Registered Functional Encryptions from Pairings

Ziqi Zhu¹, Jiangtao Li², Kai Zhang³, Junqing Gong¹,⁴, and Haifeng Qian¹,⁵

¹ East China Normal University
52275902001@stu.ecnu.edu.cn
jqgong@sei.ecnu.edu.cn
hqian@cs.ecnu.edu.cn
² Shanghai University
lijiangtao@shu.edu.cn
³ Shanghai University of Electric Power
kzhang@shiep.edu.cn
⁴ Shanghai Qi Zhi Institute
⁵ Science and Technology on Communication Security Laboratory Foundation (6142103022208), 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).

Abstract. This work initiates the study of concrete registered functional encryption (Reg-FE) beyond “all-or-nothing” functionalities:

– We build the first Reg-FE for linear function or inner-product evaluation (Reg-IPFE) from pairings. The scheme achieves adaptive IND-security under $k$-Lin assumption in the prime-order bilinear group. A minor modification yields the first Registered Inner-Product Encryption (Reg-IPE) scheme from $k$-Lin assumption. Prior work achieves the same security in the generic group model.

– We build the first Reg-FE for quadratic function (Reg-QFE) from pairings. The scheme achieves very selective simulation-based security (SIM-security) under bilateral $k$-Lin assumption in the prime-order bilinear group. Here, “very selective” means that the adversary claims challenge messages, all quadratic functions to be registered and all corrupted users at the beginning.

Besides focusing on the compactness of the master public key and helper keys, we also aim for compact ciphertexts in Reg-FE. Let $L$ be the number of slots and $n$ be the input size. Our first Reg-IPFE has weakly compact ciphertexts of size $O(n \cdot \log L)$ while our second Reg-QFE has compact ciphertexts of size $O(n + \log L)$. Technically, for our first Reg-IPFE, we employ nested dual-system method within the context of Reg-IPFE; for our second Reg-QFE, we follow Wee’s “IPFE-to-QFE” transformation [TCC’20] but devise a set of new techniques that make our pairing-based Reg-IPFE compatible. Along the way, we introduce a new notion named Pre-Constrained Registered IPFE which generalizes slotted Reg-IPFE by constraining the form of functions that can be registered.

Contents


1 Introduction

In Registered Functional Encryption (Reg-FE) [FFM⁺23,DP23], a trusted party generates a common reference string $crs$ and then can go offline. The system is maintained by curator who holds $crs$ but no secret values. When a user
registers public key \( pk \) with a specific function \( f \), the curator updates the master public key \( mpk \) and sends a helper key \( hsk \) to the new user. This \( hsk \) allows the user’s secret key \( sk \) to decrypt a ciphertext \( ct \) of \( f(x) \). Additionally, the registration process might also update helper keys for existing users in the system. Two crucial features of \( RFE \) are: (1) all actions performed by the curator are deterministic and auditable, and (2) \( mpk \) and \( hsk \) should be compact and update procedure must be efficient; ideally, objective sizes and algorithm costs are polylogarithmic in the number of registered users in the system.

Conceptually, \( RFE \) covers the notion of registered attribute-based encryption (\( RABE \)) [HLWW23]. In particular, each user registers a predicate \( p \) instead of a function \( f \), and a ciphertext encrypts \( m \) with respect to an attribute \( a \); decrypting the ciphertext using the secret key \( sk \) corresponding to predicate \( p \) recovers \( m \) if \( p(a) = 1 \). The most fundamental instance of \( RABE \) is called registration-based encryption (\( RBE \)) [GHMR18] corresponding to IBE [BF01, BB04, Wat05].

Historically, several constructions for \( RBE \) were first proposed via non-black-box technique based on garbling scheme [GHMR18, GHM∗19, GV20, CES21]. Constructions via black-box technique were recently proposed based on bilinear maps [GKMR22] and learning with error (LWE) [DKL+23]. Almost simultaneously, \( RABE \) that goes beyond \( RBE \) was realized using bilinear maps [HLWW23, FFM∗23, ZZGQ23] and witness encryption [FWW23]. However, for more general \( RFE \), we only see two recent work that presented schemes based on iO [FFM∗23, DP23].

In this work, we will focus on \( RFE \) for \textit{concrete} functionalities instead of \textit{general} functions in [FFM∗23, DP23] and pursue \textit{pairing-based} constructions from standard assumptions, notably \( k\)-Lin assumption and variants.

### 1.1 Results

Our main results are two-fold:

1. We build the first \( RFE \) for linear functions or inner-product evaluation (\( RIFPE \)) from pairings: Each user is allowed to register \( pk \) with a linear function represented by a vector \( y \); decrypting a ciphertext of vector \( x \) gives \( xy^\top \). The scheme achieves adaptive indistinguishability-based security (IND-security) under \( k\)-Lin assumption in the prime-order bilinear group.

2. We build the first \( RFE \) for quadratic functions (\( RFQFE \)) from pairings: Each user is allowed to register \( pk \) with a quadratic function represented by a vector \( f \); decrypting a ciphertext of \( (x_1, x_2) \) gives \( (x_1 \otimes x_2)f^\top \). The scheme achieves \textit{very selective} simulation-based security (SIM-security) under bilateral \( k\)-Lin assumption in the prime-order bilinear group. Here, “very selective” means that the adversary claims challenge messages, all quadratic functions to be registered, and all corrupted users at the beginning.

This is the first time we have concrete \( RFE \) from standard assumptions with functionalities beyond “all-or-nothing” decryption. As prior pairing-based schemes [HLWW23, FFM∗23, ZZGQ23], all our \( RFE \) schemes support \textit{bounded} number of slots, have a \textit{structural} crs and require a specific procedure checking the validity of public key in the registration. Let \( L \) be the number of slots and \( n \) be the input size (which refers to \( |x| \) in \( RIFPE \) and \( |x_1|, |x_2| \) in \( RFQFE \), respectively). Our schemes respectively have compact \( mpk \) of size \( O(n \cdot \log L) \) and \( O(n + \log L) \), and both schemes have compact \( hsk \) of size \( O(n \cdot \log L) \). By contrast with \( RBE \) and \( RABE \), we also concern ciphertext size in terms of \( n \) and \( L \): our first \( RIFPE \) has \textit{weak compact} \( ct \) of size \( O(n \cdot \log L) \) while our second \( RFQFE \) has \textit{compact} \( ct \) of size \( O(n + \log L) \). We summarize our results in Figure 1.

### More Results

Our first \( RIFPE \) scheme implies the following results:

1. A minor modification to our \( RIFPE \) scheme yields the first Registered Inner-Product Encryption (\( RIFE \)) scheme that supports full attribute-hiding feature from \( k\)-Lin assumption. Prior work [FFM∗23] achieves the
same security in the generic group model; this resolved the open problem posed in \cite{FFM23}. The scheme is similar to the Reg-ABE for zero inner-product predicate in \cite{ZZGQ23} (and IPE in \cite{OT12,CGKW18,CGW18}). However their generic framework failed to give a proof for full attribute-hiding; our work show that, for the concrete scheme, it is actually feasible to give a proof from $k$-Lin.

Along the way to our second Reg-QFE scheme, we obtain the following results which can be of independent interest:

**(ii)** We obtain two Reg-IPFE schemes with compact ciphertext of size $O(n + \log L)$ and shorter hsk of size independent of $n$ but weaker security guarantee; the selectively IND-secure scheme is based on $k$-Lin assumption while the very selectively SIM-secure scheme is based on bi-$k$-Lin assumption. See Figure 1. We believe they will find more theoretical applications in the future.

**(iii)** We introduce a new notion \textit{Pre-Constrained Registered IPFE} (PReg-IPFE) which generalizes slotted Reg-IPFE. It generates \texttt{crs} with a set of matrices $M_1, \ldots, M_L$ and decryption gives $xM_if_i^\top$ for slot $i$ that is with $f_i$. We conceptually consider $y_i^\top = M_if_i^\top$ as the linear function related to slot $i$. Imagine $M$ is a “tall” matrix, we are forcing $y_i^\top \in \text{span}(M_i)$. We believe this will motivate the study of \textit{registration patterns} orthogonal to functionalities.

**Open Problems.** We list some open problems:

- We consider pre-constrained Reg-IPFE as a theoretical tool for a specific task. We believe it is worthwhile to investigate more general definitions, security, and constructions. They can be of independent interest even in real-world applications. Here we mention a related notion called \textit{Pre-Constrained Encryption} \cite{AJJM22} which has many theoretical implications. It is also nice to clarify the relation between these two notions.

- For Reg-IPFE, our work suggests that compact ciphertext and adaptive security can not be achieved simultaneously. One can disprove this conjecture by showing a Reg-IPFE scheme with both properties or providing an impossibility result to confirm it.

- Our Reg-QFE has \texttt{crs} of size $n^2 \cdot L^2 \cdot \log L$ where $n$ is the input size and $L$ is the number of slots. It is unclear whether such a huge \texttt{crs} is inevitable and is nice to have a more efficient Reg-QFE scheme with $|\texttt{crs}| = n \cdot L^2 \cdot \log L$.

**Related Work.** We mention several recent work on RBE. \cite{FKdP23} proposed a new black-box construction of RBE from Cuckoo hashing, which supports unbounded identity spaces based on pairings. \cite{MQR22} found the trade-off between the size of public parameters and the number of decryption updates in RBE, they find out that the optimal number of decryption updates is $\Omega(\log L / \log \log L)$, when the size of public parameters is at most $\text{poly}(\log L)$. They prove their result by constructing a polynomial-time adversary with the “good” identities tuples for attack,
when the RBE scheme is beyond the trade-off they claim. [MQ23] constructed an RBE that achieves the optimal number of decryption updates with an online merger. In particular, they constructed an (approximately) optimal online merger, and applied it to the iO-based construction of [GHMR18] to achieve the optimal decryption update of RBE. In [HMQS23], Hajiabadi et al. showed the impossibility of black-box construction of RBE solely based on the idealized models of random trapdoor permutations (TDP) or Shoup’s generic group model, without any other concrete assumption. With the black-box equivalence between RBE and public-key compression (PKCom), they proved their impossibility by showing there exists an adversary with polynomial queries, who breaks any PKCom which is solely based on either TDP model or Shoup’s GGM. Their impossibility holds even if the size of crs is growing with the number of registered users.

Concurrent Work. As an independent work, Datta et al. [DPY23] (which is an updated version of [DP23]) provided a pairing-based Reg-IPFE from (plain) IPFE proposed by Abdalla et al. [ABDP15], and extended their Reg-IPFE to support fine-grained access control with linear secret sharing access structure (LSSS) policy. Their schemes are secure in the generic bilinear group model.

1.2 Slotted Reg-IPFE from k-Lin

Thanks to “powers-of-two” transformation [GHMR18,HLWW23,FFM23], we focus on slotted Reg-IPFE where we do not worry about the complex update procedure. Let lower-case boldface denote row vectors and upper-case boldface denote matrices. An \( L \)-slotted Reg-IPFE simplifies Reg-IPFE for \( L \) users as follows: After collecting all \( R = ((pk_1, y_1), \ldots, (pk_L, y_L)) \), the aggregator generates a master public key mpk for encryption and a set of helper keys hsk\(_1\), \ldots, hsk\(_L\) for all registered users. Conceptually, the adaptive security requires that the adversary cannot distinguish the ciphertext \( ct^* \) of message \( x^*_0 \) and \( x^*_1 \) given \( mpk \), \( hsk_1 \), \ldots, \( hsk_L \) and secret keys \( sk_i \) from adversarially chosen slots with the restriction \( x^*_0 y_i^T = x^*_1 y_i^T \). In this overview, we assume all \( pk_1 \), \ldots, \( pk_L \) are generated by the challenger and the case with malicious keys can be handled via quasi-adaptive NIZK [ZZGQ23].

Recap: ABDP IPFE [ABDP15]. Assume \( \mathbb{G} \) is a finite cyclic group of prime order \( p \) with generator \( g \). Write \( [x] = g^x \in \mathbb{G} \) for \( x \in \mathbb{Z}_p \). Our starting point is the IPFE scheme for \( n \) dimensional space from [ABDP15]:

\[
\text{mpk} = [w]; \quad \text{ct} = [s, sw + x]; \quad \text{sk} = wy^T
\]  

where \( w \leftarrow \mathbb{Z}_p^n \) and \( s \leftarrow \mathbb{Z}_p \). The correctness uses the equality

\[
\frac{ct}{(sw + x) \cdot y^T} - \frac{ct}{s} \cdot wy^T = xy^T
\]  

The selective security of the scheme is based on DDH assumption. We omit the proof here since it is not quite related to our final proof for slotted Reg-IPFE.

Warm-up. We employ the strategy in [ZZGQ23] and [HLWW23] to build a one-slot Reg-IPFE and then extend it to a \( L \)-slotted Reg-IPFE. Let us give a slightly detailed explanation. Based on the correctness of equation (2), we first enables a user with an ElGamal-type key pair \((pk, sk)\) to register \((pk, y)\), as follows:

- \( \text{crs} = [w] \) is basically the mpk of ABDP IPFE (1) and the key pair of user is \((pk, sk) = ([u], u)\) with \( u \leftarrow \mathbb{Z}_p \).  

Formally, the adversary is given \( \text{crs} \) that allows it to derive \( mpk, hsk_1, \ldots, hsk_L \) on its own; our conceptual definition gives a simple mind model analogous to FE.
To register \( R = (pk, y) \), the aggregator generates the corresponding master public key \( mpk_R = [u + wy', w] \).

Under this \( mpk_R \), we encrypt \( x \) as \( ct = [s, su + swy', sw + x] \) where \( s \leftarrow \mathbb{Z}_p \).

The main idea above is to embed the decryption shown in (2) into the ciphertext and use an ElGamal encryption to hide the key \( wy' \). The correctness uses

\[
\underbrace{ct - (sw + x) \cdot y'}_{\text{(2)}} - \underbrace{ct}_{\text{(2)}} + \underbrace{s \cdot u}_{\text{(2)}} = xy'.
\]

(3)

The security roughly follows from the case study below.

- When \( sk = u \) is secret, DDH assumption implies that \( ct^* = [s, \bar{u} + wy', w + x'_b] \) where \( \bar{u} \) are independent and uniformly distributed, and the security follows from the fact that \((\bar{u} + wy', w + x'_b) \equiv (\bar{u}, w + x'_b) \equiv (\bar{u}, w) \) hides \( x'_b \) in its entirety.
- When \( sk = u \) is leaked, DDH assumption implies that \( ct \approx [s, su + wy', \tilde{w} + x'_b] \) where we cannot change \( su \) to \( \bar{u} \) as before. In this case, we do not expect that \( ct \) hides \( x'_b \); instead, we can argue that that \((\tilde{w}y', \tilde{w} + x'_1) \approx_s (wy', \tilde{w} + x'_1) \) since \( x'_0y' = x'_1y' \).

This simple scheme is the so-called one-slot Reg-IPFE. The L-slot Reg-IPFE is the “sum” of \( L \) parallel instances of the above one-slot Reg-IPFE (namely, with fresh \( w \) for each slot) that ensures compact \( mpk \) (and \( ct \) as well):

- \( crs = [w_1, \ldots, w_L] \) is the concatenation of \( crs \)'s from \( L \) fresh one-slot Reg-IPFE instances, i.e., \( crs_i = [w_i] \) for all \( i \in [L] \), and the \( i \)-th user has key pair \((pk_i, sk_i) = ([u_i], u_i) \) with \( u_i \leftarrow \mathbb{Z}_p \) for all \( i \in [L] \).
- To register \( R = ((pk_1, y_1), \ldots, (pk_L, y_L)) \), the aggregator generates the corresponding master public key \( mpk_R = [\sum_j (u_j + w_jy'_j), \sum_j w_j] \) where index \( j \) ranges from 1 to \( L \); this sums up all \( mpk_{pk_i,y_i} \) in the one-slot Reg-IPFE with \( crs_i \) for all \( i \in [L] \).
- Under this \( mpk_R \), one encrypts \( x \) as \( ct = [s, s \sum_j (u_j + w_jy'_j), s \sum_j w_j + x] \); this is analogous to the encryption procedure in the one-slot scheme.

However “addition” of \( L \) one-slot Reg-IPFE breaks the correctness: a user even holding the correct secret key cannot decrypt as in the one-slot setting. Analogous to [ZZGQ23], we turn to bilinear groups and use source group \( G_2 \) to accommodate the helper keys. Let \( G_1 = \langle g_1 \rangle \), \( G_2 = \langle g_2 \rangle \) be finite cyclic source groups of bilinear maps \( e \) and \( G_7 \) be the target group; the order of all groups is prime \( p \). We place the above parallel instances over \( G_1 \), and define helper keys:

\[
hsk_i = [r_i, r_i \sum_{j \neq i} (u_j + w_jy'_j), r_i \sum_{j \neq i} w_j]_2, \quad i \in [L].
\]

Observe that, for each \( i \in [L] \), \( hsk_i \) over \( G_2 \) helps to recover a ciphertext of the same message \( x \) over \( G_7 \) in the one-slot Reg-IPFE instance under \( mpk_{pk_i,y_i} \) (generated from \( crs_i \) and \( pk_i, y_i \)) with random coin \( sr_i \) instead of \( s \):

\[
\begin{align*}
\underbrace{r_i \cdot (s \sum_j w_j + x)}_{ct} - \underbrace{s \cdot (r_i \sum_{j \neq i} w_j)}_{hsk_i} &= sr_i w_i + x \\
\underbrace{r_i \cdot (s \sum_j (u_j + w_jy'_j))}_{ct} - \underbrace{s \cdot (r_i \sum_{j \neq i} (u_j + w_jy'_j))}_{hsk_i} &= sr_i (u_i + w_iy'_i)
\end{align*}
\]

Given \( sk_i = u_i \) and \( y_i \), decryption then works as in the one-slot scheme over \( G_7 \), cf. (3). Note that the helper keys \( hsk_1, \ldots, hsk_L \) are generated by the curator during the registration and \( crs \) will contain terms \([r_i, r_i w_j]_2\), where \( i, j \) ranges from 1 to \( L \) with the restriction that \( i \neq j \); this ensures that all helper keys can be computed publicly and deterministically.
Recall that \(c_t\) follows from (a) subgroup decision assumption \(x\) the dual-system method, see Fig 2a. We begin with a challenge ciphertext of \(\text{sk}_x\) encryption of \(h\text{sk}_x\) \(p\) has extensive applications in IPE with full attribute-hiding features [OT12,CGKW18,CGW18]. In this overview, we will explain the idea using bilinear group \((G_1,G_2,G_T,e)\) of composite order \(N = p_1 p_2 p_3 p_4\) where \(p_1, p_2, p_3, p_4\) are prime. For each \(y \in \{1, 2, T\}\), group \(G_y\) can be decomposed as \(G_{y,1} \times G_{y,2} \times G_{y,3} \times G_{y,4}\) where the four subgroups have orders \(p_1, p_2, p_3, p_4\), respectively. For \(\sigma \in \{4\}\), let \(G_{y,\sigma} = \langle G_{y,\sigma} \rangle\). We will use implicit representation analogous to the prime-order group: for each \(y \in \{1, 2, T\}\) and \(S \subseteq \{1, 2, 3, 4\}\), we will write \([x]_y^S = \prod_{\sigma \in S} g_{y,\sigma}^x\). As usual, this applies to matrices and vectors. When \(|S| = 1\), i.e., \(S = \{\sigma\}\), we may simplify the notation as \([x]_y^\sigma\). We quickly review properties of composite-order bilinear groups:

- orthogonality: for \(\sigma, \delta \in \{1, 2, 3, 4\}\), we have \(e([1]_y^\sigma, [1]_y^\delta) = [0]_T\) when \(\sigma \neq \delta\);
- non-degenerate: for \(\sigma, \delta \in \{1, 2, 3, 4\}\), we have \(e([1]_y^\sigma, [1]_y^\delta) \neq [0]_T\) when \(\sigma = \delta\).

The common computational assumption is subgroup decision assumption indicating indistinguishability between random samples from two specific subgroups. We will give concrete assumptions when we use them in the proof.

Proof in Composite-order Groups. We embed our warm-up scheme into subgroups of order \(p_1\) and \(p_2\):

\[
\text{crs} = [w_j]_1^1, \quad \forall j \in [L];
\]

\[
[r_i, r_jw_j]_2^{[1,4]}, \quad \forall (i, j) \in [L] \times [L] \text{ s.t. } i \neq j;
\]

\[
\text{mp}_{k_T} = \left[ \sum_j (u_j + w_j y_j), \sum_j w_j \right]_1^1;
\]

\[
\text{sk}_i = [r_i, r_i \sum_{j \neq i} (u_j + w_j y_j), r_i \sum_{j \neq i} w_j]_2^{[1,4]};
\]

\[
\text{ct} = [s, s \sum_j (u_j + w_j y_j), s \sum_j w_j + s x_0^0]_1^1.
\]

where \(\text{sk}_i = u_i \leftarrow \mathbb{Z}_N\) and \(p_{k_i} = ([u_i]_1, \{[r_ju_j]_2^{[1,4]}\}_{j \neq i})\) for all \(i \in [L]\). Here we replace \(x\) with \(sx\), highlighted with a dashed box; the reader will see the reason later. Also, we will assume that the message \(x\) is sufficiently small so that \(x \mod p_1 = x \mod p_2 = x \mod p_3 = x \mod p_4\) (e.g., \(x \in B^n\) where \(B = \{1, \ldots, \min\{p_1, p_2, p_3, p_4\}\}\). This is a restriction applied to our composite-order group but not to our prime-order scheme.

Dual-system Method. Recall that \(x_0^0, x_1^0\) are challenge messages. Let \(([u_i]_1, y_i)\) be with slot \(i \in [L]\) and assume \(u_1, \ldots, u_L\) are all honestly chosen (but can be leaked to the adversary later). From a very high level, we will follow the dual-system method, see Fig 2a. We begin with a challenge ciphertext of \(x_0^0\) where \(b\) is the secret bit, see \(G_0\).

- First, we change the challenge ciphertext as below, which corresponds to \(G_2\):

\[
[s, s \sum_j (u_j + w_j y_j), s \sum_j w_j + s x_0^0]_1^1; \quad [s, s \sum_j (u_j + w_j y_j), s \sum_j w_j + s x_0^0]_1^{[3,4]}.
\]

- Second, we change all \(\text{sk}_i\)'s to the following form, which corresponds to \(G_3\):

\[
[r_i, r_i \sum_{j \neq i} (u_j + w_j y_j), r_i \sum_{j \neq i} w_j]_2^{[1,4]}; \quad [r_i, r_i \sum_{j \neq i} (u_j + w_j y_j), r_i \sum_{j \neq i} w_j]_2^{[1,4]}.
\]

By this, we let \(c_t\) and all \(\text{sk}_i\) interplay only through \(p_2\)-subgroup where \(x_0^0\) is in the place of \(x_1^0\). This allows us to reach a challenge ciphertext of \(x_0^0\), i.e., \(G_4\), via a simple argument that makes use of the absence of \(p_3\) and \(p_4\)-components in \(\text{sk}_i\), \(\ldots, \text{sk}_L\). To complete the proof, we need to justify the above two bullets. The first bullet follows from (a) subgroup decision assumption \([s_1]_1^{[1]} \approx c [s_1]_1^{[2,3,4]}\) given \([1]_1^{[1]}, [1]_1^{[2,3,4]}\) that “moves” \(p_1\)-component of \(c_t\) to \(p_2p_3p_4\)-subgroup, i.e., \(G_0 \mapsto G_1\), and (b) the same argument above but over \(p_2\)-subgroup instead of \(p_3\) and \(p_4\)-subgroup, see \(G_1 \mapsto G_2\). The second bullet is intended to work in a one-by-one fashion and uses nested dual-system method sketched below.
<table>
<thead>
<tr>
<th>Gm</th>
<th>Grp</th>
<th>ct*</th>
<th>hsk</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go</td>
<td>p1</td>
<td>x_b</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p2</td>
<td>—</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p3</td>
<td>—</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p4</td>
<td>—</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Go</td>
<td>p1</td>
<td>—</td>
<td>✓</td>
<td>Real Game</td>
</tr>
<tr>
<td></td>
<td>p2</td>
<td>—</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>p3</td>
<td>—</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p4</td>
<td>—</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

| G1 | p1  | x_b | ✓   |    |
|    | p2  | —   |   ✓|    |
|    | p3  | —   | x_b|    |
|    | p4  | —   | ✓  |    |

SD: p1 → p2p3p4

| G2 | p1  | x_b | ✓   |    |
|    | p2  | —   |   ✓|    |
|    | p3  | —   | x_b|    |
|    | p4  | —   | ✓  |    |

Statistical

| G3 | p1  | x_b | ✓   |    |
|    | p2  | —   |   ✓|    |
|    | p3  | —   | x_b|    |
|    | p4  | —   | ✓  |    |

Fig. 2: Game sequence in the composite-order group. For each game, we show the challenge ciphertext ct* and helper keys hsk in the template: \([s, s \sum_j (u_j + w_j y_j^q), s \sum_j w_j + sx_b^\ell]\) and \([r_s, r_i \sum_{j \neq i} (u_j + w_j y_j^q), r_i \sum_{j \neq i} w_j y_j^q]\) where \(S, S_1, \ldots, S_t \in \mathbb{Z}^4\).

- If slot \(\ell\) is honest, then \(u_\ell \mod p_3\) is hidden and thus \(w_\ell \mod p_3\) hides \(x_b^\ell\).
- If slot \(\ell\) is corrupted, then we have \(w_\ell y_\ell = w_\ell x_b^\ell + w_\ell + x_b^\ell \mod p_3\) with the restriction \(x_b^\ell y_\ell = x_b^\ell y_\ell\).

This corresponds to \(G_{3, \ell, 1} \rightarrow G_{3, \ell, 3}\) in Figure 2b. By this, we have \(x_b^\ell\) over both \(p_{2^e}\) and \(p_{3}\)-subgroup, and the subgroup decision assumption \([r_\ell^3 \approx_c r_\ell^2]\) gives desired form of hsk. Finally, we roll back the \(p_{3}\)-component of ct to encrypt \(x_b^\ell\) for future use, i.e., for handling hsk_{\ell+1} in the next loop.
Our Scheme in the Prime-order Group. Neglecting subscripts $i$, $j$, we do the following substitution with basis $A \in \mathbb{Z}_p^{k \times (k+1)}$ and $B \in \mathbb{Z}_p^{(2k+1) \times k}$ as in [CGW18]:

\[
\begin{align*}
 w &\in \mathbb{Z}_p^n \mapsto W \in \mathbb{Z}_p^{(k+1) \times (2k+1)n} \\
 [s]_i^1 &\in G_{1,1}, [r]_2^{[1,4]} \in G_{2,1} \times G_{2,4} \mapsto [sA]_1 \in G_{1}^{1 \times (k+1)}, [Br]_2 \in G_2^{2k+1} \\
 [w]_1 &\in \mathbb{Z}_p^{2k+1}, [sw]_1 \in G_{1,1} \mapsto [AW]_1 \in \mathbb{G}_1^{1 \times (2k+1)n}, [sAW]_1 \in \mathbb{G}_1^{1 \times (2k+1)n} \\
 [rw]_2^{[1,4]} &\in (G_{2,1} \times G_{2,4})^n \mapsto [W(I_n \otimes Br)]_2 \in \mathbb{G}_2^{(k+1)n}
\end{align*}
\]

This yields our $L$-slotted Reg-IPFE scheme in the prime-order group:

\[
\begin{align*}
\text{crs} &= \{[AV]_1, [AW]_1, \forall j \in \mathcal{L}\} \\
[Br]_2, [W_j(I_n \otimes Br)]_2, &\quad \forall (i, j) \in \mathcal{L} \times \mathcal{L} \text{ s.t. } i \neq j \\
\text{mpk}_R &= \{[\sum_j (AW_j)(y_j^i \otimes I_{2k+1})], \sum_j AW_j, AV\}_1 \\
\text{hsk}_k &= \{[Br]_2, \sum_{j \neq i} (U_j Br_j^i + W_j (y_j^i \otimes Br_j^i)), \sum_{j \neq i} W_j (I_n \otimes Br_j^i), [W_0 Br_j^i + k^i]_2\}_2 \\
\text{ct} &= \{sA, s \sum_j (AU_j + AW_j (y_j^i \otimes I_{2k+1})), s \sum_j AW_j + x \otimes sAV, sAV\}_1
\end{align*}
\]

where $sk_i = U_i \leftarrow \mathbb{Z}_p^{(k+1) \times (2k+1)}$ and $pk_i = \{\{AU\}_1, \{U_i Br_j \}_{j \neq i}\}$ for all $i \in \mathcal{L}$. This is almost the final scheme except for some extra elements for handling malicious public keys. All subgroup decision assumptions we used can be replaced by MDDH assumption. Roughly, basis $B$ corresponds to a $(2k+1)$-dimensional space; we use two $k$-dimensional subspaces to simulate $p_3$, $p_4$-subgroup, respectively, and the remaining 1-dimensional subspace to simulate $p_2$-subgroup. We leave more details to Section 3. For simplicity, we will continue our technical overview based on this slightly weaker scheme.

Extension to Reg-IPE. Recall that the proof of our slotted Reg-IPFE is motivated by that for IPE with full attribute hiding [OT12,CGKW18,CGW18]. This similarity inspires the following $L$-slotted Reg-IPE:

\[
\begin{align*}
\text{crs} &= \{[AK]_{\mathcal{T}}, [AV]_1, [AW]_1, [AW_0]_1, [AW]_1, \forall j \in \mathcal{L}\} \\
[Br]_2, [W_0 Br_j + k]_2 W_j (I_n \otimes Br_j) \}_2, &\quad \forall (i, j) \in \mathcal{L} \times \mathcal{L} \text{ s.t. } i \neq j \\
\text{mpk}_R &= \{[AW_0]_1 + \sum_j (AU_j + AW_j (y_j^i \otimes I_{2k+1})), \sum_j AW_j, AV\}_1 \\
\text{hsk}_k &= \{[Br]_2, \sum_{j \neq i} (U_j Br_j^i + W_j (y_j^i \otimes Br_j^i)), \sum_{j \neq i} W_j (I_n \otimes Br_j^i), [W_0 Br_j^i + k]_2\}_2 \\
\text{ct} &= \{sA, sAW_0 + s \sum_j (AU_j + AW_j (y_j^i \otimes I_{2k+1})), s \sum_j AW_j + x \otimes sAV, [sAK]_{\mathcal{T}} \cdot m\}
\end{align*}
\]

where $sk_i$ and $pk_i$ are as in (4). We highlight the difference between our slotted Reg-IPFE and the slotted Reg-IPE above. The “powers-of-two” technique finally gives us a Reg-IPE scheme. We leave all details to Appendix B.

1.3 Reg-QFE from Bilateral k-Lin

This section explains our Reg-QFE where each user registers a quadratic function $f \in \mathbb{Z}_p^{n_1 \times n_2}$ so that decrypting a ciphertext of $(x_1, x_2) \in \mathbb{Z}_p^{n_1} \times \mathbb{Z}_p^{n_2}$ recovers $(x_1 \otimes x_2) f^i$. Given (a) our slotted Reg-IPFE in Section 1.2 and Section 3, (b) Wee’s “IPFE-to-QFE” transformation [Wee20] and (c) “powers-of-two” transformation [GHMR18,HLWW23,FFM’23,DP23], we want to follow the technical line:

\[
\begin{align*}
\text{slotted Reg-IPFE} \overset{(a)}{\Rightarrow} \text{slotted Reg-QFE} \overset{(b)}{\Rightarrow} \text{Reg-QFE} \overset{(c)}{\Rightarrow} \text{Reg-QFE}.
\end{align*}
\]

This defers complicated update procedure to the very last stage and keeps most steps simple. Only the first “$\Rightarrow$” in technical line (5) can be problematic since the transformation is not for Reg-FE but FE. Let us begin with a sketch of Wee’s transformation.
Recap. Given an IPFE \((i\text{Key}, i\text{Enc})\), the QFE scheme from [Wee20] works as follows:

\[
\text{mpk} = [A_1]_1, [A_2]_2, \quad \text{ct} = [y_1]_1, [y_2]_2, i\text{Enc}(x), \quad \text{sk}_f = i\text{Key}([\mathbf{Mf}]_2)
\]

where we sample random coins \(s_1, s_2\) and set \(y_1 = x_1 + s_1 A_1, y_2 = x_2 + s_2 A_2, x = (s_1 \otimes x_2 \parallel s_2 \otimes x_1)\) and

\[
\mathbf{M} = \begin{pmatrix} A_1 \otimes I_n & I_n \otimes A_2 \\ I_n \otimes A_2 & A_1 \otimes A_2 \end{pmatrix}.
\]

Note that this is slightly different from the original scheme in [Wee20] but they are essentially the same. The correctness follows from

\[
(y_1 \otimes y_2)^f = (x_1 \otimes x_2)^f + x\mathbf{Mf}
\]

The selective SIM-security requires a simulator \((i\overline{\text{Enc}}, i\overline{\text{Key}})\) so that we can carry out the following argument:

\[
\begin{align*}
& [y_1]_1, [y_2]_2, i\overline{\text{Enc}}(x), i\overline{\text{Key}}([\mathbf{Mf}]_2) \\
\approx_c & [y_1]_1, [y_2]_2, i\overline{\text{Enc}}(x), i\overline{\text{Key}}([\mathbf{Mf}]_2, [x\mathbf{Mf}]_2) \text{ // IPFE} \\
\equiv & [y_1]_1, [y_2]_2, i\overline{\text{Enc}}(x), i\overline{\text{Key}}([\mathbf{Mf}]_2, [(y_1 \otimes y_2)^f - (x_1 \otimes x_2)^f]_2) \text{ // (6)} \\
\approx_c & [\overline{y}_1]_1, [\overline{y}_2]_2, i\overline{\text{Enc}}(x), i\overline{\text{Key}}([\mathbf{Mf}]_2, [(\overline{y}_1 \otimes \overline{y}_2)^f - (x_1 \otimes x_2)^f]_2) \text{ // bi-MDDH}
\end{align*}
\]

where \(\overline{y}_1 \leftarrow \mathbb{Z}_p^{n_1}\) and \(\overline{y}_2 \leftarrow \mathbb{Z}_p^{n_2}\) are independently and uniformly distributed. Here, the first \(\approx_c\) uses the simulator over groups to embed the result \(z' = x\mathbf{Mf}\) into the simulated key; the second \(\equiv\) follows from the equation for correctness; the third \(\approx_c\) is ensured by bilateral MDDH assumption w.r.t. \(A_1\) and \(A_2\).

Challenges. The first “\(\implies\)” in technical line (5) is expected to apply a similar transformation to our slotted Reg-IPFE in Section 1.2 and Section 3. However, there are three main challenges pertinent to both correctness and security:

- **Challenge 1: Decryption with Fixed Base.** Our slotted Reg-IPFE gives decryption result in the form of \([b, zb]_T\), where \(z\) is the result and base \(b = s\text{AVBr}_i\); here \(b\) varies with the user who are decrypting. This is fine in the case of small \(z\) since brute-force search recovers \(z\). For the use in Wee’s QFE, \(z\) involves random coins \(s_1, s_2\) and can be quite large, more precisely, \(z \in \mathbb{Z}_p\); clearly, we cannot extract \([z]_T\) in this case. Therefore, we need a slotted Reg-IPFE that recovers \([z]_T\) for all slots, i.e., with fixed base \([1]_T\).

- **Challenge 2: Group-friendly Registration.** A subtle point in Wee’s scheme is that we need to encode \(\mathbf{Mf}\) (and also \(y_1, y_2\)) over proper groups in order to apply bi-MDDH assumption later. In the context of slotted Reg-IPFE, this means that a user can register a function of the form \([\mathbf{Mf}]_2\) over \(G_2\). However our slotted Reg-IPFE already has \(\text{crs}\) and \(\text{mpk}\), \(\text{hsk}_1, \ldots, \text{hsk}_L\) over \(G_1, G_2\), there is no space to use pairing which seems to be inevitable if we want to “multiply” \(\mathbf{Mf}\) with terms from \(\text{crs}\). Therefore, we need a slotted Reg-IPFE with group-friendly registration.

- **Challenge 3: Simulation-based Security.** In the first step of Wee’s proof, we make use of the simulator to assemble \(x\) and \(\mathbf{Mf}\) in \(\text{sk}_f\). In fact, when serving as a building block, we prefer an IPFE with SIM-security. However, our slotted Reg-IPFE scheme only achieves strictly weaker IND-security. Furthermore, Wee’s QFE requires that the simulator takes \([\mathbf{Mf}]_2\) and \([x\mathbf{Mf}]_2\) as well; this is a requirement related to Challenge 2.

Therefore, we need a slotted Reg-IPFE achieving SIM-security with group-based simulator.

We will explain our solutions to all three challenges. Note that, in [Wee20], all above are easy to satisfy since the underlying IPFE [ALS16] is pairing-free and embedding it into a bilinear group simply works; however, our slotted Reg-IPFE already uses bilinear groups and those embedding tricks fail.

---

6 Here we hardcode the master public key and master secret key inside \(i\text{Enc}\) and \(i\text{Key}\), respectively, for notation simplicity.
Solution 1: Decryption with Fixed Base. Let us review the structure of ciphertext in our slotted Reg-IPFE (4):

\[
[sA, s \sum_j(AU + AW_j(y'_j \otimes I)), s \sum_j AW_j + x \otimes sAV, sAV]_1
\]

The reason of decryption with variable base is that we put \(x\) with term \(\sum_j sAW_j\), highlighted in the gray box, which involves terms from all slots; during the decryption, it interplays with \(Br'_i\) depending on the user/slot. Our revision starts from the following substitution:

\[
s \sum_j AW_j + x \otimes sAV \mapsto sAW + x
\]

namely, we simply hide \(x\) using \(sAW\) where \(W\) is conceptually shared by all users/slots. In fact, this is exactly the ciphertext by Agrawal et al’s IPFE [ALS16] that is compatible with Wee’s QFE. This breaks the correctness, but a minor modification saves us: we remove terms involving \(V\) and we put \([AW]_1\) into \(crs\) and \(mpk\) for encryption; the most crucial change is the substitution

\[
W_j(I_n \otimes Br'_i) \mapsto W_j(I_n \otimes Br'_i) + W
\]

which connects the two terms in the ciphertext. This yields the following scheme with new terms highlighted in the boxes:

\[
\begin{align*}
ct &= [sA, s \sum_j(AU + AW_j(y'_j \otimes I)), s \sum_j AW_j + x \otimes sAV, sAV]_1 \\
mpk_k &= \sum_j(AU + AW_j(y'_j \otimes I)), [AW]_1 \\
hsk_k &= [Br'_i, \sum_j(I_k Br'_j + W_j(y'_j \otimes Br'_i)), W_i(I_n \otimes Br'_i) + W]_2 \\
\end{align*}
\]

where \((pk_k, sk_k)\) is in the same form as in (4). Here we leave some \(I\)s with dimension undefined for now. For correctness, revised terms (with boxes) in \(hsk_k\) and \(ct\) now interplay as below during decryption:

\[
-sA \cdot (W_i(y'_i \otimes Br'_i) + Wy'_i) + (sAW + x) \cdot y'_i = -sAW_i(y'_i \otimes Br'_i) + xy'_i
\]

while the remaining unchanged terms give \(sAU_iBr'_j + sAW_i(y'_j \otimes Br'_i)\) as before; they are sufficient to recover \(xy'_i\) for the legitimate user holding \(sk_k = U_i\). For security, our revised scheme only achieves a selective variant where the adversary claims the challenge message before seeing \(crs\). The good news is we will not need the complex “nested dual-system” technique — the standard dual-system method is already sufficient as in prior Reg-ABE [HLWW23,ZZGQ23]. In a bit more detail, after changing the challenge ciphertext to the form:

\[
ct' = [cC, \sum_j(cC U_j + cC W_j(y'_j \otimes I)), cC W + x'_j]_1
\]

via MDDH assumption w.r.t. \(A\), we can “embed” \(x'_j\) to \(crs\) via the substitution: \(W \mapsto W - c^\perp x'_j\) where \(cc^\perp = 1\) but \(cA = 0\). Then, we can prove the security via the dual-system method: \(x'_j\) now conceptually locates in \(hsk_k\), \(sk_k\), and we can handle them one-by-one. Therefore, we only need \(B\) to be a vector of dimension \(k + 1\) following [CGW15] and this requires the size of \(I\) to be \((k + 1) \times (k + 1)\). (Sizes of related matrices should be changed accordingly.) One can see that embedding \(x'_j\) into \(crs\) is what renders the scheme selectively secure. This addresses **Challenge 1** at the cost of weaker security. Even though, this is sufficient for our purpose; in fact, we do not know how to achieve adaptive security even for plain QFE (from standard assumptions).

Solution 2: Pre-Constrained Registered IPFE. Recall that **Challenge 2** requires group-friendly registration. We believe this problem is quite hard in general and focus on functions in the specific form \([\text{MF}]_2\). Observe that
Our slotted Reg-IPFE with fixed base, i.e., (7), can be easily modified to give an instance of PReg-IPFE:

We do not need to wait and ask the curator to register $[Mf_i]_2$; instead, we can embed the “troublemaker” $[M]_2$ over group into crs beforehand in the setup phase and ask the curator to register only $y$ over integers.

We capture this idea by introducing a new notion called Pre-Constrained Registered IPFE (PReg-IPFE):

- crs is generated with $M$ where $M$ is sampled from a pre-defined distribution.
- For each $i \in [L]$, the user holding $(pk_i, sk_i)$ can register $(pk_i, f_i)$ to slot $i$.
- Given a ciphertext of $y$, decrypting it using $sk_i$ (and hsk) gives $[xMf_i]_T$.

This generalizes the notion of slotted Reg-IPFE. Conceptually, $y_i = Mf_i$ is the function for slot $i$. Imagine that $M$ is a “tall” matrix defining a subspace. Our PReg-IPFE forces that all user’s functions $y_1, \ldots, y_L$ should be in span$(M)$.

Our slotted Reg-IPFE with fixed base, i.e., (7), can be easily modified to give an instance of PReg-IPFE:

$$\begin{align*}
\text{crs}_M &= [AW]_1, [AW/f[M \otimes I_{k+1}]]_1 & \forall j \in [L] \\
[Br_j, W_j ((M \otimes Br_j), W_j ((M \otimes Br_j) + W[M]_2), \forall (i, j) \in [L] \times [L] \text{ s.t. } i \neq j \\
\text{mpk}_R &= [\sum_j (AU_j + AW/j[Mf_i \otimes I_{k+1}]), AW]_1 \\
\text{hsk}_i &= [Br_i, \sum_{j \neq i} (U_i Br_i + W_j ((Mf_i \otimes Br_j) + W[Mf_i]_2) \\
\text{ct} &= [sA, s \sum_j (AU_j + AW/j[Mf_i \otimes I_{k+1}]), sAW + x]_1
\end{align*}$$

(8)

where $(pk_i, sk_i)$ is as in (7) and we highlight the difference with (7) using boxes. To reach this scheme from (7), we simply set $y'_i = Mf_i$ for all $i \in [L]$ in $\text{hsk}_i$ and $\text{ct}$ from (7) and rebuild $\text{crs}_M$ and $\text{mpk}_R$ with $M$ embedded. In fact, one can see that setting $M = I_0$ degrades it to the original scheme (7). Clearly, correctness and selective security can be proved analogously, but the registration now only involves crs and $R = ((pk_1, f_1), \ldots, (pk_L, f_L))$ and has nothing to do with $M$. Furthermore, we can check that if we publish $[M]_2$ in crs and $[Mf_i]_2$ in $\text{hsk}_i$ (which is used to compute $[(sAW + x) \cdot Mf_i]_T$ during decryption), then all occurrences of $M$ have been encoded over groups. This addresses Challenge 2.

**Solution 3.1: Defining Simulation-based Security.** We begin to work on Challenge 3. This is the first time to consider simulation-based security (SIM-security) in the context of Reg-FE. Since we will work on scheme (8), our discussion will be restricted to PReg-IPFE and we will not pursue security stronger than selective security. Another technical reason is that there is no IPFE scheme supporting group-based functions with adaptive SIM-security. Assume $x^*$ is the challenge message and registration $R = ((pk_1, f_1), \ldots, (pk_L, f_L))$, the SIM-secure PReg-IPFE requires a simulator that can simulate the view of adversary using $Z = \{x^* Mf_i\}_{i \in C}$ where $C \in [L]$ is the set of corrupted slots. Inspired by the selective SIM-security of plain IPFE, we expect the simulator to embed $Z$ into $\text{hsk}_1, \ldots, hsk_L$. However, in the PReg-IPFE system, $\text{hsk}_1, \ldots, hsk_L$ is generated by aggregator under the supervision of adversary, hence the simulator has no chance to embed anything inside $\text{hsk}_1, \ldots, hsk_L$. In this work, we embed $Z$ into $\text{crs}$ which is fully controlled by the simulator and is used to generate $\text{hsk}_1, \ldots, hsk_L$; we note that this will also require the adversary to claim the challenge $x^*$, the set $C$ along with corresponding $f_i, i \in C$ at the beginning so that $Z$ is well-defined during the setup phase. This is analogous to the very selective security in the setting of ABE [AMY19] where the adversary claims the challenge and all key queries at the beginning. We finally mention that the adversary is still free to choose $pk_1, \ldots, pk_L$ after seeing $\text{crs}$. 

11
Solution 3.2: Simulation-based Security via PReg-IPFE. Roughly, we will make use of pre-constrained registration in (8) to implement the idea of function-hiding IPFE [LV16] and slotted IPFE [LL20]: instead of embedding a private “slot” into a key, we will embed this private “slot” into $M$ in crs. (Note that the “slot” here has different means with the slots in the context of Reg-IPFE.) For this, we first extend the notion of pre-constrained registration:

- $crs$ is generated along with $[M_1, \ldots, M_L]_2$ for the $L$ slots, respectively.
- Decrypting a ciphertext of $x$ using $sk_i$ for slot $i$ gives $[xM_i f_i]_T$ for all $i \in [L]$. 

A minor revision of (8) below already works with analogous correctness and selective IND-security.

\[
\begin{align*}
    crs &= [AW]_1, [AW_j([M_j] \otimes I_{k+1})]_1, [M_j]_2, \quad \forall j \in [L] \\
    &\quad [Br_j, W_j([M_j] \otimes Br_j')], W_j([M_j] \otimes Br_j') + W[M_j]_2, \quad \forall (i, j) \in [L] \times [L] \text{ s.t. } i \neq j \\
    mpk_r &= \{\sum_j (U_j + AW_j([M_j] f_j \otimes I_{k+1})), AW\}_1 \\
    hsk_r &= [Br_j, \sum_{j \neq i} (U_j Br_j' + W_j([M_j] f_j \otimes Br_j'))], W_j([M_j] f_j \otimes Br_j') + W[M_j f_j]_2 \\
    ct &= [sA, s \sum_j (U_j + AW_j([M_j] f_j \otimes I_{k+1})), sAW + x]_1
\end{align*}
\]

where $(pk_i, sk_i)$ is as in (8).

**Scheme.** We achieve SIM-security from IND-security as follows: we use the scheme (9) with the following special $[M_1, \ldots, M_L]_2$ in $crs$:

\[
[M_j]_2 = \begin{pmatrix} [M]_2 & [0]_2 \\ [0]_2 & Enc(pk, 0) \end{pmatrix}
\]

where $(Gen, Enc, Dec)$ is a PKE with linear decryption whose message space serves as the private “slot”. For concreteness, we leave formal definition in Section 2.5 and mention that ElGamal PKE with ciphertexts over $\mathbb{G}_2$ suffices:

\[
pk = [A, wA]_2, \quad sk = (-w, 1) \in \mathbb{Z}_p^{1 \times (k+2)}, \quad Enc(pk, x) = \begin{pmatrix} [As]_2 \\ [x + wAs]_2 \end{pmatrix} \in \mathbb{G}_2^{k+2}
\]

where $A \leftarrow \mathbb{Z}_p^{(k+1) \times k}$, $w \leftarrow \mathbb{Z}_p^{1 \times (k+1)}$, $s \leftarrow \mathbb{Z}_p^k$ and it is easy to verify the linear decryption. Accordingly,

- in the registration phase, given $R = ((pk_1, f_1), \ldots, (pk_L, f_L))$, we register extended $\tilde{R} = ((pk_i, \tilde{f}_i), \ldots, (pk_L, \tilde{f}_L))$ where $\tilde{f}_i = (f_i || 1)$ for all $i \in [L]$;
- to encrypt $x$, we encrypt extended message $\tilde{x} = (x || 0)$.

The correctness follows from the fact that $xM_i \tilde{f}_i = xM_i f_i$ for all $i \in [L]$.

**Simulator & Proof.** Let us sketch the idea to simulator. Given $f_i$ and $xM_i f_i$ for all $i \in C$, we first change $M_i$ to $\tilde{M}_i$ for all $i \in C$ and then switch $\tilde{x}$ to $\tilde{x}$ where

\[
\tilde{M}_i = \begin{pmatrix} M & 0 \\ 0 & Enc(pk, xM_i f_i) \end{pmatrix} \quad \text{and} \quad \tilde{x} = (0 || sk)
\]

Here, the first $M_i \mapsto \tilde{M}_i$ step follows from the security of $(Gen, Enc, Dec)$, the second $x \mapsto \tilde{x}$ step follows from the selective IND-security of (9) by the fact that, for all $i \in C$, we have

\[
\tilde{x}M_i \tilde{f}_i = sk \cdot Enc(pk, xM_i f_i) = Dec(sk, Enc(pk, xM_i f_i)) = xM_i f_i = \tilde{x}M_i \tilde{f}_i
\]

Here we do not need to maintain a similar relation for the case $i \in H$. At this point, we can simulate everything without knowing $x$ but $xM_i f_i$ for all $i \in C$. This yields a very selective simulator. Furthermore, we embed the results $Z$ into $M_1, \ldots, M_L$ which are over groups as in (8). This addresses **Challenge 3.**
**Final Scheme with Compact Ciphertexts.** Putting all these together, the technical line depicted in (5) works but leads to a Reg-QFE with ciphertexts of size $O(n \log L)$. The reason is: to move from $L$-slotted Reg-QFE to Reg-QFE with $L$ slots, “powers-of-two” transformation runs $\log L$ parallel instances of slotted Reg-QFE. This means we encrypt the same message $(x_1, x_2)$ of size $2n$ for $\log L$ times (in the worst case). To avoid this, we let all $\log L$ instances have shared $A_1$ and $A_2$ in crs and encrypt the message once for all. However this is not enough: the $\log L$ underlying slotted PReg-IPFE instances encrypt the same $x = (s_1 \otimes x_2 \| x_1 \otimes s_2 \| s_1 \otimes s_2)$ for $\log L$ times. To fix this, we let all instances share $W$ in crs and encrypt $x$ once for all with shared random coin $s$; by this, all instances share the term $sAW + x$ in ciphertexts, cf. (9). Formally, we introduce the multi-instance variants of scheme (9) and update the technical line (5) as follows:

\[
\text{multi-instance slotted PReg-IPFE} \implies \text{multi-instance slotted Reg-QFE} \implies \text{compact Reg-QFE} \quad (10)
\]

This line gives Reg-QFE with ciphertexts of size $O(n + \log L)$ and we consider this as our main result (2).

**More Results & Roadmap.** In Section 5, we treat the second “ $\implies$ ” in the new technical line (10) as a general transformation. For more details, we give the definitions of multi-instance slotted Reg-FE for general functions in Section 5.1; and present the transformation from multi-instance slotted Reg-FE to compact Reg-FE for general functions in Section 5.2. This leads to result (ii):

- Setting $M_i = I$ for all $i \in [L]$ (i.e., scheme (7)) gives IND-secure multi-instance slotted Reg-IPFE scheme, with fixed decryption base, c.f. Section 6.5. With the generic transformation in Section 5.2, it leads to the IND-secure compact Reg-IPFE.
- Setting $M = I$ gives us the SIM-secure multi-instance slotted Reg-IPFE scheme, c.f. Section 6.4. With the generic transformation in Section 5.2, it leads to the SIM-secure compact Reg-IPFE.

We summarize the roadmap of this part in Fig. 3, where “mi” means “multi-instance” for short.

![Roadmap of the technical part](image)

Fig. 3: Roadmap of the technical part. Here “mi” stands for “multi-instance”. Those solid arrows show transformations proposed in this work; the dashed arrow means a mild adaptation; the double-line arrow indicates an implication.

### 2 Preliminaries

**Notations.** For a finite set $S$, we use $s \gets S$ to denote the procedure of sampling $s$ from $S$ uniformly. For an ordered list or array $L$, we use $|L|$ to denote its size (i.e., the number of entries in the list) and use $L[i]$ 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 a 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., $a$) and upper-case boldface to denote matrices (e.g. $M$). We use span$(M)$ to denote the row span of $M$, and use basis$(M)$ to denote a basis of the column space of $M$. Let $\mathbb{F}$ be a field. We use $A \otimes B$ to denote Kronecker Product for matrices $A \in \mathbb{F}^{k\times m}$ and $B \in \mathbb{F}^{n\times p}$. For matrices $A, B, C, D$ of proper sizes, we have: $(A \otimes B)(C \otimes D) = AC \otimes BD$. We use $n \oplus m$ to denote XOR for numbers $n, m \in \mathbb{N}$.

### 2.1 Prime-Order Bilinear Groups

Assume an efficient $G$ that takes as input a security parameter $1^\lambda$ and outputs $G := (p, G_1, G_2, G_T, e)$. Here $G_1, G_2$ and $G_T$ are cyclic groups of prime order $p$, $e : G_1 \times G_2 \rightarrow G_T$ is a non-degenerate bilinear map, and all group operations and bilinear map are efficient. Let $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and $g_T = e(g_1, g_2)$, we employ implicit representation of group elements: for a matrix $M = (m_{ij})$ over $\mathbb{Z}_p$, define $[M]_s = g_M^s = (g_s^{m_{ij}})$ for all $s \in \{1, 2, T\}$; given $[A]_1, [B]_2$, we write $e([A]_1, [B]_2) = [AB]_T$.

**Assumption 1** ($(k, \ell, d)$-$\text{MDDH}$) [EHK’13] over $G_s$, $s \in \{1, 2\}$ Let $k, \ell, d \in \mathbb{N}$. We say that the $(k, \ell, d)$-$\text{MDDH}$ assumption holds if for all PPT adversaries $\mathcal{A}$, the following advantage function is negligible in $\lambda$.

$$\text{Adv}_{\mathcal{A},k,\ell,d}^{\text{MDDH}}(\lambda) = \left| \Pr[\mathcal{A}(G, [M]_s, [SM]_s) = 1] - \Pr[\mathcal{A}(G, [M]_s, [US]_s) = 1] \right|$$

where $G := (p, G_1, G_2, G_T, e) \leftarrow G(1^\lambda), M \leftarrow \mathbb{G}_p^{k\times\ell}, S \leftarrow \mathbb{G}_p^{d\times k}$ and $U \leftarrow \mathbb{G}_p^{d\times\ell}$.

It is shown that the assumption is implied by $k$-$\text{Lin}$ [EHK’13]. The *bilateral* $\text{MDDH}$ assumption is defined analogously with the advantage function:

$$\text{Adv}_{\mathcal{A},k,\ell,d}^{\text{b-MDDH}}(\lambda) = \left| \Pr[\mathcal{A}(G, \{[M]_s, [SM]_s\}_{s\in\{1,2\}}) = 1] - \Pr[\mathcal{A}(G, \{[M]_s, [US]_s\}_{s\in\{1,2\}}) = 1] \right|$$

### 2.2 Registered Functional Encryption (Reg-Fe)

**Algorithms.** A registered functional encryption [FFM’23,DP23] (Reg-Fe for short) for functionality $F = \{f : X \rightarrow Z\}$ consists of six algorithms:

- **Setup($1^\lambda$, $1^L$, $F$) $\rightarrow$ crs:** It takes as input security parameter $1^\lambda$, maximum number of users $1^L$, functionality $F$, outputs a common reference string crs.
- **Gen(crs, aux) $\rightarrow$ (pk, sk):** It takes as input crs and state aux, outputs key pair $(pk, sk)$.
- **Reg(crs, aux, pk, $f$) $\rightarrow$ (mpk, aux’):** It takes as input crs, aux, and pk along with function $f \in F$, outputs master public key mpk and updated state aux’.
- **Enc(mpk, $x$) $\rightarrow$ ct:** It takes as input mpk, $x \in X$, outputs a ciphertext ct.
- **Upd(crs, aux, pk) $\rightarrow$ hsk:** It take as input crs, aux, pk, outputs a helper key hsk.
- **Dec(hsk, ct, $\bot$) $\rightarrow$ $\bot$/getupd:** It take as input hsk, ct and outputs $\bot \in Z$ or a special symbol $\bot$ to indicate a decryption failure, or a special flag getupd to indicate the need of an updated helper key.

**Correctness, Compactness and Update Efficiency.** Correctness means, for all stateful (unbounded) adversary $\mathcal{A}$ making a polynomial number of oracle queries (defined below) and all $L$, the following advantage function is negligible in $\lambda$:

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

where oracles work as follows with aux = $\bot$, $E = \emptyset$, $R = \emptyset$ and $t = \bot$:

7 The $(k, \ell, d)$-$\text{MDDH}$ assumption holds unconditionally when $\ell > k$. 

---

14
where the oracles work as follows with initial setting 
\[ D[|R|, mpk, aux]; \]
- ORegNT(pk, f): run (mpk, aux') \leftarrow Reg(crs, aux, pk, f), update aux = aux', append (mpk, aux) to \mathcal{R} and return (|\mathcal{R}|, mpk, aux);
- ORegT(f'): run (pk', sk') \leftarrow Gen(crs, aux), (mpk, aux') \leftarrow Reg(crs, aux, pk', f'), update aux = aux', compute hsk' \leftarrow Upd(crs, aux, pk'), append (mpk, aux) to \mathcal{R}, return (t = |\mathcal{R}|, mpk, aux, pk', sk', hsk');
- OEnc(i, x): let \mathcal{R}[i] = (mpk, \star), run ct' \leftarrow Enc(mpk, x), append (x, ct) to \mathcal{E} and return (|\mathcal{E}|, ct);
- ODec(j): let \mathcal{E}[j] = (x_j, ct_j), compute z_j \leftarrow Dec(sk', hsk', ct_j); if z_j = getupd, run hsk' \leftarrow Upd(crs, aux, pk') and recompute z_j \leftarrow Dec(sk', hsk', ct_j). Set \( b = 1 \) when \( z_j \neq f^*(x_j) \).

with the following restrictions:

- there are at most \( L - 1 \) queries to ORegNT and there is exactly one query to ORegT; therefore, we will consider \( f^*, pk', sk', hsk' \) to be global;
- for query \((i, x)\) to OEnc, it holds that \( i \geq t, \mathcal{R}[i] \neq \perp \);
- for query \((j)\) to ODec, it holds that \( \mathcal{E}[j] \neq \perp \).

Compactness means that, for all mpk and hsk in the above, we have

\[ |mpk| = \text{poly}(\lambda, \text{par}, \log L), \quad |hsk| = \text{poly}(\lambda, \text{par}, \log L); \]

where par is a parameter depending on the functionality \( F \). Furthermore, update efficiency means that the number of invocations of Upd in ODec is at most \( O(\log |\mathcal{R}|) \) and each invocation costs \( \text{poly}(\log |\mathcal{R}|) \) time.

**Indistinguishability-based Security (IND-security).** For all stateful PPT adversary \( A \), the adaptive (resp., selective) indistinguishability-based security requires the advantage function \( \text{IndAdv}_{A}^{\text{Ad-Reg-FE}} \) (resp., \( \text{IndAdv}_{A}^{\text{Sel-Reg-FE}} \)) defined as follows is negligible in \( \lambda \):

\[
\text{IndAdv}_{A}^{\text{Ad-Reg-FE}}(\lambda) = \Pr \left[ b = b', \begin{array}{c}
\text{crs} \leftarrow \text{Setup}(1^\lambda, F);
\text{x}_0', \text{x}_1' \leftarrow A^{\text{ORegCK}(\cdot), \text{ORegHK}(\cdot), \text{OCorHK}(\cdot)}(\text{crs});
\end{array} \right] - 1/2,
\]

where the oracles work as follows with initial setting aux = \( \perp \), mpk = \( \perp \), \mathcal{H} = \emptyset, C = \emptyset and \mathcal{D} being a dictionary with \( \mathcal{D}[pk] = \emptyset \) for all possible pk:

- ORegCK(pk, f): run (mpk', aux') \leftarrow Reg(crs, aux, pk, f), update mpk = mpk', aux = aux', \mathcal{D}[pk] = \mathcal{D}[pk] \cup \{ f \}, append pk to \( C \) and return (mpk, aux);
- ORegHK(f): run (pk, sk) \leftarrow Gen(crs, aux) and (mpk', aux') \leftarrow Reg(crs, aux, pk, f), update mpk = mpk', aux = aux', \mathcal{D}[pk] = \mathcal{D}[pk] \cup \{ f \}, append (pk, sk) to \( \mathcal{H} \) and return (|\mathcal{H}|, mpk, aux, pk);
- OCorHK(i): let \( \mathcal{H}[i] = (pk, sk) \), append pk to \( C \) and return sk;

with the following restrictions:

- for query \( i \) to OCorHK, it holds that \( \mathcal{H}[i] \neq \perp \);
- for all \( f \in \bigcup_{pk \in \mathcal{C}} \mathcal{D}[pk] \), it holds that \( f(x_0') = f(x_1') \).

The selective IND-security is analogous to above definition of adaptive security, except that \( A \) claim the challenge \( x_0', x_1' \) at the beginning.
2.3 Slotted Registered Functional Encryption

Algorithms. A slotted Reg-FE (sReg-FE for short) for functionality $F = \{ f : X \to Z \}$ consists of six efficient algorithms:

- Setup($1^\lambda, 1^L, F$) $\to$ crs: It takes as input security parameter $1^\lambda$, maximum number of slots $1^L$, functionality $F$, and outputs a common reference string crs.
- Gen($\lambda$, i) $\to$ ($pk_i$, $sk_i$): It takes as input crs and slot number $i \in [L]$, outputs key pair ($pk_i$, $sk_i$).
- Ver($\lambda$, i, $pk_i$) $\to$ 0/1: It takes as input crs, i, $pk_i$ and outputs a bit.
- Agg($\lambda$, crs, ($pk_i$, $f_i$)$_i\in[L]$) $\to$ ($mpk$, ($hsk_k$)$_i\in[L]$): It takes as input crs and a series of $pk_i$ with $f_i \in F$ for all $i \in [L]$, outputs master public key $mpk$ and a series of helper keys $hsk_k$ for all $f_i \in [L]$.
- Enc($\lambda$, $mpk$, x) $\to$ ct: It takes as input $mpk$, $x \in X$, outputs a ciphertext ct.
- Dec($\lambda$, $sk$, $hsk$, ct) $\to$ z/⊥: It takes as input $sk$, $hsk$, ct and outputs $z \in Z$ or a special symbol ⊥.

We require that Agg and Dec are deterministic.

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

$$\Pr \left[ \text{Ver}(\lambda, i, pk_i) = 1 \mid \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^L, F); (pk_i, sk_i) \leftarrow \text{Gen}(\lambda, i) \right] = 1.$$  

Correctness. For all $\lambda, L \in \mathbb{N}$, all $F$, all $i^* \in [L]$, all $crs \leftarrow \text{Setup}(1^\lambda, 1^L, F)$, all ($pk_i$, $sk_i$) $\leftarrow$ Gen($\lambda$, i), all ($pk_i$)$_{i \in [L] \{ i^* \}}$ such that Ver($\lambda$, i, $pk_i$) = 1, all $x \in X$ and $f_1, \ldots, f_L \in F$, we have

$$\Pr \left[ \text{Dec}(sk_{i^*}, hsk_{i^*}, ct) = f_{i^*}(x) \mid (mpk, (hsk_k)_{k \in [L]}) \leftarrow \text{Agg}(\lambda, crs, (pk_i, f_i)_{i \in [L]}); ct \leftarrow \text{Enc}(mpk, x) \right] = 1.$$  

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

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

Indistinguishability-based Security (IND-security). For all stateful PPT adversary $\mathcal{A}$, the adaptive (resp., selective) indistinguishability-based security requires the advantage function $\text{Adv}_{\mathcal{A}}^{\text{Ad-sReg-FE}}$ (resp., $\text{Adv}_{\mathcal{A}}^{\text{Sel-sReg-FE}}$) defined as follows is negligible in $\lambda$:

$$\text{Adv}_{\mathcal{A}}^{\text{Ad-sReg-FE}}(\lambda) = \Pr \left[ b = b' \mid \begin{array}{l}
L \leftarrow \mathcal{A}(1^\lambda); \quad \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^L, F) \\
(pk_i, f_i)_{i \in [L]}, x_0^i, x_1^i \leftarrow \mathcal{A}^{\text{OGen}(\cdot), \text{OCor}(\cdot)}(\text{crs}) \\
(mpk, \ldots) \leftarrow \text{Agg}(\text{crs}, (pk_i, f_i), \ldots, (pk_L, f_L)) \\
b \leftarrow \{0, 1\}, \quad ct \leftarrow \text{Enc}(mpk, x_b^i), \quad b' \leftarrow \mathcal{A}(ct')
\end{array} \right] = 1/2,$$

where the oracles work as follows with the initial setting $C = \emptyset$ and $D_i = \emptyset$ for all $i \in [L]$:

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

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

$$D_i[pk_i^*] = \bot \implies \text{Ver}(\lambda, i, pk_i^*) = 1 \quad \text{and} \quad (i, pk_i^*) \in C \lor D_i[pk_i^*] = \bot \implies f_i^*(x_0^i) = f_i^*(x_1^i).$$

The selective IND-security is analogous to above definition of adaptive security, except that $\mathcal{A}$ claim the challenge $x_0^i, x_1^i$ at the begining. Analogous to sReg-ABE [HLWW23], there is no need to give $mpk$ and $hsk_1, \ldots, hsk_L$ to $\mathcal{A}$ explicitly and to consider post-challenge queries.

---

Note that we use two difference indices $i$ and $j$ for $pk_i$ and $hsk_j$, respectively; both of them range from 1 to $L$. 

---
2.4 Quasi-Adaptive Non-Interactive Zero-Knowledge Argument

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:

- \( \text{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [M]_1) \rightarrow (\text{crs}, \text{td}) \): It takes as input the security parameter \( 1^\lambda \), language parameter \( 1^n \), \( 1^m \), \( 1^\ell \), and a matrix \( [M]_1 \in \mathbb{G}_1^{n \times m} \) defining a linear space, outputs common reference string \( \text{crs} \) and trapdoor \( \text{td} \).
- \( \text{LPrv}(\text{crs}, [Y]_1, X) \rightarrow \pi \): It takes as input \( \text{crs} \), a matrix \( [Y]_1 \in \mathbb{G}_1^{n \times \ell} \) along with \( X \in \mathbb{Z}_p^{m \times \ell} \), such that \( Y = MX \), outputs a proof \( \pi \).
- \( \text{LVer}(\text{crs}, [Y]_1, \pi) \rightarrow 0/1 \): It takes as input \( \text{crs} \), \( [Y]_1 \), and \( \pi \), outputs a bit showing the validity of \( \pi \).
- \( \text{LSim}(\text{crs}, \text{td}, [Y]_1) \rightarrow \tilde{\pi} \): It takes as input \( \text{crs}, \text{td}, [Y]_1 \), outputs a simulated proof \( \tilde{\pi} \).

Perfect Completeness. For all \( \lambda, M \), and all \( X, Y \) such that \( Y = MX \):

\[
\Pr \left[ \text{LVer}(\text{crs}, [Y]_1, \pi) = 1 \mid \text{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [M]_1), \pi \leftarrow \text{LPrv}(\text{crs}, [Y]_1, X) \right] = 1.
\]

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

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

Unbounded Simulation Soundness. For all adversary \( \mathcal{A} \), the advantage

\[
\Pr \left[ \begin{array}{c}
(Y^*)_1, \pi \notin Q \\
Y^* \notin \text{span}(M) \\
\text{LVer}(\text{crs}, [Y^*)_1, \pi^*) = 1
\end{array} \right] - \Pr \left[ \begin{array}{c}
\text{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [M]_1), M \leftarrow \mathbb{G}_1^{n \times m} \\
\text{LGen}(1^\lambda, 1^n, 1^m, 1^\ell, [M]_1)
\end{array} \right] \leq \mathbb{Adv}^{\text{VSS}}_{\mathcal{A}}(\lambda)
\]

is negligible in \( \lambda \), where \( Q \) records all queries to \( \text{LSim}(\text{crs}, \text{td}, \cdot) \) along with response. We use \( \mathbb{Adv}^{\text{VSS}}_{\mathcal{A},\lambda}(\lambda) \) to denote the advantage function. Note that our definition is stronger in the sense that the adversary is given \( M \) instead of \( [M]_1 \).

Scheme from Pairings. It is shown in [KW15] that there exists QA-NIZK scheme for \( \ell = 1 \) in the prime-order bilinear group whose enhanced soundness (defined above) relies on MMDH assumption (see Assumption 1). For general \( \ell > 1 \), we simply employ \( \ell \) parallel fresh instances. See [ZZGQ23] for more details.

2.5 Bilateral Public-Key Encryption with Linear Decryption

Algorithms. A bilateral public-key encryption (Bi-PKE) over bilinear group \( \mathbb{G} \) consists of three efficient algorithms:

- \( \text{Gen}(1^\lambda) \rightarrow ([pk]_1, [pk]_2, sk) \): It takes as input the security parameter \( 1^\lambda \), outputs public keys \( [pk]_1 \) (over \( \mathbb{G}_1 \)) and \( [pk]_2 \) (over \( \mathbb{G}_2 \)) and a secret key \( sk \) (over \( \mathbb{Z}_p \)).
- \( \text{Enc}([pk]_1, [pk]_2, m) \rightarrow ([ct]_1, [ct]_2) \): It takes as input \( [pk]_1, [pk]_2 \) and a message \( m \in \mathbb{Z}_p \), outputs ciphertext \( [ct]_1 \) (over \( \mathbb{G}_1 \)) and \( [ct]_2 \) (over \( \mathbb{G}_2 \)).
- \( \text{Dec}_s([ct]_s, sk) \rightarrow m', s \in \{1, 2\} \): It takes as input a (partial) ciphertext \( [ct]_s \) over \( \mathbb{G}_s \) and a secret key \( sk \), outputs \( m' \).
Correctness. For all $\lambda \in \mathbb{N}$, all $m \in \mathbb{Z}_p$ all $s \in \{1, 2\}$, we have:

$$\Pr \left[ \text{Dec}_s([ct]_s, sk) = m \right] = \frac{1}{2}.$$  

Linear Decryption. For all $\lambda \in \mathbb{N}$, all $m \in \mathbb{Z}_p$, we have $[ct]_1$, $[ct]_2$ and $sk$ are vectors with same size (respectively over $G_1, G_2$ and $\mathbb{Z}_p$), and for all $s \in \{1, 2\}$, we have:

$$sk \cdot ct = \text{Dec}_s([ct]_s, sk).$$

Security. For all stateful $\mathcal{A}$, the following advantage function is negligible

$$\text{Adv}_{\mathcal{A}}^{\text{Bi-PKE}} = \Pr \left[ b = b' \right] = \frac{1}{2}.$$  

Group-based Encryption. For all $([pk]_1, [pk]_2) \in \text{Gen}(1^k)$, there exists a group-based algorithm $\text{Enc}'$ such that

$$\text{Enc}'([pk]_1, [pk]_2, [m]_1, [m]_2) \equiv \text{Enc}([pk]_1, [pk]_2, m)$$

A Concrete Bi-PKE. We present a Bi-PKE transformed from ElGamal PKE:

- $\text{Gen}(1^k)$: Run $G := (p, G_1, G_2, G_T, e) \leftarrow \mathcal{G}(1^k)$. Sample

  $$A \leftarrow \mathbb{Z}_p^{k \times (k+1)}, \ w \leftarrow \mathbb{Z}_p^{1 \times (k+1)}$$

  Output:

  $$[pk]_1 = [A, Aw^1], [pk]_2 = [A, Aw^2]$$

- $\text{Enc}([pk]_1, [pk]_2, m)$: Sample $s \leftarrow \mathbb{Z}_p^{1 \times k}$, output

  $$[ct]_1 = [sA, sAw^1 + m]_1, \ [ct]_2 = [sA, sAw^1 + m]_2$$

- $\text{Dec}([ct]_s, sk)$: Compute

  $$[z]_s = [sk \cdot ct]_s$$

  Recover $z$ from $[z]_s$ via brute-force DLOG and output $z$.

3 Slotted Registered Inner-product Functional Encryption

In this section, we present our slotted Reg-IPFE scheme for the inner product functionality which is defined by

$$X = \mathbb{Z}_p^{1 \times n}, \ Z = \mathbb{Z}_p,$$

\[\text{IP}_n = \{ y : x \mapsto xy' \} \]

The scheme achieves the adaptive IND-security defined in Section 2.3 under the $k$-Lin assumption. Applying generic transformation [HLWW23, FTM+23, DP23] gives our Reg-IPFE scheme. Let us define dual basis and show related facts and assumptions.
Dual Basis. Let $\ell_1, \ell_2, \ell_3 \geq 1$ and $\ell := \ell_1 + \ell_2 + \ell_3$. We use basis

$$B_1 \leftarrow \mathbb{Z}_p^{\ell \times \ell_1}, B_2 \leftarrow \mathbb{Z}_p^{\ell \times \ell_2}, B_3 \leftarrow \mathbb{Z}_p^{\ell \times \ell_3},$$

we denote $B_1^\parallel, B_2^\parallel, B_3^\parallel$ as its dual basis, for all $\sigma, \delta \in \{1, 2, 3\}$, it holds that:

$$B_\sigma^\dagger B_\delta = \begin{cases} 1 & \text{when } \sigma = \delta \quad \text{(non-degeneracy)} \\ 0 & \text{when } \sigma \neq \delta \quad \text{(orthogonality)} \end{cases}$$

Facts. With basis $B_1, B_2, B_3$ and its dual basis $B_1^\parallel, B_2^\parallel, B_3^\parallel$, for all $v \in \mathbb{Z}_p^{1 \times n\ell}$, we can uniquely decompose $v$ as

$$v = \sum_{\sigma \in \{1,2,3\}} v^{(\sigma)} \quad \text{where } v^{(\sigma)} \in \text{span}(I_\sigma \otimes (B_\sigma^\dagger))$$

Note that for all $\sigma \in \{1, 2, 3\}$ and $n \in \mathbb{N}$, $v^{(\sigma)}$ can be seen as the projection of $v$ onto $\text{span}(I_\sigma \otimes (B_\sigma^\dagger))$, and for each $S \subseteq \{1, 2, 3\}$, we write $v^S = \sum_{\sigma \in S} s^{(\sigma)}$. Moreover, it holds that:

$$v B_\sigma = v^{(\sigma)} B_\sigma, \quad \text{and } \{v^{(\sigma)}, \{v^{(\delta)}\}_{\delta \neq \sigma}\} \equiv \{v^*, \{v^{(\delta)}\}_{\delta \neq \sigma}\}$$

where $v^* \leftarrow \text{span}(I_\sigma \otimes (B_\sigma^\dagger))$.

Assumption 2 (SD_{B_1^\parallel \rightarrow B_2^\parallel} for $s \in \{1, 2\}$) Let $\ell_1, \ell_2, \ell_3 \geq 1$ and $\ell := \ell_1 + \ell_2 + \ell_3$. We say that the subspace decision assumption $SD_{B_1^\parallel \rightarrow B_2^\parallel}$ holds in $G_s$ if there exist an efficient sampler outputting random $[B_1]_s \in G_s^{\ell \times \ell_1}, [B_2]_s \in G_s^{\ell \times \ell_2}, [B_3]_s \in G_s^{\ell \times \ell_3}$ along with its dual basis: $B_1^\parallel, B_2^\parallel, B_3^\parallel$ such that for all PPT adversaries $A$, the following advantage function is negligible in $\lambda$.

$$\text{Adv}^{SD_{B_1^\parallel \rightarrow B_2^\parallel}}_{A, \lambda, \ell_1, \ell_2, \ell_3} = |\text{Pr}[A(G_s, D, [t_0]_s) = 1] - \text{Pr}[\mathcal{A}(G_s, D, [t_0]_s) = 1]|$$

where $G := (p, G_1, G_2, G_3, e) \leftarrow \mathcal{G}(1^\lambda), D := ([B_1]_s, [B_2]_s, [B_3]_s), \text{basis}(B_1^\parallel, B_2^\parallel), \text{basis}(B_3^\parallel)$ and $t_0 \leftarrow \text{span}(B_1^\parallel), t_1 \leftarrow \text{span}(B_2^\parallel)$.

3.1 Scheme

Assuming QA-NIZK $\Pi_0 = (\text{LGen, LPri, LVer, LSim})$ for linear space over bilinear groups, see Section 2.4, our slotted Reg-IPFE scheme in prime-order bilinear groups works as follows:

- Setup($1^\lambda, 1^n, 1^L$) : Run $G := (p, G_1, G_2, G_3, e) \leftarrow \mathcal{G}(1^\lambda)$. Sample $A \leftarrow \mathbb{Z}_p^{(2k+1) \times (2k+1)}, B_1 \leftarrow \mathbb{Z}_p^{(2k+1) \times (2k+1)}, V \leftarrow \mathbb{Z}_p^{(2k+1) \times (2k+1)}$.

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

$$W_i \leftarrow \mathbb{Z}_p^{(2k+1) \times (2k+1)^n}, R_i \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+1)}, r_i \leftarrow \mathbb{Z}_p^{1 \times k}.$$ For all $i \in [L]$, write $A_i = \left(\begin{array}{c} A_i \\ R_i \end{array}\right) \in \mathbb{Z}_p^{(3k+2) \times (2k+1)}$, run

$$(\text{crs}_i, \text{td}_i) \leftarrow \text{LGen}(1^\lambda, G_1, [A_i]_1).$$

Output

$$\text{crs} = \left[\begin{array}{c} [A, AV], \{[B_1 t_j^*]_2\}_{j \in [L]} \\ \{\text{crs}_i, [R_i, AW_i]_1\}_{i \in [L]} \\ \{[W_i (I_1 \oplus B_1 r_j^*)]_2\}_{j \in [L], i \in [L] \setminus \{i\}} \end{array}\right].$$

Note that we do not use $\text{td}_1, \ldots, \text{td}_L$ in the actual scheme.
- \text{Gen}(\text{crs}, i): \text{Sample } U_i \leftarrow Z_p^{(2k+1)\times(2k+1)}. \text{ Define } F_i = (T_i, Q_i) = (A_i U_i) \in \mathbb{Z}_p^{(3k+2)\times(2k+1)} \text{ and run }

\pi_i \leftarrow \text{LPrv}(\text{crs}_i, [F_i])_1, U_i).

\text{Fetch } \{[B_j, r_j]_2\}_{j \in [L] \setminus \{i\}} \text{ from crs and output }

pk_i = \left(\left[T_i, Q_i, m_i, \left(U_i [B_i r_j]_2\right)_{j \in [L] \setminus \{i\}, \pi_j}\right]\right) \text{ and } sk_i = U_i.

- \text{Ver}(\text{crs}, i, pk_i): \text{Parse } pk_i = \left(\left[T_i, Q_i, \left[h_i[j]_2\right]_{j \in [L] \setminus \{i\}, \pi_i}\right]\right). \text{ Write } F_i = (T_i, Q_i) \text{ and check }

\text{LVer}(\text{crs}_i, [F_i], \pi_i) \equiv 1.

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

e([A]_1, [h_i[j]]_2) \equiv e([T_i]_1, [B_i r_j]_2).

\text{If all these checks pass, output } 1; \text{ otherwise, output } 0.

- \text{Agg}(\text{crs}, (pk_i, y_i))_{i \in [L]}: \text{For all } i \in [L], \text{ parse } pk_i = \left(\left[T_i, Q_i, \left[h_i[j]_2\right]_{j \in [L] \setminus \{i\}, \pi_i}\right]\right). \text{ Output: }

mpk = \left(\left[A, \sum_{i \in [L]} (T_i + AW_i (y_i^j \otimes I_{2k+1})), \sum_{i \in [L]} Aw_i, Av \right]_1\right)

\text{and for all } j \in [L]

hsk_j = \left(\left[B_i r_j, \sum_{i \in [L] \setminus \{j\}} (h_i + W_i (I_{2k+1} \otimes B_i r_j) y_i), \sum_{i \in [L] \setminus \{j\}} W_i (I_{2k+1} \otimes B_i r_j) \right]_2\right),

\text{Enc}(mpk, x): \text{Sample } s \leftarrow Z_p^{1 \times k}. \text{ Output: }

ct = \left(\left[sA, \sum_{i \in [L]} (sT_i + sAw_i (y_i^j \otimes I_{2k+1})), x \otimes sAv + \sum_{i \in [L]} sAw_i, sAv \right]_1\right).

\text{Dec}(sk_i, hsk_j, ct): \text{Parse }

sk_i = U_i, \quad hsk_i = ([K_0, K_1, K_2]_2), \quad ct = ([c_0, c_1, c_2, c_3]_1).

\text{Recover }

[z_1]_T = e([c_0]_1, [I_{2k+1} \otimes K_0]_2), \quad [z_2]_T = e([c_0]_1, [K_2]_2);

[z_3]_T = e([c_1]_1, [K_0]_2), \quad [z_4]_T = e([c_0]_1, [K_1]_2);

[z_5]_T = e([c_0]_1, [I_{2k+1} \otimes U_i]_1, [K_0]_2),

[z_6]_T = e([c_3]_1, [K_0]_2).

\text{Compute }

[z']_T = [(z_1 - z_2) y_i^j, (z_3 - z_4 - z_5)]_T.

\text{Recover } z \text{ from } [z']_T \text{ over } [z_6]_T \text{ via brute-force DLOG and output } z.
Completeness. For all $\lambda, L, n \in \mathbb{N}$, all $i \in [L]$, all $\text{crs} \leftarrow \text{Setup}(1^L, 1^n, 1^L)$ and $(pk_i, sk_i) \leftarrow \text{Gen}(\text{crs}, i)$, we have
\[ pk_i = ([T_i, Q_i], \{ [h_i, j] \}_{j \in [L] \setminus \{ i \}}, \pi_i) = ([AU_i, R_i U_i], \{ [U_i B_i r_j] \}_{j \in [L] \setminus \{ i \}}, \pi_i) \]
for some $U_i \leftarrow \mathcal{Z}_p^{(2k+1) \times (2k+1)}$ and $\pi_i \leftarrow \text{LPrv}(\text{crs}_i, [A_i U_i], U_i)$ where $(\text{crs}_i, \text{td}_i) \leftarrow \text{LGen}(1^L, \mathcal{G}_1, [A_i]_1)$ and $A_i = (\mathcal{A}_i)_{ \mathcal{R}_i}$ with $A \leftarrow \mathcal{Z}_p^{k \times (2k+1)}$, $R_i \leftarrow \mathcal{Z}_p^{(2k+2) \times (2k+1)}$. Then
- Write $F_i = (T_i)_{Q_i} = (\mathcal{A}_i)_{R_i U_i}$, we have $\text{LVer}(\text{crs}_i, [F_i]_1, \pi_i) = 1$ by the perfect completeness of $\Pi_0$ (see Section 2.4) and the fact that $F_i = A_i U_i$;
- For each $j \in [L] \setminus \{ i \}$, we have $e([A_i]_1, [U_i B_i r_j]_2) = e([AU_i]_1, [B_i r_j]_2)$ by the definition of bilinear map $e$ (see Section 2.1) and the fact that $A \cdot U_i B_i r_j = AU_i \cdot B_i r_j$.

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

Correctness. For all $\lambda, L, n \in \mathbb{N}$, all $i^* \in [L]$, all $\text{crs} \leftarrow \text{Setup}(1^L, 1^n, 1^L)$, all $(pk_{i^*}, sk_{i^*}) \leftarrow \text{Gen}(\text{crs}, i^*)$, all $(pk_i)_{i \in [L] \setminus \{ i^* \}}$ such that $\text{Ver}(\text{crs}, i, pk_i) = 1$, for all $y_1, \ldots, y_L \in \mathbb{Z}_p^n$ and $x \in \mathbb{Z}_p^n$, we have:
\[
\begin{align*}
\text{sk}_{i^*} &= U_{i^*}, \\
\text{ct} &= \left( \left( \sum_{i \in [L]} (sT_i + sAW_i(y_i^* \otimes I_{2k+1})), x \otimes sAW_i + \sum_{i \in [L]} sAW_i, sAW_i \right), x \right)
\end{align*}
\]
\[
\begin{align*}
\text{hsk}_{i^*} &= \left( \sum_{i \in [L] \setminus \{ i^* \}} (h_i + W_i(I_n \otimes B_i r_j) y_j^*), \sum_{i \in [L] \setminus \{ i^* \}} W_i(I_n \otimes B_i r_j) y_j^* \right)
\end{align*}
\]
where
\[
A_{h_i} = T_i B_i r_i, \quad \forall i \in [L] \setminus \{ i^* \} \quad \text{and} \quad AU_i = T_i.
\]

Note that here we actually consider $\text{hsk}_j$ for $j = i^*$ and $\text{sk}_i$ for $i = i^*$ and all above equalities are ensured by $\text{Ver}$ and $\text{Gen}$. we have
\[
\begin{align*}
z_1 &= (x \otimes sAW)(I_n \otimes B_i r_i) + \sum_{i \in [L]} sAW_i(I_n \otimes B_i r_i) \\
&= sAW(x \otimes I_{2k+1})(I_n \otimes B_i r_i) + \sum_{i \in [L]} sAW_i(I_n \otimes B_i r_i) \\
&= sAVB_i r_i, x + \sum_{i \in [L]} sAW_i(I_n \otimes B_i r_i) \tag{11}
\end{align*}
\]
\[
\begin{align*}
z_2 &= \sum_{i \in [L] \setminus \{ i^* \}} sAW_i(I_n \otimes B_i r_i) \\
z_3 &= \sum_{i \in [L]} (sT_i B_i r_i + sAW_i(y_i^* \otimes I_{2k+1}) B_i r_i) \\
&= \sum_{i \in [L]} (sT_i B_i r_i + sAW_i(I_n \otimes B_i r_i) y_i^* ) \tag{12}
\end{align*}
\]
\[
\begin{align*}
z_4 &= \sum_{i \in [L] \setminus \{ i^* \}} (sAh_i + sAW_i(I_n \otimes B_i r_i) y_i^* ) \\
z_5 &= sAU_i B_i r_i \\
z_6 &= sAVB_i r_i.
\end{align*}
\]
and then
\[
[z']_T = [(z_1 - z_2)y_{i, r}^* - (z_3 - z_4)]_T
\]
\[
= [(sAVB_1r_{i, r}^* - xy_{i, r}^* + sAW(I_i \otimes B_{i}r_{i, r}^*)) - (sT_{i, r}B_{i}r_{r, i}^* + sAW(I_i \otimes B_{i}r_{r, i}^*))y_{i, r}^* - sAU(I_i \otimes B_{i}r_{r, i}^*)]_T
\]
\[
= [sAVB_1r_{i, r}^* - xy_{i, r}^*]_T
\]
(13)
Here, equality (11) and equality (12) follows from the property of tensor product: \((M \otimes I)(I \otimes a^i) = M \otimes a^i = (I \otimes a^i)M\) for matrices of proper size; equality (13) follows from the fact that \(A_{il} = T_iB_{i}r_{r, i}^*\) for all \(i \in [L] \setminus \{i^*\}\); equality (14) follows from the fact that \(T_{i, r} = AU_{i, r}\). Treat \([z_6]_T = [sAVB_1r_{i, r}^*]_T\) as the basis, and recover \(z\) from \([z']_T = [sAVB_1r_{i, r}^* - xy_{i, r}^*]_T\) via brute-force DLOG, we have \(z = xy_{i, r}^*\).

This proves the correctness.

**Compactness and Efficiency.** Our slotted Reg-IPFE has the following properties:

\[
|\text{crs}| = L^2 \cdot n \cdot \text{poly}(\lambda); \quad |\text{mpk}| = n \cdot \text{poly}(\lambda); \quad |\text{hsk}| = n \cdot \text{poly}(\lambda); \quad |\text{ct}| = n \cdot \text{poly}(\lambda).
\]

Note that the total size of \((\text{crs})_{i \in [L]}\) is \(L \cdot \text{poly}(\lambda)\) 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 \(\text{poly}(\lambda)\).

**Security.** We have the following theorem. Given pairing-based QA-NIZK in [KW15] with unbounded simulation soundness under MDDH assumption and the fact that MDDH assumption implies subspace decision assumption [CGKW18], our slotted Reg-IPFE scheme achieves adaptive IND-security from MDDH assumption.

**Theorem 1.** Assume \(\Pi_0 = (\text{LGen}, \text{LPrv}, \text{LVer}, \text{LSim})\) is a QA-NIZK with perfect completeness, perfect zero-knowledge and unbounded simulation soundness for linear space defined in Section 2.4, our slotted Reg-IPFE scheme achieves the adaptive IND-security defined in Section 2.3 under MDDH assumption and subspace decision 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 \(\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3\) and \(\mathcal{B}_4\) such that:

\[
\text{IndAdv}_\mathcal{A}^{\text{Ad-sReg-IPFE}}(\lambda) \leq L \cdot \text{Adv}^{\text{US}}_{\Pi_0}(\lambda) + \text{Adv}^{\text{MDDH}}_{\Pi_2}(\lambda) + L \cdot \text{Adv}^{\text{SD}_{\Pi_0}}_{\Pi_3}(\lambda) + L \cdot \text{Adv}^{\text{SD}_{\Pi_0}}_{\Pi_4}(\lambda) + \text{negl}(\lambda)
\]

where \(L\) is the number of slots and \(\text{Time}(\mathcal{B}_1), \text{Time}(\mathcal{B}_2), \text{Time}(\mathcal{B}_3), \text{Time}(\mathcal{B}_4) \approx \text{Time}(\mathcal{A})\).

**Game Sequence.** Suppose that \(\text{crs}\) is the common reference string, \((x_0^*, x_1^*)\) is the challenge pair, \(\{pk_i^*, y_i^*\}_{i \in [L]}\) are challenge public keys along with challenge functions to be registered. For all \(i \in [L]\), define \(D_i = \{pk_i : \mathcal{D}_i[pk_i] = s_{ki} \neq \perp\}\) be responses to OGen(i) and \(C_i = \{pk_i : (i, pk_i) \in C\}\) records public keys in \(D_i\) that have been sent to OCor(i,\cdot). Recall that, for each \(i \in [L]\), we require that \(pk_i^* \not\in D_i \implies \text{Ver}(\text{crs}, i, pk_i^*) = 1; pk_i^* \in C_i \lor pk_i^* \not\in D_i \implies x_0^*(y_i^*)^\top = x_1^*(y_i^*)^\top\).

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 assignments for \(pk_i\) since the adversary can invoke OGen(i) for many times. We prove the Lemma 1 via nested dual-system method using the following game sequence.
- G₀: This is the real game, recall that we have
  - crs is in the form:
    \[
    \text{crs} = \begin{pmatrix}
    \{A, AV\}_1, \{B_1^r, r^*_j\}_j \in [L] \\
    \{crs_i, [R_i, AW_i]_1\}_i \in [L] \\
    \{[W]_i(I_0 \otimes B_1^r)\}_j \in [L], i \in [L]\} \\
    \end{pmatrix}
    \]
  where \( crs_i \in \text{LGen}(1^λ, G_1, [A_i]_1) \), \( A_i = (A_i^r) \).
  - For each \( i \in [L] \), each \( pk_i \in D_i \) is in the form:
    \[
    pk_i = \begin{pmatrix}
    \{A_i, AV_i\}_1, [R_i, U_i]_1, \{[U_iB_1^r, r^*_j]\}_j \in [L]\} \\
    \{\pi_i\} \\
    \end{pmatrix}
    \]
  where \( \pi_i \leftarrow \text{LPrv}(crs_i, [F_i]_1, U_i) \), \( F_i = (A_i^u, R_i, U_i) \), and \( U_i \) is the corresponding \( sk_i \).
  - For all \( i \in [L] \), \( pk'_i \) is in the form:
    \[
    pk'_i = \begin{pmatrix}
    \{T_i^*, Q_i^*\}_1, \{r_{ij}, \pi_i^*\}_j \in [L]\} \\
    \end{pmatrix}
    \]
  such that \( \text{Ver}(crs, i, pk'_i) = 1 \) which means \( \text{LVer}\left(\text{crs}_i, \begin{pmatrix} T_i^* \\ Q_i^* \end{pmatrix}, \pi_i^*\right) = 1 \) and \( A_ih_i = T_i^*B_1^r \) for each \( j \in [L] \setminus \{i\} \).
  - \( ct' \) for \( (x'_0, x'_1) \) is in the form:
    \[
    ct' = \begin{pmatrix}
    \sum_{i \in [L]} (sT_i + sAW_i((y'_i)^{\top} \otimes I_{2k+1})), x'_0 \otimes sAV + \sum_{i \in [L]} sAW_i, sAV \\
    \end{pmatrix}_1
    \]
    where \( b \leftarrow \{0, 1\} \) is the secret bit.

- G₁: Identical to G₀ except that, for all \( i \in [L] \) and all \( pk_i \in D_i \), we replace \( \pi_i \) with
  \[
  \pi_i \leftarrow \text{LSim}(crs_i, td_i, [F_i]_1) \text{ where } F_i = \begin{pmatrix} A_i^u \\ R_i, U_i \end{pmatrix}.
  \]
  We have \( G_1 \equiv G_0 \). This follows from the perfect zero-knowledge of \( \Pi_0 \).

- G₂: Identical to G₁ except that we sample \( s \leftarrow \mathbb{Z}_{p}^{1 \times k} \) along with \( A \) and replace all \( R_i \) in crs with
  \[
  \tilde{R}_i = \tilde{R}_i \begin{pmatrix} sA \\ I_{2k+1} \end{pmatrix}, \quad \tilde{R}_i \leftarrow \mathbb{Z}_{p}^{(2k+2) \times (2k+2)}.
  \]
  We have \( G_2 \equiv G_1 \). This follows from the fact that both \( R_i \) (in G₁) and \( \tilde{R}_i \) (in G₂) are truly random since matrix \( (sA)_{2k+1} \) is full-rank.

- G₃: Identical to G₂ except that we generate the \( c_i' \) as follows:
  \[
  c_i' = \sum_{i \in [L]} (e_i\tilde{R}_i^{-1}Q_i + sAW_i(y'_i \otimes I_{2k+1})).
  \]
  We have \( G_3 \approx G_2 \). This follows from stronger unbounded simulation soundness of \( \Pi_0 \) along with the fact that \( \text{LVer}(crs, [F_i^*], \pi_i^*) = 1 \) for all \( i \in [L] \) where \( F_i^* = \begin{pmatrix} T_i^* \\ Q_i^* \end{pmatrix} \). Assume \( pk'_i \notin D_i \), i.e., \( pk'_i \) is malicious. In the reduction, we guess \( i^* \leftarrow [L] \) and obtain \( A, \tilde{R}_{i^*}, crs_{i^*} \) as input; we simulate honestly as in G₃ except that for all \( pk_i \in D_i \), we make an oracle query \( [F_i^*]_1 \) and get \( \pi_i^* \) in it; we finally output \( ([F_i^*]_1, \pi_i^*) \) in \( pk'_i \notin D_i \). Observe that once it happens that \( e_i\tilde{R}_i^{-1}Q_i \neq sT_i^* \), we must have \( F_i^* \notin \text{span}(A_{i^*}) \). When \( pk'_i \in D_i \), we always have \( G_3 \equiv G_2 \).
\( - \) $G_4$: Identical to $G_3$ except that we replace all $sA$ with $c \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}$; in particular, we generate $\tilde{R}_t$ as follows:
\[
\tilde{R}_t = \tilde{R}_t \left( \begin{bmatrix} \mathbb{C} \\ \mathbb{I}_{2k+1} \end{bmatrix} \right), \quad \tilde{R}_t \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+2)}
\]
and generate the challenge ciphertext as follows:
\[
ct' = \left( \begin{bmatrix} \mathbb{C} \\ \sum_{i \in [L]} \left( e_i \tilde{R}_t^{-1} Q_i^* + \mathbb{C} \mathbb{W}_i \left( (y_i^*)^\top \otimes \mathbb{I}_{2k+1} \right) \right), \mathbb{X}_b^\top \mathbb{V} + \sum_{i \in [L]} \mathbb{C} \mathbb{W}_i, \mathbb{C} \mathbb{V} \end{bmatrix} \right)_{1,1}.
\]

We have $G_4 \approx G_3$. This follows from MDDH assumption which ensures that $([A]_1, [sA]_1) \approx_c ([A]_1, [c]_1)$ when $A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}$, $s \leftarrow \mathbb{Z}_p^{1 \times k}$, $c \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}$.

\( - \) $G_5$: Identical to $G_4$ except that for all $i \in [L]$, we replace $AV$ in $crs$ with
\[
\mathbb{V} \leftarrow \mathbb{Z}_p^{k \times (2k+1)}
\]
we replace $cV$ in challenge ciphertext with
\[
v \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}
\]
In particular, we generate $crs$ as below:
\[
\text{crs} = \left( [A, \mathbb{V}]_1, \{ [B_i \mathbb{r}_j]_2 \}_{j \in [L]} \right),
\]
and generate the challenge ciphertext as
\[
ct' = \left( \begin{bmatrix} \mathbb{C} \\ \sum_{i \in [L]} \left( e_i \tilde{R}_t^{-1} Q_i^* + \mathbb{C} \mathbb{W}_i \left( (y_i^*)^\top \otimes \mathbb{I}_{2k+1} \right) \right), \mathbb{X}_b^\top \mathbb{V} + \sum_{i \in [L]} \mathbb{C} \mathbb{W}_i, \mathbb{C} \mathbb{V} \end{bmatrix} \right)_{1,1}.
\]

We have $G_5 \equiv G_4$. This follows from the fact that when $V$ is uniformly sampled from $\mathbb{Z}_p^{(2k+1) \times (2k+1)}$ and not published elsewhere, $(AV, cV)$ (in $G_4$) is stastically equivalent with the uniformly sampled $(\mathbb{V}, v)$ where $\mathbb{V} \leftarrow \mathbb{Z}_p^{k \times (2k+1)}$, $v \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}$ (in $G_5$), since both $A$ and $c$ are full row rank (with overwhelming probability).

\( - \) $G_6$: Identical to $G_5$ except that we randomly sample $B_2 \leftarrow \mathbb{Z}_p^{2k+1}$, $B_3 \leftarrow \mathbb{Z}_p^{(2k+1) \times k}$, and compute the dual basis $B_1^\top, B_2^\top, B_3^\top$. And we change $c_2^\top$ as follows:
\[
c_2^\top = x_b^\top \otimes v^{(1,3)} + x_0^\top \otimes v^{(2)} + \sum_{i \in [L]} cW_i
\]
We have $G_6 \equiv G_5$. This follows from the following argument for $b' = b$ (in $G_6$) or $b' = 0$ (in $G_6$):
\[
x_{b'}^\top \otimes v^{(2)} + \sum_{i \in [L]} (cW_i)^{(2)} \equiv \sum_{i \in [L]} (cW_i)^{(2)}
\]
This argument follows from the fact that the basis $B_2$ and dual basis $B_2^\top$ are not revealed, so we have $(cW_i)^{(2)}$ is hidden, this can imply that $\sum_{i \in [L]} (cW_i)^{(2)}$ hides $x_{b'}^\top \otimes v^{(2)}$.

\( - \) $G_{7,\ell}$, ($\ell \in [0, L]$): Identical to $G_6$ except that for all $j \in [\ell]$ we replace all $B_i \mathbb{r}_j$ in $crs$ with
\[
d_j \quad \text{where} \quad d_j \leftarrow \text{span}(B_j^\top)
\]
We have that
• $G_{7,0} = G_6$; the two games are actually identical, since $[0] = 0$;
• $G_{7,\ell-1} \approx_c G_{7,\ell}$ for all $\ell \in [L]$, we will employ a sub-sequence of games for the proof described later.

- $G_8$: Identical to $G_{7,L}$ except that we generate the $c^*_2$ as follows:

$$c^*_2 = \sum_{i \in [L]} x_i^2 \otimes v^{(1,3)} + x_i^0 \otimes v^{(2)} + \sum_{i \in [L]} cW_i$$

We have $G_8 \equiv G_{7,L}$. The proof is analogous to that of $G_6 \equiv G_5$, with the fact that basis $B_1, B_3$ and dual basis $B_1^\parallel, B_3^\parallel$ are not revealed in $G_{7,L}$, we have the following argument for $b' = b$ (in $G_{7,L}$) or $b' = 0$ (in $G_8$):

$$x^*_0 \otimes v^{(1,3)} + \sum_{i \in [L]} (cW_i)^{(1,3)} \equiv \sum_{i \in [L]} (cW_i)^{(1,3)}$$

Observe that, in the final game $G_8$ the challenge ciphertext $ct$ is independent of the random bit $b$ and the adversary’s advantage is exactly 0.

**From $G_{7,\ell-1}$ to $G_{7,\ell}$**. We are ready to prove $G_{7,\ell-1} \approx_c G_{7,\ell}$ and this will complete the proof of Lemma 1. For this, we need the following sub-sequence of games for each $\ell \in [L]$: 

- $G_{7,\ell-1,0}$: Identical to $G_{7,\ell-1}$ where we recall $crs, pk_\ell, \in D_1$ and $c^*_2$, with highlighting relevant terms in the following sub-sequence with dashed boxes as follows:

$$crs = \left( \begin{array}{c} [A, V]_1, \{[d^j]_2\}_{j \in \{\ell-1\}}, \{[B_1 r^j]_2\}_{j \in \{\ell\} \setminus \{\ell\}}, \{[B_1 r^j]_2\}_{j \in \{\ell\} \setminus \{\ell\}} \\
\{crs, [\tilde{R}, AW]_1\}_{\ell \in [L]} \\
\{[W_i (I_n \otimes d^j)]_2\}_{j \in \{\ell-1\}, i \in [L] \setminus \{\ell\}} \\
\{[W_i (I_n \otimes B_1 r^j)]_2\}_{j \in \{\ell\}, i \in [L] \setminus \{\ell\}} \\
\{[W_i (I_n \otimes B_2 r^j)]_2\}_{j \in \{\ell\}, i \in [L] \setminus \{\ell\}} \end{array} \right)$$

$$pk_\ell = \left( \begin{array}{c} \{[AU, R U]_1, \{[U d^j]_2\}_{j \in \{\ell-1\}}, \{[U B_1 r^j]_2\}_{j \in \{\ell\}}\} \\
\{[AU, R U]_1, \{[U d^j]_2\}_{j \in \{\ell-1\}}, \{[U B_2 r^j]_2\}_{j \in \{\ell\}}\} \\
\{[AU, R U]_1, \{[U d^j]_2\}_{j \in \{\ell-1\}}, \{[U B_2 r^j]_2\}_{j \in \{\ell\}}\} \\
\{[AU, R U]_1, \{[U d^j]_2\}_{j \in \{\ell-1\}}, \{[U B_2 r^j]_2\}_{j \in \{\ell\}}\} \\
\{[AU, R U]_1, \{[U d^j]_2\}_{j \in \{\ell-1\}}, \{[U B_2 r^j]_2\}_{j \in \{\ell\}}\} \end{array} \right)$$

$$c^*_2 = x^*_0 \otimes v^{(1)} + x^*_0 \otimes v^{(2)} + \sum_{i \in [L] \setminus \{\ell\}} cW_i + \sum_{i \in [L] \setminus \{\ell\}} cW_i$$

Where $d^j \leftarrow \text{span}(B_2)$ for all $j \in \{\ell-1\}$. We have $G_{7,\ell-1,0} = G_{7,\ell-1}$; all changes are conceptual.

- $G_{7,\ell-1,1}$: Identical to $G_{7,\ell-1,0}$ except that we replace all $B_1 r^j$ in $crs$ with $d^j$ where $d^j \leftarrow \text{span}(B_2)$.

In particular, we change the dashed boxed term in $crs$ and $pk_\ell$ as follows:

$$\left( \begin{array}{c} \{[d^j]_2\}_{j \in \{\ell\}} \\
\{[W_i (I_n \otimes d^j)]_2\}_{j \in \{\ell\}} \\
\{[W_i (I_n \otimes B_1 r^j)]_2\}_{j \in \{\ell\}} \\
\{[W_i (I_n \otimes B_2 r^j)]_2\}_{j \in \{\ell\}} \end{array} \right)$$

We have $G_{7,\ell-1,1} \approx_c G_{7,\ell-1,0}$. This follow from the SD$^{G_2}_{B_1 \leftarrow B_3}$ assumption which ensure that

$$[t_0]_2 \approx_c [t_1]_2 \quad \text{given} \quad [B_1]_2, [B_2]_2, [B_3]_2, \text{basis}(B_1^\parallel, B_3^\parallel), \text{basis}(B_2^\parallel)$$

Where $t_0 \leftarrow \text{span}(B_2)$ corresponding to $G_{7,\ell-1,0}$, and $d^j \leftarrow \text{span}(B_2)$ corresponding to $G_{7,\ell-1,1}$. 

25
- $G_{7,\ell-1,2}$: Identical to $G_{7,\ell-1,1}$ except that we generate the $c_2^+$ as follows:

$$c_2^+ = x_\delta^+ \otimes v^{(1)} + x_0^+ \otimes v^{(2)} + x_{\delta1} \otimes v^{(3)} + cW_\ell + \sum_{i \in [L] \setminus \{\ell\}} cW_i$$

We have $G_{7,\ell-1,2} \approx_c G_{7,\ell-1,1}$. We provide an overview of the proof in Section 3.3.

- $G_{7,\ell-1,3}$: Identical to $G_{7,\ell-1,2}$ except that we replace all $d_i^\tau$ in $\text{crs}$ with $d_\ell$ where $d_\ell \leftarrow \text{span}(B_2^\tau)$

In particular, we change the dashed boxed term in $\text{crs}$ and $pk_t$ as follows:

$$[[[d_\ell]]_2, \{[W_i (I_0 \otimes [d_\ell^\tau]])_2, [U_i [d_\ell^\tau]]_2\}_{i \in [L] \setminus \{\ell\}}]$$

We have $G_{7,\ell-1,3} \approx_c G_{7,\ell-1,2}$. This follow from the $SD_{B_1 \rightarrow B_2}$ assumption which ensure that $[t_0]_2 \approx_c [t_1]_2$ given $[B_1]_2, [B_2]_2, [B_3]_2, \text{basis}(B_2^\parallel), \text{basis}(B_1^\parallel)$

Where $t_0 \leftarrow \text{span}(B_3^\tau)$ corresponding to $G_{7,\ell-2,2},$ and $d_\ell \leftarrow \text{span}(B_2^\tau)$ corresponding to $G_{7,\ell-1,3}$.

- $G_{7,\ell-1,4}$: Identical to $G_{7,\ell-1,3}$ except that we generate the $c_2^+$ as follows:

$$c_2^+ = x_\delta^+ \otimes v^{(1)} + x_0^+ \otimes v^{(2)} + x_{\delta1} \otimes v^{(3)} + cW_\ell + \sum_{i \in [L] \setminus \{\ell\}} cW_i$$

We have $G_{7,\ell-1,4} \approx_c G_{7,\ell-1,3}$. The proof is identical to that for $G_{7,\ell-1,2} \approx_c G_{7,\ell-1,1}$.

Observe that $G_{7,\ell-1,4} = G_{7,\ell}$ and this prove $G_{7,\ell-1} \approx_c G_{7,\ell}$.

### 3.3 From $G_{7,\ell-1,1}$ to $G_{7,\ell-1,2}$

We review $G_{7,\ell-1,1}$ and $G_{7,\ell-1,2}$ in the following form. Here we use solid box to indicate the difference between two games and use dashed boxes to highlight those terms that are relevant to our proof. For all $j \in \{\ell - 1\}$, we rewrite $d_j \leftarrow \text{span}(B_2^\tau)$ with $B_2^\tau r_j$, for some $r_j \sim \mathbb{Z}_p$.

$$c_2^+ = x_\delta^+ \otimes v^{(1)} + x_0^+ \otimes v^{(2)} + x_{\delta1} \otimes v^{(3)} + cW_\ell + \sum_{i \in [L] \setminus \{\ell\}} cW_i$$
where \( d_\ell \leftarrow \text{span}(B_\ell) \). We define \( c^+ \in \mathbb{Z}_{p}^{2k+1} \) such that \( Ac^+ = 0 \) and \( cc^+ = 1 \). With the orthogonality of dual basis, for all \( v^{(3)} \in \text{span}(B_3^+) \), we have:

\[
v^{(3)}B_1 = 0, \quad v^{(3)}B_2 = 0.
\]

We will proof \( G_{7,\ell-1,2} \approx_c G_{7,\ell-1,1} \) by considering two cases: (1) \( pk_\ell^* \) is honest; (2) \( pk_\ell^* \) is corrupted or maliciously generated by the adversary.

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

**Lemma 2.** For all basis \( B_1 \sim \mathbb{Z}_p^{(2k+1)\times k} \), \( B_2 \sim \mathbb{Z}_p^{(2k+1)\times k} \), and its dual basis \( B_1^\|, B_2^\| \). For all \( d^+ \in \text{span}(B_1^+) \) such that \( d^+B_1 = 0 \) and \( d^+B_2 = 0 \). For any adversary \( A \), there exist an adversary \( B_2 \) such that

\[
\Pr[A(A, c, [R], B_1, B_2, d^+, AU, cU_1, [RU]_1, UB_1, UB_2) = 1] - \Pr[A(A, c, [R], B_1, B_2, d^+, AU, cU_1, [RU + \overrightarrow{u}d^+], UB_1, UB_2) = 1] \leq 2 \cdot \text{Adv}_{\mathcal{DH}_2} + \text{negl}(\lambda)
\]

where \( A \sim \mathbb{Z}_p^{k \times (2k+1)} \), \( c \sim \mathbb{Z}_p^{(2k+1)\times 1} \), \( R \sim \mathbb{Z}_p^{(2k+2)\times (2k+1)} \), \( U \sim \mathbb{Z}_p^{(2k+1)\times (2k+1)} \) and \( \overrightarrow{u} \sim \mathbb{Z}_p^{1\times (2k+2)} \).

**Honest Case.** In this case, we have \( pk_\ell^* = ([T_\ell^*, Q_\ell^*], \{[B_\ell^*]_{j \in [L] \setminus \{\ell\}, \pi_\ell^*}\} \in D_\ell \setminus C_\ell \). Namely, we know \( U_\ell^* \) (such that \( T_\ell^* = AU_\ell^* \) and \( Q_\ell^* = R_\ell U_\ell^* \)) and \( U_\ell^* \) is hidden from the adversary. We can write the dashboxed terms in \( c_1^+ \) as follows:

\[
\text{cU}_\ell^* + \text{cW}_\ell((y_\ell^*)^\top \otimes I_{2k+1})
\]

and replace \( \overrightarrow{R} \) in \( \text{crs} \) with a random \( \overrightarrow{R} \) as in \( G_1 \).

Let's use \( x_\ell^* \) to denote the challenge message, which is \( x_\ell^* \) in \( G_{7,\ell-1,1} \) and \( x_\ell^* \) in \( G_{7,\ell-1,2} \) respectively. We have the following argument holds for both \( b' = b \) (in \( G_{7,\ell-1,1} \)) and \( b' = 0 \) (in \( G_{7,\ell-1,2} \)), which proves that \( G_{7,\ell-1,1} \approx_c G_{7,\ell-1,2} \) in the honest case:

\[
\begin{array}{l}
A, c^+, [R], B_1, B_2, d_\ell, \text{AW}_\ell, W_\ell(I_n \otimes B_1), W_\ell(I_n \otimes B_2) \quad \text{//crs, pk}_\ell \\
c, cU_\ell^* + cW_\ell((y_\ell^*)^\top \otimes I_{2k+1}), x_\ell^* \otimes \text{v}^{(3)} + cW_\ell \\
AU_\ell^*, [R, U]^*], U_\ell^*B_1, U_\ell^*B_2 \quad \text{//pk}_\ell^*
\end{array}
\]

\[
\begin{array}{l}
\approx_c A, c^+, [R], B_1, B_2, d_\ell, \text{AW}_\ell, W_\ell(I_n \otimes B_1), W_\ell(I_n \otimes B_2) \\
c, cU_\ell^* + cW_\ell((y_\ell^*)^\top \otimes I_{2k+1}), x_\ell^* \otimes \text{v}^{(3)} + cW_\ell \\
AU_\ell^*, [R, U]^*], U_\ell^*B_1, U_\ell^*B_2 \\
\approx_s A, c^+, [R], B_1, B_2, d_\ell, \text{AW}_\ell, W_\ell(I_n \otimes B_1), W_\ell(I_n \otimes B_2) \\
c, cU_\ell^* + cW_\ell((y_\ell^*)^\top \otimes I_{2k+1}) + [\overrightarrow{u}v^{(3)} + \overrightarrow{w_\ell(y_\ell^*)v^{(3)}}], x_\ell^* \otimes \text{v}^{(3)} + cW_\ell + \text{w}_\ell \otimes \text{v}^{(3)} \\
AU_\ell^*, [R, U]^*], U_\ell^*B_1, U_\ell^*B_2 \\
\approx_s A, c^+, [R], B_1, B_2, d_\ell, \text{AW}_\ell, W_\ell(I_n \otimes B_1), W_\ell(I_n \otimes B_2) \\
c, cU_\ell^* + cW_\ell((y_\ell^*)^\top \otimes I_{2k+1}) + \overrightarrow{u}v^{(3)} + \overrightarrow{w_\ell(y_\ell^*)v^{(3)}}, x_\ell^* \otimes \text{v}^{(3)} + cW_\ell + \text{w}_\ell \otimes \text{v}^{(3)} \\
AU_\ell^*, [R, U]^*], U_\ell^*B_1, U_\ell^*B_2
\end{array}
\]
where \( \tilde{u} \leftarrow \mathbb{Z}_p^{1 \times (2k+2)} \) and \( u_\ell \leftarrow \mathbb{Z}_p^{1 \times n} \). We justify each step as below: The first \( \approx_c \) uses Lemma 2 with \( R = R_\ell, U = U_\ell, u = \tilde{u} \) and \( d = v^{(3)} \). The second \( \approx_s \) uses change of variables

\[
W_\ell \mapsto W_\ell + c^+(w_\ell \otimes v^{(3)}) \quad \text{and} \quad U_\ell \mapsto U_\ell + c^+ u_\ell v^{(3)}
\]

The last \( \approx_s \) follows from the fact that \( \tilde{u} \) hides \( R \) \( u_\ell \), this implies that \( u_\ell \) can hide \( w_\ell (y_\ell^*)^T \) in \( c_1^* \), and \( w_\ell \) hides \( x_0^* \) in \( c_2^* \).

**Corrupted & Malicious Case.** In this case, we have \( \text{pk}_e^* = ([T^*_e, Q^*_e], \{[h^*_e]^{-1}_j \}_{j \in [L]}(\ell), \pi_\ell^*) \in C_\ell \cup D_\ell \). It is required that \( x_0^*(y_\ell^*)^T = x_1^*(y_\ell^*)^T \). We prove \( G_{7, \ell-1.2} \approx_c G_{7, \ell-1.1} \) in this case using the following argument for all \( b \in \{0, 1\}:

\[
A, c^+, B_1, B_2, d, AW_\ell, W_\ell(I_i \otimes B_1), W_\ell(I_i \otimes B_2) \quad \text{//crs}
\]

\[
c, e, R^*_e + cW_\ell((y_\ell^*)^T \otimes I_{2k+1}), x_0^* \otimes v^{(3)} + cW_\ell \quad \text{//ct}^* \text{ in } G_{7, \ell-1.1}
\]

\[
\approx_s A, c^+, B_1, B_2, d_\ell, AW_\ell, W_\ell(I_i \otimes B_1), W_\ell(I_i \otimes B_2)
\]

\[
c, e, R^*_e + cW_\ell((y_\ell^*)^T \otimes I_{2k+1}) - x_0^*(y_\ell^*)^T v^{(3)} x_0^* \otimes v^{(3)} + cW_\ell
\]

\[
= A, c^+, B_1, B_2, d_\ell, AW_\ell, W_\ell(I_i \otimes B_1), W_\ell(I_i \otimes B_2)
\]

\[
c, e, R^*_e + cW_\ell((y_\ell^*)^T \otimes I_{2k+1}) - x_0^*(y_\ell^*)^T v^{(3)} x_0^* \otimes v^{(3)} + cW_\ell
\]

\[
\approx_s A, c^+, B_1, B_2, d_\ell, AW_\ell, W_\ell(I_i \otimes B_1), W_\ell(I_i \otimes B_2) \quad \text{//crs}
\]

\[
c, e, R^*_e + cW_\ell((y_\ell^*)^T \otimes I_{2k+1}) + x_0^*(y_\ell^*)^T v^{(3)} x_0^* \otimes v^{(3)} + cW_\ell \quad \text{//ct}^* \text{ in } G_{7, \ell-1.2}
\]

We justify each step as follows: the first \( \approx_s \) uses the change of variables

\[
W_\ell \mapsto W_\ell - c^+(x_0^* \otimes v^{(3)})
\]

The second = uses the fact that \( x_0^*(y_\ell^*)^T = x_1^*(y_\ell^*)^T \) in this case. The last \( \approx_s \) uses the change of variables

\[
W_\ell \mapsto W_\ell + c^+(x_0^* \otimes v^{(3)})
\]

4 Simulation-based Security for Reg-FE

In this section, we define the notion of simulation-based security in the context of Reg-FE. We give both the adaptive variant and the very selective variant followed by several remarks.

4.1 Adaptive SIM-security for Reg-FE

**Definition.** For all stateful PPT adversary \( \mathcal{A} \), there exists simulator \( \widetilde{\text{Setup}}, \widetilde{\text{Gen}}, \widetilde{\text{Enc}} \) such that:

\[
\begin{align*}
\text{crs} & \leftarrow \text{Setup}(1^3, 1^L, F); \\
x^* & \leftarrow \mathcal{A}^{\widetilde{\text{RegCK}}(-), \widetilde{\text{RegHK}}(-), \widetilde{\text{OCorHK}}(-)}(\text{crs}); \\
c^* & \leftarrow \text{Enc}(\text{mpk}, x^*); \\
[\mathcal{A}^{\widetilde{\text{OCorHK}}(-)}(c^*), \alpha] & \leftarrow \mathcal{A}(c^*) \approx_c \left[ (\text{crs}, \text{td}) \leftarrow \text{Setup}(1^3, 1^L, F); \\
x^* & \leftarrow \mathcal{A}^{\widetilde{\text{RegCK}}(-), \widetilde{\text{RegHK}}(-), \widetilde{\text{OCorHK}}(-)}(\text{crs}); \\
c^* & \leftarrow \widetilde{\text{Enc}}((\text{pk}_e^1, \ldots, \text{pk}_e^L); \text{td}) \\
[\mathcal{A}^{\widetilde{\text{OCorHK}}(-)}(c^*), \alpha] & \leftarrow \mathcal{A}(c^*) \right]
\end{align*}
\]

Here, in the real world (on the left-hand side), the oracles work as follows with initial setting \( \text{aux} = \perp, \text{mpk} = \perp, \mathcal{H} = 0, C = 0 \) and \( \mathcal{D} \) being a dictionary with \( \mathcal{D}[^{\perp}] = 0 \) for all possible \( \text{pk} \):
- \( \text{ORegCK}(\text{pk}, f) \): run \((\text{mpk}', \text{aux}') \leftarrow \text{Reg}(\text{crs}, \text{aux}, \text{pk}, f)\), update \(\text{mpk} = \text{mpk}'\), \(\text{aux} = \text{aux}'\), \(\mathcal{D}[\text{pk}] = \mathcal{D}[\text{pk}] \cup \{f\}\), append \(\text{pk}\) to \(C\) and return \((\text{mpk}, \text{aux})\);
- \( \text{ORegHK}(f) \): run \((\text{pk}, \text{sk}) \leftarrow \text{Gen}(\text{crs}, \text{aux})\) and \((\text{mpk}', \text{aux}') \leftarrow \text{Reg}(\text{crs}, \text{aux}, \text{pk}, f)\), update \(\text{mpk} = \text{mpk}'\), \(\text{aux} = \text{aux}'\), \(\mathcal{D}[\text{pk}] = \mathcal{D}[\text{pk}] \cup \{f\}\), append \((\text{pk}, \text{sk})\) to \(\mathcal{H}\) and return \((|\mathcal{H}|, \text{mpk}, \text{aux}, \text{pk})\);
- \( \text{OCorHK}(i) \): let \(\mathcal{H}[i] = (\text{pk}, \text{sk})\), append \(\text{pk}\) to \(C\) and return \(\text{sk}\);

with the following restrictions:

- for query \(i\) to \(\text{OCorHK}\), it holds that \(\mathcal{H}[i] \neq \perp\).

In the ideal world (on the right-hand side), the oracles are analogous to that in the real world; except that they use \(\text{crs}\) simulated by \(\text{Setup}\) instead of \(\text{crs}\), and \(\text{ORegHK}\) invokes \(\text{Gen}\) instead of \(\text{Gen}\).

### 4.2 Very Selective SIM-security for Reg-FE

In the very selective setting, the adversary claims the challenge, challenge functions, and the types of challenge public keys at the beginning. The specific definition is as follows:

**Definition.** For all stateful PPT adversary \(\mathcal{A}\), there exists simulator \((\widetilde{\text{Setup}}, \widetilde{\text{Gen}}, \widetilde{\text{Enc}})\) such that:

\[
\begin{align*}
    \begin{bmatrix}
    L, L', x^*, \{f_i^x\}_{i \in [L']} \mapsto \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^\ell, F); \\
    \mathcal{A}^{O_{\text{Reg}(\text{crs}, f_i^x_{i \in [L']}, C, K, H, \ldots)}(\text{crs})} \\
    c_t^* \leftarrow \text{Enc}(\text{mpk}, x^*) \mapsto \mathcal{A}(c_t^*)
    \end{bmatrix}
    \overset{\approx}{\Rightarrow}
    \begin{bmatrix}
    \begin{bmatrix}
    L, L', x^*, \{f_i^x\}_{i \in [L']} \mapsto C, K, H, CH \leftarrow \mathcal{A}(1^\lambda); \\
    (\text{crs}, \text{td}) \leftarrow \text{Setup}(1^\lambda, 1^\ell, F); \{f_i^x\}_{i \in C, K, H, CH} \mapsto \mathcal{A}(\text{crs}, \text{td}) \leftarrow \text{Enc}(\text{mpk}, x^*) \mapsto \mathcal{A}(c_t^*)
    \end{bmatrix}
    \end{bmatrix}
    \end{align*}
\]

where \(C, K, H, \overline{C} \subseteq [L']\), \(C \cup H = [L']\) for some \(L' \leq L\), \(C \subseteq H\) and \(C \cap H = \emptyset\), and \(O\) works as follows with a counter \(\ell = 1 + 1\) and the same set of auxiliary data structure as in the definition of IND-security: on input \((i, \text{pk}_i^*\), return \(\perp\) when \(i \neq \ell\), otherwise set \(\ell = \ell + 1\) and do

- when \(i \in C\), return \(\text{ORegCK}(\text{pk}_i^*, f_i^x)\);
- when \(i \in H\), return \(\text{ORegHK}(f_i^x)\); furthermore, if \(i \in CH\), return \(\text{OCorHK}(|H \cap [i]|)\).

Here \(\text{ORegCK}\) and \(\text{ORegHK}\) invoke Reg in both cases: in the real world (on the left-hand side), they use \(\text{crs}\) generated by \(\text{Setup}\) and \(\text{ORegHK}\) invokes \(\text{Gen}\); in the ideal world (on the right-hand side), they use \(\text{crs}\) simulated by \(\text{Setup}\) and \(\text{ORegHK}\) invokes \(\text{Gen}\).

**Remark.** We give several remarks on our formalization.

- We do not require simulated version of Reg and Upd since both of them are public.
- We allow the adversary to choose \(\text{pk}_i^*\) at any point, only functions \(f_i\) and *types* of public keys (i.e., honest, malicious, honest but corrupted) are chosen “very selectively”.
- The set \(CH\) does not give the timing to invoke \(\text{OCorHK}\). One could let the adversary make an explicit query; however we call the oracle automatically just after invocation of \(\text{ORegHK}\). This gives a simple but not weaker model in the very selective setting. In the definition, \(|H \cap [i]|\) is the first item of the response of \(\text{ORegHK}(f_i^x)\).
- In very selective SIM-security, there is no need to consider post-challenge queries. This relies on the fact that the adversary should state the set \(CH\) at the beginning, so the pre-challenge and post-challenge corruption queries are equivalent in the very-selective SIM-security setting.
5 Compact Reg-FE from Multi-instance Slotted Reg-FE

In this section, we define multi-instance slotted Reg-FE and give a transformation to get compact Reg-FE. Our transformation works well for both IND and SIM security.

5.1 Multi-instance Slotted Reg-FE

Algorithms. A multi-instance slotted Reg-FE for the functionality $F = \{f : Y \rightarrow Z\}$, consists of eight efficient algorithms:

- Setup($1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, F$) $\rightarrow$ crs: It takes as input security parameter $1^\lambda$, maximum instance index $1^m$, maximum slot indices $1^{L_1}, \ldots, 1^{L_m}$ of every instances and functionalities $F$, outputs common reference string crs.
- Gen(crs, q, i) $\rightarrow$ (pk$_{q,i}$, sk$_{q,i}$): It takes as input crs, instance index $q \in [m]$, and slot index $i \in [L_q]$, outputs key pair (pk$_{q,i}$, sk$_{q,i}$).
- Ver(crs, q, i, pk$_{q,i}$) $\rightarrow$ 0/1: It takes as input crs, $q \in [m]$, $i \in [L_q]$ and pk$_{q,i}$, outputs a bit.
- Agg$_+$ (crs) $\rightarrow$ mpk$_+$. It takes as input crs, outputs the shared parts of master public key mpk$_+$.
- Agg(crs, q, (pk$_{q,i}$, f$_q,i$)$_{i \in [L_q]}$) $\rightarrow$ (mpk$_q$, (hsk$_{q,j}$)$_{j \in [L_q]}$): It takes as input crs, $q \in [m]$, a series of pk$_{q,i}$ with f$_q,i$ $\in F$ for all $i \in [L_q]$, outputs master public key mpk$_q$ and helper keys hsk$_{q,j}$ for instance $q$.
- Enc$_+$ (mpk$_+$, x) $\rightarrow$ ct$_+$. It takes mpk$_+$ and message $x \in X$ as input, outputs ciphertext ct$_+$.
- Enc(mpk$_q$) $\rightarrow$ ct$_q$: It takes as input mpk$_q$ (for some $q \in [m]$), outputs ciphertext ct$_q$.
- Dec(sk, hsk, (ct$_+$, ct$_q$)) $\rightarrow$ z/⊥: It takes as input sk, hsk, ct$_+$ and ct$_q$ (for some $q \in [m]$), outputs $z \in \mathbb{Z}_p$ or a special symbol ⊥.

We require that Agg$_+$, Agg and Dec are deterministic, and Enc$_+$ and Enc share the random coin space Coin. And we allow the case that some instance $q^*$ to be empty, namely Agg(crs, $q^*$, ·) takes (pk$_{q^*,i}$, f$_{q^*,i}$) = (⊥, ⊥) for all $i \in [L_{q^*}]$ as input, and return mpk$_{q^*}$ = ⊥ and hsk$_{q^*,j}$ = ⊥ for all $j \in [L_{q^*}]$, and we allow Enc to take mpk$_{q^*}$ = ⊥ as input and output ct$_{q^*}$ = ⊥.

Completeness. For all $\lambda, m, L_1, \ldots, L_m \in \mathbb{N}$, all $F$, all $q \in [m]$ and $i \in [L_q]$, we have

$$\Pr \left[ \text{Ver(crs, q, i, pk}_{q,i} \right] = 1 \left| \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, F); (pk_{q,i}, sk_{q,i}) \leftarrow \text{Gen(crs, q, i)} \right] = 1.$$

Correctness. For all $\lambda, m, L_1, \ldots, L_m \in \mathbb{N}$, all $F$, all $q^* \in [m]$ and $i^* \in [L_{q^*}]$; all crs $\leftarrow$ Setup($1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, F$), all (pk$_{q^*,j}$, sk$_{q^*,j}$) $\leftarrow$ Gen(crs, $q^*$, $i^*$); all {pk$_{q,j}$}$_{j \in [L_q]\setminus\{i^*\}}$ such that Ver(crs, $q^*$, i, pk$_{q,j}$) = 1; for all $x \in X$, f$_{q,i}$ $\in F$; we have

$$\Pr \left[ \text{Dec(sk}_{q^*,j}, hsk_{q^*,j}, (ct_+, ct_{q^*})) = f_{q^*,j}(x) \right| \text{mpk}_+ \leftarrow \text{Agg}_+(\text{crs}); \text{(mpk}_q, (\text{hsk}_{q,j})_{j \in [L_q]}) \leftarrow \text{Agg}(\text{crs}, q^*, (\text{pk}_{q^*,j}, f_{q^*,j})_{j \in [L_{q^*}]}); s \leftarrow \text{Coin}; ct_+ \leftarrow \text{Enc}_+(\text{mpk}_+, x); ct_{q^*} \leftarrow \text{Enc}(\text{mpk}_q, s) \right] = 1.$$

Ciphertext Compactness. For all $\lambda, m, L_1, \ldots, L_m \in \mathbb{N}$, all $F$, all $q \in [m]$ and $i \in [L_q]$; all crs $\leftarrow$ Setup($1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, F$), all (pk$_{q,i}$, sk$_{q,i}$) $\leftarrow$ Gen(crs, q, i) such that Ver(crs, q, i, pk$_{q,i}$) = 1; for all $x \in X$, f$_{q,i}$ $\in F$; all mpk$_+$ $\leftarrow$ Agg$_+$ (crs), all (mpk$_q$, (hsk$_{q,j}$)$_{j \in [L_q]}$) $\leftarrow$ Agg(crs, q, (pk$_{q,i}$, f$_{q,i}$)$_{i \in [L_q]}$), all ct$_+$ $\leftarrow$ Enc$_+$ (mpk$_+$, x), all ct$_q$ $\leftarrow$ Enc (mpk$_q$, we have

$$|ct_+| = |x| + \text{poly}(\lambda) \quad \text{and} \quad |ct_q| = \text{poly}(\lambda).$$
**IND-security in Joint Challenge Setting.** For all stateful PPT adversary $\mathcal{A}$, the following advantage function is negligible in $\lambda$:

\[
\text{IndAdv}_{\mathcal{A}}^{\text{miReg-FE}}(\lambda) = \Pr[b = b']
\]

\[
\begin{align*}
&\text{where the oracles work as follows with the initial setting } C = \emptyset \text{ and } D_{q,i} = \emptyset \text{ for all } q \in [m], i \in [L_q]: \\
&- \text{OGen}(q,i) \text{ run } (\text{pk}, sk) \leftarrow \text{Gen}(\text{crs}, q, i), \text{ set } D_{q,i}[\text{pk}] = sk \text{ and return pk}. \\
&- \text{OCor}(q,i, pk) \text{ return } D_{q,i}[pk] \text{ and update } C = C \cup \{(q, i, pk)\}. \\
&\text{and for all } q \in [m], i \in [L_q], \text{ we require that}
\end{align*}
\]

\[
D_{q,i}[pk^*_q] = \bot \iff \text{Ver}(\text{crs}, q, i, pk^*_q) = 1 \quad \text{and} \quad (q, i, pk^*_q) \in C \lor D_{q,i}[pk^*_q] = \bot \implies f^*_q(x^*_q) = f^*_q(x^*_i).
\]

In IND-security model, we allow the case that some instance $q^*$ to be empty, namely $\mathcal{A}$ submit the challenge pairs $(pk^*_{q^*}, f^*_{q^*}) = (\bot, \bot)$ for all $i \in [L_{q^*}]$, and challenge ciphertext $ct^*_q = \bot$. We use $\text{IndAdv}_{\mathcal{A}}^{\text{miReg-FE}}(\lambda)$ to denote the advantage function. Analogous to sReg-ABE [HLWW23], there is no need to give mpk and hsk_1,...,hsk_L to $\mathcal{A}$ explicitly and to consider post-challenge queries.

**Very Selective SIM-security in Joint Challenge Setting.** For all stateful PPT adversary $\mathcal{A}$, there exists simulator (Setup, Gen, Enc_*, Enc) such that the following distributions are indistinguishable

\[
\begin{align*}
&\{x^*, \{L_q, M^*_q, C^*_q, \{f^*_{q,i}\}_{i \in [L_q]}\}_{q \in [m]} \leftarrow \mathcal{A}(1^\lambda)\} \\
&\{\text{crs} \leftarrow \text{Setup}(1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, F)\} \\
&\{pk^*_q \leftarrow \text{Agg}_*(\text{crs})\}_{q \in [m]} \\
&\{\text{mpk}_q \leftarrow \text{Agg}(\text{crs}, q, (pk^*_{q,1}, f^*_{q,1}), \ldots, (pk^*_{q,L_q}, f^*_{q,L_q}))\}_{q \in [m]} \\
&\{\text{sk} \leftarrow \text{Coin}, ct^*_q \leftarrow \text{Enc}_*(\text{mpk}_q, x^*_q, s)\}_{q \in [m]} \\
&\{\text{ct}_q \leftarrow \text{Enc}(\text{mpk}_q, s)\}_{q \in [m]} \\
&\{\text{ct}_q \leftarrow \mathcal{A}(ct^*_q, \{\text{ct}_q\}_{q \in [m]})\}_{q \in [m]} \\
&\{\text{sk} \leftarrow \text{Gen}(\text{crs}), \{pk^*_q, f^*_{q,i}\}_{q \in [m]} \leftarrow \mathcal{A}(\text{crs})\} \\
&\{\text{ct}^*_q \leftarrow \text{Enc}_*(\text{td}), ct^*_q \leftarrow \text{Enc}(\text{pk}^*_{q,1}, \ldots, pk^*_{q,L_q}, \text{td})\}_{q \in [m]} \\
&\{\text{ct}_q \leftarrow \mathcal{A}(ct^*_q, \{\text{ct}_q\}_{q \in [m]})\}_{q \in [m]}
\end{align*}
\]

where $M^*_q, C^*_q \subseteq [L_q]$ for $q \in [m]$ denote the sets of malicious and corrupted slots in instance $q$, and the oracles work as follows with initial setting $C_q = \emptyset$ and $D_{q,i} = \emptyset$ for all $i \in [L_q]$ and $q \in [m]$: 

- OGen$(q,i)$: run $(\text{pk}, sk) \leftarrow \text{Gen}(\text{crs}, q, i)$, set $D_{q,i}[pk] = sk$ and return pk.
- OCor$(q,i, pk)$: return $D_{q,i}[pk]$ and update $C_q = C_q \cup \{(i, pk)\}$.

In the ideal world, OGen invokes $\tilde{\text{Gen}}$ instead of Gen; and the following restrictions: for all $q \in [m],$

\[
\begin{align*}
i \in M^*_q &\implies D_{q,i}[pk^*_q] = \bot \land \text{Ver}(\text{crs}, q, i, pk^*_q) = 1 \\
i \in C^*_q &\implies (i, pk^*_q) \in C_q \\
i \in [L_q] \setminus (M^*_q \cup C^*_q) &\implies D_{q,i}[pk^*_q] \neq \bot \land (i, pk^*_q) \notin C_q
\end{align*}
\]
In SIM-security model, we allow the case that some instance \( q^* \) to be empty, namely \( \mathcal{A} \) gives \( \mathcal{M}^*_q, C^*_q = \emptyset \), and the challenge functions \( f^*_{q^*} = \bot \), challenge public keys \( \text{pk}^*_q = \bot \) for all \( i \in [L_{q^*}] \), and we have challenge ciphertext \( \text{ct}^*_q = \bot \) (resp. \( \text{ct}^*_q = \bot \)) in real (resp. ideal) world. We use \( \text{Adv}^\text{miReg-FE}_\mathcal{A} \) to denote the advantage function. Similarly, there is no need to give \( \text{mpk}_q, \{ \text{mpk}_q, \text{hsk}_{q,1}, \ldots, \text{hsk}_{q,L_q} \}_{q \in [m]} \) to \( \mathcal{A} \) explicitly in real game (or explicitly in simulation game) and consider post-challenge queries.

### 5.2 Compact Reg-FE

We give a generic transformation from multi-instance slotted Reg-FE to Reg-FE (c.f. Section 2.2) with compact ciphertext. Here we will apply a conceptual change to multi-instance slotted Reg-FE: we will always add an instance with index 0 and count slot index from 0 instead of 1. Namely, Setup that takes \( 1^m \) and \( 1^L, 1^{L_1}, \ldots, 1^{L_m} \) as input will give us \( m + 1 \) instances indexed by 0, 1, \ldots, \( m \); for each \( q \in [0, m] \), the \( q \)-th instance has \( L_q \) slots indexed by 0, 1, \ldots, \( L_q - 1 \). Clearly, this does not change correctness and security. Note that in the remaining subsections of Section 5, we use two difference indices \( i \) and \( j \), respectively referring to the global range from 0 to \( L - 1 \) and instances’ internal ranges from 0 to \( L_q - 1 \) (for each \( q \in [0, m] \)).

**Auxiliary Data Structure.** We will count users from 0 and set \( \text{aux} = (\text{ctr}, \mathcal{D}_1, \mathcal{D}_2, \text{mpk}) \):

- Counter \( \text{ctr} \in [0, L] \) is the current number of registered users in the system, or the index of the next user.
- \( \mathcal{D}_1 \) is a dictionary that maps \( q \in [0, m] \) and \( j \in [0, 2^q - 1] \) to public key \( \text{pk}_q^j \) and function \( f_{q,j} \).
- \( \mathcal{D}_2 \) is a dictionary that maps \( q \in [0, m] \) and \( i \in [0, L - 1] \) to a helper key \( \text{hsk}_q^i \).
- \( \text{mpk} \) will be in the form \( (\text{ctr}, \text{mpk}_q, \ldots, \text{mpk}_m) \).

Initially, we set \( \text{ctr} = 0, \mathcal{D}_1 = \emptyset, \mathcal{D}_2 = \emptyset, \text{mpk} = (0, \bot, \ldots, \bot) \); the system is overloaded when \( \text{ctr} = L \).

**Generic Transformation.** Our Reg-FE with compact ciphertext works as follows with multi-instance slotted Reg-FE \( (\text{mSetup}, \text{mGen}, \text{mVer}, \text{mAgg}, \text{mAgg}, \text{mEnc}, \text{mEnc}, \text{mDec}) \):

- Setup\( (1^L, 1^L, F) \): Compute \( m = \log L \), output
  \[
  \text{crs} \leftarrow \text{mSetup}(1^L, 1^m, 1^{2^0}, \ldots, 1^{2^m}, F)
  \]
- Gen\( (\text{crs}, \text{aux}) \): Parse \( \text{aux} = (\text{ctr}, \mathcal{D}_1, \mathcal{D}_2, \text{mpk}) \) and run
  \[
  (\text{pk}^{\text{cr}_q}_q, \text{sk}^{\text{cr}_q}_q) \leftarrow \text{mGen}(\text{crs}, q, \text{ctr} \ mod \ 2^q), \ \forall q \in [0, m]
  \]
  Output \( \text{pk} = (\text{ctr}, \text{pk}_q^{\text{cr}_0}, \ldots, \text{pk}_q^{\text{cr}_m}) \) and \( \text{sk} = (\text{ctr}, \text{sk}_q^{\text{cr}_0}, \ldots, \text{sk}_q^{\text{cr}_m}) \).
- Reg\( (\text{crs}, \text{aux}, \text{pk}, f) \): Parse \( \text{aux} = (\text{ctr}, \mathcal{D}_1, \mathcal{D}_2, \text{mpk}) \) where \( \text{mpk} = (\text{ctr}, \text{mpk}_q, \ldots, \text{mpk}_m) \) and \( \text{pk} = (\text{ctr}, \text{pk}_q^{\text{cr}_0}, \ldots, \text{pk}_m^{\text{cr}_m}) \). Abort if the following does not hold: \( \text{ctr} = \text{ctr}_\text{pk} \) and \( \text{mVer}(\text{crs}, q, \text{ctr} \ mod \ 2^q, \text{pk}^{\text{cr}_q}_q) = 1 \), \( \forall q \in [0, m] \).

For each \( q \in [0, m] \), update \( \mathcal{D}_1[q, \text{ctr} \ mod \ 2^q] = (\text{pk}^{\text{cr}_q}_q, f) \); furthermore, if \( \text{ctr} + 1 = 0 \ mod \ 2^q \), run
\[
(\text{mpk}'_q, \text{hsk}'_{q,0}, \ldots, \text{hsk}'_{q,2^q-1}) \leftarrow \text{mAgg}(\text{crs}, q, \mathcal{D}_1[q, 0], \ldots, \mathcal{D}_1[q, 2^q - 1])
\]
and update \( \mathcal{D}_2[q, \text{ctr} - 2^q + 1 + j] = \text{hsk}'_{q,j} \) for all \( j \in [0, 2^q - 1] \); otherwise, set \( \text{mpk}'_q = \text{mpk}_q \). Output \( \text{mpk} = (\text{ctr} + 1, \text{mpk}_q, \ldots, \text{mpk}_m) \) and \( \text{aux} = (\text{ctr} + 1, \mathcal{D}_1, \mathcal{D}_2, \text{mpk}) \).
Let $(m\text{Setup}, m\text{Gen}, m\text{Enc}_*, m\text{Dec})$ be the simulator of multi-instance slotted Reg-FE, to build the simulator of the Reg-FE, we need the following auxiliary data structure and deterministic algorithm which simulate the slot filling procedure in Reg, to determine the slots’ filling state after all users have registered.
Auxiliary Data Structure.

- $D, R$ are dictionaries that map $q \in [0, m]$ and $j \in [0, 2^q - 1]$ to index $i$.
- $M_q^*, C_q^*$ are the same sets as the definition of SIM-security of multi-instance slotted Reg-FE.

Initially, we set $D = \emptyset, R = \emptyset$ and $M_q^*, C_q^* = \emptyset$ for all $q \in [0, m]$.

Auxiliary Algorithm. Assume the Reg-FE mostly supports $L = 2^m$ users, and $CK, HK \subseteq [0, L' - 1], CH \cup HK = [0, L' - 1]$ for some $L' \leq L$, the algorithms works as follow:

- Fillslot($CK, HK, CH$): For all $i \in [0, L' - 1]$: for each $q \in [0, m]$, update $D[q, i \mod 2^q] = i$; furthermore, if $i + 1 = 0 \mod 2^q$, update $R[q, j] = D[q, j] \forall j \in [0, 2^q - 1]$. Output $R$.

Simulator. The simulator of our multi-instance Reg-QFE is as follows:

- $\widehat{\text{Setup}}(1^\lambda, 1^\mu, F; \{f_i\}_{i \in CK \cup HK}, \{\mu_i\}_{i \in CK \cup CH})$: Let $m = \log L$, run $R \leftarrow$ Fillslot($CK, HK, CH$). For all $q \in [0, m]$:  
  - If $2^q \leq L'$, for all $j \in [0, 2^q - 1]$: fetch $R[q, j] = i$ and output $f_{q,j} = f_i$, furthermore, if $i \in CK \cup CH$, output $\mu_{q,j} = \mu_i$ and update
    
    \[
    \begin{cases}
    M_q^* = M_q^* \cup \{j\} & \text{if } i \in CK \\
    C_q^* = C_q^* \cup \{j\} & \text{if } i \in CH \\
    \end{cases}
    \]
  
  - If $2^q > L'$, for all $j \in [0, 2^q - 1]$, output $f_{q,j} = \bot$.

And run

\[
(\overline{crs}, mtd) \leftarrow m\widehat{\text{Setup}}(1^\lambda, 1^m, 1^{2^0}, \ldots, 1^{2^m}, F, \{\{f_{q,j}\}_{j \in [0, 2^q - 1]}, \{\mu_{q,j}\}_{j \in M_q \cup C_q}\}_{q \in [0, m]})
\]

Output $\overline{crs}$, and set trapdoor as $td = mtd \cup R$.

- $\widehat{\text{Gen}}(\overline{crs}, aux; td)$: Parse aux = ($ctr, D_1, D_2, mpk$) and run

\[
(\overline{pk}_q, \overline{sk}_q) \leftarrow m\widehat{\text{Gen}}(\overline{crs}, q, ctr \mod 2^q; td), \forall q \in [0, m]
\]

Output

$\overline{pk} = (ctr, \overline{pk}_0, \ldots, \overline{pk}_m)$ and $\overline{sk} = (ctr, \overline{sk}_0, \ldots, \overline{sk}_m)$.

- $\widehat{\text{Enc}}((\overline{pk}_1, \ldots, \overline{pk}_q); td)$: Parse $td = (mtd, R)$ and $pk_q = (i, pk_{q,0}^l, \ldots, pk_{q,m}^l)$. For all $q \in [0, m]$:  
  - If $2^q \leq L'$, for all $j \in [0, 2^q - 1]$: fetch $R[q, j] = i$ and set $\overline{pk}_{q,j} = pk_{q,j}^i$
  - If $2^q > L'$, for all $j \in [0, 2^q - 1]$, set $\overline{pk}_{q,j} = \bot$.

Compute

\[
\overline{ct}_q \leftarrow m\overline{\text{Enc}}(\overline{pt}_q; mtd), \forall q \in [0, m]
\]

Output

$\overline{ct} = (ctr, \overline{ct}_0, \overline{ct}_1, \ldots, \overline{ct}_m)$.

The reader can find the sanity check in Appendix D.
5.4 Proof

We prove the following technical lemma this immediately proves Theorem 3.

**Lemma 3.** For all adversaries $\mathcal{A}$, there exist adversary $\mathcal{B}$ such that:

$$\text{Adv}^{Reg\text{-FE}}_\mathcal{A}(\lambda) \leq \text{Adv}^{\text{mReg\text{-FE}}}_\mathcal{B}(\lambda) + \text{negl}(\lambda)$$

where $\text{Time}(\mathcal{B}) \approx \text{Time}(\mathcal{A})$.

**Game Sequence.** Suppose that $c_{\mathcal{K}}$ is the common reference string, $x^*$ is the challenge, with some $L' \leq L$, $\{f^*_q\}_{q\in[0,L'-1]}$ are challenge functions that chosen at the beginning, $C_{\mathcal{K}}, H_{\mathcal{K}}$ and $C_{\mathcal{H}}$ are the set of register corrupted (malicious) key index, register honest key index and corrupted honest key index such that $C_{\mathcal{K}}, H_{\mathcal{K}} \subseteq [0, L' - 1]$, $C_{\mathcal{K}} \cup H_{\mathcal{K}} = [0, L' - 1], \text{CH} \subseteq H_{\mathcal{K}}$ and $C_{\mathcal{K}} \cap H_{\mathcal{K}} = \emptyset$. $\{pk^*_q\}_{q\in[0,L']}$ are challenge public keys with the form of

$$\text{pk}^*_q = (i, (pk^0_q)^*, \ldots, (pk^m_q)^*)$$

Recall that $\mathcal{H}$ record the $(pk^*_q, sk^*_q)_{q\in H_{\mathcal{K}}}$ that generated in $O\text{RegHK}(\cdot)$.

- $G_0$: This is the real game, recall that we have
  - $\text{crs}$ is in the form of
    $$\text{crs} \leftarrow \text{mSetup}(1^\lambda, 1^m, 1^{2^0}, \ldots, 1^{2^m}, F)$$
  - For each $i \in H_{\mathcal{K}}$, each $(pk^*_i, sk^*_i) \in \mathcal{H}$ is in the form of
    $$\text{pk}^*_i = (i, (pk^0_i)^*, \ldots, (pk^m_i)^*) \text{ and } \text{sk}^*_i = (i, (sk^0_i)^*, \ldots, (sk^m_i)^*).$$
  - $\text{ct}^*$ for $x^*$ is in the form of
    $$\text{ct}^* = (L, \text{ct}^*_0, \ldots, \text{ct}^*_m)$$

where $\text{ct}^*_q \leftarrow \text{mEnc}_i(\text{mpk}_s, x^*; s)$, and $\text{ct}^*_q \leftarrow \text{mEnc}(\text{mpk}_q^*; s)$ for all $q \in [0, m]$, with the same random coin $s \leftarrow \text{Coin}$.

- $G_1$: Identical to $G_0$ except that we replace $(\text{mSetup}, \text{mGen}, \text{mEnc}_i, \text{mEnc})$ with $(\text{m\overline{Setup}}, \text{mGen}, \text{m\overline{Enc}}_i, \text{m\overline{Enc}})$. In particular:
  - $\text{crs}$ is replaced with $\overline{\text{crs}}$ where
    $$(\overline{\text{crs}}, \text{mtd}) \leftarrow \text{mSetup}\left(1^\lambda, 1^m, 1^{2^0}, \ldots, 1^{2^m}, F, \{\{f^*_q\}_{q\in[0,2^t-1]}\}_{q\in[0, m]}\right)$$

where

$$f^*_q = \begin{cases} f^*_{r(q,l)} & \text{if } 2^q \leq L' \\ \bot & \text{if } 2^q > L' \end{cases}$$

with $r \leftarrow \text{Fillslot}(C_{\mathcal{K}}, H_{\mathcal{K}}, C_{\mathcal{H}})$.

- For each $i \in H_{\mathcal{K}}$, each $(pk^*_i, sk^*_i) \in \mathcal{H}$ is in the form of
  $$\text{pk}^*_i = (i, (\overline{pk}^0_i)^*, \ldots, (\overline{pk}^m_i)^*) \text{ and } \text{sk}^*_i = (i, (\overline{sk}^0_i)^*, \ldots, (\overline{sk}^m_i)^*).$$

where $(\overline{pk}^*_q)^*, (\overline{sk}^*_q)^* \leftarrow \text{mGen}(\text{crs}, q, i \text{ mod } 2^t; \text{td})$, for all $q \in [0, m]$.

- $\text{ct}^*$ for $x^*$ is in the form of
  $$\text{ct}^* = (L', \overline{\text{ct}}^*_0, \overline{\text{ct}}^*_1, \ldots, \overline{\text{ct}}^*_m)$$

where $\overline{\text{ct}}^*_q \leftarrow \text{m\overline{Enc}}_i(\text{td})$, and $\overline{\text{ct}}^*_q \leftarrow \text{m\overline{Enc}}(\text{pk}^*_q, 0, \ldots, \text{pk}^*_q, 2^{t-1}); \text{td}$ for all $q \in [0, m]$. With

$$\text{pk}^*_q = \begin{cases} (\text{pk}^*_{q,0})^* & \text{if } 2^q \leq L' \\ \bot & \text{if } 2^q > L' \end{cases}$$
We reduce the security to multi-instance slotted Reg-FE, where the instances \( q^* \in \{ q : 2^q > L' \} \) are empty: we have \( f^*_{q^*,j} \cdot pk^*_{q^*,j} = \perp \) for all \( j \in [0, 2^q - 1] \), and \( M^*, C^* = \emptyset \). Observe that the game \( G_1 \) can be simulated using the simulator by setting \( \mu_i = f^*_i(x^*) \)

### 6 Pre-Constrained Slotted Reg-IPFE

In this section, we introduce the notion of pre-constrained slotted Reg-IPFE; the definition for general functionality is deferred to Appendix A. We present our pairing-based construction with very selective SIM-security in Section 6.1. We explain how this implies (standard, without pre-constrain) slotted Reg-IPFE with very selective SIM-security in Section 6.4 and how this derives (standard, without pre-constrain) slotted Reg-IPFE with selective IND-security in Section 6.5.

**Functionality and Definition.** A Pre-constrained Slotted Reg-IPFE is a generalized slotted Reg-FE for linear functionality:

\[
F = \{ f : x \rightarrow xf \}
\]

where \( x, f \in \mathbb{Z}_p^{1 \times n} \). The generalization in multi-instance version is in the following four aspects:

- **Algorithm:** Setup takes as input security parameter \( 1^\lambda \), maximum instance index \( 1^m \), maximum slot indices \( 1^{L_1}, \ldots, 1^{L_m} \) of every instances, function parameter \( 1^{n_1}, 1^{n_2} \) and pre-constrained matrix \( M \in \mathbb{Z}_p^{n_1 \times n_2} \), outputs common reference string \( crs \).
- **Correctness:** for all \( \lambda, m, L_1, \ldots, L_m, n_1, n_2 \in \mathbb{N} \), all \( q^* \in [m] \) and \( i^* \in [L_q^*] \); all \( crs \leftarrow \text{Setup}(1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, 1^{n_1}, 1^{n_2}, M) \), all \( (pk_{q^*,i^*}, sk_{q^*,i^*}) \leftarrow \text{Gen}(crs, q^*, i^*) \); all \( (pk_{q^*,i^*}, sk_{q^*,i^*}) \in [L_q^*] \); such that \( \text{Ver}(crs, q^*, i, pk_{q^*,i^*}) = 1 \); for all \( x \in \mathbb{Z}_p^{1 \times n_1} \), \( f_{q^*,i^*} \in \mathbb{Z}_p^{1 \times n_2} \), we have

\[
\Pr \left[ \text{Dec}(sk_{q^*,i^*}, hsk_{q^*,i^*}, (ct_+, ct_{q^*}^i)) = xMf_{q^*,i^*}^T \right]\bigg| \begin{bmatrix} mpk_+ \leftarrow \text{Agg}_+(crs); \\
(mpk_{q^*}, (hsk_{q^*,i^*})_{j \in [L_q^*]}) \leftarrow \text{Agg}(crs, q^*, (pk_{q^*,i^*}, f_{q^*,i^*})_{i \in [L_q^*]}); \\
s \leftarrow \text{Coin}; ct_+ \leftarrow \text{Enc}_+(mpk_+, x; s); ct_{q^*} \leftarrow \text{Enc}(mpk_{q^*}, s) \end{bmatrix} = 1.
\]

- **IND-security:** We let the adversary to choose \( M \) at the beginning and require that \( x_0^* M(f_{q^*,i^*})^T = x_1^* M(f_{q^*,i^*})^T \) for the case \( \{q, i, pk_{q,i}^*\} \in C \lor D_{q,i} \left| pk_{q,i}^* \right| = \perp \).
- **SIM-security:** We let the adversary to choose \( M \) at the beginning and give \( M \) and \( \{x^* M(f_{q,i}^*)\}_{i \in q^* \cup C_q^*} \) to Setup.

It is straightforward to verify that setting \( n_1 = n_2 = n \) and \( M = I_n \) yields standard slotted Reg-IPFE defined above.

**Group-based Simulator.** We also require the existence of the group-based simulator \( \text{Setup}_G \). For all \( \lambda, m, n_1, n_2 \in \mathbb{N} \), all \( L_1, \ldots, L_m \in \mathbb{N} \), all \( M^*, C^* \subseteq [L_q] \), all \( M \in \mathbb{Z}_p^{n_1 \times n_2} \), all \( f_{q,1}, \ldots, f_{q,L_q} \in \mathbb{Z}_p^{1 \times n_2} \) and \( \mu_{q,i} \in \mathbb{Z}_p \), there exist a group-based algorithm \( \text{Setup}_G \) such that

\[
\text{Setup}_G(1^\lambda, 1^m, 1^{n_1}, 1^{n_2}, [M], [M]_2; \{1^{L_q}, (f_{q,i})_{i \in [L_q]}; \{[\mu_{q,i}]_1, [\mu_{q,i}]_2\}_{i \in M \cup C_q^*} \}_{q \in [m]}) \equiv \text{Setup}(1^\lambda, 1^m, 1^{n_1}, 1^{n_2}, M; \{1^{L_q}, (f_{q,i})_{i \in [L_q]}; \{[\mu_{q,i}]_1, [\mu_{q,i}]_2\}_{i \in M \cup C_q^*} \}_{q \in [m]})
\]

For simplicity, for the group-based simulator, we do not distinguish the notation of \( \text{Setup}_G \) and \( \text{Setup} \).
6.1 Scheme

Assuming a QA-NIZK $\Pi_0 = (\text{LGen}, \text{LPrv}, \text{LVer}, \text{LSim})$ for linear space over bilinear groups, see Section 2.4; a Bi-PKE $\Pi_1 = (\text{Gen}, \text{Enc}, \text{Dec}_1)$ with linear decryption over bilinear groups, see Section 2.5. Assuming that $|\text{ict}| = |\text{isk}| = n$, our multi-instance slotted PReg-IPFE scheme, with a shared pre-constrained $M \in \mathbb{Z}_p^{n_1 \times n_2}$ works as follows in the prime-order bilinear group:

- Setup($1^k, 1^m, 1^{L_1}, \ldots, 1^{L_m}, 1^{n_1}, 1^{n_2}, M$) : Run $\emptyset := (p, G_1, G_2, G_T, e) \leftarrow G(1^k)$, ([ipk]$_1$, [ipk]$_2$, isk) $\leftarrow \text{Gen}_1(1^k)$. Sample shared parts:

$$A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}, W \leftarrow \mathbb{Z}_p^{(2k+1) \times (n_1+n_2)}.$$

For each instance $q \in [m]$, sample $B_q \leftarrow \mathbb{Z}_p^{(k+1) \times k}$, and for all $i \in [L_q]$, do following operations:

- Run ([ict]$_{q,i},$ [ict]$_{q,i})$ $\leftarrow \text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, 0)$, for $s \in \{1, 2\}$, set

$$[\mathbf{M}_{q,i}]_s = \begin{bmatrix} M & 0_{n_1} \\ 0_{n_2 \times n_1} & \text{ict}^{\gamma}_{q,i} \end{bmatrix} \in \mathbb{C}^{(n_1+n_2) \times (n_1+1)}.

- Sample

$$W_{q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)(n_1+1)}, R_{q,i} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+1)}, r_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k}.

- Run (\text{crs}_{q,i}, r_{q,i}) \leftarrow \text{LGen}(1^4, G_1, [A_{q,i}])$, where $A_{q,i} = (\begin{bmatrix} A \cr R_{q,i} \end{bmatrix}) \in \mathbb{Z}_p^{(3k+2) \times (2k+1)}$.

Output

$$\text{crs} = \begin{cases} [A, AW]_1, \\
\{\text{crs}_{q,i}, \begin{bmatrix} R_{q,i}, AW_{q,i}(\mathbf{M}_{q,i} \otimes I_{k+1})), AW_{q,i} \end{bmatrix} \} \in [L_q], \\
\{[\mathbf{M}_{q,i}, B_q \mathcal{F}^r_{q,i}, W_{q,i}(\mathbf{M}_{q,i} \otimes B_q \mathcal{F}^r_{q,i}) + WM_{q,i}] \} \in [L_q], \\
\{[W_{q,i}(\mathbf{M}_{q,i} \otimes B_q \mathcal{F}^r_{q,i})] \} \{\mathbb{Z}_p^{1 \times k} \}. 
\end{cases}_{q \in [m]}$$

- Gen(crs, $q, i$): Sample $U_{q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)}$. Define $F_{q,i} = (\begin{bmatrix} T_{q,i} \\ Q_{q,i} \end{bmatrix}) = [A_{q,i} U_{q,i}] \in \mathbb{Z}_p^{(3k+2) \times (2k+1)}$ and run

$$\pi_{q,i} \leftarrow \text{LPrv}([\text{crs}_{q,i}, [F_{q,i}]_1, U_{q,i}]).$$

Fetch $\{[B_q \mathcal{F}^r_{q,i}] \} \in [L_q \setminus \{i\}]$ from crs and output

$$p_k_{q,i} = ([A_{q,i}, R_{q,i}, U_{q,i}]) \begin{cases} T_{q,i} \\ Q_{q,i} \end{cases} \begin{cases} h_{q,i} \\ h_{q,i} \end{cases} \begin{cases} h_{q,i} \\ h_{q,i} \end{cases} \quad \text{and} \quad \text{sk}_{q,i} = U_{q,i}.$$

- Ver(crs, $q, i, p_k_{q,i}$): Parse $p_k_{q,i} = ([T_{q,i}, Q_{q,i}]_1, \{[h_{q,i}]_2 \} \in [L_q \setminus \{i\}], \pi_{q,i})$. Write $F_{q,i} = (\begin{bmatrix} T_{q,i} \\ Q_{q,i} \end{bmatrix})$ and check

$$\text{LVer}([\text{crs}_{q,i}, [F_{q,i}]_1], \pi_{q,i}) = 1.$$

For all $j \in [L_q \setminus \{i\}$, check

$$e([A]_1, [h_{q,j}]_2) = e([T_{q,j}]_1, [B_q \mathcal{F}^r_{q,j}]_2).$$

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

- Agg$_s$(crs): Output:

$$\text{mpk}_s = ([A, AW]_1).$$

---

Note that we employ $i$ as the index for $W_q$’s and $M_q$’s while $j$ is the index for $r_q$’s; both of them range from 1 to $L_q$. One exception is the terms with $W_q$, which is conceptually $W_{q,i}(\mathbf{M}_{q,i} \otimes B_q \mathcal{F}^r_{q,i})$ with $i = j$. Note that we do not use $t_{q,i}, \ldots, t_{q,L_q}$ and isk in the actual scheme.
- Agg(crs, q, (pk_{q,i}, f_{q,i})_{i \in [L_q]}): If q is an empty instance, on input (pk_{q,i}, f_{q,i}) = (\perp, \perp) for all i \in [L_q], abort and return \text{mpk}_q = \perp, hsk_{q,i} = \perp for all j \in [L_q]. For all i \in [L_q], parse pk_{q,i} = ([T_{q,i}, Q_{q,i}], \{[h_{q,i,j}]_{j \in [L_q]}\}_{i \in [L_q]}, \pi_{q,i}), and set \tilde{f}_{q,i} = (f_{q,i}\|\perp) \in \mathbb{Z}_p^{1 \times (n_2+1)}. Output:

$$\text{mpk}_q = \left[ \sum_{i \in [L_q]} (T_{q,i} + \text{AW}_{q,i}(M_{q,i}, f_{q,i} \otimes I_{k+1})) \right]_1,$$

and for all j \in [L_q]

$$hsk_{q,j} = \left( B_{q,j}^r f_{q,j}^r, \sum_{i \in [L_q]} \left( h_{q,j,i} + \text{W}_{q,j}(M_{q,j}, f_{q,j} \otimes B_{q,j}^r) \right) \right) \in \mathbb{Z}_p^{1 \times (n_2+1)}.$$

- Enc_s(\text{mpk}_q, x): Set \overline{x} = (x||0_n) \in \mathbb{Z}_p^{1 \times (n_1+n)}. Sample s \leftarrow \mathbb{Z}_p^{1 \times k}. Output:

$$ct_s = ([sA, sAW + \overline{x}])$$

- Enc(\text{mpk}_q): Abort and return \perp if \text{mpk}_q = \perp. Sample s \leftarrow \mathbb{Z}_p^{1 \times k}, output

$$ct_q = \left[ \sum_{i \in [L_q]} (sT_{q,i} + \text{AW}_{q,i}(M_{q,i} f_{q,i} \otimes I_{k+1})) \right]_1.$$

- Dec(s_{q',r'}, hsk_{q',i'}, (ct_s, ct_q')): Abort and return \perp if ct_q' = \perp. Parse

$$s_{q',r'} = U_{q',r'}, \quad hsk_{q',i'} = ([k_0, k_1, k_2, k_3], (ct_s, ct_q')) = ([c_{s,0}, c_{s,1}, c_{q'}])$$

Recover

$$[z_1]_T = e([c_{q'}, 1], [k_0]), \quad [z_2]_T = e([c_{s,0}, 1], [k_1]), \quad [z_3]_T = e([c_{s,0}, U_{q',r'}], [k_2]), \quad [z_4]_T = e([c_{s,0}, 1], [k_3]), \quad [z_5]_T = e([c_{s,1}], [k_3]).$$

Compute

$$[z]_T = [z_1 - z_2 - z_3 - z_4 + z_5]_T.$$

Recover z from [z]_T via brute-force DLOG and output z.

**Completeness.** For all \lambda, m, n_1, n_2 \in \mathbb{N}, all L_1, \ldots, L_m \in \mathbb{N}, all M \in \mathbb{Z}_p^{n_1 \times n_2}, all q \in [m] and i \in [L_q], all crs \leftarrow \text{Setup}(1^\lambda, 1^m, 1^{n_1}, 1^{n_2}, M, 1^{L_1}, \ldots, 1^{L_m}), and (pk_{q,i}, sk_{q,i}) \leftarrow \text{Gen}(crs, q, i), we have

$$pk_{q,i} = ([T_{q,i}, Q_{q,i}], \{[h_{q,i,j}]_{j \in [L_q]}\}_{i \in [L_q]}, \pi_{q,i}) \leftarrow ([AU_{q,i}, R_{q,i}U_{q,i}], \{[U_{q,i}B_{q,j}^r f_{q,j}^r]_{j \in [L_q]}\}_{i \in [L_q]}, \pi_{q,i}),$$

for some U_{q,i} \leftarrow \mathbb{Z}_p^{(2k_1+1) \times (k+1)} and \pi_{q,i} \leftarrow \text{LPrv}(crs_{q,i}, [A_{q,i}U_{q,i}], U_{q,i}) where (crs_{q,i}, td_{q,i}) \leftarrow \text{LGen}(1^\lambda, 1^\lambda, \pi_{q,i}) and A_{q,i} = (A_{q,i}^A) with A \leftarrow \mathbb{Z}_p^{(2k_2+1) \times (k+1)}, R_{q,i} \leftarrow \mathbb{Z}_p^{(2k_1+1) \times (k+1)}$. Then

- Write F_{q,i} = (T_{q,i}, Q_{q,i}) = (A_{q,i}U_{q,i}, U_{q,i}), we have LVer(crs_{q,i}, [F_{q,i}]_1, \pi_{q,i}) = 1 by the perfect completeness of \Pi_0 (see Section 2.4) and the fact that F_{q,i} = A_{q,i}U_{q,i};

- For each j \in [L_q] \setminus \{i\}, we have e([A], [U_{q,i}B_{q,j}^r f_{q,j}^r]) = e([AU_{q,i}], [B_{q,j}^r f_{q,j}^r]) by the definition of bilinear map e (see Section 2.1) and the fact that A \cdot U_{q,i}B_{q,j}^r f_{q,j}^r = AU_{q,i} \cdot B_{q,j}^r f_{q,j}^r.

This ensures that Ver(crs, q, i, pk_{q,i}) = 1 by the specification of Ver and readily proves the completeness.
Correctness. For all $\lambda, m, n_1, n_2 \in \mathbb{N}$, all $L_1, \ldots, L_m \in \mathbb{N}$, all $\bm{M} \in \mathbb{Z}_p^{n_1 \times n_2}$, all $q^* \in [m]$ and $i^* \in [L_q]$; all $\text{crs} \leftarrow \text{Setup}(1^\lambda, 1^m, 1^{n_1}, 1^{n_2}, \bm{M}, 1^{l_1}, \ldots, 1^{l_m})$, all $(\text{pk}_{q^*,i^*}, \text{sk}_{q^*,i^*}) \leftarrow \text{Gen}(\text{crs}, q^*, i^*)$; all $\{\text{pk}_{q^*,i^*}\}_{i \in [L_q]} \setminus \{i^*\}$ such that $\text{Ver}(\text{crs}, q^*, i, \text{pk}_{q^*,i}) = 1$; all $\bm{x} \in \mathbb{Z}_p^{1 \times n_1}$ and $\bm{f}_{q^*,i} \in \mathbb{Z}_p^{1 \times n_2}$; for $s \in \{1, 2\}$, we have:

$$
\Xi = (\bm{x}[0_n], \bar{\bm{f}}_{q^*,i^*} = (\bm{f}_{q^*,i^*}||1), [\bm{M}_{q^*,i^*}]_s = \begin{bmatrix} \bm{M} & 0_{n_1} \\ 0_{n \times n_2} & 1_{n_2} \end{bmatrix}_{ic_{q^*,i^*}}^s)
$$

where $[ic_{q^*,i^*}]_s \in \text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, 0)$ and $([\text{ipk}]_1, [\text{ipk}]_2) \in \text{Gen}_1(1^\lambda)$. And for all $s \in \mathbb{Z}_p^{1 \times k}$, we have

$$(ct^*_{s}, ct^*_{q^*}) = \left( \begin{array}{c} s_{\text{AU}} - \Xi, s_{\text{AW}} + \Xi, \sum_{i \in [L_q]} (s_{T_{q^*,i^*}} + s_{\text{AW}_{q^*,i^*}}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes \bm{I}_{k+1})) \\ e_{s_u} \end{array} \right)_{c_{q^*}}$$

$h_{\text{sk}_{q^*,i^*}} = \left( \begin{array}{c} B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*}, \sum_{i \in [L_q]} (h_{q^*,i^*} + W_{q^*,i^*}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*})) \\ k_{q^*,i^*} \end{array} \right)_{k_{q^*,i^*}}$

where

$$A_{\text{h}_{q^*,i^*}} = T_{q^*,i^*}B_{q^*,i^*}, \quad \forall i \in [L_q] \setminus \{i^*\} \quad \text{and} \quad A_{\text{U}_{q^*,i^*}} = T_{q^*,i^*}.$$

Note that here we actually consider $h_{\text{sk}_{q^*,j}}$ for $j = i^*$ and $s_{\text{sk}_{q^*,j}}$ for $i = i^*$ and all above equalities are ensured by $\text{Ver}$ and $\text{Gen}$. We have

$$
\begin{align*}
z_1 &= \sum_{i \in [L_q]} (s_{T_{q^*,i^*}}B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*} + s_{\text{AW}_{q^*,i^*}}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes \bm{I}_{k+1})B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*}) \\
&= \sum_{i \in [L_q]} (s_{T_{q^*,i^*}}B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*} + s_{\text{AW}_{q^*,i^*}}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*})) \\
&= \sum_{i \in [L_q] \setminus \{i^*\}} (s_{\text{h}_{q^*,i^*}} + s_{\text{AW}_{q^*,i^*}}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*})) \\
z_2 &= \sum_{i \in [L_q] \setminus \{i^*\}} (s_{\text{h}_{q^*,i^*}} + s_{\text{AW}_{q^*,i^*}}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*})) \\
z_3 &= \text{sk}_{q^*,i^*}B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*}, \\
z_4 &= \text{sAW}_{q^*,i^*}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*}) + \text{sAW}_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} + \text{sAW}_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} \\
z_5 &= \text{sAW}_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} + \Xi_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} + \Xi_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} \\
\end{align*}
$$

and then

$$
\begin{align*}
z &= z_1 - z_2 - z_3 - z_4 + z_5 \\
&= s_{T_{q^*,i^*}}B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*} + s_{\text{AW}_{q^*,i^*}}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*}) - \text{sAU}_{q^*,i^*}B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*} \\
&\quad - s_{\text{AW}_{q^*,i^*}}(\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \otimes B_{q^*,i^*}\bar{\bm{r}}_{q^*,i^*}) + \text{sAW}_{q^*,i^*}\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*} \\
&\quad + \text{sAW}_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} + \Xi_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} + \Xi_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} \\
&= \Xi_{\bm{M}_{q^*,i^*}\bar{\bm{f}}_{q^*,i^*}} \\
&= (\bm{x}[0_n]) \begin{bmatrix} \bm{M} & 0_{n_1} \\ 0_{n \times n_2} & 1_{n_2} \end{bmatrix}_{ic_{q^*,i^*}}^s \begin{bmatrix} \bm{f}_{q^*,i^*} \\ 1 \end{bmatrix} \\
&= \bm{x}\bar{\bm{f}}_{q^*,i^*}.
\end{align*}
$$
Here, equality (16) follows from the property of tensor product: \((a^\top \otimes I)M = a^\top \otimes M\) for matrices of proper size; equality (17) follows from the fact that \(A_{h_i,q_i,i^*} = T_iB_q r_{q_i,i^*}\) for all \(i \in [L_q] \setminus \{i^*\}\); equality (18) follows from the fact that \(T_i A_{p_i} = A_{q_i,i^*}\); equality (19) follows from the fact (15). This proves the correctness.

**Compactness and Efficiency.** Our multi-instance PReg-IPFE has the following properties:

\[
|\text{crs}| = O(L^2 \cdot n_1 \cdot n_2) \cdot \text{poly}(\lambda), \quad |\text{hsk}_{q,i}| = O(n_1) \cdot \text{poly}(\lambda), \\
|\text{mpk}_{i}| = O(n_1) \cdot \text{poly}(\lambda), \quad |\text{mpk}_{q}| = \text{poly}(\lambda), \\
|\text{ct}_{i}| = O(n_1) + \text{poly}(\lambda), \quad |\text{ct}_{q}| = \text{poly}(\lambda),
\]

where \(L = L_1 + \cdots + L_m\). Note that the total size of \(\{\text{crs}_i\}_{i \in [L]}\) is \(L \cdot \text{poly}(\lambda)\) 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 \(\text{poly}(\lambda)\).

**Security.** We have the following theorem. Given pairing-based QA-NIZK in [KW15] with unbounded simulation soundness under MDDH assumption, given Bi-PKE with linear decryption and IND-security under bi-MDDH assumption, our multi-instance slotted PReg-IPFE scheme uses prime-order bilinear group and the security can be reduced to bi-MDDH assumption.

**Theorem 4.** Assume \(\Pi_0 = (\text{LGen}, \text{LPrv}, \text{LVer}, \text{LSim})\) is a QA-NIZK with perfect completeness, perfect zero-knowledge and unbounded simulation soundness for linear space defined in Section 2.4, assuming \(\Pi_1 = (\text{Gen}_1, \text{Enc}_1, \text{Dec}_1)\) is a Bi-PKE with correctness, linear decryption and IND-security defined in Section 2.5, our multi-instance slotted PReg-IPFE scheme achieves the very selective SIM-security as the definition in Section 5.1, under bi-MDDH assumption.

### 6.2 Simulator

Recall that we allow some instance \(q^*\) to be empty, namely \(M_{q^*} = \emptyset, C_{q^*} = \emptyset\) and \(f_{q^*} = \perp, p_{k^*} = \perp\) for all \(i \in [L_q]\).

Our simulator is as follows:

- \(\text{Setup}(1^l, 1^m, 1^{L_1}, \ldots, 1^{L_m}, 1^{n_1}, 1^{n_2}, M; \{f_{q,i}\}_{i \in [L_q]}, \{\mu_{q,i}\}_{i \in M_q \cup C_q}; q \in [m]): \) Run \(\mathbb{G} := (p, G_1, G_2, G_T, e) \leftarrow G(1^l), ([ipk_1], [ipk_2], isk) \leftarrow \text{Gen}_1(1^l)\). Sample shared parts:

\[
c \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}, \quad A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}, \quad W \leftarrow \mathbb{Z}_p^{(2k+1) \times (n_1 + n_2)}.
\]

For each instance \(q \in [m]\), sample \(B_q \leftarrow \mathbb{Z}_p^{(k+1) \times k}\), for all \(i \in [L_q], s \in \{1, 2\}\), set

\[
[M_{q,i}]_s = \begin{cases}
M_{q,i} & \text{if } s = 1, \\
0 & \text{if } s = 2
\end{cases}, \quad [\text{ict}_{q,i}]_s = \begin{cases}
\text{Enc}_1([ipk_1], [ipk_2], 0) & \text{if } i \in [L_q] \setminus (M_q \cup C_q) \\
\text{Enc}_1([ipk_1], [ipk_2], \mu_{q,i}) & \text{if } i \in M_q \cup C_q
\end{cases}
\]

and for all \(i \in [L_q]\), do following operations:

- **Sample**

\[
W_{q,i} \leftarrow \mathbb{Z}_p^{(2k+2) \times (n_1 + n_2)}, \quad \tilde{R}_{q,i} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+2)}, \quad r_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k}.
\]

and compute

\[
\tilde{R}_{q,i} = \tilde{R}_{q,i} \left( \begin{array}{c} c \\ I_{2k+1} \end{array} \right).
\]

- **Run** \((\text{crs}_{q,i}, \text{td}_{q,i}) \leftarrow \text{LGen}(1^4, \mathbb{G}, [A_{q,i}]_1), \) where \(A_{q,i} = \left( \begin{array}{c} A \\ \text{R}_{q,i} \end{array} \right) \in \mathbb{Z}_p^{(3k+2) \times (2k+1)}\).
Lemma 4. We prove the following technical lemma this immediately proves Theorem 4.  

\[
\text{Adv}_A^{\mathsf{mPKE-IPE}}(\lambda) \leq L \cdot \text{Adv}_A^{\mathsf{JSS}}(\lambda) + L \cdot \text{Adv}_A^{\mathsf{BiPKE}}(\lambda) + (2L + 2L \cdot Q + 1) \text{Adv}_B^{\mathsf{MDDH}}(\lambda) + \text{negl}(\lambda)
\]

where \( L = L_1 + \ldots + L_m \) is the number of slots, \( Q \) is the maximum number of queries on a slot made by \( A \) and \( \text{Time}(B_1), \text{Time}(B_2), \text{Time}(B_3) \approx \text{Time}(A) \).
For simplicity, we prove Lemma 4 in the case of nonempty 1-instance and remove the index q in the following proof. For an empty instance, we only need to remove the terms about ct\_i\* and all pk\_i\* in the following game sequence, and notice that M\*, C\* = 0 for empty instance. In the case of m-instance, it only needs to add back index q and apply sub-sequence G_{7,t-1,0}, \ldots, G_{7,t-1,3} to each instance.

**Game Sequence.** Suppose that crs is the common reference string, M is the pre-constrained matrix, x* is the challenge, \{pk\_i\*, f\_i\*\}_{i \in [L]} are challenge public keys along with challenge functions to be registered, M\*, C\* \subseteq [L] are the sets of malicious and corrupted slots. For all i \in [L], define D\_i = \{pk\_i : D\_i, [pk\_i] = sk\_i \neq \perp\} be responses to OGen(i) and C\_i = \{pk\_i : (i, pk\_i) \in C\_1\} records public keys in D\_i that have been sent to OCor(i, -). Recall that, for each i \in [L], we require

\[
i \in M\* \implies pk\_i \notin D\_i \land Ver(crs, 1, i, pk\_i) = 1
\]

\[
i \in C\* \implies pk\_i \in C\_i
\]

\[
i \in [L] \setminus (M\* \cup C\*) \implies pk\_i \in D\_i \land pk\_i \notin C\_i
\]

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 OGen(i) for many times. We prove the Lemma 4 via dual-system method using the following game sequence.

- G\_0: This is the real game, recall that we have
  - crs is in the form:

    \[
    crs = \left(\begin{array}{c}
    [A, AW]\_1 \\
    \{crs\_i, [R\_i, AW_i(M\_i \circ \gamma\_1), AW\_i]\}_i[\in [L]] \\
    \{[M\_j, Br\_j, W\_j(M\_j \circ Br\_j) + WM\_j]_j\}_j[\in [L]] \\
    \{[W\_i(M\_i \circ Br\_j)]_j\}_i[\in [L]], i[\in [L], j[\in (L \setminus i)]
    \end{array}\right).
    \]

    where \([M\_i]_s = [M, 0\_n\_i]_s, [ict\_i]_s \in Enc_1([ipk\_1, ipk\_2, 0]) for s \in \{1, 2\}, ([ipk\_1, ipk\_2]) \in \text{Gen}\_1(1^L);\) and

    \[
    crs\_i \in \text{LGen}(1^L, G\_1, [A\_i]_1), \text{with } A\_i = \left(\begin{array}{c}
    A \\
    R\_i
    \end{array}\right).
    \]

  - For each i \in [L], each pk\_i \in D\_i is in the form

    \[pk\_i = ([AU\_i, Ru\_i]_1, [AU\_i, Ru\_i]_2)_{j \in \text{[L] \setminus \{i\}}, [\pi\_i]\_T\_i, [\pi\_i]\_Q\_i, [\pi\_i]\_h\_i\]

    where \[\pi\_i \leftarrow \text{LPrv}(crs\_i, [F\_i]_1, U\_i), F\_i = ([AU\_i]_{R\_i}), and U\_i \text{is the corresponding sk\_i.}\]

    - For all i \in [L], pk\_i\* is in the form:

      \[pk\_i\* = ([T\_i, Q\_1]_1, [B\_i, Q\_1]_2)_{j \in \text{[L] \setminus \{i\}}, [\pi\_i]\_T\_i, [\pi\_i]\_Q\_i\]

      such that Ver(crs, 1, i, pk\_i\*) = 1 which means LVer\( crs\_i, \left[\begin{array}{c}
      T\_i \\
      Q\_1
      \end{array}\right]_1, [\pi\_i]\_T\_i, [\pi\_i]\_Q\_i = 1\) and \[\text{Ah\_i, j} = T\_i, Br\_j\] for each j \in [L] \setminus \{i\}.

- \((ct\_i\*, ct\_i)\) for x* is in the form:

\[
(ct\_i\*, ct\_i) = \left(\begin{array}{c}
    sA\_i + sAW + x\_i \sum_{i \in [L]} (sT\_i + sAW_i(M\_i \circ T\_i) \circ \gamma\_1) \\
    c\_i
    \end{array}\right)_{1, c\_i}
\]

42
where \( \tilde{f}_i^* = (f_i^* \| 1) \) and \( \bar{x}^* = (x^* \| 0_n) \).

- **G_1**: Identical to **G_0**, except that for all \( i \in [L] \) and \( s \in \{1, 2\} \), we replace \( [M_i]_s \) with

\[
[M_i]_s = \begin{bmatrix}
M & 0_{n,1} \\
0_{n \times n_2} & ic_{t,i,s}
\end{bmatrix}
\]

where \( [ic_{t,i,s}]_s \) is full-rank.

\[
\text{Enc}_1([ipk]_1, [ipk]_2, \bar{0}) \quad \text{if } i \in [L] \setminus (M \cup C)
\]

\[
\text{Enc}_1([ipk]_1, [ipk]_2, \bar{x}^* M(f_i^*)^T) \quad \text{if } i \in M \cup C
\]

In particular, we generate \( crs \) as

\[
crs = \begin{cases}
[A, AW]_1 \\
\{ crs_{i_1}, [R_i, AW_i (M_i \otimes I_{k+1})], AW_i \} \}_{i \in [L]}
\end{cases}
\]

\[
\{ [M_j, Br_j, W_j (M_j \otimes Br_j^T) + W_j] \}_{j \in [L]}
\]

\[
\{ [W_i (M_i \otimes Br_i^T)]_2 \}_{j \in [L], i \in [L] \setminus (j)}
\]

and generate challenge ciphertext as

\[
(\text{ct}_i^*, \text{ct}_i') = \begin{cases}
\begin{cases}
sA_{e_i} + sAW_{e_i} + \bar{x}^* & \sum_{i \in [L]} (sT_i + sAW_i (M_i (\tilde{f}_i^*)^T \otimes I_{k+1})) \\
\end{cases}
\end{cases}
\]

We have \( G_3 \approx_c G_0 \). This follows from the security of \( \Pi_1 \).

- **G_2**: Identical to **G_1** except that for all \( i \in [L] \) and all \( pk_i \in D_i \), we replace \( \pi_i \) with

\[
\bar{\pi}_i \leftarrow \left[ \sim \right]_{crs_{i_1}, td_{i_2}, [F_i]_1} \quad \text{where} \quad F_i = \begin{bmatrix} AU_i \\
R_i \end{bmatrix}
\]

We have \( G_2 \equiv G_1 \). This follows from the perfect zero-knowledge of \( \Pi_0 \).

- **G_3**: Identical to **G_2** except that we sample \( s \leftarrow \mathbb{Z}_p^{1 \times k} \) along with \( A \) and replace all \( R_i \) in \( crs \) with

\[
\bar{R}_i = \tilde{R}_i \begin{bmatrix} sA \\
I_{2k+1}
\end{bmatrix}, \quad \bar{R} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+2)}
\]

We have \( G_3 \equiv G_2 \). This follows from the fact that both \( R_i \) (in **G_2**) and \( \tilde{R}_i \) (in **G_3**) are truly random since matrix \( \begin{bmatrix} sA \\
I_{2k+1}
\end{bmatrix} \) is full-rank.

- **G_4**: Identical to **G_3** except that we generate the \( c_i^* \) as follows:

\[
c_i^* = \sum_{i \in [L]} (e_i R_i^{-1} Q_i + sAW_i (M_i (\tilde{f}_i^*)^T \otimes I_{k+1}))
\]

We have \( G_4 \approx_c G_3 \). This follows from stronger unbounded simulation soundness of \( \Pi_0 \) along with the fact that

\[
\text{LVer}(crs_{i_1}, [F_i^*], \pi_i^*) = 1 \quad \text{for all} \quad i \in [L] \quad \text{where} \quad F_i^* = \begin{bmatrix} T_i^* \\
Q_i^*
\end{bmatrix}.
\]

The details are identical to that in game **G_3** of our sReg-IPFE (c.f. Section 3).

- **G_5**: Identical to **G_4** except that we replace all \( sA \) with \( c \leftarrow \mathbb{Z}_p^{1 \times (2k+1)} \); in particular, we generate \( \bar{R}_i \) as follows:

\[
\bar{R}_i = \tilde{R}_i \begin{bmatrix} c \\
I_{2k+1}
\end{bmatrix}, \quad \bar{R} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+2)}
\]

and generate the challenge ciphertext as follows:

\[
(\text{ct}_i^*, \text{ct}_i') = \begin{cases}
\begin{cases}
\bar{F}_i + \bar{W}W_{e_i} + \bar{x}^* + sAW_{e_i} (M_i (\tilde{f}_i^*)^T \otimes I_{k+1})) \\
\end{cases}
\end{cases}
\]
We have $G_5 \approx_c G_4$. This follows from MDDH assumption which ensures that $([\mathbf{A}]_1, [s\mathbf{A}]_1) \approx_c ([\mathbf{A}]_1, [\mathbf{c}]_1)$ when $\mathbf{A} \leftarrow \mathcal{Z}_{p}^{k \times (2k+1)}$, $s \leftarrow \mathcal{Z}_{p}^{1 \times k}$, $\mathbf{c} \leftarrow \mathcal{Z}_{p}^{1 \times (2k+1)}$.

- $G_6$: Identical to $G_5$ except that
  - we generate $c^*_{+,1}$ as follows:
  $$c^*_{+,1} = cW + \overline{x^*}$$
  where $x^* = (0_n, ||sk)$, $sk \in \text{Gen}_1(1^4)$;
  - in crs, we change $[W_j(M_j \otimes B_{r_j}) + W\bar{M}_j]_2$ for all $j \in [L]$ as follows:
  $$[W_j(M_j \otimes B_{r_j}) + W\bar{M}_j + c^a_j]_2$$
  where $c^a \in \mathbb{Z}_{p}^{2k+1}$ such that $cc^a = 1$ and $Ac^a = 0$; and
  $$a_j = \begin{cases} 
  (-x^*M)[0] & \text{if } j \in [L] \setminus (M^* \cup C^*) \\
  (-x^*M)[x^*M(f'_j)] & \text{if } j \in M^* \cup C^*
  \end{cases}$$

We have $G_6 \approx_s G_5$. This follows from the change of variable $W \mapsto W + c^*(-x^*||sk)$. With above variable substitution, we have

$$cW + (x^*||0_n) \quad // c^*_{+,1} \text{ in } G_5$$

$$\approx_s cW + (-x^*||sk) + (x^*||0_n)$$

$$= cW + (0_n||sk) \quad // c^*_{+,1} \text{ in } G_6$$

For all $j \in M^* \cup C^*$, we have

$$[W\bar{M}_j]_2 \quad // \text{crs in } G_5$$

$$\approx_s \begin{bmatrix} W\bar{M}_j + c^*(-x^*||sk) [M \ 0_{n}] \end{bmatrix} _2$$

$$= \begin{bmatrix} W\bar{M}_j + c^*(x^*M[0_{n \times 1}]) \end{bmatrix} _2 \quad // \text{crs in } G_6$$

the third "=" follows from the fact that $[ict]_2 \in \text{Enc}_1([ipk]_1, [ipk]_2, x^*M(f'_j)]$ for $j \in M^* \cup C^*$, and the linear decryption of $\Pi_1$ (defined in Section 2.5). And for all $j \in [L] \setminus (M^* \cup C^*)$, we have

$$[W\bar{M}_j]_2 \quad // \text{crs in } G_5$$

$$\approx_s \begin{bmatrix} W\bar{M}_j + c^*(-x^*||sk) [M \ 0_{n}] \end{bmatrix} _2$$

$$= \begin{bmatrix} W\bar{M}_j + c^*(x^*M[0_{1 \times n}]) \end{bmatrix} _2 \quad // \text{crs in } G_6$$

the third "=" follows from the fact that $[ict]_2 \in \text{Enc}_1([ipk]_1, [ipk]_2, 0)$ for $j \in [L] \setminus (M^* \cup C^*)$, and the linear decryption of $\Pi_1$ (defined in Section 2.5).

- $G_{7,t}$ ($t \in [0, L]$): Identical to $G_6$ except that for all $j \in [\ell]$, we change $[W_j(M_j \otimes B_{r_j}) + WM_j + c^a_j]_2$ in crs as follows:
  $$[W_j(M_j \otimes B_{r_j}) + WM_j + \overline{c^a}_j]_2$$

We have that

- $G_{7,0} = G_6$: the two games are actually identical, since $[0] = 0$;
- $G_{7,t-1} \approx_c G_{7,t}$ for all $\ell \in [L]$, we will employ a sub-sequence of games for the proof described later.

Observe that in the final game $G_{7,L}$ can be simulated using the simulator by setting $\mu_i = x^*M(f'_i)'$, where we embed $x^*M(f'_i)'$ into crs so that hsk, for all $i \in M^* \cup C^*$ and remove $x^*$ from ct'.

44
From $G_{7,ℓ−1}$ to $G_{7,ℓ}$. We are ready to prove $G_{7,ℓ−1} \approx_c G_{7,ℓ}$ and this will complete the proof of Lemma 4. For this, we need the following sub-sequence of games for each $ℓ \in [L]$:

- $G_{7,ℓ−1,0}$: Identical to $G_{7,ℓ−1}$ where we recall $crs, pk_t \in D_t$ and $c_t^i$, with highlighting relevant terms in the following sub-sequence with dashed boxes as follows:

\[
\begin{align*}
\text{crs} = & \left[\{\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, 0)\} \text{ if } i \in [L] \setminus (M^* \cup C^*) \middle| \begin{array}{ll}
\{\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, x^*M(f_t)^{\top})\} & \text{ if } i \in M^* \cup C^*
\end{array} \right]
\end{align*}
\]

For all $j \in [L] \setminus [ℓ−1]$, recall that

\[
a_j = \begin{cases} (-x^*M||0) & \text{if } j \in [L] \setminus (M^* \cup C^*) \\ (-x^*M||x^*M(f_t)^{\top}) & \text{if } j \in M^* \cup C^* \end{cases}
\]

- $G_{7,ℓ−1,1}$: Identical to $G_{7,ℓ−1,0}$ except that we replace all $Br_t^r$ with $d_t^r \leftarrow Z_p^{k+1}$ in crs; in particular, we change the dashed boxed term in crs and pk$_t$ as follows:

\[
\begin{align*}
\left[\{\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, 0)\} \text{ if } i \in [L] \setminus (M^* \cup C^*) \middle| \\
\{\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, x^*M(f_t)^{\top})\} & \text{ if } i \in M^* \cup C^* \right]
\end{align*}
\]

We have $G_{7,ℓ−1,1} \approx_c G_{7,ℓ−1,0}$. This follows from MDDH assumption w.r.t. $[B]_2$ which ensures that $([B]_2, [Br_t^r]_2) \approx_c ([B]_2, [d_t^r]_2)$ when $B \leftarrow Z_p^{(k+1) \times k}$, $r_t \leftarrow Z_p^{1 \times (k+1)}$.

- $G_{7,ℓ−1,2}$: Identical to $G_{7,ℓ−1,1}$, except that we replace $\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, x^*M(f_t)^{\top})$ with $\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, x^*M(f_t)^{\top} + c^i a_t^i)$.

We have $G_{7,ℓ−1,2} \approx_c G_{7,ℓ−1,1}$. With defining $c^i \in Z_p^{k+1}$ and $d_t^r \in Z_p^{1 \times (k+1)}$ such that $c^i = 1, Ac^i = 0$ and $d_t^r = 1, d_t^r B = 0$. We consider two cases

- Honest case ($ℓ \in [L] \setminus (M^* \cup C^*)$): In this case, for all $s \in \{1, 2\}$, with $\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, 0)$, we have $a_t = (-x^*M||0)$, $M_t^i = \begin{bmatrix} M^0_{n1} \end{bmatrix}_{0 \times n2}$.
And we have \( \mathsf{pk}_f' = (\{T'_\ell, Q'_\ell\}, \{\{R'_\ell, h'_\ell\}\}) \in D_\ell \setminus C_\ell \) in this case. Namely, we know \( \mathsf{U'}_\ell \) (such that \( T'_\ell = \mathsf{A}'_\ell \mathsf{U'}_\ell \) and \( Q'_\ell = \overline{\mathsf{R}}_\ell \mathsf{U'}_\ell \)) and \( \mathsf{U'}_\ell \) is hidden from the adversary. We can write the dash boxed terms in \( \mathsf{c}'_i \) as follows:

\[
\mathsf{c}'_i = \mathsf{c}'_i(\overline{\mathsf{M}}_\ell(\overline{\mathsf{f}}'_\ell) \otimes I_{k+1})
\]

and replace \( \overline{\mathsf{R}}_\ell \) in \( \mathsf{crs} \) with a random \( \mathsf{R}_\ell \) as in \( \mathsf{G}_3 \). And we can prove \( \mathsf{G}_{7,\ell-1,2} \approx_c \mathsf{G}_{7,\ell-1,1} \) in this case using the following argument for all \( b \in \{0, 1\} \):

\[
\begin{align*}
\mathsf{A}, \mathsf{c}^+, [\mathsf{R}_\ell], 1, \mathsf{d}_\ell, \mathsf{AW}_\ell, [\mathsf{W}_\ell(\overline{\mathsf{M}}_\ell \otimes \mathsf{B}), \mathsf{W}_\ell(\overline{\mathsf{M}}_\ell \otimes \mathsf{d}_\ell^\top) + \overline{\mathsf{W}}_\ell + \mathsf{c}^+ \mathsf{a}_\ell]_2; & \quad // \mathsf{crs}, \mathsf{pk}_f' \\
& [\mathsf{c}, \mathsf{c}'_i + \mathsf{c}'_i(\overline{\mathsf{M}}_\ell(\overline{\mathsf{f}}'_\ell) \otimes I_{k+1})]_1, \mathsf{U}'_\ell, [\mathsf{R}_\ell \mathsf{U}'_\ell]_1, \mathsf{U}'_\ell \mathsf{B} & // \mathsf{ct}'^* , \mathsf{pk}_f'
\end{align*}
\]

\[
\begin{align*}
\approx_c \mathsf{A}, \mathsf{c}^+, [\mathsf{R}_\ell], 1, \mathsf{d}_\ell, \mathsf{AW}_\ell, [\mathsf{W}_\ell(\overline{\mathsf{M}}_\ell \otimes \mathsf{B}), \mathsf{W}_\ell(\overline{\mathsf{M}}_\ell \otimes \mathsf{d}_\ell^\top) + \overline{\mathsf{W}}_\ell + \mathsf{c}^+ \mathsf{a}_\ell]_2; & \quad // \mathsf{crs}, \mathsf{pk}_f' \\
& [\mathsf{c}, \mathsf{c}'_i + \mathsf{c}'_i(\overline{\mathsf{M}}_\ell(\overline{\mathsf{f}}'_\ell) \otimes I_{k+1})]_1, \mathsf{U}'_\ell, [\mathsf{R}_\ell \mathsf{U}'_\ell + \overline{\mathsf{u}} \mathsf{d}^\top]_1, \mathsf{U}'_\ell \mathsf{B} & // \mathsf{ct}'^* , \mathsf{pk}_f'
\end{align*}
\]

where \( \overline{\mathsf{u}} \leftarrow \mathbb{Z}_p^{n(2k+2)}, \mathsf{u}_\ell \leftarrow \mathbb{Z}_p^n \) and \( \mathsf{w}_\ell \leftarrow \mathbb{Z}_p^{n_1} \). We justify each step as below: The first \( \approx_c \) follows the argument:

\[
(A, \mathsf{c}, [\mathsf{R}_\ell], 1, \mathsf{B}, \mathsf{d}^\top, \mathsf{AU}_\ell, \mathsf{c}'_i [\mathsf{RU}_\ell]_1, \mathsf{U}_\ell \mathsf{B})
\]

\[
\approx_c (A, \mathsf{c}, [\mathsf{R}_\ell], 1, \mathsf{B}, \mathsf{d}^\top, \mathsf{AU}_\ell, \mathsf{c}'_i [\mathsf{RU}_\ell + \overline{\mathsf{u}} \mathsf{d}^\top]_1, \mathsf{U}_\ell \mathsf{B})
\]

which is analogous to the Lemma 2 in [ZZGQ23]. The second \( \approx_s \) uses the change of variables:

\[
\mathsf{U}_\ell \leftarrow \mathsf{U}_\ell + \mathsf{c}^+ \mathsf{u}_\ell \mathsf{d}^\top \quad \text{and} \quad \mathsf{W}_\ell \leftarrow \mathsf{W}_\ell + \mathsf{c}^+ ((\mathsf{w}_\ell \mathsf{M}[f'_\ell]) \otimes \mathsf{d}^\top)
\]

The last \( \approx_s \) is straightforward with the observation that \( \overline{\mathsf{u}} \) hides \( \mathsf{R}_\ell \mathsf{c}^+ \mathsf{u}_\ell \), this implies that \( \mathsf{u}_\ell \) hides \( \mathsf{w}_\ell \mathsf{M}[f'_\ell] \), and \( \mathsf{w}_\ell \mathsf{M}[0] \) is sufficient to hide \( \mathsf{a}_\ell = (-\mathsf{x}^\top \mathsf{M}[0]) \).

\[
\mathsf{a}_\ell = (-\mathsf{x}^\top \mathsf{M}[\mathsf{x}^\top (\mathsf{f}'_\ell)]) , \quad [\mathsf{M}]_s = \begin{bmatrix} \mathsf{M} & 0_{n_1} \\ 0_{n \times n_2} & i\mathsf{cF}'_\ell s \end{bmatrix} \tag{20}
\]

- Corrupted & Malicious Case (\( \ell \in (\mathcal{M}^* \cup \mathcal{C}^*) \)) And in this case, we have \( \mathsf{pk}_f' = (\{T'_\ell, Q'_\ell\}, \{\{h'_\ell, j'_\ell\}\})_\ell \in \{L\}(\ell), \mathsf{pk}_f' \) in \( \mathsf{G}_{7,\ell-1,2} \approx_c \mathsf{G}_{7,\ell-1,1} \) in this case using the following argument:

\[
\begin{align*}
\mathsf{A}, \mathsf{c}^+, \mathsf{B}, \mathsf{d}_\ell, \mathsf{AW}_\ell, [\mathsf{W}_\ell(\mathsf{M}_\ell \otimes \mathsf{B}), \mathsf{W}_\ell(\mathsf{M}_\ell \otimes \mathsf{d}_\ell^\top) + \mathsf{W}_\ell + \mathsf{c}^+ \mathsf{a}_\ell]_2; & \quad // \mathsf{crs} \\
& [\mathsf{c}, \mathsf{e}\overline{\mathsf{R}}'_\ell \mathsf{Q}'_\ell + \mathsf{c}'_i(\mathsf{M}_\ell(\mathsf{f}'_\ell) \otimes I_{2k+1})]_1 & // \mathsf{ct}'^* \text{ in } \mathsf{G}_{7,\ell-1,1}
\end{align*}
\]

\[
\begin{align*}
\approx_s \mathsf{A}, \mathsf{c}^+, \mathsf{B}, \mathsf{d}_\ell, \mathsf{AW}_\ell, [\mathsf{W}_\ell(\mathsf{M}_\ell \otimes \mathsf{B}), \mathsf{W}_\ell(\mathsf{M}_\ell \otimes \mathsf{d}_\ell^\top) + \mathsf{W}_\ell + \mathsf{c}^+ \mathsf{a}_\ell]_2; & \quad // \mathsf{crs} \\
& [\mathsf{c}, \mathsf{e}\overline{\mathsf{R}}'_\ell \mathsf{Q}'_\ell + \mathsf{c}'_i(\mathsf{M}_\ell(\mathsf{f}'_\ell) \otimes I_{2k+1}) + \mathsf{c}^+ (\mathsf{x}^\top \mathsf{M}[\mathsf{x}^\top (\mathsf{f}'_\ell)]) \mathsf{d}^\top]_1 & // \mathsf{ct}'^* \text{ in } \mathsf{G}_{7,\ell-1,2}
\end{align*}
\]
where isk ∈ Gen₁(1^4). We justify each step as follows:

The first follows from the change of variable:

\[ W_\ell' \mapsto W_\ell + c^\perp ((X^\perp - isk) \otimes d^\perp) \]

The second follows from the fact in equation (20), \( \tilde{f}_\ell = (f_\ell^r \| 1) \) and the linear decryption of \( \Pi_1 \) (defined in Section 2.5), which ensure that

\[ \left( (x^\perp - isk) \left( \begin{array}{cc} M & 0_{n_1} \\ 0_{n \times n_2} & ic_{\ell}^t \end{array} \right) (f_\ell^r \| 1)^\perp \right) = [0]_1 \]

\[ \left( (x^\perp - isk) \left( \begin{array}{cc} M & 0_{n_1} \\ 0_{n \times n_2} & ic_{\ell}^t \end{array} \right) + (-x^\perp M \| x^\perp M (f_\ell^r)^\perp) \right) = [0]_2 \]

- G₇,ℓ-1.3: Identical to G₇,ℓ-1.2 except that we replace all \( d^r_\ell \) with \( Br_\ell \) where \( r_\ell \leftarrow \mathbb{Z}_p^k \) in crs; in particular, we change the dashed boxed term in crs and \( pk_\ell \) as follows:

\[ [Br_\ell, W_i(\bar{M}_\ell \otimes Br_\ell^r)]_2, [W_i(\bar{M}_\ell \otimes Br_\ell^r)]_2, [U_i(\bar{Br}_\ell^r)]_2 \in \{ \ell \}\setminus \{t\} \]

We have \( G_{7,\ell-1} \approx_c G_{7,\ell-1.0} \). This follows from MDDH assumption w.r.t. \( [B]_2 \) which ensures that \( ([B]_2, [Br_\ell^r]_2) \approx_c ([B]_2, [d_\ell^r]_2) \) when \( B \leftarrow \mathbb{Z}_p^{n_1 \times (k+1)} \), \( r_\ell \leftarrow \mathbb{Z}_p^{1 \times (k+1)} \), \( d_\ell \leftarrow \mathbb{Z}_p^{1 \times (k+1)} \).  

### 6.4 Implication: Slotted Reg-IPFE with Very Selective SIM-Security

Setting the constraint \( M \) as \( \mathbf{I} \), we immediately have a multi-instance slotted Reg-IPFE which implies a Reg-IPFE with compact ciphertext. The scheme achieves very selective simulation-based security as our PReg-IPFE. And we delay the concrete scheme and proof in Appendix C.

### 6.5 A Variant: Slotted Reg-IPFE with Selective IND-security

The scheme is basically the same with our pre-constrained Reg-IPFE except that we set \( M_i \) as \( \mathbf{I} \) and remove the extra components for simulation-based security.

**Scheme.** Assuming QA-NIZK \( \Pi_0 = \langle \text{LGen}, \text{LPrv}, \text{LVer}, \text{LSim} \rangle \) for linear space over bilinear groups, see Section 2.4, our slotted Reg-IPFE scheme in prime-order bilinear groups works as follows:

- **Setup(1^4, 1^m, 1^{L_1}, \ldots, 1^{L_m}, 1^n) :** Run \( \emptyset := (p, G_1, G_2, G_\ell, e) \leftarrow \mathcal{G}(1^4) \). Sample \( A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}, W \leftarrow \mathbb{Z}_p^{(2k+1) \times n} \).

For each \( q \in [m] \), sample \( B_q \leftarrow \mathbb{Z}_p^{(k+1) \times k} \), and for all \( i \in [L_q] \), sample

\[ W_{q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)n}, B_{q,i} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+1)}, r_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k} \]

Run

\[ (\text{crs}_{q,i}, \text{td}_{q,i}) \leftarrow \text{LGen}(1^4, G_1, [A_q,i], 1) \]

where \( A_{q,i} = (A_{q,i}) \in \mathbb{Z}_p^{(3k+2) \times (2k+1)} \) for all \( q \in [m] \) and \( i \in [L_q] \). Output

\[ \text{crs} = \left[ \begin{array}{l}
[A, AW]_1,
\{ \text{crs}_{q,i}, [A_{q,i}, AW_{q,i}]_1 \}_{i \in [L_q]}
\{ [B_{q,i} f_{q,i}], [W_{q,i}(I_n \otimes B_{q,i} f_{q,i}) + W]_2 \}_{j \in [L_q]} \}
\{ [W_{q,i}(I_n \otimes B_{q,i} f_{q,i})]_{j \in [L_q], j \in [L_q] \setminus \{j\}} \}_{q \in [m]} \end{array} \right] \].

47
- Gen(crs, q, i): Sample $U_{q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)}$. Define $F_{q,i} = \frac{T_{q,i}}{Q_{q,i}} = \frac{AU_{q,i}}{U_{q,i}} \in \mathbb{Z}_p^{(3k+2) \times (k+1)}$ and run

$$\pi_{q,i} \leftarrow \text{LPrv}(\text{crs}_{q,i}, [F_{q,i}]_1, U_{q,i}).$$

Fetch $\{[B_{q}^{r_{q,i}}]_2\}_{j \in [L_q] \setminus \{i\}}$ from crs and output

$$pk_{q,i} = \left([AU_{q,i}, R_{q,i}U_{q,i}]_1, \{[U_{q,j}B_{q}^{r_{q,j}}]_2\}_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}\right) \quad \text{and} \quad sk_{q,i} = U_{q,i}.$$  

- Ver(crs, q, i, pk_{q,i}): Parse $pk_{q,i} = \{[T_{q,i}, Q_{q,i}]_1, \{[H_{i,j}]_2\}_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}\}$. Write $F_{q,i} = \frac{T_{q,i}}{Q_{q,i}}$ and check

$$\text{LVer}(\text{crs}_{q,i}, [F_{q,i}]_1, \pi_{q,i}) = 1.$$  

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

$$e([A]_1, [H_{i,j}]_2) = e([T_{q,i}]_1, [B_{q}^{r_{q,j}}]_2).$$  

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

- Agg_{s}(crs): Output:

$$\text{mpk}_{s} = ([A, AW]_1)$$

- Agg(crs, q, (pk_{q,i}, y_{q,i})_{i \in [L_q]}): If $q$ is an empty instance, on input $(pk_{q,i}, f_{q,i}) = (\bot, \bot)$ for all $i \in [L_q]$, abort and return $\text{mpk}_{q} = \bot$, $\text{hsk}_{q,j} = \bot$ for all $j \in [L_q]$. For all $i \in [L_q]$, parse $pk_{q,i} = \{[T_{q,i}, Q_{q,i}]_1, \{[H_{i,j}]_2\}_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}\}$. Output:

$$\text{mpk}_{q} = \left(\sum_{i \in [L_q]} (T_{q,i} + AW_{q,i}(y_{q,i}^{\top} \otimes I_{k+1}))\right)_1$$

and for all $j \in [L_q]$

$$\text{hsk}_{q,j} = \left([B_{q}^{r_{q,j}}]_2, \sum_{i \in [L_q] \setminus \{j\}} (H_{q,i,j} + W_{q,i}(I_{n} \otimes B_{q}^{r_{q,j}})y_{q,i}^{\top} + W_{q,i}(y_{q,i}^{\top} \otimes B_{q}^{r_{q,j}}) + Wy_{q,j}^{\top})\right)_2.$$  

- Enc_{s}(mpk_{s}, x): Sample $s \leftarrow \mathbb{Z}_p^{1 \times k}$. Output:

$$ct_{s} = ([sA, sAW + x]_1)$$

- Enc(mpk_{q}): Abort and return $\bot$ if $\text{mpk}_{q} = \bot$. Sample $s \leftarrow \mathbb{Z}_p^{1 \times k}$. Output:

$$ct_{q} = \left(\sum_{i \in [L_q]} (ST_{q,i} + sAW_{q,i}(y_{q,i}^{\top} \otimes I_{k+1}))\right)_1.$$  

- Dec(sk_{q', i'}, hsk_{q', i'}, (ct_{s}, ct_{q})): Abort and return $\bot$ if $ct_{q'} = \bot$. Parse

$$sk_{q', i'} = U_{q', i'}, \quad hsk_{q', i'} = ([K_{0}, K_{1}, K_{2}]_2), \quad (ct_{s}, ct_{q'}) = ([c_{s,0}, c_{s,1}, c_{q'}]_1).$$

Recover

$$[z_1]_T = e([c_{q'}]_1, [K_{0}]_2), \quad [z_2]_T = e([c_{s,0}]_1, [K_{1}]_2);$$

$$[z_3]_T = e([c_{s,0}U_{q', i'}]_1, [K_{0}]_2), \quad [z_4]_T = e([c_{s,0}]_1, [K_{2}]_2);$$

$$[z_5]_T = e([c_{s,1}]_1, [y_{q', i'}^{\top}]_2).$$

Compute

$$[z]_T = [z_1 - z_2 - z_3 - z_4 + z_5]_T.$$  

Recover $z$ from $[z]_T$ via brute-force DLOG and output $z$. 

48
Completeness. For all $\lambda, m, n \in \mathbb{N}$, all $L_1, \ldots, L_m \in \mathbb{N}$, all $q \in [m]$ and $i \in [L_q]$, all $\text{crs} \leftarrow \text{Setup}(1^\lambda, 1^m, 1^n, 1^{L_1}, \ldots, 1^{L_m})$, and $(pk_q,i, sk_q,i) \leftarrow \text{Gen}(\text{crs}, q, i)$, we have

$$pk_q,i = \{(T_{q,i}, Q_{q,i}), \{[h_{q,i,j}]_2\}_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}\}$$

$$= \{[AU_{q,i}, R_{q,i}U_{q,i}], \{[U_{q,i}B_{q,i}F_{q,i}]_2\}_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}\}$$

for some $U_{q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)}$ and $\pi_{q,i} \leftarrow \text{LPrv}(\text{crs}_{q,i}, [A_{q,i}]_1, U_i)$ where $(\text{crs}_{q,i}, t_{q,i}) \leftarrow \text{LGen}(1^\lambda, G_1, [A_{q,i}]_1)$ and $A_{q,i} = (R_{q,i})$ with $A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}$, $R_{q,i} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+1)}$. Then

- Write $F_{q,i} = (T_{q,i}) = (\mathbb{A}_{q,i})$, we have $L\text{Ver}(\text{crs}_{q,i}, [F_{q,i}]_1, \pi_{q,i}) = 1$ by the perfect completeness of $\Pi_0$ (see Section 2.4) and the fact that $F_{q,i} = A_{q,i}$.

- For each $j \in [L_q] \setminus \{i\}$, we have $e([A_{q,i}], [U_{q,i}B_{q,i}F_{q,j}]_2) = e([AU_{q,i}], [B_{q,i}F_{q,j}]_2)$ by the definition of bilinear map $e$ (see Section 2.1) and the fact that $A \cdot U_{q,i}B_{q,i}F_{q,j} = AU_{q,i} \cdot B_{q,i}F_{q,j}$.

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

Correctness. For all $\lambda, m, n \in \mathbb{N}$, all $q^* \in [m]$ and $i^* \in [L_{q^*}]$; all $\text{crs} \leftarrow \text{Setup}(1^\lambda, 1^m, 1^n, 1^{L_1}, \ldots, 1^{L_m})$, all $(pk_{q^*,i^*}, sk_{q^*,i^*}) \leftarrow \text{Gen}(\text{crs}, q^*, i^*)$; all $(pk_{q,i})_{i \in [L_{q^*}] \setminus \{i^*\}}$ such that $\text{Ver}(\text{crs}, q^*, i, pk_{q,i}) = 1$; all $x \in \mathbb{Z}_p^{1 \times n}$ and $y_{q,i} \in \mathbb{Z}_p^{1 \times n}$, for all $s \leftarrow \mathbb{Z}_p^{1 \times k}$, we have

$$sk_{q^*,i^*} = U_{q^*,i^*},$$

$$(ct_{s,0}, ct_{s^*}) = \left(\left\lfloor \sum_{i \in [L_q]} (sT_{q,i} + s\text{AW}_{q,i}(\tilde{y}_{q,i}^* \otimes I_{k+1})) \right\rfloor_{c_0}, \sum_{i \in [L_q]} (sT_{q,i} + s\text{AW}_{q,i}(\tilde{y}_{q,i}^* \otimes I_{k+1})) \right\rfloor_{c_1}\right)_{c_{q^*}},$$

$$hsk_{q^*,i^*} = \left(\sum_{i \in [L_{q^*}] \setminus \{i^*\}} \left(\sum_{k_1} \left(\frac{B_{q,i}F_{q^*,i^*}}{k_1} \cdot \frac{\text{AW}_{q^*,i^*}(\tilde{y}_{q,i^*}^* \otimes B_{q,i}F_{q^*,i^*})}{k_1} + \frac{\text{AW}_{q^*,i^*}(\tilde{y}_{q,i^*}^* \otimes B_{q,i}F_{q^*,i^*})}{k_1}\right)_2 \right)_{k_1} \cdot \frac{\text{AW}_{q^*,i^*}(\tilde{y}_{q,i^*}^* \otimes B_{q,i}F_{q^*,i^*})}{k_1} \right)_2 \right)_{k_2}.$$
and then

\[ Z = Z_1 - Z_2 - Z_3 + Z_4 + Z_5 \]

\[ = sT_{q,i} B_q \tilde{r}_{q,i} + sAW_{q,i} (\tilde{y}_{q,i} \otimes B_q \tilde{r}_{q,i}) - sAU_{q,i} B_q \tilde{r}_{q,i} \]

\[ - (sAW_{q,i} (\tilde{y}_{q,i} \otimes B_q \tilde{r}_{q,i}) + sAWy_{q,i}) \]

\[ + (sAWy_{q,i} + xy_{q,i}) \]  \hspace{1cm} (22)

\[ = xy_{q,i}. \]  \hspace{1cm} (23)

Here, equality (21) follows from the property of tensor product: \((a^\top \otimes 1)M = a^\top \otimes M\) for matrices of proper size; equality (22) follows from the fact that \(Ah_{q,i} = T_i B_q \tilde{r}_{q,i}\) for all \(i \in [L_q] \setminus \{i^*\}\); equality (23) follows from the fact that \(T_{q,i} = AU_{q,i}.\) This proves the correctness.

**Compactness and Efficiency.** Our multi-instance Reg-IPFE has the following properties:

- \(|\text{crs}| = O(L^2 \cdot n) \cdot \text{poly}(\lambda), \quad |\text{hsk}_{q,j}| = \text{poly}(\lambda),\)
- \(|\text{mpk}_i| = O(n) \cdot \text{poly}(\lambda), \quad |\text{mpk}_q| = \text{poly}(\lambda),\)
- \(|\text{ct}_i| = O(n) + \text{poly}(\lambda), \quad |\text{ct}_q| = \text{poly}(\lambda),\)

where \(L = L_1 + \cdots + L_m, \text{mpk} = (\text{mpk}_q, (\text{mpk}_q)_{q \in [m]}).\) Note that the total size of \(\{\text{crs}_i\}_{i \in [L]}\) is \(L \cdot \text{poly}(\lambda)\) 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 \(\text{poly}(\lambda).\)

**Security.** We have the following theorem. Given pairing-based QA-NIZK in [KW15] with unbounded simulation soundness under MDDH assumption and the fact that MDDH assumption implies subspace assumption [CGKW18], our slotted Reg-IPFE scheme achieves selective security from MDDH assumption.

**Theorem 5.** Assume \(\Pi_0 = (\text{LGen}, \text{LPrv}, \text{LVer}, \text{LSim})\) is a QA-NIZK with perfect completeness, perfect zero-knowledge and unbounded simulation soundness for linear space defined in Section 2.4, our slotted Reg-IPFE scheme achieves the selective IND-security defined in Section 2.3 under MDDH assumption and subspace decision assumption.

**Proof** We prove the following technical lemma this immediately proves Theorem 5.

**Lemma 5.** For all adversaries \(\mathcal{A},\) there exist adversaries \(\mathcal{B}_1, \mathcal{B}_2\) such that:

\[ \text{Adv}_{\mathcal{A}}^{\text{miReg-IPFE}}(\lambda) \leq L \cdot \text{Adv}_{\mathcal{B}_1}^{\text{USS}}(\lambda) + (2L + 2L \cdot Q + 1) \text{Adv}_{\mathcal{B}_2}^{\text{MDDH}}(\lambda) + \text{negl}(\lambda) \]

where \(L = L_1 + \cdots + L_m\) is the number of slots, \(Q\) is the maximum number of queries on a slot made by \(\mathcal{A}\) and \(\text{Time}(\mathcal{B}_1), \text{Time}(\mathcal{B}_2) \approx \text{Time}(\mathcal{A}).\)

For simplicity, we prove Lemma 4 in the case of nonempty 1-instance and remove the index \(q\) in the following proof. For an empty instance, we only need to remove the terms about \(\text{ct}_i^*\) and all \(\text{pk}_i^*\) in the following game sequence and notice there only exists “honest case” for empty instance. In the case of \(m\)-instance, it only needs to add back index \(q\) and apply sub-sequence \(G_{6,i-1,0}, \ldots, G_{6,i-1,3}\) to each instance.
Game Sequence. Suppose that crs is the common reference string, \( x_b \) is the challenge, \( \{pk_i, y_i^*\}_{i \in [L]} \) are challenge public keys along with challenge functions to be registered. For all \( i \in [L] \), define \( D_i = \{pk_i : D_i, [pk_i] = sk_i \neq \perp \} \) be responses to \( \text{OGen}(i) \) and \( C_i = \{pk_i : (i, pk_i) \in C_i \} \) records public keys in \( D_i \) that have been sent to \( \text{OCor}(i, \cdot) \). Recall that, for each \( i \in [L] \), we require that

\[
\text{pk}_i \notin D_i \implies \text{Ver}(\text{crs}, i, \text{pk}_i) = 1, \quad \text{pk}_i \in C_i \lor \text{pk}_i \notin D_i \implies x_i^0(y_i^*)^r = x_i^r(y_i^*)^r.
\]

Note that \( \text{pk}_i \) serves as a general entry in \( D_i \) while \( \text{pk}_i^* \) is the specific challenge public for slot \( i \); there can be more than one assignment for \( \text{pk}_i \) since the adversary can invoke \( \text{OGen}(i) \) for many times. We prove the Lemma 4 via dual-system method using the following game sequence.

- \( G_0 \): This is the real game, recall that we have
  
  \- \( \text{crs} \) is in the form:
  \[
  \text{crs} = \{[A, AW]_1, \{crs_i, [R_i, AW]_1\}_{i \in [L]} \} \cdot \{[B_{ij}, W_i(I_1 \odot B_{ij}) + W]_2\}_{j \in [L]}_{i \in [L] \setminus \{j\}}
  \]
  
  where \( \text{crs}_i \in LGen(1^\lambda, \emptyset_1, [A_i]_1) \), \( A_i = (A_{ri}) \).
  
  \- For each \( i \in [L] \), each \( (\text{pk}_i, \text{sk}_i) \in D_i \) is in the form
    \[
    \text{pk}_i = \{[A_{U_i}, R_i U_i]_1, \{[U_i B_{ij}]_2\}_{j \in [L] \setminus \{i\}}, \pi_i \} \quad \text{and} \quad \text{sk}_i = U_i
    \]
    
  where \( \pi_i \leftarrow LPrv(\text{crs}_i, [F_i]_1, U_i), F_i = (A_{U_i R_i}) \).
  
  \- For all \( i \in [L] \), \( \text{pk}_i^* \) is in the form
    \[
    \text{pk}_i^* = \{[T_i^r, Q_i^r]_1, \{[H_{ij}^r]_2\}_{j \in [L] \setminus \{i\}}, \pi_i^* \}
    \]
    
  such that \( \text{Ver}(\text{crs}, i, \text{pk}_i^*) = 1 \) which means \( \text{LVer}
    \begin{bmatrix}
    T_i^r \\
    Q_i^r
    \end{bmatrix}
    _{\{1, 2\}}^{\text{crs}_i}, \pi_i^* \) = 1 and \( A_{h_{ij}^r} = T_i^r B_{ij} \) for each \( j \in [L] \setminus \{i\} \).
  
  \- \( (ct_i^*, ct_i^+ \cdot c_i) \) for \( x_b^r \) is in the form:
    \[
    (ct_i^*, ct_i^+) = \begin{bmatrix}
    sA \cdot sAW + x_b^r_c, \sum_{i \in [L]} (sT_i + sAW_i ((y_i^*)^r \odot I_{k+1}))_{\{1, 2\}}^{\text{crs}_i}, \pi_i^* \\
    c_i
    \end{bmatrix}
    \]
    
- \( G_1 \): Identical to \( G_0 \) except that for all \( i \in [L] \) and all \( (pk_i, sk_i) \in D_i \), we replace \( \pi_i \) with
  \[
  \pi_i \leftarrow \text{LSim}[\text{crs}_i, \text{td}_i, [F_i]_1] \quad \text{where} \quad F_i = \begin{bmatrix} A_{U_i} \\ R_i U_i \end{bmatrix}.
  \]
  
  We have \( G_1 \equiv G_0 \). This follows from the perfect zero-knowledge of \( \Pi_0 \).

- \( G_2 \): Identical to \( G_1 \) except that we sample \( s \leftarrow Z_p^{1 \times k} \) along with \( A \) and replace all \( R_i \) in \( \text{crs} \) with
  \[
  \text{R}_i = \text{R}_i \begin{bmatrix} sA \\ I_{2k+1} \end{bmatrix}, \quad \text{R}_i \leftarrow Z_p^{(2k+2) \times (2k+2)}.
  \]
  
  We have \( G_2 \equiv G_1 \). This follows from the fact that both \( R_i \) (in \( G_2 \)) and \( \text{R}_i \) (in \( G_3 \)) are truly random since matrix \( \begin{bmatrix} sA \\ I_{2k+1} \end{bmatrix} \) is full-rank.
We have $G_3 \approx_c G_2$. This follows from stronger unbounded simulation soundness of II along with the fact that 
$LVer(\text{crs}_{i,1}, [F_i^*], \pi^*_i) = 1$ for all $i \in [L]$ where $F_i^* = \begin{pmatrix} T_i^* \\ Q_i^* \end{pmatrix}$. Assume $pk_i^* \not\in D_i^*$, i.e., $pk_i^*$ is malicious. In the reduction, we guess $i^* \leftarrow [L]$ and obtain $A, \hat{R}, crs$, as input; we simulate honestly as in $G_3$ except that for all $pk_i \in D_i^*$, we make an oracle query $[F_i^*]$, and get $\pi_i$ in it; we finally output $([F_i^*], \pi_i)$ in $pk_i^* \not\in D_i^*$. Observe that once it happens that $e_i \hat{R}_i^{-1} Q_i^* \neq sT_i^*$, we must have $F_i^* \not\in \text{span}(A_i)$. When $pk_i^* \in D_i^*$, we always have $G_4 \equiv G_3$.

- $G_4$: Identical to $G_3$ except that we replace all $sA$ with $c \leftarrow \mathcal{Z}_p^1 \times (k+1)$; in particular, we generate $\hat{R}$ as follows:

$$
\hat{R}_i = \begin{pmatrix} e_i \\ I_{2k+1} \end{pmatrix}, \quad \hat{R}_i \leftarrow \mathcal{Z}_p^{(k+2) \times (k+2)}
$$

and generate the challenge ciphertext as follows:

$$
ct^* = \left( \left( \left[ e_i \right] W + x^*_{b_i} \sum_{i \in [L]} (e_i \hat{R}_i^{-1} Q_i^* + cW_i((y_i^*)^T \otimes I_{k+1})) \right) \right)_{1}\left( e_i \right)
$$

We have $G_4 \approx_c G_3$. This follows from MDDH assumption which ensures that $([A]_1, [sA]_1) \approx ([A]_1, [c]_1)$ when $A \leftarrow \mathcal{Z}_p^{k \times (2k+1)}, s \leftarrow \mathcal{Z}_p^{1 \times k}, c \leftarrow \mathcal{Z}_p^{1 \times (2k+1)}$.

- $G_5$: Identical to $G_4$ except that
  - we generate $c^*_{+,1}$ as follows:
    $$
c^*_{+,1} = [cW + x^*_{b_i}]_1
    $$
  - in $\text{crs}$, we make the following change for all $j \in [L]$:  
    $$
    [W_j(I_n \otimes Br_j^*) + W + [c^j(x^*_0 - x^*_j)]_2
    $$

where $c^j \in \mathcal{Z}_p^{2k+1}$ such that $cc^j = 1, Ac^j = 0$.

We have $G_5 \approx_s G_4$ which follows from the fact that we can utilize the change of variable $W \mapsto W + c^j(x^*_0 - x^*_j)$.

- $G_{6,\ell}(\ell \in [0, L])$: Identical to $G_5$ except that for all $j \in [\ell]$, we change $[W_j(I_n \otimes Br_j^*) + [c^j(x^*_0 - x^*_j)]_2$ in $\text{crs}$ as follows:

$$
[W_j(I_n \otimes Br_j^*) + W + c^j(x^*_0 - x^*_j)]_2
$$

We have that
- $G_{6,0} = G_5$; the two games are actually identical, since $[0] = 0$;
- $G_{6,\ell-1} \approx_c G_{6,\ell}$ for all $\ell \in [L]$, we will employ a sub-sequence of games for the proof described later.

**From $G_{6,\ell-1}$ to $G_{6,\ell}$.** We are ready to prove $G_{6,\ell-1} \approx_c G_{6,\ell}$. We this will complete the proof of Lemma 5. For this, we need the following sub-sequence of games for each $\ell \in [L]$:
- \(G_{6,t-1.0}\): Identical to \(G_{6,t-1}\) where we recall \(\text{crs}, pk_t \in D_t\) and \(c^*_t\), with highlighting relevant terms in the following sub-sequence with dashed boxes as follows:

\[
\text{crs} = \begin{cases} 
[A, AW]_1, \{c_{rs_t}, [\tilde{R}_t, AW]_1\}_{t \in [L]} \\
\{[[B_{j'}, W_j(I_n \otimes B_{j'})] + W]_2\}_{j \in [t-1]} \\
\{[[B_{j'}, W_j(I_n \otimes B_{j'}) + W + c^+(x_0^t - x_b^t)]_2\}_{j \in [L]\{t\}} \\
\{[[W_j(I_n \otimes B_{j'})]_2\}_{j \in [L]\{t\}} \\
\end{cases}
\]

\[
pk_t = \begin{cases} 
\frac{T_t}{\mathcal{Q}_t} - \frac{h_{ij}}{\mathcal{R}_t} \\
\frac{h_{ij}}{\mathcal{R}_t} \\
h_{ij} \\
\end{cases}
\]

\[
c^*_t = e_t \tilde{R}^{-1} Q_t + cW_t((y^t_\ell) \otimes I_{k+1}) + \sum_{i \in [L]\{t\}} e_i \tilde{R}_i^{-1} Q_i + cW_i((y_i^t) \otimes I_{k+1})
\]

where \(c^+ \in \mathbb{Z}^{2k+1}_p\) such that \(cc^+ = 1, Ac^+ = 0\).

- \(G_{6,t-1.1}\): Identical to \(G_{6,t-1.0}\) except that we replace all \(B_{j'}\) with \(d^t_\ell \leftarrow \mathbb{Z}^{2k+1}_p\) in \(\text{crs}\); in particular, we change the dashed boxed term in \(\text{crs}\) and \(pk_t\) as follows:

\[
[[d^t_\ell] W_t(I_n \otimes d^t_\ell) + W + c^+(x_0^t - x_b^t)]_2, \{[[W_j(I_n \otimes d^t_\ell)]_2\}_{j \in [L]\{t\}} \}
\]

We have \(G_{6,t-1.1} \simeq_c G_{6,t-1.0}\). This follows from MDDH assumption w.r.t. \([B]_2\) which ensures that \(([[B]_2, [B_{j'}]_2] \simeq_c ([B]_2, [d^t_\ell]_2)\) when \(B \leftarrow \mathbb{Z}_p^{(k+1)\times k}, r^t_\ell \leftarrow \mathbb{Z}_p^{1\times (k+1)}\).

- \(G_{6,t-1.2}\): Identical to \(G_{6,t-1.1}\), except that we make the following change of \(\text{crs}\)

\[
[[W_t(I_n \otimes d^t_\ell) + W + c^+(x_0^t - x_b^t)]_2
\]

We have \(G_{6,t-1.2} \simeq_c G_{6,t-1.1}\). With defining \(c^+ \in \mathbb{Z}^{2k+1}_p\) such that \(cc^+ = 1, Ac^+ = 0\). We consider two cases

- Honest case: In this case, we have \(pk^*_t = ([[T^t_\ell, Q^t_\ell]_1, \{[[h^t_\ell]_2]\}_{j \in [L]\{t\}}, \pi^t_\ell] \in D_t \setminus C_t\). Namely, we know \(U^t_\ell\) (such that \(T^t_\ell = AU^t_\ell\) and \(Q^t_\ell = \tilde{R}_tU^t_\ell\)) and \(U^t_\ell\) is hidden from the adversary. We can write the dash boxed terms in \(c^*_t\) as follows:

\[
\frac{\hat{cU}^t_\ell}{\mathcal{Q}_t} + cW_t((y^t_\ell) \otimes I_{k+1})
\]

53
and replace $R_i$ in crs with a random $R_f$ as in $G_2$. And we can prove $G_{6,\ell-1,2} \approx_c G_{6,\ell-1,1}$ in this case using the following argument for all $b' \in \{0, 1\}$:

$$\begin{align*}
A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_0 \otimes d_\ell)] + W + b'c^\perp(x_b^0 - x_b^\perp)]_2; & \quad \text{crs, pk}_\ell \\
[c, cU_\ell^* + cW_\ell((y_\ell')^\top \otimes I_{k+1})], AU_\ell^*, [R_\ell U_\ell^*], U_\ell^*; & \quad \text{ct}^*, \text{ pk}_\ell^*
\end{align*}$$

\[\approx_c \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)] + W + b'c^\perp(x_b^0 - x_b^\perp)]_2; \]

\[\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)] + W; \]

\[\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)] + W; \]

\[\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)]; \]

$$\begin{align*}
\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)]; & \quad \text{crs}
\end{align*}$$

\[\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)], U_\ell^*; \]

$$\begin{align*}
& \approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)], U_\ell^*; \quad \text{ct}^* \text{ in } G_{6,\ell-1,1}
\end{align*}$$

We justific each step as follows: The first $\approx_c$ follows the argument:

$$\begin{align*}
(A, c, [R_\ell], 1, B, d_\ell, U_\ell^*, cU_\ell^*[RU_\ell^*]_1, U_\ell^* B)
\approx_c (A, c, [R_\ell], 1, B, d_\ell, U_\ell^*, cU_\ell^*[RU_\ell^*]_1, U_\ell^* B)
\end{align*}$$

which is analogous to the Lemma 2 in [ZZGQ23]. The second $\approx_s$ uses the change of variables:

$$\begin{align*}
U_\ell^* \mapsto u_\ell^* + c^\perp u_\ell d_\ell^\top \quad \text{and} \quad W_\ell \mapsto W_\ell - b'c^\perp((x_b^0 - x_b^\perp) \otimes d_\ell^\top)
\end{align*}$$

The last $\approx_s$ is straightforward with the observation that $\bar{u}$ hides $R_f c^\perp u_\ell$, this implies that $u_\ell$ hides $b'(x_b^0 - x_b^\perp)(y_\ell')^\top$.

**Corrupted & Malicious Case:** In this case, we have $\text{pk}_\ell^* = ([T_\ell, Q_\ell]_1, \{[h_i^*]_2\}_{i \in \Lambda_{\ell}})$, $\pi_\ell^* \in C_\ell \cup \bar{D}_\ell$. We prove $G_{6,\ell-1,2} \approx_c G_{6,\ell-1,1}$ in this case using the following argument:

$$\begin{align*}
A, c^\perp, B, I_\ell, AW_\ell, I_\ell \otimes d_\ell, W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)] + W + c^\perp(x_b^0 - x_b^\perp)]_2; \quad & \text{crs}
\end{align*}$$

\[\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)] + W + c^\perp(x_b^0 - x_b^\perp)]_2; \quad \text{crs}
\]

\[\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)] + W; \quad \text{ct}^* \text{ in } G_{6,\ell-1,2}
\]

$$\begin{align*}
\approx_s \quad A, c^\perp, B, [R_\ell], 1, d_\ell, AW_\ell, [W_\ell(I_0 \otimes B), W_\ell(I_\ell \otimes d_\ell)] + W
\end{align*}$$

We justific each step as follows: The first $\approx_s$ uses the change of variable:

$$\begin{align*}
W_\ell \mapsto W_\ell - c^\perp((x_b^0 - x_b^\perp) \otimes d_\ell^\top)
\end{align*}$$

The second $\approx_s$ uses the fact that $(x_b^0 - x_b^\perp)(y_\ell') = 0$ in this case.

- $G_{6,\ell-1,3}$ is identical to $G_{6,\ell-1,2}$ except that we replace all $d_\ell$ with $B r_\ell$ where $r_\ell \sim \mathbb{Z}_p^k$ in crs; in particular, we change the dashed boxed term in crs and $\text{pk}_\ell$ as follows:

$$\begin{align*}
\left[\begin{array}{c}
\text{br}_\ell^\perp \\
W(I_0 \otimes d_\ell) + W
\end{array}\right], \left[\begin{array}{c}
W(I_0 \otimes [B r_\ell]) \\
W(I_0 \otimes d_\ell)
\end{array}\right], \left[\begin{array}{c}
U_\ell^* [B r_\ell]^2 \\
U_\ell^* [B r_\ell]^2
\end{array}\right]_{e \in \Lambda_{\ell} \setminus \{\ell\}}
\end{align*}$$

We have $G_{6,\ell-1,3} \approx_c G_{6,\ell-1,2}$. This follows from MDDH assumption w.r.t. $[B]_2$ which ensures that $([B]_2, [B r_\ell]_2) \approx_c (B, [B]_2, [d_\ell]_2)$ when $B \sim \mathbb{Z}_p^{(k+1)\times k}$, $r_\ell \sim \mathbb{Z}_p^{1\times 1}$.
7 Registered Quadratic Functional Encryption

In this section, we present our Reg-QFE scheme for the quadratic functionality which is defined by $X = \mathbb{Z}_p^{1 \times n_1} \times \mathbb{Z}_p^{1 \times n_2}$, $Z = \mathbb{Z}_p$ and

$$QF_{n_1 n_2} = \{ f : (x_1, x_2) \mapsto (x_1 \otimes x_2) f \},$$

where $f \in \mathbb{Z}_p^{1 \times n_1 n_2}$. We first present the multi-instance slotted Reg-QFE by working on our multi-instance slotted PReg-IPFE scheme in Section 6; with the multi-instance Reg-QFE, we finally lead to the compact Reg-QFE which achieve the very selective SIM-security defined in Section 2.2.

7.1 Multi-instance slotted Reg-QFE

With the multi-instance slotted PReg-IPFE $\Pi_2 = (\text{iSetup}, \text{iGen}, \text{iVer}, \text{iAgg}_+, \text{iAgg}, \text{iEnc}_+, \text{iEnc}, \text{iDec})$ in Section 6, over prime-order bilinear group $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$; our multi-instance slotted Reg-QFE works as follows in the bilinear group $\mathbb{G}$:

- Setup($1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, 1^{n_1}, 1^{n_2}$): Sample $A_1 \leftarrow \mathbb{Z}_p^{k \times n_1}$, $A_2 \leftarrow \mathbb{Z}_p^{k \times n_2}$. Set $n'_j = k(n_1 + n_2 + k)$, $n'_2 = n_1n_2$, run

$$\text{icrs} \leftarrow \text{iSetup}(1^\lambda, 1^m, 1^{L_1}, \ldots, 1^{L_m}, 1^{n_1}, 1^{n_2}, \text{M})$$

where $\text{M} = \begin{pmatrix} A_1 \otimes \text{I}_{n_2} \\ \text{I}_{n_1} \otimes A_2 \\ A_1 \otimes A_2 \end{pmatrix}$.

Output

$$\text{crs} = ([A_1], [A_2], \text{icrs})$$

- Gen($\text{crs}, q, i$): Sample (impk$_q$, isk$_{qi}$) $\leftarrow \text{iGen}(\text{crs}, q, i)$, output

$$(\text{pk}_q, \text{sk}_q) = (\text{impk}_q, \text{isk}_q)$$

- Ver($\text{crs}, q, i, \text{pk}_q$): Parse $\text{pk}_q = \text{impk}_q$, output

$$\text{iVer}(\text{crs}, q, i, \text{pk}_q).$$

- $\text{Agg}_+(\text{crs})$: Sample impk$_+$ $\leftarrow \text{iAgg}_+(\text{crs})$, output

$$\text{mpk}_+ = ([A_1], [A_2], \text{impk}_+).$$

- $\text{Agg}(\text{crs}, q, (\text{pk}_q, f_{qi})_{i \in [L_q]}$): If $q$ is an empty instance, namely $(\text{pk}_q, f_{qi}) = (\perp, \perp)$ for all $i \in [L_q]$, abort and return $\text{mpk}_q = \perp$, $\text{hsk}_q, j = \perp$ for all $j \in [L_q]$. Sample $(\text{impk}_q, \text{isk}_q)_{i \in [L_q]}$ $\leftarrow \text{iAgg}(\text{crs}, q, (\text{impk}_q, f_{qi})_{i \in [L_q]}),$ output

$$\text{mpk}_q = \text{impk}_q,$$

and for all $j \in [L_q]$

$$\text{hsk}_q, j = \text{isk}_q, j.$$

- $\text{Enc}_+(\text{mpk}_+, (x_1, x_2))$: Sample $s_1, s_2 \leftarrow \mathbb{Z}_p^{1 \times k}$. Run

$$\text{ict}_+ \leftarrow \text{iEnc}_+(\text{mpk}_+, x),$$

where $x = (s_1 \otimes x_2 || x_1 \otimes s_2 || s_1 \otimes s_2)$. Output

$$\text{ct}_+ = ([s_1A_1 + x_1]_1, [s_2A_2 + x_2]_2, \text{ict}_+).$$
- Enc(mpq): Abort and return ctq = ⊥ if mpq = ⊥. Run

\[ \text{ict}_q \leftarrow \text{iEnc}(\text{impq}), \]

**Output**

\[ \text{ct}_q = \text{ict}_q. \]

- Dec(skkq, hskq, (ctq, ctq′)): Abort and return ⊥ if ictpq = ⊥. Parse

\[ \text{sk}_{q′,i'} = \text{isk}_{q′,i'}, \quad \text{hsk}_{q′,i'} = \text{ihsk}_{q′,i'}, \quad (\text{ct}_q, \text{ct}_{q'}) = ([y_1], [y_2], \text{ict}_q, \text{ict}_{q'}). \]

Compute

\[ [z]_T = [(y_1 \otimes y_2) \mathbf{f}_{q′,i'} - \text{iDec} (\text{isk}_{q′,i'}, \text{ihsk}_{q′,i'}, (\text{ict}_q, \text{ict}_{q'}))]_T \]

Recover z from [z]_T via brute-force DLOG and output z.

**Completeness.** For all λ, m, n1, n2 ∈ N, all L1, ..., Lm ∈ N, all q ∈ [m] and i ∈ [Lq]; all crs ← Setup(1, m, 1L1, ..., 1Lm, 1ni, 1n2), and (pkq, skq) ← Gen(crs, q, i), we have

\[ \text{crs} = ([A_1], [A_2], \text{icrs}) \quad \text{and} \quad (\text{pk}_q, \text{sk}_q) = (\text{ipk}_q, \text{isk}_q) \]

where (ipkq, iskq) ← iGen(ics, q, i) and crs ← Setup(1, m, 1L1, ..., 1Lm, 1ni, 1n2, M), with n′ = k(n1 + n2 + k), n′ = n1n2k. With the completeness of Π2 (c.f. Section 6.1), we have iVer(crs, q, i, ipkq) = 1. This ensures that Ver(crs, q, i, pkq) = 1 by the specification of Ver and readily proves the completeness.

**Correctness.** For all λ, m, n1, n2 ∈ N, all L1, ..., Lm ∈ N, all q ∈ [m] and i ∈ [Lq]; all crs ← Setup(1, m, 1L1, ..., 1Lm, 1ni, 1n2), all (pkq, skq, i) ← Gen(crs, q, i); all (s1, s2) ∈ Zp×m and fq′,i ∈ Zp×ni, we have:

\[ (\text{pk}_{q',i}, \text{sk}_{q',i}) = (\text{ipk}_{q',i}, \text{isk}_{q',i}) \]

\[ \text{hsk}_{q',i} = \text{ihsk}_{q',i} \]

\[ (\text{ct}_q, \text{ct}_{q'}) = ([s_1A_1 + s_2], [s_2A_2 + s_2], \text{ict}_q, \text{ict}_{q'}) \]

where

\[ \text{ict}_q \leftarrow \text{iEnc}_q (\text{impq}, x; s) \]

\[ \text{ict}_{q'} \leftarrow \text{iEnc}_q (\text{imq}; s) \]

\[ \text{impq} \leftarrow \text{iAgg}_q (\text{ics}) \]

\[ (\text{impq}, \text{ihsk}_{q',i}) \in \text{iAgg} (\text{ics}, q, (\text{pk}_{q',i}, \text{f}_{q',i}) \in [L_q]) \]

\[ (\text{pk}_{q',i}, \text{sk}_{q',i}) \leftarrow \text{iGen} (\text{ics}, q^*, i^*) \]

\[ \text{icrs} \leftarrow \text{Setup} (1, m, 1L_1, ..., 1L_m, 1n_i, 1n_2, M) \]

with n′ = k(n1 + n2 + k), n′ = n1n2k and s ← Coin. Note that all above equalities are ensured by the specification of Ver. We have

\[ z = ((s_1A_1 + s_2) \otimes (s_2A_2 + x_2)) \mathbf{f}_{q',i′} - \text{iDec} (\text{isk}_{q',i'}, \text{ihsk}_{q',i'}, (\text{ict}_q, \text{ict}_{q'})) \]

\[ = ((s_1A_1 + x_1) \otimes (s_2A_2 + x_2)) \mathbf{f}_{q',i} - x \mathbf{M} \mathbf{f}_{q',i} \quad (24) \]

\[ = (s_1A_1 \otimes s_2A_2 + s_1A_1 \otimes x_2 + x_1 \otimes s_2A_2) \mathbf{f}_{q',i} + (x_1 \otimes x_2) \mathbf{f}_{q',i} - x \mathbf{M} \mathbf{f}_{q',i} \]

\[ = (x_1 \otimes x_2) \mathbf{f}_{q',i} \quad (25) \]
where equality (24) follows from the correctness of $\Pi_2$, which is ensure by $iVer$ and $iGen$; equality (25) follows from the fact that $M = \begin{pmatrix} A_1 \otimes I_{n_2} \\ I_{n_1} \otimes A_2 \\ A_1 \otimes A_2 \end{pmatrix}$ and $x = (s_1 \otimes x_2 || x_1 \otimes s_2 || s_1 \otimes s_2)$. This proves the correctness.

**Compactness and Efficiency.** Our multi-instance slotted Reg-QFE has the following properties:

$$|\text{crs}| = O(L^2 \cdot n^2) \cdot \text{poly}(\lambda), \quad |\text{hsk}_{q,i}| = O(n) \cdot \text{poly}(\lambda),$$

$$|\text{mpk}_q| = O(n) \cdot \text{poly}(\lambda), \quad |\text{mpk}_q| = \text{poly}(\lambda),$$

$$|\text{ct}_i| = O(n) + \text{poly}(\lambda), \quad |\text{ct}_q| = \text{poly}(\lambda),$$

where $L = L_1 + \cdots + L_m$, $n = n_1 + n_2$.

**Security.** We have the following theorem. Given multi-instance slotted PReg-IPFE with very selective SIM-security under MDDH assumption, our multi-instance slotted Reg-QFE scheme uses prime-order bilinear group and the security can be reduced to bi-MDDH assumption.

**Theorem 6.** Assume $\Pi_2 = (\text{Setup}, iGen, iVer, iAgg, iAgg, iEnc, iEnc, iDec)$ is a multi-instance slotted PReg-IPFE with completeness, correctness, very selective SIM-security and has group-based simulator defined in Section 6, our multi-instance slotted Reg-QFE scheme achieves the very selective SIM-security, under bi-MDDH assumption.

### 7.2 Simulator

Recall that we allow some instance $q^*$ to be empty, namely $M_{q^*}, C_{q^*} = \emptyset$ and $f_{q^*} = \bot$, $\text{pk}_{q^*} = \bot$ for all $i \in [L_q]$. Let $(\widehat{\text{Setup}}, \widehat{iGen}, \widehat{iEnc}, \widehat{iEnc})$ be the group-based simulator of multi-instance slotted PReg-IPFE $\Pi_2$, the simulator of our multi-instance slotted Reg-QFE is as follows:

- $\widehat{\text{Setup}}(1^1, 1^m, 1^{L_1}, \ldots, 1^{L_m}, 1^{n_1}, 1^{n_2}, \{f_{q,i}\}_{i \in [L_q]}, \{\mu_{q,i}\}_{M_{q,i} \cup C_{q,i}} q \in [m])$: Sample

$$\mathbf{y}_1 \leftarrow Z_p^{1 \times n_1}, \mathbf{y}_2 \leftarrow Z_p^{1 \times n_2}, A_1 \leftarrow Z_p^{k \times n_1}, A_2 \leftarrow Z_p^{k \times n_2}$$

Set $n_1' = k(n_1 + n_2 + k)$, $n_2' = n_1 n_2$ and $M = \begin{pmatrix} A_1 \otimes I_{n_1} \\ I_{n_1} \otimes A_2 \\ A_1 \otimes A_2 \end{pmatrix}$, run

$$(\text{icrs}, \text{itd}) \leftarrow \widehat{\text{Setup}}(1^1, 1^m, 1^{L_q} q \in [m], 1^{n_1}, 1^{n_2}, \{M\}_{s \in [1,2]}; \{f_{q,i}\}_{i \in [L_q]}, \{([\mathbf{y}_1 \otimes \mathbf{y}_2 f_{q,i} \mu_{q,i}], s \in [1,2])\}_{i \in M_{q,i} \cup C_{q,i}} q \in [m])$$

Output

$$\widehat{\text{crs}} = ([A_1], [A_2], \text{icrs})$$

And set $\text{td} = (y_1, y_2, \text{itd})$.

- $\widehat{\text{Gen}}(\text{crs}, q, i; \text{td})$: Fetch itd from $\text{td}$, sample $(\text{ip}_{q,i}, \text{ihs}_{q,i}) \leftarrow \widehat{iGen}(\text{icrs}, q, i; \text{itd})$, output

$$(\text{pk}_{q,i}, \text{ihs}_{q,i}) = (\text{ip}_{q,i}, \text{ihs}_{q,i})$$

- $\widehat{\text{Enc}}(\text{td})$: Parse $\text{td} = (y_1, y_2, \text{itd})$, sample $\text{icrt} \leftarrow \widehat{iEnc}(\text{itd})$. Output

$$\text{ct}_q = ([y_1], [y_2], \text{icrt}),$$

- $\widehat{\text{Enc}}(\text{pk}_{q,1}, \ldots, \text{pk}_{q,L_q}, \text{td})$: If $q$ is an empty instance, on input $\text{pk}_{q,i} = \bot$ for all $i \in [L_q]$, abort and return $\text{ct}_q = \bot$.

For all $i \in [L_q]$, parse $\text{pk}_{q,i} = \text{ip}_{q,i}$. Fetch itd from $\text{td}$, sample $\text{icrt} \leftarrow \widehat{iEnc}((\text{ip}_{q,1}, \ldots, \text{ip}_{q,L_q}); \text{itd})$. Output

$$\text{ct}_q = \text{icrt}.$$
7.3 Proof

We prove the following technical lemma this immediately proves Theorem 6.

Lemma 6. For all adversaries $A$, there exist adversaries $B_1, B_2$ such that:

$$Adv_{A}^{\text{miReg-QFE}}(\lambda) \leq L \cdot Adv_{B_1}^{\text{miReg-IPFE}}(\lambda) + 2 \cdot Adv_{B_2}^{\text{bi-MDDH}}(\lambda) + \text{negl}(\lambda)$$

where $\text{Time}(B_1), \text{Time}(B_2) \approx \text{Time}(A)$.

For simplicity, we prove Lemma 6 in the case that all instances are not empty. For empty instance $q^*$, we simply change $ct^*$ and $pk_{q,i}^*$ to $\bot$, and we have $M_q^*, C_q^* = \emptyset$ in following game sequence.

**Game Sequence.** Suppose that $crs$ is the common reference string, $(\mathbf{x}_1^*, \mathbf{x}_2^*)$ is the challenge; for each instance $q \in \{m\}$, $(\{pk_{q,i}^*, f_{q,i}^*\})_{i \in [L_q]}$ are challenge public keys along with challenge functions to be registered, and $M_q^*, C_q^* \subseteq [L_q]$ are the sets of malicious and corrupted slots. For all $q \in \{m\}, i \in [L_q]$, define $D_{q,i} = \{pk_{q,i} : D_{q,i}[pk_{q,i}] = sk_{q,i} \neq \bot\}$ be responses to $\text{OGen}(q, i)$ and $C_{q,i} = \{pk_{q,i} : (i, pk_{q,i}) \in C_q\}$ records public keys in $D_i$ that have been sent to $\text{OCor}(q, i)$. Recall that, for each $q \in \{m\}, i \in [L_q]$, we require that

\[ i \in M_q^* \implies pk_{q,i}^* \notin D_{q,i} \land \text{Ver}(crs, q, i, pk_{q,i}^*) = 1 \]
\[ i \in C_q^* \implies pk_{q,i}^* \in C_{q,i} \]
\[ i \in [L_q] \setminus (M_q^* \cup C_q^*) \implies pk_{q,i}^* \in D_{q,i} \land pk_{q,i}^* \notin C_{q,i} \]

Note that $pk_{q,i}^*$ serves as a general entry in $D_{q,i}$ while $pk_{q,i}^*$ is the specific challenge public for slot $i$ in instance $q$; there can be more than one assignments for $pk_{q,i}^*$ since the adversary can invoke $\text{OGen}(q, i)$ (or $\overline{\text{OGen}}(q, i)$) for many times. We prove Lemma 6 using the following game sequence.

- **$G_0$:** This is the real game, recall that we have
  - $crs$ is in the form of
    
    \[ crs = ([A_1], [A_2], \text{icrs}) \]
    
    where
    
    \[ \text{icrs} \leftarrow \text{iSetup}(1^\lambda, 1^m, 1^{L_i}, \ldots, 1^{L_q}, 1^{n_i'}, 1^{n_i'}, \text{M}), \]
    
    with $n'_1 = k(n_1 + n_2 + k)$, $n'_2 = n_1n_2$ and $\text{M} = \begin{pmatrix} A_1 \otimes I_{n_2} \\ I_{n_1} \otimes A_2 \\ A_1 \otimes A_2 \end{pmatrix}$.
  
  - For each $q \in \{m\}$, $i \in [L_q]$, each $pk_{q,i} \in D_{q,i}$ and its corresponding $sk_{q,i}$ are
    
    \[ (pk_{q,i}, sk_{q,i}) = (ipk_{q,i}, isk_{q,i}) \]
    
    where $ipk_{q,i}, isk_{q,i} \leftarrow \text{iGen}(\text{icrs}, q, i)$
  
  - $(ct_{q,i}^*, ct_{q,i}^*)$ for $(\mathbf{x}_1^*, \mathbf{x}_2^*)$ is in the form:
    
    \[ (ct_{q,i}^*, ct_{q,i}^*)_{q \in \{m\}} = ([s_1A_1 + x_1^*], [s_2A_2 + x_2^*], ct_{q,i}^*, (ict_{q,i}^*)_{q \in \{m\}}) \]
    
    where $ict_{q,i}^* \leftarrow \text{iEnc}(impk_{q,i}, x^*; s), s \leftarrow \text{Coin}$.
- **$G_1$:** Identical to $G_0$, except that we replace $(\text{iSetup}, \text{iGen}, \text{iEnc}_{ss}, \text{iEnc})$ with $(\text{iSetup}, \text{iGen}, \overline{\text{iEnc}}_{ss}, \text{iEnc})$. In particular:
- crs is generated as
  \[ \text{crs} = ([A_1], [A_2], [i\text{crs}]) \]
  where
  \[ (i\text{crs}, \text{itd}) \leftarrow \text{iSetup}(1^λ, 1^m, \{1^L_q\}_{q \in [m]}, 1^{n_i}, 1^{n_2}, \{[M]_s\}_{s \in \{1, 2\}}, \{[f'_{q,i}]_s\}_{s \in \{1, 2\}}, \{[\theta_{q,i}]_s\}_{s \in \{1, 2\}} \)_{q \in [m]} \]
  with \( \theta_{q,i} = x^* M(f'_{q,i})' \).
- For each \( q \in [m], i \in [L_q] \), each \( pk_{q,i} \in D_{q,i} \) and its corresponding \( sk_{q,i} \) are generated as
  \[ (pk_{q,i}, sk_{q,i}) = (i\overline{pk}_{q,i}, isk_{q,i}) \quad \text{where} \quad (i\overline{pk}_{q,i}, isk_{q,i}) \leftarrow \text{iGen}(i\text{crs}, q, i; \text{itd}) \]
- \( (ct^*_s, (ct^*_{q})_{q \in [m]}) \) for \( (x'_1, x'_2) \) is in the form:
  \[ (ct^*_s, (ct^*_{q})_{q \in [m]}) = ([s_1 A_1 + x'_1], [s_2 A_2 + x'_2], [i\overline{ct}^*_s], (i\overline{ct}^*_{q})_{q \in [m]}) \]
  where \( i\overline{ct}^*_s \leftarrow \text{iEnc}_s(\text{itd}) \) and \( i\overline{ct}^*_{q} \leftarrow \text{iEnc}((\text{iEnc}_{s_1}, \ldots, \text{iEnc}_{s_{L_q}}); \text{itd}) \).

We have \( G_1 \approx_c G_0 \). This follows from the very selective SIM-security of \( \Pi_2 \) with the group-based simulator.

- \( G_2 \): Identical to \( G_1 \), except that we replace \( i\text{crs} \) in \( \text{crs} \) with
  \[ i\text{crs} \leftarrow \text{iSetup}(1^λ, 1^m, \{1^L_q\}_{q \in [m]}, 1^{n_i}, 1^{n_2}, \{[M]_s\}_{s \in \{1, 2\}}, \{[f'_{q,i}]_s\}_{s \in \{1, 2\}}, \{[\theta_{q,i}]_s\}_{s \in \{1, 2\}} \)_{q \in [m]} \]
  where
  \[ \theta_{q,i} = (s_1 A_1 + x'_1) \otimes (s_1 A_1 + x'_1)(f'_{q,i})' - (x'_1 \otimes x'_2)(f'_{q,i})' \]
  We have \( G_2 \equiv G_1 \). This follows from the fact that
  \[ x^* M = (s_1 \otimes x'_2 || x'_1 \otimes s_2, s_1 \otimes s_2) \]
  \[ \begin{pmatrix} A_1 \otimes I_{n_1} \\ I_{n_1} \otimes A_2 \\ A_1 \otimes A_2 \end{pmatrix} \]
  \[ = (s_1 A_1 + x'_1) \otimes (s_1 A_1 + x'_1) - (x'_1 \otimes x'_2) \]

- \( G_3 \): Identical to \( G_2 \), except that we replace all \( s_1 A_1 + x'_1 \) with \( \overline{y}_1 \leftarrow \mathbb{Z}_p^{n_1} \). In particular, we have
  \[ i\text{crs} \leftarrow \text{iSetup}(1^λ, 1^m, \{1^L_q\}_{q \in [m]}, 1^{n_i}, 1^{n_2}, \{[M]_s\}_{s \in \{1, 2\}}, \{[f'_{q,i}]_s\}_{s \in \{1, 2\}}, \{[\theta'_{q,i}]_s\}_{s \in \{1, 2\}} \)_{q \in [m]} \]
  where
  \[ [\theta'_{q,i}]_s = [\overline{y}_1 \otimes (s_1 A_1 + x'_1)(f'_{q,i})' - (x'_1 \otimes x'_2)(f'_{q,i})']_s \quad (\forall s \in \{1, 2\}) \]
  And we have
  \[ ct^*_s = ([\overline{y}_1], [s_2 A_2 + x'_2], [i\overline{ct}^*_s]) \]
  We have \( G_3 \approx_c G_2 \). This follows from the bi-MDDH assumption w.r.t. \( A_1 \) which ensure that \( ([A_1], [A_2], [s_1 A_1 + x'_1], [s_1 A_1 + x'_1]) \approx_c ([A_1], [A_2], [\overline{y}_1], [\overline{y}_1], [\overline{y}_1], [\overline{y}_1]) \) when \( s_1 \leftarrow \mathbb{Z}_p^{k \times n_1} \), \( A_1 \leftarrow \mathbb{Z}_p^{k \times n_1} \), \( x_1, \overline{y}_1 \leftarrow \mathbb{Z}_p^{1 \times n_1} \).

- \( G_4 \): Identical to \( G_3 \), except that we replace all \( s_2 A_2 + x'_2 \) with \( \overline{y}_2 \leftarrow \mathbb{Z}_p^{n_2} \). In particular, we have
  \[ i\text{crs} \leftarrow \text{iSetup}(1^λ, 1^m, \{1^L_q\}_{q \in [m]}, 1^{n_i}, 1^{n_2}, \{[M]_s\}_{s \in \{1, 2\}}, \{[f'_{q,i}]_s\}_{s \in \{1, 2\}}, \{[\theta'_{q,i}]_s\}_{s \in \{1, 2\}} \)_{q \in [m]} \]
  where
  \[ [\theta'_{q,i}]_s = [\overline{y}_1 \otimes \overline{y}_2 (f'_{q,i})' - (x'_1 \otimes x'_2)(f'_{q,i})']_s \quad (\forall s \in \{1, 2\}) \]
And we have
\[ c\mathcal{T}_i' = ([y_1], [y_2], ic\mathcal{T}_i') \]
We have \( G_3 \approx_c G_2 \). This follows from the bi-MDDH assumption w.r.t. \( A_1 \) which ensure that \( ([A_2], [A_2], [s_2A_2 + x_2^2], [s_2A_2 + x_2^2]) \approx_c ([A_2], [A_2], [y_2], [y_2], [\bar{y}_2]) \) when \( s_2 \leftarrow Z_p^{1 \times k}, A_2 \leftarrow Z_p^{k \times n_2}, x_2, \bar{y}_2 \leftarrow Z_p^{1 \times n_2}. \)

Observe that in the final game \( G_4 \) can be simulated using the simulator by setting \( \mu_{q,i} = (x_i \otimes x_2)(f^*_q) \)

References


Appendix

A Pre-constrained Reg-FE

In this section, we give the definition of pre-constrained Reg-FE for general functionality and its slotted variant.

A.1 Pre-Constrained Reg-FE

**Algorithms.** A pre-constrained registered functional encryption, with the functionalities: \( G = \{ g : X \rightarrow Y \} \), \( F = \{ f : Y \rightarrow Z \} \) and a pre-constrained \( g_0 \in G \), consists of six efficient algorithms:

- Setup\( (1^\lambda, F, G, g_0, 1^L) \rightarrow \text{crs} \): It takes as input the security parameter \( 1^\lambda \), the description of functionalities \( F, G, g_0 \) and the bounded number of user \( 1^L \), outputs a common reference string crs.
- Gen\( (\text{crs}, \text{aux}) \rightarrow (\text{pk}, \text{sk}) \): It takes as input crs and the public state aux, outputs key pair (pk, sk).
- Reg\( (\text{crs}, \text{aux}, \text{pk}, f) \rightarrow (\text{mpk}, \text{aux}') \): It takes as input crs, aux, and pk along with \( f \in F \), outputs master public key mpk and updated state aux'.
- Enc\( (\text{mpk}, x) \rightarrow \text{ct} \): It takes as input mpk, \( x \in X \), outputs a ciphertext ct.
- Upd\( (\text{aux}, \text{ct}) \rightarrow (\text{aux'}, \text{ct}) \): It takes as input aux, ct and outputs aux' along with ct.
- Dec\( (\text{sk}, \text{ct}) \rightarrow z/\bot/\text{getupd} \): It takes as input sk, ct and outputs z \in Z \) or a special symbol \( \bot \) 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 \( \lambda \):

\[
\Pr \left[ b = 1 \mid L \leftarrow \mathcal{A} ; \text{crs} \leftarrow \text{Setup}(1^\lambda, F, G, g_0, 1^L) ; \right. \right. \\
\left. \left. b = 0 ; \mathcal{A}^\text{ORegNT}(\cdot,:), \text{ORegT}(\cdot,:), \text{OEnc}(\cdot,:), \text{ODec}(\cdot)(\text{crs}) \right] \right.
\]

the oracles work as follows with initial setting aux = \( \bot \), ctr = 0, \( E = \emptyset \), \( R = \emptyset \) and \( t = \bot \):

- ORegNT\( (\text{pk}, f) \): run \( (\text{mpk}, \text{aux'}) \leftarrow \text{Reg} (\text{crs}, \text{aux}, \text{pk}, f) \), update aux = aux', ctr = ctr + 1, append (mpk, aux) to \( \mathcal{R} \) and return \( (|\mathcal{R}|, \text{mpk}, \text{aux}) \);
- ORegT\( (f^* ) \): run \( (\text{pk}^*, \text{sk}^*) \leftarrow \text{Gen} (\text{crs}, \text{aux}) , (\text{mpk}, \text{aux'}) \leftarrow \text{Reg} (\text{crs}, \text{aux}, \text{pk}^*, f^*) \), update aux = aux', ctr = ctr + 1 and set \( \text{ctr}^* = \text{ctr} \), compute \( \text{hsk}^* = \text{Upd} (\text{crs}, \text{aux}, \text{pk}^*) \), append (mpk, aux) to \( \mathcal{R} \), return \( (t = |\mathcal{R}|, \text{mpk}, \text{aux}, \text{pk}^*, \text{sk}^*, \text{hsk}^*) \);
- OEnc\( (i, x) \): let \( \mathcal{R}[i] = (\text{mpk}, \star ) \), run \( \text{ct} \leftarrow \text{Enc} (\text{mpk}, x) \), append \( (x, \text{ct}) \) to \( \mathcal{E} \) and return \( (|\mathcal{E}|, \text{ct}) \);
- ODec\( (j) \): let \( \mathcal{E}[j] = (x_j, c_j) \), compute \( z'_j \leftarrow \text{Dec} (\text{sk}^*, \text{hsk}^*, c_j) \); if \( z'_j = \text{getupd} \), run \( \text{hsk}^* = \text{Upd} (\text{crs}, \text{aux}, \text{pk}^*) \) and recompute \( z'_j \leftarrow \text{Dec} (\text{sk}^*, \text{hsk}^*, c_j) \). Set \( b = 1 \) when \( z'_j \neq f^* \circ g_0 (x_j) \).

with the following restrictions:

- for query to above oracles, it holds that \( \text{ctr} \leq L \);
- there exists one query to ORegT; (we can consider \( g_1^*, \ldots, g_L^*, f^*, \text{pk}^*, \text{sk}^*, \text{hsk}^* \) to be global;)
- for query \((i, x)\) to OEnc, it holds that \( i \geq t, \mathcal{R}[i] \neq \bot \);
- for query \((j)\) to ODec, it holds that \( \mathcal{E}[j] \neq \bot \).

**Compactness and Efficiency.** Compactness means that

\[
|\text{mpk}| = \text{poly}(\lambda, \text{par}, \log L), \quad |\text{hsk}_i| = \text{poly}(\lambda, \text{par}, \log L);
\]

where par is a parameter depending on the functionalities \( F, G \). Furthermore, update efficiency means that the number of invocations of Upd in ODec is at most \( O(\log |\mathcal{R}|) \) and each invocation costs \( \text{poly}(\log |\mathcal{R}|) \) time (in RAM model).
Very Selective Simulation-based Security (SIM-security). For all stateful adversary \( \mathcal{A} \), there exist simulator \((\text{Setup}, \text{Gen}, \text{Enc})\) such that:

\[
\begin{align*}
\left[ L, L', x^*, g_0, \{f'_i\}_{i \in [L']}, CK, HK, CH \leftarrow \mathcal{A}(1^\lambda); \\
\text{crs} \leftarrow \text{Setup}(1^\lambda, F, G, g_0, 1^\lambda); \\
\mathcal{A}^{\text{O}(\text{crs}, (f'_i)_{i \in [L']}, CK, HK, CH, \ldots)}(\text{crs}); \\
\text{ct}^* \leftarrow \text{Enc}(mpk, x^*), \alpha \leftarrow \mathcal{A}(\text{ct}^*) \\
\end{align*}
\]

\[
\begin{align*}
\left[ L, L', x^*, g_0, \{f'_i\}_{i \in [L']}, CK, HK, CH \leftarrow \mathcal{A}(1^\lambda); \\
\text{crs} \leftarrow \text{Setup}(1^\lambda, F, G, g_0, 1^\lambda); \\
\{f'_i\}_{i \in CK \cup HK}, \{f'_i \circ g_0(x^*)\}_{i \in CK \cup CH}; \\
\mathcal{A}^{\text{O}(\text{crs}, (f'_i)_{i \in [L']}, CK, HK, CH, \ldots)}(\text{crs}); \\
\text{ct}^* \leftarrow \text{Enc}_i((pk^*_1, \ldots, pk^*_L); \text{td}), \alpha \leftarrow \mathcal{A}(\text{ct}^*) \\
\end{align*}
\]

where \( CK, HK \subseteq [L'], CK \cup HK = [L'] \) for some \( L' \leq L \), \( CH \subseteq HK \) and \( CK \cap HK = \emptyset \), and \( O \) works as follows with a counter \( \ell = 1 \) and the same set of auxiliary data structure as in the definition of IND-security: on input \((i, pk^*_i)\), return \( \bot \) when \( i \neq \ell \), otherwise set \( \ell = \ell + 1 \) and do

\begin{itemize}
  \item when \( i \in CK \), return \( \text{ORegCK}(pk^*_i, f'_i) \);
  \item when \( i \in HK \), return \( \text{ORegHK}(f'_i) \); furthermore, if \( i \in CH \), return \( \text{OCorHK}([HK \cap [i]]) \).
\end{itemize}

Here \( \text{ORegCK} \) and \( \text{ORegHK} \) invoke \( \text{Reg} \) in both cases: in the real world (on the left-hand side), they use \( \text{crs} \) generated by \( \text{Setup} \) and \( \text{ORegHK} \) invokes \( \text{Gen} \); in the ideal world (on the right-hand side), they use \( \text{crs} \) simulated by \( \text{Setup} \) and \( \text{ORegHK} \) invokes \( \text{Gen} \).

Remark. We give several remarks on our formalization.

\begin{itemize}
  \item We do not require simulated version of \( \text{Reg} \) and \( \text{Upd} \) since both of them are public.
  \item We allow the adversary to choose \( pk^*_i \) at any point, only functions \( f_i \) and types of public keys (i.e., honest, malicious, honest but corrupted) are chosen “very selectively”.
  \item The set \( CH \) does not give the timing to invoke \( \text{OCorHK} \). One could let the adversary make an explicit query; however we call the oracle automatically just after the invocation of \( \text{ORegHK} \). This gives a simple but not weaker model in the very selective setting. In the definition, \( [HK \cap [i]] \) is the first item of the response of \( \text{ORegHK}(f'_i) \).
  \item In very selective SIM-security, there is no need to consider post-challenge queries. This relies on the fact that adversary should state the set \( CH \) at the beginning, so the pre-challenge and post-challenge corruption queries are equivalent in the very-selective SIM-security setting.
\end{itemize}

A.2 Pre-Constrained Slotted Reg-FE

Algorithms. A slotted pre-constrained registered functional encryption, with the functionalities: \( G = \{g : X \rightarrow Y\} \), \( F = \{f : Y \rightarrow Z\} \) and a pre-constrained \( g_0 \in G \), consists of six efficient algorithms:

\begin{itemize}
  \item \( \text{Setup}(1^\lambda, F, G, g_0, 1^\lambda) \rightarrow \text{crs} \): It takes as input the security parameter \( 1^\lambda \), the description of functionalities \( F, G \), a pre-constrained \( g_0 \in G \) and the upper bound \( 1^\lambda \) of the slot numbers, outputs a common reference string \( \text{crs} \).
  \item \( \text{Gen}(\text{crs}, i) \rightarrow (pk_i, sk_i) \): It takes as input \( \text{crs} \) and slot number \( i \in [L] \), outputs key pair \( (pk_i, sk_i) \).
  \item \( \text{Ver}(\text{crs}, i, pk_i) \rightarrow 0/1 \): It takes as input \( \text{crs}, i, pk_i \) and outputs a bit.
  \item \( \text{Agg}(\text{crs}, (pk_i, f_i)_{i \in [L]}) \rightarrow (mpk, hsk_j)_{j \in [L]} \): It takes as input \( \text{crs} \) and a series of \( pk_i \) with \( f_i \in F \) 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.
\end{itemize}
In the ideal world, OGen setting

Similarly, there is no need to give

We require that Agg and Dec are deterministic.

Completeness. For all $\lambda, L \in \mathbb{N}$, all $F, G$, all $g_0 \in G$ and all $i \in [L]$, we have

$$\Pr \left[ \text{Ver}(\text{crs}, i, \text{pk}_i) = 1 \left| \begin{array}{l}
\text{crs} \leftarrow \text{Setup}(1^\lambda, F, G, g_0, 1^L) \\
(\text{pk}_i, \text{sk}_i) \leftarrow \text{Gen}(\text{crs}, i)
\end{array} \right. \right] = 1.$$

Correctness. For all $\lambda, L \in \mathbb{N}$, all $F, G$, all $g_0 \in G$, all $i^* \in [L]$, all $\text{crs} \leftarrow \text{Setup}(1^\lambda, F, G, g_0, 1^L)$, all $(\text{pk}_i, \text{sk}_i) \leftarrow \text{Gen}(\text{crs}, i^*)$, all $(\text{pk}_i)_{i \in [L]} \setminus \{i^*\}$ such that $\text{Ver}(\text{crs}, i, \text{pk}_i) = 1$, all $x \in X$ and $f_1, \ldots, f_L \in F$, we have

$$\Pr \left[ \text{Dec}(\text{sk}_{i^*}, \text{hsk}_{i^*}, \text{ct}) = f_{i^*} \circ g_0(x) \left| \begin{array}{l}
(\text{mpk}, (\text{hsk}_j)_{j \in [L]}) \leftarrow \text{Agg}(\text{crs}, (\text{pk}_i, f_i)_{i \in [L]}) \\
(\text{ct} \leftarrow \text{Enc}(\text{mpk}, x))
\end{array} \right. \right] = 1.$$

Very Selective Simulation-based Security (SIM-security). For all stateful adversary $\mathcal{A}$, there exist simulator $(\text{Setup}, \text{Gen}, \text{Enc})$ such that

$$\left[ \begin{array}{l}
L, x^*, M^*, C^*, \{f_i^*\}_{i \in [L]} \leftarrow \mathcal{A}(1^\lambda) \\
\text{crs} \leftarrow \text{Setup}(1^\lambda, F, G, g_0, 1^L) \\
(\text{pk}_i^*)_{i \in [L]} \leftarrow \mathcal{A}^{\text{OGen}(\cdot), \text{Cor}(\cdot)}(\text{crs}) \\
(\text{mpk}, \ldots) \leftarrow \text{Agg}(\text{crs}, (\text{pk}_i^*, f_i^*)_{i \in [L]}) \\
\text{ct}^* \leftarrow \text{Enc}(\text{mpk}, x^*) \\
\alpha \leftarrow \mathcal{A}(\text{ct}^*)
\end{array} \right] \\\left[ \begin{array}{l}
L, x^*, M^*, C^*, \{f_i^*\}_{i \in [L]} \leftarrow \mathcal{A}(1^\lambda) \\
(\text{crs}, \text{td}) \leftarrow \text{Setup}(1^\lambda, F, G, g_0, 1^L; \{f_i^*\}_{i \in [L]}; \{f_i^* \circ g_0(x^*)\}_{i \in M^*; C^*}) \\
(\text{pk}_i^*)_{i \in [L]} \leftarrow \mathcal{A}^{\text{OGen}(\cdot), \text{Cor}(\cdot)}(\text{crs}) \\
\text{ct}^* \leftarrow \text{Enc}(\text{pk}_1^*, \ldots, \text{pk}_L^*; \text{td}) \\
\alpha \leftarrow \mathcal{A}(\text{ct}^*)
\end{array} \right]$$

where $M^*, C^* \subseteq [L]$ denote the sets of malicious and corrupted slots, and the oracles work as follows with initial setting $C = \emptyset$ and $D_i = \emptyset$ for all $i \in [L]$ and $q \in [m]$:

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

In the ideal world, OGen invokes Gen instead of Gen; and the following restrictions:

$$i \in M^* \implies D_i[\text{pk}_i^*] = \perp \land \text{Ver} (\text{crs}, i, \text{pk}_i^*) = 1$$

$$i \in C^* \implies (i, \text{pk}_i^*) \in C$$

$$\forall i \in [L] \setminus (M^* \cup C^*), D_i[\text{pk}_i^*] \neq \perp \land (i, \text{pk}_i^*) \notin C$$

Similarly, there is no need to give $\text{mpk}, \text{hsk}_1, \ldots, \text{hsk}_L$ to $\mathcal{A}$ explicitly in real game (or explicitly in simulation game) and consider post-challenge queries.

## B Registered Inner-product Encryption with Full Attribute Hiding

In this section, we present the slotted Reg-IPE with full attribute hiding, motivated by our slotted Reg-IPFE in Section 3.
Algorithms. A slotted registered inner-product encryption consists of six efficient algorithms:

- **Setup**($1^λ$, $1^n$, $1^L$) → crs: It takes as input the security parameter $1^λ$, the size of vector $1^n$ and the upper bound $1^L$ of the number of slots, outputs a common reference string crs.
- **Gen**(crs, $i$) → (pk$_i$, sk$_i$): It takes as input crs and slot number $i$ ∈ [L], outputs key pair (pk$_i$, sk$_i$).
- **Ver**(crs, $i$, pk$_i$) → 0/1: It takes as input crs, $i$, pk$_i$ and outputs a bit indicating whether pk$_i$ is valid.
- **Agg**(crs, (pk$_j$, y$_j$)$_{j \in [L]}$) → (mpk, (hsk$_j$)$_{j \in [L]}$): It takes as input crs and a series of pk$_i$ with y$_i$ ∈ $\mathbb{Z}_p^n$ for all $i$ ∈ [L], outputs master public key mpk and a series of helper keys hsk$_j$ for all $j$ ∈ [L]. This algorithm is deterministic.
- **Enc**(mpk, x, m) → ct: It takes as input mpk, x ∈ $\mathbb{Z}_p^n$ and message m, outputs a ciphertext ct.
- **Dec**(sk, hsk, ct) → m/⊥: It takes as input sk, hsk, ct and outputs m or a special symbol ⊥.

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

$$\Pr[\text{Ver}(\text{crs}, i, \text{pk}_i) = 1 | \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^n, 1^L); (\text{pk}_i, \text{sk}_i) \leftarrow \text{Gen}(\text{crs}, i)] = 1.$$  

**Correctness.** For all $\lambda, L, n \in \mathbb{N}$, all $i^* \in [L]$, all crs ← Setup($1^\lambda$, $1^n$, $1^L$), all (pk$_{i^*}$, sk$_{i^*}$) ← Gen(crs, $i^*$), all (pk$_j$, y$_j$)$_{j \in [L]}$ such that Ver(crs, $i$, pk$_i$) = 1, all x ∈ $\mathbb{Z}_p^n$ and y$_1$, . . . , y$_L$ ∈ $\mathbb{Z}_p^n$ such that xy$_{i^*}^\top = 0$, and all m, we have

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

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

$$\Pr\left[\begin{array}{l}
    \text{b = b'} \\
    \text{L} \leftarrow \mathcal{A}(1^\lambda); \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^n, 1^L) \\
    (\text{pk}_{i^*}^1, y_{i^*}^1)_{i \in [L]}, x_0^1, x_i^1, m_0^1, m_i^1 \leftarrow \mathcal{A}^{O\text{Gen}(\cdot), O\text{Cor}(\cdot)}(\text{crs}) \\
    (\text{mpk}, (\text{hsk}_j)_{j \in [L]}) \leftarrow \text{Agg}(\text{crs}, (\text{pk}_j, y_j)_{j \in [L]}) \\
    \text{b} \leftarrow \{0, 1\}, \text{ct}^* \leftarrow \text{Enc}(\text{mpk}, x_{b'}^1, m_{b'}^1); \text{b'} \leftarrow \mathcal{A}(\text{ct}^*)
\end{array}\right] = \frac{1}{2}\left[1 - \frac{1}{2}\right]$$  

is negligible in $\lambda$, where the oracles work as follows with initial setting $C = \emptyset$ and $D_i = \emptyset$ for all $i$ ∈ [L]:

- OGen($i$): run (pk, sk) ← Gen(crs, $i$), set $D_i$[pk] = sk and return pk.
- OCor($i$, pk): return $D_i$[pk] and update $C = C \cup \{(i, pk)\}$.

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

$$D_i[\text{pk}_i^1] = \perp \implies \text{Ver}(\text{crs}, i, \text{pk}_i^1) = 1.$$  

if $m_0^1 \neq m_1^1$, we require that

$$(i, \text{pk}_i^1) \in C \lor D_i[\text{pk}_i^1] = \perp \implies x_0^1(y_i^1)^\top \neq 0 \land x_i^1(y_i^1)^\top \neq 0$$

if $m_0^1 = m_1^1$, we require that

$$(i, \text{pk}_i^1) \in C \lor D_i[\text{pk}_i^1] = \perp \implies (x_0^1(y_i^1)^\top \neq 0 \land x_i^1(y_i^1)^\top \neq 0) \lor (x_0^1(y_i^1)^\top = x_i^1(y_i^1)^\top = 0)$$

We use $\text{Adv}_{\mathcal{A}}^{\text{Reg-SPE}(\lambda)}$ to denote the advantage function. Note that [HLWW23] showed that there is no need to give mpk and hsk$_1, . . . , hsk_L$ to $\mathcal{A}$ explicitly and to consider post-challenge queries.
B.1 Scheme

Assuming a QA-NIZK $Π₀ = (LGen, LPrv, LVer, LSim)$ for linear space over bilinear groups, our slotted Reg-IPE scheme works as follows in the prime-order bilinear group:

- Setup$(1^λ, 1^n, 1^L)$ : Run $G := (p, G₁, G₂, G₇, e) ← G(1^λ)$. Sample
  \[ A ← Z_p^{k×(2k+1)}, \quad B₁ ← Z_p^{(2k+1)×k}, \quad V ← Z_p^{(2k+1)×(2k+1)}, \quad W₀ ← Z_p^{(2k+1)×(2k+1)}, \quad k ← Z_p^{1×(2k+1)}. \]

  For all $i ∈ [L]$, sample
  \[ Wᵢ ← Z_p^{(2k+1)×(2k+1)}n}, \quad Rᵢ ← Z_p^{(2k+1)×(2k+1)}, \quad rᵢ ← Z_p^{1×k}. \]
  For all $i ∈ [L]$, write $Aᵢ = (Aᵢ | Rᵢ) ∈ Z_p^{(3k+2)×(2k+1)}$, run
  \[ (crsi, tdᵢ) ← LGen(1^λ, G₁, [Aᵢ]₁). \]

Output

\[ crs = \begin{cases} [A, AV, AW₀]₁, [Ak]₁ & \{crsi, [Rᵢ, AWᵢ]₁\}ᵢ∈[L] \{[B₁rᵢ, W₀B₁rᵢ + kᵢ]₂\}ᵢ∈[L] \{[Wᵢ(Iᵢ ⊗ B₁rᵢ)]₂\}ᵢ,l∈[L](\{i\} \setminus \{j\}) \end{cases}. \]

Note that we do not use $td₁, …, tdₖ$ in the actual scheme.

- Gen$(crs, i)$ : Sample $Uᵢ ← Z_p^{(2k+1)×(2k+1)}$. Define $Fᵢ = (T_qᵢ) = (AᵢUᵢ) ∈ Z_p^{(3k+2)×(2k+1)}$ and run
  \[ πᵢ ← LPrv(crsᵢ, [Fᵢ]₁, Uᵢ). \]

Fetch $\{[Bᵢrᵢ]₂\}ᵢ∈[L](\{i\})$ from $crs$ and output

\[ pkᵢ = \begin{cases} [UᵢTᵢ, RᵢUᵢ]₁, \{[UᵢB₁rᵢ]₂\}ᵢ∈[L](\{i\}), πᵢ \end{cases} \text{ and } skᵢ = Uᵢ. \]

- Ver$(crs, i, pkᵢ)$ : Parse $pkᵢ = ([Tᵢ, Qᵢ]₁, \{[hᵢ,j]₂\}ᵢ,l∈[L](\{i\}), πᵢ)$. Write $Fᵢ = (T_qᵢ)$ and check
  \[ LVer(crsᵢ, [Fᵢ]₁, πᵢ) ≡ 1. \]

For each $j ∈ [L] \setminus \{i\}$, check
\[ e([A]₁, [hᵢ,j]₂) ≡ e([Tᵢ]₁, [Bᵢrᵢ]₂). \]
If all these checks pass, output 1; otherwise, output 0.

- Agg$(crs, (pkᵢ, yᵢ)ᵢ∈[L])$ : For all $i ∈ [L]$, parse $pkᵢ = ([Tᵢ, Qᵢ]₁, \{[hᵢ,j]₂\}ᵢ,l∈[L](\{i\}), πᵢ)$. Output:

\[ mpk = \begin{cases} A, AW₀ + \sumᵢ [Tᵢ + AWᵢ(yᵢ ⊗ I₂k+1)], \sumᵢ AWᵢ, AV \end{cases} \]

and for all $j ∈ [L]$
\[ hsk_j = \begin{cases} Bᵢrᵢ, \sumᵢ [hᵢ,j + Wᵢ(Iᵢ ⊗ Bᵢrᵢ)yᵢ], \sumᵢ Wᵢ(Iᵢ ⊗ Bᵢrᵢ), W₀Bᵢrᵢ + kᵢ \end{cases} \]
where

Ver

U

pk

–

For each

Dec

Section 2.1) and the fact that

\{z_1, z_2\} = e([c_2]_1, [I_0 \otimes k_0]_2), \quad [z_2] = e([c_0]_1, [K_2]_2);

\{z_3, z_4\} = e([c_1]_1, [k_0]_2), \quad [z_4] = e([c_0]_1, [k_1]_2);

\{z_5, z_6\} = e([c_0 U_i]_1, [k_2]_2),

Recover

\[ z = \{z_3 - z_4 - z_5\} - \{z_1 - z_2\} y_i^c - z_6 \cdot c. \]

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

\[
\text{pk}_i = ([T_i, Q_i], ([h_{ij}]_{j \in [L] \setminus \{i\}}, \pi_i) = ([AU_i, RU_i], ([U_i B_i r_j^c]_{j \in [L] \setminus \{i\}}, \pi_i)
\]

for some \( U_i \leftarrow \mathbb{Z}_{(2k+1)}^{(2k+1)} \) and \( \pi_i \leftarrow \text{LPri}(\text{crs}_i, [A_i U_i]_1, U_i) \) where \( (\text{crs}_i, \text{td}_i) \leftarrow \text{LGen}(1^\lambda, \mathbb{G}_1, [A_i]_1) \) and \( A_i = (A_i)_R \) with \( A \leftarrow \mathbb{Z}_{(2k+1)}^{k \times (2k+1)} \). Then

– Write \( F_i = (T_i, Q_i) = (A_i U_i, \pi_i) \), we have \( \text{LVer}(\text{crs}_i, [F_i]_1, \pi_i) = 1 \) by the perfect completeness of \( \Pi_0 \) (see Section 2.4) and the fact that \( F_i = A_i U_i \);

– For each \( j \in [L] \setminus \{i\} \), we have \( e([A_i]_1, [U_i B_i r_j^c]_2) = e([A_i U_i]_1, [B_i r_j^c]_2) \) by the definition of bilinear map \( e \) (see Section 2.1) and the fact that \( A \cdot U_i B_i r_j^c = A U_i \cdot B_i r_j^c \).

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

Correctness. For all \( \lambda, L, n \in \mathbb{N} \), all \( i^* \in [L] \), all \( \text{crs} \leftarrow \text{Setup}(1^\lambda, 1^n, 1^L) \), all \( \text{pk}_{i^*}, \text{sk}_{i^*} \leftarrow \text{Gen}(\text{crs}, i^*) \), all \( \{\text{pk}_i\}_{i \in [L] \setminus \{i^*\}} \) such that \( \text{Ver}(\text{crs}, i, \text{pk}_i) = 1 \), for all \( y_1, \ldots, y_L \in \mathbb{Z}_p^\lambda \) and \( x \in \mathbb{Z}_p^\lambda \) such that \( x y_i^{c_i} = 0 \), we have:

\[
\text{sk}_{i^*} = U_i^*,
\]

\[
\text{ct} = \left( \frac{sA \cdot sAW_0 + \sum_{i \in [L]} (sT_i + sAW_i(y_i^c \otimes I_{2k+1}))), x \otimes sAV + \sum_{i \in [L]} sAW_i}{c_1 \cdot \frac{sAK_i^c \cdot m}{c}} \right)
\]

\[
\text{hsk}_{i^*} = \left( \frac{B_1 r_i^c, \sum_{i \in [L] \setminus \{i^*\}} (h_{i,i^*} + W_i (I_0 \otimes B_1 r_i^c) y_i^c), \sum_{i \in [L] \setminus \{i^*\}} W_i (I_0 \otimes B_1 r_i^c), W_0 B_1 r_i^c + k_i^c}{k_i^c} \right)
\]

where

\[
A_{h_i,i^*} = T_i B_1 r_i^c \quad \forall i \in [L] \setminus \{i^*\} \quad \text{and} \quad A U_i = T_i.
\]

68
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. we have

$$z_1 = (x \otimes s\mathbb{AV})(I_n \otimes B_1 r_i^*) + \sum_{i \in [L]} s\mathbb{AW}_i (I_n \otimes B_1 r_i^*)$$

$$= s\mathbb{AV}(x \otimes I_{2^{k+1}})(I_n \otimes B_1 r_i^*) + \sum_{i \in [L]} s\mathbb{AW}_i (I_n \otimes B_1 r_i^*)$$

$$= s\mathbb{AV}B_1 r_i^* x + \sum_{i \in [L]} s\mathbb{AW}_i (I_n \otimes B_1 r_i^*)$$

(26)

$$z_2 = \sum_{i \in [L] \setminus \{i^*\}} s\mathbb{AW}_i (I_n \otimes B_1 r_i^*)$$

$$z_3 = s\mathbb{AW}_0 + \sum_{i \in [L]} (sT_i B_1 r_i^* + s\mathbb{AW}_i(y_i^* \otimes I_{2^{k+1}})B_1 r_i^*)$$

$$= s\mathbb{AW}_0 + \sum_{i \in [L]} (sT_i B_1 r_i^* + s\mathbb{AW}_i(I_n \otimes B_1 r_i^*)y_i^*)$$

(27)

$$z_4 = \sum_{i \in [L] \setminus \{i^*\}} (sA_{h_i} r_i^* + s\mathbb{AW}_i (I_n \otimes B_1 r_i^*)y_i^*)$$

$$z_5 = sA_{u_i} B_1 r_i^*.$$  

$$z_6 = s\mathbb{AW}_0 B_1 r_i^* + sA k^*$$

and then

$$z = [(z_3 - z_4 - z_5) - (z_1 - z_2)y_i^* - z_6]r \cdot C$$

$$= [(s\mathbb{AW}_0 B_1 r_i^* + sT_i B_1 r_i^* + s\mathbb{AW}_i(I_n \otimes B_1 r_i^*)y_i^* - sA_{u_i} B_1 r_i^*) -$$

$$(s\mathbb{AV}B_1 r_i^* x e y_i^* + s\mathbb{AW}_i(I_n \otimes B_1 r_i^*)y_i^*)$$

$$- s\mathbb{AW}_0 B_1 r_i^* + sA k^*]r \cdot [sA k^*]_r \cdot m$$

(28)

$$= [-s\mathbb{AVB}_1 r_i^* \cdot x e y_i^*]_r \cdot m$$

(29)

$$= m$$

(30)

Here, equality (26) and equality (27) follows from the property of tensor product: $(M \otimes I) (I \otimes a^*) = M \otimes a^* = (I \otimes a^*) M$ for matrices of proper size; equality (28) follows from the fact that $A_{h_i} r_i^* = T_i B_1 r_i^*$, for all $i \in [L] \setminus \{i^*\}$; equality (29) follows from the fact that $T_i = A_{u_i}$; equality (30) follows from the fact that $x e y_i^* = 0$. This proves the correctness.

**Compactness and Efficiency.** Our slotted Reg-IPFE has the following properties:

$$|crs| = L^2 \cdot n \cdot poly(\lambda); \quad |mpk| = n \cdot poly(\lambda); \quad |hsk_j| = n \cdot poly(\lambda); \quad |ct| = n \cdot poly(\lambda).$$

Note that the total size of $\{crs\}_{i \in [L]}$ is $L \cdot poly(\lambda)$ 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 $poly(\lambda)$.

**Security.** We have the following theorem. Given pairing-based QA-NIZK in [KW15] with unbounded simulation soundness under MDDH assumption, our slotted Reg-IPFE scheme uses prime-order bilinear group and the security can be reduced to MDDH assumption and subgroup decision assumption.

**Theorem 7.** Assume $\Pi_0 = (LGen, LPrv, LVer, LSim) is a QA-NIZK with perfect completeness, perfect zero-knowledge and unbounded simulation soundness for linear space defined in Section 2.4, our slotted Reg-IPFE scheme achieves the attribute hiding security under MDDH assumption and subspace decision assumption.
B.2 Proof

We prove the following technical lemma; this immediately proves Theorem 7.

**Lemma 7.** For all adversaries $\mathcal{A}$, there exist adversaries $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$ and $\mathcal{B}_4$ such that:

$$\text{Adv}^{\text{Reg-IE}}_{\mathcal{A}}(\lambda) \leq L \cdot \text{Adv}^{\text{USS}}_{\mathcal{B}_1}(\lambda) + \text{Adv}^{\text{MDDH}}_{\mathcal{B}_2} + L \cdot \text{Adv}^{\text{SD}}_{\mathcal{B}_3} + L \cdot \text{Adv}^{\text{SD}}_{\mathcal{B}_4} \neg\text{negl}(\lambda)$$

where $L$ is the number of slots and $\text{Time}(\mathcal{B}_1), \text{Time}(\mathcal{B}_2), \text{Time}(\mathcal{B}_3), \text{Time}(\mathcal{B}_4) \approx \text{Time}(\mathcal{A})$.

**Game Sequence.** Suppose that $\text{crs}$ is the common reference string, $(x_0^*, x_1^*)$ and $(m_0^*, m_1^*)$ are the challenge pair, $(pk_i^*, y_i^*)_{i \in [L]}$ are challenge public keys along with challenge functions to be registered. Let $D_i = \{(pk_i, sk_i) : D_i[pk_i] = sk_i \neq \bot\}$ be responses to $O\text{Gen}(i)$ and $C$ records public keys in $D_1, \ldots, D_L$ that have been sent to $O\text{Cor}$. Recall that, for challenge public keys $(pk_i^*, y_i^*)_{i \in [L]}$, we require that

$$D_i[pk_i^*] = \bot \implies \text{Ver}(\text{crs}, i, pk_i^*) = 1,$$

and if $m_0^* \neq m_1^*$, we require that

$$(i, pk_i^*) \in C \lor D_i[pk_i^*] = \bot \implies x_0^*(y_i^*)^\top \neq 0 \land x_1^*(y_i^*)^\top \neq 0.$$ 

if $m_0^* = m_1^*$, we require that

$$(i, pk_i^*) \in C \lor D_i[pk_i^*] = \bot \implies (x_0^*(y_i^*)^\top \neq 0 \land x_1^*(y_i^*)^\top \neq 0) \lor (x_0^*(y_i^*)^\top = x_1^*(y_i^*)^\top = 0).$$

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 assignments for $pk_i$ since the adversary can invoke $O\text{Gen}(i)$ for many times. We prove the Lemma 7 via nested dual-system method using the following game sequence.

- $G_0$: This is the real game, recall that we have
  
  - $\text{crs}$ is in the form:
    $$\text{crs} = \left[ [A, AV, AW_0]_1, [Ak^c]_I \right]_{\mathcal{A}}\left[ \mathcal{crs}_i, [R_i, AW_i]_1 \right]_{i \in [L]}\left[ [B_1^r_j, W_0 B_1^r_j + k^c]_2 \right]_{j \in [L]}\left[ [W_i (I_n \otimes B_1^r_j)]_2 \right]_{j \in [L], i \in [L], j \in [L]} \right).$$
    
  - where $\mathcal{crs}_i \in \mathcal{LGen}(1^\lambda, \mathcal{G}_1, [A_1]_1), A_i = (A_i^c)_{\mathcal{R}_i}.$
  - For each $i \in [L]$, each $(pk_i, sk_i) \in D_i$ is in the form:
    $$pk_i = \left[ [AU_i, R_i U_i]_1, [[U_i B_i]^r_j]_2 \right]_{j \in [L] \setminus \{i\}}, \pi_i \text{ and } sk_i = U_i$$
    
  - where $\pi_i \leftarrow L\text{Prv}(\text{crs}_i, [F_i]_1, U_i), F_i = (A^c_U)_{R_{U_i}}.$
  - For all $i \in [L], pk_i^*$ is in the form:
    $$pk_i^* = \left[ [T_i^r, Q_i^c]_1, [[H_i^r_j]_2]_{j \in [L] \setminus \{i\}}, \pi_i^* \right]$$
    
  such that $\text{Ver}(\text{crs}, i, pk_i^*) = 1$ which means $\text{LVer} \left[ \mathcal{crs}_i, T_i^c, Q_i^c \right] = 1$ and $\text{Ah}_{i, j}^* = T_i^c B_i^r_j$ for each $j \in [L] \setminus \{i\}$. 

70
We have $G_0$. This follows from the perfect zero-knowledge of $\Pi_0$.

- $G_1$: Identical to $G_0$ except that, for all $i \in [L]$ and all $(pk_i, sk_i) \in D_i$, we replace $\pi_i$ with

$$\tilde{\pi}_i \leftarrow \text{LSim}(\text{crs}_i, \text{td}_i, [F_i]_1) \quad \text{where} \quad F_i = \begin{pmatrix} A_i \\ R_i U_i \end{pmatrix}.$$ 

We have $G_1 \equiv G_0$. This follows from the fact that both $R_i$ (in $G_1$) and $\tilde{R}_i$ (in $G_2$) are truly random since matrix $(\frac{sA}{12k+1})$ is full-rank.

- $G_2$: Identical to $G_1$ except that we sample $s \leftarrow Z_p^{1 \times k}$ along with $A$ and replace all $R_i$ in $\text{crs}$ with

$$\tilde{\mathbf{R}}_i = \mathbf{R}_i, \tilde{\mathbf{R}}_i \leftarrow Z_p^{2(2k+2) \times (2k+2)}.$$ 

We have $G_2 \equiv G_1$. This follows from the stronger unbounded simulation soundness of $\Pi_0$.

- $G_3$: Identical to $G_2$ except that we generate the $c_i'$ as follows:

$$c_i' = s\mathbf{A}W_0 + \sum_{i \in [L]} \left( e_i R_i^{-1} Q_i^* + s\mathbf{A}W_i (y_i^*) \otimes I_{2k+1} \right).$$

We have $G_3 \approx G_2$. This follows from stronger unbounded simulation soundness of $\Pi_0$ along with the fact that $\mathbf{LVer}(\text{crs}_i, [F_i^*], \pi_i^*) = 1$ for all $i \in [L]$ where $F_i^* = \begin{pmatrix} T_i \\ Q_i^* \end{pmatrix}$.

Assume $pk_i^* \notin D_i$, i.e., $pk_i^*$ is malicious. In the reduction, we guess $i^* \leftarrow [L]$ and obtain $A, \tilde{R}_{i^*}, \text{crs}_{i^*}$ as input; we simulate honestly as in $G_3$ except that for all $pk_i \in D_i$, we make an oracle query $\mathbf{LVer}([F_{i^*}, \pi_{i^*}^*])$ and get $\tilde{\pi}_{i^*}$ in it; we finally output $([F_{i^*}^*, \pi_{i^*}^*])$ in $pk_i^* \notin D_i$. Observe that once it happens that $e_i R_i^{-1} Q_i^* \neq sT_i^*$, we must have $F_i^* \notin A_{i^*}^*$. When $pk_i^* \in D_i$, we always have $G_3 \equiv G_2$.

- $G_4$: Identical to $G_3$ except that we replace all $s\mathbf{A}$ with $c \leftarrow Z_p^{1 \times (2k+1)}$; in particular, we generate $\tilde{\mathbf{R}}_i$ as follows:

$$\tilde{\mathbf{R}}_i = \mathbf{R}_i, \tilde{\mathbf{R}}_i \leftarrow Z_p^{2(2k+2) \times (2k+2)}$$

and generate the challenge ciphertext as follows:

$$\mathbf{c} = \mathbf{E} \mathbf{C} + \mathbf{C} \mathbf{W}_0 + \sum_{i \in [L]} (e_i R_i^{-1} Q_i^* + \mathbf{C} \mathbf{W}_i (y_i^*) \otimes I_{2k+1}), \mathbf{x}_b \otimes \mathbf{C} \mathbf{V} + \sum_{i \in [L]} \mathbf{C} \mathbf{W}_i$$

where $\mathbf{c} \leftarrow Z_p^{k \times (2k+1)}$.

We have $G_4 \approx G_3$. This follows from MDDH assumption which ensures that $([A], [sA]) \approx ([A], [c])$ when $A \leftarrow Z_p^{k \times (2k+1)}$, $s \leftarrow Z_p^{1 \times k}$, $c \leftarrow Z_p^{1 \times (2k+1)}$.

- $G_5$: Identical to $G_4$ except that for all $i \in [L]$, we replace $\mathbf{A} \mathbf{V}$ in $\text{crs}$ with

$$\mathbf{V} \leftarrow Z_p^{1 \times (2k+1)}$$

we replace $c\mathbf{V}$ in challenge ciphertext with

$$\mathbf{V} \leftarrow Z_p^{1 \times (2k+1)}$$
In particular, we generate crs as below:

$$\text{crs} = \left[ \begin{array}{c} [A, \tilde{v} \Delta W_0], [Ak^\top]_T \\ \{\text{crs}_i, [R_i, AW_i]\}_i \in \{L\} \\ \{[B_1, r_j, W_0, B_1, r_j + k']_j, \}_j \in \{L\} \\ \{[W_i(I_0 \oplus B_1, r'_j)]_j, \}_j \in \{L, L|\{j\}\} \end{array} \right].$$

and generate the challenge ciphertext as

$$ct^* = \left( \begin{array}{c} c_i \circ \text{w}_0 + \sum_{i \in \{L\}} (e_i, R_i \sum_{i \in \{L\}} cW_i), x'_i \circ [v] + \sum_{i \in \{L\}} cW_i \right) \uparrow \left[ [ck^\top]_T \cdot m_0 \right].$$

We have $G_5 \equiv G_4$. This follows from the fact that when $v$ is uniformly sampled from $\mathbb{Z}_p^{(2k+1) \times (2k+1)}$ and not published elsewhere, $(AV, cV)$ (in $G_4$) is statically equivalent with the uniformly sampled $(\tilde{v}, v)$ where $\tilde{v} \leftarrow \mathbb{Z}_p^{(2k+1) \times (2k+1)}$, $v \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}$ (in $G_5$), since both $A$ and $c$ are full row rank (with overwhelming probability).

- $G_6$: Identical to $G_5$ except that we randomly sample $B_2 \leftarrow \mathbb{Z}_p^{2k+1}, B_3 \leftarrow \mathbb{Z}_p^{(2k+1) \times k}$, and compute the dual basis $B_2, B_3, B_4$. And we change $c_2^*$ as follows:

$$c_2^* = x'_i \circ v(1,3) + [x_i]_0 \circ v(2) + \sum_{i \in \{L\}} cW_i$$

We have $G_6 \equiv G_5$. This follows from the following argument for $b' = b$ (in $G_5$) or $b' = 0$ (in $G_6$):

$$x'_i \circ [v] \neq \sum_{i \in \{L\}} cW_i = (cW_i) \circ [v]$$

This argument follows from the fact that the basis $B_2$ and dual basis $B_1^\perp$ are not revealed, so we have $(cW_i) \circ [v]$ is hidden, this can imply that $\sum_{i \in \{L\}} (cW_i)$ hides $x'_i \circ v(2)$.

- $G_7, \ell (\ell \in [0, L])$: Identical to $G_6$ except that for all $j \in \{\ell\}$ we replace all $B_1, r_j$ and $W_0, B_1, r_j + k'$ in crs with

$$\mathbf{d}_j \text{ and } W_0 \mathbf{d}_j + k' + c^\alpha$$

where $d_0 \leftarrow \text{span}(B_2)$, $\alpha \leftarrow \mathbb{Z}_p$ and $c \leftarrow \mathbb{Z}_p^{2k+1}$ such that $Ac^\top = 0, cc^\top = 1$. We have that

- $G_{7, 0} \equiv G_6$: the two games are actually identical, since $[0] = 0$;
- $G_{7, \ell - 1} \equiv G_7, \ell$ for all $\ell \in \{L\}$, we will employ a sub-sequence of games for the proof described later.

- $G_8$: Identical to $G_{7, L}$ except that we generate the $c_2^*$ as follows:

$$c_2^* = x'_i \circ v(1,3) + [x_i]_0 \circ v(2) + \sum_{i \in \{L\}} cW_i$$

We have $G_8 \equiv G_{7, L}$. The proof is analogous to that of $G_6 \equiv G_5$, with the fact that basis $B_1, B_3$ and dual basis $B_1^\perp, B_3^\perp$ are not revealed in $G_{7, L}$, we have the following argument for $b' = b$ (in $G_{7, L}$) or $b' = 0$ (in $G_8$):

$$x'_i \circ v(1,3) + \sum_{i \in \{L\}} (cW_i) \circ [v] \neq \sum_{i \in \{L\}} (cW_i) \circ [v]$$

- $G_9$: Identical to $G_8$ except that we replace terms $C^\top$ in $ct^*$ as $\left( C^\top \leftarrow \mathbb{T} \right)$. We have $G_9 \equiv G_8$. This follows from the following statistical argument:

$$\begin{array}{c}
\text{crs} \\
ct^* \leftarrow \mathbb{T}
\end{array}
\equiv \begin{array}{c}
(Ak^\top, \mathbf{k} + c^\top \alpha, \mathbf{c} - \alpha)
\end{array}
\equiv \begin{array}{c}
(Ak^\top, \mathbf{k}, \mathbf{c} - \alpha)
\end{array}$$

when $k \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}$ and the fact that $[\alpha]_T$ only appears in $C^\top$. We can prove the statement via change of variable $k' \leftarrow k - c^\top \alpha$.
Observe that, in the final game \( G_9 \) the challenge ciphertext \( ct \) is independent of the random bit \( b \) and the adversary’s advantage is exactly 0.

**From \( G_{7, t-1} \) to \( G_{7, t} \):** We are ready to prove \( G_{7, t-1} \approx_c G_{7, t} \) and this will complete the proof of Lemma 7. For this, we need the following sub-sequence of games for each \( \ell \in [L] \):

- \( G_{7, t-0} \): Identical to \( G_{7, t-1} \) where we recall \( \text{crs} \), \( pk_1 \in D_l \) and \( c_1', c_2' \), with highlighting relevant terms in the following sub-sequence with dashed boxes as follows:

  \[
  \text{crs} = \begin{pmatrix}
  |A, \bar{V}, AW_0|_1, |Ak|_T, \{\text{crs}_t, [\bar{R}_t, AW_t]\}_{j \in [L]} \\
  \{[d_j, W_0d_j + k' + c\alpha_j]_2\}_{j \in [t-1]}, [B_1, R_1, W_0B_1R_1 + k' + \bar{k}'_2] \\
  \{[B_1, R_j, W_0B_1R_j + k' + \bar{k}'_2]_j\}_{j \in [L]\{t\}} \\
  \{[W_i(I_n \otimes d_j)]_j\}_{j \in [t-1], i \in [L]\{t\}}, [\{W_i(I_n \otimes B_1R_j]\}_j \in [L]\{t\}] \\
  \{[W_i(I_n \otimes B_1R_j)]_j\}_{j \in [L]\{t\}, i \in [L]\{t\}} \\
  \end{pmatrix},
  \]

- \( \text{pk}_1 = \begin{pmatrix}
  T_j \quad Q_j \\
  [A_{\ell'}, \bar{R}_{\ell}, U_{\ell}, I_{\ell'}]_1, [U_{\ell'}d_j]_j \in [t-1], i \in [L]\{t\}] \\
  [A_{\ell'}, \bar{R}_{\ell}, U_{\ell}, I_{\ell'}]_1, [U_{\ell'}d_j]_j \in [t-1], i \in [L]\{t\}] \\
  T_j \quad Q_j \\
  h_{j,l} \quad h_{j,l} \\
  \end{pmatrix},
  \]

- \( c_1' = cW_0 + e_1R_1^{-1}Q_1' + cW_i((y_i')^\top \otimes I_{2k+1}) + \sum_{i \in [L]\{t\}} (e_1R_1^{-1}Q_1' + cW_i((y_i')^\top \otimes I_{2k+1})) \)

- \( c_2' = x_1' \otimes v^{(1)} + x_0' \otimes v^{(2)} + x_2' \otimes v^{(3)} + cW_0 + \sum_{i \in [L]\{t\}} cW_i \)

Where \( d_j \leftarrow \text{span}(B_j) \) for all \( \ell \in [L] \). We have \( G_{7, t-1, 0} = G_{7, t-1} \); all changes are conceptual.

- \( G_{7, t-1, 1} \): Identical to \( G_{7, t-1, 0} \) except that we replace all \( B_1R_j \) in \( \text{crs} \) with \( d_j' \) where \( d_j' \leftarrow \text{span}(B_j) \).

In particular, we change the dashed boxed term in \( \text{crs} \) and \( \text{pk}_1 \) as follows:

\[
[d_j', W_0d_j + k', [W_i(I_n \otimes d_j)]_j, [U_{\ell'}d_j]_j i \in [L]\{t\}] \\
\]

We have \( G_{7, t-1, 1} \approx_c G_{7, t-1, 0} \). This follow from the SD_{B_1 \rightarrow B_3} assumption which ensure that

\[
[t_0]_2 \approx_c [t_1]_2 \quad \text{given} \quad [B_1]_2, [B_2]_2, [B_3]_2, \text{basis}(B_1 \| B_3), \text{basis}(B_2) \]

Where \( t_0 \leftarrow \text{span}(B_j) \) corresponding to \( G_{7, t-0} \) and \( d_j' \leftarrow \text{span}(B_j) \) corresponding to \( G_{7, t-1, 1} \).

- \( G_{7, t-1, 2} \): Identical to \( G_{7, t-1, 1} \) except that we change the dashed boxed term in \( \text{crs} \) and \( \text{pk}_1 \) as follows:

\[
[d_j', W_0d_j + k', [W_i(I_n \otimes d_j)]_j, [U_{\ell'}d_j]_j i \in [L]\{t\}] \\
\]

We have \( G_{7, t-1, 2} \approx_c G_{7, t-1, 1} \). We provide some details in Section B.3.

- \( G_{7, t-1, 3} \): Identical to \( G_{7, t-1, 2} \) except that we generate the \( c_2' \) as follows:

\[
c_2' = x_1' \otimes v^{(1)} + x_0' \otimes v^{(2)} + x_2' \otimes v^{(3)} + cW_0 + \sum_{i \in [L]\{t\}} cW_i \]

We have \( G_{7, t-1, 3} \approx_c G_{7, t-1, 2} \). The proof in “honest case” is analogous to that in Section 3.3, the “corrupted or malicious case” has some difference and we provide it in Section B.4.

73
- \( G_{7, \ell - 1, 4} \): Identical to \( G_{7, \ell - 1, 3} \) except that we replace all \( d_f \) in crs with

\[
d_f \text{ where } d_f \leftarrow \text{span}(B_2)\]

In particular, we change the dashed boxed term in crs and \( pk_\ell \) as follows:

\[
\left[ d_f, W_0 d_f + k + e^c x_k \right]_2, \left[ \{ W_i (I_n \otimes d_f) \} \right]_2, \left[ U d_f \right]_2, \{ i \in [L] \setminus \{ \ell \} \}
\]

We have \( G_{7, \ell - 1, 4} \approx_c G_{7, \ell - 1, 3} \). This follow from the SD\(G_{1_2}^{\ell - 1} \rightarrow B_2 \) assumption which ensure that

\[
[t_0]_2 \approx_c [t_1]_2 \quad \text{given} \quad [B_1]_2, [B_2]_2, [B_3]_2, \text{basis}(B_2, B_3), \text{basis}(B_1)
\]

Where \( t_0 \leftarrow \text{span}(B_3) \) corresponding to \( G_{7, \ell - 1, 2} \), and \( d_f \leftarrow \text{span}(B_2) \) corresponding to \( G_{7, \ell - 1, 3} \).

- \( G_{7, \ell - 1, 5} \): Identical to \( G_{7, \ell - 1, 4} \) except that we generate the \( c^*_2 \) as follows:

\[
c^*_2 = x_i^\perp \otimes v^{(1)} + x_0^\perp \otimes v^{(2)} + x_0^\perp \otimes v^{(3)} + c W_\ell + \sum_{i \in [L] \setminus \{ \ell \}} c W_i
\]

We have \( G_{7, \ell - 1, 5} \approx_c G_{7, \ell - 1, 4} \). The proof is identical to that for \( G_{7, \ell - 1, 3} \approx_c G_{7, \ell - 1, 2} \).

Observe that \( G_{7, \ell - 1, 5} = G_{7, \ell} \) and this prove \( G_{7, \ell - 1} \approx_c G_{7, \ell} \).

### B.3 From \( G_{7, \ell - 1, 1} \) to \( G_{7, \ell - 1, 2} \)

The proof idea is analogous to that in Section 3.3 and [ZZGQ23]: For all \( j \in [\ell - 1] \), we rewrite \( d_f \leftarrow \text{span}(B_2) \) with \( B_2, f_j \), for some \( r_j \leftarrow Z_p \). And we define \( c^j = Z_p^{kx1} \) such that \( A c^j = 0 \) and \( c c^j = 1 \). With the orthogonality of dual basis, we can define \( d^j \in \text{span}(B_3^\perp) \) such that:

\[
d^j B_1 = 0, \quad d^j B_2 = 0, \quad d^j d^j_\ell = 1.
\]

With Lemma 2, we also consider following two cases:

#### Honest Case

In this case, we have \( pk^*_\ell = ([T^*_\ell, Q^*_\ell], \{ [h^*_j, \ell]_2 \} \in [L] \setminus \{ \ell \}, \pi^*_\ell) \in D_e \setminus C_e \). Namely, we know \( U^*_\ell \) (such that \( T^*_\ell = AU^*_\ell \) and \( Q^*_\ell = \hat{R} U^*_\ell \)) and \( U^*_\ell \) is hidden from the adversary. We can write the \( e_j \hat{R}^{-1}_e \) in \( c^j \) as \( \left[ c U^*_\ell \right] \) and replace \( \hat{R} \) in crs with a random \( R \) as in \( G_1 \). We prove \( G_{7, \ell - 1, 2} \approx_c G_{7, \ell - 1, 1} \) in this case using the following argument for \( b' = 1 \) (in \( G_{7, \ell - 1, 1} \)) or \( b' = 0 \) (in \( G_{7, \ell - 1, 2} \)):

\[
A^i, c^i, [R_i], B_1, B_2, d_i, AW_0, W_0, B_1, W_0 B_2, W_0 d_i + b' c^i \alpha
\]

\[
c, c W_0 + c U^*_\ell, A U^*_\ell, [R_i U^*_\ell], U^*_\ell B_1, U^*_\ell B_2
\]

\[
\approx_c A^i, c^i, [R_i], B_1, B_2, d_i, AW_0, W_0, B_1, W_0 B_2, W_0 d_i + b' c^i \alpha
\]

\[
c, c W_0 + c U^*_\ell, A U^*_\ell, [R_i U^*_\ell + \hat{u} d^i], U^*_\ell B_1, U^*_\ell B_2
\]

\[
\approx_A A^i, c^i, [R_i], B_1, B_2, d_i, AW_0, W_0, B_1, W_0 B_2, W_0 d_i + b' c^i \alpha + \left[ c^i \hat{u} \right]
\]

\[
c, c W_0 + c U^*_\ell + \left[ w d^i + u_i d^i \right], A U^*_\ell, [R_i U^*_\ell + R_i c^i u_i d^i] + \hat{u} d^i, U^*_\ell B_1, U^*_\ell B_2
\]

\[
\approx + c^i, [R_i], B_1, B_2, d_i, AW_0, W_0, B_1, W_0 B_2, W_0 d_i + b' c^i \alpha + \left[ c^i \hat{u} \right]
\]

\[
c, c W_0 + c U^*_\ell + \left[ w d^i + u_i d^i \right], A U^*_\ell, [R_i U^*_\ell + R_i c^i u_i d^i + \hat{u} d^i], U^*_\ell B_1, U^*_\ell B_2
\]

where \( \hat{u} \leftarrow Z_p^{1x(2k+2)} \) and \( u_i \leftarrow Z_p, w_i \leftarrow Z_p^{2xk} \). We justify each step as below: The first \( \approx_c \) uses Lemma 2 with \( M = (A^i), R = R_i, U = U^*_\ell, u = \hat{u} \). The second \( \approx_s \) uses change of variables:

\[
W_0 \leftarrow W_0 + c^i w d^i \quad \text{and} \quad U^*_\ell \leftarrow U^*_\ell + c^i u_i d^i
\]

The last \( \approx_s \) follows from the fact that \( \hat{u} \) hides \( R c^i u_i \), this implies that \( u_i \) can hide \( w \) in \( c^i \), and \( w \) hides \( b' \alpha \) in crs.
**Corrupted & Malicious Case.** In this case, we have \( pk^*_ρ \in C_ρ \cup \overline{D}_ρ \). And we only consider \( m^*_0 \neq m^*_1 \) here, since we don’t need to handle \( k^c \) to hide \( m^*_b \) if they are equal. It is required that \( x^*_0(y^*_ρ)^T \neq 0 \land x^*_1(y^*_ρ)^T \neq 0 \). We prove \( G_{7,ρ−1,2} \approx_c G_{7,ρ−1,1} \) in this case using the following argument for \( b' = 1 \) (in \( G_{7,ρ−1,1} \)) or \( b' = 0 \) (in \( G_{7,ρ−1,2} \)):

\[
A, c^+, B_1, B_2, d_ρ, AW_ρ, W_ρ(I_ρ \otimes B_1), W_ρ(I_ρ \otimes B_2)
\]

\[
AW_0, W_0B_1, W_0B_2, W_0d_ρ + b'c^+α
\]

\[
c, cW_0 + cW_ρ((y^*_ρ)^T \otimes I_{2k^1+1}) \cdot x^*_b \otimes v^{(3)} + cW_ρ
\]

\[
\approx_s A, c^+, B_1, B_2, d_ρ, AW_ρ, W_ρ(I_ρ \otimes B_1), W_ρ(I_ρ \otimes B_2)
\]

\[
AW_0, W_0B_1, W_0B_2, W_0d_ρ + b'c^+α + wd^μ
\]

\[
c, cW_0 + cW_ρ((y^*_ρ)^T \otimes I_{2k^1+1}) + wd^μ - x^*_b(y^*_ρ)^T v^{(3)} + cW_ρ
\]

\[
\approx_s A, c^+, B_1, B_2, d_ρ, AW_ρ, W_ρ(I_ρ \otimes B_1), W_ρ(I_ρ \otimes B_2)
\]

\[
AW_0, W_0B_1, W_0B_2, W_0d_ρ + b'c^+α + wd^μ
\]

\[
c, cW_0 + cW_ρ((y^*_ρ)^T \otimes I_{2k^1+1}) + wd^μ - x^*_b(y^*_ρ)^T v^{(3)}, cW_ρ
\]

We justify each step as follows: the first \( \approx_s \) uses the change of variables

\[
W_ρ \mapsto W_ρ - c^+(x^*_b \otimes v^{(3)}) \quad \text{and} \quad W_0 \mapsto W_0 + c^+wd^μ
\]

The second \( \approx_s \) uses the fact that \( v \) is hidden and \( x^*_b(y^*_ρ)^T \neq 0 \) (which is different to our slotted Reg-IPFE), so that \( x^*_b(y^*_ρ)^T v^{(3)} \) hides \( w \), so we have \( b'α \) is hidden.

**B.4 From \( G_{7,ρ−1,2} \) to \( G_{7,ρ−1,3} \) in Corrupted & Malicious Case**

In this section, we present the proof of "corrupted & malicious case" in \( G_{7,ρ−1,2} \approx_c G_{7,ρ−1,3} \), which is different to that in Section 3.3; and we omit the proof of "honest case", which is identical to that in Section 3.3. In the "corrupted & malicious case", we have \( pk^*_ρ \in C_ρ \cup \overline{D}_ρ \). It is required that \( x^*_0(y^*_ρ)^T \neq 0 \land x^*_1(y^*_ρ)^T \neq 0 \) or \( x^*_0(y^*_ρ)^T = x^*_1(y^*_ρ)^T = 0 \). We prove \( G_{7,ρ−1,3} \approx_c G_{7,ρ−1,2} \) in this case using the following argument for \( b' = b \) (in \( G_{7,ρ−1,2} \)) or \( b' = 0 \) (in \( G_{7,ρ−1,3} \)):

\[
A, c^+, B_1, B_2, d_ρ, AW_ρ, W_ρ(I_ρ \otimes B_1), W_ρ(I_ρ \otimes B_2)
\]

\[
c, e_1\overline{R}_ρ^{-1}Q_ρ + cW_ρ((y^*_ρ)^T \otimes I_{2k^1+1}) \cdot x^*_b \otimes v^{(3)} + cW_ρ
\]

\[
\approx_s A, c^+, B_1, B_2, d_ρ, AW_ρ, W_ρ(I_ρ \otimes B_1), W_ρ(I_ρ \otimes B_2)
\]

\[
c, e_1\overline{R}_ρ^{-1}Q_ρ + cW_ρ((y^*_ρ)^T \otimes I_{2k^1+1}) - x^*_b(y^*_ρ)^Tv^{(3)} + cW_ρ
\]

\[
\approx_s A, c^+, B_1, B_2, d_ρ, AW_ρ, W_ρ(I_ρ \otimes B_1), W_ρ(I_ρ \otimes B_2)
\]

\[
c, e_1\overline{R}_ρ^{-1}Q_ρ + cW_ρ((y^*_ρ)^T \otimes I_{2k^1+1}) - x^*_b(y^*_ρ)^Tv^{(3)}, cW_ρ
\]

We justify each step as follows: the first \( \approx_s \) uses the change of variables

\[
W_ρ \mapsto W_ρ - c^+(x^*_b \otimes v^{(3)})
\]

The second \( \approx_s \) uses the fact that \( v^{(3)} \) is hidden (which is different to that in slotted Reg-IPFE c.f. Section 3.3), so that \( x^*_b(y^*_ρ)^Tv^{(3)} \) can be hidden by \( v^{(3)} \) no matter \( x^*_b(y^*_ρ)^T \neq 0 \) or \( x^*_b(y^*_ρ)^T = 0 \).
C Slotted Reg-IPFE with Very Selective SIM-Security

This section gives a self-contained description of our slotted Reg-IPFE with very selective SIM-security which is implied by our pre-constrained slotted Reg-IPFE in Section 6.

C.1 Scheme

Assuming a QA-NIZK $\Pi_0 = (\text{LGen}, \text{LPrv}, \text{LVer}, \text{LSim})$ for linear space over bilinear groups, see Section 2.4; our multi-instance slotted Reg-IPFE scheme, works as follows in the prime-order bilinear group:

- Setup$(1^A, 1^m, 1^n, 1^{l_1}, \ldots, 1^{l_m})$: Run $\mathbb{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^A)$. Sample shared parts:
  \[ A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}, \quad V_1 \leftarrow \mathbb{Z}_p^{(2k+1) \times n}, \quad V_2 \leftarrow \mathbb{Z}_p^{(2k+1) \times n+1}, \quad \mathbf{v} \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}. \]

And sample
\[ \mathbf{D} \leftarrow \mathbb{Z}_p^{(k+1) \times k}, \quad \mathbf{w} \leftarrow \mathbb{Z}_p^{1 \times (k+1)} \]

For each instance $q \in [m]$, sample $B_q \leftarrow \mathbb{Z}_p^{(k+1) \times k}$, and for all $i \in [L_q]$, do following operations:

- Sample $t_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k}$, for $s \in \{1, 2\}$, set
  \[ [M_{q,i}]_s = \begin{bmatrix} I_n & 0_n \\ 0_{(k+1) \times n} & D_{q,i} \\ 0_n & wD_{q,i} \end{bmatrix} \in \mathbb{G}_s^{(n+1) \times (n+1)}. \]

- Sample
  \[ W_{1,q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times n(k+1)}, \quad W_{2,q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)(k+1)}, \quad W_{3,q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)}, \]
  and
  \[ R_{q,i} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+1)}, \quad R_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k}, \quad t_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k}. \]

- Run $(\text{crs}_{q,i}, \text{td}_{q,i}) \leftarrow \text{LGen}(1^A, \mathbb{G}_1, [A_{q,i}], 1)$, where
  \[ A_{q,i} = \begin{bmatrix} A_r \\ R_{q,i} \end{bmatrix} \in \mathbb{Z}_p^{(3k+2) \times (2k+1)}, \]

Output\(^{10}\)

\[ \text{crs} = \begin{cases} \{A, AV_1, AV_2, AV^\top_2, AV^\top_1, \} \\
\{\text{crs}_{q,i}, \{B_{q,i}, AW_{1,q,i}, AW_{2,q,i}, AW_3,q,i, A(W_{2,q,i}(D_{q,i} \otimes I_{k+1}) + W_{3,q,i}(wD_{q,i} \otimes I_{k+1})))\}_{i \in [L_q]} \}
\end{cases}. \]

- Gen$(\text{crs}, q, i)$: Sample $U_{q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)}$. Define $F_{q,i} = (T_{q,i} U_{q,i}) = A_{q,i} U_{q,i} \in \mathbb{Z}_p^{(3k+2) \times (2k+1)}$ and run
  \[ \pi_{q,i} \leftarrow \text{LPrv}(\text{crs}_{q,i}, [F_{q,i}], 1, U_{q,i}). \]

Fetch $\{[B_{q,i} R^\top_{q,i}]_2\}_{j \in [L_q] \setminus \{i\}}$ from crs and output
\[ pk_{q,i} = ([AU_{q,i}, R_{q,i} U_{q,i}], \{[U_{q,i} B_{q,i} R^\top_{q,i}]_2\}_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}) \]
and $sk_{q,i} = U_{q,i}$.

\(^{10}\)Note that we employ $i$ as the index for $W_q$’s and $M_q$’s while $j$ is the index for $r_q$’s; both of them range from 1 to $L_q$. One exception is the terms with $W_q$, which is conceptually $W_{q,i}(M_{q,i} \otimes B_{q,i} R^\top_{q,i})$ with $i = j$. Note that we do not use $\text{td}_{q,1}, \ldots, \text{td}_{q,L_q}$ and isk in the actual scheme.
- **ver(crs, q, i, pk_{q,i}):** Parse pk_{q,i} = ([T_{q,i}, Q_{q,i}],[[h_{q,i,j}]]_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}). Write F_{q,i} = (T_{q,i}) and check
\[ \mathsf{LVer}(\text{crs}_{q,i}, [F_{q,i}]_1, \pi_{q,i}) \equiv 1. \]
For each j \in [L_q] \setminus \{i\}, check
\[ e([A]_j, [h_{q,i,j}]_2) \equiv e([T_{q,i}]_1, [B_q r_{q,i,j}]_2). \]
If all these checks pass, output 1; otherwise, output 0.
- **Agg_{q}(crs):** Output:
\[ \text{mpk}_q = ([A, AV_1, AV_2, AV']_1). \]
- **Agg(crs, q, (pk_{q,i}, y_{q,i})_{i \in [L_q]}):** If q is an empty instance, on input (pk_{q,i}, y_{q,i}) = (∅, ∅) for all i \in [L_q], abort and return mpk_q = ∅, hsk_q = ∅ for all j \in [L_q]. For all i \in [L_q], parse pk_{q,i} = ([T_{q,i}, Q_{q,i}],[[h_{q,i,j}]]_{j \in [L_q] \setminus \{i\}}, \pi_{q,i}), and set \( y_{q,i} = (y_{q,i}||1) \in \mathbb{Z}_p^{1 \times (n_q+1)}. \) Output:
\[ \text{mpk}_q = \left[ \sum_{i \in [L_q]} (T_{q,i} + A(W_{1,q,i}(\bar{y}_{q,i} \otimes I_{k+1}) + W_{2,q,i}(D_{q,i}^r \otimes I_{k+1}) + W_{3,q,i}(\mathsf{wdT}_{q,i}^r \otimes I_{k+1}))) \right], \]
and for all j \in [L_q]
\[ h_{sk_q,j} = \left( B_q r_{q,j}, \sum_{i \in [L_q] \setminus \{j\}} (h_{q,i,j} + W_{1,q,i}(\bar{y}_{q,i} \otimes B_q r_{q,j}) + W_{2,q,i}(D_{q,i}^r \otimes B_q r_{q,j}) + W_{3,q,i}(\mathsf{wdT}_{q,i}^r \otimes B_q r_{q,j})) \right) \]
\[ \Xi_q = \left( [D_{q,i}^r, \mathsf{wdT}_{q,i}^r]_1 \right). \]
- **Enc_{q}(\text{mpk}_q, x):** Set \( x = (x||0_n) \in \mathbb{Z}_p^{1 \times (n_q+n)}. \) Sample s \( \leftarrow \mathbb{Z}_p^{1 \times k}. \) Output:
\[ \text{ct}_q = \left( \left[ \text{sA}_0, \text{sA}_1, \text{sAV}_2, \text{sAV'}_1 \right]_1. \right. \]
- **Enc(\text{mpk}_q):** Abort and return \( ∅ \) if mpk_q = ∅. Sample s \( \leftarrow \mathbb{Z}_p^{1 \times k}, \) output
\[ \text{ct}_q = \left[ \sum_{i \in [L_q]} (sT_{q,i} + sA(W_{1,q,i}(\bar{y}_{q,i} \otimes I_{k+1}) + W_{2,q,i}(D_{q,i}^r \otimes I_{k+1}) + W_{3,q,i}(\mathsf{wdT}_{q,i}^r \otimes I_{k+1}))) \right]. \]
- **Dec(\text{sk}_{q,i}, hsk_{q,i}, (\text{ct}_q, \text{ct}_{q'})):** Abort and return \( ∅ \) if \( ct_{q'} = ∅. \) Parse
\[ \text{sk}_{q,i} = U_{q,i}, \quad \text{hsk}_{q,i} = ([k_0^q, k_1^q, k_2^q, k_3^q, k_4^q]), \quad (\text{ct}_q, \text{ct}_{q'}) = ([c_{q,0}, c_{q,1}, c_{q,2}, c_{q,3}, c_{q'}]), \]
Recover
\[ [z_1]_T = e([c_{q'}]_1, [k_0]_2), \quad [z_2]_T = e([c_{q,0}]_1, [k_1]_2), \]
\[ [z_3]_T = e([c_{q,0} U_{q,i}]_1, [k_2]_2), \quad [z_4]_T = e([c_{q,0}]_1, [k_2]_2), \]
\[ [z_5]_T = e([c_{q,1}]_1, [y_{q,i}]_2), \quad [z_6]_T = e([c_{q,2}]_1, [k_3]_2), \]
\[ [z_7]_T = e([c_{q,3}]_1, [k_4]_2). \]
Compute
\[ [z]_T = [z_1 - z_2 - z_3 - z_4 + z_5 + z_6 + z_7]_T. \]
Recover z from \([z]_T \) via brute-force DLOG and output z.
Completeness. For all $\lambda, m, n \in \mathbb{N}$, all $L_1, \ldots, L_m \in \mathbb{N}$, all $q \in [m]$ and $i \in [L_q]$, all crs $\leftarrow$ Setup$(1^\lambda, 1^m, 1^n, 1^{L_1}, \ldots, 1^{L_m})$, and $(pk_{q,i}, sk_{q,i}) \leftarrow$ Gen$(crs, q, i)$, we have

$$pk_{q,i} = ([T_{q,i}, Q_{q,i}], \{h_{q,i,j}\}_{j \in [L_q]|\{i\}}, \pi_{q,i})$$

$$= ([AU_{q,i}, R_{q,i}U_{q,i}], \{[U_{q,i}B_qF_{q,j}^r]\}_{j \in [L_q]|\{i\}}, \pi_{q,i})$$

for some $U_{q,i} \leftarrow \mathbb{Z}_p^{(2k+1)\times (k+1)}$ and $\pi_{q,i} \leftarrow \text{LPrv}(crs_{q,i}, [A_{q,i}U_i], U_i)$ where $(crs_{q,i}, td_{q,i})$ $\leftarrow$ $\text{LGen}(1^\lambda, \mathbb{G}_1, [A_{q,i}])$ and $A_{q,i} = (A_r^i)$ with $A \leftarrow \mathbb{Z}_p^{(2k+1)\times (k+1)}$. Then

- Write $F_{q,i} = (T_{q,i}) = (U_{q,i}U_{q,i})$, we have $\text{LVer}(crs_{q,i}, [F_{q,i}], \pi_{q,i}) = 1$ by the perfect completeness of $\Pi_0$ (see Section 2.4) and the fact that $F_{q,i} = A_{q,i}U_{q,i}$;

- For each $j \in [L_q] \setminus \{i\}$, we have $e([A], [U_{q,i}B_qF_{q,j}]) = e([AU_{q,i}], [B_qF_{q,j}])$ by the definition of bilinear map $e$ (see Section 2.1) and the fact that $A \cdot U_{q,i}B_qF_{q,j} = AU_{q,i} \cdot B_qF_{q,j}$.

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

Correctness. For all $\lambda, m, n \in \mathbb{N}$, all $L_1, \ldots, L_m \in \mathbb{N}$, all $q \in [m]$ and $i^* \in [L_q^r]$; all crs $\leftarrow$ Setup$(1^\lambda, 1^m, 1^n, 1^{L_1}, \ldots, 1^{L_m})$, all $(pk_{q,i^*}, sk_{q,i^*}) \leftarrow$ Gen$(crs, q,i^*)$; all $(pk_{q,i})_{i \in [L_q^r]|\{i^*\}}$ such that $\text{Ver}(crs, q, i, pk_{q,i}) = 1$; all $x \in \mathbb{Z}_p^{1 \times n}$ and $y_{q,i} \in \mathbb{Z}_p^{1 \times s}$; for $s \in \{1, 2\}$, we have:

$$sk_{q,i^*} = U_{q,i^*},$$

$$ct_i = \left[\begin{array}{c}
\begin{array}{cccc}
\text{sA} & \text{sAV}_1 + x & \text{sAV}_2 & \text{sAV}_3
\end{array}
\end{array}\right],$$

$$ct_{q,i} = \left\{\begin{array}{c}
\sum_{i \in [L_q]} \left(\begin{array}{c}
\text{sT}_{q,i} + \text{sA}(W_{1,q,i}(y_{q,i} \otimes I_{k+1}) + W_{2,q,i}(D_{q,i} \otimes I_{k+1})) + W_{3,q,i}(wD_{q,i} \otimes I_{k+1}))
\end{array}\right)
\end{array}\right\},$$

$$hsk_{q,i^*} = \left\{\begin{array}{c}
\left[\begin{array}{c}
\text{B}_{q,F_{q,i^*}} + \sum_{i \in [L_q^r]|\{i^*\}} \left(\text{h}_{q,i^*,j} + W_{1,q,i^*,r}(y_{q,i^*,r} \otimes B_qF_{q,i^*,r}) + W_{2,q,i^*,r}(D_{q,i^*,r} \otimes B_qF_{q,i^*,r}) + W_{3,q,i^*,r}(wD_{q,i^*,r} \otimes B_qF_{q,i^*,r}))
\end{array}\right)
\end{array}\right],$$

$$Ah_{q,i,j^*} = T_{q,i}B_qF_{q,j^*} \quad \forall i \in [L_q^r] \setminus \{i^*\} \quad \text{and} \quad AU_{q,i} = T_{q,i^*}. $$

where
Note that here we actually consider $\text{hsk}_{q',i}$ for $j = i'$ and $\text{sav}_{q',i'}$ for $i = i'$ and all above equalities are ensured by Ver and Gen. We have

$$z_1 = \sum_{i \in [L_q]} (sT_{q',i} + sA(W_{1,q',i}(y_{q',i} \otimes I_{k+1}) + W_{2,q',i}(D_{q',i} \otimes I_{k+1}) + W_{3,q',i}(\text{wd}_{q',i} \otimes I_{k+1})))B_{q'}r_{q',i'}$$

$$= \sum_{i \in [L_q]} (sT_{q',i}B_{q'}r_{q',i' + sA(W_{1,q',i}(y_{q',i} \otimes B_{q'}r_{q',i'})) + W_{2,q',i}(D_{q',i} \otimes B_{q'}r_{q',i'}) + W_{3,q',i}(\text{wd}_{q',i} \otimes B_{q'}r_{q',i'})))$$

$$z_2 = \sum_{i \in [L_q]} (sAh_{q',i'} + sA(W_{1,q',i}(y_{q',i} \otimes B_{q'}r_{q',i'}) + W_{2,q',i}(D_{q',i} \otimes B_{q'}r_{q',i'}) + W_{3,q',i}(\text{wd}_{q',i} \otimes B_{q'}r_{q',i'})))$$

$$z_3 = sAU_{q',i}B_{q'}r_{q',i'}$$

$$z_4 = sA(W_{1,q',i}(y_{q',i} \otimes B_{q'}r_{q',i'}) + W_{2,q',i}(D_{q',i} \otimes B_{q'}r_{q',i'}) + W_{3,q',i}(\text{wd}_{q',i} \otimes B_{q'}r_{q',i'}))$$

$$+ sA(V_1y_{q',i'} + V_2D_{q',i'} + \text{wd}_{q',i'})$$

$$z_5 = sAV_1y_{q',i'} + xy_{q',i'}$$

$$z_6 = sAV_2D_{q',i'}$$

$$z_7 = sAV \text{wd}_{q',i'}$$

and then

$$z = z_1 - z_2 - z_3 - z_4 + z_5 + z_6 + z_7$$

$$= sT_{q',i}B_{q'}r_{q',i'} + sA(W_{1,q',i}(y_{q',i} \otimes B_{q'}r_{q',i'}) + W_{2,q',i}(D_{q',i} \otimes B_{q'}r_{q',i'}) + W_{3,q',i}(\text{wd}_{q',i} \otimes B_{q'}r_{q',i'}))$$

$$- sAU_{q',i}B_{q'}r_{q',i'} - sA(W_{1,q',i}(y_{q',i} \otimes B_{q'}r_{q',i'}) + W_{2,q',i}(D_{q',i} \otimes B_{q'}r_{q',i'}) + W_{3,q',i}(\text{wd}_{q',i} \otimes B_{q'}r_{q',i'}))$$

$$- sA(V_1y_{q',i'} + V_2D_{q',i'} + \text{wd}_{q',i'})$$

$$+ sAV_1y_{q',i'} + xy_{q',i'} + sAV_2D_{q',i'} + sAV \text{wd}_{q',i'}$$

$$= xy_{q',i'}.$$  \hfill \text{(32)}

Here, equality (31) follows from the property of tensor product: $(a^\top \otimes \mathbf{I})M = a^\top \otimes M$ for matrices of proper size; equality (32) follows from the fact that $Ah_{q',i'} = T_iB_{q'}r_{q',i'}$ for all $i \in [L_q] \setminus \{i'\}$; equality (33) follows from the fact that $T_{q',i'} = AU_{q',i'}$.

**Compactness and Efficiency.** Our multi-instance PReg-IPFE has the following properties:

$$|\text{crs}| = O(L^2 \cdot n) \cdot \text{poly}(\lambda), \quad |\text{hsk}_{q,i}| = \text{poly}(\lambda),$$

$$|\text{mp}_{k_1}| = O(n)\text{poly}(\lambda), \quad |\text{mp}_{k_2}| = \text{poly}(\lambda),$$

$$|\text{ct}_{a_i}| = O(n) + \text{poly}(\lambda), \quad |\text{ct}_{q}| = \text{poly}(\lambda),$$

where $L = L_1 + \cdots + L_m$. Note that the total size of $\{\text{crs}\}_{i \in [L]}$ is $L \cdot \text{poly}(\lambda)$ 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 $\text{poly}(\lambda)$.

**Security.** We have the following theorem. Given pairing-based QA-NIZK in [KW15] with unbounded simulation soundness under MDDH assumption, our multi-instance slotted Reg-IPFE scheme uses prime-order bilinear group and the security can be reduced to MDDH assumption.

**Theorem 8.** Assume $\Pi_0 = (\text{LGen}, \text{LPrv}, \text{LVer}, \text{LSim})$ is a QA-NIZK with perfect completeness, perfect zero-knowledge and unbounded simulation soundness for linear space defined in Section 2.4, our multi-instance slotted Reg-IPFE scheme achieves the very selective SIM-security as the definition in Section 5.1, under bi-MDDH assumption.
C.2 Simulator

Recall that we allow some instance $q^*$ to be empty, namely $M_{q^*}, C_{q^*} = \perp$ and $y_{q^*, i} = \perp, pk_{q^*, i} = \perp$ for all $i \in [L_{q^*}]$. Our simulator is as follows:

- Setup($1^4, 1^n, 1^{L_1}, \ldots, 1^{L_m}, 1^n$, $\{y_{q,i}\}_{i \in [L_q]}, \{\mu_{q,i}\}_{i \in M_q \cup C_q}, q \in [m]$): Run $G := (p, G_1, G_2, G_T, e) \leftarrow G(1^4)$. Sample shared parts:

  \[
  c \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}, \quad A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}, \quad V_1 \leftarrow \mathbb{Z}_p^{(2k+1) \times n}, \quad V_2 \leftarrow \mathbb{Z}_p^{(2k+1) \times k+1}, \quad V \leftarrow \mathbb{Z}_p^{1 \times (2k+1)}.
  \]

  And sample

  \[
  D \leftarrow \mathbb{Z}_p^{(k+1) \times k}, \quad w \leftarrow \mathbb{Z}_p^{1 \times (k+1)}
  \]

  For each instance $q \in [m]$, sample $B_q \leftarrow \mathbb{Z}_p^{(k+1) \times k}$, for all $i \in [L_q]$, $s \in \{1, 2\}$, set

  \[
  [\tilde{M}_q]_s = \begin{bmatrix} M & 0_{n_1} \\ 0_{n \times n_2} & \text{ict}_{q,i} \end{bmatrix}
  \]

  where

  \[
  \text{ict}_{q,i} = \begin{cases} \text{Enc}_1([ipk]_1, [ipk]_2, 0) & \text{if } i \in [L_q] \setminus (M_q \cup C_q) \\ \text{Enc}_1([ipk]_1, [ipk]_2, \mu_{q,i}) & \text{if } i \in M_q \cup C_q \\ \end{cases}
  \]

  and for all $i \in [L_q]$, do the following operations:

  - Set

    \[
    \theta_{q,i} = \begin{cases} 0 & \text{if } i \in [L_q] \setminus (M_q \cup C_q) \\ \mu_{q,i} & \text{if } i \in M_q \cup C_q \\ \end{cases}
    \]

  - Sample

    \[
    W_{1,q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times n(k+1)}, \quad W_{2,q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)(k+1)}, \quad W_{3,q,i} \leftarrow \mathbb{Z}_p^{(2k+1) \times (k+1)},
    \]

    and

    \[
    \tilde{R}_{q,i} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+2)}, \quad r_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k}, \quad t_{q,i} \leftarrow \mathbb{Z}_p^{1 \times k}.
    \]

    and compute

    \[
    \tilde{R}_{q,i} = \tilde{R}_{q,i} \left( \begin{bmatrix} c \\ I_{2k+1} \end{bmatrix} \right).
    \]

  - Run $(\text{crs}_{q,i}, \text{td}_{q,i}) \leftarrow \text{LGen}(1^4, G_1, [A_{q,i}]_1)$, where

    \[
    A_{q,i} = \left( \begin{array}{c} A \\ \tilde{R}_{q,i} \end{array} \right) \in \mathbb{Z}_p^{(3k+2) \times (2k+1)}.
    \]

Output

\[
\tilde{c} = \begin{bmatrix} [A, AV_1, AV_2, Av^\top]_1, \\ \{\text{crs}_{q,i}, [\tilde{R}_{q,i}, AW_1,q,i, AW_2,q,i, AW_3,q,i, A(W_2,q,i)(DF_{q,i} \otimes I_{k+1}) + W_3,q,i((wDF_{q,i} + \theta_{q,i}) \otimes I_{k+1}))\}_{i \in [L_q]} \\ [[DF_{q,i}, wDF_{q,i} \otimes + \theta_{q,i}, B_iq_i], W_{1,q,i}(I_n \otimes B_iq_i) + V_i]_{i \in [L_q]} \\ \{[W_{2,q,i}(DF_{q,i} \otimes B_iq_i)] + W_{3,q,i}((wDF_{q,i} + \theta_{q,i}) \otimes B_iq_i) + v_2DF_{q,i} \otimes v_i + \text{w}(wDF_{q,i} + \theta_{q,i})]_{i \in [L_q]} \\ \{[W_{1,q,i}(I_n \otimes B_iq_i), W_{2,q,i}(DF_{q,i} \otimes B_iq_i) + W_{3,q,i}((wDF_{q,i} + \theta_{q,i}) \otimes B_iq_i)]_{i \in [L_q], j \in [L_q]} \} \} \end{bmatrix}.
\]

And set the trapdoor as

\[
\text{td} = \left( c, V_1, V_2, v, w, \left( \tilde{R}_{q,i}, \text{td}_{q,i} \right)_{i \in [L_q]} \right)_{q \in [m]}
\]

for all $q \in [m]$, if $q$ is not empty instance, update

\[
\text{td} = \text{td} \cup \sum_{i \in [L_q]} (c(W_{1,q,i}(y_{q,i}) \otimes I_{k+1}) + W_{2,q,i}(DF_{q,i} \otimes I_{k+1}) + W_{3,q,i}((wDF_{q,i} + \theta_{q,i}) \otimes I_{k+1}))
\]
\[\text{apply sub-sequence Game Sequence. Suppose that slots. For all } \i \in \mathbb{M}, \text{public keys along with challenge functions to be registered, } \mathbb{M}\text{ proof. For an empty instance, we only need to remove the terms about Time where } \mathbb{L}\text{ records public keys in } \mathbb{D}, \text{define} \mathbb{M}\text{ dual-system method using the following game sequence. } \]

\[\text{pk}_{\mathbb{I}}\text{ than one assignment for } g \text{ and run } \mathbb{I} \in \{0,1\}, \text{ parse } \mathbb{L}\text{, there exist adversaries } B_i, B_j \text{ such that:} \]

\[\mathbb{L}\text{, } \mathbb{I}\text{, } \mathbb{M}\text{, } \mathbb{C}\text{ are challenge public for slot } \mathbb{I}\text{ and } \mathbb{L}\text{, output: } \]

\[\text{ct}_q = \left( c_{i,0} c_{i,1} c_{i,2} c_{i,3} \right) \]

\[-\text{En}(\mathbb{I}, \mathbb{M}, \mathbb{C}, \mathbb{L})\text{: If q is an empty instance, on input pk}_{\mathbb{I}}\text{, abort and return } \mathbb{L}\text{. For all } i \in \mathbb{I}, \text{ parse pk}_{\mathbb{I}} = (T_{\mathbb{I}} q_{\mathbb{I}}), \text{ and } \mathbb{L}\text{, output: } \]

\[-\text{En}(\mathbb{I}, \mathbb{M}, \mathbb{C}, \mathbb{L})\text{: If q is an empty instance, on input pk}_{\mathbb{I}}\text{, abort and return } \mathbb{L}\text{. For all } i \in \mathbb{I}, \text{ parse pk}_{\mathbb{I}} = (T_{\mathbb{I}} q_{\mathbb{I}}), \text{ and } \mathbb{L}\text{, output: } \]

\[\text{ct}_q = \left( c_{i,0} c_{i,1} c_{i,2} c_{i,3} \right) \]

\[\text{C.3 Proof}\]

We prove the following technical lemma this immediately proves Theorem 8.

**Lemma 8.** *For all adversaries } \mathcal{A}, \text{ there exist adversaries } B_1, B_2 \text{ such that:} \]

\[\text{Adv}^{\text{mReg-IPFE}}_\mathcal{A}(\lambda) \leq L \cdot \text{Adv}^{\text{US}}_{B_1}(\lambda) + (3L + 2L \cdot Q + 1)\text{Adv}^{\text{MDH}}_{B_2}(\lambda) + \text{negl}(\lambda) \]

where \( L = L_1 + \ldots + L_m \) is the number of slots, \( Q \) is the maximum number of queries on a slot made by \( \mathcal{A} \) and \( \text{Time}(B_1), \text{Time}(B_2) \approx \text{Time}(\mathcal{A}) \).

For simplicity, we prove Lemma 8 in the case of nonempty 1-instance and remove the index q in the following proof. For an empty instance, we only need to remove the terms about ct\_q and all pk\_q in following game sequence, and notice that M\^\*, C\^\* = 0 for empty instance. In the case of m-instance, it only needs to add back index q and apply sub-sequence G_{s,e-1,0,\ldots, G_{s,e-1,1}} to each instance.

**Game Sequence.** Suppose that crs is the common reference string, x\^\* is the challenge, \{pk\_i, y\_i\}\_i\in[\mathbb{I}] \text{ are challenge public keys along with challenge functions to be registered, } M\^\*, C\^* \subseteq [\mathbb{I}] \text{ are the sets of malicious and corrupted slots. For all } i \in [\mathbb{I}], \text{ define } D_i = \{pk\_i : D_i[|pk\_i|] = sk\_i \neq \bot\} \text{ be responses to OGen(i) and } C_i = \{pk\_i : (i, pk\_i) \in C_1\text{ records public keys in } D_i\text{ that have been sent to OCor(i, \_). Recall that, for each } i \in [\mathbb{I}], \text{ we require that} \]

\[i \in M\^* \implies pk\_i \notin D_i \land \text{Ver}(\text{crs, 1, i, pk\_i}) = 1 \]

\[i \in C\^* \implies pk\_i \in C_i \]

\[i \in [\mathbb{I}] \setminus (M\^* \cup C\^*) \implies pk\_i \in D_i \land pk\_i \notin C_i \]

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 OGen(i) for many times. We prove the Lemma 4 via dual-system method using the following game sequence.
– **$G_0$:** This is the real game, recall that we have

- **crs** is in the form:
  
  \[
  \text{crs} = \left( \begin{array}{c}
  \{ [A, AV_1, AV_2, AV^\top]_1, \\
  \{c_{rs,i}, [R_i, AW_1,i, AW_{2,i}, AW_{3,i}, A(W_{2,i}(DT^*_i \otimes I_{k+1}) + W_{3,i}(wDT^*_i \otimes I_{k+1}))]_1 \}_{i \in [L]} \\
  \end{array} \right),
  \]

  where $c_{rs,i} \in \text{LGen}(1^A, \mathbb{G}_1, [A_i]_1)$, with $A_i = \begin{bmatrix} A \end{bmatrix}$.

- For each $i \in [L]$, each $pk_i \in D_i$ is in the form
  
  \[
  pk_i = (\begin{array}{c}
  \{ [AU_i, R_i U_i]_1, \\
  \{[U_i Br^*_j]_2 \}_{j \in [L \setminus \{i\}], \pi_i} \\
  \end{array} \),
  \]

  where $\pi_i \leftarrow \text{LPrv(crs}_i, [F_i]_1, U_i)$. $F_i = (AU_i)_{R_0i}$, and $U_i$ is the corresponding $sk_i$.

- For all $i \in [L]$, $pk^*_i$ is in the form:
  
  \[
  pk^*_i = (\begin{array}{c}
  \{ [T^*_i, Q^*_i]_1, \\
  \{[H^*_i,j]_2 \}_{j \in [L \setminus \{i\}], \pi_i^*} \\
  \end{array})
  \]

  such that $\text{Ver}(\text{crs}_i, 1, i, pk^*_i) = 1$ which means $\text{LVer} \begin{bmatrix}
  crs_i, \\
  \begin{bmatrix} T^*_i & \pi_i^* \end{bmatrix} \\
  \end{bmatrix}_1 = 1$ and $Ah^*_i, j = T^*_i Br^*_j$ for each $j \in [L] \setminus \{i\}$.

- $ct^*_i$ for $x^*$ is in the form:
  
  \[
  ct^*_i = (\begin{array}{c}
  \{ sA, saV_1 + x^*, saV_2, saV^\top \}_1 \\
  \end{array} \),
  \]

  where $sA = c_{s,0}$. $c_{s,1}$. $c_{s,2}$. $c_{s,3}$

- $ct^*_i$ for $x^*$ is in the form:

  \[
  ct^*_i = \left[ \begin{array}{c}
  \sum_{l \in [L_q]} (sT_l + sA(W_{1,i}(y^*_l \otimes I_{k+1}) + W_{2,i}(DT^*_l \otimes I_{k+1}) + W_{3,i}(wDT^*_l \otimes I_{k+1})) \right] \begin{bmatrix} sA \end{bmatrix}
  \end{array} \right]_1.
  \]

– **$G_1$:** Identical to $G_0$, except that for all $i \in [L]$, we replace $wDT^*_i$ in crs with

  \[
  wDT^*_i + \Theta \quad \text{where} \quad \Theta = \begin{cases} 
  0 & \text{if } i \in [L_q] \setminus (M_q \cup C_q) \\
  X^*(y^*_l)^T & \text{if } i \in M_q \cup C_q
  \end{cases}
  \]

  In particular, we generate $\text{crs}$ as

  \[
  \text{crs} = \left( \begin{array}{c}
  \{ [A, AV_1, AV_2, AV^\top]_1, \\
  \{c_{rs,i}, [R_i, AW_1,i, AW_{2,i}, AW_{3,i}, A(W_{2,i}(DT^*_i \otimes I_{k+1}) + W_{3,i}(wDT^*_i \otimes I_{k+1}))]_1 \}_{i \in [L]} \\
  \end{array} \right).
  \]

  where $c_{rs,i} \in \text{LGen}(1^A, \mathbb{G}_1, [A_i]_1)$, with $A_i = \begin{bmatrix} A \end{bmatrix}$.

In particular, we generate $\text{crs}$ as

\[
\text{crs} = \left( \begin{array}{c}
  \{ [A, AV_1, AV_2, AV^\top]_1, \\
  \{c_{rs,i}, [R_i, AW_1,i, AW_{2,i}, AW_{3,i}, A(W_{2,i}(DT^*_i \otimes I_{k+1}) + W_{3,i}(wDT^*_i \otimes I_{k+1}))]_1 \}_{i \in [L]} \\
  \end{array} \right),
\]
and generate challenge ciphertext \( ct'_1 \) as

\[
\mathcal{C}_{i} = \left[ \sum_{t \in \mathcal{L}} \left( sT_1 + sA(W_{1,1}(y_i^1 \otimes I_{k+1}) + W_{2,1}(D\zeta_i + \theta_i \otimes I_{k+1}) + W_{3,1}( (wD\zeta_i + [b] \otimes I_{k+1}) ) \right) \right]_1
\]

We have \( G_1 \approx_c G_0 \). This follows from MDDH assumption which ensures that \( ([D]_1, [wD\zeta_i]_1) \approx_c ([D]_1, [wD\zeta_i + \theta_i]_1) \) when \( D \leftarrow \mathbb{Z}_p^{(k+1) \times k}, t_i \leftarrow \mathbb{Z}_p^{1 \times k}, w \leftarrow \mathbb{Z}_p^{1 \times (k+1)} \).

- \( G_2 \): Identical to \( G_1 \) except that for all \( i \in \mathcal{L} \) and all \( pk_i \in D_i \), we replace \( \pi_i \) with

\[
\tilde{\pi}_i \leftarrow \mathcal{L}_{\text{Sim}}(\text{crs}_{i_1}, \text{td}_{i_1}, [F_i]_1) \quad \text{where} \quad F_i = \begin{pmatrix} A U_i \\ I_{r_i} \end{pmatrix}.
\]

We have \( G_2 \equiv G_1 \). This follows from the perfect zero-knowledge of \( \Pi_0 \).

- \( G_3 \): Identical to \( G_2 \) except that we sample \( s \leftarrow \mathbb{Z}_p^{1 \times k} \) along with \( A \) and replace all \( R_i \) in \( \text{crs} \) with

\[
\tilde{R}_i = R_i \begin{pmatrix} s A \\ I_{2k+1} \end{pmatrix}, \quad \tilde{R} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+2)}.
\]

We have \( G_3 \equiv G_2 \). This follows from the fact that both \( R_i \) (in \( G_2 \)) and \( \tilde{R}_i \) (in \( G_3 \)) are truly random since matrix \( (s^A)_{1_{2k+1}} \) is full-rank.

- \( G_4 \): Identical to \( G_3 \) except that we generate the \( c'_i \) as follows:

\[
c'_i = \sum_{t \in \mathcal{L}} \left( e_1 R_i^{-1} Q_i^* + sA(W_{1,1}(y_i^1 \otimes I_{k+1}) + W_{2,1}(D\zeta_i + \theta_i \otimes I_{k+1}) + W_{3,1}( (wD\zeta_i + \theta_i) \otimes I_{k+1}) ) \right)
\]

We have \( G_4 \approx_c G_3 \). This follows from stronger unbounded simulation soundness of \( \Pi_0 \) along with the fact that \( \mathcal{L}_{\text{Ver}}(\text{crs}_{i_1}, [F_i^*], \pi_i^*) = 1 \) for all \( i \in \mathcal{L} \) where \( F_i^* = \begin{pmatrix} T_i^* \\ Q_i^* \end{pmatrix} \). The details are identical to that in game \( G_3 \) of our \( \text{sReg-IPFE} \) (c.f. Section 3).

- \( G_5 \): Identical to \( G_4 \) except that we replace all \( sA \) with \( c \leftarrow \mathbb{Z}_p^{1 \times (2k+1)} \); in particular, we generate \( \tilde{R}_i \) as follows:

\[
\tilde{R}_i = R_i \begin{pmatrix} c \\ I_{2k+1} \end{pmatrix}, \quad \tilde{R} \leftarrow \mathbb{Z}_p^{(2k+2) \times (2k+2)}
\]

and generate the challenge ciphertext \( ct'_* \) as follows:

\[
ct'_* = \left( [\begin{array}{cccc} c_{1,0} & c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,0} & c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,0} & c_{3,1} & c_{3,2} & c_{3,3} \\ \end{array}] \right).
\]

and generate the challenge ciphertext \( ct'_1 \) as follows:

\[
ct'_1 = \left( \sum_{t \in \mathcal{L}} \left( e_1 R_i^{-1} Q_i^* + c(W_{1,1}(y_i^1 \otimes I_{k+1}) + W_{2,1}(D\zeta_i + \theta_i \otimes I_{k+1}) + W_{3,1}( (wD\zeta_i + \theta_i) \otimes I_{k+1}) ) \right) \right]_1
\]

We have \( G_5 \approx_c G_4 \). This follows from MDDH assumption which ensures that \( ([A]_1, [sA]_1) \approx_c ([A]_1, [c]_1) \) when \( A \leftarrow \mathbb{Z}_p^{k \times (2k+1)}, s \leftarrow \mathbb{Z}_p^{1 \times k}, c \leftarrow \mathbb{Z}_p^{1 \times (2k+1)} \).

- \( G_6 \): Identical to \( G_5 \) except that
- we generate \( c'_t \) as follows:

\[
ct'_t = ([c, cV_1, cV_2, cV']_1)_{c_{t,0}, c_{t,1}, c_{t,2}, c_{t,3}}.
\]

- In crs, we change \([W_{1,j}(I_n \otimes Br_j^c) + V_1]_2\) for all \( j \in [L] \) as follows:

\[
[W_{1,j}(I_n \otimes Br_j^c) + V_1 + c^\perp(-x^*')]_2
\]

where \( c^\perp \in \mathbb{Z}_{2^{2k+1}} \) such that \( cc^\perp = 1 \) and \( Ac^\perp = 0 \).

We have \( G_6 \approx_c G_5 \). This follows from the change of variable \( V_1 \mapsto V_1 + c^\perp(-x^*) \).

- \( G_7 \): Identical to \( G_6 \), except that

- we generate \( c'_t \) as follows:

\[
ct'_t = ([c, cV_1, cV_2, cV']_1)_{c_{t,0}, c_{t,1}, c_{t,2}, c_{t,3}}.
\]

- In crs, we change \([W_{2,j}(Dt_j^c \otimes Br_j^c) + W_{3,j}((wDt_j^c + \theta_j) \otimes Br_j^c) + V_2Dt_j^c + v^*(wDt_j^c + \theta_j)]_2\) for all \( j \in [L] \) as follow:

\[
[W_{2,j}(Dt_j^c \otimes Br_j^c) + W_{3,j}((wDt_j^c + \theta_j) \otimes Br_j^c) + V_2Dt_j^c + v^*(wDt_j^c + \theta_j) + c^\perp \theta_j]_2
\]

where \( c^\perp \in \mathbb{Z}_{2^{2k+1}} \) such that \( cc^\perp = 1 \) and \( Ac^\perp = 0 \).

We have \( G_7 \approx_c G_6 \). This follows from the change of variable \( V_2 \mapsto V_2 + c^\perp(-w) \) and \( v \mapsto v + c^\perp \).

- \( G_{8,\ell}, (\ell \in [0, L]) \): Identical to \( G_8 \) except that for all \( j \in [\ell] \), we change \([W_{1,j}(I_n \otimes Br_j^c) + V_1 + c^\perp(-x^*)]_2\) in crs as follows:

\[
[W_{1,j}(I_n \otimes Br_j^c) + V_1 + c^\perp(-x^*)]_2
\]

and change \([W_{2,j}(Dt_j^c \otimes Br_j^c) + W_{3,j}((wDt_j^c + \theta_j) \otimes Br_j^c) + V_2Dt_j^c + v^*(wDt_j^c + \theta_j) + c^\perp \theta_j]_2\) in crs as follows:

\[
[W_{2,j}(Dt_j^c \otimes Br_j^c) + W_{3,j}((wDt_j^c + \theta_j) \otimes Br_j^c) + V_2Dt_j^c + v^*(wDt_j^c + \theta_j) + c^\perp \theta_j]_2
\]

We have that

- \( G_{8,0} = G_6 \): the two games are actually identical, since \([0] = \emptyset \);
- \( G_{8,\ell-1} \approx_c G_{8,\ell} \) for all \( \ell \in [L] \), we will employ a sub-sequence of games for the proof described later.

Observe that in the final game \( G_{8,\ell} \) can be simulated using the simulator by setting \( \mu_i = x^*(y_i^c)^c \), where we embed \( x'^*(y_i^c)^c \) into crs so that \( hsk_i \) for all \( i \in M^* \cup C^* \) and remove \( x^* \) from \( c^\ast \).

From \( G_{8,\ell-1} \) to \( G_{8,\ell} \). We are ready to prove \( G_{8,\ell-1} \approx_c G_{8,\ell} \) and this will complete the proof of Lemma 4. For this, we need the following sub-sequence of games for each \( \ell \in [L] \):
\(- G_{8,ℓ−1,0}: \) Identical to \(G_{8,ℓ−1}\) where we recall \(\text{crs}, \text{pk}_i \in D_i\) and \(\text{c}_i^*\), with highlighting relevant terms in the following sub-sequence with dashed boxes as follows:

\[
\text{crs} = \begin{pmatrix}
[A, AV_1, AV_2, AV_3]_1, \{[\text{DT}_{\ell,ℓ}(\text{wDT}_{\ell,ℓ} + \theta)]_{2} \}_{j \in [L]}
\{\text{crs}_2, [\tilde{R}_i, AW_1, AW_2, AW_3, A (W_2,2(\text{DT}_{\ell,ℓ}^c \otimes I_{k+1}) + W_3,2([\text{DT}_{\ell,ℓ} + \theta] \otimes I_{k+1})) \}_{1} \in [L]}
\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1]_{2} \}_{j \in [ℓ−1]}
\{[\text{Br}_{\ell,ℓ}^c, W_2,2(I_n \otimes \text{Br}_{\ell,ℓ}) + V_i + c^c(−x^*)] \}_{2} \}
\{[\text{Br}_{\ell,ℓ}, W_2,2(I_n \otimes \text{Br}_{\ell,ℓ}) + V_i + c^c(−x^*)] \}_{2} \}
\{[W_{2,ℓ}(\text{DT}_{\ell}^c \otimes \text{Br}_{\ell,ℓ}) + W_{3,ℓ}((\text{wDT}_{\ell} + θ) \otimes \text{Br}_{\ell,ℓ}) + V_2 \text{DT}_{\ell} + θ]_{2} \}_{j \in [ℓ−1]}
\{[W_{2,ℓ}((\text{DT}_{\ell}^c \otimes \text{Br}_{\ell,ℓ}) + W_{3,ℓ}((\text{wDT}_{\ell} + θ) \otimes Br_{\ell,ℓ}) + V_2 \text{DT}_{\ell} + θ]_{2} \}_{j \in [ℓ−1]}
\{[W_{2,ℓ}(\text{DT}_{\ell}^c \otimes \text{Br}_{\ell,ℓ}) + W_{3,ℓ}((\text{wDT}_{\ell} + θ) \otimes \text{Br}_{\ell,ℓ}) + V_2 \text{DT}_{\ell} + θ]_{2} \}_{j \in [ℓ−1]}
\{[W_{2,ℓ}(\text{DT}_{\ell}^c \otimes \text{Br}_{\ell,ℓ}) + W_{3,ℓ}((\text{wDT}_{\ell} + θ) \otimes \text{Br}_{\ell,ℓ}) + V_2 \text{DT}_{\ell} + θ]_{2} \}_{j \in [ℓ−1]}
\{[W_{2,ℓ}(\text{DT}_{\ell}^c \otimes \text{Br}_{\ell,ℓ}) + W_{3,ℓ}((\text{wDT}_{\ell} + θ) \otimes \text{Br}_{\ell,ℓ}) + V_2 \text{DT}_{\ell} + θ]_{2} \}_{j \in [ℓ−1]}
\{[W_{2,ℓ}(\text{DT}_{\ell}^c \otimes \text{Br}_{\ell,ℓ}) + W_{3,ℓ}((\text{wDT}_{\ell} + θ) \otimes \text{Br}_{\ell,ℓ}) + V_2 \text{DT}_{\ell} + θ]_{2} \}_{j \in [ℓ−1]}
\end{pmatrix}
\]

\[
\text{pk}_i = \begin{pmatrix}
\text{crs}_1, [\tilde{R}_i, AV_1, AV_2, AV_3]_1, \{[\text{DT}_{\ell,ℓ}(\text{wDT}_{\ell,ℓ} + \theta)]_{2} \}_{j \in [L]}
\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1]_{2} \}_{j \in [ℓ−1]}
\{[\text{Br}_{\ell,ℓ}^c, W_2,2(I_n \otimes \text{Br}_{\ell,ℓ}) + V_i + c^c(−x^*)] \}_{2} \}
\end{pmatrix}
\]

where \(c^+ \in \mathbb{Z}_p^{k+1} \) such that \(cc^+ = 1, Ac^+ = 0\). For all \(i \in [L]\), recall that

\[
\theta_i = \begin{cases}
0 & \text{if } i \in [L] \setminus (M \cup C) \\
x^c(y^*) & \text{if } i \in M \cup C
\end{cases}
\]

\(- G_{8,ℓ−1,1}: \) Identical to \(G_{8,ℓ−1,0}\) except that we replace all \(\text{Br}_{ℓ,ℓ}^c\) with \(\text{d}_ℓ^c \sim \mathbb{Z}_p^{k+1}\) in \(\text{crs}\); in particular, we change the dashed boxed term in \(\text{crs}\) and \(\text{pk}_i\) as follows:

\[
\begin{align*}
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \\
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \\
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \\
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \\
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \\
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \\
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \\
&\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2}
\end{align*}
\]

We have \(G_{8,ℓ−1,0} \simeq \approx G_{8,ℓ−1,0}\). This follows from MDDH assumption w.r.t. \([B]_2\) which ensures that \(([B]_2, [\text{Br}_{\ell,ℓ}]_2) \approx \mathbb{Z}_p^{k+1}x_k\) when \(B \leftarrow \mathbb{Z}_p^{k+1}x_k\), \(\text{kr} \leftarrow \mathbb{Z}_p^{k+1}x_k\), \(\text{d}_ℓ^c \leftarrow \mathbb{Z}_p^{k+1}x_k\).

\(- G_{8,ℓ−1,2}: \) Identical to \(G_{8,ℓ−1,1}\), except that we replace \(\text{wI}_{1,ℓ}(I_n \otimes \text{d}_ℓ^c) + V_1 + c^c(−x^*)\) with

\[
\text{W}_{1,ℓ}(I_n \otimes \text{d}_ℓ^c) + V_1 + c^c(−x^*)
\]

and replace \(\text{W}_{2,ℓ}(\text{DT}_{\ell}^c \otimes \text{d}_ℓ^c) + \text{W}_{3,ℓ}((\text{wDT}_{\ell} + θ) \otimes \text{d}_ℓ^c) + V_2 \text{DT}_{\ell} + θ]_{2} \}_{j \in [ℓ−1]}

\[
\{[\text{Br}_{\ell,ℓ}(I_n \otimes \text{Br}_{\ell,ℓ}) + V_1 + c^c(−x^*)]_{2} \}
\]

We have \(G_{8,ℓ−1,2} \simeq \approx G_{8,ℓ−1,1}\). With defining \(c^c \in \mathbb{Z}_p^{k+1}\) and \(d^c \in \mathbb{Z}_p^{1 \times (k+1)}\) such that \(cc^c = 1, Ac^c = 0\) and \(d^c d^c = 1, d^c B = 0\). We consider two cases
• Honest case ($\ell \in [L] \setminus (M^* \cup C^*)$): In this case, we have $\theta_\ell = 0$, and we have $pk_\ell^* = ([T_\ell^*, Q_\ell^*], \{[h_{ij}^*, \lambda^*]\}_{j \in [L] \setminus \ell}, \pi_\ell^*) \in D_\ell \setminus C_\ell$. Namely, we know $U_\ell'$ (such that $T_\ell^* = AU_\ell'$ and $Q_\ell^* = \widehat{R}_\ell U_\ell'$) and $U_\ell'$ is hidden from the adversary. We can write the dash boxed terms in $c_j^*$ as follows:

$$cU_{\ell}^* + c(W_{1,\ell}(y_{\ell}^* \otimes I_{k+1}) + W_{2,\ell}(DT_{\ell}^* \otimes I_{k+1}) + W_{3,\ell}((wDT_{\ell}^* + \theta_\ell) \otimes I_{k+1}))$$

and replace $\widehat{R}_\ell$ in $\text{crs}$ with a random $R_\ell$ as in $G_3$. And we can proof $G_{8,\ell-1,2} \approx_c G_{8,\ell-1,1}$ in this case using the following argument for all $b \in \{0,1\}$:

$$A, c^+, B, \{R_\ell\}, d_\ell, AW_{1,\ell}, AW_{2,\ell}, AW_{3,\ell}, W_{1,\ell}(I_\ell \otimes d_\ell^*) + V_1 + b c^-(x^*)\]$$

$$W_{2,\ell}(DT_{\ell}^* \otimes d_\ell^*) + W_{3,\ell}((wDT_{\ell}^* + \theta_\ell) \otimes d_\ell^*) + V_2 DT_{\ell} + v^*(wDT_{\ell}^* + \theta_\ell) + b c^+\theta_\ell\]$$

$$c, cu_{\ell}^* + c(W_{1,\ell}(y_{\ell}^* \otimes I_{k+1}) + W_{2,\ell}(DT_{\ell}^* \otimes I_{k+1}) + W_{3,\ell}((wDT_{\ell}^* + \theta_\ell) \otimes I_{k+1}))$$

$$AU_{\ell}', [R_\ell U_{\ell}^* + \text{u d}^+], U_{\ell}B$$

where $\text{u} \leftarrow \mathbb{Z}_p^{1 \times (2k+2^2)}$, $u_\ell \leftarrow \mathbb{Z}_p$ and $w_\ell \leftarrow \mathbb{Z}_p^{n_1}$. We justify each step as below: The first $\approx_c$ follows the argument:

$$(A, c, \{R_\ell\}, B, d_\ell, AU_{\ell}, cU_{\ell}[RU_{\ell}], U_{\ell}B) \approx_c (A, c, \{R_\ell\}, B, d_\ell, AU_{\ell}, cU_{\ell}[RU_{\ell} + \text{u d}^+], U_{\ell}B)$$

which is analogous to the Lemma 2 in [ZZGQ23]. The second $\approx_s$ uses the change of variables:

$$U_{\ell}' \mapsto u_{\ell}' + c^+ u_\ell d^+ \quad \text{and} \quad W_{1,\ell} \mapsto W_{1,\ell} + c^+(w_{\ell} \otimes d^+)$$

The last $\approx_s$ is straight-forward with

* the fact that $\theta_\ell = 0$ in this case;
* the observation that $\text{u}$ hides $R_\ell c^+ u_\ell$, this implies that $u_\ell$ hides $w_{\ell}(y_{\ell}^*)^y$, and $w_{\ell}$ is sufficient to hide $x^*$.  

• Corrupted & Malicious Case ($\ell \in (\mathcal{M}^* \cup C^*)$): And in this case, we have $\theta_\ell = x^*(y^*_\ell)^\top$, and we have $p_k^\ell = ([T_\ell^*,Q_\ell^*]^T_2)_{\ell \in \{\ell \mid k \} \cap \mathcal{E}_\ell} \in C_\ell \cup \mathcal{D}_\ell$. We prove $G_{8,\ell-1,3} \cong_c G_{8,\ell-1,1}$ in this case using the following argument:

\[
A, c^+, B, d^\ell, AW_\ell, AW_1, AW_2, AW_3, AW_3, [W_1,\ell, (I_0 \otimes d^\ell_\ell)] + V_1 + c^+(-x^*)_2
\]

\[
[W_2,\ell,(DT_\ell^\top \otimes d^\ell_\ell)] + [W_3,\ell,((wD_\ell^\top \otimes \theta_\ell) \otimes d^\ell_\ell)] + [V_2,\ell DT_\ell^\top + V_2^\top (wD_\ell^\top \otimes \theta_\ell) + c^+\theta_\ell]_2;
\]

\[
[c, cu^\ell_\ell + c([W_1,\ell, (y_\ell^\top \otimes I_{k+1}) + W_2,\ell,(DT_\ell^\top \otimes I_{k+1}) + W_3,\ell,((wD_\ell^\top \otimes \theta_\ell) \otimes I_{k+1}))])_1
\]

\[
\approx A, c^+, B, d^\ell, AW_\ell, AW_1, AW_2, AW_3, AW_3, [W_1,\ell, (I_0 \otimes d^\ell_\ell)] + V_1 + c^+(-x^*)_2
\]

\[
[W_2,\ell,(DT_\ell^\top \otimes d^\ell_\ell)] + [W_3,\ell,((wD_\ell^\top \otimes \theta_\ell) \otimes d^\ell_\ell)] + [V_2,\ell DT_\ell^\top + V_2^\top (wD_\ell^\top \otimes \theta_\ell) + c^+\theta_\ell]_2;
\]

\[
[c, cu^\ell_\ell + c([W_1,\ell, (y_\ell^\top \otimes I_{k+1}) + W_2,\ell,(DT_\ell^\top \otimes I_{k+1}) + W_3,\ell,((wD_\ell^\top \otimes \theta_\ell) \otimes I_{k+1}))]) + x^*(y_\ell^*)^\top d^\ell + wD_\ell^\top d^\ell + (-wD_\ell^\top - \theta_\ell)^d^\ell}_1
\]

We justifying each step as follows: The first $\approx$ uses the change of variables:

\[
W_1,\ell \mapsto W_1,\ell + c^+(x^* \otimes d^\ell), \quad W_2,\ell \mapsto W_2,\ell + c^+(w \otimes d^\ell), \quad W_3,\ell \mapsto W_3,\ell + c^+((-1) \otimes d^\ell)
\]

The second $\approx$ follows from the fact that $\theta_\ell = x^*(y_\ell^*)_\ell$ in this case.

\[G_{8,\ell-1,3}: \text{Identical to } G_{8,\ell-1,2} \text{ except that we replace all } d^\ell_\ell \text{ with } Br^\ell_\ell \text{ where } r^\ell_\ell \leftarrow \mathbb{Z}^k_p \text{ in } \text{crs}; \text{ in particular, we change the dashed boxed term in } \text{crs and } p_k^\ell \text{ as follows:}
\]

\[
\begin{align*}
&[[Br^\ell_\ell, W_1,\ell, (I_0 \otimes Br^\ell_\ell)] + V_1 + c^+(-x^*)_2 \\
&[W_2,\ell,(DT_\ell^\top \otimes Br^\ell_\ell)] + [W_3,\ell,((wD_\ell^\top \otimes \theta_\ell) \otimes Br^\ell_\ell)] + [V_2,\ell DT_\ell^\top + V_2^\top (wD_\ell^\top \otimes \theta_\ell) + c^+\theta_\ell]_2 \\
&\{[W_1,\ell, (I_0 \otimes Br^\ell_\ell), W_2,\ell,(DT_\ell^\top \otimes Br^\ell_\ell) + W_3,\ell,((wD_\ell^\top \otimes \theta_\ell) \otimes Br^\ell_\ell)] \}_\ell \in \{\ell \mid k \} \cap \mathcal{E}_\ell
\end{align*}
\]

We have $G_{8,\ell-1,1} \cong_c G_{8,\ell-1,0}$. This follows from MDDH assumption w.r.t. $[B]_2$ which ensures that $([B]_2, [Br^\ell_\ell]_2) \cong ([B]_2, [d^\ell_\ell]_2)$ when $B \leftarrow \mathbb{Z}^{(k+1)\times k}_p, r^\ell_\ell \leftarrow \mathbb{Z}^{1\times k}_p, d^\ell_\ell \leftarrow \mathbb{Z}^{1\times (k+1)}_p$.

## D Sanity Check of the Simulators

This section provides sanity check for all simulators appeared in this paper.

### D.1 Sanity Check of the simulator in Section 6.2

In this section, we show that the simulator of our multi-instance slotted PReg-IPFE in Section 6.2 can pass the sanity check. The simulated $\text{crs}$ has the full capacity as the $\text{crs}$ of the scheme in Section 6.1. For all $\lambda, m, n_1, n_2 \in \mathbb{N}$, all $L_1, \ldots, L_m \in \mathbb{N}$, all $\mathbf{M} \in \mathbb{Z}^{m \times n_2}_p$, all $\{\mathcal{M}^*_q, C^*_q\}_{q \in \{m\}}$, all $\mathbf{f}^\ell_q \in \mathbb{Z}^{1 \times n_2}_p$, $\mu_q,l \in \mathbb{Z}_p$, all $q^* \in [m]$ and $i^* \in [L_q]$; all $\text{crs} \leftarrow \text{Setup}(1^\lambda, 1^m, 1^{L_i}, \ldots, 1^{L_m}, 1^{n_1}, 1^{n_2}, \mathbf{M}; \{\mathbf{f}^\ell_q\}_{\ell \in \{L_q\}}, \{\mu_q,l\}_{l \in \mathcal{M}^*_q \cup C^*_q} \}_{q \in \{m\}}$.
all \((pk_{q^*, i}, sk_{q^*, i}) \leftarrow \text{Gen}(\overline{crs}, q^*, i^*)\); all \(\{pk_{q^*, i}\}_{i \in [L_{q^*}]} \setminus \{i^*\}\) such that \(\text{Ver}(\overline{crs}, q^*, i, pk_{q^*, i}) = 1\); all \(x \in \mathbb{Z}_p^{1 \times n_1}\) and \(f_{q^*, i} \in \mathbb{Z}_p^{1 \times n_2}\); we have:

\[
\begin{align*}
\Pr \left[ \text{Dec}(sk_{q^*, i^*}, hsk_{q^*, i^*}, (ct_+, ct_{q^*})) = xM_{f_{q^*, i^*}}^l \right.
&= \text{mpk}_* \leftarrow \text{Agg}_* (\overline{crs}); \\
&= (\text{mpk}_*, (hsk_{q^*, i^*})_{i \in [L_{q^*}]} \leftarrow \text{Agg}(\overline{crs}, q^*, (pk_{q^*, i^*}, f_{q^*, i} \in [L_{q^*}]) \\
&= s \leftarrow \text{Coin}; ct_+ \leftarrow \text{Enc}_*(\text{mpk}_*, x; s); ct_{q^*} \leftarrow \text{Enc}(\text{mpk}_{q^*}; s) \\
\right. = 1.
\end{align*}
\]

This follows from the fact that the analysis of correctness in Section 6.1: for \(s \in \{1, 2\}\), we have:

\[
\overline{x} = (x\|0_n), \quad \overline{f}_{q^*, i^*} = (f_{q^*, i^*}\|1)
\]

and

\[
[M_{q^*, i^*}]_s = \left[ \begin{array}{c}
\text{M} \\
0_{n \times n_1}
\end{array} \right] \quad \text{where} \quad [ict_{q^*, i^*}]_s = \begin{cases}
\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, 0) & \text{if } i^* \in [L_{q^*}] \setminus (M_{q^*}^* \cup C_{q^*}^*) \\
\text{Enc}_1([\text{ipk}]_1, [\text{ipk}]_2, \mu_{q^*, i^*}) & \text{if } i^* \in M_{q^*}^* \cup C_{q^*}^*
\end{cases}
\]

with \(([\text{ipk}]_1, [\text{ipk}]_2) \in \text{Gen}_1(1^k)\). And for all \(s \leftarrow \mathbb{Z}_p^{1 \times k}\), we have

\[
\text{sk}_{q^*, i^*} = U_{q^*, i^*},
\]

\[
(\text{ct}_+, ct_{q^*}) = \left( \begin{array}{c}
sA \cdot sAW + \overline{x}, \\
\sum_{i \in [L_{q^*}]} (sT_{q^*, i} + sAW_{q^*, i}(M_{q^*, i}f_{q^*, i} \otimes I_{k+i+1}))
\end{array} \right)
\]

\[
\text{hsk}_{q^*, i^*} = \left( \begin{array}{c}
B_{q^*}f_{q^*, i^*}, \\
\sum_{i \in [L_{q^*}] \setminus \{i^*\}} (h_{q^*, i, i^*} + W_{q^*, i}(M_{q^*, i}f_{q^*, i} \otimes B_{q^*}r_{q^*, i}^-)), \\
W_{q^*, i^*}(M_{q^*, i^*}f_{q^*, i^*} \otimes B_{q^*}r_{q^*, i}^-) + WM_{q^*, i^*}f_{q^*, i^*}, \\
\sum_{i \in [L_{q^*}] \setminus \{i^*\}} (sAh_{q^*, i, i^*} + sAW_{q^*, i}(M_{q^*, i}f_{q^*, i} \otimes B_{q^*}r_{q^*, i}^-))
\end{array} \right)
\]

where

\[
A_{h_{q^*, i, i^*}} = T_{q^*, i}B_{q^*}f_{q^*, i^*}, \quad \forall i \in [L_{q^*} \setminus \{i^*\} \quad \text{and} \quad A_{U_{q^*, i^*}} = T_{q^*, i^*}.
\]

Note that here we actually consider \(\text{hsk}_{q^*, j}\) for \(j = i^*\) and \(\text{sk}_{q^*, i} for i = i^*\) and all above equalities are ensured by \(\text{Ver}\) and \(\text{Gen}\). We have

\[
\begin{align*}
z_1 &= \sum_{i \in [L_{q^*}]} (sT_{q^*, i}B_{q^*}f_{q^*, i^*}^- + sAW_{q^*, i}(M_{q^*, i}f_{q^*, i}^- \otimes I_{k+i+1})B_{q^*}r_{q^*, i^-}) \\
&= \sum_{i \in [L_{q^*}]} (sT_{q^*, i}B_{q^*}f_{q^*, i^-}^- + sAW_{q^*, i}(M_{q^*, i}f_{q^*, i^-}^- \otimes B_{q^*}r_{q^*, i^-})) \\
z_2 &= \sum_{i \in [L_{q^*}] \setminus \{i^*\}} (sAh_{q^*, i, i^*}^- + sAW_{q^*, i}(M_{q^*, i}f_{q^*, i^-} \otimes B_{q^*}r_{q^*, i^-})) \\
z_3 &= sAU_{q^*, i^-}B_{q^*}f_{q^*, i^-}, \\
z_4 &= sAW_{q^*, i^-}(M_{q^*, i^-}f_{q^*, i^-} \otimes B_{q^*}r_{q^*, i^-}) + sAW_{q^*, i^-}f_{q^*, i^-}, \\
z_5 &= sAW_{q^*, i^-}f_{q^*, i^-} \otimes \overline{x}M_{q^*, i^-}f_{q^*, i^-}
\end{align*}
\]
and then

\[
Z = Z_1 - Z_2 - Z_3 - Z_4 + Z_5 \\
= sT_{q,i}B_{q,i}f_{q,i} + sAW_{q,i}(\tilde{M}_{q,i}f_{q,i} \odot B_{q,i}r_{q,i}^*) - sAU_{q,i}B_{q,i}r_{q,i}^* \\
- (sAW_{q,i}(\tilde{M}_{q,i}f_{q,i} \odot B_{q,i}r_{q,i}^*) + sAW_{q,i}(\tilde{M}_{q,i}f_{q,i}^*) \\
+ (sAW_{q,i}(\tilde{M}_{q,i}f_{q,i}^*) + \tilde{x}_{q,i}f_{q,i}^*) \\
= (x||0_n)
\begin{pmatrix}
\tilde{M} & 0_{n_1} \\
0_{n \times n_2} & \text{ict}_{q,i}^*
\end{pmatrix}
\begin{pmatrix}
f_{q,i} \\
1
\end{pmatrix}
= xM_{q,i}^f.
\]

Here, equalities hold analogous to the analysis of correctness in Section 6.1, and the last equality holds even if we replace \([\tilde{M}_{q,i}^f]_s\) with

\[
[\tilde{M}_{q,i}^f]_s = \begin{bmatrix}
M & 0_{n_1} \\
0_{n \times n_2} & \text{ict}_{q,i}^*
\end{bmatrix}_s,
\]

where \([\text{ict}_{q,i}^*]_s\) is defined in Section 7.1. For all \(\lambda, m, n_1, n_2 \in \mathbb{N}\), all \(L_1, \ldots, L_m \in \mathbb{N}\), all \(\{\tilde{M}_{q,i}^f\}_{q \in [m]}\), \(\{f_{q,i}^*\}_{i \in [L_q]}\), \(\{\mu_{q,i}\}_{i \in [L_q]}\) and \(\{\tilde{M}_{q,i}^f\}_{q \in [m]}\),

\[
\tilde{crs} \leftarrow \text{Setup}(1^n, 1^m, 1^{l_1}, \ldots, 1^{l_n}, 1^{n_1}, 1^{n_2}, \{\{f_{q,i}^*\}_{i \in [L_q]}, \{\mu_{q,i}\}_{i \in [L_q]}, \{\text{ict}_{q,i}^*\}_{q \in [m]}\}),
\]

all \((pk_{q,i}^f, sk_{q,i}^f) \leftarrow \text{Gen}(\tilde{crs}, q^*, i^*)\); all \(\{pk_{q,i}^f\}_{i \in [L_q]}, \{sk_{q,i}^f\}_{i \in [L_q]}\) such that \(\text{Ver}(\tilde{crs}, q^*, i, pk_{q,i}^f) = 1\); all \(x_1 \in \mathbb{Z}_p^{1 \times n_1}\), \(x_2 \in \mathbb{Z}_p^{1 \times n_2}\),

\[
\text{Pr}_{\text{Dec}(sk_{q,i}^f, hsk_{q,i}^f, (ct_*, ct_{q^*})) = (x_1 \otimes x_2)f_{q,i}^*} \begin{bmatrix}
\text{mpk}_* & \text{Agg}_*(\tilde{crs}) \\
\text{mpk}_*, (hsk_{q,i}^f)_{i \in [L_q]} & \text{Agg}(\tilde{crs}, q^*, \{pk_{q,i}^f\}_{i \in [L_q]})
\end{bmatrix}_{s \leftarrow \text{Coin}; ct_* \leftarrow \text{Enc}_*(\text{mpk}_*, (x_1, x_2); s); ct_{q^*} \leftarrow \text{Enc}(\text{mpk}_{q^*}; s)} = 1.
\]

This follows from the analysis of correctness in Section 7.1 and the fact that the simulator of our multi-instance slotted PReg-IPFE (\text{iSetup}, iGen, iEnc, iDec) can pass the sanity check as shown in Appendix D.1.

D.2 Sanity Check of the simulator in Section 7.2

In this section, we show that the simulator of our multi-instance slotted Reg-QFE can pass the sanity check. The simulated \(\tilde{crs}\) has the full capacity as the crs of the scheme in Section 7.1. For all \(\lambda, m, n_1, n_2 \in \mathbb{N}\), all \(L_1, \ldots, L_m \in \mathbb{N}\), all \(\{\tilde{M}_{q,i}^f\}_{q \in [m]}\), \(\{f_{q,i}^*\}_{i \in [L_q]}\), \(\{\mu_{q,i}\}_{i \in [L_q]}\), \(\{\text{ict}_{q,i}^*\}_{q \in [m]}\),

\[
\tilde{crs} \leftarrow \text{Setup}(1^n, 1^m, 1^{l_1}, \ldots, 1^{l_n}, 1^{n_1}, 1^{n_2}, \{\{f_{q,i}^*\}_{i \in [L_q]}, \{\mu_{q,i}\}_{i \in [L_q]}, \{\text{ict}_{q,i}^*\}_{q \in [m]}\}),
\]

all \((pk_{q,i}^f, sk_{q,i}^f, ct_{q^*}) \leftarrow \text{Gen}(\tilde{crs}, q^*, i^*)\); all \(\{pk_{q,i}^f\}_{i \in [L_q]}\) such that \(\text{Ver}(\tilde{crs}, q^*, i, pk_{q,i}^f) = 1\); all \(x_1 \in \mathbb{Z}_p^{1 \times n_1}\), \(x_2 \in \mathbb{Z}_p^{1 \times n_2}\),

\[
\text{Pr}_{\text{Dec}(sk_{q,i}^f, hsk_{q,i}^f, (ct_*, ct_{q^*})) = (x_1 \otimes x_2)f_{q,i}^*} \begin{bmatrix}
\text{mpk}_* & \text{Agg}_*(\tilde{crs}) \\
\text{mpk}_*, (hsk_{q,i}^f)_{i \in [L_q]} & \text{Agg}(\tilde{crs}, q^*, \{pk_{q,i}^f\}_{i \in [L_q]})
\end{bmatrix}_{s \leftarrow \text{Coin}; ct_* \leftarrow \text{Enc}_*(\text{mpk}_*, (x_1, x_2); s); ct_{q^*} \leftarrow \text{Enc}(\text{mpk}_{q^*}; s)} = 1.
\]

This follows from the analysis of correctness in Section 7.1 and the fact that the simulator of our multi-instance slotted PReg-IPFE (\text{iSetup}, iGen, iEnc, iDec) can pass the sanity check as shown in Appendix D.1.

D.3 Sanity Check of the simulator in Section 5.3

In this section, we show that when apply our multi-instance slotted Reg-QFE in Section 7.1 to the transformation in Section 5.2, the simulator of the compact Reg-QFE can pass the sanity check. The simulated \(\tilde{crs}\) has the full capacity as the crs. For all \(L \in \mathbb{N}\), all \(f_{i}^* \in F, \mu_i \in Z\) and all \(CH, \tilde{CH}\) such that \(CH, \tilde{CH} \subseteq [0, L’ - 1]\), \(CH \cup \tilde{CH} = [0, L’ - 1]\) for some \(L’ \leq L\). For all stateful adversary \(\mathcal{A}\) making a polynomial number of oracle queries (defined as in Section 2.2) and all \(L\), we have the following advantage function is negligible in \(\lambda\):

\[
\text{Pr}_{b = 1} = \text{Setup}(1^n, 1^l, F; \{f_{i}^*\}_{i \in [L]}, \{\mu_{i}\}_{i \in [L]}); b = 0; \mathcal{A}(\text{ORegNT}(\cdot)), \text{ORegT}(\cdot), \text{ORenc}(\cdot), \text{Odec}(\cdot)(\tilde{crs}))
\]

we recall that oracles work as follows with \(\text{aux} = 1, \text{E} = 0, \mathcal{R} = 0\) and \(t = 1\):
- ORegNT(pk, f): run (mpk, aux’) ← Reg(\(\tilde{c}r_s\), aux, pk, f), update aux = aux’, append (mpk, aux) to \(\mathcal{R}\) and return (|\(\mathcal{R}\)|, mpk, aux);
- ORegT(\(f^*\)): run (pk’, sk’) ← Gen(\(\tilde{c}r_s\), aux), (mpk, aux’) ← Reg(\(\tilde{c}r_s\), aux, pk’, \(f^*\)), update aux = aux’, compute hsk’ ← Upd(\(\tilde{c}r_s\), aux, pk’), append (mpk, aux) to \(\mathcal{R}\), return (t = |\(\mathcal{R}\)|, mpk, aux, pk’, sk’, hsk’);
- OEnc(i, x): let \(\mathcal{R}[i] = (mpk, \star)\), run ct ← Enc(mpk, x), append (x, ct) to \(\mathcal{E}\) and return (|\(\mathcal{E}\)|, ct);
- ODec(j): let \(\mathcal{E}[j] = (x_j, ct_j)\), compute \(z_j \leftarrow \text{Dec}(sk^*, hsk^*, ct_j)\); if \(z_j = \text{getupd}\), run hsk’ ← Upd(\(\tilde{c}r_s\), aux, pk’)
  and recompute \(z_j \leftarrow \text{Dec}(sk^*, hsk^*, ct_j)\). Set \(b = 1\) when \(z_j \neq f^*(x_j)\).

with the following restrictions:

- there are at most \(L - 1\) queries to ORegNT and there is exactly one query to ORegT; therefore, we will consider \(f^*, pk^*, sk^*, hsk^*\) to be global;
- for query \((i, x)\) to OEnc, it holds that \(i \geq t, \mathcal{R}[i] \neq \bot\);
- for query \((j)\) to ODec, it holds that \(\mathcal{E}[j] \neq \bot\).

This follows from the analysis of correctness in Section 5.2, and the fact that the simulator of our multi-instance slotted Reg-QFE (\(\tilde{m}\)Setup, \(m\)Gen, \(m\)Enc\(_c\), \(m\)Enc) can pass the sanity check as shown in Appendix D.2.