}\\
[rv]_2 \in \mathbb{G}_2, [r\mathbf{w}]_2 \in \mathbb{G}_2^n &\mapsto [\mathbf{VBr}^\top]_2 \in \mathbb{G}_2^{k+1}, [\mathbf{W}(\mathbf{I}_n \otimes \mathbf{Br}^\top)]_2 \in \mathbb{G}_2^{(k+1)\times n}
\end{aligned}$$

Note that $u \in \mathbb{Z}_N$ is translated to $\mathbf{U} \in \mathbb{Z}_p^{(k+1)\times(k+1)}$ as $v \in \mathbb{Z}_N$ and each entry in $\mathbf{w}$ is actually treated as v too[4]. The proof is analogous to the dual-system proof for ABE [Wat09,Wee14,CGW15]:

1. we switch $[\mathbf{sA}]_1$ to a random vector $[\mathbf{c}]_1$ over $\mathbb{G}_1$;
2. for $j = 1, \ldots, L$, we switch $[\mathbf{Br}_j^\top]_2$ to a random vector $[\mathbf{d}_j^\top]_2$ over $\mathbb{G}_2$ and make use of the entropy in $\mathbf{U}_j, \mathbf{V}_j, \mathbf{W}_j$ to argue the "partial" secrecy of $\mathbf{k}$ in term $\mathbf{V}_j\mathbf{Br}_j^\top + \mathbf{k}^\top$.

Recall that we use the idea of collusion resistance to build $\mathsf{hsk}_1, \ldots, \mathsf{hsk}_L$. Therefore, in the proof, we conceptually view $\mathsf{hsk}_1, \ldots, \mathsf{hsk}_L$ as secret keys in ABE and exactly follow the dual-system method. Of course, the actual proof makes changes in crs instead of $\mathsf{hsk}_1, \ldots, \mathsf{hsk}_L$ since aggregation is public and the adversary with crs along with a series of public keys can compute them by itself, see Section 2 for formal definition.

Handle Malicious pk. We finally mention a subtlety in the proof. Recall that, in Section 1.2, we neglect the case where pk is malicious. In this case, the first step mentioned in the proof overview can not go through since the simulator does not know $\mathsf{sk} = \mathbf{U}$. In particular, the simulator takes $[\mathbf{A}, \mathbf{t}]_1$ as input where $\mathbf{t} = \mathbf{sA}$ or $\mathbf{t} = \mathbf{c}$ and need to simulate the term $[\mathbf{sAU}]_1$ (or $[\mathbf{cU}]_1$) appeared in the challenge ciphertext where $[\mathbf{AU}]_1$ is the public key registered by the adversary; clearly, this is infeasible without $\mathbf{U}$. Our solution is to allow the simulator to "program" $[\mathbf{sA}]_1$ (or $[\mathbf{c}]_1$) into crs so that the user is forced to compute $[\mathbf{sAU}]_1$ (or $[\mathbf{cU}]_1$) for us when the user submitted pk. In particular, we make two changes to the prime-order scheme.

[4] Let $\mathbf{w} = (w_1, \ldots, w_n)$. With the same substitution $w_i \in \mathbb{Z}_N \mapsto \mathbf{W}_i \in \mathbb{Z}_p^{(k+1)\times(k+1)}$ and

$$[sw_i]_1 \mapsto [\mathbf{sAW}_i]_1, \quad [rw_i]_2 \mapsto [\mathbf{W}_i\mathbf{Br}^\top]_2,$$

we have

$$\begin{aligned}
[s\mathbf{w}]_1 &= [sw_1\| \ldots \|sw_n]_1 = [\mathbf{sAW}_1\| \cdots \|\mathbf{sAW}_n]_1 = [\mathbf{sA}(\mathbf{W}_1\| \cdots \|\mathbf{W}_n)]_1\\
[r\mathbf{w}]_2 &= [rw_1\| \ldots \|rw_n]_2 = [\mathbf{W}_1\mathbf{Br}^\top\| \cdots \|\mathbf{W}_n\mathbf{Br}^\top]_2 = [(\mathbf{W}_1\| \cdots \|\mathbf{W}_n)(\mathbf{I}_n \otimes \mathbf{Br}^\top)]_2
\end{aligned}$$

where we obtain $\mathbf{W} = (\mathbf{W}_1\| \cdots \|\mathbf{W}_n) \in \mathbb{Z}_p^{(k+1)\times(k+1)n}$.

1. We introduce an extra term $[\mathbf{R}]_1$ where $\mathbf{R} \leftarrow \mathbb{Z}_p^{(k+2)\times(k+1)}$ to crs; user's public key also includes an extra term $[\mathbf{RU}]_1$. In the reduction, we program

$$\mathbf{R} = \widetilde{\mathbf{R}} \begin{pmatrix} \mathbf{t} \\ \mathbf{I}_{k+1} \end{pmatrix}, \quad \widetilde{\mathbf{R}} \leftarrow \mathbb{Z}_p^{(k+2)\times(k+2)}$$

   In both cases, $\widetilde{\mathbf{R}}$ ensures that $\mathbf{R}$ is random. Receiving $\mathsf{pk} = [\mathbf{T} = \mathbf{AU}, \mathbf{Q} = \mathbf{RU}]_1$, we use $[\mathbf{e}_1\widetilde{\mathbf{R}}^{-1}\mathbf{Q}]_1 = [\mathbf{tU}]_1$ to simulate the ciphertext, which is either $[\mathbf{sAU}]_1$ or $[\mathbf{cU}]_1$ as required.
2. Since the adversary can give an inconsistent pk where $\mathbf{T} = \mathbf{AU}$ and $\mathbf{Q} = \mathbf{RU}'$ with $\mathbf{U} \neq \mathbf{U}'$. We additionally ask for a proof π showing

$$\begin{pmatrix} \mathbf{T} \\ \mathbf{Q} \end{pmatrix} \in \mathsf{span} \begin{pmatrix} \mathbf{A} \\ \mathbf{R} \end{pmatrix}$$

   This ensures $\mathbf{U} = \mathbf{U}'$. One can generate the proof via any non-interactive zero-knowledge proof/argument (NIZK) for sufficiently large language such as Groth-Sahai Proof [GS08]. In this work, we choose to employ quasi-adaptive NIZK (QA-NIZK) for linear space from pairing [JR13] due to the fact that $[\mathbf{A}]_1$ and $[\mathbf{R}]_1$ (i.e., the language) are determined at a quite early stage. We mention that we need a stronger unbounded simulation soundness [GHR15,LPJY15] where the adversary is given $\mathbf{A}$ and $\mathbf{R}$ "in the clear"; we leave more details to Section 2.4.

However, the additional term $[\mathbf{Q}]_1$ leaks almost all information of $\mathbf{U}$, which is crucial for the security when the user is honest. To fix the issue, we employ a wider $\mathbf{A}$ and $\mathbf{R}$ along with a higher $\mathbf{U}$ so that given $\mathbf{AU}, \mathbf{RU}$, we still have left-over entropy in $\mathbf{cU}$ for the security; see Section 3.4 for more details. We finally note that our method is indeed inspired by the idea of [HLWW23] in the composite-order group, however, this is not derived from theirs via a composite-order-to-prime-order transformation.

1.4 Discussions

On Hohenberger et al.'s Reg-ABE [HLWW23]. The recent work [HLWW23] showed a registered CP-ABE for span program and mentioned that "... if we ignore the slot-specific ciphertext component, then the structure of the ciphertexts in our scheme coincides with those in the ciphertext policy ABE scheme of Lewko et al. [LOS+10]." But the connection with predicate encoding is not as straightforward as stated. For $S \subseteq [n]$, let us define $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ where $x_i = 1$ for $i \in S$ and $x_i = 0$ for $i \notin S$. [HLWW23] uses the following unusual structure to encode S in mpk and ct (Note that we are not showing mpk and ct *accurately*, there are some minor differences.):

$$\mathsf{mpk} : \{(1-x_i)w_i\}_{i\in[n]} \quad \text{and} \quad \mathsf{ct} : \{\alpha_i + (1-x_i)w_i s_i, s_i\}_{i\in[n]}$$

where $\mathbf{w} = (w_1, \ldots, w_n)$ is the public parameter and α_i are secret sharings of a secret value according to the policy that associated with ct. The key point is the fact that term $\alpha_i + (1-x_i)w_i s_i$ in ct encodes both the policy (via α_i's) and set (i.e., x_i); this is not the case in predicate encodings where we encode them separately (due to the syntax of standard ABE). However, a simple calculation shows that

$$\alpha_i + (1-x_i)w_i s_i = (\alpha_i + w_i s_i) - (x_i w_i s_i), \quad \forall i \in [n] \tag{7}$$

namely we can easily "unpack" ct as

$$\mathsf{ct}' : s_i, \{\alpha_i + w_i s_i\}_{i\in[n]}, \{x_i w_i s_i\}_{i\in[n]}$$

where the two terms encode policy (via α_i) and set separately; in fact, they are exactly the encoding for CP-ABE presented in [CGW15, Appendix A.5] and equation (7) is actually the first step of *decryption*. This clarifies the connection between ours and [HLWW23]; this also suggests a possibility of optimizing the efficiency. Roughly, this requires some kind of pre-processing property for predicate encoding and we leave this as a future work.

Towards (Weak) Attribute-Hiding. As we have mentioned in Section 1.1, the RIPE proposed in [FFM+23] achieves attribute-hiding which roughly means that x associated with the ciphertext is also hidden from the adversary. Given the notion of attribute-hiding predicate encoding formulated in [CGW15], it is expected that our scheme can also support *weak* attribute-hiding (as the dual-system ABE via predicate encoding in [CGW15]). However, we argue that this is not straight-forward as expected: in order to remove the mixed part from the ciphertext using helper key, the decryption procedure needs to know x to get $\mathbf{C}_x$, see scheme (6) and Section 3.1; therefore, even with attribute-hiding predicate encoding, our Reg-ABE does not achieve (weak) attribute-hiding. It is still open to get (weak) attribute-hiding *under standard assumption* such as k-Lin; note that the Reg-IPE by [FFM+23] indeed achieves attribute-hiding but in the *generic group model*.

More Expressive Reg-ABE from Pair Encoding. Pair Encoding proposed by Attrapadung [Att14] is a more powerful tool to build ABE; for instance, this allows us to support multi-use of attribute and uniform computation such as DFA. However, our scheme can not work with pair encoding in a straight-forward way. We provide a quick discussion: Compared with the predicate encoding whose security is information-theoretical, the security of pair encoding (especially, for those predicates we just mentioned) is defined computationally when encodings w.r.t. ciphertext and key (analogous to $\mathbf{C}_x$ and $\mathbf{K}_y$) are encoded over $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. However, in the context of our Reg-ABE scheme, we encode both of them over $\mathbb{G}_1$ and thus all existing pair encodings with computational security should be revised. We leave this as a future work to adapt the notion of pair encoding and build Reg-ABE from this. Furthermore, we point out that the use of pair encoding may introduce strong assumptions such as q-type assumption. To obtain those functionalities and properties we mentioned at the beginning under standard assumptions, more work will be needed to adapt specialized solutions for ABE such as [KW19,GWW19,GW20,LL20] to the context of Reg-ABE.

Concurrent Work. As an independent work, Freitag *et al.* [FWW23] proposed a Reg-ABE scheme for arbitrary circuit families from witness encryption (WE) [GGSW13] and newly proposed function-binding hash function. Given the WE in [VWW22], the scheme can be based on (evasive) LWE. In contract to our work and the pairing-based construction in [HLWW23], this construction is more like iO-based Reg-ABE in [HLWW23]: it enjoys transparent setup, supports unbounded number of users. However, it only achieves a weaker notion of selective-policy security without corruption in the standard model; the restriction on corruption can be removed in the random oracle model. Furthermore, this work also pointed out that Reg-ABE implies flexible/distributed broadcast encryption. Applying this observation, we mention that our Reg-ABE scheme implies the recent distributed broadcast encryption based on k-Lin assumption [KMW23]; their another construction based on DBHE assumption [KMW23] does not seem to be relevant to our Reg-ABE scheme.

Organization. Our paper is organized as follows: We review some background knowledge in Section 2. Section 3 presents our slotted Reg-ABE via predicate encoding, this readily implies full-fledged Reg-ABE. We show the first slotted Reg-ABE for ABP in Section 4 and more concrete instantiations in Appendix D.

2 Preliminaries

Notations. For a finite set S, we use $s \leftarrow S$ to denote the procedure of sampling s from S uniformly. For an ordered list or array $\mathcal{L}$, we use $|\mathcal{L}|$ to denote its size (i.e., the number of entries in the list) and use $\mathcal{L}[i]$ to refer to its i-th entry. When $i > |\mathcal{L}|$ or $i < 1$, we define $\mathcal{L}[i] = \perp$; when we append x to $\mathcal{L}$, we set $\mathcal{L}[|\mathcal{L}|+1] = x$. We use $\star$ as wildcard. Let $\approx_s$ (resp. $\approx_c$) stand for two distributions being statistically (resp. computationally) indistinguishable. We use lower-case boldface to denote *row* vectors (e.g., $\mathbf{a}$) and upper-case boldface to denote matrices (e.g. $\mathbf{M}$). We let $\mathbf{e}_1 = (1, 0, \ldots, 0)$ of proper dimension (which is clear from the context) and use "$\|$" to denote vector or matrix concatenation (e.g. $(\mathbf{A}\|\mathbf{B})$).

Kronecker Product. Let $\mathbb{F}$ be a field. The *Kronecker Product* for matrices $\mathbf{A} = (a_{i,j}) \in \mathbb{F}^{\ell \times m}$ and $\mathbf{B} \in \mathbb{F}^{n \times p}$ is

$$\mathbf{A} \otimes \mathbf{B} = (a_{i,j}\mathbf{B}) = \begin{pmatrix} a_{1,1}\mathbf{B} & \cdots & a_{1,m}\mathbf{B} \\ \vdots & & \vdots \\ a_{\ell,1}\mathbf{B} & \ldots & a_{\ell,m}\mathbf{B} \end{pmatrix} \in \mathbb{F}^{\ell n \times mp}. \tag{8}$$

For matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ of proper sizes, we have $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$.

2.1 Prime-Order Bilinear Groups

Assume an efficient algorithm $\mathcal{G}$ that takes as input a security parameter 1^λ and outputs $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$. Here $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are cyclic groups of prime order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a non-degenerate bilinear map, and all group operations and bilinear map are efficient. Let $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$ and $g_T = e(g_1, g_2)$, we employ *implicit representation* of group elements: for a matrix $\mathbf{M} = (m_{ij})$ over $\mathbb{Z}_p$, define $[\mathbf{M}]_s = g_s^{\mathbf{M}} = (g_s^{m_{ij}})$ for all $s \in \{1, 2, T\}$; given $[\mathbf{A}]_1, [\mathbf{B}]_2$, we write $e([\mathbf{A}]_1, [\mathbf{B}]_2) = [\mathbf{AB}]_T$. We review *matrix Diffie-Hellman (MDDH) assumption* [EHK+13]; it is shown that it is implied by k-Lin [EHK+13].

Assumption 1 ((k, ℓ, d)-MDDH over $\mathbb{G}_s$, $s \in \{1, 2\}$) *Let $k, \ell, d \in \mathbb{N}$ with $k < \ell$. We say that the (k, ℓ, d)-MDDH assumption holds in $\mathbb{G}_s$ if for all PPT adversaries $\mathcal{A}$, the following advantage function is negligible in λ.*

$$\mathsf{Adv}^{\mathrm{MDDH}}_{\mathcal{A},s,k,\ell,d}(\lambda) = \left| \Pr[\mathcal{A}(\mathbb{G}, [\mathbf{M}]_s, [\mathbf{SM}]_s) = 1] - \Pr[\mathcal{A}(\mathbb{G}, [\mathbf{M}]_s, [\mathbf{U}]_s) = 1] \right|$$

where $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$, $\mathbf{M} \leftarrow \mathbb{Z}_p^{k \times \ell}$, $\mathbf{S} \leftarrow \mathbb{Z}_p^{d \times k}$ and $\mathbf{U} \leftarrow \mathbb{Z}_p^{d \times \ell}$.

2.2 Slotted Registered Attribute-Based Encryption

We review the notion of *slotted* registered attribute-based encryption (Reg-ABE) adapted from [HLWW23]. The formal definition of Reg-ABE can be found in Appendix A along with a brief overview of "slotted Reg-ABE $\Longrightarrow$ Reg-ABE".

Algorithms. A slotted registered attribute-based encryption (Reg-ABE) for predicate $P : X \times Y \to \{0, 1\}$ consists of six efficient algorithms:

- $\mathsf{Setup}(1^\lambda, P, 1^L) \to \mathsf{crs}$: It takes as input the security parameter 1^λ, description of predicate P and the upper bound 1^L of the number of slots, outputs a common reference string crs.
- $\mathsf{Gen}(\mathsf{crs}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$: It takes as input crs and slot number $i \in [L]$, outputs key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$.
- $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i) \to 0/1$: It takes as input crs, i, pk_i and outputs a bit indicating whether pk_i is valid.
- $\mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i, y_i)_{i \in [L]}) \to (\mathsf{mpk}, (\mathsf{hsk}_j)_{j \in [L]})$: It takes as input crs and a series of pk_i with $y_i \in Y$ for all $i \in [L]$, outputs master public key mpk and a series of helper keys hsk_j for all $j \in [L]$. This algorithm is deterministic.
- $\mathsf{Enc}(\mathsf{mpk}, x, \mathsf{m}) \to \mathsf{ct}$: It takes as input mpk, $x \in X$ and message m, outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{hsk}, \mathsf{ct}) \to \mathsf{m}/\perp$: It takes as input sk, hsk, ct and outputs m or a special symbol $\perp$.

For Setup, input P has different meanings for different predicates: for span program, it indicates the number of attributes; for inner-product predicates, it indicates the dimension of vectors, see Section 4 and Appendix D. We also note that we use two different indices i and j for pk_i and hsk_j, respectively; both of them range from 1 to L but this convention will simplify the exposition.

Completeness. For all $\lambda, L \in \mathbb{N}$, all P, and all $i \in [L]$, we have

$$\Pr\left[\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i) = 1 \,\middle|\, \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, P, 1^L);\ (\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(\mathsf{crs}, i)\right] = 1.$$

Correctness. For all $\lambda, L \in \mathbb{N}$, all P, all $i^* \in [L]$, all $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, P, 1^L)$, all $(\mathsf{pk}_{i^*}, \mathsf{sk}_{i^*}) \leftarrow \mathsf{Gen}(\mathsf{crs}, i^*)$, all $\{\mathsf{pk}_i\}_{i\in[L]\setminus\{i^*\}}$ such that $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i) = 1$, all $x \in X$ and $y_1, \ldots, y_L \in Y$ such that $P(x, y_{i^*}) = 1$, and all m, we have

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}_{i^*}, \mathsf{hsk}_{i^*}, \mathsf{ct}) = \mathsf{m} \,\middle|\, (\mathsf{mpk}, (\mathsf{hsk}_j)_{j\in[L]}) \leftarrow \mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i, y_i)_{i\in[L]});\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x, \mathsf{m})\right] = 1.$$

Compactness. For all $\lambda, L \in \mathbb{N}$, all P, and all $i \in [L]$, we have

$$|\mathsf{mpk}| = \mathsf{poly}(\lambda, P, \log L) \quad \text{and} \quad |\mathsf{hsk}_i| = \mathsf{poly}(\lambda, P, \log L).$$

Security. For all stateful adversary $\mathcal{A}$, the advantage

$$\Pr\left[b = b' \,\middle|\, \begin{array}{l} L \leftarrow \mathcal{A}(1^\lambda);\ \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, P, 1^L) \\ (\mathsf{pk}_i^*, y_i^*)_{i\in[L]}, x^*, \mathsf{m}_0^*, \mathsf{m}_1^* \leftarrow \mathcal{A}^{\mathsf{OGen}(\cdot),\mathsf{OCor}(\cdot)}(\mathsf{crs}) \\ (\mathsf{mpk}, (\mathsf{hsk}_j)_{j\in[L]}) \leftarrow \mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i^*, y_i^*)_{i\in[L]}) \\ b \leftarrow \{0,1\}, \mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{mpk}, x^*, \mathsf{m}_b^*);\ b' \leftarrow \mathcal{A}(\mathsf{ct}^*) \end{array}\right] - \frac{1}{2}$$

is negligible in λ, where the oracles work as follows with initial setting $C = \emptyset$ and $\mathcal{D}_i = \emptyset$ for all $i \in [L]$:

- $\mathsf{OGen}(i)$: run $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}, i)$, set $\mathcal{D}_i[\mathsf{pk}] = \mathsf{sk}$ and return pk.
- $\mathsf{OCor}(i, \mathsf{pk})$: return $\mathcal{D}_i[\mathsf{pk}]$ and update $C = C \cup \{(i, \mathsf{pk})\}$.

and, for all $i \in [L]$, we require that

$$\begin{aligned} \mathcal{D}_i[\mathsf{pk}_i^*] = \perp &\implies \mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i^*) = 1, \\ (i, \mathsf{pk}_i^*) \in C \vee \mathcal{D}_i[\mathsf{pk}_i^*] = \perp &\implies P(x^*, y_i^*) = 0. \end{aligned}$$

We use $\mathsf{Adv}^{\text{sReg-ABE}}_{\mathcal{A}}(\lambda)$ to denote the advantage function. Note that [HLWW23] showed that there is no need to give mpk and $\mathsf{hsk}_1, \ldots, \mathsf{hsk}_L$ to $\mathcal{A}$ explicitly and to consider post-challenge queries.

2.3 Predicate Encodings

We review the notion of predicate encoding [Wee14,CGW15]; for simplicity, we use the formulation in [ABS17,ACGU20]. A predicate $P : X \times Y \to \{0,1\}$ has a (n, n_c, n_k)-predicate encoding if: For all $x \in X$, $y \in Y$, there exist

$$\mathbf{C}_x \in \mathbb{Z}_p^{n\times n_c},\ \mathbf{K}_y \in \mathbb{Z}_p^{n\times n_k},\ \mathbf{a}_y \in \mathbb{Z}_p^{1\times n_k},\ \mathbf{d}_{x,y} \in \mathbb{Z}_p^{1\times(n_k+n_c)}$$

such that, letting

$$\mathbf{M}_{x,y} = \begin{pmatrix} \mathbf{a}_y & \mathbf{0}_{n_c} \\ \mathbf{K}_y & \mathbf{C}_x \end{pmatrix} \in \mathbb{Z}_p^{(1+n)\times(n_k+n_c)}$$

we have

- **correctness:** for $x \in X$ and $y \in Y$ such that $P(x, y) = 1$:

$$\mathbf{M}_{x,y}\mathbf{d}_{x,y}^\top = \mathbf{e}_1^\top;$$

- **security:** for $x \in X$ and $y \in Y$ such that $P(x, y) = 0$ and for all $\alpha \in \mathbb{Z}_p$:

$$\{x, y, \alpha, (\alpha\|\mathbf{w})\mathbf{M}_{x,y}\} \approx_s \{x, y, \alpha, (0\|\mathbf{w})\mathbf{M}_{x,y}\}, \quad \mathbf{w} \leftarrow \mathbb{Z}_p^n.$$

Also, we require that (1) given P, one can efficiently determine n, n_c, n_k; (2) given x, one can efficiently compute $\mathbf{C}_x$; (3) given y, one can efficiently compute $\mathbf{K}_y$ and $\mathbf{a}_y$; (4) given both x and y, one can efficiently compute $\mathbf{d}_{x,y}$.

2.4 Quasi-Adaptive Non-Interactive Zero-Knowledge Argument

We review the notion of quasi-adaptive non-interactive zero-knowledge argument (QA-NIZK) tailored for linear space over group [JR13,KW15]. In this paper, we require a stronger unbounded simulation soundness in [GHR15,LPJY15].

Algorithms. A Quasi-adaptive Non-interactive Zero-knowledge Argument (QA-NIZK) for linear space over bilinear group $\mathbb{G}$ [JR13,KW15] consists of four efficient algorithms:

- $\mathsf{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [\mathbf{M}]_1) \to (\mathsf{crs}, \mathsf{td})$: It takes as input the security parameter 1^λ, language parameter $1^n, 1^m, 1^\ell$, and a matrix $[\mathbf{M}]_1 \leftarrow \mathbb{G}_1^{n\times m}$ defining a linear space, outputs common reference string crs and trapdoor td.
- $\mathsf{LPrv}(\mathsf{crs}, [\mathbf{Y}]_1, \mathbf{X}) \to \pi$: It takes as input crs, a matrix $[\mathbf{Y}]_1 \in \mathbb{G}_1^{n\times \ell}$ with witness $\mathbf{X} \in \mathbb{Z}_p^{m\times \ell}$, outputs a proof π.
- $\mathsf{LVer}(\mathsf{crs}, [\mathbf{Y}]_1, \pi) \to 0/1$: It takes as input crs, $[\mathbf{Y}]_1$ and π, outputs a bit indicating the validity of π.
- $\mathsf{LSim}(\mathsf{crs}, \mathsf{td}, [\mathbf{Y}]_1) \to \widetilde{\pi}$: It takes as input crs, td, $[\mathbf{Y}]_1$, outputs a simulated proof $\widetilde{\pi}$.

Perfect Completeness. For all λ, $\mathbf{M}$, and all $\mathbf{X}, \mathbf{Y}$ such that $\mathbf{Y} = \mathbf{M}\mathbf{X}$:

$$\Pr\left[\mathsf{LVer}(\mathsf{crs}, [\mathbf{Y}]_1, \pi) = 1 \;\middle|\; (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [\mathbf{M}]_1);\ \pi \leftarrow \mathsf{LPrv}(\mathsf{crs}, [\mathbf{Y}]_1, \mathbf{X})\right] = 1.$$

Perfect Zero-knowledge. For all λ, $\mathbf{M}$, $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [\mathbf{M}]_1)$, and all $\mathbf{X}, \mathbf{Y}$ such that $\mathbf{Y} = \mathbf{M}\mathbf{X}$:

$$\mathsf{LPrv}(\mathsf{crs}, [\mathbf{Y}]_1, \mathbf{X}) \equiv \mathsf{LSim}(\mathsf{crs}, \mathsf{td}, [\mathbf{Y}]_1).$$

Stronger Unbounded Simulation Soundness. For all adversary $\mathcal{A}$, the advantage

$$\Pr\left[\begin{array}{l|l} ([\mathbf{Y}^*]_1, \mathsf{pk}^*) \notin Q \quad \wedge & \mathbf{M} \leftarrow \mathbb{Z}_p^{n\times m} \\ \mathbf{Y}^* \notin \mathsf{span}(\mathbf{M}) \quad \wedge & (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [\mathbf{M}]_1) \\ \mathsf{LVer}(\mathsf{crs}, [\mathbf{Y}^*]_1, \pi^*) = 1 & ([\mathbf{Y}^*]_1, \pi^*) \leftarrow \mathcal{A}^{\mathsf{LSim}(\mathsf{crs},\mathsf{td},\cdot)}(1^\lambda, \mathsf{crs}, \mathbf{M}) \end{array}\right]$$

is negligible in λ, where Q records all queries to $\mathsf{LSim}(\mathsf{crs}, \mathsf{td}, \cdot)$ along with responses. We use $\mathsf{Adv}^{\mathrm{USS}}_{\mathcal{A},n,m,\ell}(\lambda)$ to denote the advantage function. Note that our definition is stronger in the sense that the adversary is given $\mathbf{M}$ instead of $[\mathbf{M}]_1$, this allows us to manipulate $\mathbf{M}$ in reduction (see the proof of Lemma 5 and [GHR15,LPJY15] for more discussions).

Scheme from Pairings. Due to the simplicity and efficiency, we choose to use QA-NIZK in [KW15] for the case $\ell = 1$. It is direct to verify that this scheme achieves stronger unbounded simulation soundness (defined above) under MDDH assumption; see Appendix B. For general $\ell > 1$, we simply employ ℓ parallel fresh instances.

3 Our Slotted Registered ABE

This section presents our slotted Reg-ABE via predicate encoding from k-Lin assumption. By the generic transformation in [HLWW23], this yields a Reg-ABE scheme via predicate encoding under the k-Lin assumption, cf. Appendix A. We provide some concrete instances in Section 4 and Appendix D.

3.1 Scheme

Assuming a QA-NIZK $\Pi = (\mathsf{LGen}, \mathsf{LPrv}, \mathsf{LVer}, \mathsf{LSim})$ for linear space over bilinear groups, our slotted Reg-ABE scheme for predicates that have predicate encoding works as follows in the prime-order bilinear group:

– $\mathsf{Setup}(1^\lambda, P, 1^L)$: Run $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$. Sample

$$\mathbf{A} \leftarrow \mathbb{Z}_p^{k\times(2k+1)},\ \mathbf{B} \leftarrow \mathbb{Z}_p^{(k+1)\times k},\ \mathbf{k} \leftarrow \mathbb{Z}_p^{1\times(2k+1)}.$$

Compute parameter (n, n_c, n_k) from P, see Section 2.3. For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)n},\ \mathbf{R}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+1)},\ \mathbf{r}_i \leftarrow \mathbb{Z}_p^{1\times k}.$$

For all $i \in [L]$, write $\mathbf{A}_i = \begin{pmatrix}\mathbf{A}\\ \mathbf{R}_i\end{pmatrix} \in \mathbb{Z}_p^{(3k+2)\times(2k+1)}$ and run

$$(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1).$$

Output

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1,\ [\mathbf{A}\mathbf{k}^\top]_T,\ \{\mathsf{crs}_i, [\mathbf{R}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\}_{i\in[L]} \\ \{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\}_{j\in[L]} \\ \{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}.$$

We note that we employ i as the index for $\mathbf{V}$'s and $\mathbf{W}$'s while j is the index for $\mathbf{r}$'s; both of them range from 1 to L. One exception is the terms with $\mathbf{k}$, which is conceptually $\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top$ with $i = j$. This is different from our notation in Section 1.2. Note that we do not use $\mathsf{td}_1, \ldots, \mathsf{td}_L$ in the actual scheme.

– $\mathsf{Gen}(\mathsf{crs}, i)$: Sample $\mathbf{U}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)}$. Define $\mathbf{M}_i = \begin{pmatrix}\mathbf{T}_i\\ \mathbf{Q}_i\end{pmatrix} = \begin{pmatrix}\mathbf{A}\mathbf{U}_i\\ \mathbf{R}_i\mathbf{U}_i\end{pmatrix} = \mathbf{A}_i\mathbf{U}_i \in \mathbb{Z}_p^{(3k+2)\times(k+1)}$ and run

$$\pi_i \leftarrow \mathsf{LPrv}(\mathsf{crs}_i, [\mathbf{M}_i]_1, \mathbf{U}_i).$$

Fetch $[\mathbf{R}_i]_1$ and $\{[\mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{i\}}$ from crs and output

$$\mathsf{pk}_i = \big([\underbrace{\mathbf{A}\mathbf{U}_i}_{\mathbf{T}_i}, \underbrace{\mathbf{R}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{\underbrace{[\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top]_2}_{\mathbf{h}_{i,j}^\top}\}_{j\in[L]\setminus\{i\}}, \pi_i\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

– $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i)$: Parse $\mathsf{pk}_i = \big([\mathbf{T}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\big)$. Write $\mathbf{M}_i = \begin{pmatrix}\mathbf{T}_i\\ \mathbf{Q}_i\end{pmatrix}$ and check

$$\mathsf{LVer}(\mathsf{crs}_i, [\mathbf{M}_i]_1, \pi_i) \stackrel{?}{=} 1.$$

For each $j \in [L] \setminus \{i\}$, check

$$e([\mathbf{A}]_1, [\mathbf{h}_{i,j}^\top]_2) \stackrel{?}{=} e([\mathbf{T}_i]_1, [\mathbf{B}\mathbf{r}_j^\top]_2).$$

If all these checks pass, output 1; otherwise, output 0.

– $\mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i, y_i)_{i\in[L]})$: For all $i \in [L]$, compute $\mathbf{K}_{y_i}$ from y_i, and parse $\mathsf{pk}_i = \big([\mathbf{T}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\big)$. Output:

$$\mathsf{mpk} = \left([\mathbf{A}]_1, \left[\sum_{i\in[L]} ((\mathbf{A}\mathbf{V}_i + \mathbf{T}_i)(\mathbf{a}_{y_i} \otimes \mathbf{I}_{k+1}) + \mathbf{A}\mathbf{W}_i(\mathbf{K}_{y_i} \otimes \mathbf{I}_{k+1}))\right]_1, \left[\sum_{i\in[L]} \mathbf{A}\mathbf{W}_i\right]_1,\ [\mathbf{A}\mathbf{k}^\top]_T\right)$$

and for all $j \in [L]$

$$\mathsf{hsk}_j = \left([\underbrace{\mathbf{B}\mathbf{r}_j^\top}_{\mathbf{k}_0^\top}]_2, [\underbrace{\mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top}_{\mathbf{k}_1^\top}]_2, \left[\underbrace{\sum_{i\in[L]\setminus\{j\}} ((\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top + \mathbf{h}_{i,j}^\top)\mathbf{a}_{y_i} + \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)\mathbf{K}_{y_i})}_{\mathbf{K}_2}\right]_2, \left[\underbrace{\sum_{i\in[L]\setminus\{j\}} \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)}_{\mathbf{K}_3}\right]_2\right).$$

– $\mathsf{Enc}(\mathsf{mpk}, x, \mathsf{m})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^{1\times k}$ and compute $\mathbf{C}_x$. Output:

$$\mathsf{ct}_x = \left(\underbrace{[\ \mathbf{sA}\ ]_1}_{\mathbf{c}_0}, \underbrace{\left[\sum_{i\in[L]} ((\mathbf{sAV}_i + \mathbf{sT}_i)(\mathbf{a}_{y_i} \otimes \mathbf{I}_{k+1}) + \mathbf{sAW}_i(\mathbf{K}_{y_i} \otimes \mathbf{I}_{k+1})) \right]_1}_{\mathbf{c}_1}, \underbrace{\left[\sum_{i\in[L]} \mathbf{sAW}_i(\mathbf{C}_x \otimes \mathbf{I}_{k+1}) \right]_1}_{\mathbf{c}_2}, \underbrace{[\mathbf{sAk}^\top]_T \cdot \mathsf{m}}_{C} \right).$$

– $\mathsf{Dec}(\mathsf{sk}_{i^*}, \mathsf{hsk}_{i^*}, \mathsf{ct}_x)$: Parse

$$\mathsf{sk}_{i^*} = \mathbf{U}_{i^*}, \quad \mathsf{hsk}_{i^*} = [\mathbf{k}_0^\top, \mathbf{k}_1^\top, \mathbf{K}_2, \mathbf{K}_3]_2, \quad \mathsf{ct}_x = ([\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2]_1, C).$$

Compute $\mathbf{C}_x$ from x and recover

$$\begin{aligned} [\mathbf{z}_1]_T &= e([\mathbf{c}_1 \| \mathbf{c}_2]_1, [\mathbf{I}_{n_k+n_c} \otimes \mathbf{k}_0^\top]_2), & [\mathbf{z}_2]_T &= e([\mathbf{c}_0]_1, [\mathbf{K}_2 \| \mathbf{K}_3 \mathbf{C}_x]_2), \\ [z_3]_T &= e([\mathbf{c}_0 \mathbf{U}_{i^*}]_1, [\mathbf{k}_0^\top]_2), & [z_4]_T &= e([\mathbf{c}_0]_1, [\mathbf{k}_1^\top]_2). \end{aligned}$$

Compute $\mathbf{d}_{x,y_{i^*}}$ from x and y_{i^*} and output

$$z = [(\mathbf{z}_1 - \mathbf{z}_2)\mathbf{d}_{x,y_{i^*}}^\top - z_3 - z_4]_T \cdot C.$$

Completeness. For all $\lambda, L \in \mathbb{N}$, all P, all $i \in [L]$, all $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, P, 1^L)$ and $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{Gen}(\mathsf{crs}, i)$, we have

$$\mathsf{pk}_i = \left([\mathbf{T}_i, \mathbf{Q}_i]_1, \{[\mathbf{H}_{i,j}]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right) = \left([\mathbf{AU}_i, \mathbf{R}_i\mathbf{U}_i]_1, \{[\mathbf{U}_i\mathbf{Br}_j^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right)$$

for some $\mathbf{U}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)}$ and $\pi_i \leftarrow \mathsf{LPrv}(\mathsf{crs}_i, [\mathbf{A}_i\mathbf{U}_i]_1, \mathbf{U}_i)$, where $(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1)$ and $\mathbf{A}_i = \begin{pmatrix} \mathbf{A} \\ \mathbf{R}_i \end{pmatrix}$ with $\mathbf{A} \leftarrow \mathbb{Z}_p^{k\times(2k+1)}$, $\mathbf{R}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+1)}$. Then

– Write $\mathbf{M}_i = \begin{pmatrix} \mathbf{T}_i \\ \mathbf{Q}_i \end{pmatrix} = \begin{pmatrix} \mathbf{AU}_i \\ \mathbf{R}_i\mathbf{U}_i \end{pmatrix}$, we have $\mathsf{LVer}(\mathsf{crs}_i, [\mathbf{M}_i]_1, \pi_i) = 1$ by the perfect completeness of Π (see Section 2.4) and the fact that $\mathbf{M}_i = \mathbf{A}_i\mathbf{U}_i$;
– For each $j \in [L] \setminus \{i\}$, we have $e([\mathbf{A}]_1, [\mathbf{U}_i\mathbf{Br}_j^\top]_2) = e([\mathbf{AU}_i]_1, [\mathbf{Br}_j^\top]_2)$ by the definition of bilinear map e (see Section 2.1) and the fact that $\mathbf{A} \cdot \mathbf{U}_i\mathbf{Br}_j^\top = \mathbf{AU}_i \cdot \mathbf{Br}_j^\top$.

This ensures that $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i) = 1$ by the specification of Ver and readily proves the completeness.

Correctness. For all $\lambda, L \in \mathbb{N}$, all P, all $i^* \in [L]$, all $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, P, 1^L)$, all $(\mathsf{pk}_{i^*}, \mathsf{sk}_{i^*}) \leftarrow \mathsf{Gen}(\mathsf{crs}, i^*)$, all $\{\mathsf{pk}_i\}_{i\in[L]\setminus\{i^*\}}$ such that $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i) = 1$, for all $y_1, \ldots, y_L \in Y$ and $x \in X$ with $P(x, y_{i^*}) = 1$ and all m, we have:

$$\begin{aligned} \mathsf{sk}_{i^*} &= \mathbf{U}_{i^*}, \\ \mathsf{ct}_x &= \left(\underbrace{[\ \mathbf{sA}\ ]_1}_{\mathbf{c}_0}, \underbrace{\left[\sum_{i\in[L]} ((\mathbf{sAV}_i + \mathbf{sT}_i)(\mathbf{a}_{y_i} \otimes \mathbf{I}_{k+1}) + \mathbf{sAW}_i(\mathbf{K}_{y_i} \otimes \mathbf{I}_{k+1})) \right]_1}_{\mathbf{c}_1}, \underbrace{\left[\sum_{i\in[L]} \mathbf{sAW}_i(\mathbf{C}_x \otimes \mathbf{I}_{k+1}) \right]_1}_{\mathbf{c}_2}, \underbrace{[\mathbf{sAk}^\top]_T \cdot \mathsf{m}}_{C} \right) \\ \mathsf{hsk}_{i^*} &= \left(\underbrace{[\ \mathbf{Br}_{i^*}^\top\ ]_2}_{\mathbf{k}_0^\top}, \underbrace{[\mathbf{V}_{i^*}\mathbf{Br}_{i^*}^\top + \mathbf{k}^\top]_2}_{\mathbf{k}_1^\top}, \underbrace{\left[\sum_{i\in[L]\setminus\{i^*\}} ((\mathbf{V}_i\mathbf{Br}_{i^*}^\top + \mathbf{h}_{i,i^*}^\top)\mathbf{a}_{y_i} + \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{Br}_{i^*}^\top)\mathbf{K}_{y_i}) \right]_2}_{\mathbf{K}_2}, \underbrace{\left[\sum_{i\in[L]\setminus\{i^*\}} \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{Br}_{i^*}^\top) \right]_2}_{\mathbf{K}_3} \right) \end{aligned}$$

where

$$\mathbf{Ah}_{i,i^*}^\top = \mathbf{T}_i\mathbf{Br}_{i^*}^\top \quad \forall i \in [L] \setminus \{i^*\} \quad \text{and} \quad \mathbf{AU}_{i^*} = \mathbf{T}_{i^*}. \tag{9}$$

Note that here we actually consider hsk_j for $j = i^*$ and sk_i for $i = i^*$ and all above equalities are ensured by Ver and Gen. Then, as in Section 2.3, let

$$\mathbf{M}_{x,y_i} = \begin{pmatrix} \mathbf{a}_{y_i} & \mathbf{0}_{n_c} \\ \mathbf{K}_{y_i} & \mathbf{C}_x \end{pmatrix}, \quad \forall i \in [L],$$

we have

$$\begin{aligned}
\mathbf{z}_1 &= \sum_{i\in[L]} (\mathbf{sAV}_i + \mathbf{sT}_i \| \mathbf{sAW}_i)(\mathbf{M}_{x,y_i} \otimes \mathbf{I}_{k+1})(\mathbf{I}_{n_k+n_c} \otimes \mathbf{Br}_{i^*}^\top) \\
&= \sum_{i\in[L]} (\mathbf{sAV}_i + \mathbf{sT}_i \| \mathbf{sAW}_i)(\mathbf{I}_{1+n} \otimes \mathbf{Br}_{i^*}^\top)\mathbf{M}_{x,y_i} \qquad (10) \\
&= \sum_{i\in[L]} (\mathbf{sAV}_i\mathbf{Br}_{i^*}^\top + \mathbf{sT}_i\mathbf{Br}_{i^*}^\top \| \mathbf{sAW}_i(\mathbf{I}_n \otimes \mathbf{Br}_{i^*}^\top))\mathbf{M}_{x,y_i} \\
\mathbf{z}_2 &= \sum_{i\in[L]\setminus\{i^*\}} (\mathbf{sAV}_i\mathbf{Br}_{i^*}^\top + \mathbf{sAh}_{i,i^*}^\top \| \mathbf{sAW}_i(\mathbf{I}_n \otimes \mathbf{Br}_{i^*}^\top))\mathbf{M}_{x,y_i} \\
z_3 &= \mathbf{sAU}_{i^*}\mathbf{Br}_{i^*}^\top \\
z_4 &= \mathbf{sAV}_{i^*}\mathbf{Br}_{i^*}^\top + \mathbf{sAk}^\top
\end{aligned}$$

and then

$$\begin{aligned}
z &= [(\mathbf{z}_1 - \mathbf{z}_2)\mathbf{d}_{x,y_{i^*}}^\top - z_3 - z_4]_T \cdot [\mathbf{sAk}^\top]_T \cdot \mathsf{m} \\
&= [(\mathbf{sAV}_{i^*}\mathbf{Br}_{i^*}^\top + \mathbf{sT}_{i^*}\mathbf{Br}_{i^*}^\top \| \mathbf{sAW}_{i^*}(\mathbf{I}_n \otimes \mathbf{Br}_{i^*}^\top))\mathbf{M}_{x,y_{i^*}}\mathbf{d}_{x,y_{i^*}}^\top - \mathbf{sAU}_{i^*}\mathbf{Br}_{i^*}^\top - (\mathbf{sAV}_{i^*}\mathbf{Br}_{i^*}^\top + \mathbf{sAk}^\top)]_T \cdot [\mathbf{sAk}^\top]_T \cdot \mathsf{m} \qquad (11) \\
&= [(\mathbf{sAV}_{i^*}\mathbf{Br}_{i^*}^\top + \mathbf{sT}_{i^*}\mathbf{Br}_{i^*}^\top) - \mathbf{sAU}_{i^*}\mathbf{Br}_{i^*}^\top + \mathbf{sAk}^\top - (\mathbf{sAV}_{i^*}\mathbf{Br}_{i^*}^\top + \mathbf{sAk}^\top)]_T \cdot \mathsf{m} \qquad (12) \\
&= \mathsf{m} \qquad (13)
\end{aligned}$$

Here, equality (10) follows from the property of tensor product: $(\mathbf{M} \otimes \mathbf{I})(\mathbf{I} \otimes \mathbf{a}^\top) = \mathbf{M} \otimes \mathbf{a}^\top = (\mathbf{I} \otimes \mathbf{a}^\top)\mathbf{M}$ for matrices of proper size; equality (11) follows from the fact that $\mathbf{Ah}_{i,i^*}^\top = \mathbf{T}_i\mathbf{Br}_{i^*}^\top$ for all $i \in [L] \setminus \{i^*\}$ (see equality (9)); equality (12) follows from the correctness of predicate encoding; equality (13) follows from the fact that $\mathbf{T}_{i^*} = \mathbf{AU}_{i^*}$ (see equality (9)). This proves the correctness.

Compactness. Assume P has (n, n_c, n_k)-predicate encoding, our slotted Reg-ABE has the following properties:

$$|\mathsf{mpk}| = (n_k + n) \cdot \mathsf{poly}(\lambda) \quad \text{and} \quad |\mathsf{hsk}_j| = (n_k + n) \cdot \mathsf{poly}(\lambda)$$

We also have

$$|\mathsf{crs}| = L^2 \cdot n \cdot \mathsf{poly}(\lambda) \quad \text{and} \quad |\mathsf{ct}| = (n_k + n_c) \cdot \mathsf{poly}(\lambda).$$

Here $\mathsf{crs}_1, \ldots, \mathsf{crs}_L$ contribute $L \cdot \mathsf{poly}(\lambda)$ according to the efficiency of the pairing-based QA-NIZK scheme by Kiltz and Wee [KW15] and the fact that the size of language description is $\mathsf{poly}(\lambda)$.

Security. We have the following theorem. Given pairing-based QA-NIZK in [KW15], our slotted Reg-ABE scheme uses prime-order bilinear group and the security can be reduced to MDDH assumption.

Theorem 1. *Assume* $\Pi = (\mathsf{LGen}, \mathsf{LPrv}, \mathsf{LVer}, \mathsf{LSim})$ *is a QA-NIZK with perfect completeness, perfect zero-knowledge and stronger unbounded simulation soundness for linear space defined in Section 2.4, our slotted Reg-ABE scheme achieves the security defined in Section 2.2 under* MDDH *assumption.*

3.2 Proof

We prove the following technical lemma; this immediately proves Theorem 1.

Lemma 1. *For all adversaries $\mathcal{A}$, there exist adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that:*

$$\mathsf{Adv}^{\text{sReg-ABE}}_{\mathcal{A}}(\lambda) \le L \cdot \mathsf{Adv}^{\text{USS}}_{\mathcal{B}_1}(\lambda) + (2L + 2L \cdot Q + 1) \cdot \mathsf{Adv}^{\text{MDDH}}_{\mathcal{B}_2} + \mathsf{negl}(\lambda)$$

where L is the number of slots, Q is the maximum number of queries on a slot made by $\mathcal{A}$ and $\mathsf{Time}(\mathcal{B}_1), \mathsf{Time}(\mathcal{B}_2) \approx \mathsf{Time}(\mathcal{A})$.

Game Sequence. Let L be the number slots chosen by the adversary, crs be the common reference string, x^* be the challenge "attribute", $(\mathsf{m}^*_0, \mathsf{m}^*_1)$ be challenge message pair, $(\mathsf{pk}^*_i, y^*_i)_{i\in[L]}$ be challenge public keys and challenge "policy" to be registered and ct^* be the challenge ciphertext. For all $i \in [L]$, define $D_i = \{\mathsf{pk}_i : \mathcal{D}_i[\mathsf{pk}_i] = \mathsf{sk}_i \neq \bot\}$ which records responses to $\mathsf{OGen}(i)$ and $C_i = \{\mathsf{pk}_i : (i, \mathsf{pk}_i) \in C\}$ which records public keys in D_i that have been sent to $\mathsf{OCor}(i, \cdot)$. Recall that, we require that, for each $i \in [L]$,

$$\begin{aligned} \mathsf{pk}^*_i \notin D_i &\implies \mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}^*_i) = 1, \\ \mathsf{pk}^*_i \in C_i \vee \mathsf{pk}^*_i \notin D_i &\implies P(x^*, y^*_i) = 0. \end{aligned}$$

Note that pk_i serves as a *general* entry in D_i while pk^*_i is the *specific* challenge public for slot i; there can be more than one assignment for pk_i since the adversary can invoke $\mathsf{OGen}(i)$ for many times. We prove the Lemma 1 via dual-system method using the following game sequence.

- G_0: Real game. Recall that we have
  - crs is in the form:
  $$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1,\ [\mathbf{A}\mathbf{k}^\top]_T,\ \left\{\mathsf{crs}_i, [\mathbf{R}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\right\}_{i\in[L]} \\ \left\{[\mathbf{B}\mathbf{r}^\top_j, \mathbf{V}_j\mathbf{B}\mathbf{r}^\top_j + \mathbf{k}^\top]_2\right\}_{j\in[L]} \\ \left\{[\mathbf{V}_i\mathbf{B}\mathbf{r}^\top_j, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}^\top_j)]_2\right\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}$$
  where $\mathsf{crs}_i \in \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1)$ and $\mathbf{A}_i = \begin{pmatrix}\mathbf{A} \\ \mathbf{R}_i\end{pmatrix}$.
  - For each $i \in [L]$, each $\mathsf{pk}_i \in D_i$ is in the form:
  $$\mathsf{pk}_i = \left([\mathbf{A}\mathbf{U}_i, \mathbf{R}_i\mathbf{U}_i]_1, \{[\mathbf{U}_i\mathbf{B}\mathbf{r}^\top_j]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right)$$
  where $\pi_i \leftarrow \mathsf{LPrv}(\mathsf{crs}_i, [\mathbf{M}_i]_1, \mathbf{U}_i)$ and $\mathbf{M}_i = \begin{pmatrix}\mathbf{A}\mathbf{U}_i \\ \mathbf{R}_i\mathbf{U}_i\end{pmatrix}$; note that $\mathbf{U}_i$ is the corresponding secret key sk_i.
  - For all $i \in [L]$, pk^*_i is in the form:
  $$\mathsf{pk}^*_i = ([\mathbf{T}^*_i, \mathbf{Q}^*_i]_1, \{[\mathbf{h}^{*\top}_{i,j}]_2\}_{j\in[L]\setminus\{i\}}, \pi^*_i)$$
  such that $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}^*_i) = 1$ which means $\mathsf{LVer}\left(\mathsf{crs}_i, \left[\begin{smallmatrix}\mathbf{T}^*_i \\ \mathbf{Q}^*_i\end{smallmatrix}\right]_1, \pi^*_i\right) = 1$ and $\mathbf{A}\mathbf{h}^{*\top}_{i,j} = \mathbf{T}^*_i\mathbf{B}\mathbf{r}^\top_j$ for each $j \in [L] \setminus \{i\}$.
  - ct^* for x^* and $(\mathsf{m}^*_0, \mathsf{m}^*_1)$ is in the form:
  $$\mathsf{ct}^* = \left([\underbrace{\mathbf{s}\mathbf{A}}_{\mathbf{c}^*_0}]_1, \left[\underbrace{\sum_{i\in[L]} ((\mathbf{s}\mathbf{A}\mathbf{V}_i + \mathbf{s}\mathbf{T}^*_i)(\mathbf{a}_{y^*_i} \otimes \mathbf{I}_{k+1}) + \mathbf{s}\mathbf{A}\mathbf{W}_i(\mathbf{K}_{y^*_i} \otimes \mathbf{I}_{k+1}))}_{\mathbf{c}^*_1}\right]_1, \left[\underbrace{\sum_{i\in[L]} \mathbf{s}\mathbf{A}\mathbf{W}_i(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1})}_{\mathbf{c}^*_2}\right]_1, \underbrace{[\mathbf{s}\mathbf{A}\mathbf{k}^\top]_T \cdot \mathsf{m}^*_b}_{C^*}\right)$$
  where $b \leftarrow \{0, 1\}$ is the secret bit.
- G_1: Identical to G_0 except that, for all $i \in [L]$ and all $\mathsf{pk}_i \in D_i$, we replace π_i in pk_i with
$$\widetilde{\pi}_i \leftarrow \boxed{\mathsf{LSim}}(\mathsf{crs}_i, \mathsf{td}_i, [\mathbf{M}_i]_1) \quad \text{where} \quad \mathbf{M}_i = \begin{pmatrix}\mathbf{A}\mathbf{U}_i \\ \mathbf{R}_i\mathbf{U}_i\end{pmatrix}.$$
We have $\mathsf{G}_1 \equiv \mathsf{G}_0$. This follows from the perfect zero-knowledge of Π. See Lemma 3 for more details.

– G_2: Identical to G_1 except that we sample $\mathbf{s} \leftarrow \mathbb{Z}_p^{1\times k}$ along with $\mathbf{A}$ and replace all $\mathbf{R}_i$ in crs with

$$\widehat{\mathbf{R}}_i = \widetilde{\mathbf{R}}_i \begin{pmatrix} \mathbf{sA} \\ \mathbf{I}_{2k+1} \end{pmatrix}, \quad \widetilde{\mathbf{R}}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)}$$

We have $\mathsf{G}_2 \approx_s \mathsf{G}_1$. This follows from the fact that both $\mathbf{R}_i$ (in G_1) and $\widehat{\mathbf{R}}_i$ (in G_2) are truly random since matrix $\begin{pmatrix} \mathbf{sA} \\ \mathbf{I}_{2k+1} \end{pmatrix}$ is full rank. See Lemma 4 for more details.

– G_3: Identical to G_2 except that we replace $\mathbf{sT}_i^*$ with $\mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*$ in $\mathbf{c}_1^*$; namely, we have

$$\mathbf{c}_1^* = \sum_{i\in[L]} ((\mathbf{sAV}_i + \boxed{\mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*})(\mathbf{a}_{y_i^*} \otimes \mathbf{I}_{k+1}) + \mathbf{sAW}_i(\mathbf{K}_{y_i^*} \otimes \mathbf{I}_{k+1})).$$

We have $\mathsf{G}_3 \approx_c \mathsf{G}_2$. This follows from stronger unbounded simulation soundness of Π along with the fact that $\mathsf{LVer}(\mathsf{crs}_i, [\mathbf{M}_i^*], \pi_i^*) = 1$ for all $i \in [L]$ where $\mathbf{M}_i^* = \begin{pmatrix} \mathbf{T}_i^* \\ \mathbf{Q}_i^* \end{pmatrix}$. Assume $\mathsf{pk}_{i^*}^* \notin D_{i^*}$, i.e., $\mathsf{pk}_{i^*}^*$ is malicious. In the reduction, we guess $i^* \leftarrow [L]$ and obtain $\mathbf{A}, \widehat{\mathbf{R}}_{i^*}, \mathsf{crs}_{i^*}$ as input; we simulate honestly as in G_3 except that for all $\mathsf{pk}_{i^*} \in D_{i^*}$, we make an oracle query $[\mathbf{M}_{i^*}]_1$ and get $\widetilde{\pi}_{i^*}$ in it; we finally output $([\mathbf{M}_{i^*}^*]_1, \pi_{i^*}^*)$ in $\mathsf{pk}_{i^*}^* \notin D_{i^*}$. Observe that once it happens that $\mathbf{e}_1\widetilde{\mathbf{R}}_{i^*}^{-1}\mathbf{Q}_{i^*}^* \neq \mathbf{sT}_{i^*}^*$, we must have $\mathbf{M}_{i^*}^* \notin \mathsf{span}(\mathbf{A}_{i^*})$. When $\mathsf{pk}_{i^*}^* \in D_{i^*}$, we always have $\mathsf{G}_3 \equiv \mathsf{G}_2$. See Lemma 5 for more details.

– G_4: Identical to G_3 except that we replace all $\mathbf{sA}$ with $\mathbf{c} \leftarrow \mathbb{Z}_p^{1\times(2k+1)}$; in particular, we generate all $\widehat{\mathbf{R}}_i$ as follows:

$$\widehat{\mathbf{R}}_i = \widetilde{\mathbf{R}}_i \begin{pmatrix} \boxed{\mathbf{c}} \\ \mathbf{I}_{2k+1} \end{pmatrix}, \quad \widetilde{\mathbf{R}}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)}$$

and generate the challenge ciphertext as follows:

$$\mathsf{ct}^* = \Bigg(\underbrace{[\boxed{\mathbf{c}}]_1}_{\mathbf{c}_0^*}, \underbrace{\left[\sum_{i\in[L]} ((\boxed{\mathbf{c}}\mathbf{V}_i + \mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*)(\mathbf{a}_{y_i^*} \otimes \mathbf{I}_{k+1}) + \boxed{\mathbf{c}}\mathbf{W}_i(\mathbf{K}_{y_i^*} \otimes \mathbf{I}_{k+1})) \right]_1}_{\mathbf{c}_1^*}, \underbrace{\left[\sum_{i\in[L]} \boxed{\mathbf{c}}\mathbf{W}_i(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) \right]_1}_{\mathbf{c}_2^*}, \underbrace{[\boxed{\mathbf{c}}\mathbf{k}^\top]_T \cdot \mathsf{m}_b^*}_{C^*} \Bigg).$$

We have $\mathsf{G}_4 \approx_c \mathsf{G}_3$. This follows from MDDH assumption which ensures that $([\mathbf{A}]_1, [\mathbf{sA}]_1) \approx_c ([\mathbf{A}]_1, [\mathbf{c}]_1)$ when $\mathbf{A} \leftarrow \mathbb{Z}_p^{k\times(2k+1)}, \mathbf{s} \leftarrow \mathbb{Z}_p^{1\times k}$ and $\mathbf{c} \leftarrow \mathbb{Z}_p^{1\times(2k+1)}$. This is analogous to the transition from normal ciphertext to semi-functional ciphertext in the dual-system method [Wat09]. See Lemma 6 for more details.

– $\mathsf{G}_{5,\ell}$, $(\ell \in [0, L])$: Identical to G_4 except we change $[\mathbf{V}_j\mathbf{Br}_j^\top + \mathbf{k}^\top]_2$ for all $j \in [\ell]$ as follows:

$$[\mathbf{V}_j\mathbf{Br}_j^\top + \mathbf{k}^\top + \boxed{\mathbf{c}^\perp\alpha}]_2$$

where $\mathbf{c}^\perp \in \mathbb{Z}_p^{2k+1}$ such that $\mathbf{cc}^\perp = 1$ and $\mathbf{Ac}^\perp = \mathbf{0}$ and $\alpha \leftarrow \mathbb{Z}_p$. We have that

- $\mathsf{G}_{5,0} = \mathsf{G}_4$; the two games are exactly identical, since $[0] = \emptyset$;
- $\mathsf{G}_{5,\ell} \approx_c \mathsf{G}_{5,\ell-1}$ for all $\ell \in [L]$; this is analogous to the transition from normal keys to semi-functional keys one-by-one in the dual-system method. However, the situation is much more complicated in the context of Reg-ABE, we will describe the sub-sequence of games for this step later in Section 3.3.

– G_6: Identical to $\mathsf{G}_{5,L}$ except that we replace term C^* in ct^* as $\boxed{C^* \leftarrow \mathbb{G}_T}$. We have $\mathsf{G}_6 \equiv \mathsf{G}_{5,L}$. This follows from the following statistical argument:

$$(\overbrace{\mathbf{Ak}^\top, \mathbf{k}^\top + \mathbf{c}^\perp\alpha}^{\mathsf{crs}}, \overbrace{\mathbf{ck}^\top}^{C^* \text{ in } \mathsf{ct}^*}) \equiv (\mathbf{Ak}^\top, \mathbf{k}^\top, \mathbf{ck}^\top - \alpha)$$

when $\mathbf{k} \leftarrow \mathbb{Z}_p^{1\times(2k+1)}$ and the fact that $[\alpha]_T$ only appears in C^*. We can prove the statement via change of variable $\mathbf{k}^\top \mapsto \mathbf{k}^\top - \mathbf{c}^\perp\alpha$. See Lemma 7 for more details.

Observe that, in G_6, the challenge ciphertext ct^* is independent of b and the adversary's advantage is exactly 0.

3.3 From $\mathsf{G}_{5,\ell-1}$ to $\mathsf{G}_{5,\ell}$

We prove $\mathsf{G}_{5,\ell-1} \approx_c \mathsf{G}_{5,\ell}$ which completes our proof of Lemma 1. For this, we need the following sub-sequence of games for each $\ell \in [L]$:

– $\mathsf{G}_{5,\ell-1,0}$: Identical to $\mathsf{G}_{5,\ell-1}$. We recall crs and $\mathsf{pk}_i \in D_i$ in the following form, where we highlight $\mathbf{r}_\ell$-related terms using dashed boxes which will be changed in this sub-sequence.

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1, [\mathbf{A}\mathbf{k}^\top]_T, \{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\}_{i\in[L]} \\ \{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2\}_{j\in[\ell-1]}, \boxed{[\mathbf{B}\mathbf{r}_\ell^\top, \mathbf{V}_\ell\mathbf{B}\mathbf{r}_\ell^\top + \mathbf{k}^\top]_2}, \{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\}_{j\in[L]\setminus[\ell]} \\ \{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\}_{j\in[L]\setminus\{\ell\}, i\in[L]\setminus\{j\}}, \boxed{\{[\mathbf{V}_i\mathbf{B}\mathbf{r}_\ell^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_\ell^\top)]_2\}_{i\in[L]\setminus\{\ell\}}} \end{pmatrix},$$

$$\mathsf{pk}_i = \begin{cases} ([\mathbf{A}\mathbf{U}_i, \widehat{\mathbf{R}}_i\mathbf{U}_i]_1, \{[\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{i,\ell\}}, \boxed{[\mathbf{U}_i\mathbf{B}\mathbf{r}_\ell^\top]_2}, \widetilde{\pi}_i) & \text{if } i \neq \ell \\ ([\mathbf{A}\mathbf{U}_\ell, \widehat{\mathbf{R}}_\ell\mathbf{U}_\ell]_1, \{[\mathbf{U}_\ell\mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{\ell\}}, \widetilde{\pi}_\ell) & \text{if } i = \ell \end{cases}$$

Clearly, we have $\mathsf{G}_{5,\ell-1,0} = \mathsf{G}_{5,\ell-1}$; all changes are conceptual.

– $\mathsf{G}_{5,\ell-1,1}$: Identical to $\mathsf{G}_{5,\ell-1,0}$ except that we replace all $\mathbf{B}\mathbf{r}_\ell^\top$ with $\mathbf{d}_\ell^\top \leftarrow \mathbb{Z}_p^{k+1}$ in crs; in particular, we change the dashed boxed term as follows:

$$[\boxed{\mathbf{d}_\ell^\top}, \mathbf{V}_\ell\boxed{\mathbf{d}_\ell^\top} + \mathbf{k}^\top]_2, \{[\mathbf{V}_i\boxed{\mathbf{d}_\ell^\top}, \mathbf{W}_i(\mathbf{I}_n \otimes \boxed{\mathbf{d}_\ell^\top})]_2, [\mathbf{U}_i\boxed{\mathbf{d}_\ell^\top}]_2\}_{i\in[L]\setminus\{\ell\}}$$

We have $\mathsf{G}_{5,\ell-1,1} \approx_c \mathsf{G}_{5,\ell-1,0}$. This follows from MDDH assumption w.r.t. $[\mathbf{B}]_2$ which ensures that $([\mathbf{B}]_2, [\mathbf{B}\mathbf{r}_\ell^\top]_2) \approx_c ([\mathbf{B}]_2, [\mathbf{d}_\ell^\top]_2)$ when $\mathbf{B} \leftarrow \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{r}_\ell \leftarrow \mathbb{Z}_p^{1\times k}$, $\mathbf{d}_\ell \leftarrow \mathbb{Z}_p^{1\times(k+1)}$. See Lemma 8 for more details.

– $\mathsf{G}_{5,\ell-1,2}$: Identical to $\mathsf{G}_{5,\ell-1,1}$ except that we change the dashed boxed terms as follows:

$$[\mathbf{d}_\ell^\top, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{k}^\top + \boxed{\mathbf{c}^\perp\alpha}]_2, \{[\mathbf{V}_i\mathbf{d}_\ell^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{d}_\ell^\top)]_2, [\mathbf{U}_i\mathbf{d}_\ell^\top]_2\}_{i\in[L]\setminus\{\ell\}}$$

We have $\mathsf{G}_{5,\ell-1,2} \approx_c \mathsf{G}_{5,\ell-1,1}$. We provide an overview of the proof in Section 3.4.

– $\mathsf{G}_{5,\ell-1,3}$: Identical to $\mathsf{G}_{5,\ell-1,2}$ except that we replace all $\mathbf{d}_\ell^\top$ with $\mathbf{B}\mathbf{r}_\ell^\top$ where $\mathbf{r}_\ell^\top \leftarrow \mathbb{Z}_p^k$ in crs; in particular, we generate crs as follow:

$$[\boxed{\mathbf{B}\mathbf{r}_\ell^\top}, \mathbf{V}_\ell\boxed{\mathbf{B}\mathbf{r}_\ell^\top} + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2, \{[\mathbf{V}_i\boxed{\mathbf{B}\mathbf{r}_\ell^\top}, \mathbf{W}_i(\mathbf{I}_n \otimes \boxed{\mathbf{B}\mathbf{r}_\ell^\top})]_2, [\mathbf{U}_i\boxed{\mathbf{B}\mathbf{r}_\ell^\top}]_2\}_{i\in[L]\setminus\{\ell\}}$$

We have $\mathsf{G}_{5,\ell-1,3} \approx_c \mathsf{G}_{5,\ell-1,2}$. Analogous to $\mathsf{G}_{5,\ell-1,1} \approx_c \mathsf{G}_{5,\ell-1,0}$, it follows from MDDH assumption w.r.t. $[\mathbf{B}]_2$ which ensures that $([\mathbf{B}]_2, [\mathbf{B}\mathbf{r}_\ell^\top]_2) \approx_c ([\mathbf{B}]_2, [\mathbf{d}_\ell^\top]_2)$ when $\mathbf{B} \leftarrow \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{r}_\ell \leftarrow \mathbb{Z}_p^{1\times k}$, $\mathbf{d}_\ell \leftarrow \mathbb{Z}_p^{1\times(k+1)}$. See Lemma 10 for more details.

Observe that, we have $\mathsf{G}_{5,\ell-1,3} = \mathsf{G}_{5,\ell}$ and this proves $\mathsf{G}_{5,\ell-1} \approx_c \mathsf{G}_{5,\ell}$.

3.4 From $\mathsf{G}_{5,\ell-1,1}$ to $\mathsf{G}_{5,\ell-1,2}$

We review $\mathsf{G}_{5,\ell-1,1}$ and $\mathsf{G}_{5,\ell-1,2}$ in the following form. Here we use solid boxes to indicate the difference between two games and use dashed boxes to highlight those terms that are relevant to our proof.

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1, [\mathbf{A}\mathbf{k}^\top]_T, \{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\}_{i\in[L]} \\ \{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2\}_{j\in[\ell-1]}, \boxed{[\mathbf{d}_\ell^\top, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{k}^\top + \boxed{\mathbf{c}^\perp\alpha}]_2}, \{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\}_{j\in[L]\setminus[\ell]} \\ \{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\}_{j\in[L]\setminus\{\ell\}, i\in[L]\setminus\{j\}}, \boxed{\{[\mathbf{V}_i\mathbf{d}_\ell^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{d}_\ell^\top)]_2\}_{i\in[L]\setminus\{\ell\}}} \end{pmatrix},$$

$$\mathsf{pk}_i = \begin{cases} ([\mathbf{A}\mathbf{U}_i, \widehat{\mathbf{R}}_i\mathbf{U}_i]_1, \{[\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{i,\ell\}}, \boxed{[\mathbf{U}_i\mathbf{d}_\ell^\top]_2}, \widetilde{\pi}_i) & \text{if } i \neq \ell \\ ([\mathbf{A}\mathbf{U}_\ell, \widehat{\mathbf{R}}_\ell\mathbf{U}_\ell]_1, \{[\mathbf{U}_\ell\mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{\ell\}}, \widetilde{\pi}_\ell) & \text{if } i = \ell \end{cases}$$

$$\mathbf{c}_1^* = \boxed{(\mathbf{c}\mathbf{V}_\ell + \mathbf{e}_1\widetilde{\mathbf{R}}_\ell^{-1}\mathbf{Q}_\ell^*)(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1})} + \sum_{i\in[L]\setminus\{\ell\}} ((\mathbf{c}\mathbf{V}_i + \mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*)(\mathbf{a}_{y_i^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_i(\mathbf{K}_{y_i^*} \otimes \mathbf{I}_{k+1}))$$

$$\mathbf{c}_2^* = \boxed{\mathbf{cW}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1})} + \sum_{i \in [L] \setminus \{\ell\}} \mathbf{cW}_i(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1})$$

we define $\mathbf{c}^\perp \in \mathbb{Z}_p^{2k+1}$ and $\mathbf{d}^\perp \in \mathbb{Z}_p^{1\times(k+1)}$ such that $\mathbf{Ac}^\perp = \mathbf{0}$, $\mathbf{cc}^\perp = 1$, $\mathbf{d}^\perp\mathbf{B} = \mathbf{0}$ and $\mathbf{d}^\perp\mathbf{d}_\ell = 1$. We will proof $\mathsf{G}_{5,\ell-1,2} \approx_c \mathsf{G}_{5,\ell-1,1}$ by considering two cases: (1) pk_ℓ^* is honest; (2) pk_ℓ^* is corrupted or maliciously generated by the adversary.

Useful Lemma. Before we proceed, we prepare the following lemma.

Lemma 2. *For all $\mathbf{B} \leftarrow \mathbb{Z}_p^{(k+1)\times k}$ and $\mathbf{d}^\perp \leftarrow \mathbb{Z}_p^{1\times(k+1)}$ such that $\mathbf{d}^\perp\mathbf{B} = 0$. For any adversary $\mathcal{A}$, there exist an adversary $\mathcal{B}_2$ such that*

$$\begin{aligned} &\big|\Pr[\mathcal{A}(\mathbf{M}, [\mathbf{R}]_1, \mathbf{B}, \mathbf{d}^\perp, \mathbf{MU}, [\mathbf{RU}]_1, \quad \mathbf{UB}) = 1] - \\ &\quad \Pr[\mathcal{A}(\mathbf{M}, [\mathbf{R}]_1, \mathbf{B}, \mathbf{d}^\perp, \mathbf{MU}, [\mathbf{RU} + \boxed{\mathbf{u}^\top\mathbf{d}^\perp}]_1, \mathbf{UB}) = 1]\big| \\ \leq\ & 2 \cdot \mathsf{Adv}^{\mathrm{MDDH}}_{\mathcal{B}_2} + \mathsf{negl}(\lambda) \end{aligned}$$

where $\mathbf{M} \leftarrow \mathbb{Z}_p^{(k+1)\times(2k+1)}$, $\mathbf{R} \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+1)}$, $\mathbf{U} \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)}$ and $\mathbf{u} \leftarrow \mathbb{Z}_p^{1\times(2k+2)}$.

Before proving the lemma, we give some intuition by investigating a simplified version without $\mathbf{B}$ and $\mathbf{d}$:

$$\mathbf{M}, [\mathbf{R}]_1, \mathbf{MU}, [\mathbf{RU}]_1 \approx_c \mathbf{M}, [\mathbf{R}]_1, \mathbf{MU}, [\widehat{\mathbf{U}}]_1$$

where $\mathbf{M}, \mathbf{R}, \mathbf{U}$ are defined as before and $\widehat{\mathbf{U}} \leftarrow \mathbb{Z}_p^{(2k+2)\times(k+1)}$. If we encode $\mathbf{M}$ and $\mathbf{MU}$ over $\mathbb{G}_1$, this is simply MDDH assumption and there is nothing special. The main point here is that we give out $\mathbf{M}$ directly to the adversary. This allows it to get the kernel space of $\mathbf{M}$ which is crucial for its future application. Looking ahead, we will set $\mathbf{M} = \binom{\mathbf{A}}{\mathbf{c}}$ and want to know/simulate $\mathbf{c}^\perp$. However, this hurts the indistinguishability; the adversary can recover $\mathbf{U}$ and check whether the last term is truly random. At this point the shape of $\mathbf{M}$ saves us. Note that $\mathbf{M}$ is a wide matrix rather than a square one. The main idea behind the proof is that given $\mathbf{M}, \mathbf{MU}$, there is still some entropy left inside $[\mathbf{RU}]_1$ so that we can argue its pseudorandomness even given $[\mathbf{R}]_1$ as MDDH. A detailed proof of the lemma is as follows.

Proof. We prove the lemma with the following argument:

$$\begin{aligned} &\mathbf{M}, [\mathbf{R}]_1, && \mathbf{B}, \mathbf{d}^\perp, \mathbf{MU}, [\mathbf{RU}]_1, && \mathbf{UB} && \\ \approx_c\ &\mathbf{M}, [\boxed{\widetilde{\mathbf{R}}\mathbf{D}}]_1, && \mathbf{B}, \mathbf{d}^\perp, \mathbf{MU}, [\boxed{\widetilde{\mathbf{R}}\mathbf{D}}\mathbf{U}]_1, && \mathbf{UB} && \text{// MDDH} \\ \approx_s\ &\mathbf{M}, [\widetilde{\mathbf{R}}\mathbf{D}]_1, && \mathbf{B}, \mathbf{d}^\perp, \mathbf{MU}, [\widetilde{\mathbf{R}}\mathbf{D}\mathbf{U} + \boxed{\widetilde{\mathbf{R}}\widetilde{\mathbf{u}}^\top\mathbf{d}^\perp}]_1, && \mathbf{UB} && \text{// change of variable} \\ \approx_c\ &\mathbf{M}, [\boxed{\mathbf{R}}]_1, && \mathbf{B}, \mathbf{d}^\perp, \mathbf{MU}, [\boxed{\mathbf{R}}\mathbf{U} + \boxed{\mathbf{u}^\top}\mathbf{d}^\perp]_1, && \mathbf{UB} && \text{// MDDH} \end{aligned}$$

where $\widetilde{\mathbf{R}} \leftarrow \mathbb{Z}_p^{(2k+2)\times k}$, $\mathbf{D} \leftarrow \mathbb{Z}_p^{k\times(2k+1)}$ and $\widetilde{\mathbf{u}} \leftarrow \mathbb{Z}_p^{1\times k}$. We justify each step as follows: The first $\approx_c$ follows from MDDH assumption w.r.t. $[\widetilde{\mathbf{R}}]_1$ which ensures that $[\mathbf{R}]_1 \approx_c [\widetilde{\mathbf{R}}\mathbf{D}]_1$. The second $\approx_s$ follows from change of variable

$$\mathbf{U} \mapsto \mathbf{U} + \mathbf{D}^\perp\widetilde{\mathbf{u}}^\top\mathbf{d}^\perp$$

where $\widetilde{\mathbf{u}} \leftarrow \mathbb{Z}_p^{1\times k}$ and $\mathbf{D}^\perp \in \mathbb{Z}_p^{(2k+1)\times k}$ such that $\mathbf{DD}^\perp = \mathbf{I}$ and $\mathbf{MD}^\perp = \mathbf{0}$; this uses the fact that $\binom{\mathbf{M}}{\mathbf{D}}$ has full rank w.h.p. The third $\approx_c$ follows from MDDH assumption w.r.t. $[\widetilde{\mathbf{R}}]_1$ which ensures that $[\widetilde{\mathbf{R}}, \widetilde{\mathbf{R}}(\mathbf{D}\|\widetilde{\mathbf{u}}^\top)]_1 \approx_c [\widetilde{\mathbf{R}}, (\mathbf{R}\|\mathbf{u}^\top)]_1$. This readily proves the lemma. □

Honest Case. In this case, we have $\mathsf{pk}_\ell^* = ([\mathbf{T}_\ell^*, \mathbf{Q}_\ell^*]_1, \{[\mathbf{h}_{\ell,j}^{*\top}]_2\}_{j\in[L]\setminus\{\ell\}}, \pi_\ell^*) \in D_\ell \setminus C_\ell$. Namely, we know $\mathbf{U}_\ell^*$ (such that $\mathbf{T}_\ell^* = \mathbf{AU}_\ell^*$ and $\mathbf{Q}_\ell^* = \widehat{\mathbf{R}}_\ell\mathbf{U}_\ell^*$) and $\mathbf{U}_\ell^*$ is hidden from the adversary. We can write the dashboxed terms in $\mathbf{c}_1^*$ as follows:

$$(\mathbf{cV}_\ell + \boxed{\mathbf{cU}_\ell^*})(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{cW}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1})$$

and replace $\widehat{\mathbf{R}}_\ell$ in crs with a random $\mathbf{R}_\ell$ as in G_1. We prove $\mathsf{G}_{5,\ell-1,2} \approx_c \mathsf{G}_{5,\ell-1,1}$ in this case using the following argument for all $b \in \{0,1\}$:

$$
\begin{aligned}
& \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha && \text{//crs, } \mathsf{pk}_\ell \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}_\ell^*; \mathbf{A}\mathbf{U}_\ell^*, [\mathbf{R}_\ell\mathbf{U}_\ell^*]_1, \mathbf{U}_\ell^*\mathbf{B} && \text{//ct}^*, \text{ } \mathsf{pk}_\ell^* \\
\approx_c\ & \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha && \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}_\ell^*; \mathbf{A}\mathbf{U}_\ell^*, [\mathbf{R}_\ell\mathbf{U}_\ell^* + \boxed{\widehat{\mathbf{u}}^\top\mathbf{d}^\perp}]_1, \mathbf{U}_\ell^*\mathbf{B} && \text{// Lemma 2} \\
\approx_s\ & \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \boxed{\mathbf{c}^\perp v_\ell} + b\mathbf{c}^\perp\alpha && \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}_\ell^* + \boxed{v_\ell\mathbf{d}^\perp + u_\ell\mathbf{d}^\perp}; \mathbf{A}\mathbf{U}_\ell^*, [\mathbf{R}_\ell\mathbf{U}_\ell^* + \boxed{\mathbf{R}_\ell\mathbf{c}^\perp u_\ell\mathbf{d}^\perp} + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1, \mathbf{U}_\ell^*\mathbf{B} && \text{// change of variable} \\
\approx_s\ & \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp v_\ell + \cancel{b\mathbf{c}^\perp\alpha} && \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}_\ell^* + v_\ell\mathbf{d}^\perp + u_\ell\mathbf{d}^\perp; \mathbf{A}\mathbf{U}_\ell^*, [\mathbf{R}_\ell\mathbf{U}_\ell^* + \mathbf{R}_\ell\mathbf{c}^\perp u_\ell\mathbf{d}^\perp + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1, \mathbf{U}_\ell^*\mathbf{B} && \text{// statistic}
\end{aligned}
$$

where $\widehat{\mathbf{u}} \leftarrow \mathbb{Z}_p^{1\times(2k+2)}$ and $v_\ell, u_\ell \leftarrow \mathbb{Z}_p$. We justify each step as below: The first $\approx_c$ uses Lemma 2 with $\mathbf{M} = \left(\begin{smallmatrix}\mathbf{A}\\ \mathbf{c}\end{smallmatrix}\right)$, $\mathbf{R} = \mathbf{R}_\ell$, $\mathbf{U} = \mathbf{U}_\ell^*$ and $\mathbf{u} = \widehat{\mathbf{u}}$; in the reduction, we sample $\mathbf{V}_\ell$, α and $\mathbf{c}^\perp$. The second $\approx_s$ uses change of variables

$$\mathbf{V}_\ell \mapsto \mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp \quad \text{and} \quad \mathbf{U}_\ell^* \mapsto \mathbf{U}_\ell^* + \mathbf{c}^\perp u_\ell \mathbf{d}^\perp.$$

The last $\approx_s$ is straight-forward with the observation that $\widehat{\mathbf{u}}^\top$ hides $\mathbf{R}_\ell\mathbf{c}^\perp u_\ell$. See a more detailed proof in Lemma 9.

Corrupted & Malicious Case. In this case, we have $\mathsf{pk}_\ell^* = ([\mathbf{T}_\ell^*, \mathbf{Q}_\ell^*]_1, \{[\mathbf{h}_{\ell,j}^{*\top}]_2\}_{j\in[L]\setminus\{\ell\}}, \pi_\ell^*) \in C_\ell \cup \overline{D}_\ell$. It is required that $P(x^*, y_\ell^*) = 0$. We prove $\mathsf{G}_{5,\ell-1,2} \approx_s \mathsf{G}_{5,\ell-1,1}$ in this case using the following argument for all $b \in \{0,1\}$:

$$
\begin{aligned}
& \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{A}\mathbf{W}_\ell, \mathbf{W}_\ell(\mathbf{I}_n\otimes\mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha && \text{//crs} \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell(\mathbf{a}_{y_\ell^*}\otimes\mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*}\otimes\mathbf{I}_{k+1}), \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*}\otimes\mathbf{I}_{k+1}) && \text{//ct}^* \\
\approx_s\ & \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{A}\mathbf{W}_\ell, \mathbf{W}_\ell(\mathbf{I}_n\otimes\mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + \boxed{\mathbf{c}^\perp v_\ell} + b\mathbf{c}^\perp\alpha && \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell(\mathbf{a}_{y_\ell^*}\otimes\mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*}\otimes\mathbf{I}_{k+1}) + \boxed{v_\ell\mathbf{a}_{y_\ell^*}\otimes\mathbf{d}^\perp + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*}\otimes\mathbf{d}^\perp}, \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*}\otimes\mathbf{I}_{k+1}) + \boxed{\mathbf{w}_\ell\mathbf{C}_{x^*}\otimes\mathbf{d}^\perp} && \\
\approx_s\ & \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{A}\mathbf{W}_\ell, \mathbf{W}_\ell(\mathbf{I}_n\otimes\mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp v_\ell + b\mathbf{c}^\perp\alpha && \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell(\mathbf{a}_{y_\ell^*}\otimes\mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*}\otimes\mathbf{I}_{k+1}) + \cancel{v_\ell\mathbf{a}_{y_\ell^*}\otimes\mathbf{d}^\perp} + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*}\otimes\mathbf{d}^\perp, \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*}\otimes\mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{C}_{x^*}\otimes\mathbf{d}^\perp && \\
\approx_s\ & \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{A}\mathbf{W}_\ell, \mathbf{W}_\ell(\mathbf{I}_n\otimes\mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp v_\ell + \cancel{b\mathbf{c}^\perp\alpha} && \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell(\mathbf{a}_{y_\ell^*}\otimes\mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*}\otimes\mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*}\otimes\mathbf{d}^\perp, \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*}\otimes\mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{C}_{x^*}\otimes\mathbf{d}^\perp &&
\end{aligned}
$$

where $v_\ell \leftarrow \mathbb{Z}_p$ and $\mathbf{w}_\ell \leftarrow \mathbb{Z}_p^n$. We justify each step as follows: The first $\approx_s$ uses the change of variables:

$$\mathbf{V}_\ell \mapsto \mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp \quad \text{and} \quad \mathbf{W}_\ell \mapsto \mathbf{W}_\ell + \mathbf{c}^\perp(\mathbf{w}_\ell\otimes\mathbf{d}^\perp)$$

The second $\approx_s$ uses the fact that $P(x^*, y_\ell^*) = 0$ and the security of predicate encoding defined in Section 2.3. The last $\approx_s$ is straight-forward. See a more detailed proof in Lemma 9.

4 Concrete Slotted Reg-ABE

This section presents our concrete slotted Reg-ABE for arithmetic branching programs (ABP), derived from the generic scheme in Section 3. We use the predicate encoding of arithmetic span programs (ASP) [CGW15, Appendix A] which captures ABP [IW14]. As mentioned before, we employ the pairing-based QA-NIZK scheme by Kiltz and Wee, see Appendix B. Our concrete slotted Reg-ABE for span program and zero inner-product predicate and slotted RBE are deferred to Appendix D.

Preliminaries. An Arithmetic Span Program [IW14], denoted by V, is defined by $(\mathbf{Y}, \mathbf{Z}) \in \mathbb{Z}_p^{m\times\ell} \times \mathbb{Z}_p^{m\times\ell}$ where

$$V(\mathbf{x}) = 1 \iff \mathbf{x} \in \mathbb{Z}_p^{1\times m} \text{ satisfies } V \iff \exists\, \boldsymbol{\omega} \in \mathbb{Z}_p^{1\times m} \text{ such that } \mathbf{e}_1 = \boldsymbol{\omega}(\mathsf{diag}(\mathbf{x}) \cdot \mathbf{Y} + \mathbf{Z}).$$

Here we use notation: $\mathsf{diag}(\mathbf{x}) := \begin{pmatrix} x_1 & & \\ & \ddots & \\ & & x_m \end{pmatrix} \in \mathbb{Z}_p^{m\times m}$ for $\mathbf{x} = (x_1, \ldots, x_m)$ and note that $\mathsf{diag}(\mathbf{x}) = \mathsf{diag}(\mathbf{x})^\top$. We review the predicate encoding for ASP predicate (ciphertext-policy variant):

$$P(V, \mathbf{x}) = 1 \Longleftrightarrow V(\mathbf{x}) = 1$$

as follows [CGW15, Appendix A.6]: let $n = 2m + \ell$, $n_c = 2m$ and $n_k = m + 1$, define

$$\mathbf{C}_{\mathbf{Y},\mathbf{Z}} = \begin{pmatrix} \mathbf{I}_m & \mathbf{0}_{m\times m} \\ \mathbf{0}_{m\times m} & \mathbf{I}_m \\ \mathbf{Y}^\top & \mathbf{Z}^\top \end{pmatrix}, \quad \mathbf{K}_{\mathbf{x}} = \begin{pmatrix} \mathbf{0}_m^\top & \mathsf{diag}(\mathbf{x}) \\ \mathbf{0}_m^\top & \mathbf{I}_m \\ \mathbf{e}_1^\top & \mathbf{0}_{\ell\times m} \end{pmatrix}, \quad \mathbf{a}_{\mathbf{x}} = (1 \| \mathbf{0}_m), \quad \mathbf{d}_{\mathbf{x},\mathbf{Y},\mathbf{Z}} = (1 \| \boldsymbol{\omega} \| -\boldsymbol{\omega} \cdot \mathsf{diag}(\mathbf{x}) \| -\boldsymbol{\omega})$$

where $\mathbf{0}_m$ is a *row* zero vector of size m. Note that we work with *read-once* ASP as in [CGW15].

Scheme. Our concrete slotted Registered CP-ABE for read-once ASP from SXDH (1-Lin) assumption works as follows:

– $\mathsf{Setup}(1^\lambda, P, 1^L)$: Run $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$. Sample

$$\mathbf{a} \leftarrow \mathbb{Z}_p^{1\times 3},\ \mathbf{b}^\top \leftarrow \mathbb{Z}_p^2,\ \mathbf{k} \leftarrow \mathbb{Z}_p^{1\times 3}.$$

For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{3\times 2},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{3\times 2(2m+\ell)},\ \mathbf{R}_i \leftarrow \mathbb{Z}_p^{4\times 3},\ r_i \leftarrow \mathbb{Z}_p.$$

For all $i \in [L]$, write $\mathbf{A}_i = \left(\begin{smallmatrix} \mathbf{a} \\ \mathbf{R}_i \end{smallmatrix}\right)$ and sample

$$\mathbf{a}_i' \leftarrow \mathbb{Z}_p^{1\times 2},\ \mathbf{b}_i'^\top \leftarrow \mathbb{Z}_p^2,\ \mathbf{K}_i' \leftarrow \mathbb{Z}_p^{5\times 2},\ \mathbf{K}_{i,0}', \mathbf{K}_{i,1}' \leftarrow \mathbb{Z}_p^{2\times 2}$$

and compute

$$\mathbf{P}_i = \mathbf{A}_i^\top \mathbf{K}_i', \quad \mathbf{p}_{i,0} = \mathbf{a}_i' \mathbf{K}_{i,0}', \quad \mathbf{p}_{i,1} = \mathbf{a}_i' \mathbf{K}_{i,1}';$$
$$\mathbf{c}_i'^\top = \mathbf{K}_i' \mathbf{b}_i'^\top, \quad \mathbf{c}_{i,0}'^\top = \mathbf{K}_{i,0}' \mathbf{b}_i'^\top, \quad \mathbf{c}_{i,1}'^\top = \mathbf{K}_{i,1}' \mathbf{b}_i'^\top.$$

For all $i \in [L]$, set

$$\mathsf{crs}_i = ([\mathbf{a}_i', \mathbf{P}_i, \mathbf{p}_{i,0}, \mathbf{p}_{i,1}]_1, [\mathbf{b}_i'^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2) \quad \mathsf{td}_i = \mathbf{K}_i'.$$

Output

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{a}]_1, [\mathbf{a}\mathbf{k}^\top]_T,\ \{\mathsf{crs}_i, [\mathbf{R}_i, \mathbf{a}\mathbf{V}_i, \mathbf{a}\mathbf{W}_i]_1\}_{i\in[L]} \\ \{[\mathbf{b}^\top r_j, \mathbf{V}_j \mathbf{b}^\top r_j + \mathbf{k}^\top]_2\}_{j\in[L]} \\ \{[\mathbf{V}_i \mathbf{b}^\top r_j, \mathbf{W}_i(\mathbf{I}_{2m+\ell} \otimes \mathbf{b}^\top r_j)]_2\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}.$$

– $\mathsf{Gen}(\mathsf{crs}, i)$: Sample $\mathbf{U}_i \leftarrow \mathbb{Z}_p^{3\times 2}$. Define $\mathbf{M}_i = \left(\begin{smallmatrix} \mathbf{t}_i \\ \mathbf{Q}_i \end{smallmatrix}\right) = \left(\begin{smallmatrix} \mathbf{a}\mathbf{U}_i \\ \mathbf{R}_i\mathbf{U}_i \end{smallmatrix}\right)$, sample $\mathbf{s}_i^\top \leftarrow \mathbb{Z}_p^2$, and compute

$$\pi_i = [\underbrace{\mathbf{U}_i^\top \mathbf{P}_i + \mathbf{s}_i^\top (\mathbf{p}_{i,0} + \mathbf{p}_{i,1})}_{\pi_{i,0}}, \underbrace{\mathbf{s}_i^\top \mathbf{a}_i'}_{\pi_{i,1}}]_1$$

Fetch $[\mathbf{R}_i]_1$ and $\{[\mathbf{b}^\top r_j]_2\}_{j\in[L]\setminus\{i\}}$ from crs and output

$$\mathsf{pk}_i = \big([\underbrace{\mathbf{a}\mathbf{U}_i}_{\mathbf{t}_i}, \underbrace{\mathbf{R}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{\underbrace{[\mathbf{U}_i \mathbf{b}^\top r_j]_2}_{\mathbf{h}_{i,j}^\top}\}_{j\in[L]\setminus\{i\}}, \pi_i\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

– $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i)$: Parse $\mathsf{pk}_i = \left([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right)$ and fetch $[\mathbf{b}_i^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2$ from crs_i in crs. Write $\mathbf{M}_i = \binom{\mathbf{t}_i}{\mathbf{Q}_i}$ and parse $\pi_i = [\pi_{i,0}, \pi_{i,1}]_1$, check

$$e([\pi_{i,0}]_1, [\mathbf{b}_i'^\top]_2) \stackrel{?}{=} e([\mathbf{M}_i^\top]_1, [\mathbf{c}_i'^\top]_2) \cdot e([\pi_{i,1}]_1, [\mathbf{c}_{i,0}'^\top + \mathbf{c}_{i,1}'^\top]_2)$$

For each $j \in [L] \setminus \{i\}$, check

$$e([\mathbf{a}]_1, [\mathbf{h}_{i,j}^\top]_2) \stackrel{?}{=} e([\mathbf{t}_i]_1, [\mathbf{b}^\top r_j]_2).$$

If all these checks pass, output 1; otherwise, output 0.

– $\mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i, \mathbf{x}_i)_{i\in[L]})$: For all $i \in [L]$, parse

$$\mathsf{pk}_i = \left([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right).$$

Output:

$$\mathsf{mpk} = \left([\mathbf{a}]_1, \left[\sum_{i\in[L]} \left((\mathbf{a}\mathbf{V}_i + \mathbf{t}_i)((1\|\mathbf{0}_m)\otimes\mathbf{I}_2) + \mathbf{a}\mathbf{W}_i\left(\begin{pmatrix}\mathbf{0}_m & \mathsf{diag}(\mathbf{x}_i)\\ \mathbf{0}_m & \mathbf{I}_m\\ \mathbf{e}_1^\top & \mathbf{0}_{\ell\times m}\end{pmatrix}\otimes\mathbf{I}_2\right)\right)\right]_1, \left[\sum_{i\in[L]}\mathbf{a}\mathbf{W}_i\right]_1, [\mathbf{a}\mathbf{k}^\top]_T\right)$$

and for all $j \in [L]$

$$\mathsf{hsk}_j = \Bigg([\underbrace{\mathbf{b}^\top r_j}_{\mathbf{k}_0^\top}]_2, [\underbrace{\mathbf{V}_j\mathbf{b}^\top r_j + \mathbf{k}^\top}_{\mathbf{k}_1^\top}]_2, \underbrace{\left[\sum_{i\in[L]\setminus\{j\}}\left((\mathbf{V}_i\mathbf{b}^\top r_j + \mathbf{h}_{i,j}^\top)(1\|\mathbf{0}_m) + \mathbf{W}_i(\mathbf{I}_{2m+\ell}\otimes\mathbf{b}^\top r_j)\begin{pmatrix}\mathbf{0}_m & \mathsf{diag}(\mathbf{x}_i)\\ \mathbf{0}_m & \mathbf{I}_m\\ \mathbf{e}_1^\top & \mathbf{0}_{\ell\times m}\end{pmatrix}\right)\right]_2}_{\mathbf{K}_2},$$

$$\underbrace{\left[\sum_{i\in[L]\setminus\{j\}}\mathbf{W}_i(\mathbf{I}_{2m+\ell}\otimes\mathbf{b}^\top r_j)\right]_2}_{\mathbf{K}_3}\Bigg).$$

– $\mathsf{Enc}(\mathsf{mpk}, (\mathbf{Y}, \mathbf{Z}), \mathsf{m})$: Sample $s \leftarrow \mathbb{Z}_p$. Output:

$$\mathsf{ct}_{\mathbf{Y},\mathbf{Z}} = \Bigg([\underbrace{s\mathbf{a}}_{\mathbf{c}_0}]_1, \underbrace{\left[\sum_{i\in[L]}\left((s\mathbf{a}\mathbf{V}_i + s\mathbf{t}_i)((1\|\mathbf{0}_m)\otimes\mathbf{I}_2) + s\mathbf{a}\mathbf{W}_i\left(\begin{pmatrix}\mathbf{0}_m & \mathsf{diag}(\mathbf{x}_i)\\ \mathbf{0}_m & \mathbf{I}_m\\ \mathbf{e}_1^\top & \mathbf{0}_{\ell\times m}\end{pmatrix}\otimes\mathbf{I}_2\right)\right)\right]_1}_{\mathbf{c}_1},$$

$$\underbrace{\left[\sum_{i\in[L]} s\mathbf{a}\mathbf{W}_i\left(\begin{pmatrix}\mathbf{I}_m & \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m} & \mathbf{I}_m\\ \mathbf{Y}^\top & \mathbf{Z}^\top\end{pmatrix}\otimes\mathbf{I}_2\right)\right]_1}_{\mathbf{c}_2}, \underbrace{[s\mathbf{a}\mathbf{k}^\top]_T \cdot \mathsf{m}}_{C}\Bigg).$$

– $\mathsf{Dec}(\mathsf{sk}_{i^*}, \mathsf{hsk}_{i^*}, \mathsf{ct}_{\mathbf{Y},\mathbf{Z}})$: Parse

$$\mathsf{sk}_{i^*} = \mathbf{U}_{i^*}, \quad \mathsf{hsk}_{i^*} = [\mathbf{k}_0^\top, \mathbf{k}_1^\top, \mathbf{K}_2, \mathbf{K}_3]_2, \quad \mathsf{ct}_x = ([\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2]_1, C).$$

recover

$$[\mathbf{z}_1]_T = e([\mathbf{c}_1\|\mathbf{c}_2]_1, [\mathbf{I}_{3m+1}\otimes\mathbf{k}_0^\top]_2), \qquad [\mathbf{z}_2]_T = e\left([\mathbf{c}_0]_1, \left[\mathbf{K}_2\|\mathbf{K}_3\begin{pmatrix}\mathbf{I}_m & \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m} & \mathbf{I}_m\\ \mathbf{Y}^\top & \mathbf{Z}^\top\end{pmatrix}\right]_2\right),$$

$$[z_3]_T = e([\mathbf{c}_0\mathbf{U}_{i^*}]_1, [\mathbf{k}_0^\top]_2), \qquad [z_4]_T = e([\mathbf{c}_0]_1, [\mathbf{k}_1^\top]_2).$$

Compute $\boldsymbol{\omega}$ such that $\mathbf{e}_1 = \boldsymbol{\omega}(\mathsf{diag}(\mathbf{x}_{i^*}) \cdot \mathbf{Y} + \mathbf{Z})$, output

$$\mathsf{m}' = [(\mathbf{z}_1 - \mathbf{z}_2) \cdot (1\|\boldsymbol{\omega}\| - \boldsymbol{\omega}\cdot\mathsf{diag}(\mathbf{x}_{i^*})\| - \boldsymbol{\omega})^\top - z_3 - z_4]_T \cdot C.$$

Acknowledgement. We want to thank anonymous reviewers from Asiacrypt 2023 for their insightful comments and Hoeteck Wee for his encouragement! This work is partially supported by National Natural Science Foundation of China (62002120), Shanghai Rising-Star Program (22QA1403800), Innovation Program of Shanghai Municipal Education Commission (2021-01-07-00-08-E00101) and the "Digital Silk Road" Shanghai International Joint Lab of Trustworthy Intelligent Software (22510750100).

References

ABS17. Miguel Ambrona, Gilles Barthe, and Benedikt Schmidt. Generic transformations of predicate encodings: Constructions and applications. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 36–66. Springer, Heidelberg, August 2017. 2, 3, 10

ACGU20. Michel Abdalla, Dario Catalano, Romain Gay, and Bogdan Ursu. Inner-product functional encryption with fine-grained access control. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 467–497. Springer, Heidelberg, December 2020. 3, 10

Att14. Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 557–577. Springer, Heidelberg, May 2014. 8

BHR12. Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM CCS 2012*, pages 784–796. ACM Press, October 2012. 1

CES21. Kelong Cong, Karim Eldefrawy, and Nigel P. Smart. Optimizing registration based encryption. *IACR Cryptol. ePrint Arch.*, page 499, 2021. 1, 2

CGW15. Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system ABE in prime-order groups via predicate encodings. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 595–624. Springer, Heidelberg, April 2015. 1, 3, 6, 7, 8, 10, 19, 20, 39, 40, 41

DKL+23. Nico Döttling, Dimitris Kolonelos, Russell W. F. Lai, Chuanwei Lin, Giulio Malavolta, and Ahmadreza Rahimi. Efficient laconic cryptography from learning with errors. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 417–446. Springer, Heidelberg, April 2023. 1, 2

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, Heidelberg, August 2013. 9

FFM+23. Danilo Francati, Daniele Friolo, Monosij Maitra, Giulio Malavolta, Ahmadreza Rahimi, and Daniele Venturi. Registered (inner-product) functional encryption. Cryptology ePrint Archive, Paper 2023/395, 2023. https://eprint.iacr.org/2023/395. 1, 2, 4, 8

FWW23. Cody Freitag, Brent Waters, and David J. Wu. How to use (plain) witness encryption: Registered abe, flexible broadcast, and more. Cryptology ePrint Archive, Paper 2023/812, 2023. https://eprint.iacr.org/2023/812. 8

GGH+13. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013. 1

GGSW13. Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013. 8

GHM+19. Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, Ahmadreza Rahimi, and Sruthi Sekar. Registration-based encryption from standard assumptions. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 63–93. Springer, Heidelberg, April 2019. 1, 2

GHMR18. Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 689–718. Springer, Heidelberg, November 2018. 1, 2

GHR15. Alonso González, Alejandro Hevia, and Carla Ràfols. QA-NIZK arguments in asymmetric groups: New tools and new constructions. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 605–629. Springer, Heidelberg, November / December 2015. 7, 11

GKMR22. Noemi Glaeser, Dimitris Kolonelos, Giulio Malavolta, and Ahmadreza Rahimi. Efficient registration-based encryption. Cryptology ePrint Archive, Report 2022/1505, 2022. https://eprint.iacr.org/2022/1505. 1, 2

GPSW06. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309. 1

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, Heidelberg, April 2008. 7

GV20. Rishab Goyal and Satyanarayana Vusirikala. Verifiable registration-based encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 621–651. Springer, Heidelberg, August 2020. 1, 2

GW09. Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 171–188. Springer, Heidelberg, April 2009. 5

GW20. Junqing Gong and Hoeteck Wee. Adaptively secure ABE for DFA from k-Lin and more. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 278–308. Springer, Heidelberg, May 2020. 8

GWW19. Junqing Gong, Brent Waters, and Hoeteck Wee. ABE for DFA from k-Lin. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 732–764. Springer, Heidelberg, August 2019. 8

HLWW23. Susan Hohenberger, George Lu, Brent Waters, and David J. Wu. Registered attribute-based encryption. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 511–542. Springer, Heidelberg, April 2023. 1, 2, 4, 7, 8, 9, 10, 11, 26, 27

IW14. Yuval Ishai and Hoeteck Wee. Partial garbling schemes and their applications. Cryptology ePrint Archive, Paper 2014/995, 2014. https://eprint.iacr.org/2014/995. 19, 20

JLS22. Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 670–699. Springer, Heidelberg, May / June 2022. 1

JR13. Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2013. 7, 11

KMW23. Dimitris Kolonelos, Giulio Malavolta, and Hoeteck Wee. Distributed broadcast encryption from bilinear groups. Cryptology ePrint Archive, Paper 2023/874, 2023. https://eprint.iacr.org/2023/874. 8

KW15. Eike Kiltz and Hoeteck Wee. Quasi-adaptive NIZK for linear subspaces revisited. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 101–128. Springer, Heidelberg, April 2015. 11, 14, 27, 28, 29

KW19. Lucas Kowalczyk and Hoeteck Wee. Compact adaptively secure ABE for NC^1 from k-Lin. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 3–33. Springer, Heidelberg, May 2019. 8

LL20. Huijia Lin and Ji Luo. Compact adaptively secure ABE from k-Lin: Beyond NC^1 and towards NL. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 247–277. Springer, Heidelberg, May 2020. 8

LOS+10. Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 62–91. Springer, Heidelberg, May / June 2010. 7

LPJY15. Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Compactly hiding linear spans - tightly secure constant-size simulation-sound QA-NIZK proofs and applications. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 681–707. Springer, Heidelberg, November / December 2015. 7, 11

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, Heidelberg, February 2010. 43

MRV16. Paz Morillo, Carla Ràfols, and Jorge Luis Villar. The kernel matrix Diffie-Hellman assumption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 729–758. Springer, Heidelberg, December 2016. 27

SW05. Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005. 1

VWW22. Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Witness encryption and null-IO from evasive LWE. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 195–221. Springer, Heidelberg, December 2022. 8

Wat09. Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Heidelberg, August 2009. 6, 16

Wee14. Hoeteck Wee. Dual system encryption via predicate encodings. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 616–637. Springer, Heidelberg, February 2014. 1, 3, 6, 10

Yao82. Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982. 1

Appendix

A Registered Attribute-Based Encryption

Algorithms. A *registered attribute-based encryption* (Reg-ABE) for predicate $P : X \times Y \to \{0,1\}$ consists of six algorithms:

- $\mathsf{Setup}(1^\lambda, P) \to \mathsf{crs}$: It takes as input the security parameter 1^λ, description of predicate P, outputs a common reference string crs.
- $\mathsf{Gen}(\mathsf{crs}, \mathsf{aux}) \to (\mathsf{pk}, \mathsf{sk})$: It takes as input crs and the public state aux, outputs key pair $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Reg}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, y) \to (\mathsf{mpk}, \mathsf{aux}')$: It takes as input crs, aux, and pk along with $y \in Y$, outputs master public key mpk and updated state aux′.
- $\mathsf{Enc}(\mathsf{mpk}, x, \mathsf{m}) \to \mathsf{ct}$: It takes as input mpk, $x \in X$ and message m, outputs a ciphertext ct.
- $\mathsf{Upd}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}) \to \mathsf{hsk}$: It take as input crs, aux, pk, outputs a helper key hsk.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{hsk}, \mathsf{ct}) \to \mathsf{m}/\perp/\mathtt{getupd}$: It take as input sk, hsk, ct and outputs m or a special symbol $\perp$ to indicate a decryption failure, or a special flag `getupd` to indicate the need of an updated helper key.

Correctness. For all stateful adversary $\mathcal{A}$, the following advantage function is negligible in λ:

$$\Pr[b = 1 | \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, P); b = 0; \mathcal{A}^{\mathsf{ORegNT}(\cdot,\cdot),\mathsf{ORegT}(\cdot),\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{ODec}(\cdot)}(\mathsf{crs})]$$

where the oracles work as follows with initial setting $\mathsf{aux} = \perp$, $\mathcal{E} = \emptyset$, $\mathcal{R} = \emptyset$ and $t = \perp$:

- $\mathsf{ORegNT}(\mathsf{pk}, y)$: run $(\mathsf{mpk}, \mathsf{aux}') \leftarrow \mathsf{Reg}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, y)$, update $\mathsf{aux} = \mathsf{aux}'$, append $(\mathsf{mpk}, \mathsf{aux})$ to $\mathcal{R}$ and return $(|\mathcal{R}|, \mathsf{mpk}, \mathsf{aux})$;
- $\mathsf{ORegT}(y^*)$: run $(\mathsf{pk}^*, \mathsf{sk}^*) \leftarrow \mathsf{Gen}(\mathsf{crs}, \mathsf{aux})$, $(\mathsf{mpk}, \mathsf{aux}') \leftarrow \mathsf{Reg}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}^*, y^*)$, update $\mathsf{aux} = \mathsf{aux}'$, compute $\mathsf{hsk}^* \leftarrow \mathsf{Upd}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}^*)$, append $(\mathsf{mpk}, \mathsf{aux})$ to $\mathcal{R}$, return $(t = |\mathcal{R}|, \mathsf{mpk}, \mathsf{aux}, \mathsf{pk}^*, \mathsf{sk}^*, \mathsf{hsk}^*)$;
- $\mathsf{OEnc}(i, x, \mathsf{m})$: let $\mathcal{R}[i] = (\mathsf{mpk}, \star)$, run $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x, \mathsf{m})$, append $(x, \mathsf{m}, \mathsf{ct})$ to $\mathcal{E}$ and return $(|\mathcal{E}|, \mathsf{ct})$;
- $\mathsf{ODec}(j)$: let $\mathcal{E}[j] = (x_j, \mathsf{m}_j, \mathsf{ct}_j)$, compute $\mathsf{m}'_j \leftarrow \mathsf{Dec}(\mathsf{sk}^*, \mathsf{hsk}^*, \mathsf{ct}_j)$; if $\mathsf{m}'_j = \mathtt{getupd}$, run $\mathsf{hsk}^* \leftarrow \mathsf{Upd}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}^*)$ and recompute $\mathsf{m}'_j \leftarrow \mathsf{Dec}(\mathsf{sk}^*, \mathsf{hsk}^*, \mathsf{ct}_j)$. Set $b = 1$ when $\mathsf{m}'_j \neq \mathsf{m}_j$.

with the following restrictions:

- there exists one query to ORegT; (we can consider $y^*, \mathsf{pk}^*, \mathsf{sk}^*, \mathsf{hsk}^*$ to be global;)
- for query $(i, x, \star)$ to OEnc, it holds that $i \geq t, \mathcal{R}[i] \neq \perp$ and $P(x, y^*) = 1$;
- for query (j) to ODec, it holds that $\mathcal{E}[j] \neq \perp$.

Compactness and Efficiency. Let $\mathcal{R}$ be defined as before. *Compactness* means that

$$|\mathsf{mpk}_i| = \mathsf{poly}(\lambda, \mathsf{par}, \log i), \quad |\mathsf{hsk}^*| = \mathsf{poly}(\lambda, \mathsf{par}, \log |\mathcal{R}|);$$

where we let $\mathcal{R}[i] = (\mathsf{mpk}_i, \star)$ for all $i \in [|\mathcal{R}|]$ and par is a parameter depending on the predicate P. Furthermore, *update efficiency* means that the number of invocations of Upd in ODec is at most $O(\log |\mathcal{R}|)$ and each invocation costs $\mathsf{poly}(\log |\mathcal{R}|)$ time (in RAM model).

Security. For all stateful adversary $\mathcal{A}$, the advantage

$$\Pr\left[b = b' \;\middle|\; \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, P); \\ x^*, \mathsf{m}_0^*, \mathsf{m}_1^* \leftarrow \mathcal{A}^{\mathsf{ORegCK}(\cdot,\cdot),\mathsf{ORegHK}(\cdot),\mathsf{OCorHK}(\cdot)}(\mathsf{crs}); \\ b \leftarrow \{0,1\}, \mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{mpk}, x^*, \mathsf{m}_b^*), b' \leftarrow \mathcal{A}(\mathsf{ct}^*) \end{array}\right] - \frac{1}{2}$$

is negligible in λ, where the oracles as follows with initial setting $\mathsf{aux}, \mathsf{mpk} = \perp$, $\mathcal{R} = \emptyset, \mathcal{C} = \emptyset$ and $\mathcal{D}$ being a dictionary with $\mathcal{D}[\mathsf{pk}] = \emptyset$ for all possible pk:

– ORegCK(pk, y): run $(\mathsf{mpk}', \mathsf{aux}') \leftarrow \mathsf{Reg}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, y)$, update $\mathsf{mpk} = \mathsf{mpk}'$, $\mathsf{aux} = \mathsf{aux}'$, $\mathcal{D}[\mathsf{pk}] = \mathcal{D}[\mathsf{pk}] \cup \{y\}$, append pk to C and return $(\mathsf{mpk}, \mathsf{aux})$;
– ORegHK(y): run $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}, \mathsf{aux})$ and $(\mathsf{mpk}', \mathsf{aux}') \leftarrow \mathsf{Reg}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, y)$, update $\mathsf{mpk} = \mathsf{mpk}', \mathsf{aux} = \mathsf{aux}'$, $\mathcal{D}[\mathsf{pk}] = \mathcal{D}[\mathsf{pk}] \cup \{y\}$, append $(\mathsf{pk}, \mathsf{sk})$ to $\mathcal{R}$ and return $(|\mathcal{R}|, \mathsf{mpk}, \mathsf{aux}, \mathsf{pk})$;
– OCor(i): let $\mathcal{R}[i] = (\mathsf{pk}, \mathsf{sk})$, append pk to C and return sk;

with the following restrictions:

– for query i to OCor, it holds that $\mathcal{R}[i] \neq \perp$;
– for all $y \in \bigcup_{\mathsf{pk} \in C} \mathcal{D}[\mathsf{pk}]$, it holds that $P(x^*, y) = 0$.

We use $\mathsf{Adv}_{\mathcal{A}}^{Reg-ABE}(\lambda)$ to denote the advantage function.

From Slotted Reg-ABE to Reg-ABE. To transform a slotted Reg-ABE to a full-fledged Reg-ABE, we need the "power-of-two" approach from [HLWW23]. Suppose a full-fledged Reg-ABE mostly supports $L = 2^\ell$ users, this approach needs ℓ+1 copies of slotted Reg-ABE with $1, 2, 4, \cdots, 2^\ell$ slots. And the public state $\mathsf{aux} = (\mathcal{D}_1, \mathcal{D}_2, \mathsf{mpk})$ consists of the following terms:

– $\mathcal{D}_1[k, i] = (\mathsf{pk}, y)$: where $k \in [0, \ell]$ and $i \in [2^k]$. And this dictionary assigns a user's (pk, y) to the slot i of the 2^k-slotted Reg-ABE scheme.
– $\mathcal{D}_2[k, n] = \mathsf{hsk}$: where $k \in [0, \ell]$ and $n \in [L]$. And this dictionary assigns a hsk of slotted Reg-ABE to the 2^k-slotted Reg-ABE scheme and the user index n.
– $\mathsf{mpk} = (\mathsf{ctr}, \mathsf{mpk}_0, \cdots, \mathsf{mpk}_\ell)$ denotes the current master public key. Where $(\mathsf{mpk}_k)_{k \in [0,\ell]}$ denote the master public keys of ℓ+1 copies of slotted Reg-ABE, and ctr denotes the number of currently registered users. When no registered user, we initial set $\mathsf{mpk} = \perp = (0, \perp, \cdots \perp)$.

When no registered user, we initial set $\mathsf{aux} = \perp = (\emptyset, \emptyset, \perp)$.

Assuming a slotted Reg-ABE $\Pi_s = (s.\mathsf{Setup}, s.\mathsf{Gen}, s.\mathsf{Ver}, s.\mathsf{Agg}, s.\mathsf{Dec})$, the full-fledged Reg-ABE from "power-of-two" approach is as follow:

– $\mathsf{Setup}(1^\lambda, P, 1^L)$: Compute $\ell = \log(L)$, for all $k \in [0, \ell]$, run $\mathsf{crs}_k \leftarrow s.\mathsf{Setup}(1^\lambda, P, 1^{(2^k)})$. Output

$$\mathsf{crs} = (\mathsf{crs}_0, \cdots, \mathsf{crs}_\ell)$$

– $\mathsf{Gen}(\mathsf{crs}, \mathsf{aux})$: Fetch $\mathsf{crs} = (\mathsf{crs}_k)_{k \in [0,\ell]}$ and $\mathsf{aux} = (\mathcal{D}_1, \mathcal{D}_2, \mathsf{mpk})$, where $\mathsf{mpk} = \big(\mathsf{ctr}_{\mathsf{aux}}, (\mathsf{mpk}_k)_{k \in [0,\ell]}\big)$. For all $k \in [0, \ell]$, compute

$$i_k = (\mathsf{ctr} \pmod{2^k}) + 1$$

and run $(\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow s.\mathsf{Gen}(\mathsf{crs}_k, i_k)$. Set $\mathsf{ctr}' = \mathsf{ctr}$ and output

$$\mathsf{pk} = (\mathsf{ctr}', \mathsf{pk}_0, \cdots, \mathsf{pk}_\ell) \quad \text{and} \quad \mathsf{sk} = (\mathsf{ctr}', \mathsf{sk}_0 \cdots, \mathsf{sk}_\ell)$$

– $\mathsf{Reg}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, y)$: Fetch $\mathsf{crs} = (\mathsf{crs}_k)_{k \in [0,\ell]}$, $\mathsf{aux} = (\mathcal{D}_1, \mathcal{D}_2, \mathsf{mpk})$, and $\mathsf{pk} = \big(\mathsf{ctr}', (\mathsf{pk}_k)_{k \in [0.\ell]}\big)$; where $\mathsf{mpk} = (\mathsf{ctr}, (\mathsf{mpk}_k)_{k \in [0,\ell]})$. For all $k \in [0, \ell]$, do the following operates:
  • Compute $i_k = (\mathsf{ctr} \pmod{2^k}) + 1$.
  • Check if $\mathsf{Ver}(\mathsf{crs}_k, i_k, \mathsf{pk}_k) = 1$ and $\mathsf{ctr}' = \mathsf{ctr}$. If the check passes, set $\mathsf{ctr} = \mathsf{ctr} + 1$, if the check fails, the algorithm halts and output $(\mathsf{mpk}, \mathsf{aux})$.
  • Update $\mathcal{D}_1[k, i_k] = (\mathsf{pk}, y)$
  • If $i_k = 2^k$: compute $(\mathsf{mpk}'_k, (\mathsf{hsk}_{k,j})_{j \in [2^k]}) \leftarrow s.\mathsf{Agg}(\mathsf{crs}_k, (\mathcal{D}_1[k, i])_{i \in [2^k]})$. Update $\mathsf{mpk}_k = \mathsf{mpk}'_k$, and for all $j \in [2^k]$, update $\mathcal{D}_2[k, \mathsf{ctr} + 1 - 2^k + j] = \mathsf{hsk}_{k,j}$.

  Update the master public key $\mathsf{mpk} = (\mathsf{ctr}, (\mathsf{mpk}_0, \cdots, \mathsf{mpk}_\ell)$ and $\mathsf{aux} = (\mathcal{D}_1, \mathcal{D}_2, \mathsf{mpk})$, output $(\mathsf{mpk}, \mathsf{aux})$.

– $\mathsf{Enc}(\mathsf{mpk}, x, \mathsf{m})$: Fetch $\mathsf{mpk} = (\mathsf{ctr}, (\mathsf{mpk}_k)_{k\in[0,\ell]})$. For all $k \in [0, \ell]$, compute:

$$\mathsf{ct}_k = \begin{cases} \bot & \text{if } \mathsf{mpk}_k = \bot \\ s.\mathsf{Enc}(\mathsf{mpk}_k, x, \mathsf{m}) & \text{if } \mathsf{mpk}_k \neq \bot \end{cases}$$

Output

$$\mathsf{ct} = (\mathsf{ctr}, \mathsf{ct}_0, \cdots, \mathsf{ct}_\ell)$$

– $\mathsf{Upd}(\mathsf{crs}, \mathsf{aux.pk})$: Fetch $\mathsf{crs} = (\mathsf{crs}_k)_{k\in[0,\ell]}$, $\mathsf{aux} = (\mathcal{D}_1, \mathcal{D}_2, \mathsf{mpk})$, and $\mathsf{pk} = \left(\mathsf{ctr}', (\mathsf{pk}_k)_{k\in[0.\ell]}\right)$; where $\mathsf{mpk} = (\mathsf{ctr}, (\mathsf{mpk}_k)_{k\in[0,\ell]})$. Output

$$\mathsf{hsk} = \begin{cases} \bot & \text{if } \mathsf{ctr}' \geq \mathsf{ctr} \\ (\overbrace{\mathcal{D}_2[0, \mathsf{ctr}+1]}^{\mathsf{hsk}_0}, \cdots \overbrace{\mathcal{D}_2[\ell, \mathsf{ctr}+1]}^{\mathsf{hsk}_\ell}) & \text{if } \mathsf{ctr}' < \mathsf{ctr} \end{cases}$$

– $\mathsf{Dec}(\mathsf{sk}, \mathsf{hsk}, \mathsf{ct})$: Fetch $\mathsf{sk} = (\mathsf{ctr}', (\mathsf{sk}_k)_{k\in[0,\ell]})$, $\mathsf{hsk} = (\mathsf{hsk}_k)_{k\in[0,\ell]}$ and $\mathsf{ct} = (\mathsf{ctr}, (\mathsf{ct}_k)_{k\in[0,\ell]})$. Proceed as follows:
- If $\mathsf{ctr}' \geq \mathsf{ctr}$: output $\bot$.
- Otherwise, compute $\mathsf{ctr} = (a_\ell, \cdots, a_0)_2$ and $\mathsf{ctr}' = (b_\ell, \cdots, b_0)_2$. We denote k_d as the maximum $k(\in [0, \ell])$ such that $a_k \neq b_k$. If $\mathsf{hsk}_{k_d} = \bot$: output $\mathtt{getupd}$.
- Otherwise, output $s.\mathsf{Dec}(\mathsf{sk}_{k_d}, \mathsf{hsk}_{k_d}, \mathsf{ct}_{k_d})$.

And [HLWW23, Appendix 6] proved that perfect correctness, compactness, and efficiency (defined in Section 2.2) of the slotted Reg-ABE scheme Π_s implies perfect correctness, compactness, and efficiency (defined in Appendix A) of the Reg-ABE construction presented above.

B QANIZK with Stronger Soundness

We review the pairing-based QA-NIZK for linear space in [KW15] based on *Kernel Diffie-Hellman Assumption* [MRV16] and verify that the proof works well for stronger simulation soundness defined in Section 2.4. We review the *Kernel Diffie-Hellman Assumption* [MRV16] as follows and note that it is implied by MDDH assumption 2.1.

Assumption 2 ((k, ℓ)-KerMDH for $s \in \{1, 2\}$) *Let $k, \ell \in \mathbb{N}$ with $k < \ell$. We say that the (k, ℓ)-KerMDH assumption holds in $\mathbb{G}_s$ if for all PPT adversaries $\mathcal{A}$, the following advantage function is negligible in λ.*

$$\mathsf{Adv}^{\mathrm{KerMDH}}_{\mathcal{A},s,k,\ell}(\lambda) = \Pr[\mathbf{M}\mathbf{c}^\top = \mathbf{0} \wedge \mathbf{c} \neq \mathbf{0} \mid [\mathbf{c}]_{3-s} \leftarrow \mathcal{A}(\mathbb{G}, [\mathbf{M}]_s)]$$

where $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and $\mathbf{M} \leftarrow \mathbb{Z}_p^{k\times\ell}$.

Scheme. The QA-NIZK scheme in [KW15] works as follows. When applying it to our slotted Reg-ABE scheme, $\mathbf{M}$ corresponds to $\mathbf{A}_i$, $\mathbf{D}$ corresponds to $\mathbf{M}_i$ and crs' corresponds to crs_i where additional subscript is using to indicate slot.

– $\mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{M}]_1 \in \mathbb{G}_1^{(3k+2)\times(2k+1)})$: Sample

$$\mathbf{A}' \leftarrow \mathbb{Z}_p^{k\times(k+1)},\ \mathbf{B}' \leftarrow \mathbb{Z}_p^{(k+1)\times k},\ \mathbf{K}' \leftarrow \mathbb{Z}_p^{(3k+2)\times(k+1)},\ \mathbf{K}'_0, \mathbf{K}'_1 \leftarrow \mathbb{Z}_p^{(k+1)\times(k+1)}$$

Compute

$$\mathbf{P} = \mathbf{M}^\top\mathbf{K}', \quad \mathbf{P}_0 = \mathbf{A}'\mathbf{K}'_0, \quad \mathbf{P}_1 = \mathbf{A}'\mathbf{K}'_1;$$
$$\mathbf{C}' = \mathbf{K}'\mathbf{B}', \quad \mathbf{C}'_0 = \mathbf{K}'_0\mathbf{B}', \quad \mathbf{C}'_1 = \mathbf{K}'_1\mathbf{B}'.$$

And output

$$\mathsf{crs}' = ([\mathbf{A}', \mathbf{P}, \mathbf{P}_0, \mathbf{P}_1]_1, [\mathbf{B}', \mathbf{C}', \mathbf{C}'_0, \mathbf{C}'_1]_2) \quad \mathsf{td} = \mathbf{K}'$$

– $\mathsf{LPrv}(\mathsf{crs}', [\mathbf{D}]_1 \in \mathbb{G}_1^{(3k+2)\times(k+1)}, \mathbf{U} \in \mathbb{Z}_p^{(2k+1)\times(k+1)})$: Sample $\mathbf{S} \leftarrow \mathbb{Z}_p^{(k+1)\times k}$, output

$$\pi = [\underbrace{\mathbf{U}^\top\mathbf{P} + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1)}_{\pi_0}, \underbrace{\mathbf{S}\mathbf{A}'}_{\pi_1}]_1$$

– $\mathsf{LVer}(\mathsf{crs}', [\mathbf{D}]_1, \pi)$: Parse $\pi = [\pi_0, \pi_1]_1$ and check

$$e([\pi_0]_1, [\mathbf{B}']_2) \stackrel{?}{=} e([\mathbf{D}^\top]_1, [\mathbf{C}']_2) \cdot e([\pi_1]_1, [\mathbf{C}'_0 + \mathbf{C}'_1])$$

if this pass, output 1; otherwise, output 0.

– $\mathsf{LSim}(\mathsf{crs}', \mathsf{td}, [\mathbf{D}]_1)$: Sample $\mathbf{S} \leftarrow \mathbb{Z}_p^{(k+1)\times k}$, output

$$\widetilde{\pi} = [\underbrace{\mathbf{D}^\top\mathbf{K}' + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1)}_{\widetilde{\pi}_0}, \underbrace{\mathbf{S}\mathbf{A}'}_{\widetilde{\pi}_1}]_1$$

Perfect Completeness. For all $\lambda \in \mathbb{N}$, all $\mathbf{M} \in \mathbb{Z}_p^{(3k+2)\times(2k+1)}$, all $\mathbf{U} \in \mathbb{Z}_p^{(2k+1)\times(k+1)}$, $\mathbf{D} \in \mathbb{Z}_p^{(3k+2)\times(k+1)}$ such that $\mathbf{D} = \mathbf{M}\mathbf{U}$, let $\mathsf{crs}' \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{M}]_1)$, $\pi \leftarrow \mathsf{LPrv}(\mathsf{crs}', [\mathbf{D}]_1, \mathbf{U})$, where

$$\pi = [\pi_0, \pi_1]_1 = [\mathbf{U}^\top\mathbf{P} + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1), \mathbf{S}\mathbf{A}']_1.$$

We have

$$e([\mathbf{U}^\top\mathbf{P} + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1)]_1, [\mathbf{B}']_2) = e([\mathbf{D}^\top]_1, [\mathbf{C}']_2) \cdot e([\mathbf{S}\mathbf{A}']_1, [\mathbf{C}'_0 + \mathbf{C}'_1])$$

This follows from the definition of bilinear map e (see Section 2.1), and the fact that $\mathbf{U}^\top\mathbf{P} \cdot \mathbf{B}' = \mathbf{U}^\top\mathbf{M}^\top \cdot \mathbf{K}'\mathbf{B}' = \mathbf{D}^\top \cdot \mathbf{C}'$ and the fact that $\mathbf{P}_b\mathbf{B}' = \mathbf{A}'\mathbf{C}'_b$ where $b \in \{0, 1\}$.

Perfect Zero-knowledge. For all $\lambda \in \mathbb{N}$, all $\mathbf{M} \in \mathbb{Z}_p^{(3k+2)\times(2k+1)}$, let $\mathsf{crs}' \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{M}]_1)$, and all $\mathbf{U} \in \mathbb{Z}_p^{(2k+1)\times(k+1)}$, $\mathbf{D} \in \mathbb{Z}_p^{(3k+2)\times(k+1)}$ such that $\mathbf{D} = \mathbf{M}\mathbf{U}$, we have

$$(\mathbf{U}^\top\mathbf{P} + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1), \mathbf{S}\mathbf{A}') \equiv (\mathbf{D}^\top\mathbf{K}' + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1), \mathbf{S}\mathbf{A}').$$

This follows from the fact that $\mathbf{U}^\top \cdot \mathbf{P} = \mathbf{U}^\top\mathbf{M}^\top \cdot \mathbf{K}' = \mathbf{D}^\top \cdot \mathbf{K}'$.

Proof Sketch of Unbounded Simulation Soundness. In [KW15], they prove the unbounded simulation soundness via the following game sequence, we will show that this game sequence can also prove the strong soundness, which mean that the $\mathbf{M}$ is public.

– H_0: This is the real game as define in Secntion 2.4. The adversary can get $\mathbf{M} \in \mathbb{Z}_p^{(3k+2)\times(2k+1)}$ and we have

$$\mathsf{crs}' = ([\mathbf{A}', \underbrace{\mathbf{M}^\top\mathbf{K}'}_{\mathbf{P}}, \underbrace{\mathbf{A}'\mathbf{K}'_0}_{\mathbf{P}_0}, \underbrace{\mathbf{A}'\mathbf{K}'_1}_{\mathbf{P}_1}]_1, [\mathbf{B}', \underbrace{\mathbf{K}'\mathbf{B}'}_{\mathbf{C}'}, \underbrace{\mathbf{K}'_0\mathbf{B}'}_{\mathbf{C}'_0}, \underbrace{\mathbf{K}'_1\mathbf{B}'}_{\mathbf{C}'_1}]_2) \quad \text{and} \quad \mathsf{td} = \mathbf{K}'$$

where $\mathbf{A}' \leftarrow \mathbb{Z}_p^{k\times(k+1)}$, $\mathbf{B}' \leftarrow \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{K}' \leftarrow \mathbb{Z}_p^{(3k+2)\times(k+1)}$, $\mathbf{K}'_0, \mathbf{K}'_1 \leftarrow \mathbb{Z}_p^{(k+1)\times(k+1)}$. For any query $[\mathbf{D}]_1$, the output of $\mathsf{LSim}(\mathsf{crs}', \mathsf{td}, [\mathbf{D}]_1)$ is

$$\widetilde{\pi} = [\underbrace{\mathbf{D}^\top\mathbf{K}' + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1)}_{\widetilde{\pi}_0}, \underbrace{\mathbf{S}\mathbf{A}'}_{\widetilde{\pi}_1}]_1$$

with the challenge $([\mathbf{D}^*]_1, \pi^*)$, parse $\pi^* = [\pi_0^*, \pi_1^*]_1$ and the LVer check:

$$e([\pi_0^*]_1, [\mathbf{B}']_2) \stackrel{?}{=} e([(\mathbf{D}^*)^\top]_1, [\mathbf{C}']_2) \cdot e([\pi_1^*]_2, [\mathbf{C}'_0 + \mathbf{C}'_1])$$

– H_1: Identical to H_0, except that on input $[\mathbf{D}^*]_1$ and $\pi^* = [\pi_0^*, \pi_1^*]_1$, the LVer check

$$[\pi_0^*]_1 \stackrel{?}{=} [(\mathbf{D}^*)^\top \mathbf{K}' + \pi_1^*(\mathbf{K}_0' + \mathbf{K}_1')]_1$$

We have $\mathsf{H}_1 \approx_c \mathsf{H}_0$, [KW15] argued this follows from the KerMDH assumption for $[\mathbf{B}]_2$ defined in Section 2.1, even if $\mathbf{M}$ is public, this argument can still hold.

– H_2: Identical to H_1 except that we generate $\widetilde{\pi}$ as follow

$$\widetilde{\pi} = [\underbrace{\mathbf{D}^\top \mathbf{K}' + \boxed{\mathbf{v}\mathbf{d}^\perp} + \mathbf{S}(\mathbf{P}_0 + \mathbf{P}_1)}_{\widetilde{\pi}_0}, \underbrace{\mathbf{S}\mathbf{A}'}_{\widetilde{\pi}_1}]_1$$

where $\mathbf{v} \leftarrow \mathbb{Z}_p^{k+1}$ and $\mathbf{d}^\perp \in \mathbb{Z}_p^{k+1}$ such that $\mathbf{d}^\perp \mathbf{B}' = \mathbf{0}$. We have $\mathsf{H}_2 \approx_c \mathsf{H}_1$, [KW15] argued this follows from the MDDH assumption for $[\mathbf{A}']_1$ (the details are analogous to our proof of $\mathsf{G}_{5,\ell} \approx_c \mathsf{G}_{5,\ell-1}$ in Section 3.2), even if $\mathbf{M}$ is public, this argument can still hold.

– H_3: Identical to H_2 except that we replace $\mathbf{K}'$ with $\mathbf{K}' + \mathbf{u}\mathbf{d}^\perp$, where $\mathbf{u} \leftarrow \mathbb{Z}_p^{3k+2}$. We have $\mathsf{H}_3 \approx_s \mathsf{H}_2$ and the advantage in H_3 is negligible. [KW15] argued this follows from

- $\mathbf{C}' = (\mathbf{K}' + \mathbf{u}\mathbf{d}^\perp)\mathbf{B}' = \mathbf{K}'\mathbf{B}'$ completely hides $\mathbf{u}$;
- $\mathbf{P} = \mathbf{M}^\top(\mathbf{K} + \mathbf{u}\mathbf{d}^\perp)$ leaks $\mathbf{M}^\top\mathbf{u}$;
- $\mathbf{D}^\top(\mathbf{K}' + \mathbf{u}\mathbf{d}^\perp) + \mathbf{v}\mathbf{d}^\perp = \mathbf{D}^\top\mathbf{K}' + (\mathbf{v} + \mathbf{u})\mathbf{d}^\perp$ in $\widetilde{\pi}_0$ completely hides $\mathbf{u}$.

Since $\mathbf{M} \in \mathbb{Z}_p^{(3k+2)\times(2k+1)}$, even if $\mathbf{M}$ is public, the probability that adversary can recover the correct $\mathbf{u}$ from $\mathbf{M}^\top\mathbf{u}$ is at most $1/p^{2k+1}$, which is negligible in λ.

C Lemmata

Let $\mathsf{Adv}^{\mathsf{xxx}}_{\mathcal{A}}(\lambda)$ be the advantage of $\mathcal{A}$ in $\mathsf{G}_{\mathsf{xxx}}$ defined in Section 3, we present all lemmata and their proofs.

Lemma 3. *($\mathsf{G}_0 \equiv \mathsf{G}_1$). For any adversary $\mathcal{A}$, we have $\mathsf{Adv}^0_{\mathcal{A}}(\lambda) = \mathsf{Adv}^1_{\mathcal{A}}(\lambda)$.*

Proof. Observe that the only difference between game G_0 and G_1 is that we replace π_i in G_0 with $\widetilde{\pi}_i$ in G_1 for all $i \in [L]$ and all $(\mathsf{pk}_i, \mathsf{sk}_i) \in D_i$, where

– $\pi_i \leftarrow \mathsf{LPrv}(\mathsf{crs}_i, [\mathbf{M}_i]_1, \mathbf{U}_i)$
– $\widetilde{\pi}_i \leftarrow \mathsf{LSim}(\mathsf{crs}_i, \mathsf{td}_i, [\mathbf{M}_i]_1)$

here, we have $(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1)$, $\mathbf{A}_i = \binom{\mathbf{A}}{\mathbf{R}_i} \leftarrow \mathbb{Z}_p^{(3k+2)\times(2k+1)}$ and $\mathbf{M}_i = \binom{\mathbf{A}\mathbf{U}_i}{\mathbf{R}_i\mathbf{U}_i}$. The lemma follows from the perfect zero-knowledge of Π which ensures that $\pi_i \equiv \widetilde{\pi}_i$. □

Lemma 4. *($\mathsf{G}_1 \equiv \mathsf{G}_2$). For any adversary $\mathcal{A}$, we have $\mathsf{Adv}^1_{\mathcal{A}}(\lambda) = \mathsf{Adv}^2_{\mathcal{A}}(\lambda)$.*

Proof. Observe that the only difference between G_1 and G_2 is that the challenger samples $\mathbf{s}$ in advance and replaces the $\mathbf{R}$ in G_1 with

$$\widehat{\mathbf{R}}_i = \widetilde{\mathbf{R}}_i \begin{pmatrix} \mathbf{s}\mathbf{A} \\ \mathbf{I}_{2k+1} \end{pmatrix}, \qquad \widetilde{\mathbf{R}} \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)}$$

in G_2. This follows from the following statistical argument: For all $\mathbf{s} \in \mathbb{Z}_p^{1\times k}$ and $\mathbf{A} \in \mathbb{Z}_p^{k\times(2k+1)}$, we have

$$\mathbf{R}_i \equiv \widetilde{\mathbf{R}}_i \begin{pmatrix} \mathbf{s}\mathbf{A} \\ \mathbf{I}_{2k+1} \end{pmatrix}$$

when $\mathbf{R}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+1)}$ and $\widetilde{\mathbf{R}}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)}$. This is justified by the fact that $\binom{\mathbf{s}\mathbf{A}}{\mathbf{I}_{2k+1}}$ is column full-rank. This readily proves the lemma. □

Lemma 5. *($\mathsf{G}_2 \approx_c \mathsf{G}_3$). For any adversary $\mathcal{A}$, there exist algorithm $\mathcal{B}_1$ such that $\mathsf{Time}(\mathcal{B}_1) \approx \mathsf{Time}(\mathcal{A})$ and*

$$|\mathsf{Adv}^3_{\mathcal{A}}(\lambda) - \mathsf{Adv}^2_{\mathcal{A}}(\lambda)| \le L \cdot \mathsf{Adv}^{USS}_{\mathcal{B}_1}(\lambda) + \mathsf{negl}(\lambda).$$

Proof. Define events $\mathsf{Bad}_1, \ldots, \mathsf{Bad}_L$ in G_2 and G_3 as follows:

– Bad_i, $i \in [L]$: it holds that $\mathcal{D}_i[\mathsf{pk}_i^*] = \perp$ and $\mathbf{M}_i^* \notin \mathsf{span}(\mathbf{A}_i)$ where $\mathsf{pk}_i^* = \left([\mathbf{T}_i^*, \mathbf{Q}_i^*]_1, \{[\mathbf{H}_{i,j}^*]_2\}_{j\in[L]\setminus\{i\}}, \pi_i^*\right)$, and $\mathbf{A}_i = \begin{pmatrix}\mathbf{A}\\ \mathbf{R}_i\end{pmatrix}$, $\mathbf{M}_i^* = \begin{pmatrix}\mathbf{T}_i^*\\ \mathbf{Q}_i^*\end{pmatrix}$.

Observe that G_3 and G_2 are identical except that $\mathsf{Bad}_1 \vee \cdots \vee \mathsf{Bad}_L$ happens. By the differential lemma and union bound, for any adversary $\mathcal{A}$, we have

$$|\mathsf{Adv}^3_{\mathcal{A}}(\lambda) - \mathsf{Adv}^2_{\mathcal{A}}(\lambda)| \le \sum_{i^*\in[L]} \Pr[\mathsf{Bad}_{i^*}].$$

It remains to bound $\Pr[\mathsf{Bad}_{i^*}]$ and show that it is negligible. This follows from the unbounded simulation soundness of Π defined in Section 2.4, guessing $i^* \leftarrow [L]$, on input $\mathbf{A}_{i^*} = \left(\begin{smallmatrix}\mathbf{A}\\ \widehat{\mathbf{R}}_{i^*}\end{smallmatrix}\right)$, crs_{i^*}, and having access to oracle $\mathsf{LSim}(\mathsf{crs}_{i^*}, \mathsf{td}_{i^*}, \cdot)$, where $(\mathsf{crs}_{i^*}, \mathsf{td}_{i^*}) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_{i^*}]_1)$, the algorithms $\mathcal{B}_1$ works as follow:

(Setup) Sample

$$\mathbf{s} \leftarrow \mathbb{Z}_p^{1\times k},\ \mathbf{B} \leftarrow \mathbb{Z}_p^{(k+1)\times k},\ \mathbf{k}^\top \leftarrow \mathbb{Z}_p^{2k+1}.$$

Compute

$$[\mathbf{A}\mathbf{k}^\top]_T$$

using $\mathbf{k}$ we sampled and $\mathbf{A}$ from the input. Sample $\widetilde{\mathbf{R}}_{i^*}$ such that:

$$\widehat{\mathbf{R}}_{i^*} = \widetilde{\mathbf{R}}_{i^*}\begin{pmatrix}\mathbf{s}\mathbf{A}\\ \mathbf{I}_{2k+1}\end{pmatrix},$$

here using $\mathbf{A}, \widehat{\mathbf{R}}_{i^*}$ from the input and $\mathbf{s}$ we sampled. For all $i \in [L]\setminus\{i^*\}$, sample $\widetilde{\mathbf{R}}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)}$ and compute

$$\widehat{\mathbf{R}}_i = \widetilde{\mathbf{R}}_i\begin{pmatrix}\mathbf{s}\mathbf{A}\\ \mathbf{I}_{2k+1}\end{pmatrix},$$

here using $\mathbf{A}$ from the input and $\mathbf{s}, \widetilde{\mathbf{R}}_i$ we sampled. For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)n},\ \mathbf{r}_i^\top \leftarrow \mathbb{Z}_p^k.$$

and compute $[\mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1$ from $\mathbf{V}_i, \mathbf{W}_i$ we sampled and $\mathbf{A}$ from the input. For all $i \in [L]\setminus\{i^*\}$, using $\mathbf{A}$ from the input and $\widehat{\mathbf{R}}_i$ we have computed, run

$$(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1) \quad \text{where} \quad \mathbf{A}_i = \begin{pmatrix}\mathbf{A}\\ \widehat{\mathbf{R}}_i\end{pmatrix}.$$

Fetch crs_{i^*} from the input and all remaining terms in crs do not involve $\mathbf{A}$ and can be simulated honestly. Output:

$$\mathsf{crs} = \begin{pmatrix}[\mathbf{A}]_1,\ [\mathbf{A}\mathbf{k}^\top]_T,\ \left\{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\right\}_{i\in[L]}\\ \left\{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\right\}_{j\in[L]}\\ \left\{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\right\}_{j\in[L],i\in[L]\setminus\{j\}}\end{pmatrix}.$$

(Query) For all $i \in [L]\setminus\{i^*\}$ and each $(\mathsf{pk}_i, \mathsf{sk}_i) \in D_i$ is generated as:

$$\mathsf{pk}_i = \big([\underbrace{\mathbf{A}\mathbf{U}_i}_{\mathbf{T}_i}, \underbrace{\widehat{\mathbf{R}}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{\underbrace{[\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top]_2}_{\mathbf{h}_{i,j}}\}_{j\in[L]\setminus\{i\}}, \widetilde{\pi}_i\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

with $\mathbf{U}_i$, $[\mathbf{T}_i]_1$ is computed using $\mathbf{A}$ from the input, $[\mathbf{Q}_i]_1$ is computed using $\widehat{\mathbf{R}}_i$ we have computed; $[\mathbf{h}_{i,j}]_2$ is computed from $[\mathbf{B}\mathbf{r}_j^\top]_2$ in crs we have simulated, and

$$\widetilde{\pi}_i \leftarrow \mathsf{LSim}(\mathsf{crs}_i, \mathsf{td}_i, [\mathbf{M}_i]_1) \quad \text{where} \quad [\mathbf{M}_i]_1 = \begin{bmatrix}\mathbf{T}_i\\ \mathbf{Q}_i\end{bmatrix}_1$$

can be computed using $[\mathbf{T}_i, \mathbf{Q}_i]_1$ we have computed and crs_i, td_i we have generated. And each $(\mathsf{pk}_{i^*}, \mathsf{sk}_{i^*}) \in D_{i^*}$ is generated as:

$$\mathsf{pk}_{i^*} = \big([\underbrace{\mathbf{A}\mathbf{U}_{i^*}}_{\mathbf{T}_{i^*}}, \underbrace{\widehat{\mathbf{R}}_{i^*}\mathbf{U}_{i^*}}_{\mathbf{Q}_{i^*}}]_1, \{\underbrace{[\mathbf{U}_{i^*}\mathbf{B}\mathbf{r}_j^\top]_2}_{\mathbf{h}_{i^*,j}}\}_{j\in[L]\setminus\{i^*\}}, \widetilde{\pi}_{i^*}\big) \quad \text{and} \quad \mathsf{sk}_{i^*} = \mathbf{U}_{i^*}.$$

with $\mathbf{U}_{i^*}$, $[\mathbf{T}_{i^*}, \mathbf{Q}_{i^*}]_1$ are computed using $\mathbf{A}$ and $\widehat{\mathbf{R}}_{i^*}$ from the input; $[\mathbf{h}_{i^*,j}]_2$ is computed from $[\mathbf{B}\mathbf{r}_j^\top]_2$ in crs we have simulated and $\widetilde{\pi}_{i^*}$ is obtained by query $[\mathbf{M}_{i^*}]_1 = \begin{bmatrix}\mathbf{T}_{i^*}\\ \mathbf{Q}_{i^*}\end{bmatrix}_1$ to oracle: $\mathsf{LSim}(\mathsf{crs}_{i^*}, \mathsf{td}_{i^*}, \cdot)$.

(Challenge) On input the challenge $(x^*, (\mathsf{m}_0^*, \mathsf{m}_1^*), (\mathsf{pk}_i^*, y_i^*)_{i\in[L]})$, if $\mathcal{D}_{i^*}[\mathsf{pk}_{i^*}^*] \neq \perp$, $\mathcal{B}_1$ halts and outputs $\perp$; otherwise, $\mathcal{B}_1$ do the following checks for all $i \in [L] \setminus \{i^*\}$:

- When $\mathcal{D}_i[\mathsf{pk}_i^*] = \perp$, check: $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i^*) \stackrel{?}{=} 1 \wedge P(x^*, y_i^*) \stackrel{?}{=} 0$, abort if not.
- When $(i, \mathsf{pk}_i^*) \in C$, check $P(x^*, y_i^*) \stackrel{?}{=} 0$, abort if not.

parse $\mathsf{pk}_i^* = \big([\mathbf{T}_i^*, \mathbf{Q}_i^*]_1, \{[\mathbf{h}_{i,j}^*]_2\}_{j\in[L]\setminus\{i\}}, \pi_i^*\big)$; using $\mathbf{s}$, $\widetilde{\mathbf{R}}_i$ we have sampled, $[\mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1, [\mathbf{A}\mathbf{k}^\top]_T$ in crs we have simulated, return the challenge ciphertext with secret bit b as follow:

$$\mathsf{ct}_{x^*} = \Bigg([\underbrace{\mathbf{s}\mathbf{A}}_{\mathbf{c}_0^*}]_1, \underbrace{\Bigg[\sum_{i\in[L]}((\mathbf{s}\mathbf{A}\mathbf{V}_i + \mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*)(\mathbf{a}_{y_i^*}\otimes\mathbf{I}_{k+1}) + \mathbf{s}\mathbf{A}\mathbf{W}_i(\mathbf{K}_{y_i^*}\otimes\mathbf{I}_{k+1}))\Bigg]_1}_{\mathbf{c}_1^*}, \underbrace{\Bigg[\sum_{i\in[L]}\mathbf{s}\mathbf{A}\mathbf{W}_i(\mathbf{C}_{x^*}\otimes\mathbf{I}_{k+1})\Bigg]_1}_{\mathbf{c}_2^*}, \underbrace{[\mathbf{s}\mathbf{A}\mathbf{k}^\top]_T \cdot \mathsf{m}_b^*}_{C^*}\Bigg).$$

With above simulation, we can observe that for all $i^* \in [L]$, we have $\Pr[\mathsf{Bad}_{i^*}] \leq \mathsf{Adv}^{\mathrm{USS}}_{\mathcal{B}_1} + \mathsf{negl}(\lambda)$. So, we have

$$|\mathsf{Adv}^3_{\mathcal{A}}(\lambda) - \mathsf{Adv}^2_{\mathcal{A}}(\lambda)| \leq \sum_{i\in[L]}\Pr[\mathsf{Bad}_i] \leq L\cdot\mathsf{Adv}^{\mathrm{USS}}_{\mathcal{B}_1} + \mathsf{negl}(\lambda).$$

This proves the lemma. □

Lemma 6. *($\mathsf{G}_3 \approx_c \mathsf{G}_4$). For any adversary $\mathcal{A}$, there exist algorithm $\mathcal{B}_2$ such that* $\mathsf{Time}(\mathcal{B}_2) \approx \mathsf{Time}(\mathcal{A})$ *and*

$$|\mathsf{Adv}^3_{\mathcal{A}}(\lambda) - \mathsf{Adv}^4_{\mathcal{A}}| \leq \mathsf{Adv}^{\mathrm{MDDH}}_{\mathcal{B}_2}(\lambda) + \mathsf{negl}(\lambda).$$

Proof. This follows from the $(k, 2k+1, 1)$-MDDH assumption:

$$([\mathbf{A}]_1, [\mathbf{s}\mathbf{A}]_1) \approx_c ([\mathbf{A}]_1, [\mathbf{c}]_1)$$

where $\mathbf{A} \leftarrow \mathbb{Z}_p^{k\times(2k+1)}$, $\mathbf{s} \leftarrow \mathbb{Z}_p^{1\times k}$ and $\mathbf{c} \leftarrow \mathbb{Z}_p^{1\times(2k+1)}$. On input $[\mathbf{A}]_1, [\mathbf{t}]_1$ where $\mathbf{t} = \mathbf{s}\mathbf{A}$ or $\mathbf{t} = \mathbf{c}$, the algorithm works as follow:

(Setup) Sample

$$\mathbf{B} \leftarrow \mathbb{Z}_p^{(k+1)\times k},\ \mathbf{k}^\top \leftarrow \mathbb{Z}_p^{1\times(2k+1)}.$$

Compute $[\mathbf{A}\mathbf{k}^\top]_T$ from $\mathbf{k}$ we sampled and $[\mathbf{A}]_1$ from the input. For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)n},\ \widetilde{\mathbf{R}}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)},\ \mathbf{r}_i \leftarrow \mathbb{Z}_p^{1\times k}$$

and compute

$$[\widehat{\mathbf{R}}_i]_1 = \left[\widetilde{\mathbf{R}}_i\begin{pmatrix}\mathbf{t}\\ \mathbf{I}_{2k+1}\end{pmatrix}\right]_1 \quad \text{and} \quad [\mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1$$

from $\widetilde{\mathbf{R}}_i, \mathbf{V}_i, \mathbf{W}_i$ we sampled and $[\mathbf{A}, \mathbf{t}]_1$ from the input. For all $i \in [L]$, using $[\widehat{\mathbf{R}}_i]_1$ we have computed and $[\mathbf{A}]_1$ from the input, compute

$$[\mathbf{A}_i]_1 = \begin{bmatrix}\mathbf{A}\\ \widehat{\mathbf{R}}_i\end{bmatrix}_1$$

and run

$$(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1).$$

All remaining terms in crs do not involve $\mathbf{A}$ and can be simulated honestly. Output:

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1,\ [\mathbf{A}\mathbf{k}^\top]_T,\ \left\{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\right\}_{i\in[L]} \\ \left\{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\right\}_{j\in[L]} \\ \left\{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\right\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}.$$

(Query) For all $i \in [L]$ and each $(\mathsf{pk}_i, \mathsf{sk}_i) \in D_i$ is generated honestly as:

$$\mathsf{pk}_i = \big([\underbrace{\mathbf{A}\mathbf{U}_i}_{\mathbf{T}_i}, \underbrace{\widehat{\mathbf{R}}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{\underbrace{[\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top]_2}_{\mathbf{h}_{i,j}}\}_{j\in[L]\setminus\{i\}}, \widetilde{\pi}_i\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

with $\mathbf{U}_i$, $[\mathbf{T}_i]_1$ is computed using $[\mathbf{A}]_1$ from the input; $[\mathbf{Q}_i]_1$ is computed using $[\widehat{\mathbf{R}}_i]_1$ we have computed; $[\mathbf{h}_{i,j}]_2$ is computed from $[\mathbf{B}\mathbf{r}_j^\top]_2$ in crs we have simulated, and

$$\widetilde{\pi}_i \leftarrow \mathsf{LSim}(\mathsf{crs}_i, \mathsf{td}_i, [\mathbf{M}_i]_1), \quad \text{where} \quad [\mathbf{M}_i]_1 = \begin{bmatrix} \mathbf{T}_i \\ \mathbf{Q}_i \end{bmatrix}_1$$

can be computed using $\mathsf{crs}_i, \mathsf{td}_i$ we have generated and $[\mathbf{T}_i, \mathbf{Q}_i]_1$ we have computed.

(Challenge) On input the challenge $(x^*, (\mathsf{m}_0^*, \mathsf{m}_1^*), (\mathsf{pk}_i^*, y_i^*)_{i\in[L]})$, $\mathcal{B}_1$ do the following checks for all $i \in [L]$:

- When $\mathcal{D}_i[\mathsf{pk}_i^*] = \perp$, check: $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i^*) \stackrel{?}{=} 1 \wedge P(x^*, y_i^*) \stackrel{?}{=} 0$, abort if not.
- When $(i, \mathsf{pk}_i^*) \in C$, check $P(x^*, y_i^*) \stackrel{?}{=} 0$, abort if not.

parse $\mathsf{pk}_i^* = \big([\mathbf{T}_i^*, \mathbf{Q}_i^*]_1, \{[\mathbf{h}_{i,j}^*]_2\}_{j\in[L]\setminus\{i\}}, \pi_i^*\big)$, using $\widetilde{\mathbf{R}}_i, \mathbf{V}_i, \mathbf{W}_i, \mathbf{k}$ we have sampled, using $[\mathbf{t}]_1$ from the input, return the challenge ciphertext with secret bit $b \in \{0,1\}$ as follow:

$$\mathsf{ct}_{x^*} = \Bigg([\underbrace{\mathbf{t}}_{\mathbf{c}_0^*}]_1, \underbrace{\left[\sum_{i\in[L]} ((\mathbf{t}\mathbf{V}_i + \mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*)(\mathbf{a}_{y_i^*} \otimes \mathbf{I}_{k+1}) + \mathbf{t}\mathbf{W}_i(\mathbf{K}_{y_i^*} \otimes \mathbf{I}_{k+1}))\right]_1}_{\mathbf{c}_1^*}, \underbrace{\left[\sum_{i\in[L]} \mathbf{t}\mathbf{W}_i(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1})\right]_1}_{\mathbf{c}_2^*}, \underbrace{[\mathbf{t}\mathbf{k}^\top]_T \cdot \mathsf{m}_b^*}_{C^*}\Bigg).$$

Observe that when $\mathbf{t} = \mathbf{c}\mathbf{A}$, the simulation is identical to G_3; when $\mathbf{t} = \mathbf{c}$, the simulation is identical to G_4. This readily proves the lemma. □

Lemma 7. *($\mathsf{G}_{5,L} \approx_s \mathsf{G}_6$). For any adversary $\mathcal{A}$, we have*

$$|\mathsf{Adv}^{5,L}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{6}_{\mathcal{A}}(\lambda)| = 0$$

Proof. First, in the process of simulating crs, we program $\mathbf{k}^\top$ in both $\mathsf{G}_{5,L}$ and G_6 as follow:

$$\mathbf{k}^\top \mapsto \mathbf{k}^\top - \mathbf{c}^\perp\alpha$$

where $\mathbf{k}^\top \leftarrow \mathbb{Z}_p^{2k+1}, \alpha \leftarrow \mathbb{Z}_p$ Due to the fact that $\mathbf{A}\mathbf{c}^\perp = \mathbf{0}$, We can simulate the crs as follow:

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1,\ [\mathbf{A}\mathbf{k}^\top]_T,\ \left\{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\right\}_{i\in[L]} \\ \left\{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\right\}_{j\in[L]} \\ \left\{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\right\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}.$$

And observe that for secret bit b, the challenge ciphertext C^* in $\mathsf{G}_{5,L}$ is:

$$C^* = [\mathbf{c}\mathbf{k}^\top - \alpha]_T \cdot \mathsf{m}_b^*$$

this follows form the fact that $\mathbf{c}\mathbf{c}^\perp = 1$. After substitute $\mathbf{k}^\top \mapsto \mathbf{k}^\top - \mathbf{c}^\perp\alpha$, we can observe α only correlate to C^* and $[\alpha]_T$ is uniformly distribute over $\mathbb{G}_T$. It implies that the distribution of C^* is identical to a random coin in $\mathbb{G}_T$, just like in G_6. This readily prove the lemma. □

Lemma 8. *($\mathsf{G}_{5,\ell-1,0} \approx_c \mathsf{G}_{5,\ell-1,1}$). For any adversary $\mathcal{A}$, there exist algorithm $\mathcal{B}_2$ such that* $\mathsf{Time}(\mathcal{B}_2) \approx \mathsf{Time}(\mathcal{A})$ *and*

$$|\mathsf{Adv}^{5,\ell-1,0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{5,\ell-1,1}_{\mathcal{A}}| \leq \mathsf{Adv}^{\mathrm{MDDH}}_{\mathcal{B}_2}(\lambda) + \mathsf{negl}(\lambda).$$

Proof. This follows from the $(k, k+1, 1)$-MDDH assumption:

$$[\mathbf{B}]_2, [\mathbf{B}\mathbf{r}_\ell^\top]_2 \approx_c [\mathbf{B}]_2, [\mathbf{d}_\ell^\top]_2$$

where $\mathbf{B} \leftarrow \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{r}_\ell \leftarrow \mathbb{Z}_p^{1\times k}$ and $\mathbf{d}_\ell \leftarrow \mathbb{Z}_p^{1\times(k+1)}$. On input $[\mathbf{B}]_2, [\mathbf{t}^\top]_2$ where $\mathbf{t}^\top = \mathbf{B}\mathbf{r}_\ell^\top$ or $\mathbf{t}^\top = \mathbf{d}_\ell^\top$, the algorithm works as follow:

(Setup) Sample

$$\mathbf{A} \leftarrow \mathbb{Z}_p^{k\times(2k+1)},\ \mathbf{k} \leftarrow \mathbb{Z}_p^{1\times(2k+1)},\ \mathbf{c} \leftarrow \mathbb{Z}_p^{1\times(2k+1)},\ \alpha \leftarrow \mathbb{Z}_p.$$

Compute

$$[\mathbf{A}\mathbf{k}^\top]_T \quad \text{and} \quad \mathbf{c}^\perp \in \mathbb{Z}_p^{2k+1},$$

such that $\mathbf{c}\mathbf{c}^\perp = 1$ and $\mathbf{A}\mathbf{c}^\perp = 0$, with $\mathbf{A}, \mathbf{k}, \mathbf{c}$ we sampled. For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)n},\ \widetilde{\mathbf{R}}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)},$$

and compute

$$[\widehat{\mathbf{R}}_i]_1 = \left[\widetilde{\mathbf{R}}_i \begin{pmatrix} \mathbf{c} \\ \mathbf{I}_{2k+1} \end{pmatrix}\right]_1,\ [\mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1$$

from $\widetilde{\mathbf{R}}_i, \mathbf{V}_i, \mathbf{W}_i, \mathbf{A}, \mathbf{c}$ we sampled. For all $j \in [L] \setminus \{\ell\}$, sample

$$\mathbf{r}_j \leftarrow \mathbb{Z}_p^{1\times k}.$$

For all $j \in [\ell - 1]$, compute

$$[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2,$$

using $\mathbf{r}_j, \mathbf{V}_j, \mathbf{k}, \alpha$ we sampled, $\mathbf{c}^\perp$ we computed and $[\mathbf{B}]_2$ from the input. And compute

$$[\mathbf{t}^\top, \mathbf{V}_\ell\mathbf{t}^\top + \mathbf{k}^\top]_2,$$

using $\mathbf{V}_\ell, \mathbf{k}$ we sampled and $[\mathbf{t}^\top]_2$ from the input. For all $j \in [L] \setminus [\ell]$, compute

$$[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2,$$

using $\mathbf{r}_j, \mathbf{V}_j, \mathbf{k}$ we sampled and $[\mathbf{B}]_2$ from the input. For all $j \in [L] \setminus \{\ell\}$ and all $i \in [L] \setminus \{j\}$, compute

$$[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2,$$

using $\mathbf{V}_i, \mathbf{W}_i, \mathbf{r}_j$ we sampled and $[\mathbf{B}]_2$ from the input. For all $i \in [L] \setminus \{\ell\}$, compute

$$[\mathbf{V}_i\mathbf{t}^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{t}^\top)]_2,$$

using $\mathbf{V}_i, \mathbf{W}_i$ we sampled and $[\mathbf{t}^\top]_2$ from the input. For all $i \in [L]$, using $[\widehat{\mathbf{R}}_i]_1$ we have computed and $[\mathbf{A}]_1$ we sampled, run

$$(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1), \quad \text{where} \quad [\mathbf{A}_i]_1 = \begin{bmatrix} \mathbf{A} \\ \widehat{\mathbf{R}}_i \end{bmatrix}_1$$

Output:

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1, [\mathbf{A}\mathbf{k}^\top]_T, \left\{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\right\}_{i\in[L]} \\ \left\{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2\right\}_{j\in[\ell-1]}, [\mathbf{t}^\top, \mathbf{V}_\ell\mathbf{t}^\top + \mathbf{k}^\top]_2, \left\{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\right\}_{j\in[L]\setminus[\ell]} \\ \left\{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\right\}_{j\in[L]\setminus\{\ell\}, i\in[L]\setminus\{j\}}, \left\{[\mathbf{V}_i\mathbf{t}^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{t}^\top)]_2\right\}_{i\in[L]\setminus\{\ell\}} \end{pmatrix}.$$

(Query) For all $i \in [L]$ and each $(\mathsf{pk}_i, \mathsf{sk}_i) \in D_i$ is generated as:

$$(\mathsf{pk}_i, \mathsf{sk}_i) = \begin{cases} \big([\overbrace{\mathbf{AU}_i}^{\mathbf{T}_i}, \overbrace{\widehat{\mathbf{R}}_i\mathbf{U}_i}^{\mathbf{Q}_i}]_1, \{[\overbrace{\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top}^{\mathbf{h}_{i,j}}]_2\}_{j\in[L]\setminus\{i,\ell\}}, [\overbrace{\mathbf{U}_i\mathbf{t}^\top}^{\mathbf{h}_{i,\ell}}]_2, \widetilde{\pi}_i\big), \mathbf{U}_i & \text{if } i \neq \ell \\ \big([\underbrace{\mathbf{AU}_\ell}_{\mathbf{T}_\ell}, \underbrace{\widehat{\mathbf{R}}_\ell\mathbf{U}_\ell}_{\mathbf{Q}_\ell}]_1, \{[\underbrace{\mathbf{U}_\ell\mathbf{B}\mathbf{r}_j^\top}_{\mathbf{h}_{\ell,j}}]_2\}_{j\in[L]\setminus\{\ell\}}, \widetilde{\pi}_\ell\big), \mathbf{U}_\ell & \text{if } i = \ell \end{cases}$$

with $\mathbf{U}_i$, $[\mathbf{T}_i, \mathbf{Q}_i]_1$ are computed from $[\mathbf{A}, \widehat{\mathbf{R}}_i]_1$ in crs we have simulated; $[\mathbf{h}_{i,\ell}]_2$ is computed using $[\mathbf{t}^\top]_2$ from the input; remaining $[\mathbf{h}_{i,j}]_2$ is computed using $[\mathbf{B}]_2$ from the input and $\mathbf{r}_j^\top$ we have sampled; and

$$\widetilde{\pi}_i \leftarrow \mathsf{LSim}(\mathsf{crs}_i, \mathsf{td}_i, [\mathbf{M}_i]_1), \quad \text{where} \quad [\mathbf{M}_i]_1 = \begin{bmatrix} \mathbf{T}_i \\ \mathbf{Q}_i \end{bmatrix}_1$$

can be computed using $\mathsf{crs}_i, \mathsf{td}_i$ we have generated and $[\mathbf{T}_i, \mathbf{Q}_i]_1$ we have computed.

(Challenge) On input the challenge $(x^*, (\mathsf{m}_0^*, \mathsf{m}_1^*), (\mathsf{pk}_i^*, y_i^*)_{i\in[L]})$, $\mathcal{B}_2$ do the following checks for all $i \in [L]$:

- When $\mathcal{D}_i[\mathsf{pk}_i^*] = \perp$, check: $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i^*) \stackrel{?}{=} 1 \wedge P(x^*, y_i^*) \stackrel{?}{=} 0$, abort if not.
- When $(i, \mathsf{pk}_i^*) \in C$, check $P(x^*, y_i^*) \stackrel{?}{=} 0$, abort if not.

parse $\mathsf{pk}_i^* = \big([\mathbf{T}_i^*, \mathbf{Q}_i^*]_1, \{[\mathbf{h}_{i,j}^*]_2\}_{j\in[L]\setminus\{i\}}, \pi_i^*\big)$, using $\mathbf{c}, \widetilde{\mathbf{R}}_i, \mathbf{V}_i, \mathbf{W}_i, \mathbf{k}$ we have sampled, return the challenge ciphertext with secret bit $b \in \{0,1\}$ as follow:

$$\mathsf{ct}_{x^*} = \Bigg([\underbrace{\mathbf{c}}_{\mathbf{c}_0^*}]_1, \underbrace{\left[\sum_{i\in[L]} ((\mathbf{c}\mathbf{V}_i + \mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*)(\mathbf{a}_{y_i^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_i(\mathbf{K}_{y_i^*} \otimes \mathbf{I}_{k+1}))\right]_1}_{\mathbf{c}_1^*}, \underbrace{\left[\sum_{i\in[L]} \mathbf{c}\mathbf{W}_i(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1})\right]_1}_{\mathbf{c}_2^*}, \underbrace{[\mathbf{c}\mathbf{k}^\top]_T \cdot \mathsf{m}_b^*}_{C^*}\Bigg).$$

Observe that when $\mathbf{t}^\top = \mathbf{B}\mathbf{r}_\ell^\top$, the simulation is identical to $\mathsf{G}_{5,\ell-1,0}$; when $\mathbf{t}^\top = \mathbf{d}_\ell^\top$, the simulation is identical to $\mathsf{G}_{5,\ell-1,1}$. This readily proves the lemma. □

Lemma 9. *($\mathsf{G}_{5,\ell-1,1} \approx_c \mathsf{G}_{5,\ell-1,2}$). For any adversary $\mathcal{A}$, there exist algorithm $\mathcal{B}_2$ such that* $\mathsf{Time}(\mathcal{B}_2) \approx \mathsf{Time}(\mathcal{A})$ *and*

$$|\mathsf{Adv}_{\mathcal{A}}^{5,\ell-1,1}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{5,\ell-1,2}(\lambda)| \leq 2Q \cdot \mathsf{Adv}_{\mathcal{B}_2}^{\mathrm{MDDH}} + \mathsf{negl}(\lambda)$$

where Q is the maximum number of queries on a slot made by $\mathcal{A}$.

Proof. Recall that in these two games, the crs are in the following form:

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1, [\mathbf{A}\mathbf{k}^\top]_T, \{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\}_{i\in[L]} \\ \{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2\}_{j\in[\ell-1]}, [\mathbf{d}_\ell^\top, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{k}^\top + \boxed{b\mathbf{c}^\perp\alpha}]_2, \{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\}_{j\in[L]\setminus[\ell]} \\ \{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\}_{j\in[L]\setminus\{\ell\}, i\in[L]\setminus\{j\}}, \{[\mathbf{V}_i\mathbf{d}_\ell^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{d}_\ell^\top)]_2\}_{i\in[L]\setminus\{\ell\}} \end{pmatrix}$$

where $b = 0$ in $\mathsf{G}_{5,\ell-1,1}$, and $b = 1$ in $\mathsf{G}_{5,\ell-1,2}$. For all $i \in [L]$ and for each $\mathsf{pk}_i \in D_i$, we have

$$(\mathsf{pk}_i, \mathsf{sk}_i) = \begin{cases} \big([\mathbf{A}\mathbf{U}_i, \widehat{\mathbf{R}}_i\mathbf{U}_i]_1, \{[\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{i,\ell\}}, [\mathbf{U}_i\mathbf{d}_\ell^\top]_2, \widetilde{\pi}_i\big), \mathbf{U}_i & \text{if } i \neq \ell \\ \big([\mathbf{A}\mathbf{U}_\ell, \widehat{\mathbf{R}}_\ell\mathbf{U}_\ell]_1, \{[\mathbf{U}_\ell\mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{\ell\}}, \widetilde{\pi}_\ell\big), \mathbf{U}_\ell & \text{if } i = \ell \end{cases}$$

And we recall $\mathbf{c}_1^*$ and $\mathbf{c}_2^*$ in the following terms:

$$\mathbf{c}_1^* = ((\mathbf{c}\mathbf{V}_\ell + \mathbf{e}_1\widetilde{\mathbf{R}}_\ell^{-1}\mathbf{Q}_\ell^*)(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1})) + \textstyle\sum_{i\in[L]\setminus\{\ell\}}((\mathbf{c}\mathbf{V}_i + \mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}_i^*)(\mathbf{a}_{y_i^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_i(\mathbf{K}_{y_i^*} \otimes \mathbf{I}_{k+1}))$$
$$\mathbf{c}_2^* = \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) + \textstyle\sum_{i\in[L]\setminus\{\ell\}} \mathbf{c}\mathbf{W}_i(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1})$$

And we define $\mathbf{c}^\perp \in \mathbb{Z}_p^{2k+1}$ and $\mathbf{d}^\perp \in \mathbb{Z}_p^{1\times(k+1)}$ such that

$$\mathbf{A}\mathbf{c}^\perp = \mathbf{0}, \quad \mathbf{c}\mathbf{c}^\perp = 1; \quad \mathbf{d}^\perp\mathbf{B} = \mathbf{0}, \quad \mathbf{d}^\perp\mathbf{d}_\ell^\top = 1 \tag{14}$$

On slot ℓ, regarding the challenge public key pk_ℓ^*, we will discuss two cases: (1) pk_ℓ^* is honest, which means that $\mathsf{pk}_\ell^* \in D_\ell \setminus C_\ell$; (2) pk_ℓ^* is corrupted or maliciously generated by the adversary, which means that $\mathsf{pk}_\ell^* \in C_\ell \cup \bar{D}_\ell$.

Honest Case. In this case, we have $\mathsf{pk}^*_\ell = ([\mathbf{AU}^*_\ell, \widetilde{\mathbf{R}}_\ell \mathbf{U}^*_\ell]_1, \{[\mathbf{U}^*_\ell \mathbf{B}\mathbf{r}_j^\top]_2\}_{j\in[L]\setminus\{\ell\}}, \widetilde{\pi}_\ell)$, where the $\mathbf{U}^*_\ell$ is honestly generated by challenger and is hidden from the adversary. And we can write the first term in $\mathbf{c}^*_1$ as follows:

$$(\mathbf{cV}_\ell + \boxed{\mathbf{cU}^*_\ell})(\mathbf{a}_{y^*_\ell} \otimes \mathbf{I}_{k+1}) + \mathbf{cW}_\ell(\mathbf{K}_{y^*_\ell} \otimes \mathbf{I}_{k+1})$$

which means that the we can directly simulate $\mathbf{cU}^*_\ell$ and don't need to program $\mathbf{c}$ into $\widehat{\mathbf{R}}_\ell$, thus, we replace $\widehat{\mathbf{R}}_\ell$ in crs with a random $\mathbf{R}_\ell$ analogous to the proof of Lemma 4. To prove the indistinguishability of $\mathsf{G}_{5,\ell-1,1}$ and $\mathsf{G}_{5,\ell-1,2}$ in this case, we firstly recall the related terms which is known by adversary as follows:

$$\begin{aligned}
&\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{AV}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha && \text{// in crs, } \mathsf{pk}_\ell\\
&\mathbf{c}, \mathbf{cV}_\ell + \mathbf{cU}^*_\ell;\ \mathbf{AU}^*_\ell, [\mathbf{R}_\ell\mathbf{U}^*_\ell]_1, \mathbf{U}^*_\ell\mathbf{B} && \text{// in ct}^*\text{. } \mathsf{pk}^*_\ell
\end{aligned}$$

And we establish a series of argument as below:

1. We argue that:

$$\begin{aligned}
&\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{AV}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha\\
&\mathbf{c}, \mathbf{cV}_\ell + \mathbf{cU}^*_\ell;\ \mathbf{AU}^*_\ell, [\mathbf{R}_\ell\mathbf{U}^*_\ell]_1, \mathbf{U}^*_\ell\mathbf{B}\\
\approx_c\ &\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{AV}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha\\
&\mathbf{c}, \mathbf{cV}_\ell + \mathbf{cU}^*_\ell;\ \mathbf{AU}^*_\ell, [\mathbf{R}_\ell\mathbf{U}^*_\ell + \boxed{\widehat{\mathbf{u}}^\top\mathbf{d}^\perp}]_1, \mathbf{U}^*_\ell\mathbf{B}
\end{aligned} \tag{15}$$

where $\widehat{\mathbf{u}} \leftarrow \mathbb{Z}_p^{1\times(2k+2)}$. We define events $\mathsf{Bad}_1, \ldots, \mathsf{Bad}_Q$ in the honest case as follows:

- $\mathsf{Bad}_q, q \in [Q]$: on slot ℓ, adversary honestly chooses the q^{th} pk_ℓ returned to $\mathsf{OGen}(\ell)$ as the challenge public key pk^*_ℓ, and when simulating this pk^*_ℓ, the simulator uses $[\mathbf{Q}^*_\ell]_1$ which is $[\mathbf{R}_\ell\mathbf{U}^*_\ell]_1$ or $[\mathbf{R}_\ell\mathbf{U}^*_\ell + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1$.

Observe that the distributions in our argument will occur when $\mathsf{Bad}_1 \vee \cdots \vee \mathsf{Bad}_Q$, and the advantage between the distributions in our argument is bounded by $\sum_{q\in[L]} \Pr[\mathsf{Bad}_q]$. It remains to bound $\Pr[\mathsf{Bad}_q]$ and show that it is negligible. This follows from Lemma 2 whihc ensure that:

$$\begin{aligned}
&\left(\begin{smallmatrix}\mathbf{A}\\ \mathbf{c}\end{smallmatrix}\right), [\mathbf{R}_\ell]_1, \mathbf{B}, \mathbf{d}^\perp, \left(\begin{smallmatrix}\mathbf{AU}^*_\ell\\ \mathbf{cU}^*_\ell\end{smallmatrix}\right), [\mathbf{R}_\ell\mathbf{U}^*_\ell]_1, && \mathbf{U}^*_\ell\mathbf{B}\\
\approx_c\ &\left(\begin{smallmatrix}\mathbf{A}\\ \mathbf{c}\end{smallmatrix}\right), [\mathbf{R}_\ell]_1, \mathbf{B}, \mathbf{d}^\perp, \left(\begin{smallmatrix}\mathbf{AU}^*_\ell\\ \mathbf{cU}^*_\ell\end{smallmatrix}\right), [\mathbf{R}_\ell\mathbf{U}^*_\ell + \boxed{\widehat{\mathbf{u}}^\top\mathbf{d}^\perp}]_1, && \mathbf{U}^*_\ell\mathbf{B}
\end{aligned}$$

Guessing $q \in [L]$, on input $\mathbf{A}$, $\mathbf{c}$, $[\mathbf{R}_\ell]_1$, $\mathbf{B}$, $\mathbf{d}^\perp$, $\mathbf{T}^*_\ell$, $\widehat{\mathbf{c}}$, $[\mathbf{Q}^*_\ell]_1$ and $\widehat{\mathbf{B}}$ where $\mathbf{T}^*_\ell = \mathbf{AU}^*_\ell$, $\widehat{\mathbf{c}} = \mathbf{cU}^*_\ell$, $\widehat{\mathbf{B}} = \mathbf{U}^*_\ell\mathbf{B}$; $[\mathbf{Q}^*_\ell]_1 = [\mathbf{R}_\ell\mathbf{U}^*_\ell]_1$ or $[\mathbf{Q}^*_\ell]_1 = [\mathbf{R}_\ell\mathbf{U}^*_\ell + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1$, there exist algorithm $\mathcal{B}$ works as follows:

(Setup) Sample

$$\mathbf{k} \leftarrow \mathbb{Z}_p^{1\times(2k+1)},\ \alpha \leftarrow \mathbb{Z}_p,\ \mathbf{d}_\ell \leftarrow \mathbb{Z}_p^{1\times(k+1)}.$$

Compute

$$[\mathbf{Ak}^\top]_T \quad \text{and} \quad \mathbf{c}^\perp \in \mathbb{Z}_p^{2k+1}$$

such that $\mathbf{cc}^\perp = 1$ and $\mathbf{Ac}^\perp = 0$, with $\mathbf{A}$, $\mathbf{c}$ from the input and $\mathbf{k}$ we sampled. For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{(2k+1)\times(k+1)n}$$

and compute

$$[\mathbf{AV}_i, \mathbf{AW}_i]_1$$

using $\mathbf{V}_i$, $\mathbf{W}_i$ we sampled and $\mathbf{A}$ from the input. For all $i \in [L] \setminus \{\ell\}$, sample

$$\widetilde{\mathbf{R}}_i \leftarrow \mathbb{Z}_p^{(2k+2)\times(2k+2)},$$

and compute

$$[\widehat{\mathbf{R}}_i]_1 = \left[\widetilde{\mathbf{R}}_i \begin{pmatrix}\mathbf{c}\\ \mathbf{I}_{2k+1}\end{pmatrix}\right]_1$$

using $\widetilde{\mathbf{R}}_i$ we sampled and $\mathbf{c}$ from the input. For all $j \in [L] \setminus \{\ell\}$, sample

$$\mathbf{r}_j \leftarrow \mathbb{Z}_p^{1\times k}.$$

For all $j \in [\ell - 1]$, compute

$$[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2,$$

using $\mathbf{r}_j, \mathbf{V}_j, \mathbf{k}, \alpha$ we sampled, $\mathbf{c}^\perp$ we computed and $\mathbf{B}$ from the input. With $b = 0$ when simulate $\mathsf{G}_{5,\ell-1,1}$ and $b = 1$ when simulate $\mathsf{G}_{5,\ell-1,2}$, compute

$$[\mathbf{d}_\ell^\top, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{k}^\top + b\mathbf{c}^\perp\alpha]_2,$$

using $\mathbf{d}_\ell, \mathbf{V}_\ell, \mathbf{k}, \alpha$ we sample. For all $j \in [L] \setminus [\ell]$, compute

$$[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2,$$

using $\mathbf{r}_j, \mathbf{V}_j, \mathbf{k}$ we sampled and $\mathbf{B}$ from the input. For all $j \in [L] \setminus \{\ell\}$ and all $i \in [L] \setminus \{j\}$, compute

$$[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2,$$

using $\mathbf{V}_i, \mathbf{W}_i, \mathbf{r}_j$ we sampled and $\mathbf{B}$ from the input. For all $i \in [L] \setminus \{\ell\}$, compute

$$[\mathbf{V}_i\mathbf{d}_\ell^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{d}_\ell^\top)]_2,$$

with $\mathbf{V}_i, \mathbf{W}_i, \mathbf{d}_\ell$ we sampled. For all $i \in [L] \setminus \{\ell\}$, using $[\widehat{\mathbf{R}}_i]_1$ we have computed and $\mathbf{A}$ from the input, run

$$(\mathsf{crs}_i, \mathsf{td}_i) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_i]_1), \quad \text{where} \quad [\mathbf{A}_i]_1 = \begin{bmatrix}\mathbf{A}\\ \widehat{\mathbf{R}}_i\end{bmatrix}_1$$

Using $\mathbf{A}$ and $[\mathbf{R}_\ell]_1$ from the input, run

$$(\mathsf{crs}_\ell, \mathsf{td}_\ell) \leftarrow \mathsf{LGen}(1^\lambda, \mathbb{G}_1, [\mathbf{A}_\ell]_1), \quad \text{where} \quad [\mathbf{A}_\ell]_1 = \begin{bmatrix}\mathbf{A}\\ \mathbf{R}_\ell\end{bmatrix}_1$$

Output:

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{A}]_1, [\mathbf{A}\mathbf{k}^\top]_T, \left\{\mathsf{crs}_i, [\mathbf{A}\mathbf{V}_i, \mathbf{A}\mathbf{W}_i]_1\right\}_{i\in[L]}, \{[\widehat{\mathbf{R}}_i]_1\}_{i\in[L]\setminus\{\ell\}}, [\mathbf{R}_\ell]_1 \\ \left\{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2\right\}_{j\in[\ell-1]}, [\mathbf{d}_\ell^\top, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{k}^\top + b\mathbf{c}^\perp\alpha]_2, \left\{[\mathbf{B}\mathbf{r}_j^\top, \mathbf{V}_j\mathbf{B}\mathbf{r}_j^\top + \mathbf{k}^\top]_2\right\}_{j\in[L]\setminus[\ell]} \\ \left\{[\mathbf{V}_i\mathbf{B}\mathbf{r}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{B}\mathbf{r}_j^\top)]_2\right\}_{j\in[L]\setminus\{\ell\}, i\in[L]\setminus\{j\}}, \left\{[\mathbf{V}_i\mathbf{d}_\ell^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{d}_\ell^\top)]_2\right\}_{i\in[L]\setminus\{\ell\}} \end{pmatrix}.$$

(Query) For all $i \in [L] \setminus \{\ell\}$ and each $(\mathsf{pk}_i, \mathsf{sk}_i) \in D_i$ is generated as:

$$\mathsf{pk}_i = \big([\underbrace{\mathbf{A}\mathbf{U}_i}_{\mathbf{T}_i}, \underbrace{\widehat{\mathbf{R}}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{[\underbrace{\mathbf{U}_i\mathbf{B}\mathbf{r}_j^\top}_{\mathbf{h}_{i,j}}]_2\}_{j\in[L]\setminus\{i,\ell\}}, [\underbrace{\mathbf{U}_i\mathbf{d}_\ell^\top}_{\mathbf{h}_{i,\ell}}]_2, \widetilde{\pi}_i\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

with $\mathbf{U}_i$, $[\mathbf{T}_i]_1$ is computed using $\mathbf{A}$ from the input, $[\mathbf{Q}_i]_1$ is computed using $[\widehat{\mathbf{R}}_i]_1$ we have simulated; $[\mathbf{h}_{i,\ell}]_2$ is computed using $[\mathbf{d}_\ell^\top]_2$ we have simulated, remaining $[\mathbf{h}_{i,j}]_2$ is computed using $\mathbf{B}$ from the input and $\mathbf{r}_j$ we have sampled; and

$$\widetilde{\pi}_i \leftarrow \mathsf{LSim}(\mathsf{crs}_i, \mathsf{td}_i, [\mathbf{M}_i]_1), \quad \text{where} \quad [\mathbf{M}_i]_1 = \begin{bmatrix}\mathbf{T}_i\\ \mathbf{Q}_i\end{bmatrix}_1$$

can be computed using $\mathsf{crs}_i, \mathsf{td}_i$ we have generated and $[\mathbf{T}_i, \mathbf{Q}_i]_1$ we have computed. And on slot ℓ, except for the q^{th}-round query on $\mathsf{OGen}(\ell)$, other $(\mathsf{pk}_\ell, \mathsf{sk}_\ell) \in D_\ell$ is generated as:

$$\mathsf{pk}_\ell = \big([\underbrace{\mathbf{A}\mathbf{U}_\ell}_{\mathbf{T}_\ell}, \underbrace{\mathbf{R}_\ell\mathbf{U}_\ell}_{\mathbf{Q}_\ell}]_1, \{[\underbrace{\mathbf{U}_\ell\mathbf{B}\mathbf{r}_j^\top}_{\mathbf{h}_{\ell,j}}]_2\}_{j\in[L]\setminus\{\ell\}}, \widetilde{\pi}_\ell\big) \quad \text{and} \quad \mathsf{sk}_\ell = \mathbf{U}_\ell$$

with $\mathbf{U}_\ell$ we generated, $[\mathbf{T}_\ell, \mathbf{Q}_\ell]_1$ are computed using $\mathbf{A}$ and $[\mathbf{R}_\ell]$ from the input, $[\mathbf{h}_{\ell,j}]_2$ is computed using $\mathbf{B}$ from the input and $\mathbf{r}_j$ we have sampled, and

$$\widetilde{\pi}_\ell \leftarrow \mathsf{LSim}(\mathsf{crs}_\ell, \mathsf{td}_\ell, [\mathbf{M}_\ell]_1), \quad \text{where} \quad [\mathbf{M}_\ell]_1 = \begin{bmatrix} \mathbf{T}_\ell \\ \mathbf{Q}_\ell \end{bmatrix}_1$$

can be computed using $\mathsf{crs}_\ell, \mathsf{td}_\ell$ we have generated and $[\mathbf{T}_\ell, \mathbf{Q}_\ell]_1$ we have computed. As for the q^{th}-round query on $\mathsf{OGen}(\ell)$, we return the pk^*_ℓ as follows:

$$\mathsf{pk}^*_\ell = ([\mathbf{T}^*_\ell, \mathbf{Q}^*_\ell]_1, \{\underbrace{[\,\widehat{\mathbf{B}}\mathbf{r}_j^\top\,]_2}_{\mathbf{h}^*_{\ell,j}}\}_{j\in[L]\setminus\{\ell\}}, \widetilde{\pi}^*_\ell)$$

with $\mathbf{T}^*_\ell$ and $[\mathbf{Q}^*_\ell]_1$ from the input; and $[\mathbf{h}^*_{\ell,j}]_2$ is computed using $\widehat{\mathbf{B}}$ from the input and $\mathbf{r}_j$ we have sampled, and

$$\widetilde{\pi}^*_\ell \leftarrow \mathsf{LSim}(\mathsf{crs}_\ell, \mathsf{td}_\ell, [\mathbf{M}^*_\ell]_1), \quad \text{where} \quad [\mathbf{M}^*_\ell]_1 = \begin{bmatrix} \mathbf{T}^*_\ell \\ \mathbf{Q}^*_\ell \end{bmatrix}_1$$

can be computed using $\mathsf{crs}_\ell, \mathsf{td}_\ell$ we have generated and $[\mathbf{T}_\ell, \mathbf{Q}_\ell]_1$ we have computed. Note that in Bad_q, the adversary will choose pk^*_ℓ as a challenge public key honestly, and in honest case, we have $\mathsf{pk}^*_\ell \notin C_\ell$, which means that the simulation can still finish even if it doesn't know the $\mathsf{sk}^*_\ell = \mathbf{U}^*_\ell$ from the input.

(Challenge) On input the challenge $(x^*, (\mathsf{m}^*_0, \mathsf{m}^*_1), (\mathsf{pk}^*_i, y^*_i)_{i\in[L]})$, $\mathcal{B}$ do the following checks for all $i \in [L]\setminus\{\ell\}$:

- When $\mathcal{D}_i[\mathsf{pk}^*_i] = \perp$, check: $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}^*_i) \stackrel{?}{=} 1 \wedge P(x^*, y^*_i) \stackrel{?}{=} 0$, abort if not.
- When $(i, \mathsf{pk}^*_i) \in C$, check $P(x^*, y^*_i) \stackrel{?}{=} 0$, abort if not.

For all $i \in [L]\setminus\{\ell\}$, parse $\mathsf{pk}^*_i = \big([\mathbf{T}^*_i, \mathbf{Q}^*_i]_1, \{[\mathbf{h}^*_{i,j}]_2\}_{j\in[L]\setminus\{i\}}, \pi^*_i\big)$, using $\widetilde{\mathbf{R}}_i, \mathbf{V}_i, \mathbf{W}_i$ we have sampled; $\mathbf{c}, \widehat{\mathbf{c}}$ from the input and $\mathbf{V}_\ell, \mathbf{W}_\ell, \mathbf{k}$ we have sampled, we compute the challenge ciphertext with secret bit $b' \in \{0,1\}$ as follow:

$$\begin{aligned}
\mathbf{c}^*_0 &= \mathbf{c} \\
\mathbf{c}^*_1 &= ((\mathbf{c}\mathbf{V}_\ell + \widehat{\mathbf{c}})(\mathbf{a}_{y^*_\ell} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y^*_\ell} \otimes \mathbf{I}_{k+1})) \\
&\quad + \textstyle\sum_{i\in[L]\setminus\{\ell\}} ((\mathbf{c}\mathbf{V}_i + \mathbf{e}_1\widetilde{\mathbf{R}}_i^{-1}\mathbf{Q}^*_i)(\mathbf{a}_{y^*_i} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_i(\mathbf{K}_{y^*_i} \otimes \mathbf{I}_{k+1})) \\
\mathbf{c}^*_2 &= \textstyle\sum_{i\in[L]} \mathbf{c}\mathbf{W}_i(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) \\
C^* &= [\mathbf{c}\mathbf{k}^\top]_T \cdot \mathsf{m}^*_{b'}
\end{aligned}$$

And return

$$\mathsf{ct}_{x^*} = \big([\mathbf{c}^*_0, \mathbf{c}^*_1, \mathbf{c}^*_2]_1, C^*\big)$$

With above simulation and along with Lemma 2, we can observe that for all $q \in Q$, there exist $\mathcal{B}_2$ such that $\Pr[\mathsf{Bad}_q] \le 2 \cdot \mathsf{Adv}^{\mathrm{MDDH}}_{\mathcal{B}_2} + \mathsf{negl}(\lambda)$. So, we can readily prove that the advantage of argument (15) is bounded by $2Q \cdot \mathsf{Adv}^{\mathrm{MDDH}}_{\mathcal{B}_2} + \mathsf{negl}(\lambda)$.

2. We argue that:

$$\begin{aligned}
& \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}^\top_\ell, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}^\top_\ell + b\mathbf{c}^\perp\alpha \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}^*_\ell;\ \mathbf{A}\mathbf{U}^*_\ell, [\mathbf{R}_\ell\mathbf{U}^*_\ell + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1, \mathbf{U}^*_\ell\mathbf{B} \\
\approx_s\ & \mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}^\top_\ell, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}^\top_\ell + \boxed{\mathbf{c}^\perp v_\ell} + b\mathbf{c}^\perp\alpha \\
& \mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}^*_\ell + \boxed{v_\ell\mathbf{d}^\perp + u_\ell\mathbf{d}^\perp};\ \mathbf{A}\mathbf{U}^*_\ell, [\mathbf{R}_\ell\mathbf{U}^*_\ell + \boxed{\mathbf{R}_\ell\mathbf{c}^\perp u_\ell\mathbf{d}^\perp} + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1, \mathbf{U}^*_\ell\mathbf{B}
\end{aligned} \tag{16}$$

where $v_\ell, u_\ell \leftarrow \mathbb{Z}_p$. We change the variables as follows:

$$\mathbf{V}_\ell \mapsto \mathbf{V}_\ell + \mathbf{c}^\perp v_\ell\mathbf{d}^\perp \quad \text{and} \quad \mathbf{U}^*_\ell \mapsto \mathbf{U}^*_\ell + \mathbf{c}^\perp u_\ell\mathbf{d}^\perp.$$

With the fact (14), when simulating crs, we have

$$\mathbf{A}(\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell\mathbf{d}^\perp) = \mathbf{A}\mathbf{V}_\ell, \quad (\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell\mathbf{d}^\perp)\mathbf{B} = \mathbf{V}_\ell\mathbf{B}, \quad (\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell\mathbf{d}^\perp)\mathbf{d}^\top_\ell + b\mathbf{c}^\perp\alpha = \mathbf{V}_\ell\mathbf{d}^\top_\ell + \mathbf{c}^\perp v_\ell;$$

when simulating the challenge public key pk_ℓ^* on slot ℓ, we have

$$\mathbf{A}(\mathbf{U}_\ell^* + \mathbf{c}^\perp u_\ell \mathbf{d}^\perp) = \mathbf{A}\mathbf{U}_\ell^*, \quad (\mathbf{U}_\ell^* + \mathbf{c}^\perp u_\ell \mathbf{d}^\perp)\mathbf{B} = \mathbf{U}_\ell^*\mathbf{B};$$

when simulating $\mathbf{c}_1^*$, we have

$$\mathbf{c}(\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp) = \mathbf{c}\mathbf{V}_\ell + v_\ell \mathbf{d}^\perp, \quad \mathbf{c}(\mathbf{U}_\ell^* + \mathbf{c}^\perp u_\ell \mathbf{d}^\perp) = \mathbf{c}\mathbf{U}_\ell^* + u_\ell \mathbf{d}^\perp.$$

This readily prove the argument (16).

3. We argue that:

$$\begin{aligned}
&\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp v_\ell + b\mathbf{c}^\perp\alpha \\
&\mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}_\ell^* + v_\ell\mathbf{d}^\perp + u_\ell\mathbf{d}^\perp;\ \mathbf{A}\mathbf{U}_\ell^*, [\mathbf{R}_\ell\mathbf{U}_\ell^* + \mathbf{R}_\ell\mathbf{c}^\perp u_\ell\mathbf{d}^\perp + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1, \mathbf{U}_\ell^*\mathbf{B} \\
\approx_s\ &\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, [\mathbf{R}_\ell]_1, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp v_\ell + \cancel{b\mathbf{c}^\perp\alpha} \\
&\mathbf{c}, \mathbf{c}\mathbf{V}_\ell + \mathbf{c}\mathbf{U}_\ell^* + v_\ell\mathbf{d}^\perp + u_\ell\mathbf{d}^\perp;\ \mathbf{A}\mathbf{U}_\ell^*, [\mathbf{R}_\ell\mathbf{U}_\ell^* + \mathbf{R}_\ell\mathbf{c}^\perp u_\ell\mathbf{d}^\perp + \widehat{\mathbf{u}}^\top\mathbf{d}^\perp]_1, \mathbf{U}_\ell^*\mathbf{B}
\end{aligned} \tag{17}$$

This argument follows from the fact that $\widehat{\mathbf{u}}$ only appears in pk_ℓ^*, it implies that $\widehat{\mathbf{u}}$ can hide $\mathbf{R}_\ell\mathbf{c}^\perp u_\ell$; which means that u_ℓ will not be revealed by pk_ℓ^*. Thus, u_ℓ is sufficient to hide v_ℓ which only appears with u_ℓ in $\mathbf{c}_1^*$ and appears with $b\alpha$ in crs, and is sufficient to hide $b\alpha$. This readily prove the argument (17).

With argument (15), (16) and (17), we can readily prove the lemma in the honest case.

Corrupted & Malicious Case. In this case, we have $\mathsf{pk}_\ell^* = ([\mathbf{T}_\ell^*, \mathbf{Q}_\ell^*]_1, \{[\mathbf{h}_{\ell,j}^{*\top}]_2\}_{j\in[L]\setminus\{\ell\}}, \pi_\ell^*) \in C_\ell \cup \overline{D}_\ell$. And it is required that $P(x^*, y_\ell^*) = 0$. To prove the indistinguishability of $\mathsf{G}_{5,\ell-1,1}$ and $\mathsf{G}_{5,\ell-1,2}$ in this case, we firstly recall the related terms which is known by adversary as follows:

$$\begin{aligned}
&\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{A}\mathbf{W}_\ell, \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha && \text{// in crs} \\
&\mathbf{c}, \mathbf{c}\mathbf{V}_\ell(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1}), \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) && \text{// in } \mathsf{ct}^*
\end{aligned}$$

And we establish a series of argument as below:

1. We argue that:

$$\begin{aligned}
&\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{A}\mathbf{W}_\ell, \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + b\mathbf{c}^\perp\alpha \\
&\mathbf{c}, \mathbf{c}\mathbf{V}_\ell(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1}), \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) \\
\approx_s\ &\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{A}\mathbf{V}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{A}\mathbf{W}_\ell, \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + \boxed{\mathbf{c}^\perp v_\ell} + b\mathbf{c}^\perp\alpha \\
&\mathbf{c}, \mathbf{c}\mathbf{V}_\ell(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{c}\mathbf{W}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \boxed{v_\ell\mathbf{a}_{y_\ell^*} \otimes \mathbf{d}^\perp + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*} \otimes \mathbf{d}^\perp}, \mathbf{c}\mathbf{W}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) + \boxed{\mathbf{w}_\ell\mathbf{C}_{x^*} \otimes \mathbf{d}^\perp}
\end{aligned} \tag{18}$$

where $v_\ell \leftarrow \mathbb{Z}_p$ and $\mathbf{w}_\ell \leftarrow \mathbb{Z}_p^n$. We change the variables as follows:

$$\mathbf{V}_\ell \mapsto \mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp \quad \text{and} \quad \mathbf{W}_\ell \mapsto \mathbf{W}_\ell + \mathbf{c}^\perp(\mathbf{w}_\ell \otimes \mathbf{d}^\perp).$$

With the fact (14), when simulating crs, we have

$$\mathbf{A}(\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp) = \mathbf{A}\mathbf{V}_\ell, \quad (\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp)\mathbf{B} = \mathbf{V}_\ell\mathbf{B}, \quad (\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp)\mathbf{d}_\ell^\top = \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp v_\ell$$

and

$$\mathbf{A}(\mathbf{W}_\ell + \mathbf{c}^\perp(\mathbf{w}_\ell \otimes \mathbf{d}^\perp)) = \mathbf{A}\mathbf{W}_\ell, \quad (\mathbf{W}_\ell + \mathbf{c}^\perp(\mathbf{w}_\ell \otimes \mathbf{d}^\perp))(\mathbf{I}_n \otimes \mathbf{B}) = \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B});$$

when simulating $\mathbf{c}_1^*$ and $\mathbf{c}_2^*$, we have

$$\mathbf{c}(\mathbf{V}_\ell + \mathbf{c}^\perp v_\ell \mathbf{d}^\perp) = \mathbf{c}\mathbf{V}_\ell + v_\ell \mathbf{d}^\perp, \quad \mathbf{c}(\mathbf{W}_\ell + \mathbf{c}^\perp(\mathbf{w}_\ell \otimes \mathbf{d}^\perp)) = \mathbf{c}\mathbf{W}_\ell + (\mathbf{w}_\ell \otimes \mathbf{d}^\perp).$$

This readily prove the argument (18).

2. We argue that:

$$\begin{aligned}
&\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{AV}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{AW}_\ell, \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top\mathbf{c}^\perp\nu_\ell + b\mathbf{c}^\perp\alpha \\
&\mathbf{c}, \mathbf{cV}_\ell(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{cW}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \nu_\ell\mathbf{a}_{y_\ell^*} \otimes \mathbf{d}^\perp + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*} \otimes \mathbf{d}^\perp, \mathbf{cW}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{C}_{x^*} \otimes \mathbf{d}^\perp \\
\approx_s\ &\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{AV}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{AW}_\ell, \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp\nu_\ell + b\mathbf{c}^\perp\alpha \\
&\mathbf{c}, \mathbf{cV}_\ell(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{cW}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \cancel{\nu_\ell\mathbf{a}_{y_\ell^*} \otimes \mathbf{d}^\perp} + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*} \otimes \mathbf{d}^\perp, \mathbf{cW}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{C}_{x^*} \otimes \mathbf{d}^\perp
\end{aligned} \tag{19}$$

This argument follows from the fact that $P(x^*, y_\ell^*) = 0$, so that ν_ℓ in $\mathbf{c}_1^*$ can be hidden by $\mathbf{w}_\ell$ with the security of predicate encoding defined in Section 2.3.

3. We argue that:

$$\begin{aligned}
&\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{AV}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{AW}_\ell, \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp\nu_\ell + b\mathbf{c}^\perp\alpha \\
&\mathbf{c}, \mathbf{cV}_\ell(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{cW}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*} \otimes \mathbf{d}^\perp, \mathbf{cW}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{C}_{x^*} \otimes \mathbf{d}^\perp \\
\approx_s\ &\mathbf{A}, \mathbf{c}^\perp, \mathbf{B}, \mathbf{d}_\ell^\top, \mathbf{AV}_\ell, \mathbf{V}_\ell\mathbf{B}, \mathbf{AW}_\ell, \mathbf{W}_\ell(\mathbf{I}_n \otimes \mathbf{B}), \mathbf{V}_\ell\mathbf{d}_\ell^\top + \mathbf{c}^\perp\nu_\ell + \cancel{b\mathbf{c}^\perp\alpha} \\
&\mathbf{c}, \mathbf{cV}_\ell(\mathbf{a}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{cW}_\ell(\mathbf{K}_{y_\ell^*} \otimes \mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{K}_{y_\ell^*} \otimes \mathbf{d}^\perp, \mathbf{cW}_\ell(\mathbf{C}_{x^*} \otimes \mathbf{I}_{k+1}) + \mathbf{w}_\ell\mathbf{C}_{x^*} \otimes \mathbf{d}^\perp
\end{aligned} \tag{20}$$

This argument follows from the fact that ν_ℓ only appears with $b\alpha$ in crs, and it is sufficient to hide $b\alpha$.

With argument (18), (19) and (20), we can readily prove the lemma in the corrupted and malicious case. □

Lemma 10. *($\mathsf{G}_{5,\ell-1,2} \approx_c \mathsf{G}_{5,\ell-1,3}$). For any adversary $\mathcal{A}$, there exist algorithm $\mathcal{B}_2$ such that* $\mathsf{Time}(\mathcal{B}_2) \approx \mathsf{Time}(\mathcal{A})$ *and*

$$|\mathsf{Adv}_{\mathcal{A}}^{5,\ell-1,2}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{5,\ell-1,3}| \le \mathsf{Adv}_{\mathcal{B}_2}^{\mathrm{MDDH}}(\lambda) + \mathsf{negl}(\lambda).$$

Proof. The proof is analogous to Lemma 8, except that we replace $\mathbf{V}_\ell\mathbf{t}^\top + \mathbf{k}^\top$ with $\mathbf{V}_\ell\mathbf{t}^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha$ in the setup phase of simulation. Namely, the simulation have the following change in crs:

$$\mathsf{crs} = \begin{pmatrix}
[\mathbf{A}]_1, [\mathbf{Ak}^\top]_T, \left\{\mathsf{crs}_i, [\widehat{\mathbf{R}}_i, \mathbf{AV}_i, \mathbf{AW}_i]_1\right\}_{i\in[L]} \\
\left\{[\mathbf{Br}_j^\top, \mathbf{V}_j\mathbf{Br}_j^\top + \mathbf{k}^\top + \mathbf{c}^\perp\alpha]_2\right\}_{j\in[\ell-1]}, [\mathbf{t}^\top, \mathbf{V}_\ell\mathbf{t}^\top + \mathbf{k}^\top]_2, \left\{[\mathbf{Br}_j^\top, \mathbf{V}_j\mathbf{Br}_j^\top + \mathbf{k}^\top + \boxed{\mathbf{c}^\perp\alpha}]_2\right\}_{j\in[L]\setminus[\ell]} \\
\left\{[\mathbf{V}_i\mathbf{Br}_j^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{Br}_j^\top)]_2\right\}_{j\in[L]\setminus\{\ell\}, i\in[L]\setminus\{j\}}, \left\{[\mathbf{V}_i\mathbf{t}^\top, \mathbf{W}_i(\mathbf{I}_n \otimes \mathbf{t}^\top)]_2\right\}_{i\in[L]\setminus\{\ell\}}
\end{pmatrix}.$$

Observe that when $\mathbf{t}^\top = \mathbf{d}_\ell^\top$, the simulation is identical to $\mathsf{G}_{5,\ell-1,2}$; when $\mathbf{t}^\top = \mathbf{Br}_\ell^\top$, the simulation is identical to $\mathsf{G}_{5,\ell-1,3}$. This readily proves the lemma. □

D More Concrete Slotted Reg-ABE

D.1 Slotted Reg-ABE for Span Program

This section present a concrete slotted Reg-ABE for boolean span program. We use the predicate encoding of (monotone) boolean span programs [CGW15, Appendix A.5].

Preliminaries. A (monotone) boolean span program [CGW15, Appendix A.5], denoted by V, is defined by $\mathbf{Y} \in \mathbb{Z}_p^{m\times\ell}$ where

$$V(\mathbf{x}) = 1 \iff \mathbf{x} \in \{0,1\}^{1\times m} \text{ satisfies } \mathbf{Y} \iff \exists \boldsymbol{\omega} \in \mathbb{Z}_p^{1\times m} \text{ such that } \mathbf{e}_1 = \boldsymbol{\omega} \cdot \mathsf{diag}(\mathbf{x})\mathbf{Y}$$

Here we use notation: $\mathsf{diag}(\mathbf{x}) := \begin{pmatrix} x_1 & & \\ & \ddots & \\ & & x_m \end{pmatrix} \in \mathbb{Z}_p^{m\times m}$ for $\mathbf{x} = (x_1, \ldots, x_m)$ and note that $\mathsf{diag}(\mathbf{x}) = \mathsf{diag}(\mathbf{x})^\top$. We review the predicate encoding for boolean span program predicate (ciphertext-policy variant):

$$P(V, \mathbf{x}) = 1 \iff V(\mathbf{x}) = 1$$

as follows [CGW15, Appendix A.5]: let $n = m + \ell$, $n_c = m$ and $n_k = m + 1$, define

$$\mathbf{C_Y} = \begin{pmatrix} \mathbf{I}_m \\ \mathbf{Y}^\top \end{pmatrix},\ \mathbf{K_x} = \begin{pmatrix} \mathbf{0}_m & \mathsf{diag}(\mathbf{x}) \\ \mathbf{e}_1^\top & \mathbf{0}_{\ell \times m} \end{pmatrix},\ \mathbf{a_x} = (1 \| \mathbf{0}_m),\ \mathbf{d_{Y.x}} = (1 \| \boldsymbol{\omega} \| - \mathsf{diag}(\mathbf{x}) \cdot \boldsymbol{\omega})$$

where $\mathbf{0}_m$ is a *row* zero vector of size m. Note that we work with *read-once* boolean span program as in [CGW15].

Scheme. Our concrete slotted Registered CP-ABE for read-once boolean span program from SXDH (1-Lin) assumption works as follows:

– $\mathsf{Setup}(1^\lambda, P, 1^L)$: Run $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$. Sample

$$\mathbf{a} \leftarrow \mathbb{Z}_p^{1\times 3},\ \mathbf{b}^\top \leftarrow \mathbb{Z}_p^2,\ \mathbf{k} \leftarrow \mathbb{Z}_p^{1\times 3}.$$

For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{3\times 2},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{3\times 2(m+\ell)},\ \mathbf{R}_i \leftarrow \mathbb{Z}_p^{4\times 3},\ r_i \leftarrow \mathbb{Z}_p.$$

For all $i \in [L]$, write $\mathbf{A}_i = \left(\begin{smallmatrix} \mathbf{a} \\ \mathbf{R}_i \end{smallmatrix}\right)$ and sample

$$\mathbf{a}_i' \leftarrow \mathbb{Z}_p^{1\times 2},\ \mathbf{b}_i'^\top \leftarrow \mathbb{Z}_p^2,\ \mathbf{K}_i' \leftarrow \mathbb{Z}_p^{5\times 2},\ \mathbf{K}_{i,0}', \mathbf{K}_{i,1}' \leftarrow \mathbb{Z}_p^{2\times 2}$$

and compute

$$\mathbf{P}_i = \mathbf{A}_i^\top \mathbf{K}_i', \quad \mathbf{p}_{i,0} = \mathbf{a}_i' \mathbf{K}_{i,0}', \quad \mathbf{p}_{i,1} = \mathbf{a}_i' \mathbf{K}_{i,1}';$$
$$\mathbf{c}_i'^\top = \mathbf{K}_i' \mathbf{b}_i'^\top \quad \mathbf{c}_{i,0}'^\top = \mathbf{K}_{i,0}' \mathbf{b}_i'^\top, \quad \mathbf{c}_{i,1}'^\top = \mathbf{K}_{i,1}' \mathbf{b}_i'^\top.$$

For all $i \in [L]$, set

$$\mathsf{crs}_i = ([\mathbf{a}_i', \mathbf{P}_i, \mathbf{p}_{i,0}, \mathbf{p}_{i,1}]_1, [\mathbf{b}_i'^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2) \quad \mathsf{td}_i = \mathbf{K}_i'.$$

Output

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{a}]_1, [\mathbf{a}\mathbf{k}^\top]_T \{\mathsf{crs}_i, [\mathbf{R}_i, \mathbf{a}\mathbf{V}_i, \mathbf{a}\mathbf{W}_i]_1\}_{i\in[L]} \\ \{[\mathbf{b}^\top r_j, \mathbf{V}_j \mathbf{b}^\top r_j + \mathbf{k}^\top]_2\}_{j\in[L]} \\ \{[\mathbf{V}_i \mathbf{b}^\top r_j, \mathbf{W}_i (\mathbf{I}_{m+\ell} \otimes \mathbf{b}^\top r_j)]_2\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}.$$

– $\mathsf{Gen}(\mathsf{crs}, i)$: Sample $\mathbf{U}_i \leftarrow \mathbb{Z}_p^{3\times 2}$. Define $\mathbf{M}_i = \left(\begin{smallmatrix} \mathbf{t}_i \\ \mathbf{Q}_i \end{smallmatrix}\right) = \left(\begin{smallmatrix} \mathbf{a}\mathbf{U}_i \\ \mathbf{R}_i\mathbf{U}_i \end{smallmatrix}\right)$, sample $\mathbf{s}_i^\top \leftarrow \mathbb{Z}_p^2$, and compute

$$\pi_i = [\underbrace{\mathbf{U}_i^\top \mathbf{P}_i + \mathbf{s}_i^\top (\mathbf{p}_{i,0} + \mathbf{p}_{i,1})}_{\pi_{i,0}}, \underbrace{\mathbf{s}_i^\top \mathbf{a}_i'}_{\pi_{i,1}}]_1$$

Fetch $[\mathbf{R}_i]_1$ and $\{[\mathbf{b}^\top r_j]_2\}_{j\in[L]\setminus\{i\}}$ from crs and output

$$\mathsf{pk}_i = \big([\underbrace{\mathbf{a}\mathbf{U}_i}_{\mathbf{t}_i}, \underbrace{\mathbf{R}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{\underbrace{[\mathbf{U}_i \mathbf{b}^\top r_j]_2}_{\mathbf{h}_{i,j}^\top}\}_{j\in[L]\setminus\{i\}}, \pi_i\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

– $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i)$: Parse $\mathsf{pk}_i = \big([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\big)$ and fetch $[\mathbf{b}_i'^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2$ from crs_i in crs. Write $\mathbf{M}_i = \left(\begin{smallmatrix} \mathbf{t}_i \\ \mathbf{Q}_i \end{smallmatrix}\right)$ and parse $\pi_i = [\pi_{i,0}, \pi_{i,1}]_1$, check

$$e([\pi_{i,0}]_1, [\mathbf{b}_i'^\top]_2) \stackrel{?}{=} e([\mathbf{M}_i^\top]_1, [\mathbf{c}_i'^\top]_2) \cdot e([\pi_{i,1}]_1, [\mathbf{c}_{i,0}'^\top + \mathbf{c}_{i,1}'^\top]_2)$$

For each $j \in [L] \setminus \{i\}$, check

$$e([\mathbf{a}]_1, [\mathbf{h}_{i,j}^\top]_2) \stackrel{?}{=} e([\mathbf{t}_i]_1, [\mathbf{b}^\top r_j]_2).$$

If all these checks pass, output 1; otherwise, output 0.

– $\mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i, \mathbf{x}_i)_{i\in[L]})$: For all $i \in [L]$, parse

$$\mathsf{pk}_i = \left([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right).$$

Output:

$$\mathsf{mpk} = \left([\mathbf{a}]_1, \left[\sum_{i\in[L]} \left((\mathbf{a}\mathbf{V}_i + \mathbf{t}_i)((1\|\mathbf{0}_m)\otimes\mathbf{I}_2) + \mathbf{a}\mathbf{W}_i\left(\begin{pmatrix}\mathbf{0}_m & \mathsf{diag}(\mathbf{x}_i)\\ \mathbf{e}_1^\top & \mathbf{0}_{\ell\times m}\end{pmatrix}\otimes\mathbf{I}_2\right)\right)\right]_1, \left[\sum_{i\in[L]}\mathbf{a}\mathbf{W}_i\right]_1, [\mathbf{a}\mathbf{k}^\top]_T\right)$$

and for all $j \in [L]$

$$\mathsf{hsk}_j = \Bigg([\underbrace{\mathbf{b}^\top r_j}_{\mathbf{k}_0^\top}]_2, \underbrace{[\mathbf{V}_j\mathbf{b}^\top r_j + \mathbf{k}^\top]_2}_{\mathbf{k}_1^\top}, \underbrace{\left[\sum_{i\in[L]\setminus\{j\}}\left((\mathbf{V}_i\mathbf{b}^\top r_j + \mathbf{h}_{i,j}^\top)(1\|\mathbf{0}_m) + \mathbf{W}_i(\mathbf{I}_{m+\ell}\otimes\mathbf{b}^\top r_j)\begin{pmatrix}\mathbf{0}_m & \mathbf{I}_{\mathbf{x}_i}\\ \mathbf{e}_1^\top & \mathbf{0}_{\ell\times m}\end{pmatrix}\right)\right]_2}_{\mathbf{K}_2},$$
$$\underbrace{\left[\sum_{i\in[L]\setminus\{j\}}\mathbf{W}_i(\mathbf{I}_{m+\ell}\otimes\mathbf{b}^\top r_j)\right]_2}_{\mathbf{K}_3}\Bigg).$$

– $\mathsf{Enc}(\mathsf{mpk}, \mathbf{Y}, \mathsf{m})$: Sample $s \leftarrow \mathbb{Z}_p$. Output:

$$\mathsf{ct}_\mathbf{Y} = \Bigg([\underbrace{s\mathbf{a}}_{\mathbf{c}_0}]_1, \underbrace{\left[\sum_{i\in[L]}\left((s\mathbf{a}\mathbf{V}_i + s\mathbf{t}_i)((1\|\mathbf{0}_m)\otimes\mathbf{I}_2) + s\mathbf{a}\mathbf{W}_i\left(\begin{pmatrix}\mathbf{0}_m & \mathsf{diag}(\mathbf{x}_i)\\ \mathbf{e}_1^\top & \mathbf{0}_{\ell\times m}\end{pmatrix}\otimes\mathbf{I}_2\right)\right)\right]_1}_{\mathbf{c}_1},$$
$$\underbrace{\left[\sum_{i\in[L]} s\mathbf{a}\mathbf{W}_i\left(\begin{pmatrix}\mathbf{I}_m\\ \mathbf{Y}^\top\end{pmatrix}\otimes\mathbf{I}_2\right)\right]_1}_{\mathbf{c}_2}, \underbrace{[s\mathbf{a}\mathbf{k}^\top]_T\cdot\mathsf{m}}_{C}\Bigg).$$

– $\mathsf{Dec}(\mathsf{sk}_{i^*}, \mathsf{hsk}_{i^*}, \mathsf{ct}_\mathbf{Y})$: Parse

$$\mathsf{sk}_{i^*} = \mathbf{U}_{i^*}, \quad \mathsf{hsk}_{i^*} = [\mathbf{k}_0^\top, \mathbf{k}_1^\top, \mathbf{K}_2, \mathbf{K}_3]_2, \quad \mathsf{ct}_x = ([\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2]_1, C).$$

recover

$$[\mathbf{z}_1]_T = e([\mathbf{c}_1\|\mathbf{c}_2]_1, [\mathbf{I}_{2m+1}\otimes\mathbf{k}_0^\top]_2), \qquad [\mathbf{z}_2]_T = e\left([\mathbf{c}_0]_1, \left[\mathbf{K}_2\|\mathbf{K}_3\begin{pmatrix}\mathbf{I}_m\\ \mathbf{Y}^\top\end{pmatrix}\right]_2\right),$$
$$[z_3]_T = e([\mathbf{c}_0\mathbf{U}_{i^*}]_1, [\mathbf{k}_0^\top]_2), \qquad [z_4]_T = e([\mathbf{c}_0]_1, [\mathbf{k}_1^\top]_2).$$

Compute $\boldsymbol{\omega}$ such that $\mathbf{e}_1 = \mathbf{w}\cdot\mathsf{diag}(\mathbf{x}_{i^*})\mathbf{Y}$, output

$$\mathsf{m}' = [(\mathbf{z}_1 - \mathbf{z}_2)(1\|\boldsymbol{\omega}\| - \boldsymbol{\omega}\cdot\mathsf{diag}(\mathbf{x}_{i^*}))^\top - z_3 - z_4]_T \cdot C.$$

D.2 Slotted Reg-ABE for zero inner-product

This section present a concrete slotted Reg-ABE for zero inner product. We use the predicate encoding of inner product from [CGW15, Appendix A.1].

Preliminaries. We review the predicate encoding for zero inner-product:

$$P(\mathbf{x}, \mathbf{y}) = 1 \iff \mathbf{x}\mathbf{y}^\top = 0 \quad \text{where} \quad (\mathbf{x}, \mathbf{y}) \in \mathbb{Z}_p^{1\times m}\times\mathbb{Z}_p^{1\times m}.$$

as follows [CGW15, Appendix A.1]: let $n = m+1$, $n_c = m$ and $n_k = 1$, define

$$\mathbf{C}_\mathbf{x} = \begin{pmatrix}\mathbf{x}\\ \mathbf{I}_m\end{pmatrix}, \ \mathbf{K}_\mathbf{y} = \begin{pmatrix}0\\ \mathbf{y}^\top\end{pmatrix}, \ \mathbf{a}_\mathbf{y} = (1), \ \mathbf{d}_{\mathbf{x},\mathbf{y}} = (1\|-\mathbf{y})$$

Scheme. Our concrete slotted Reg-ABE for zero inner product from SXDH (1-Lin) assumption works as follows:

– $\mathsf{Setup}(1^\lambda, P, 1^L)$: Run $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$. Sample

$$\mathbf{a} \leftarrow \mathbb{Z}_p^{1\times 3},\ \mathbf{b}^\top \leftarrow \mathbb{Z}_p^2,\ \mathbf{k} \leftarrow \mathbb{Z}_p^{1\times 3}.$$

For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{3\times 2},\ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{3\times 2(m+1)},\ \mathbf{R} \leftarrow \mathbb{Z}_p^{4\times 3},\ r_i \leftarrow \mathbb{Z}_p.$$

For all $i \in [L]$, write $\mathbf{A}_i = \binom{\mathbf{a}}{\mathbf{R}_i}$ and sample

$$\mathbf{a}_i' \leftarrow \mathbb{Z}_p^{1\times 2},\ \mathbf{b}_i'^\top \leftarrow \mathbb{Z}_p^2,\ \mathbf{K}_i' \leftarrow \mathbb{Z}_p^{5\times 2},\ \mathbf{K}_{i,0}', \mathbf{K}_{i,1}' \leftarrow \mathbb{Z}_p^{2\times 2}$$

and compute

$$\mathbf{P}_i = \mathbf{A}_i^\top \mathbf{K}_i', \quad \mathbf{p}_{i,0} = \mathbf{a}_i' \mathbf{K}_{i,0}', \quad \mathbf{p}_{i,1} = \mathbf{a}_i' \mathbf{K}_{i,1}';$$
$$\mathbf{c}_i'^\top = \mathbf{K}_i' \mathbf{b}_i'^\top \quad \mathbf{c}_{i,0}'^\top = \mathbf{K}_{i,0}' \mathbf{b}_i'^\top, \quad \mathbf{c}_{i,1}'^\top = \mathbf{K}_{i,1}' \mathbf{b}_i'^\top.$$

For all $i \in [L]$, set

$$\mathsf{crs}_i = ([\mathbf{a}_i', \mathbf{P}_i, \mathbf{p}_{i,0}, \mathbf{p}_{i,1}]_1, [\mathbf{b}_i'^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2) \quad \mathsf{td}_i = \mathbf{K}_i'.$$

Output

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{a}]_1, [\mathbf{a}\mathbf{k}^\top]_T \{\mathsf{crs}_i, [\mathbf{R}_i, \mathbf{a}\mathbf{V}_i, \mathbf{a}\mathbf{W}_i]_1\}_{i\in[L]} \\ \{[\mathbf{b}^\top r_j, \mathbf{V}_j \mathbf{b}^\top r_j + \mathbf{k}^\top]_2\}_{j\in[L]} \\ \{[\mathbf{V}_i \mathbf{b}^\top r_j, \mathbf{W}_i(\mathbf{I}_{m+1} \otimes \mathbf{b}^\top r_j)]_2\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}.$$

– $\mathsf{Gen}(\mathsf{crs}, i)$: Sample $\mathbf{U}_i \leftarrow \mathbb{Z}_p^{3\times 2}$. Define $\mathbf{M}_i = \binom{\mathbf{t}_i}{\mathbf{Q}_i} = \binom{\mathbf{a}\mathbf{U}_i}{\mathbf{R}_i\mathbf{U}_i}$, sample $\mathbf{s}_i^\top \leftarrow \mathbb{Z}_p^2$, and compute

$$\pi_i = [\underbrace{\mathbf{U}_i^\top \mathbf{P}_i + \mathbf{s}_i^\top (\mathbf{p}_{i,0} + \mathbf{p}_{i,1})}_{\pi_{i,0}}, \underbrace{\mathbf{s}_i^\top \mathbf{a}_i'}_{\pi_{i,1}}]_1$$

Fetch $[\mathbf{R}_i]_1$ and $\{[\mathbf{b}^\top r_j]_2\}_{j\in[L]\setminus\{i\}}$ from crs and output

$$\mathsf{pk}_i = \left([\underbrace{\mathbf{a}\mathbf{U}_i}_{\mathbf{t}_i}, \underbrace{\mathbf{R}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{[\underbrace{\mathbf{U}_i \mathbf{b}^\top r_j}_{\mathbf{h}_{i,j}^\top}]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

– $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i)$: Parse $\mathsf{pk}_i = \left([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right)$ and fetch $[\mathbf{b}_i^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2$ from crs_i in crs. Write $\mathbf{M}_i = \binom{\mathbf{t}_i}{\mathbf{Q}_i}$ and parse $\pi_i = [\pi_{i,0}, \pi_{i,1}]_1$, check

$$e([\pi_{i,0}]_1, [\mathbf{b}_i'^\top]_2) \stackrel{?}{=} e([\mathbf{M}_i^\top]_1, [\mathbf{c}_i'^\top]_2) \cdot e([\pi_{i,1}]_1, [\mathbf{c}_{i,0}'^\top + \mathbf{c}_{i,1}'^\top]_2)$$

For each $j \in [L] \setminus \{i\}$, check

$$e([\mathbf{a}]_1, [\mathbf{h}_{i,j}^\top]_2) \stackrel{?}{=} e([\mathbf{t}_i]_1, [\mathbf{b}^\top r_j]_2).$$

If all these checks pass, output 1; otherwise, output 0.

– $\mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i, \mathbf{y}_i)_{i\in[L]})$: For all $i \in [L]$, parse

$$\mathsf{pk}_i = \left([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\right).$$

Output:

$$\mathsf{mpk} = \left([\mathbf{a}]_1, \left[\sum_{i\in[L]} \left((\mathbf{a}\mathbf{V}_i + \mathbf{t}_i) + \mathbf{a}\mathbf{W}_i\left(\begin{pmatrix} 0 \\ \mathbf{y}_i^\top \end{pmatrix} \otimes \mathbf{I}_2\right)\right)\right]_1, \left[\sum_{i\in[L]} \mathbf{a}\mathbf{W}_i\right]_1, [\mathbf{a}\mathbf{k}^\top]_T\right)$$

and for all $j \in [L]$

$$\mathsf{hsk}_j = \left([\underbrace{\mathbf{b}^\top r_j}_{\mathbf{k}_0^\top}]_2, [\underbrace{\mathbf{V}_j \mathbf{b}^\top r_j + \mathbf{k}^\top}_{\mathbf{k}_1^\top}]_2, \left[\underbrace{\sum_{i\in[L]\setminus\{j\}} \left((\mathbf{V}_i \mathbf{b}^\top r_j + \mathbf{h}_{i,j}^\top) + \mathbf{W}_i(\mathbf{I}_{m+1} \otimes \mathbf{b}^\top r_j)\begin{pmatrix} 0 \\ \mathbf{y}_i^\top \end{pmatrix}\right)}_{\mathbf{K}_2}\right]_2, \left[\underbrace{\sum_{i\in[L]\setminus\{j\}} \mathbf{W}_i(\mathbf{I}_{m+1} \otimes \mathbf{b}^\top r_j)}_{\mathbf{K}_3}\right]_2\right).$$

– $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, \mathsf{m})$: Sample $s \leftarrow \mathbb{Z}_p$. Output:

$$\mathsf{ct}_{\mathbf{x}} = \left(\underbrace{[\ s\mathbf{a}\ ]_1}_{\mathbf{c}_0}, \underbrace{\left[\sum_{i\in[L]} \left((s\mathbf{a}\mathbf{V}_i + s\mathbf{t}_i) + s\mathbf{a}\mathbf{W}_i \left(\begin{pmatrix} 0 \\ \mathbf{y}_i^\top \end{pmatrix} \otimes \mathbf{I}_2 \right) \right) \right]_1}_{\mathbf{c}_1}, \underbrace{\left[\sum_{i\in[L]} s\mathbf{a}\mathbf{W}_i \left(\begin{pmatrix} \mathbf{x} \\ \mathbf{I}_m \end{pmatrix} \otimes \mathbf{I}_2 \right) \right]_1}_{\mathbf{c}_2}, \underbrace{[s\mathbf{a}\mathbf{k}^\top]_T \cdot \mathsf{m}}_{C} \right).$$

– $\mathsf{Dec}(\mathsf{sk}_{i^*}, \mathsf{hsk}_{i^*}, \mathsf{ct}_{\mathbf{x}})$: Parse

$$\mathsf{sk}_{i^*} = \mathbf{U}_{i^*}, \quad \mathsf{hsk}_{i^*} = [\mathbf{k}_0^\top, \mathbf{k}_1^\top, \mathbf{K}_2, \mathbf{K}_3]_2, \quad \mathsf{ct}_x = ([\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2]_1, C).$$

recover

$$[\mathbf{z}_1]_T = e([\mathbf{c}_1 \| \mathbf{c}_2]_1, [\mathbf{I}_{m+1} \otimes \mathbf{k}_0^\top]_2), \qquad [\mathbf{z}_2]_T = e\left([\mathbf{c}_0]_1, \left[\mathbf{K}_2 \| \mathbf{K}_3 \begin{pmatrix} \mathbf{x} \\ \mathbf{I}_m \end{pmatrix} \right]_2 \right),$$
$$[z_3]_T = e([\mathbf{c}_0 \mathbf{U}_{i^*}]_1, [\mathbf{k}_0^\top]_2), \qquad [z_4]_T = e([\mathbf{c}_0]_1, [\mathbf{k}_1^\top]_2).$$

Output

$$\mathsf{m}' = [(\mathbf{z}_1 - \mathbf{z}_2)(1 \| -\mathbf{y}_{i^*})^\top - z_3 - z_4]_T \cdot C.$$

D.3 Slotted Registration-Based Encryption (RBE)

This section present a concrete slotted Registration-Based Encryption (RBE). We use the predicate encoding from [LW10] for IBE.

Preliminaries. We review the predicate encoding for IBE:

$$P(\mathsf{id}', \mathsf{id}) = 1 \iff \mathsf{id}' = \mathsf{id}$$

as follows [LW10]: let $n = 2$ and $n_c = n_k = 1$, define

$$\mathbf{C}_{\mathsf{id}'} = \begin{pmatrix} 1 \\ \mathsf{id}' \end{pmatrix}, \ \mathbf{K}_{\mathsf{id}} = \begin{pmatrix} 1 \\ \mathsf{id} \end{pmatrix}, \ \mathbf{a}_{\mathsf{id}} = (1), \ \mathbf{d}_{x,y} = (1, -1).$$

Scheme. Our concrete slotted RBE works as follows:

– $\mathsf{Setup}(1^\lambda, 1^L)$: Run $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$. Sample

$$\mathbf{a} \leftarrow \mathbb{Z}_p^{1\times 3}, \ \mathbf{b}^\top \leftarrow \mathbb{Z}_p^2, \ \mathbf{k} \leftarrow \mathbb{Z}_p^{1\times 3}.$$

For all $i \in [L]$, sample

$$\mathbf{V}_i \leftarrow \mathbb{Z}_p^{2\times 2}, \ \mathbf{W}_i \leftarrow \mathbb{Z}_p^{2\times 4}, \ \mathbf{R} \leftarrow \mathbb{Z}_p^{4\times 3}, \ r_i \leftarrow \mathbb{Z}_p.$$

For all $i \in [L]$, write $\mathbf{A}_i = \left(\begin{smallmatrix} \mathbf{a} \\ \mathbf{R}_i \end{smallmatrix}\right)$ and sample

$$\mathbf{a}_i' \leftarrow \mathbb{Z}_p^{1\times 2}, \ \mathbf{b}_i'^\top \leftarrow \mathbb{Z}_p^2, \ \mathbf{K}_i' \leftarrow \mathbb{Z}_p^{5\times 2}, \ \mathbf{K}_{i,0}', \mathbf{K}_{i,1}' \leftarrow \mathbb{Z}_p^{2\times 2}$$

and compute

$$\mathbf{P}_i = \mathbf{A}_i^\top \mathbf{K}_i', \quad \mathbf{p}_{i,0} = \mathbf{a}_i' \mathbf{K}_{i,0}', \quad \mathbf{p}_{i,1} = \mathbf{a}_i' \mathbf{K}_{i,1}';$$
$$\mathbf{c}_i'^\top = \mathbf{K}_i' \mathbf{b}_i'^\top \quad \mathbf{c}_{i,0}'^\top = \mathbf{K}_{i,0}' \mathbf{b}_i'^\top, \quad \mathbf{c}_{i,1}'^\top = \mathbf{K}_{i,1}' \mathbf{b}_i'^\top.$$

For all $i \in [L]$, set

$$\mathsf{crs}_i = ([\mathbf{a}_i', \mathbf{P}_i, \mathbf{p}_{i,0}, \mathbf{p}_{i,1}]_1, [\mathbf{b}_i'^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2) \quad \mathsf{td}_i = \mathbf{K}_i'.$$

Output

$$\mathsf{crs} = \begin{pmatrix} [\mathbf{a}]_1, [\mathbf{a}\mathbf{k}^\top]_T \{\mathsf{crs}_i, [\mathbf{R}_i, \mathbf{a}\mathbf{V}_i, \mathbf{a}\mathbf{W}_i]_1\}_{i\in[L]} \\ \left\{ [\mathbf{b}^\top r_j, \mathbf{V}_j \mathbf{b}^\top r_j + \mathbf{k}^\top]_2 \right\}_{j\in[L]} \\ \left\{ [\mathbf{V}_i \mathbf{b}^\top r_j, \mathbf{W}_i (\mathbf{I}_2 \otimes \mathbf{b}^\top r_j)]_2 \right\}_{j\in[L], i\in[L]\setminus\{j\}} \end{pmatrix}.$$

– $\mathsf{Gen}(\mathsf{crs}, i)$: Sample $\mathbf{U}_i \leftarrow \mathbb{Z}_p^{3\times 2}$. Define $\mathbf{M}_i = \binom{\mathbf{t}_i}{\mathbf{Q}_i} = \binom{\mathbf{a}\mathbf{U}_i}{\mathbf{R}_i\mathbf{U}_i}$, sample $\mathbf{s}_i^\top \leftarrow \mathbb{Z}_p^2$, and compute

$$\pi_i = [\underbrace{\mathbf{U}_i^\top \mathbf{P}_i + \mathbf{s}_i^\top(\mathbf{p}_{i,0} + \mathbf{p}_{i,1})}_{\pi_{i,0}}, \underbrace{\mathbf{s}_i^\top \mathbf{a}_i'}_{\pi_{i,1}}]_1$$

Fetch $[\mathbf{R}_i]_1$ and $\{[\mathbf{b}^\top r_j]_2\}_{j\in[L]\setminus\{i\}}$ from crs and output

$$\mathsf{pk}_i = \big([\underbrace{\mathbf{a}\mathbf{U}_i}_{\mathbf{t}_i}, \underbrace{\mathbf{R}_i\mathbf{U}_i}_{\mathbf{Q}_i}]_1, \{\underbrace{[\mathbf{U}_i\mathbf{b}^\top r_j]_2}_{\mathbf{h}_{i,j}^\top}\}_{j\in[L]\setminus\{i\}}, \pi_i\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{U}_i.$$

– $\mathsf{Ver}(\mathsf{crs}, i, \mathsf{pk}_i)$: Parse $\mathsf{pk}_i = \big([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\big)$ and fetch $[\mathbf{b}_i^\top, \mathbf{c}_i'^\top, \mathbf{c}_{i,0}'^\top, \mathbf{c}_{i,1}'^\top]_2$ from crs_i in crs. Write $\mathbf{M}_i = \binom{\mathbf{t}_i}{\mathbf{Q}_i}$ and parse $\pi_i = [\pi_{i,0}, \pi_{i,1}]_1$, check

$$e([\pi_{i,0}]_1, [\mathbf{b}_i'^\top]_2) \stackrel{?}{=} e([\mathbf{M}_i^\top]_1, [\mathbf{c}_i'^\top]_2) \cdot e([\pi_{i,1}]_1, [\mathbf{c}_{i,0}'^\top + \mathbf{c}_{i,1}'^\top]_2)$$

For each $j \in [L] \setminus \{i\}$, check

$$e([\mathbf{a}]_1, [\mathbf{h}_{i,j}^\top]_2) \stackrel{?}{=} e([\mathbf{t}_i]_1, [\mathbf{b}^\top r_j]_2).$$

If all these checks pass, output 1; otherwise, output 0.

– $\mathsf{Agg}(\mathsf{crs}, (\mathsf{pk}_i, \mathsf{id}_i)_{i\in[L]})$: For all $i \in [L]$, parse

$$\mathsf{pk}_i = \big([\mathbf{t}_i, \mathbf{Q}_i]_1, \{[\mathbf{h}_{i,j}^\top]_2\}_{j\in[L]\setminus\{i\}}, \pi_i\big).$$

Output:

$$\mathsf{mpk} = \left([\mathbf{a}]_1, \left[\sum_{i\in[L]} \left((\mathbf{a}\mathbf{V}_i + \mathbf{t}_i) + \mathbf{a}\mathbf{W}_i\left(\binom{1}{\mathsf{id}_i} \otimes \mathbf{I}_2\right)\right)\right]_1, \left[\sum_{i\in[L]} \mathbf{a}\mathbf{W}_i\right]_1, [\mathbf{a}\mathbf{k}^\top]_T\right)$$

and for all $j \in [L]$

$$\mathsf{hsk}_j = \left([\underbrace{\mathbf{b}^\top r_j}_{\mathbf{k}_0^\top}]_2, [\underbrace{\mathbf{V}_j\mathbf{b}^\top r_j + \mathbf{k}^\top}_{\mathbf{k}_1^\top}]_2, \left[\underbrace{\sum_{i\in[L]\setminus\{j\}} \left((\mathbf{V}_i\mathbf{b}^\top r_j + \mathbf{h}_{i,j}^\top) + \mathbf{W}_i(\mathbf{I}_2 \otimes \mathbf{b}^\top r_j)\binom{1}{\mathsf{id}_i}\right)}_{\mathbf{K}_2}\right]_2, \left[\underbrace{\sum_{i\in[L]\setminus\{j\}} \mathbf{W}_i(\mathbf{I}_2 \otimes \mathbf{b}^\top r_j)}_{\mathbf{K}_3}\right]_2\right).$$

– $\mathsf{Enc}(\mathsf{mpk}, \mathsf{id}', \mathsf{m})$: Sample $s \leftarrow \mathbb{Z}_p$. Output:

$$\mathsf{ct}_{\mathsf{id}'} = \left([\underbrace{s\mathbf{a}}_{\mathbf{c}_0}]_1, \left[\underbrace{\sum_{i\in[L]} \left((s\mathbf{a}\mathbf{V}_i + s\mathbf{t}_i) + s\mathbf{a}\mathbf{W}_i\left(\binom{1}{\mathsf{id}_i} \otimes \mathbf{I}_2\right)\right)}_{\mathbf{c}_1}\right]_1, \left[\underbrace{\sum_{i\in[L]} s\mathbf{a}\mathbf{W}_i\left(\binom{1}{\mathsf{id}'} \otimes \mathbf{I}_2\right)}_{\mathbf{c}_2}\right]_1, \underbrace{[s\mathbf{a}\mathbf{k}^\top]_T \cdot \mathsf{m}}_{C}\right).$$

– $\mathsf{Dec}(\mathsf{sk}_{i^*}, \mathsf{hsk}_{i^*}, \mathsf{ct}_{\mathsf{id}'})$: Parse

$$\mathsf{sk}_{i^*} = \mathbf{U}_{i^*}, \quad \mathsf{hsk}_{i^*} = [\mathbf{k}_0^\top, \mathbf{k}_1^\top, \mathbf{K}_2, \mathbf{K}_3]_2, \quad \mathsf{ct}_x = ([\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3]_1, C).$$

recover

$$[\mathbf{z}_1]_T = e([\mathbf{c}_1 \| \mathbf{c}_2]_1, [\mathbf{I}_2 \otimes \mathbf{k}_0^\top]_2), \qquad [\mathbf{z}_2]_T = e\left([\mathbf{c}_0]_1, \left[\mathbf{K}_2 \| \mathbf{K}_3\binom{1}{\mathsf{id}'}\right]_2\right),$$
$$[z_3]_T = e([\mathbf{c}_0\mathbf{U}_{i^*}]_1, [\mathbf{k}_0^\top]_2), \qquad [z_4]_T = e([\mathbf{c}_0]_1, [\mathbf{k}_1^\top]_2).$$

Output

$$\mathsf{m}' = [(\mathbf{z}_1 - \mathbf{z}_2)(1, -1)^\top) - z_3 - z_4]_T \cdot C.$$

Table of Contents