Small Field Attack, and Revisiting RLWE-Based Authenticated Key Exchange from Eurocrypt'15[⋆]

Abstract. Authenticated key exchange (AKE) plays a fundamental role in modern cryptography. Up to now, the HMQV protocol family is among the most efficient provably secure AKE protocols, which has been widely standardized and in use. Given recent advances in quantum computing, it would be highly desirable to develop lattice-based analogue of HMQV protocols for the possible upcoming post-quantum era. Towards this goal, an important step was recently made by Zhang et al. at Eurocrypt'15 [ZZD$^+$15,ZZDS14]. Similar to HMQV, this ring-LWE based analogue of HMQV proposed there consist of two variants: a two-pass protocol Π_2, as well as a one-pass protocol Π_1 that implies a signcryption scheme (named as "deniable encryption" in [ZZD$^+$15]). All these protocols are claimed to be provably secure under the ring-LWE (RLWE) assumption.

In this work, we propose a new type of attack, referred to as *small field attack* (SFA), against the one-pass protocol Π_1 as well as its resultant deniable encryption scheme. With SFA, a malicious user can efficiently recover the static private key of the honest victim user in Π_1 with overwhelming probability. Moreover, the SFA attack is *realistic and powerful in practice*, in the sense that it is hard in practice for the honest victim party to prevent, *or even detect*, the attack. Besides, some new property regarding the CRT basis for $\mathcal{R}_q$ is also developed in this work, which is *essential* for our small field attack and may be of independent interest.

The security proof of the two-pass protocol Π_2 is then revisited. We are stunk at Claim 16 in [ZZDS14], with a gap identified and discussed in the security proof. To us, we do not know how to fix the gap, which traces back to some critical differences between the security proof of HMQV and that of its RLWE-based analogue.

1 Introduction

Authenticated key exchange (AKE) plays a fundamental role in modern cryptography. Up to now, the HMQV protocol family [LMQ$^+$03,Kra05] is generally considered to be the most efficient provably secure AKE protocol family, and has been standardized and widely in use. HMQV is built upon Diffie-Hellman (DH) [DH76], and consists of two variants: two-pass HMQV with provable security in the Canetti-Kraczye (CK) model [CK01], and one-pass HMQV with provable security in a tailored CK model for one-pass AKE [HK11]. Although two-pass HMQV is more frequently used in practice, one-pass HMQV itself is of great value and has many applications as well. For example, it was shown in [HK11] that one-pass HMQV implies secure higncryption (aka the *deniable encryption* in [ZZDS14,ZZD$^+$15]), and has natural applications to key wrapping. However, HMQV will become insecure in the possible upcoming post-quantum era. Consequently, it would be much desirable to develop the HMQV-analogue based on lattice problems, since lattice-based cryptosystems are commonly believed to be resistant to quantum attacks.

As a ring variant of the learning with errors (LWE) problem [Reg09], the ring-LWE (RLWE) problem [LPR13a] was introduced to resolve some inefficiency issues of LWE-based cryptosystems. It is versatile, and has been well-studied ever since its introduction. Each ring-LWE instance is parameterized by a positive integer n, a positive rational prime q, a parameter $\alpha > 0$, and a monic irreducible polynomial $f(x) \in \mathbb{Q}[x]$ of degree n. The hardness of ring-LWE problem is captured by a (possibly) quantum reduction from the approximate SVP problem in any ideal lattice of K to the decisional ring-LWE problem [LPR13a], where $K \cong \mathbb{Q}[x]/\langle f \rangle$ denotes a number field of degree n. Ever since its introduction, significant cryptographic progress based on the RLWE problem has been made, *e.g.*, [LPR13a,LPR13b,Pei14,LS15,SS11,Gen09,GGH13a,GGH$^+$13b,Lyu12,DDLL13]; For an excellent survey, the reader is referred to [Pei16].

In this work, the term "ring-LWE problem" refers to a *special* case of the *original* ring-LWE problem that is *widely used* in practice, *i.e.*, $n \geq 16$ is a power-of-two, q is a positive rational prime such that $q \equiv 1 \pmod{2n}$, and $f(x) = \Phi_{2n}(x) \in \mathbb{Z}[x]$ is the $2n$-th cyclotomic polynomial. For this (specific) ring-LWE problem, its search and decisional variants can be proven *computationally equivalent* under mild constraints on the parameters [LPR13a,DD12].

RLWE-based Diffie-Hellman and HMQV We briefly review the abstract basic structure of key exchange over RLWE [ZZDS14,ZZD$^+$15,DXL12,Pei14,BCNS15,ADPS16]. Let Alice and Bob denote the two involved parties for simplicity. To Alice (party i), the static private key is $(\boldsymbol{s}_i, \boldsymbol{e}_i) \in \mathcal{R}_q \times \mathcal{R}_q$ (both are"small"), the static public

[⋆] Communications with the corresponding authors of [ZZD$^+$15,ZZDS14] are briefly summarized in Section A.

key is $\boldsymbol{p}_i = \boldsymbol{a}\boldsymbol{s}_i + c \cdot \boldsymbol{e}_i$, where $\boldsymbol{a} \in \mathcal{R}_q$ denotes the public system parameter; the ephemeral private key is $(\boldsymbol{r}_i, \boldsymbol{f}_i) \in \mathcal{R}_q \times \mathcal{R}_q$ (both are "small"), and the ephemeral public key is $\boldsymbol{x}_i = \boldsymbol{a}\boldsymbol{r}_i + c \cdot \boldsymbol{f}_i$. Similar notations $(\boldsymbol{s}_j, \boldsymbol{e}_j), \boldsymbol{p}_j, (\boldsymbol{r}_j, \boldsymbol{f}_j), \boldsymbol{x}_j$ apply to Bob (party j). The value $c \in \mathbb{F}_q^\times$ is a public constant, which is set to be 1 in [Pei14] and to be 2 in [DXL12].

For the basic KE protocol without entity authentication (*i.e.*, Alice and Bob do not necessarily possess the static public/private keys), first Alice sends $\boldsymbol{x}_i$ to Bob; Then Bob replies to Alice with $(\boldsymbol{x}_j, \boldsymbol{w}_j = g(\boldsymbol{x}_i, \boldsymbol{r}_j, \boldsymbol{f}_j))$, where g denotes a probabilistic polynomial-time (PPT) signal-generation function; Finally, Alice (resp., Bob) applies a key derivation function denoted K_i (resp., K_j), such that $K_i(\boldsymbol{x}_j, \boldsymbol{r}_i, \boldsymbol{f}_i, \boldsymbol{w}_j) = K_j(\boldsymbol{x}_i, \boldsymbol{r}_j, \boldsymbol{f}_j, \boldsymbol{w}_j)$. Briefly speaking, the underlying key derivation mechanism is based on the bilinear map $(\boldsymbol{r}_1, \boldsymbol{r}_2) \rightarrow \boldsymbol{d} = \boldsymbol{r}_1 \cdot \boldsymbol{a} \cdot \boldsymbol{r}_2$ of $\mathcal{R}_q \times \mathcal{R}_q$ into $\mathcal{R}_q$, where $\boldsymbol{d} \in \mathcal{R}_q$ denotes the dominant value from which the session-key is derived. However, neither Alice nor Bob could *directly* compute the dominant value $\boldsymbol{d}$ due to the small noises $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$ involved, and the tricky part of the key derivation functions K_i and K_j is to reach the consensus on the shared key from two values that are "close" to the dominant value $\boldsymbol{d}$. This basic key exchange protocol is thus analogous to Diffie-Hellman, which in turn is based on bilinear map over cyclic group.[1] In two-pass RLWE-based HMQV analogue, we have $\boldsymbol{w}_j = g(\boldsymbol{p}_i, \boldsymbol{x}_i, \boldsymbol{s}_j, \boldsymbol{e}_j, \boldsymbol{r}_j, \boldsymbol{f}_j)$, and $K_i(\boldsymbol{p}_j, \boldsymbol{x}_j, \boldsymbol{s}_i, \boldsymbol{e}_i, \boldsymbol{r}_i, \boldsymbol{f}_i, \boldsymbol{w}_j) = K_j(\boldsymbol{p}_i, \boldsymbol{x}_i, \boldsymbol{s}_j, \boldsymbol{e}_j, \boldsymbol{r}_j, \boldsymbol{f}_j, \boldsymbol{w}_j)$. Conversely, in one-pass RLWE-based HMQV analogue where the values $(\boldsymbol{x}_j, \boldsymbol{p}_j)$ sent by Bob in the second round is waived, Alice sends $\boldsymbol{x}_i$ as well as the signal $\boldsymbol{w}_i = g(\boldsymbol{p}_j, \boldsymbol{s}_i, \boldsymbol{e}_i, \boldsymbol{r}_i, \boldsymbol{f}_i)$; In this case, K_i and K_j are defined to be: $K_i(\boldsymbol{p}_j, \boldsymbol{s}_i, \boldsymbol{e}_i, \boldsymbol{r}_i, \boldsymbol{f}_i, \boldsymbol{w}_i) = K_j(\boldsymbol{p}_i, \boldsymbol{x}_i, \boldsymbol{s}_j, \boldsymbol{e}_j, \boldsymbol{w}_i)$.

An important step towards constructing the RLWE-based analogue of HMQV was recently made at Eurocrypt'15 [ZZD+15] and in its full version [ZZDS14], where the two-pass protocol Π_2 and one-pass protocol Π_1 were proposed. Both Π_2 and Π_1 are claimed to be provably secure under the ring-LWE assumption in the random oracle model, relative to a variant of the Bellare-Rogaway model [BR93] where adversary is not allowed to register public keys on behalf of dishonest users. In particular, for the one-pass variant Π_1, it is claimed to be provably secure "*in a weak model similar to [Kra05] which avoids some reasonable insufficiencies for one-pass protocol*" ([ZZD+15], page 744). However, the exact security model for Π_1 is not made clear in [ZZD+15,ZZDS14], and the actual proof is omitted there. Similar to that one-pass HMQV implies signcryption, the work [ZZD+15,ZZDS14] also describes the signcryption scheme (*i.e.*, the "deniable encryption" in [ZZD+15,ZZDS14]) resultant from the one-pass variant Π_1, where the derived session-key is used for a CPA-secure symmetric encryption and a MAC scheme. The resultant signcryption is also claimed to be CCA-secure in [ZZD+15,ZZDS14], by following the analogue to one-pass HMQV.

The two-pass protocol Π_2 is presented in Section 7. Below, we briefly review Π_1 in Figure 4.1, in accordance with our abstract protocol structure above.

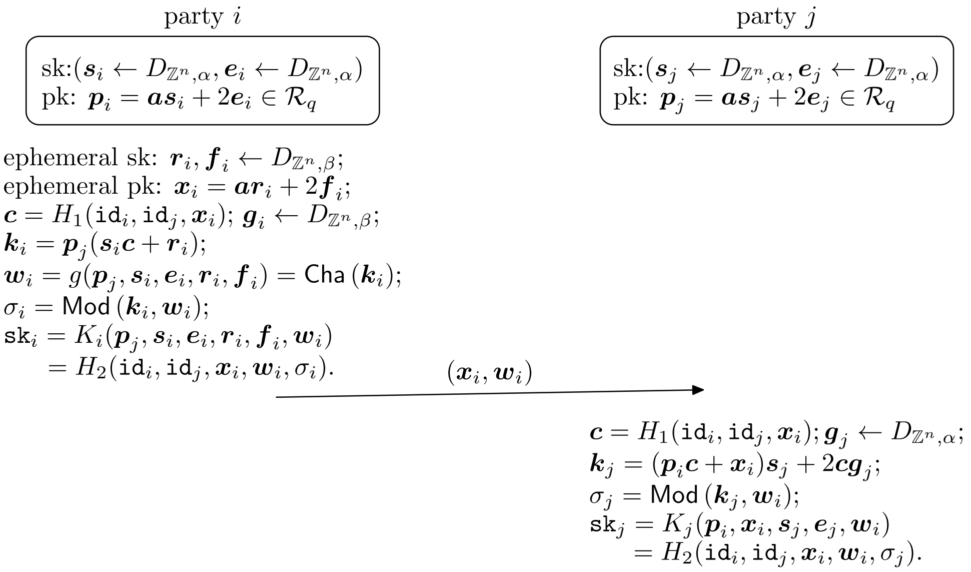
The one-pass AKE scheme Π_1 The scheme Π_1, proposed in [ZZDS14,ZZD+15], is built upon the ring-LWE assumption. In Π_1, $\boldsymbol{a} \leftarrow \mathcal{R}_q$ is the global public parameter, and $M > 0$ is a constant that is sufficiently large. As a two-party AKE protocol, users in Π_1 are represented by party i and party j. For party i: the static private key is $(\boldsymbol{s}_i, \boldsymbol{e}_i)$, where $\boldsymbol{s}_i, \boldsymbol{e}_i \leftarrow D_{\mathbb{Z}^n,\alpha}$; Its associated public key is $\boldsymbol{p}_i \triangleq \boldsymbol{a}\boldsymbol{s}_i + 2\boldsymbol{e}_i \in \mathcal{R}_q$; And its identity issued by the Certificate Authority (CA) is $\mathtt{id}_i$. Similar notations, $\boldsymbol{s}_j, \boldsymbol{e}_j \leftarrow D_{\mathbb{Z}^n,\alpha}, \boldsymbol{p}_j \triangleq \boldsymbol{a}\boldsymbol{s}_j + 2\boldsymbol{e}_j \in \mathcal{R}_q$ and $\mathtt{id}_j$, apply to party j. Let $H_1 : \{0,1\}^* \rightarrow D_{\mathbb{Z}^n,\gamma}$ be a hash function that outputs invertible elements in $\mathcal{R}_q$, and $H_2 : \{0,1\}^* \rightarrow \{0,1\}^*$ be the key derivation function. Both H_1 and H_2 are regarded as random oracles in Π_1.

The following functions are essential for the definition of Π_1. First, When the prime q is clear from the context, define the function $\mathsf{Parity} : \mathbb{F}_q \rightarrow \{0,1\}$, where $\mathsf{Parity}\,(u) \triangleq u(\bmod 2) \in \{0,1\}$. Moreover, we define the function $\mathsf{Mod} : \mathbb{F}_q \times \{0,1\} \rightarrow \{0,1\}$, where $\mathsf{Mod}\,(u,w) \triangleq \mathsf{Parity}\,((u + w \cdot q_0) \bmod q) \in \{0,1\}$. Finally, define the function $\mathsf{Cha} : \mathbb{F}_q \rightarrow \{0,1\}$, such that $\mathsf{Cha}\,(u) = 0$ if and only if $u \in \left\{-\frac{q-1}{4}, \cdots, \frac{q-1}{4}\right\}$; Otherwise, $\mathsf{Cha}\,(u) = 1$. All these functions could be easily generalized to the n-dimensional case *in the component-wise manner*; For instance, $\mathsf{Cha}\,(\boldsymbol{v}) \triangleq [\mathsf{Cha}\,(v_j)]_{j\in[n]} \in \{0,1\}^n$ for every $\boldsymbol{v} = [v_j]_{j\in[n]} \in \mathbb{F}_q^n$, and it is understood that $\mathsf{Cha}\,(\boldsymbol{u}) \triangleq \mathsf{Cha}\left([u_j]_{j\in[n]}\right)$ for $\boldsymbol{u} = \sum_{j\in[n]} u_j\zeta^{j-1} \in \mathcal{R}_q$.

1.1 Our Contributions

In this work, we propose a new type of efficient attack, referred to as *small field attack* (SFA), against the one-pass protocol Π_1, as well as its resultant "deniable encryption" proposed in [ZZDS14,ZZD+15]. With SFA, a malicious user i can efficiently recover the static private key of the honest user j in Π_1 with overwhelming

[1] Unlike traditional DH protocol where a key pair may be catched for a short time to improve performance, for RLWE-based DH-analogue it is crucial that both parties use fresh ephemeral private keys in each session [Flu16,ADPS16].



party i
sk:$(\boldsymbol{s}_i \leftarrow D_{\mathbb{Z}^n,\alpha}, \boldsymbol{e}_i \leftarrow D_{\mathbb{Z}^n,\alpha})$
pk: $\boldsymbol{p}_i = \boldsymbol{a}\boldsymbol{s}_i + 2\boldsymbol{e}_i \in \mathcal{R}_q$
party j
sk:$(\boldsymbol{s}_j \leftarrow D_{\mathbb{Z}^n,\alpha}, \boldsymbol{e}_j \leftarrow D_{\mathbb{Z}^n,\alpha})$
pk: $\boldsymbol{p}_j = \boldsymbol{a}\boldsymbol{s}_j + 2\boldsymbol{e}_j \in \mathcal{R}_q$
ephemeral sk: $\boldsymbol{r}_i, \boldsymbol{f}_i \leftarrow D_{\mathbb{Z}^n,\beta}$;
ephemeral pk: $\boldsymbol{x}_i = \boldsymbol{a}\boldsymbol{r}_i + 2\boldsymbol{f}_i$;
$\boldsymbol{c} = H_1(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i)$; $\boldsymbol{g}_i \leftarrow D_{\mathbb{Z}^n,\beta}$;
$\boldsymbol{k}_i = \boldsymbol{p}_j(\boldsymbol{s}_i\boldsymbol{c} + \boldsymbol{r}_i)$;
$\boldsymbol{w}_i = g(\boldsymbol{p}_j, \boldsymbol{s}_i, \boldsymbol{e}_i, \boldsymbol{r}_i, \boldsymbol{f}_i) = \mathsf{Cha}\,(\boldsymbol{k}_i)$;
$\sigma_i = \mathsf{Mod}\,(\boldsymbol{k}_i, \boldsymbol{w}_i)$;
$\mathtt{sk}_i = K_i(\boldsymbol{p}_j, \boldsymbol{s}_i, \boldsymbol{e}_i, \boldsymbol{r}_i, \boldsymbol{f}_i, \boldsymbol{w}_i)$
$= H_2(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{w}_i, \sigma_i)$.
$(\boldsymbol{x}_i, \boldsymbol{w}_i)$
$\boldsymbol{c} = H_1(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i)$; $\boldsymbol{g}_j \leftarrow D_{\mathbb{Z}^n,\alpha}$;
$\boldsymbol{k}_j = (\boldsymbol{p}_i\boldsymbol{c} + \boldsymbol{x}_i)\boldsymbol{s}_j + 2\boldsymbol{c}\boldsymbol{g}_j$;
$\sigma_j = \mathsf{Mod}\,(\boldsymbol{k}_j, \boldsymbol{w}_i)$;
$\mathtt{sk}_j = K_j(\boldsymbol{p}_i, \boldsymbol{x}_i, \boldsymbol{s}_j, \boldsymbol{e}_j, \boldsymbol{w}_i)$
$= H_2(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{w}_i, \sigma_j)$.

Fig. 1. Description of Π_1. Note that in the full description in [ZZDS14,ZZD$^+$15], party i applies the rejection sampling operation to generate a "good" $(\boldsymbol{x}_i, \boldsymbol{w}_i)$ pair, which does not affect SFA and is omitted here for simplicity.

probability, after issuing a set of "random-looking" session queries with party j. The SFA attack is *realistic and powerful in practice*, in the sense that it is *hard in practice* for the honest party j to prevent, *or even detect*, the attack.[2] Besides, we also develop in this work a new property, *i.e.*, Lemma 5, regarding the CRT basis for $\mathcal{R}_q$, which is *essential* for our small field attack against Π_1 and may be of independent interest.

We also notice that the SFA attack may not violate the security claim for Π_1 made in [ZZD$^+$15], as with SFA the malicious user needs to register its public key on its own (while in the security model of [ZZD$^+$15], the adversary may not be allowed to register public keys on behalf of dishonest users). Nevertheless, in our SFA, it is hard to distinguish the public key registered by a malicious user and the public key honestly generated. Moreover, as the malicious user does know the private key corresponding to the registered public key, traditional mechanisms for proof-of-knowledge (POK) or proof-of-possession (POP) of private key, *e.g.*, via requiring the user to sign a random message with the registered public key, does not prevent our SFA attack. From our point of view, forbidding adversary from registering public keys on behalf of dishonest users, on the one hand, seems to be unrealistic in practice; And, on the other hand, it may result in over weak security model as naturally insecure protocol like Π_1 could be proved "secure" within.

Then, the security analysis of the two-pass AKE scheme Π_2 in [ZZDS14,ZZD$^+$15] is revisited. Loosely speaking, the security proof of Π_2 considers five types of adversaries, Type-I through Type-V; Only Type-I is analyzed in [ZZD$^+$15], and the complete analysis is presented in the full version [ZZDS14]. We are stuck at Claim 16 in [ZZDS14], which deals with Type-II adversary impersonating an honest user in the test-session without a matching session. A different conclusion on Claim 16 is reached, with a gap in the security proof identified and discussed. To us, we do not know how to fix the gap, which traces back to some *critical differences between the security proof of HMQV and that of its RLWE-based analogue*. Details are referred to Section 7.

Roughly speaking, given a pair of views $(\mathsf{view}_1, \mathsf{view}_2)$ of a PPT adversary $\mathcal{A}$, consider the ability that $\mathcal{A}$ could successfully output a value σ_t, $t \in \{1, 2\}$, where σ_2 is a random value independent of view_2 but σ_1 is essentially committed to view_1 and is infeasible to be efficiently computed from view_1. The corresponding author of Claim 16 [ZZDS14] concludes that: if view_1 and view_2 are computationally indistinguishable and $\mathcal{A}$ could not output σ_2 from view_2 with non-negligible probability, then $\mathcal{A}$ should also not be able to output σ_1 from view_1 with non-negligible probability; In particular, whether σ_1 is distinguishable from σ_2 does not affect the conclusion. However,

[2] SFA works, in general, against the abstract structure of one-pass AKE discussed in Introduction, where both static private keys and ephemeral private keys get mixed in generating the key material from which the session key is derived. But it does not work against the analogues of RLWE-based Diffie-Hellman or SIGMA/TLS [DXL12,Pei14,BCNS15,ADPS16], where only ephemeral private keys are involved in session key generation.

from our view, to get the conclusion, we need to prove that the joint distribution of $(\mathsf{view}_1, \sigma_1)$ is computationally indistinguishable from that of $(\mathsf{view}_2, \sigma_2)$; At least we need to argue the indistinguishability between σ_1 and σ_2, as the adversarial event is defined not only on the views $(\mathsf{view}_1, \mathsf{view}_2)$ but also on the hidden values (σ_1, σ_2). We do not know how to fix the gap. Details are referred to Section 7.

1.2 Outline of Small Field Attack

Here we present the outline of SFA, the main technical contribution of this work.

The CRT basis for $\mathcal{R}_q$ and its new property Before introducing small field attack, we stress that our SFA makes full use of the notion of the CRT basis (for $\mathcal{R}_q$) first proposed in [LPR13a]. Its basic properties can be summarized as follows. First, the CRT basis for $\mathcal{R}_q$ is *unique*, and could be found efficiently [LPR13a]. Furthermore, the CRT basis $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ is an $\mathbb{F}_q$-basis for $\mathcal{R}_q$, when $\mathcal{R}_q$ is seen as an $\mathbb{F}_q$*-module* of rank n in the natural way; For instance, every element $\boldsymbol{u} \in \mathcal{R}_q$ could be *uniquely* written as $\boldsymbol{u} = \sum_{i\in[n]} u_i \boldsymbol{c}_i, u_i \in \mathbb{F}_q$; For simplicity, in this work let $\eta_i(\boldsymbol{u}) \triangleq u_i \in \mathbb{F}_q$ denote the i-th CRT-coefficient of $\boldsymbol{u} \in \mathcal{R}_q$, and let $\mathsf{Dim}\,(\boldsymbol{u}) \triangleq \{i \in [n] \mid \eta_i(\boldsymbol{u}) \neq 0\}$. Finally, the equality $\boldsymbol{uv} = \sum_{i\in[n]} \eta_i(\boldsymbol{u})\eta_i(\boldsymbol{v}) \cdot \boldsymbol{c}_i$ holds in the *ring* $\mathcal{R}_q$ for every $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{R}_q$.

In addition to these basic ones, some interesting property regarding $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ is further developed in this work. To be precise, by assuming $\boldsymbol{c}_i = \sum_{j\in[n]} c_{i,j}\zeta^{j-1} \in \mathcal{R}_q, c_{i,j} \in \mathbb{F}_q$ for every $i \in [n]$, we shall prove in Lemma 5, Section 3.2, that for each i, the coefficients $c_{i,1}, c_{i,2}, \cdots, c_{i,n} \in \mathbb{F}_q$ form a geometric sequence (in $\mathbb{F}_q$). This new property is essential for our SFA against Π_1, and may be of independent value.

The small field attack (SFA) against Π_1 Our analysis on Π_1 can be briefly summarized as follows, as is depicted in Figure 2.

- In Section 4, we first review Π_1, and then abstract part of valid functionalities of party j as an oracle $\mathcal{M}_0$ with private key. To efficiently recover the static private key of party j in Π_1, it suffices to show how to recover the private key of $\mathcal{M}_0$ efficiently. To do so, we then define a simplified variant of $\mathcal{M}_0$, *i.e.*, $\mathcal{M}_1$ with secret, which corresponds to the special case when the static public key of the attacker against $\mathcal{M}_0$ is $\mathbf{0} \in \mathcal{R}_q$ by default. Finally, a basic yet efficient attacker $\mathcal{A}_1$ is constructed that can recover the secret of $\mathcal{M}_1$.
  As a concrete instance of SFA, $\mathcal{A}_1$ implies that there exists an efficient attacker with static public key $\mathbf{0} \in \mathcal{R}_q$ who can break party j in Π_1. This basic attacker already demonstrates the insecurity of Π_1 in practice, which is, however, can be easily detected and prevented.
- In Section 5, we continue our analysis on $\mathcal{M}_1$ and construct an efficient attacker $(\mathcal{V}/\mathcal{A}'_1)_\delta$ that can break $\mathcal{M}_1$; And, the queries made by $(\mathcal{V}/\mathcal{A}'_1)_\delta$ are well designed such that it is hard for $\mathcal{M}_1$ to distinguish those malicious queries from honestly generated ones in practice.
  The attacker $(\mathcal{V}/\mathcal{A}'_1)_\delta$ implies that there exists an efficient attacker, still with static public key $\mathbf{0} \in \mathcal{R}_q$, who can break party j in Π_1; Moreover, every query made by this attacker to party j is as "random-looking" as possible.
- In Section 6, we switch our analysis back to $\mathcal{M}_0$, and design an "undetectable" attacker $\mathcal{A}_0$ against $\mathcal{M}_0$, which is very similar to $(\mathcal{V}/\mathcal{A}'_1)_\delta$ and can break $\mathcal{M}_0$ via a sequence of random-looking queries; Furthermore, motivated by the construction of $(\mathcal{V}/\mathcal{A}'_1)_\delta$, the static public key of $\mathcal{A}_0$ can be well formed such that it is hard in practice for $\mathcal{M}_0$ to distinguish the static public key of $\mathcal{A}_0$ from that of an honest user.
  It follows immediately from the construction of $\mathcal{A}_0$ that we can design an "undetectable" attacker $\mathcal{A}$ that can break party j in Π_1, and it is hard in practice for party j to identify either the static public key of $\mathcal{A}$ or those malicious queries made by $\mathcal{A}$.[3]

More details about our attack The vulnerability of Π_1 is demonstrated as follows. First, we abstract some of the valid functionalities of the honest party j in Π_1 as an oracle $\mathcal{M}_0$ with private key, where the private key of $\mathcal{M}_0$ corresponds to the static private key of the honest party j. Each query made to $\mathcal{M}_0$ consists of the caller's public key, the message msg for $\mathcal{M}_0$ to create a new session, and the session key $\mathtt{sk}_i$ of the matching session. On each query, $\mathcal{M}_0$ first creates a new session associated with msg, computes its session key $\mathtt{sk}_j$, and finally returns 1 if $\mathtt{sk}_i = \mathtt{sk}_j$; Otherwise, 0 is returned. Notice that such one-bit oracle is always available in practice, either by the session-key exposure oracle in the security model for one-pass HMQV [Kra05] or by the decryption oracle in the CCA-secure "deniable encryption" proposed in [ZZD$^+$15,ZZDS14]. Moreover, for one-pass protocol Π_1 is deployed in reality with mutual authentications by additionally exchanging MACs, the action differences of

[3] We remark that SFA does not apply to the two-pass protocol Π_2 in [ZZDS14,ZZD$^+$15], since the static private key of party j is protected by its ephemeral private key in Π_2.

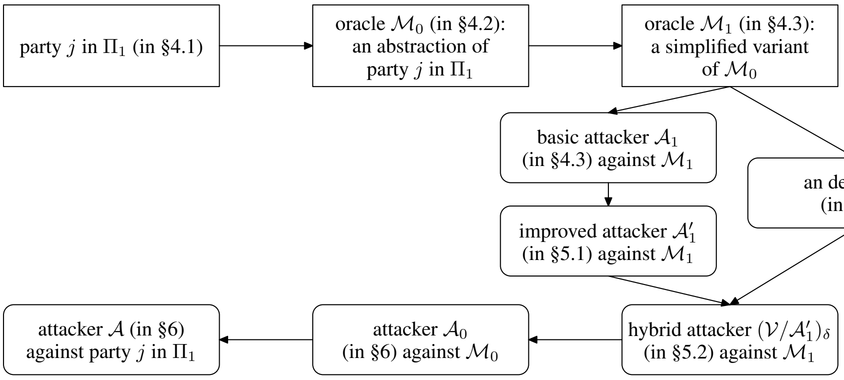

party j in Π_1 (in §4.1)
oracle $\mathcal{M}_0$ (in §4.2): an abstraction of party j in Π_1
oracle $\mathcal{M}_1$ (in §4.3): a simplified variant of $\mathcal{M}_0$
basic attacker $\mathcal{A}_1$ (in §4.3) against $\mathcal{M}_1$
an decider $\mathcal{V}$ (in §5.2)
improved attacker $\mathcal{A}_1'$ (in §5.1) against $\mathcal{M}_1$
attacker $\mathcal{A}$ (in §6) against party j in Π_1
attacker $\mathcal{A}_0$ (in §6) against $\mathcal{M}_0$
hybrid attacker $(\mathcal{V}/\mathcal{A}_1')_\delta$ (in §5.2) against $\mathcal{M}_1$

Fig. 2. The presentation organization of the small field attack. Here, the arrows indicate the order in which our analysis is carried out.

j upon receiving a valid MAC value or an invalid one can be used as such one-bit oracle. To demonstrate the vulnerability of Π_1, it is *sufficient* to construct an *efficient* attacker $\mathcal{A}_0$ that can recover the private key of $\mathcal{M}_0$.

The precise construction of $\mathcal{A}_0$ involves too many details. Fortunately, the following simplified analysis implies how $\mathcal{A}_0$ against $\mathcal{M}_0$ works. For the moment, we assume that the public key of $\mathcal{A}_0$ is $\mathbf{0} \in \mathcal{R}_q$ (this assumption could be dropped finally). *In such simplified setting*, we can define an oracle $\mathcal{M}_1$ with secret $\boldsymbol{s} \leftarrow D_{\mathbb{Z}^n,\alpha}$ which could be seen as a *simplified* variant of $\mathcal{M}_0$. Here, $\boldsymbol{s}$ corresponds to the static private key of party j in Π_1, and could be seen as an element of $\mathcal{R}_q$ in the natural way, provided q is sufficiently large. On input $\left(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z} = [z_j]_{j\in[n]}\right) \in \mathcal{R}_q \times \{0,1\}^n \times \{0,1\}^n$, the oracle $\mathcal{M}_1$ first generates a small error $\varepsilon \leftarrow \mathbb{Z}_{1+2\theta}^n = \{-\theta, \cdots, \theta\}^n$, and then computes $\sigma \triangleq \mathsf{Parity}\left(\boldsymbol{x}\boldsymbol{s} + q_0\boldsymbol{w} + 2\varepsilon\right) = [\sigma_j]_{j\in[n]}$, and finally returns 1 if and only if $[\sigma_j]_{j\in[n]} = [z_j]_{j\in[n]}$. In this work, $\mathsf{Parity}\,(x)$ represents the parity of $x \in \mathbb{F}_q = \left\{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\right\}$; Moreover, for $\boldsymbol{u} = \sum_{j=1}^n u_j\zeta^{j-1} \in \mathcal{R}_q, u_j \in \mathbb{F}_q$, it is understood $\mathsf{Parity}\,(\boldsymbol{u}) = [\mathsf{Parity}\,(u_j)]_{j\in[n]} \in \{0,1\}^n$.

A simple efficient attacker $\mathcal{A}_1$ against $\mathcal{M}_1$ is first constructed in Section 4.3. Observe that if the $\boldsymbol{x}$-entry of every query is of the form $k \cdot \boldsymbol{c}_i, k \in \mathbb{F}_q, i \in [n]$, then the product $\boldsymbol{x}\boldsymbol{s} = k\eta_i(\boldsymbol{s}) \cdot \boldsymbol{c}_i$ always belongs to a "small" set $\langle \boldsymbol{c}_i \rangle = \{k' \cdot \boldsymbol{c}_i \,|\, k' \in \mathbb{F}_q\}$, which is a *subfield* of the ring $\mathcal{R}_q$ and is of size $q = \mathrm{poly}(\lambda) \ll q^n = \left|\mathcal{R}_q\right|$, making it *possible* for us to recover $\eta_i(\boldsymbol{s}) \in \mathbb{F}_q$ efficiently. This explains how our small field attack bears its name. Such observation implies the *general structure* of the desired attacker $\mathcal{A}_1$: the main body of $\mathcal{A}_1$ is an n-round loop, and the i-th round is devoted to the recovery of $s_i \triangleq \eta_i(\boldsymbol{s}) \in \mathbb{F}_q, i \in [n]$; In the i-th round, given $s_1, \cdots, s_{i-1} \in \mathbb{F}_q$ and oracle access to $\mathcal{M}_1$, it first picks $\tilde{s_i} \leftarrow \mathbb{F}_q$ randomly, guesses $s_i = \tilde{s_i} \in \mathbb{F}_q$, and then verifies the correctness of this guess via a set $\mathcal{Q}_i(\tilde{s_i})$ of queries to $\mathcal{M}_1$; The set $\mathcal{Q}_i(\tilde{s_i})$ is carefully chosen such that the $\boldsymbol{x}$-entry is always of the form $k\boldsymbol{c}_i, k \in \mathbb{F}_q^\times$, and the distribution of those query replies under the condition $\tilde{s_i} = s_i$ is *computationally distinguishable* from that under the condition $\tilde{s_i} \neq s_i$; In this manner, when $\tilde{s_i}$ runs over the set $\mathbb{F}_q$, the exact value of $s_i \in \mathbb{F}_q$ would be recovered successfully. In the end, the whole secret $\boldsymbol{s} = \sum_{i\in[n]} s_i \cdot \boldsymbol{c}_i$ is recovered.

Of every query made by $\mathcal{A}_1$, its $\boldsymbol{w}$- and $\boldsymbol{z}$-entries are "random" enough, but its $\boldsymbol{x}$-entry is easy to recognize, since $\boldsymbol{x} \in \left\{k\boldsymbol{c}_i \,\middle|\, k \in \mathbb{F}_q^\times, i \in [n]\right\}$ always holds; Equivalently, $|\mathsf{Dim}\,(\boldsymbol{x})| = 1$. As a result, it is easy for $\mathcal{M}_1$ to prevent $\mathcal{A}_1$ by requiring that $|\mathsf{Dim}\,(\boldsymbol{x})| \geq 2$ for each incoming query. Such restriction is *reasonable* in the sense that the set $\left\{k\boldsymbol{c}_i \,\middle|\, k \in \mathbb{F}_q^\times, i \in [n]\right\}$ is of "small" size compared with $\mathcal{R}_q$. Now the *first motivating question* arises: how to improve $\mathcal{A}_1$ so that we can still recover the secret of $\mathcal{M}_1$, even if the foregoing requirement on the $\boldsymbol{x}$-entry is imposed?

Actually, it is not hard to construct an improved variant of $\mathcal{A}_1$, *i.e.*, $\mathcal{A}_1'$, to resolve this problem, as we shall see in Section 5.1. First, given that in the i-th round, the CRT-coefficients $s_1, \cdots, s_{i-1}$ has already been recovered successfully, we can make full use of these known CRT-coefficients to re-design the $\boldsymbol{x}$-entry. Moreover, since $\boldsymbol{s} \leftarrow D_{\mathbb{Z}^n,\alpha}$ is "small", it can be proven that the product $\boldsymbol{s}\boldsymbol{e} \in \mathcal{R}_q$ is "small" as well for $\boldsymbol{e} \leftarrow [-\alpha'\sqrt{n}, \cdots, \alpha'\sqrt{n}]^n$. These two observations are essential for us to design the desired $\mathcal{A}_1'$: its *general structure* is almost the same as that of $\mathcal{A}_1$, except that in its i-th round, the $\boldsymbol{x}$-entry every query in $\mathcal{Q}_i(\tilde{s_i})$ is always of the form $\boldsymbol{x} = k\cdot\boldsymbol{c}_i + \boldsymbol{h} + 2\boldsymbol{e}$, where $k \in \mathbb{F}_q^\times, \boldsymbol{h} \leftarrow \left\{\boldsymbol{u} \in \mathcal{R}_q \,\middle|\, \mathsf{Dim}\,(\boldsymbol{u}) = [i-1]\right\}$, and $\boldsymbol{e} \leftarrow \mathbb{Z}_{1+2\alpha'\sqrt{n}}^n = \left\{-\alpha'\sqrt{n}, \cdots, \alpha'\sqrt{n}\right\}^n$. By re-designing the $\boldsymbol{w}$- and $\boldsymbol{z}$-entries appropriately, the efficient attacker $\mathcal{A}_1'$ can be proven to recover every CRT-coefficient $s_i \in \mathbb{F}_q$ with overwhelming probability.

Clearly $\mathsf{Dim}\,(k \cdot \boldsymbol{c}_i + \boldsymbol{h}) = [i] = \{1, 2, \cdots, i\}$, and hence the more CRT-coefficients of $\boldsymbol{s}$ we get, the more difficult for $\mathcal{M}_1$ to identify those queries made by $\mathcal{A}_1'$. However, there is still another problem with $\mathcal{A}_1'$: in its *first* round, for every query made by $\mathcal{A}_1'$, $\boldsymbol{h} = \boldsymbol{0} \in \mathcal{R}_q$ and hence the $\boldsymbol{x}$-entry is of the form $\boldsymbol{x} = k\boldsymbol{c}_i + 2\boldsymbol{e}$. To protect its secret, $\mathcal{M}_1$ may reject those queries for which the $\boldsymbol{x}$-entries are of the form $\boldsymbol{x} = k\boldsymbol{c}_i + 2\boldsymbol{e}$ such that $k \in \mathbb{F}_q, i \in [n]$ and $\|\boldsymbol{e}\|_\infty$ is "small". Such requirement seems reasonable as well. Now the *second motivating question* arises: how to improve $\mathcal{A}_1'$ so that we can still recover the secret of $\mathcal{M}_1$, even if the foregoing requirement on the $\boldsymbol{x}$-entry is imposed?

It turns out this question could be resolved *indirectly*. As in Section 5.3, we can define an *efficient* solver $\mathcal{V}$ to the following problem: given a nonempty index set $I \subseteq [n]$, an $|I|$-dimensional vector $[\tilde{s_i}]_{i\in I} \in \mathbb{F}_q^{|I|}$, and oracle access to $\mathcal{M}_1$, decide whether $[\tilde{s_i}]_{i\in I} = [s_i]_{i\in I}$ or not. Moreover, as we shall see, to solve the instance $(I, [\tilde{s_i}]_{i\in I})$, the $\boldsymbol{x}$-entry of every query made by $\mathcal{V}$ to $\mathcal{M}_1$ is of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$, where $\mathsf{Dim}\,(\boldsymbol{x}_0) = I$ and $\boldsymbol{e} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. With the aid of $\mathcal{V}$, we can construct an *efficient hybrid* attacker against $\mathcal{M}_1$ as follow:

Phase 1 First, choose a *constant* δ of moderately large, and an index set $I \subseteq [n]$ of size δ *randomly*. Then, feed $\mathcal{V}$ with q^δ instances, each of the form $(I, [\tilde{s_i}]_{i\in I}), \tilde{s_i} \in \mathbb{F}_q$. In this manner, the CRT-coefficients $s_i, i \in I$, would be recovered successfully when $[\tilde{s_i}]_{i\in I}$ runs over the set $\mathbb{F}_q^\delta$.
In particular, the $\boldsymbol{x}$-entry of every query made in this phase is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$, where $\mathsf{Dim}\,(\boldsymbol{x}_0) = I$, and $\boldsymbol{e} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. Given the randomness of I, it is *hard in practice* for $\mathcal{M}_1$ to identify those queries made by $\mathcal{V}$.

Phase 2 This phase consists of $n - \delta$ rounds, each devoted to recovering one of the remaining $n - \delta$ CRT-coefficients of $\boldsymbol{s}$, as is done in $\mathcal{A}_1'$.
In particular, the $\boldsymbol{x}$-entry of every query made in this phase is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$ where $\mathsf{Dim}\,(\boldsymbol{x}_0) \supseteq I$ and $\boldsymbol{e} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$, making it *more difficult* for $\mathcal{M}_1$ to identify them.

The notation $(\mathcal{V}/\mathcal{A}_1')_\delta$ is applied to indicate this efficient *hybrid* attacker against $\mathcal{M}_1$. Clearly, it is *almost impossible in practice* for $\mathcal{M}_1$ to identify those malicious queries made by $(\mathcal{V}/\mathcal{A}_1')_\delta$. This finishes our discussion about our small field attack against $\mathcal{M}_1$, a simplified variant of $\mathcal{M}_0$.

The desired efficient attacker $\mathcal{A}_0$ against $\mathcal{M}_0$ is similar to $(\mathcal{V}/\mathcal{A}_1')_\delta$ against $\mathcal{M}_1$. Likewise, these queries made by $\mathcal{A}_0$ is so "random-looking" that it is *hard in practice* for $\mathcal{M}_0$ to identify them. Last but not the least, motivated by the construction of $\mathcal{V}$, we can also set the public key of $\mathcal{A}_0$ in a clever way such that it is *hard in practice* to distinguish the public key of $\mathcal{A}_0$ from that of an honest user.

Security model of Π_1 It is noteworthy that [ZZDS14,ZZD+15] do not describe precise definitions and proofs of its security claims for Π_1. Instead, some properties of this security model are discussed. For instance, it is claimed in [ZZDS14,ZZD+15] that Π_1 cannot provide forward secrecy and does not stop replays, just as the one-pass HMQV, and that Π_1 "can be proven in a weak model similar to [Kra05]"; What is more, Π_1 "can essentially be used as a KEM, and can be transformed into a CCA-secure encryption in the random oracle model by combining it with a CPA-secure symmetric-key encryption together with a MAC algorithm in a standard way (where both keys are derived from the session key in the one-pass protocol)". In addition, this CCA-secure encryption allows sender authentication and deniability.

Though some of the claims regarding the security model/proof of Π_1 are unclear to us, as far as we can see, the natural interpretation of the foregoing CCA-security claim about Π_1 is disproved by our SFA attack. We argue that, no matter what the precise interpretations of original security claims were, our attack shows that Π_1 is dangerous in practice, on the following grounds. In our SFA attack, it is hard to distinguish the public key registered by a malicious user from the public key honestly generated. Moreover, as the malicious user does know the private key corresponding to the registered public key, traditional mechanisms for proof-of-knowledge (POK) or proof-of-possession (POP) of private key, *e.g.*, via requiring the user to sign a random message with the registered public key, does not prevent our SFA attack. From our point of view, forbidding adversary from registering public keys on behalf of dishonest users, on the one hand, seems to be unrealistic in practice; And, on the other hand, such a security model seems to be too "weak" as naturally insecure protocol could be proved "secure" within.

Related works The work [Flu16] demonstrates, *in general*, the danger of reusing public key for RLWE-based key exchange. But there are some differences, *in nature*, between our work and [Flu16]. First, [Flu16] considers the simple scenario where the attacker only uses ephemeral public keys, while for SFA against Π_1 the attacker has to mix its public key and ephemeral public keys, making it significantly harder to attack and analyze. Second, it is easy to detect and prevent the attack proposed in [Flu16], which is in contrast to our SFA that is almost undetectable in practice. Third, our SFA critically relies on the properties of the CRT basis for $\mathcal{R}_q$, while [Flu16] employs the power basis. Though our SFA could be simplified to work in the scenario considered in [Flu16], to our knowledge, we do not know how to apply the approach of [Flu16] and its like to break Π_1. Actually, as discussed in [DARF16], no existing concrete one-pass AKE scheme was affected with attack considered in [Flu16].

2 Preliminaries

Let λ denote the security parameters throughout this work. Let $\mathbb{B} \triangleq \{0,1\}$. For an odd integer $p > 0$, let $\mathbb{Z}_p \triangleq \{-\frac{p-1}{2}, \cdots, \frac{p-1}{2}\}$; For instance, $\mathbb{Z}_3 = \{-1,0,1\}$. For every positive integer k, let $[k]$ denote the finite set $\{1,2,\cdots,k\} \subseteq \mathbb{Z}$.

Throughout this work, let $n \geq 16$ be a power-of-two, and $q = \text{poly}(\lambda)$ be a positive rational prime that is polynomial in λ; Moreover, $q \equiv 1 \pmod{2n}$ is necessarily required. When q is clear from the context, define $q_0 \triangleq \frac{q-1}{2}$. Let $\mathbb{F}_q \triangleq \mathbb{Z}/q\mathbb{Z}$ be the finite field of prime order q; In this work, every element in $\mathbb{F}_q$ is represented by a unique element in $\{-q_0, \cdots, q_0\}$. Define $\mathbb{F}_q^\times \triangleq \mathbb{F}_q \setminus \{0\}$, and $\mathsf{zone}_0 \triangleq \{-q_0/2, \cdots, +q_0/2\} \subseteq \mathbb{F}_q$. And, when comparison is carried out between $a, b \in \mathbb{F}_q$, both a, b are regarded as *real numbers*.

Unless otherwise stated, vectors are represented in *column* form in this work, and the n-dimensional vector $(u_1, \cdots, u_n)^t$ is usually abbreviated as $[u_j]_{j\in[n]}$. Let $\mathbf{0} \triangleq (0, \cdots, 0)^t \in \mathbb{F}_q^n$. When either $\boldsymbol{u} \in \mathbb{F}_q^n$ or $\boldsymbol{u} \in \mathbb{Z}^n$, let $\|\boldsymbol{u}\|_2$ and $\|\boldsymbol{u}\|_\infty$ denote the ℓ_2- and ℓ_∞-norms of $\boldsymbol{u}$; For instance, $\|[u_j]_{j\in[n]}\|_2 = \sqrt{u_1^2 + \cdots + u_n^2} \in \mathbb{R}^{\geq 0}$. When n and q are clear in the context, define the projection $\mu_j : \mathbb{F}_q^n \to \mathbb{F}_q$ for every $j \in [n]$ such that $\mu_j\left([u_j]_{j\in[n]}\right) \triangleq u_j$.

For an event E, the notation "$\Pr[E] = \mathsf{negl}(\lambda)$" indicates the probability that E occurs is negligible in λ. Conversely, if $\Pr[E] = 1 - \mathsf{negl}(\lambda)$, we say that E occurs *with overwhelming probability*; For simplicity, the phrase "with overwhelming probability" is usually abbreviated as "*w.o.p.*" in this work. For a *deterministic* algorithm A that returns y on input $x_1, x_2, \cdots, x_k$, this process is usually written as $y := A(x_1, x_2, \cdots, x_k)$. Conversely, for a *randomized* algorithm B, which returns y on input $x_1, x_2, \cdots, x_k$ as well as the random nonce r, this process is usually abbreviated as $y := B(x_1, x_2, \cdots, x_k; r)$, or simply $y \leftarrow B(x_1, x_2, \cdots, x_k)$. The notation $\Pr[R_1; \cdots; R_k : E]$ denotes the probability of the event E after the *ordered* execution of random processes $R_1, \cdots, R_k$. For a *finite* set S, let $x \leftarrow S$ denote a sample drawn from the *uniform* distribution over the (finite) set S.

3 The CRT Basis in the Ring-LWE Setting

This section reviews the ring-LWE problem and its associated notions/results. The definition of the ring-LWE problem, as well as its hardness results, is briefly reviewed in Section 3.1. In Section 3.2, the CRT basis in the RLWE setting is first recalled, followed by its algebraic properties that are essential for our small field attack against the one-pass AKE scheme Π_1 proposed in [ZZD$^+$15,ZZDS14].

3.1 The Ring-LWE Problem

Below are the preliminaries about ideal lattices and the ring-LWE problem. For the full detail, please refer to [LPR13a,LPR13b,DD12].

The rings $\mathcal{R}$ and $\mathcal{R}_q$ Let ζ denote a primitive $2n$-th root of unity in $\mathbb{C}$, and its minimum polynomial over $\mathbb{Q}$ is the $2n$-th cyclotomic polynomial $\Phi_{2n}(x) \triangleq x^n + 1 \in \mathbb{Z}[x]$. Let $\mathcal{R} \triangleq \mathbb{Z}[\zeta]$ be the ring of integers of the number field $\mathbb{Q}(\zeta)/\mathbb{Q}$. Moreover, define the principal ideal $\langle q \rangle = q\mathcal{R}$ and its associated quotient ring $\mathcal{R}_q \triangleq \mathcal{R}/q\mathcal{R}$. In this work, each coset of $q\mathcal{R}$ in $\mathcal{R}$ is naturally represented by a *unique* element in the set $\left\{\sum_{j\in[n]} u_j\zeta^{j-1} \,\middle|\, u_j \in \mathbb{F}_q\right\}$.

Since $\zeta^0, \cdots, \zeta^{n-1}$ constitute an $\mathbb{F}_q$-basis for the free $\mathbb{F}_q$-module $\mathcal{R}_q$ of rank n, every $\boldsymbol{u} = \sum_{j\in[n]} u_j\zeta^{j-1} \in \mathcal{R}_q$ could be identified with the (column) vector $(u_1, \cdots, u_n)^t = [u_j]_{j\in[n]} \in \mathbb{F}_q^n$, *and vice versa*. Such identification is denoted as $\boldsymbol{u} \sim [u_i]_{i\in[n]}$ in this work. Hence, every n-dimensional (column) vector in $\mathbb{F}_q^n$ can be regarded as an element of $\mathcal{R}_q$ in the *natural* way when necessary, *and vice versa*. It follows that the domain of the projection $\mu_j(\cdot)$ defined previously could be generalized to $\mathcal{R}_q$ in the sense that $\mu_j\left(\sum_{j\in[n]} u_j \cdot \zeta^{j-1}\right) \triangleq u_j \in \mathbb{F}_q$ for every $j \in [n]$. Moreover, let $\mu(\boldsymbol{u}) \triangleq [\mu_j(\boldsymbol{u})]_{j\in[n]}$, which induces an $\mathbb{F}_q$-module isomorphism. It is understood that $\|\boldsymbol{u}\|_2 \triangleq \|\mu(\boldsymbol{u})\|_2$ and $\|\boldsymbol{u}\|_\infty \triangleq \|\mu(\boldsymbol{u})\|_\infty$ for every $\boldsymbol{u} \in \mathcal{R}_q$. To emphasize this $\mathbb{F}_q$-module isomorphism, elements of $\mathcal{R}_q$ are represented by lower-case bold letters in this work. In particular, let $\mathbf{0}$ denote both the vector $(0, \cdots, 0)^t \in \mathbb{F}_q^n$ and the zero element of $\mathcal{R}_q$ in the sequel, and it would be clear from the context.

The discrete Gaussian distribution Given the positive real $\alpha > 0$, define the real Gaussian function $\rho_\alpha(x) \triangleq \exp\left(-x^2/2\alpha^2\right)/\sqrt{2\pi\alpha^2}$ for $x \in \mathbb{R}$. Let $D_{\mathbb{Z},\alpha}$ denote the 1-dimensional *discrete* Gaussian distribution over $\mathbb{Z}$, determined by its density function $D_{\mathbb{Z},\alpha}(x) = \rho_\alpha(x)/\rho_\alpha(\mathbb{Z}), x \in \mathbb{Z}$. Finally, let $D_{\mathbb{Z}^n,\alpha}$ denote the n-dimensional *spherical* discrete Gaussian distribution over $\mathbb{Z}^n$, where each coordinate is drawn *independently* from $D_{\mathbb{Z},\alpha}$.

When $\alpha = \omega(\sqrt{\log n})$, *almost* every sample $\varepsilon \leftarrow D_{\mathbb{Z}^n,\alpha}$ is "short" in the sense that $\Pr\left[\|\varepsilon\|_2 \le \alpha\sqrt{n}\right] = 1 - \mathsf{negl}(\lambda)$ [Reg09,LPR13a]. For the "short" noise $\varepsilon \leftarrow D_{\mathbb{Z}^n,\alpha}$, it could be seen as an element of $\mathcal{R}$ in the natural way; Moreover, when $q > 1 + 2\alpha\sqrt{n}$, *except with negligible probability*, ε could be considered to be an element of $\mathbb{F}_q^n$ (and hence of $\mathcal{R}_q$) in the *natural* way as well.

The following lemma implies that for several "short" noises in $\mathcal{R}$, their product is also "short".

Lemma 1. *Let $n = \text{poly}(\lambda) \ge 16$ be a power-of-two. If $\alpha_i = \omega(\sqrt{\log n})$ and $\boldsymbol{e}_i \leftarrow D_{\mathbb{Z}^n,\alpha_i}$ for every $i \in \{1, 2, 3\}$, then every $\boldsymbol{e}_i$ could be regarded as an element of $\mathcal{R}$ in the natural way with overwhelming probability; Moreover, the following inequalities hold with overwhelming probability:*

$$\|\boldsymbol{e}_1\boldsymbol{e}_2\|_\infty \le n \cdot \alpha_1\alpha_2, \quad \|\boldsymbol{e}_1\boldsymbol{e}_2\boldsymbol{e}_3\|_\infty \le n^2 \cdot \alpha_1\alpha_2\alpha_3.$$

□

The ring-LWE (RLWE) problem The following reviews a *special* case [DD12] of the *original* ring-LWE problem [LPR13a], since this special case, instead of the original one, serves as the underlying hard problem of both Π_1 and Π_2 [ZZDS14,ZZD+15], and suffices for the discussions in this work.

Each ring-LWE instance is parameterized by $n = n(\lambda), q = q(\lambda)$ and $\alpha = \alpha(\lambda)$, where $n \ge 16$ is a power-of-two, q is a positive rational prime such that $q \equiv 1 \pmod{2n}$, and $\alpha \ge 0$. For every (fixed) $\boldsymbol{s} \in \mathcal{R}_q$, we define the ring-LWE distribution $A_{n,q,\alpha,\boldsymbol{s}}$ over $\mathcal{R}_q \times \mathcal{R}_q$: a sample drawn from $A_{n,q,\alpha,\boldsymbol{s}}$ is generated by first choosing $\boldsymbol{a} \leftarrow \mathcal{R}_q, \varepsilon \leftarrow D_{\mathbb{Z}^n,\alpha}$, and then outputting the *ring-LWE sample* $(\boldsymbol{a}, \boldsymbol{b} \triangleq \boldsymbol{a}\boldsymbol{s} + \varepsilon) \in \mathcal{R}_q \times \mathcal{R}_q$ or equivalently $(\boldsymbol{a}, \boldsymbol{b} = \boldsymbol{a}\boldsymbol{s} + 2\varepsilon)$ as in [ZZD+15].

Definition 2 ([LPR13a,LPR13b,DD12]). *For the ring-LWE problem, its* search variant *is defined as follows: given access to arbitrarily many* independent *samples drawn from $A_{n,q,\alpha,\boldsymbol{s}}$ for some arbitrary $\boldsymbol{s} \in \mathcal{R}_q$, the problem asks to recover $\boldsymbol{s} \in \mathcal{R}_q$. In contrast, the* decisional variant *asks to distinguish, with non-negligible advantage, between arbitrarily many independent samples from $A_{n,q,\alpha,\boldsymbol{s}}$ for a random $\boldsymbol{s} \leftarrow \mathcal{R}_q$, and the same number of uniformly random and independent samples drawn from the set $\mathcal{R}_q \times \mathcal{R}_q$.*

It can be shown that for Definition 2, its decisional and search variants are *computationally equivalent*, under mild constraints on the parameters [LPR13a,DD12]. Furthermore, the hardness of (search/decisional) ring-LWE problem is implied by the following Theorem.

Theorem 3 ([LPR13a,DD12]). *For the RLWE problem defined in Definition 2, if there is an efficient algorithm that can distinguish, with $1/\text{poly}(\lambda)$ advantage, between ℓ samples drawn from the uniform distribution over $\mathcal{R}_q \times \mathcal{R}_q$ and ℓ samples drawn from $A_{n,q,\alpha,\boldsymbol{s}}$, where $\beta = \sqrt{n}\alpha q \cdot (n\ell/\log(n\ell))^{1/4}$, then there exists an efficient quantum algorithm that runs in time $O(q \cdot \text{poly}(m))$ and solves the approximate SVP problem to within a factor $\tilde{O}(\sqrt{2n}/\alpha)$ in any ideal of $\mathbb{Z}[\zeta]$.* □

3.2 The CRT Basis for $\mathcal{R}_q$ and Its Properties

The notion of the CRT basis (in the ring-LWE setting) was first proposed in [LPR13a]. Here we first review its definition and basic properties; After that, we develop a new algebraic property, *i.e.*, Lemma 5, regarding the CRT basis for $\mathcal{R}_q$, which would be *essential* for our efficient attackers to be developed later.

When n, q are clear, let $\{\omega_1, \cdots, \omega_n\} \subseteq \mathbb{F}_q^\times \setminus \{\pm 1\}$ be the set of elements in $\mathbb{F}_q^\times$ that are of multiplicative order $2n$. Since the polynomial $\Phi_{2n}(x) \equiv \prod_{i\in[n]}(x - \omega_i) \pmod{q}$ is separable over $\mathbb{F}_q$ by the assumption on parameters, the principal ideal $q\mathcal{R}$ of $\mathcal{R}$ could be factored as $q\mathcal{R} = \prod_{i\in[n]} \mathfrak{q}_i$, where every nonzero prime ideal $\mathfrak{q}_i = \langle q, \zeta - \omega_i\rangle$ by a suitable ordering, and the norm of every $\mathfrak{q}_i$ in $\mathcal{R}$ is $|\mathcal{R}/\mathfrak{q}_i| = q^{\deg(x-\omega_i)} = q$. Hence, every quotient ring $\mathcal{R}/\mathfrak{q}_i$ is a finite field of prime order q, indicating that $\mathcal{R}/\mathfrak{q}_i \cong \mathbb{F}_q$. Hence, the ring $\mathcal{R}_q$ could be identified with the direct product of n *small* finite field, each of prime order $q = \text{poly}(\lambda)$. This explains how our notion of small field attack bears its name.

As the *distinct* nonzero prime ideals $\mathfrak{q}_1, \cdots, \mathfrak{q}_n$ are necessarily pairwise coprime, it follows from the Chinese remainder theorem that $\mathcal{R}_q = \mathcal{R}/q\mathcal{R} \cong \prod_{i\in[n]} \mathcal{R}/\mathfrak{q}_i$ under the *natural ring isomorphism* $\boldsymbol{u} + q\mathcal{R} \mapsto (\boldsymbol{u} +$

$\mathfrak{q}_1, \cdots, \boldsymbol{u} + \mathfrak{q}_n)$. Under the natural isomorphism, we can find n elements $\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n \in \mathcal{R}$ such that $\boldsymbol{c}_i \equiv \delta_{i,j} \pmod{\mathfrak{q}_j}$ for every $i, j \in [n]$, where $\delta_{\cdot,\cdot}$ denotes the Kronecker delta function. The elements $\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n \in \mathcal{R}$ form an *integral* basis for $\mathcal{R}$ relative to $\mathfrak{q}_1, \cdots, \mathfrak{q}_n$, and is called *a CRT basis* for $\mathcal{R}$ relative to $\mathfrak{q}_1, \cdots, \mathfrak{q}_n$. Moreover, it is easy to see for any two CRT bases of $\mathcal{R}$ (relative to $\mathfrak{q}_1, \cdots, \mathfrak{q}_n$), they are *equivalent* up to $\bmod\ q\mathcal{R}$. In particular, when $\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n$ fall into the set $\mathcal{R}_q = \left\{ \sum_{j \in [n]} u_j \zeta^{j-1} \,\middle|\, u_j \in \mathbb{F}_q \right\}$, they form an $\mathbb{F}_q$-basis for the free $\mathbb{F}_q$-module $\mathcal{R}_q$; By definition, this $\mathbb{F}_q$-basis is *unique* in $\mathcal{R}_q$ up to ordering, which would be called *the CRT basis* for $\mathcal{R}_q$ hereafter.

The basic properties of the CRT basis for $\mathcal{R}_q$ are summarized as follows.

Fact 4 ([LPR13a,LPR13b]). *For the CRT basis $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ for $\mathcal{R}_q$,*

(a) Given n and q (in unary *form), the CRT basis for $\mathcal{R}_q$ could be found* efficiently.
(b) Every $\boldsymbol{u} \in \mathcal{R}_q$ can be written uniquely *as $\boldsymbol{u} = \sum_{i \in [n]} u_i \cdot \boldsymbol{c}_i$, $u_i \in \mathbb{F}_q$.*
(c) Let $\boldsymbol{u} = \sum_{i \in [n]} u_i \cdot \boldsymbol{c}_i$ and $\boldsymbol{v} = \sum_{i \in [n]} v_i \cdot \boldsymbol{c}_i$, $u_i, v_i \in \mathbb{F}_q$. Then for every $k \in \mathbb{F}_q$, we have:

$$k \cdot \boldsymbol{u} = \sum (k \cdot u_i) \cdot \boldsymbol{c}_i, \quad \boldsymbol{u} + \boldsymbol{v} = \sum (u_i + v_i) \cdot \boldsymbol{c}_i, \quad \boldsymbol{u} \cdot \boldsymbol{v} = \sum (u_i \cdot v_i) \cdot \boldsymbol{c}_i . \square$$

Throughout this work, when n, q are clear from the context, let $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ denote the CRT basis for $\mathcal{R}_q$; Moreover, define $c_{i,j} \triangleq \mu_j(\boldsymbol{c}_i) \in \mathbb{F}_q$ for every $i, j \in [n]$. Thus, every $\boldsymbol{c}_i = \sum_{j \in [n]} c_{i,j} \cdot \zeta^{j-1}$.

With the CRT basis $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ for $\mathcal{R}_q$ in mind, define the map $\eta_i : \mathcal{R}_q \to \mathbb{F}_q$ for every $i \in [n]$, where $\eta_i \left(\sum_{i \in [n]} (u_i \cdot \boldsymbol{c}_i) \right) \triangleq u_i$, $u_i \in \mathbb{F}_q$. Clearly, the map $\eta_i(\cdot)$ is *well-defined* and *efficiently computable*. Every $\eta_i(\boldsymbol{u})$ is called a *CRT-coefficient* of $\boldsymbol{u} \in \mathcal{R}_q$. Moreover, define the map $\eta : \mathcal{R}_q \to \mathbb{F}_q^n$, where $\eta(\boldsymbol{u}) \triangleq [\eta_i(\boldsymbol{u})]_{i \in [n]}$ for every $\boldsymbol{u} \in \mathcal{R}_q$. Direct verification shows that $\eta(\cdot)$ is a ring isomorphism, and is an $\mathbb{F}_q$-module isomorphism when both are regarded as free $\mathbb{F}_q$-modules.

For every $\boldsymbol{u} \in \mathcal{R}_q$, define $\mathsf{Dim}\,(\boldsymbol{u}) \triangleq \{ i \in [n] \,|\, \eta_i(\boldsymbol{u}) \neq 0 \in \mathbb{F}_q \} \subseteq [n]$, and the cardinality $|\mathsf{Dim}\,(\boldsymbol{u})| \in \{0, 1, \cdots, n\}$ is called the *CRT-dimension* of $\boldsymbol{u}$.

We conclude this section by presenting a new property of the CRT basis $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ for $\mathcal{R}_q$, which is *essential* for our SFA attacks against Π_1. The proof of this lemma is already implict in [LPR13a,LPR13b], but we reprove it here for completeness.

Lemma 5. *With the notations defined previously, we have $c_{i,j} = c_{i,n} \cdot \omega_i^{n-j} \neq 0 \in \mathbb{F}_q$ for every $i, j \in [n]$.*

Proof. Fix $i \in [n]$. For the element $\zeta \in \mathcal{R}$, we have

$$\zeta + \mathfrak{q}_i = \zeta + \langle q, \zeta - \omega_i \rangle = \omega_i + \langle q, \zeta - \omega_i \rangle \in \mathcal{R}/\mathfrak{q}_i .$$

It follows that in the ring $\mathcal{R}$, $\zeta \equiv \omega_i \cdot \boldsymbol{c}_i \pmod{\mathfrak{q}_i}$.

On the one hand, by Fact 4, we have that in $\mathcal{R}_q$,

$$\zeta \cdot \boldsymbol{c}_i = \omega_i \cdot \boldsymbol{c}_i = \sum_{j \in [n]} (\omega_i \cdot c_{i,j}) \zeta^{j-1} \in \mathcal{R}_q .$$

On the other hand, the following equality holds in $\mathcal{R}_q$:

$$\begin{aligned} \zeta \cdot \boldsymbol{c}_i &= \zeta \cdot \left(c_{i,1} + c_{i,2}\zeta \cdots + c_{i,n}\zeta^{n-1} \right) \\ &= c_{i,n}\zeta^n + \sum_{j \in [n-1]} c_{i,j}\zeta^j \\ &= -c_{i,n} + \sum_{j \in [n-1]} c_{i,j}\zeta^j , \end{aligned}$$

where the equality $\mathbf{0} = \Phi_{2n}(\zeta) = \zeta^n + 1 \in \mathcal{R}_q$ is implicitly applied.

As $\zeta^0, \zeta, \cdots, \zeta^{n-1}$ form an $\mathbb{F}_q$-basis for the $\mathbb{F}_q$-module $\mathcal{R}_q$, we have

$$c_{i,1} = \omega_i c_{i,2}, \quad \cdots \quad c_{i,n-1} = \omega_i c_{i,n}, \quad -c_{i,n} = \omega_i c_{i,1};$$

Equivalently, $c_{i,j} = c_{i,n} \cdot \omega_i^{n-j} \in \mathbb{F}_q$ for every $j \in [n]$.

If $c_{i,n} = 0$, then $c_{i,j} = 0$ for every $j \in [n]$, making $\boldsymbol{c}_i = \mathbf{0}$, contradictory to the definition of the CRT basis for $\mathcal{R}_q$. It follows that $c_{i,n} \neq 0$, and hence $c_{i,j} \neq 0$ for every $j \in [n]$. And the correctness of this theorem is thus established. $\square$

4 How to Attack Π_1: a Warm-up

We first review the one-pass AKE scheme Π_1 [ZZDS14,ZZD$^+$15] in Section 4.1; Then in Section 4.2, the honest party j in Π_1 is abstracted as an oracle $\mathcal{M}_0$ with private key, such that to efficiently recover the static private key of party j in Π_1, it suffices to show how to recover the private key of $\mathcal{M}_0$ efficiently, *provided that adversaries are allowed to register public keys on behalf of dishonest users.*

4.1 The One-Pass AKE Scheme Π_1

The one-pass AKE scheme Π_1 [ZZDS14,ZZD$^+$15] is built upon the RLWE problem in Definition 2, where every RLWE sample is of the form $(\boldsymbol{a}, \boldsymbol{a}\boldsymbol{s} + 2\varepsilon)$.

In Π_1, $\boldsymbol{a} \leftarrow \mathcal{R}_q$ is a global public parameter, and $M > 0$ is a sufficiently large constant. As a two-party AKE scheme, users in Π_1 are represented by party i (*i.e.*, the initiator) and party j (*i.e.*, the responder), respectively. For party i: the static private key is $(\boldsymbol{s}_i, \boldsymbol{e}_i)$, where $\boldsymbol{s}_i, \boldsymbol{e}_i \leftarrow D_{\mathbb{Z}^n,\alpha}$; Its associated static public key is $\boldsymbol{p}_i \triangleq \boldsymbol{a}\boldsymbol{s}_i + 2\boldsymbol{e}_i \in \mathcal{R}_q$; And its identity issued by the Certificate Authority (CA) is $\mathtt{id}_i$. Similar notations, $\boldsymbol{s}_j, \boldsymbol{e}_j \leftarrow D_{\mathbb{Z}^n,\alpha}, \boldsymbol{p}_j \triangleq \boldsymbol{a}\boldsymbol{s}_j + 2\boldsymbol{e}_j \in \mathcal{R}_q$, and $\mathtt{id}_j$, apply to party j. Let $H_1 : \{0,1\}^* \to D_{\mathbb{Z}^n,\gamma}$ be a hash function that outputs *invertible* elements in $\mathcal{R}_q$, and $H_2 : \{0,1\}^* \to \{0,1\}^*$ be the key derivation function. Both H_1 and H_2 are regarded as random oracles in Π_1.

The following functions are essential for the definition of Π_1. First, When the prime q is clear from the context, define the function $\mathsf{Parity} : \mathbb{F}_q \to \{0,1\}$, where $\mathsf{Parity}(u) \triangleq u (\bmod 2) \in \{0,1\}$. Moreover, we define the function $\mathsf{Mod} : \mathbb{F}_q \times \{0,1\} \to \{0,1\}$, where $\mathsf{Mod}(u,w) \triangleq \mathsf{Parity}((u + w \cdot q_0) \bmod q) \in \{0,1\}$. Finally, define the function $\mathsf{Cha} : \mathbb{F}_q \to \{0,1\}$, such that $\mathsf{Cha}(u) = 0$ if and only if $u \in \left\{-\frac{q-1}{4}, \cdots, \frac{q-1}{4}\right\}$; Otherwise, $\mathsf{Cha}(u) = 1$. All these functions could be easily generalized to the n-dimensional case *in the component-wise manner*. For instance, $\mathsf{Cha}(\boldsymbol{v}) \triangleq [\mathsf{Cha}(v_j)]_{j\in[n]} \in \{0,1\}^n$ for every $\boldsymbol{v} = [v_j]_{j\in[n]} \in \mathbb{F}_q^n$, and it is understood that $\mathsf{Cha}\left(\sum_{j\in[n]} u_j \zeta^{j-1}\right) \triangleq \mathsf{Cha}\left([u_j]_{j\in[n]}\right)$.

The following fact is *essential* both for Π_1 and for our small field attack.

Fact 6. *For every $u \in \mathbb{F}_q$,*

(a) We always have $v \triangleq u + \mathsf{Cha}(u) \cdot q_0 \in \{-q_0/2, \cdots, +q_0/2\} \subseteq \mathbb{F}_q$.

(b) The value $\mathsf{Parity}(v) = \mathsf{Mod}(u, \mathsf{Cha}(u)) \in \mathbb{B}$ *is* immune to a short even noise *in the sense that*

$$\mathsf{Parity}(v) = \mathsf{Parity}(v + 2e)$$

for every $-q_0/4 < e < q_0/4$.

(c) The value $\mathsf{Parity}(v) = \mathsf{Mod}(u, \mathsf{Cha}(u)) \in \mathbb{B}$ *is* sensitive to a short odd noise *in the sense that*

$$\mathsf{Parity}(v + 2e - 1) \neq \mathsf{Parity}(v) \neq \mathsf{Parity}(v + 2e + 1)$$

for every $-q_0/4 < e < q_0/4$. □

The one-pass scheme Π_1 can be roughly described in the following Figure 3 [ZZDS14,ZZD$^+$15].

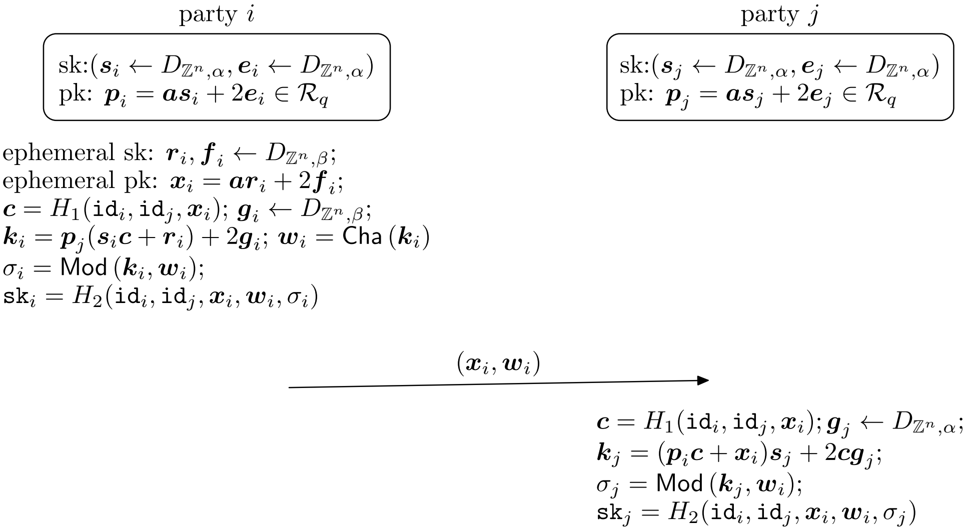
Correctness analysis of Π_1 Roughly speaking, the correctness of this scheme states that $\mathtt{sk}_i = \mathtt{sk}_j$ holds *w.o.p.* On the one hand, since $H_2(\cdot)$ is modeled as a random oracle and $\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{w}_i$ are known to both parties, it suffices to show the equality $\sigma_i = \sigma_j$ holds *w.o.p.* On the other hand, it is routine to verify

$$\boldsymbol{k}_i - \boldsymbol{k}_j = 2 \cdot \left(\boldsymbol{c}\boldsymbol{s}_i\boldsymbol{e}_j + \boldsymbol{r}_i\boldsymbol{e}_j + \boldsymbol{g}_i - \boldsymbol{c}\boldsymbol{e}_i\boldsymbol{s}_j - \boldsymbol{f}_i\boldsymbol{s}_j - \boldsymbol{c}\boldsymbol{g}_j\right).$$

By Lemma 1, $\|\boldsymbol{k}_i - \boldsymbol{k}_j\|_\infty \le 2\left(n\alpha\gamma + \beta\sqrt{n} + 2n\alpha\beta + 2n^2\alpha^2\gamma\right)$ holds *w.o.p.* Thus, when q is sufficiently large, $\|\boldsymbol{k}_i - \boldsymbol{k}_j\|_\infty < q_0/2$ holds *w.o.p.* It follows from Fact 6(b) that $\sigma_i = \mathsf{Mod}(\boldsymbol{k}_i, \mathsf{Cha}(\boldsymbol{k}_i)) = \mathsf{Mod}(\boldsymbol{k}_j, \mathsf{Cha}(\boldsymbol{k}_i)) = \sigma_j$ holds *w.o.p.* This finishes the correctness analysis of Π_1.

Clearly Lemma 1 is *essential* for the correctness of Π_1. However, when n is *not* a power-of-two, or $q \not\equiv 1 \pmod{2n}$, inequalities in Lemma 1 may *no longer* hold, which implies the underlying hard problem of Π_1 *must be* Definition 2, a *special* case of the original ring-LWE problem defined in [LPR13a,LPR13b].

Security analysis of Π_1 It is claimed in [ZZDS14,ZZD$^+$15] that when n is a power-of-two, $0.97n \ge 2\lambda$, $\beta = \omega(\alpha\gamma n\sqrt{n \log n})$, the prime $q > 203$ satisfies $q \equiv 1 \pmod{2n}$, if the associated ring-LWE problem is "hard", then Π_1 is *secure*. In addition, four groups of parameters are suggested in [ZZD$^+$15,ZZDS14] to instantiate Π_1. For instance, in one *typical* group of suggested parameters, $n = 1024, q \approx 2^{30}, \alpha = 3.397, \beta \approx 2^{16.1}, \gamma = \alpha$.



party i

sk:$(\boldsymbol{s}_i \leftarrow D_{\mathbb{Z}^n,\alpha}, \boldsymbol{e}_i \leftarrow D_{\mathbb{Z}^n,\alpha})$
pk: $\boldsymbol{p}_i = \boldsymbol{a}\boldsymbol{s}_i + 2\boldsymbol{e}_i \in \mathcal{R}_q$

party j

sk:$(\boldsymbol{s}_j \leftarrow D_{\mathbb{Z}^n,\alpha}, \boldsymbol{e}_j \leftarrow D_{\mathbb{Z}^n,\alpha})$
pk: $\boldsymbol{p}_j = \boldsymbol{a}\boldsymbol{s}_j + 2\boldsymbol{e}_j \in \mathcal{R}_q$

ephemeral sk: $\boldsymbol{r}_i, \boldsymbol{f}_i \leftarrow D_{\mathbb{Z}^n,\beta}$;
ephemeral pk: $\boldsymbol{x}_i = \boldsymbol{a}\boldsymbol{r}_i + 2\boldsymbol{f}_i$;
$\boldsymbol{c} = H_1(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i)$; $\boldsymbol{g}_i \leftarrow D_{\mathbb{Z}^n,\beta}$;
$\boldsymbol{k}_i = \boldsymbol{p}_j(\boldsymbol{s}_i\boldsymbol{c} + \boldsymbol{r}_i) + 2\boldsymbol{g}_i$; $\boldsymbol{w}_i = \mathsf{Cha}\,(\boldsymbol{k}_i)$
$\sigma_i = \mathsf{Mod}\,(\boldsymbol{k}_i, \boldsymbol{w}_i)$;
$\mathtt{sk}_i = H_2(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{w}_i, \sigma_i)$

$\xrightarrow{(\boldsymbol{x}_i, \boldsymbol{w}_i)}$

$\boldsymbol{c} = H_1(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i); \boldsymbol{g}_j \leftarrow D_{\mathbb{Z}^n,\alpha}$;
$\boldsymbol{k}_j = (\boldsymbol{p}_i\boldsymbol{c} + \boldsymbol{x}_i)\boldsymbol{s}_j + 2\boldsymbol{c}\boldsymbol{g}_j$;
$\sigma_j = \mathsf{Mod}\,(\boldsymbol{k}_j, \boldsymbol{w}_i)$;
$\mathtt{sk}_j = H_2(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{w}_i, \sigma_j)$

Fig. 3. General description of Π_1. Note that in the *full* description of Π_1 in [ZZDS14,ZZD$^+$15], party i applies the rejection sampling to generate a "good" $(\boldsymbol{x}_i, \boldsymbol{w}_i)$ pair. Since the efficient attacker always generates his queries *dishonestly*, this technical detail does not affect our analysis and hence is omitted here for simplicity.

4.2 Definition of the Oracle $\mathcal{M}_0$

To simplify our discussion on how to corrupt party j in Π_1, we define in this section an oracle $\mathcal{M}_0$ which captures some functionalities of party j. The oracle $\mathcal{M}_0$ is well-defined so that to corrupt party j in Π_1, it suffices to show how to construct an "undetectable" attacker against $\mathcal{M}_0$, *provided that adversaries are allowed to register public keys on behalf of dishonest users in* Π_1.

First, notice that in Π_1, the session key derivation function $H_2(\cdot)$ is modeled as an random oracle. Moreover, notice that every time party j generates its session key by invoking $\mathtt{sk}_j \triangleq H_2(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{w}_i, \sigma_j)$, *all the input values except* σ_j are known to party i. It follows that *except with negligible probability*, if party i is able to figure out the session key $\mathtt{sk}_j$ of party j correctly *before* it issues the associated session-key query to party j, then party i must be able to figure out the associated σ_j *beforehand, and vice versa.*

Now we can define an oracle $\mathcal{M}_0$ with private key, which aims to simulate some valid functionalities of party j in Π_1. The private key of $\mathcal{M}_0$ is $(\boldsymbol{s}, \boldsymbol{e})$ and the associated public key is $\boldsymbol{p}$, where $\boldsymbol{s}, \boldsymbol{e} \leftarrow D_{\mathbb{Z}^n,\alpha}$ and $\boldsymbol{p} \triangleq \boldsymbol{a}\cdot\boldsymbol{s}+2\boldsymbol{e} \in \mathcal{R}_q$ (recall that $\boldsymbol{a} \leftarrow \mathcal{R}_q$ is a global parameter in Π_1). Moreover, as a simulator of party j, the identifier of $\mathcal{M}_0$ is denoted by $\mathtt{id}$. On input $(\mathtt{id}^*, \boldsymbol{p}^*, \boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z})$, where $\boldsymbol{x} \in \mathcal{R}_q, \boldsymbol{z}, \boldsymbol{w} \in \mathbb{B}^n$, and $\mathtt{id}^*$ denotes the identifier of the initiator with static public key $\boldsymbol{p}^* \in \mathcal{R}_q$, $\mathcal{M}_0$ does the following: it first samples $\boldsymbol{g} \leftarrow D_{\mathbb{Z}^n,\alpha}$, then computes $\boldsymbol{c} \leftarrow H_1(\mathtt{id}^*, \mathtt{id}, \boldsymbol{x})$, $\boldsymbol{k} := (\boldsymbol{p}^*\boldsymbol{c} + \boldsymbol{x})\boldsymbol{s} + q_0\boldsymbol{w} + 2\boldsymbol{c}\boldsymbol{g}$, and $\sigma := \mathsf{Parity}\,(\boldsymbol{k}) \in \mathbb{B}^n$; Finally, $\mathcal{M}_0$ returns 1 if and only if $\boldsymbol{z} = \sigma$.

This 1-bit oracle $\mathcal{M}_0$ is defined to capture that in every session interaction, a malicious attacker could get from party j *only* 1-bit *useful* information, *i.e.*, whether the session-key returned by party j is equal to the expected one or not. Clearly, such one-bit oracle is available in practice.

We shall construct, in Section 6, an efficient attacker $\mathcal{A}_0$ that can recover the private key $(\boldsymbol{s}, \boldsymbol{e})$ of $\mathcal{M}_0$, *provided that* $\mathcal{A}_0$ *can register its static public/private key pair on its own*; Moreover, the desired $\mathcal{A}_0$ is carefully designed such that both its static public key and those queries it makes are as "random-looking" as possible. Clearly the existence of $\mathcal{A}_0$ implies the vulnerability of Π_1.

4.3 The Oracle $\mathcal{M}_1$ and Its Associated Efficient Attacker $\mathcal{A}_1$

For the moment, we assume that there exists an efficient attacker against $\mathcal{M}_0$ with static public key $\boldsymbol{p}^* = \boldsymbol{0} \in \mathcal{R}_q$. By definition, for the input $(\mathtt{id}^*, \boldsymbol{p}^* = \boldsymbol{0}, \boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z})$ made by this attacker, $\mathcal{M}_0$ first samples $\boldsymbol{g} \leftarrow D_{\mathbb{Z}^n,\alpha}$, then computes $\boldsymbol{c} \leftarrow H_1(\mathtt{id}^*, \mathtt{id}, \boldsymbol{x})$, $\boldsymbol{k} := (\boldsymbol{p}^*\boldsymbol{c} + \boldsymbol{x})\boldsymbol{s} + q_0\boldsymbol{w} + 2\boldsymbol{c}\boldsymbol{g} = \boldsymbol{x}\boldsymbol{s} + q_0\boldsymbol{w} + 2\boldsymbol{c}\boldsymbol{g}$, and $\sigma := \mathsf{Parity}\,(\boldsymbol{k}) \in \mathbb{B}^n$; Finally, $\mathcal{M}_0$ returns 1 if and only if $\boldsymbol{z} = \sigma$. Notice that $\|\boldsymbol{c}\boldsymbol{g}\|_\infty \leq n \cdot \alpha\gamma$ by Lemma 1.

This simplified analysis motivates us to define the oracle $\mathcal{M}_1$ with secret $\boldsymbol{s} \leftarrow D_{\mathbb{Z}^n,\alpha}$: on input $\left(\boldsymbol{x},\ \boldsymbol{w},\ \boldsymbol{z} = [z_j]_{j\in[n]}\right) \in \mathcal{R}_q \times \mathbb{B}^n \times \mathbb{B}^n$, the oracle $\mathcal{M}_1$ first generates a small error $\varepsilon \leftarrow \mathbb{Z}^n_{1+2\theta} = \{-\theta, \cdots, \theta\}^n$, and then computes $\sigma \triangleq \mathsf{Parity}\left(\boldsymbol{xs} + q_0\boldsymbol{w} + 2\varepsilon\right) = [\sigma_j]_{j\in[n]}$, and finally returns 1 if and only if $[\sigma_j]_{j\in[n]} = [z_j]_{j\in[n]}$; Otherwise, 0 is returned. Here, $\theta > 0$ is a constant parameter.

Clearly, $\mathcal{M}_1$ could be seen as a simplified variant of $\mathcal{M}_0$ with $\theta = n\alpha\gamma$. And, the rest of this section is devoted to the construction of a *basic* yet efficient attacker $\mathcal{A}_1$ that, given oracle access to $\mathcal{M}_1$, can recover the secret of $\mathcal{M}_1$ *w.o.p.*

The CRT basis w.r.t. $\mathcal{A}_1$ The construction of $\mathcal{A}_1$ could be seen as a simple application of the CRT basis $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ for $\mathcal{R}_q$. Recall that with $\{\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n\}$ that is efficiently computable, every element $\boldsymbol{u} \in \mathcal{R}_q$ can be *uniquely* written as $\boldsymbol{u} = \sum_{i\in[n]} \eta_i(\boldsymbol{u}) \cdot \boldsymbol{c}_i$; Moreover, we have $\boldsymbol{uv} = \sum_{i\in[n]} \eta_i(\boldsymbol{u})\eta_i(\boldsymbol{v}) \cdot \boldsymbol{c}_i$ for every $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{R}_q$. Thus, to recover $\boldsymbol{s} \in \mathcal{R}_q$, it suffices to recover all of its CRT-coefficients $s_i \triangleq \eta_i(\boldsymbol{s}) \in \mathbb{F}_q, i \in [n]$. Furthermore, notice that for every query $(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z})$ made to $\mathcal{M}_1$, if $\boldsymbol{x} = k \cdot \boldsymbol{c}_i$ for some $k \in \mathbb{F}_q, i \in [n]$, then we have $\boldsymbol{xs} = ks_i \cdot \boldsymbol{c}_i$; In such setting, the product $\boldsymbol{xs}$ always falls into a subfield $\{k\boldsymbol{c}_i \mid k \in \mathbb{F}_q\}$ of $\mathcal{R}_q$, which is of "small" size $q = \mathrm{poly}(\lambda)$, making it possible for us to recover every $s_i \in \mathbb{F}_q$ in a *sequential* manner.

General structure of $\mathcal{A}_1$ By properties of the CRT basis for $\mathcal{R}_q$, to recover the secret $\boldsymbol{s} \in \mathcal{R}_q$ of $\mathcal{M}_1$, it suffices to recover every $s_i \triangleq \eta_i(\boldsymbol{s}) \in \mathbb{F}_q$. And by the search-to-decisional reduction, the recovery of s_i could be boiled down to deciding whether $s_i = \tilde{s_i} \in \mathbb{F}_q$ or not for every $\tilde{s_i} \in \mathbb{F}_q$. These observations indicate the *general structure* of our desired $\mathcal{A}_1$ as follows: the main body of $\mathcal{A}_1$ consist of an n-round loop, and the i-th round is devoted to the recovery of $s_i = \eta_i(\boldsymbol{s}) \in \mathbb{F}_q, i \in [n]$; In the i-th round, given $s_1, \cdots, s_{i-1}$ and oracle access to $\mathcal{M}_1$, it first picks $\tilde{s_i} \leftarrow \mathbb{F}_q$ randomly, guesses $s_i = \tilde{s_i} \in \mathbb{F}_q$, and then verifies the correctness of this guess via a set $\mathcal{Q}_i(\tilde{s_i})$ of queries to $\mathcal{M}_1$; The set $\mathcal{Q}_i(\tilde{s_i})$ should be carefully chosen such that the distribution of these query replies under the condition $s_i = \tilde{s_i}$ is *computationally distinguishable* from that under the condition $s_i \neq \tilde{s_i}$; Thus, the exact value of $s_i \in \mathbb{F}_q$ would be recovered *w.o.p.* after $\tilde{s_i}$ runs over the set $\mathbb{F}_q$.

Design of $\mathcal{Q}_i(\tilde{s_i})$ For the moment, assume $s_1, \cdots, s_{i-1}$ have been recovered *successfully* and $\tilde{s_i} \in \mathbb{F}_q$ is fixed, and we are devoted to verifying the correctness of the guess $s_i = \tilde{s_i}$ by designing the *desired* query set $\mathcal{Q}_i(\tilde{s_i})$. Jumping ahead,

$$\mathcal{Q}_i(\tilde{s_i}) = \left\{ Q_k \triangleq \left(\boldsymbol{x}_k = k\boldsymbol{c}_i,\ \boldsymbol{w}_k = [w_{k,j}]_{j\in[n]},\ \boldsymbol{z}_k = [z_{k,j}]_{j\in[n]}\right) \,\middle|\, k \in S_g \subseteq \mathbb{F}_q^\times \right\},$$

where S_g is a nonempty proper subset of $\mathbb{F}_q^\times$ to be defined later. It remains to specify the $\boldsymbol{w}$- and $\boldsymbol{z}$-entries of every query in $\mathcal{Q}_i(\tilde{s_i})$.

For the moment, we assume $s_i = \tilde{s_i}$. In such case, for the query $Q_k = (k\boldsymbol{c}_i, \boldsymbol{w}_k, \boldsymbol{z}_k) \in \mathcal{Q}_i(\tilde{s_i})$, if $w_{k,j} = \mathsf{Cha}\left(\mu_j(k\tilde{s_i}\boldsymbol{c}_i)\right)$ for every $j \in [n]$, then every $\mu_j(k\boldsymbol{c}_i\boldsymbol{s} + q_0\boldsymbol{w}_k)$ into the "secure zone" zone_0 by Fact 6(a); Moreover, when every $\mu_j(k\boldsymbol{c}_i\boldsymbol{s} + q_0\boldsymbol{w}_k) \in \mathrm{zone}_0$, Fact 6(b) guarantees that the *small and even* noise 2ε does not affect the parity values, provided that θ is "small" relative to q; Therefore, $\mathcal{M}_1$ would return 1 on Q_k if $\boldsymbol{z}_k = \mathsf{Mod}\left(k\boldsymbol{c}_i, \boldsymbol{w}_k\right) = \mathsf{Parity}\left(k\boldsymbol{c}_i\boldsymbol{s} + q_0\boldsymbol{w}_k\right)$. Conversely, if $s_i \neq \tilde{s_i}$, then some of the associated equalities may not hold *intuitively*. The correctness of our foregoing intuition is justified by the following lemma.

Lemma 7. *Let $g \in \mathbb{F}_q^\times$ denote a primitive element of $\mathbb{F}_q$, and let $S_g \triangleq \{g^r \mid r \in [d]\}$ and $d \triangleq \frac{q-1}{2n}$. Define*

$$\mathcal{Q}_i(\tilde{s_i}) = \left\{ Q_k = \left(k\boldsymbol{c}_i, [w_{k,j}]_{j\in[n]}, [z_{k,j}]_{j\in[n]}\right) \,\middle|\, \begin{array}{c} k \in S_g,\ \ j \in [n],\ \ u_{k,j} = \tilde{s_i} \cdot kc_{i,j}, \\ w_{k,j} = \mathsf{Cha}\left(u_{k,j}\right), z_{k,j} = \mathsf{Mod}\left(u_{k,j}, w_{k,j}\right) \end{array} \right\}.$$

If $q > 1 + \max\{8\theta,\ 2\alpha\sqrt{n}\}$, then except with negligible probability*, $s_i = \tilde{s_i}$ if and only if $\mathcal{M}_1$ returns 1 on every query in $\mathcal{Q}_i(\tilde{s_i})$.*

Proof. Throughout the proof, we assume that $\boldsymbol{s} \in \mathcal{R}_q$, which occurs *w.o.p.* by the assumption $q > 1 + 2\alpha\sqrt{n}$. Moreover, let $\varDelta s_i \triangleq s_i - \tilde{s_i} \in \mathbb{F}_q$, and hence $s_i = \tilde{s_i}$ if and only if $\varDelta s_i = 0$. Finally, by assumption there exists a $t \in [2n]$ such that $g^d = \omega_i^t$ and $\gcd(t, d) = 1$.

Recall that for the query $Q_k = \left(\boldsymbol{x}_k = k \cdot \boldsymbol{c}_i, \boldsymbol{w}_k = [w_{k,j}]_{j\in[n]}, \boldsymbol{z}_k = [z_{k,j}]_{j\in[n]}\right) \in \mathcal{Q}_i(\tilde{s_i})$, $\mathcal{M}_1$ first generates $\varepsilon_k \leftarrow \mathbb{Z}^n_{1+2\theta}$, and then computes

$$\begin{aligned} \boldsymbol{v}_k &\triangleq \boldsymbol{x}_k \cdot \boldsymbol{s} + q_0 \cdot \boldsymbol{w}_k + 2\varepsilon_k = ks_i \cdot \boldsymbol{c}_i + q_0 \cdot \boldsymbol{w}_k + 2\varepsilon_k \\ &\sim \begin{bmatrix} s_i \cdot k \cdot c_{i,1} + q_0 \cdot w_{k,1} \\ \vdots \\ s_i \cdot k \cdot c_{i,n} + q_0 \cdot w_{k,n} \end{bmatrix} + \begin{bmatrix} 2\varepsilon_{k,1} \\ \vdots \\ 2\varepsilon_{k,n} \end{bmatrix} \qquad (\varepsilon_{k,j} \triangleq \mu_j(\varepsilon_k)) \end{aligned}$$

$$= \begin{bmatrix} \Delta s_i \cdot kc_{i,1} + (\tilde{s_i} \cdot k \cdot c_{i,1} + q_0 \cdot w_{k,1}) \\ \vdots \\ \Delta s_i \cdot kc_{i,n} + (\tilde{s_i} \cdot k \cdot c_{i,n} + q_0 \cdot w_{k,n}) \end{bmatrix} + \begin{bmatrix} 2\varepsilon_{k,1} \\ \vdots \\ 2\varepsilon_{k,n} \end{bmatrix}$$
$$= \begin{bmatrix} \Delta s_i \cdot kc_{i,1} + u_{k,1} + \mathsf{Cha}\,(u_{k,1}) \cdot q_0 \\ \vdots \\ \Delta s_i \cdot kc_{i,n} + u_{k,n} + \mathsf{Cha}\,(u_{k,n}) \cdot q_0 \end{bmatrix} + \begin{bmatrix} 2\varepsilon_{k,1} \\ \vdots \\ 2\varepsilon_{k,n} \end{bmatrix};$$

Finally, $\mathcal{M}_1$ returns 1 if $\mathsf{Parity}\,(v_{k,j}) = z_{k,j}$ for every $j \in [n]$, where $v_{k,j} \triangleq \mu_j(\boldsymbol{v}_k)$; Otherwise, $\mathcal{M}_1$ returns 0.

Notice that we always have $u_{k,j} + \mathsf{Cha}\,(u_{k,j}) \cdot q_0 \in \mathsf{zone}_0$ by Fact 6(a). Also, it follows from the inequality $q > 1 + 8\theta$ that we have $-q_0/2 < 2\varepsilon_{k,j} < q_0/2$.

First consider the simple case when $\Delta s_i = 0$. Since the short even noise $2\varepsilon_{k,j}$ satisfy $-q_0/2 < 2\varepsilon_{k,j} < q_0/2$, according to Fact 6(b), it is routine to verify that $\mathcal{M}_1$ returns 1 on every Q_k by definition.

In the sequel, we assume $\Delta s_i \neq 0$. Define the set

$$\mathsf{offset}(\Delta s_i) \triangleq \{\Delta s_i \cdot kc_{i,j} \,|\, k \in S_g, j \in [n]\},$$

and we *claim* that $\{-1, 1\} \cap \mathsf{offset}(\Delta s_i) \neq \emptyset$, *i.e.*, either $1 \in \mathsf{offset}(\Delta s_i)$ or $-1 \in \mathsf{offset}(\Delta s_i)$. Since $c_{i,j} = c_{i,n} \cdot \omega_i^{n-j}$ by Lemma 5, we have

$$\Delta s_i \cdot k \cdot c_{i,j} = \Delta s_i c_{i,n} \cdot k \cdot \omega_i^{n-j}.$$

Let $\Delta s_i c_{i,n} = g^{e^*}$ where $e^* \in [q-1]$. Clearly there exists a $r^* \in [d]$ such that $d \mid (e^* + r^*)$, and $(e^* + r^*)/d \in [2n]$. Let $k^* \triangleq g^{r^*} \in S_g$. Then

$$\Delta s_i c_{i,n} \cdot k^* \cdot \omega_i^{n-j} = g^{e^* + r^*} \cdot \omega_i^{n-j} = \omega_i^{t(e^*+r^*)/d + n - j}.$$

It is easy to see there exists a $j^* \in [n]$ such that either $t(e^* + r^*)/d + n - j^* \equiv n \pmod{2n}$ or $t(e^* + r^*)/d + n - j^* \equiv 0 \pmod{2n}$, or equivalently, either $\Delta s_i \cdot c_{i,n} \cdot k^* \cdot \omega_i^{n-j^*} = \omega_i^n = -1 \in \mathbb{F}_q$ or $\Delta s_i \cdot c_{i,n} \cdot k^* \cdot \omega_i^{n-j^*} = \omega_i^0 = 1 \in \mathbb{F}_q$.

When $\Delta s_i \cdot k^* \cdot c_{i,j^*} = \pm 1$, it is easy to verify that $z_{k^*,j^*} \neq \mathsf{Parity}\,(v_{k^*,j^*})$ by Fact 6(c). Equivalently, the associated j^*-th equality of Q_{k^*} does not hold, and $\mathcal{M}_1$ returns 0 on the query $Q_{k^*} \in \mathcal{Q}_i(\tilde{s_i})$. □

The following theorem follows immediately from Lemma 7.

Theorem 8. *When $q > 1 + \max\{8\theta,\ 2\alpha\sqrt{n}\}$, the efficient attacker $\mathcal{A}_1$ defined previously can recover the secret $\boldsymbol{s}$ of $\mathcal{M}_1$* with overwhelming probability, *by making at most $n \cdot q \cdot \frac{q-1}{2n} = \mathrm{poly}(\lambda)$ queries to $\mathcal{M}_1$.* □

Remarks First, the proof of Lemma 7 implies that the exact distribution form of the noise term ε generated by $\mathcal{M}_1$ does not affect the correctness of $\mathcal{A}_1$; Only its support does. This explains why the noise term in $\mathcal{M}_1$ is simply defined to be drawn from the *uniform* distribution over its support $\mathbb{Z}_{1+2\theta}^n$. Moreover, it is not hard to see that the index set $S_g = \{g^r \,|\, r \in [d]\}$ could be replaced by *any* complete system of representatives of cosets in the quotient group $\mathbb{F}_q^\times/G$, where $G \triangleq \langle\omega_i\rangle$ is *the unique* subgroup of the cyclic $\mathbb{F}_q^\times$ satisfying $|G| = 2n$. All these observations show that s a concrete instance of SFA, the basic aattacker $\mathcal{A}_1$ is *versatile*.

The success of $\mathcal{A}_1$ relies heavily on the notion of the CRT basis $\boldsymbol{c}_1, \cdots, \boldsymbol{c}_n$ for $\mathcal{R}_q$ as well as its property, which explains why our attackers ($\mathcal{A}_1$ as well as its improved variants) are called *small field attackers*. Computer experiments have justified the correctness of our small field attacker $\mathcal{A}_1$ against $\mathcal{M}_1$; In particular, $\mathcal{A}_1$ succeeds when the oracle $\mathcal{M}_1$ is instantiated with these four groups of suggested parameters in [ZZD+15,ZZDS14] ($\theta := n\alpha\gamma$).

The existence of $\mathcal{A}_1$ implies that there exists an efficient attacker with public key $\mathbf{0} \in \mathcal{R}_q$ that can recover the private key of $\mathcal{M}_0$ *w.o.p.* Hence, $\mathcal{A}_1$ itself suffices to show the vulnerability of Π_1.

In essence, our small field attack against Π_1 is equivalent to solving a problem that is much *harder* than the *adaptive* variant of RLWE problem, where attacker can adaptively choose the $\boldsymbol{a}$-part in each sample $(\boldsymbol{a}, \boldsymbol{b})$. Although the *adaptive* RLWE can be solved without the CRT basis of $\mathcal{R}_q$, the CRT basis is *necessary* and plays an *essential* role in our SFA.

Although we assume $s_1, \cdots, s_{i-1} \in \mathbb{F}_q$ are known before the i-th round of $\mathcal{A}_1$, such assumption is unnecessary for $\mathcal{A}_1$. Nevertheless, this assumption is *essential* for us to construct an improved variant of $\mathcal{A}_1$, as we shall see in Section 5.

5 Making Small Field Attack "Undetectable"

In this section, we continue our analysis on the oracle $\mathcal{M}_1$, and show that we can construct an efficient hybrid attacker $(\mathcal{V}/\mathcal{A}'_1)_\delta$ against $\mathcal{M}_1$, whose malicious queries are as "random-looking" as possible.

5.1 The Improved Attacker $\mathcal{A}'_1$ Against $\mathcal{M}_1$

Motivating question #1 The success of $\mathcal{A}_1$ relies on the assumption that $\mathcal{M}_1$ imposes no restrictions on the incoming queries. Intuitively, for every query made by $\mathcal{A}_1$, the $\boldsymbol{w}$- and $\boldsymbol{z}$-entries seem "random" enough; However, the algebraic structure of $\boldsymbol{x}$-entry is rather simple, as the $\boldsymbol{x}$-entry always falls into the set $\{k\boldsymbol{c}_i \mid k \in \mathbb{F}_q, i \in [n]\}$ with size $nq \ll q^n$. Hence, it is easy for $\mathcal{M}_1$ to identify those malicious queries made by $\mathcal{A}_1$, and the attacker $\mathcal{A}_1$ will no longer work if the oracle $\mathcal{M}_1$ additionally requires that $\boldsymbol{x} \notin \{k\boldsymbol{c}_i \mid k \in \mathbb{F}_q, i \in [n]\}$. Such *additional* requirement seems *reasonable*, given that $nq \ll q^n$.

Thus, our *first motivating question* arises: can we improve $\mathcal{A}_1$ so that it can still recover the secret of $\mathcal{M}_1$, even if the aforementioned requirement is imposed by $\mathcal{M}_1$? The answer is affirmative, as we shall see later.

5.2 Construction of $\mathcal{A}'_1$

Nevertheless, we can construct an improved variant of $\mathcal{A}_1$, *i.e.*, $\mathcal{A}'_1$, to resolve the issue propsed by the motivating question #1. Its general structure is similar to that of $\mathcal{A}_1$, and the only difference lies in the definition of the query set in each round.

The design of query set in $\mathcal{A}'_1$ relies on the following two observations. First, recall that in the i-th round of $\mathcal{A}_1$, $s_1, \cdots, s_{i-1} \in \mathbb{F}_q$ are assumed to be known already. Thus, we can used these known CRT-coefficients of $\boldsymbol{s} \in \mathcal{R}_q$ to re-design the $\boldsymbol{x}$-entry so that it looks much more "complex". Moreover, by Lemma 1, the attack still succeeds if we add a "small" *even* noise into the $\boldsymbol{x}$-entry, *provided that q is sufficiently large*. In sum, in the i-th round, for every query made by $\mathcal{A}'_1$ to $\mathcal{M}_1$, the $\boldsymbol{x}$-entry is of the form

$$\boldsymbol{x} = k\boldsymbol{c}_i + \boldsymbol{h} + 2\boldsymbol{e},$$

where $\boldsymbol{h} \leftarrow \left\{\boldsymbol{u} \in \mathcal{R}_q \,\middle|\, \mathsf{Dim}\left(\boldsymbol{u}\right) = [i-1]\right\}$, and $\boldsymbol{e} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. Intuitively, the introduction of $\boldsymbol{h}$- and $\boldsymbol{e}$-parts into the $\boldsymbol{x}$-entry is to make the queries made by $\mathcal{A}'_1$ as "random-looking" as possible. Of course, this improvement asks us to re-design the settings of $\boldsymbol{w}$- and $\boldsymbol{z}$-entries appropriately.

It should be stressed that the $\boldsymbol{h}$-part cannot be drawn from the uniform distribution over $\mathcal{R}_q$: when $\boldsymbol{h} \leftarrow \mathcal{R}_q$, it is very likely that $\mathsf{Dim}\left(\boldsymbol{hs}\right) \not\subseteq [i]$, making it *impossible* for us to set $\boldsymbol{w}$- and $\boldsymbol{z}$-entries appropriately.

The following lemma characterizes the query set $\mathcal{Q}'_i(\tilde{s}_i)$ used by $\mathcal{A}'_1$ in its i-th round.

Lemma 9. *Let $g \in \mathbb{F}_q^\times$ denote a primitive element of $\mathbb{F}_q$, and let $S_g \triangleq \{g^r \mid r \in [d]\}$ and $d \triangleq \frac{q-1}{2n}$. Define*

$$\mathcal{Q}'_i(\tilde{s}_i) = \left\{ Q'_k = \left(k \cdot \boldsymbol{c}_i + \boldsymbol{h}_k + 2\boldsymbol{e}_k, [w_{k,j}]_{j\in[n]}, [z_{k,j}]_{j\in[n]}\right) \,\middle|\, \begin{array}{c} k \in S_g,\ j \in [n],\ h_{k,1}, \cdots, h_{k,i-1} \leftarrow \mathbb{F}_q^\times, \\ \boldsymbol{h}_k = \sum_{r\in[i-1]} h_{k,r}\boldsymbol{c}_r, \boldsymbol{e}_k \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}, \\ u_{k,j} = \tilde{s}_i \cdot kc_{i,j} + \sum_{r\in[i-1]} s_r h_{k,r} c_{r,j}, \\ w_{k,j} = \mathsf{Cha}\left(u_{k,j}\right), z_{k,j} = \mathsf{Mod}\left(u_{k,j}, w_{k,j}\right) \end{array} \right\}.$$

If $q > 1 + 8(\theta + n\alpha\alpha')$, then except with negligible probability, *$s_i = \tilde{s}_i$ if and only if $\mathcal{M}_1$ returns 1 on every query in $\mathcal{Q}'_i(\tilde{s}_i)$.*

Proof. Let $\Delta s_i \triangleq s_i - \tilde{s}_i \in \mathbb{F}_q$. Moreover, for $\boldsymbol{s} \leftarrow D_{\mathbb{Z}^n,\alpha}$ and $\boldsymbol{e}_k \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$, define $\varepsilon'_k \triangleq \boldsymbol{s} \cdot \boldsymbol{e}_k \in \mathcal{R}_q$; By Lemma 1, the inequality $\|\varepsilon'_k\|_\infty \leq n\alpha\alpha'$ holds *w.o.p.*

For the query $Q'_k = \left(\boldsymbol{x}_k = k \cdot \boldsymbol{c}_i + \boldsymbol{h}_k + 2\boldsymbol{e}_k,\ \boldsymbol{w}_k = [w_{k,j}]_{j\in[n]},\ \boldsymbol{z}_k = [z_{k,j}]_{j\in[n]}\right) \in \mathcal{Q}'_i(\tilde{s}_i)$, $\mathcal{M}_1$ first generates $\varepsilon_k \leftarrow \mathbb{Z}^n_{1+2\theta} = \{-\theta, \cdots, \theta\}^n$, and then computes

$$\begin{aligned} \boldsymbol{v}_k &\triangleq \boldsymbol{s} \cdot \boldsymbol{x}_k + q_0\boldsymbol{w}_k + 2\varepsilon_k \\ &= \boldsymbol{s} \cdot (k\boldsymbol{c}_i + \boldsymbol{h}_k + 2\boldsymbol{e}_k) + q_0\boldsymbol{w}_k + 2\varepsilon_k \\ &= k\Delta s_i\boldsymbol{c}_i + \left(k\tilde{s}_i\boldsymbol{c}_i + \sum_{r\in[i-1]} s_r h_{k,r}\boldsymbol{c}_r + q_0\boldsymbol{w}_k\right) + 2(\boldsymbol{s}\boldsymbol{e}_k + \varepsilon_k) \end{aligned}$$

$$
\begin{aligned}
&\sim \begin{bmatrix} \mathit{\Delta} s_i k c_{i,1} + \left(k\tilde{s}_i c_{i,1} + \sum_{r\in[i-1]} s_r h_{k,r} c_{r,1} + q_0 w_{k,1}\right) \\ \vdots \\ \mathit{\Delta} s_i k c_{i,n} + \left(k\tilde{s}_i c_{i,n} + \sum_{r\in[i-1]} s_r h_{k,r} c_{r,n} + q_0 w_{k,n}\right) \end{bmatrix} + \begin{bmatrix} 2(\varepsilon'_{k,1} + \varepsilon_{k,1}) \\ \vdots \\ 2(\varepsilon'_{k,n} + \varepsilon_{k,n}) \end{bmatrix} \\
&= \begin{bmatrix} \mathit{\Delta} s_i \cdot k c_{i,1} + u_{k,1} + \mathsf{Cha}\left(u_{k,1}\right) \cdot q_0 \\ \vdots \\ \mathit{\Delta} s_i \cdot k c_{i,n} + u_{k,n} + \mathsf{Cha}\left(u_{k,n}\right) \cdot q_0 \end{bmatrix} + \begin{bmatrix} 2(\varepsilon'_{k,1} + \varepsilon_{k,1}) \\ \vdots \\ 2(\varepsilon'_{k,n} + \varepsilon_{k,n}) \end{bmatrix},
\end{aligned}
$$

where $\varepsilon_{k,j} \triangleq \mu_j(\varepsilon_k)$ and $\varepsilon'_{k,j} \triangleq \mu_j(\boldsymbol{s}\boldsymbol{e}_k)$. Finally, if every $\mathsf{Parity}\left(v_{k,j}\right) = z_{k,j}$ where $v_{k,j} \triangleq \mu_j(\boldsymbol{v}_k)$, then $\mathcal{M}_1$ returns 1; Otherwise, $\mathcal{M}_1$ returns 0.

Since $q > 1 + 8 \cdot (\theta + n\alpha\alpha')$, we have for every $j \in [n]$ that

$$
\left|\varepsilon'_{k,j} + \varepsilon_{k,j}\right| \le \left|\varepsilon'_{k,j}\right| + \left|\varepsilon_{k,j}\right| \le n\alpha\alpha' + \theta < \frac{q-1}{8}.
$$

In such setting, similar to the proof of Lemma 7, it is not hard to verify that *except with negligible probability*,

- If $\mathit{\Delta} s_i = 0$, then $\mathcal{M}_1$ returns 1 on *every* $Q'_k \in \mathcal{Q}'_k(\tilde{s}_i)$ by Fact 6(b); And
- If $\mathit{\Delta} s_i \neq 0$, then $\mathcal{M}_1$ returns 0 on *some* $Q'_{k^*} \in \mathcal{Q}'_k(\tilde{s}_i)$ by Fact 6(c). □

The success of $\mathcal{A}'_1$ is summarized in the following theorem. See Appendix B.1 for the pseudocode of $\mathcal{A}'_1$.

Theorem 10. *When $q > 1 + 8(\theta + n\alpha\alpha')$, the efficient algorithm $\mathcal{A}'_1$ can recover the secret of $\mathcal{M}_1$* w.o.p.*, by making at most $n \cdot q \cdot \frac{q-1}{2n}$ queries to $\mathcal{M}_1$. Furthermore, in the i-th round, for every query $(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z})$ made by $\mathcal{A}'_1$, the $\boldsymbol{x}$-entry is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$, where* $\mathsf{Dim}\left(\boldsymbol{x}_0\right) = [i]$ *and $\boldsymbol{e} \in \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$.* □

5.3 The "Undetectable" Attacker $\left(\mathcal{V}/\mathcal{A}'_1\right)_\delta$ Against $\mathcal{M}_1$

Motivating question #2 We have defined an improved efficient attacker $\mathcal{A}'_1$ against $\mathcal{M}_1$. To us, the most *practical* way for $\mathcal{M}_1$ to identify those malicious queries made by $\mathcal{A}'_1$ is to analyze the algebraic structure of the $\boldsymbol{x}$-entry. And the $\boldsymbol{h}$- and $\boldsymbol{e}$-parts were introduced to complicate the algebraic structure of $\boldsymbol{x}$. Clearly, the more CRT coefficients of $\boldsymbol{s}$ we get, the more difficult for $\mathcal{M}_1$ to distinguish those queries made by $\mathcal{A}'_1$ from ordinary ones.

However, there is still a problem: when $\mathcal{A}'_1$ seeks to recover the *first* CRT-coefficient of $\boldsymbol{s}$ in its first round, $\boldsymbol{h} = \boldsymbol{0}$ and hence the $\boldsymbol{x}$-entry falls into the set $\left\{k \cdot \boldsymbol{c}_i + 2\boldsymbol{e} \,\middle|\, k \in \mathbb{F}_q, i \in [n], \boldsymbol{e} \in \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}\right\} \subseteq \mathcal{R}_q$, which is of "small" size compared with $\mathcal{R}_q$. It is not hard for $\mathcal{M}_1$ to identify, and thus reject, the *first* set of queries, and such rejection sounds "reasonable" for $\mathcal{M}_1$, given this set is of "small" size relative to $\mathcal{R}_q$. If such similar restrictions are imposed by $\mathcal{M}_1$, $\mathcal{A}'_1$ would fail since it cannot recover the first (and hence the remaining) CRT-coefficient of the secret $\boldsymbol{s}$.

Thus, our *second motivating question* arises: can we improve $\mathcal{A}'_1$ so that it can still recover the secret of $\mathcal{M}_1$, even if the aforementioned requirement is imposed by $\mathcal{M}_1$? The answer is affirmative, too, and we shall construct such a desired efficient attackers $(\mathcal{V}/\mathcal{A}'_1)_\delta$ in this subsection.

The hybrid attacker $(\mathcal{V}/\mathcal{A}'_1)_\delta$ Let $\mathcal{P}_1$ denote the problem of recovering the secret of $\mathcal{M}_1$. Similarly, we can define a related problem $\mathcal{P}_2$ as follows: given a nonempty index set $I \subseteq [n]$, $[\tilde{s}_i]_{i\in I} \in \mathbb{F}_q^{|I|}$, and oracle access to $\mathcal{M}_1$, decide whether $[s_i]_{i\in I} = [\tilde{s}_i]_{i\in I}$ or not, or more precisely, whether $\tilde{s}_i = s_i$ for every $i \in I$. Recall that $s_i = \eta_i(\boldsymbol{s})$, where $\boldsymbol{s}$ is the secret of $\mathcal{M}_1$. In this subsection, we shall construct an efficient solver $\mathcal{V}$ against the problem $\mathcal{P}_2$. Jumping ahead, to solve the instance $(I, [\tilde{s}_i]_{i\in I})$ of $\mathcal{P}_2$, for every query made by $\mathcal{V}$ to $\mathcal{M}_1$, the $\boldsymbol{x}$-entry is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$, where $\mathsf{Dim}\left(\boldsymbol{x}_0\right) = I$ and $\boldsymbol{e} \in \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$.

With the help of $\mathcal{V}$, we can construct a *hybrid* efficient attacker against $\mathcal{M}_1$, which consists of two *consecutive* phases as follows:

Phase 1 First, choose a *constant* δ of moderately large, and an index set $I \subseteq [n]$ of size δ *randomly*. Then, feed $\mathcal{V}$ with q^δ instances, each of the form $(I, [\tilde{s}_i]_{i\in I})$, $\tilde{s}_i \in \mathbb{F}_q$. In this manner, the CRT-coefficients $s_i, i \in I$, would be recovered successfully when $[\tilde{s}_i]_{i\in I}$ runs over the set $\mathbb{F}_q^\delta$.

In particular, the $\boldsymbol{x}$-entry of every query made in this phase is always of the form $\boldsymbol{x}_0+2\boldsymbol{e}$, where $\mathsf{Dim}\left(\boldsymbol{x}_0\right) = I$, and $\boldsymbol{e} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. Given the randomness of I, it is *hard in practice* for $\mathcal{M}_1$ to identify those queries made by $\mathcal{V}$.

Phase 2 This phase consists of $n - \delta$ rounds, each devoted to recovering one of the remaining $n - \delta$ CRT-coefficients of $\boldsymbol{s}$, as is done in $\mathcal{A}_1'$.
In particular, the $\boldsymbol{x}$-entry of every query made in this phase is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$ where $\mathsf{Dim}\left(\boldsymbol{x}_0\right) \supseteq I$ and $\boldsymbol{e} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$, making it *harder in practice* for $\mathcal{M}_1$ to identify them.

The notation $(\mathcal{V}/\mathcal{A}_1')_\delta$ is applied to emphasize the structure of this attacker.

Some remarks are in order. First, we stress that due to the randomness of the index set I, this makes it *hard in practice* for $\mathcal{M}_1$ to identify (and hence reject) those queries made by $(\mathcal{V}/\mathcal{A}_1')_\delta$. the $(\mathcal{V}/\mathcal{A}_1')_\delta$ shows that we can construct an "undetectable" attacker with static public key $\boldsymbol{0} \in \mathcal{R}_q$ that can efficiently recover the static private key of party j in Π_1.

Moreover, notice that the algorithms $\mathcal{A}_1'$ and $\mathcal{V}$ are firmly related to each other: when $\mathcal{M}_1$ imposes no restriction on the incoming queries, $\mathcal{A}_1$ could be adapted to solve the problem $\mathcal{P}_2$ efficiently, and $\mathcal{V}$ can be used to recover the *whole* secret $\boldsymbol{s}$ of $\mathcal{M}_1$ efficiently as well.

Finally, notice that we could have used $\mathcal{V}$ to recover the other CRT-coefficients of the secret in Phase 2. However, this is *less efficient*: it takes more queries *on average* for $\mathcal{V}$ to recover one CRT-coefficient of $\boldsymbol{s}$ than $\mathcal{A}_1'$ does, as we shall see.

General structure of $\mathcal{V}$ We are about to design the desired $\mathcal{V}$. To simplify the following discussion, we only consider the *special* case where $I = [n]$, and it is easy to generalize to the more usual case where I is a *proper* subset of $[n]$.

First come some notations. Choose $k \leftarrow [n]$ randomly. Let $\Delta s_i \triangleq s_i - \tilde{s_i}$ for every $i \in [n]$. For every $j \in [n]$, define $\boldsymbol{a}_j \triangleq [\tilde{s_i} \cdot c_{i,j}]_{i\in[n]} \in \mathbb{F}_q^n$, and $\boldsymbol{b}_j \triangleq [\Delta s_i \cdot c_{i,j}]_{i\in[n]} \in \mathbb{F}_q^n$; Moreover, define $A_j(\boldsymbol{u}) \triangleq \langle \boldsymbol{u}, \boldsymbol{a}_j \rangle \in \mathbb{F}_q$, and $B_j(\boldsymbol{u}) \triangleq \langle \boldsymbol{u}, \boldsymbol{b}_j \rangle \in \mathbb{F}_q$, where $\boldsymbol{u} \in \mathbb{F}_q^n$. Define the $\mathbb{F}_q$-vector space $U_k \triangleq \{r \cdot \boldsymbol{a}_k \,|\, r \in \mathbb{F}_q\} \subseteq \mathbb{F}_q^n$. By definition, every U_k is a 1-dimensional subspace of $\mathbb{F}_q^n$, and its orthogonal complement is the $(n-1)$-dimensional subspace $U_k^\perp \triangleq \left\{\boldsymbol{v} \in \mathbb{F}_q^n \,\middle|\, A_k(\boldsymbol{v}) = 0 \in \mathbb{F}_q\right\} \subseteq \mathbb{F}_q^n$. Choose an $\mathbb{F}_q$-basis for $U_k^\perp$ *randomly*, say $\boldsymbol{F}_k \triangleq \left\{\boldsymbol{f}_{k,1}, \cdots, \boldsymbol{f}_{k,n-1}\right\} \subseteq U_k^\perp$ such that every entry of every $\boldsymbol{f}_{k,i}$ is non-zero. With U_k, the set $\mathbb{F}_q^n$ is partitioned into three parts: $S_1 \triangleq \mathbb{F}_q^n \setminus U_k$, $S_2 \triangleq U_k \setminus \{\boldsymbol{0}\}$, and $S_3 \triangleq \{\boldsymbol{0}\}$. Finally, for every $(t, \boldsymbol{u}) \in \mathbb{F}_q \times \mathbb{F}_q^n$, let $\tau(t, \boldsymbol{u})$ denote $\sum_{i\in[n]} t\mu_i(\boldsymbol{u}) \cdot \boldsymbol{c}_i \in \mathcal{R}_q$.

Some remarks are in order. First, notice that for every $i, j \in [n]$, s_i, Δs_i and $\boldsymbol{b}_j$ are unknown to us. Moreover, although the map $A_j(\cdot)$ is efficiently computable, this is *not* true for $B_j(\cdot)$. Finally, recall that every $c_{i,j} \neq 0$ by Lemma 5, so the guess $[s_i]_{i\in[n]} = [\tilde{s_i}]_{i\in[n]}$ is correct if and only if $\boldsymbol{b}_k = \boldsymbol{0} \in \mathbb{F}_q^n$. Since $\boldsymbol{0} \in S_3 \subseteq U_k$ trivially, a *necessary yet insufficient* condition for $\boldsymbol{b}_k = \boldsymbol{0}$ is: $\boldsymbol{b}_k \in U_k = S_2 \cup S_3$, or equivalently, $0 = \left\langle \boldsymbol{f}_{k,i}, \boldsymbol{b}_k \right\rangle = B_k\left(\boldsymbol{f}_{k,i}\right)$ for every $i \in [n-1]$.

The general idea behind $\mathcal{V}$ is simple: it first makes the guess, *i.e.*, $[s_i]_{i\in[n]} = [\tilde{s_i}]_{i\in[n]}$, and then verifies the correctness of the guess via a set $\mathcal{Q} \triangleq \mathcal{Q}_1 \cup \mathcal{Q}_2$ of queries to $\mathcal{M}_1$ such that *except with negligible probability*, the guess is correct if and only if $\mathcal{M}_1$ returns 1 on *every* query in $\mathcal{Q}$.

In more detail, $\mathcal{V}$ consists of two consecutive phases: Phase 1 and Phase 2.

- By issuing a set $\mathcal{Q}_1$ of queries to $\mathcal{M}_1$, *Phase 1* is devoted to deciding whether $\boldsymbol{b}_k \in U_k = S_2 \cup S_3$ or not;
- Conditioned on $\boldsymbol{b}_k \in U_k$ and hence $\boldsymbol{b}_k = r_0 \cdot \boldsymbol{a}_k$ for some $r_0 \in \mathbb{F}_q$, *Phase 2* is to decide whether $r_0 = 0$ or not, by a set $\mathcal{Q}_2$ of queries to $\mathcal{M}_1$.

It remains to design $\mathcal{Q} = \mathcal{Q}_1 \cup \mathcal{Q}_2$. Jumping ahead, for every query $(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z})$ in $\mathcal{Q}$, the $\boldsymbol{x}$-entry is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$, where $\mathsf{Dim}\left(\boldsymbol{x}_0\right) = I = [n]$, and $\boldsymbol{e} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. Similar to that of $\mathcal{A}_1'$, the $\boldsymbol{e}$-part is introduced here to make those queries made by $\mathcal{V}$ as "random-looking" as possible.

Design of Phase 1 Jumping ahead, the query set $\mathcal{Q}_1 = \mathcal{Q}_1(\boldsymbol{F}_k)$ is

$$\mathcal{Q}_1(\boldsymbol{F}_k) \triangleq \left\{Q_{t,i} = (\tau(t, \boldsymbol{f}_{k,i}) + 2\boldsymbol{e}_{t,i}, \boldsymbol{w}_{t,i}, \boldsymbol{z}_{t,i}) \,\middle|\, t \in [q_0], i \in [n-1],\ \boldsymbol{e}_{t,i} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}\right\}.$$

It remains to set the $\boldsymbol{w}_{t,i}$- and $\boldsymbol{z}_{t,i}$-entries.

Observe that for the query $Q_{t,i} = (\tau(t, \boldsymbol{f}_{k,i}) + 2\boldsymbol{e}_{t,i}, \boldsymbol{w}_{t,i} = [w_{t,i,j}]_{j\in[n]}, \boldsymbol{z}_{t,i} = [z_{t,i,j}]_{j\in[n]}) \in \mathcal{Q}_1(\boldsymbol{F}_k)$, $\mathcal{M}_1$ first generates $\varepsilon_{t,i} \leftarrow \mathbb{Z}^n_{1+2\theta}$, and then computes

$$\begin{aligned} \boldsymbol{v}_{t,i} &\triangleq \boldsymbol{s} \cdot (\tau(t, \boldsymbol{f}_{k,i}) + 2\boldsymbol{e}_{t,i}) + q_0 \cdot \boldsymbol{w}_{t,i} + 2\varepsilon_{t,i} \\ &= \boldsymbol{s} \cdot \tau(t, \boldsymbol{f}_{k,i}) + q_0 \cdot \boldsymbol{w}_{t,i} + 2(\varepsilon_{t,i} + \varepsilon'_{t,i}) \quad (\varepsilon'_{t,i} \triangleq \boldsymbol{s} \cdot \boldsymbol{e}_{t,i} \sim [\varepsilon'_{t,i,j}]_{j\in[n]}) \end{aligned}$$

$$\sim \begin{bmatrix} t\cdot A_1(\boldsymbol{f}_{k,i}) + t\cdot B_1(\boldsymbol{f}_{k,i}) + q_0\cdot w_{t,i,1} \\ \vdots \\ t\cdot A_k(\boldsymbol{f}_{k,i}) + t\cdot B_k(\boldsymbol{f}_{k,i}) + q_0\cdot w_{t,i,k} \\ \vdots \\ t\cdot A_n(\boldsymbol{f}_{k,i}) + t\cdot B_n(\boldsymbol{f}_{k,i}) + q_0\cdot w_{t,i,n} \end{bmatrix} + \begin{bmatrix} 2(\varepsilon_{t,i,1}+\varepsilon'_{t,i,1}) \\ \vdots \\ 2(\varepsilon_{t,i,k}+\varepsilon'_{t,i,k}) \\ \vdots \\ 2(\varepsilon_{t,i,n}+\varepsilon'_{t,i,n}) \end{bmatrix}.$$

Notice that when $q > 1 + 8(\theta + n\alpha\alpha')$, the noise $(\varepsilon_{t,i} + \varepsilon'_{t,i})$ is "short" in the sense that $\|\varepsilon_{t,i} + \varepsilon'_{t,i}\|_\infty < \frac{q-1}{8}$ holds *w.o.p.*

And this definition is justified by the following lemma.

Lemma 11. *Define*

$$\mathcal{Q}_1(\boldsymbol{F}_k) \triangleq \left\{ Q_{t,i} = \left(\tau(t, \boldsymbol{f}_{k,i}) + 2\boldsymbol{e}_{t,i}, [w_{t,i,j}]_{j\in[n]}, [z_{t,i,j}]_{j\in[n]}\right) \;\middle|\; \begin{array}{c} t\in[q_0],\ i\in[n-1],\ j\in[n], \boldsymbol{e}_{t,i} \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}, \\ u_{t,i,j} = tA_j(\boldsymbol{f}_{k,i}),\ w_{t,i,j} = \mathsf{Cha}\left(u_{t,i,j}\right), \\ z_{t,i,j} = \mathsf{Mod}\left(u_{t,i,j}, w_{t,i,j}\right) \end{array} \right\}.$$

When $q > 1 + 8(\theta + n\alpha\alpha')$, except with negligible probability*, we have*

(a) If $\mathcal{M}_1$ returns 0 *on* some *queries in $\mathcal{Q}_1(\boldsymbol{F}_k)$, then $\mathbf{b}_k \neq \mathbf{0}$ and hence $[s_i]_{i\in[n]} \neq [\tilde{s}_i]_{i\in[n]}$;*
(b) If $\mathcal{M}_1$ returns 1 *on* every *query in $\mathcal{Q}_1(\boldsymbol{F}_k)$, then $\mathbf{b}_k \in U_k = S_2 \cup S_3$.*

Proof. First, if $\boldsymbol{b}_k \in S_3 = \{\mathbf{0}\}$, then our guess is correct, every $\boldsymbol{b}_j = \mathbf{0}$ and hence every $B_j(\cdot) = 0$; By Fact 6(b), $\mathcal{M}_1$ returns 1 *w.o.p.* for every query in $\mathcal{Q}_1(\boldsymbol{F}_k)$.

Moreover, if $\boldsymbol{b}_k \in S_1 = \mathbb{F}_q^n \setminus U_k$, then there exists $\boldsymbol{f}_{k,i^*} \in \boldsymbol{F}_k$ such that $B_k(\boldsymbol{f}_{k,i^*}) \neq 0$; Moreover, there exists a $t^* \in [q_0]$ such that $t^* \cdot B_k(\boldsymbol{f}_{k,i^*}) = \pm 1$. By Fact 6(c), $\mathcal{M}_1$ returns 0 *w.o.p.* for at least one query in $\mathcal{Q}_1(\boldsymbol{F}_k)$. □

It should be noted that, in Phase 1, it is *difficult* to analyze the distribution of query replies when $\boldsymbol{b}_k \in S_2$, which explains the necessity of Phase 2.

Design of Phase 2 In Phase 2, conditioned on the hypothesis $\boldsymbol{b}_k \in U_k = S_2 \cup S_3$, it remains to consider whether $\boldsymbol{b}_k \in S_2$ or $\boldsymbol{b}_k \in S_3 = \{\mathbf{0}\}$. By hypothesis, we have $\boldsymbol{b}_k \in U_k = \{r \cdot \boldsymbol{a}_k \,|\, r \in \mathbb{F}_q\}$; Hence, we can assume $\boldsymbol{b}_k = r_0 \cdot \boldsymbol{a}_k$ for some $r_0 \in \mathbb{F}_q$. Then $B_k(\boldsymbol{u}) = r_0 \cdot A_k(\boldsymbol{u})$ for every $\boldsymbol{u} \in \mathbb{F}_q^n$. Moreover, our guess now could be expressed in terms of r_0, *i.e.*, whether $r_0 = 0$ or not.

Choose $\boldsymbol{u}^* \leftarrow \mathcal{R}_q$ *randomly* such that $A_k(\boldsymbol{u}^*) = 1$ and *every entry of $\boldsymbol{u}^*$ is non-zero*. It follows $t \cdot A_k(\boldsymbol{u}^*) + t \cdot B_k(\boldsymbol{u}^*) = t(1 + r_0)$ for every $t \in \mathbb{F}_q$. Jumping ahead, the set $\mathcal{Q}_2 = \mathcal{Q}_2(\boldsymbol{u}^*)$ is

$$\mathcal{Q}_2(\boldsymbol{u}^*) \triangleq \left\{ Q'_t = (\tau(t, \boldsymbol{u}^*) + 2\boldsymbol{e}_t, \boldsymbol{w}_t, \boldsymbol{z}_t) \,\middle|\, t \in [q_0], \boldsymbol{e}_t \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}} \right\}.$$

It remains to set the $\boldsymbol{w}_t$- and $\boldsymbol{z}_t$-entries.

Observe that for every query $Q'_t = (\tau(t, \boldsymbol{u}^*) + 2\boldsymbol{e}_t, \boldsymbol{w}_t = [w_{t,j}]_{j\in[n]}, \boldsymbol{z}_t = [z_{t,j}]_{j\in[n]})$, $\mathcal{M}_1$ first generates $\varepsilon_t \leftarrow \mathbb{Z}^n_{1+2\theta}$, and then computes

$$\begin{aligned} \boldsymbol{v}'_t &\triangleq \boldsymbol{s} \cdot (\tau(t, \boldsymbol{u}^*) + 2\boldsymbol{e}_t) + q_0 \cdot \boldsymbol{w}_t + 2\varepsilon_t \\ &= \boldsymbol{s} \cdot \tau(t, \boldsymbol{u}^*) + q_0 \cdot \boldsymbol{w}_t + 2(\varepsilon_t + \varepsilon'_t) \quad (\varepsilon'_t \triangleq \boldsymbol{s} \cdot \boldsymbol{e}_t \sim [\varepsilon'_{t,j}]_{j\in[n]}) \\ &\sim \begin{bmatrix} t\cdot A_1(\boldsymbol{u}^*) + t\cdot B_1(\boldsymbol{u}^*) + q_0 w_{t,1} \\ \vdots \\ t + t\cdot r_0 + q_0 w_{t,k} \\ \vdots \\ t\cdot A_n(\boldsymbol{u}^*) + t\cdot B_n(\boldsymbol{u}^*) + q_0 w_{t,n} \end{bmatrix} + \begin{bmatrix} 2(\varepsilon_{t,1} + \varepsilon'_{t,1}) \\ \vdots \\ 2(\varepsilon_{t,k} + \varepsilon'_{t,k}) \\ \vdots \\ 2(\varepsilon_{t,n} + \varepsilon'_{t,n}) \end{bmatrix}. \end{aligned}$$

Again, when $q > 1 + 8(\theta + n\alpha\alpha')$, the inequality $\|\varepsilon_t + \varepsilon'_t\|_\infty < \frac{q-1}{8}$ holds *w.o.p.* With this in mind, we can define

$$\mathcal{Q}_2(\boldsymbol{u}^*) \triangleq \left\{ Q'_t = (\tau(t, \boldsymbol{u}^*) + 2\boldsymbol{e}_t, [w_{t,j}]_{j\in[n]}, [z_{t,j}]_{j\in[n]}) \;\middle|\; \begin{array}{c} t\in[q_0],\ j\in[n],\ \boldsymbol{e}_t \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}},\ u_{t,j} = tA_j(\boldsymbol{u}^*), \\ w_{t,j} = \mathsf{Cha}\left(u_{t,j}\right),\ z_{t,j} = \mathsf{Mod}\left(u_{t,j}, w_{t,j}\right) \end{array} \right\}.$$

And the definition is justified by the following lemma.

Lemma 12. *With the notations defined previously, if* $q > 1 + 8(\theta + n\alpha\alpha')$ *and* $\mathbf{b}_k \in U_k$ *are guaranteed, then* except with negligible probability, *we have* $[s_i]_{i\in[n]} = [\tilde{s_i}]_{i\in[n]}$ *if and only if* $\mathcal{M}_1$ *returns* 1 *on every query in* $\mathcal{Q}_2(\boldsymbol{u}^*)$.

Proof. First, if $[s_i]_{i\in[n]} = [\tilde{s_i}]_{i\in[n]}$, then our guess is correct, $r_0 = 0$, and every $B_j(\cdot) = 0$. By Fact 6(b), $\mathcal{M}_1$ returns 1 on every query in $\mathcal{Q}_2(\boldsymbol{u}^*)$.

Conversely, if $r_0 \neq 0$, then there exists a $t^* \in [q_0]$ such that $t^* r_0 = \pm 1$. By Fact 6(c), for the specific query $Q'_{t^*} = (\tau(t^*, \boldsymbol{u}^*) + 2\boldsymbol{e}_{t^*}, \boldsymbol{w}_{t^*}, \boldsymbol{z}_{t^*}) \in \mathcal{Q}_2(\boldsymbol{u}^*)$, its associated k-th equality does not hold *w.o.p.* By definition, $\mathcal{M}_1$ returns 0 *w.o.p.* on this specific query $Q'_{t^*} \in \mathcal{Q}_2(\boldsymbol{u}^*)$. □

This finishes the construction of $\mathcal{V}$, as well as its correctness analysis, for the *special* case when the index set $I = [n]$. Clearly it takes at most $n \cdot q_0 = \mathrm{poly}(\lambda)$ queries for $\mathcal{V}$ to solve this *special* case of $\mathcal{P}_2$, indicating that $\mathcal{V}$ runs in polynomial time. See Appendix B.2 for the pseudocode of $\mathcal{V}$.

Moreover, it is easy to generalize the foregoing construction such that $\mathcal{V}$ could be applied to solve the more general case of $\mathcal{P}_2$, *i.e.*, $\emptyset \neq I \subsetneq [n]$. In general, the number of queries made by $\mathcal{V}$ is upper-bounded by $q_0 \cdot |I| = q_0\delta = \mathrm{poly}(\lambda)$.

Theorem 13. *Assume* $q > 1 + 8(\theta + n\alpha\alpha')$. *With the notations defined previously, given* $\emptyset \neq I \subseteq [n]$, $[\tilde{s_i}]_{i\in I}$ *and oracle access to* $\mathcal{M}_1$, *it takes at most* $q_0 \cdot |I| = q_0 \cdot \delta$ *queries for* $\mathcal{V}$ *to decide whether* $[s_i]_{i\in[n]} = [\tilde{s_i}]_{i\in[n]}$ *or not:* except with negligible probability, *the equality holds if and only if* $\mathcal{M}_1$ *returns* 1 *on every query in* $\mathcal{Q} = \mathcal{Q}_1 \cup \mathcal{Q}_2$. *In particular, for every query in* $\mathcal{Q}$, *its* $\boldsymbol{x}$*-entry could be written as* $\boldsymbol{x} = \boldsymbol{x}_0 + 2\boldsymbol{e}$ *satisfying* $\mathsf{Dim}\,(\boldsymbol{x}_0) = I$ *and* $\boldsymbol{e} \in \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. □

Theorem 14. *When* $q > 1 + 8(\theta + n\alpha\alpha')$, *there exists an efficient attacker* $(\mathcal{V}/\mathcal{A}'_1)_\delta$ *that can recover the secret of* $\mathcal{M}_1$ w.o.p., *after making* $q_0\delta \cdot q^\delta + (n-\delta) \cdot q \cdot \frac{q-1}{2n} = \mathrm{poly}(\lambda)$ *queries to* $\mathcal{M}_1$. *In particular, for every query made by* $(\mathcal{V}/\mathcal{A}'_1)_\delta$ *to* $\mathcal{M}_1$, *it* $\boldsymbol{x}$*-entry is always of the form* $\boldsymbol{x} = \boldsymbol{x}_0 + 2\boldsymbol{e}$ *where* $|\mathsf{Dim}\,(\boldsymbol{x}_0)| \geq \delta$ *and* $\boldsymbol{e} \in \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. □

Remarks To us, the best way for $\mathcal{M}_1$ to decide whether an incoming query is made by our small field attacker or not is to check the algebraic structure of its $\boldsymbol{x}$-entry, *i.e.*, whether $\boldsymbol{x}$ can be written in the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$ such that $\mathsf{Dim}\,(\boldsymbol{x}_0)$ is "small" and $\boldsymbol{e}$ is "short". Such check is similar to the bounded distance decoding problem, and it seems to us that the best way to do such check is close to the brute-force. In practice, the instantiation of $\mathcal{M}_1$ should decide whether an incoming query is malicious or not in a *timely* manner; Moreover, such restriction imposed by $\mathcal{M}_1$ should be *reasonable* in the sense that the *false negative rate* (*i.e.*, the probability that an "innocent" query made by an "honest" caller is considered a malicious one made by our small field attackers) cannot be too high. To identify those queries made by $(\mathcal{V}/\mathcal{A}'_1)_\delta$, the best way is to identify those queries made by $(\mathcal{V}/\mathcal{A}'_1)_\delta$ in its Phase 1. Hence, when δ is moderately large, given the randomness of I, it takes too much time in practice to do so, and the false negative rate is intuitively high. In sum, it is *too costly to be practical* for $\mathcal{M}_1$ to protect its secret $\boldsymbol{s}$ from our efficient attacker $(\mathcal{V}/\mathcal{A}'_1)_\delta$, even if the foregoing restrictions are imposed by $\mathcal{M}_1$ on incoming queries.

6 The Small Field Attack Against Π_1

In this section, we switch our analysis back to the oracle $\mathcal{M}_0$ (defined in Section 4.2), and show how to construct an efficient attacker $\mathcal{A}_0$ which can recover the static private key of $\mathcal{M}_0$ (defined in Section 4.2), after a set of queries made to $\mathcal{M}_0$; Moreover, both the static public key of $\mathcal{A}_0$ and those queries it makes to $\mathcal{M}_0$ are as "random-looking" as possible. The existence of $\mathcal{A}_0$ implies Π_1 is vulnerable to our small field attack.

Construction of $\mathcal{A}_0$ As $\boldsymbol{p} = \boldsymbol{a}\boldsymbol{s} + 2\boldsymbol{e} \in \mathcal{R}_q$ is made public, to recover the private key $(\boldsymbol{s}, \boldsymbol{e})$ of $\mathcal{M}_0$, it suffices for $\mathcal{A}_0$ to recover $\boldsymbol{s} \in \mathcal{R}_q$. Moreover, it suffices for $\mathcal{A}_0$ to recover every CRT-coefficient $s_i \triangleq \eta_i(\boldsymbol{s})$ of $\boldsymbol{s} \in \mathcal{R}_q$. Recall that the "undetectable" and efficient attacker $(\mathcal{V}/\mathcal{A}'_1)_\delta$ against $\mathcal{M}_1$ constructed in Section 5.3 implies we can already construct an efficient attacker against $\mathcal{M}_0$ *with static public key* $\mathbf{0} \in \mathcal{R}_q$, whose queries made to $\mathcal{M}_0$ are as "random-looking" as possible. Thus, the difficulty of designing $\mathcal{A}_0$ lies in how to design its static public key $\boldsymbol{p}^*$ that looks "random". It turns out the construction of $\mathcal{V}$ already indicates that how to choose a "random-looking" $\boldsymbol{p}^*$ for $\mathcal{A}_0$.

The *general structure* of $\mathcal{A}_0$ is defined as follows. $\mathcal{A}_0$ consists of three *consecutive* phases: *Phase 0*, *Phase 1*, and *Phase 2*. In Phase 0, $\mathcal{A}_0$ generates static its public/private key pair as follows: first, it chooses a positive integer δ that is moderately large, say δ; Then, it chooses a proper index set $I \subseteq [n]$ of size δ *uniformly at random*; After that, it samples $\boldsymbol{p}_0^* \leftarrow \left\{\boldsymbol{u} \in \mathcal{R}_q \,\middle|\, \mathsf{Dim}\,(\boldsymbol{u}) = I\right\}$ and $\boldsymbol{e}_0^* \leftarrow \mathbb{Z}_{1+2\alpha\sqrt{n}}$ uniformly at random; The static public key of $\mathcal{A}_0$ is $\boldsymbol{p}^* \triangleq \boldsymbol{p}_0^* + 2\boldsymbol{e}_0^*$, and the associated static private key of $\mathcal{A}_0$ is $(\boldsymbol{s}^*, \boldsymbol{e}^*)$, where $\boldsymbol{s}^* \leftarrow \mathcal{R}_q$ is drawn randomly and $\boldsymbol{e}^* \triangleq 2^{-1}(\boldsymbol{p}^* - \boldsymbol{a}\boldsymbol{s}^*)$. It should be stressed that the static public/private key pair of $\mathcal{A}_0$ is *not* honestly generated.

The Phase 1 and Phase 2 of $\mathcal{A}_0$ are devoted to the recovery of every $s_i = \eta_i(\boldsymbol{s}), i \in [n]$, and their functionalities are similar to those of $(\mathcal{V}/\mathcal{A}_1')_\delta$, respectively: Phase 1 is devoted to recovering δ CRT-coefficients of $\boldsymbol{s}$, *i.e.*, $\{s_i \mid i \in I\}$ where $I \subseteq [n]$ is of size δ, and Phase 2 is to recover the others ones.

The Phase 1 of $\mathcal{A}_0$ is very close to the design of $(\mathcal{V}/\mathcal{A}_1')_\delta$ with slight differences; Nevertheless, they share the same time complexity, which is measured in terms of the number of queries made. For simplicity, only the Phase 2 of $\mathcal{A}_0$ is fully described in the sequel. For the moment, we assume that $I \subseteq [i-1]$ and the CRT-coefficients $s_1, \cdots, s_{i-1} \in \mathbb{F}_q$ have already been recovered successfully, and we are about to see how $\mathcal{A}_0$ recovers a *new* CRT-coefficient of $\boldsymbol{s}$, say $s_i \in \mathbb{F}_q$, via a set of queries to $\mathcal{M}_0$. The general strategy is simple: first pick $\tilde{s_i} \leftarrow \mathbb{F}_q$ randomly, and guess $s_i = \tilde{s_i}$; Then conduct a set $\mathcal{Q}_i(\tilde{s_i})$ of queries to $\mathcal{M}_0$ such that *except with negligible probability*, $s_i = \tilde{s_i}$ if and only if $\mathcal{M}_0$ returns 1 on every query in $\mathcal{Q}_i(\tilde{s_i})$; When $\tilde{s_i}$ runs over the set $\mathbb{F}_q$, the exact value of s_i would be recovered *w.o.p.*

Jumping ahead, every query in $\mathcal{Q}_i(\tilde{s_i})$ is of the form $(\texttt{id}^*, \boldsymbol{p}^*,\ \boldsymbol{x}_k = k\boldsymbol{c}_i + \boldsymbol{h}_k + 2\boldsymbol{e}_k, \boldsymbol{w}_k, \boldsymbol{z}_k)$, where $k \in [q_0]$, $\boldsymbol{h}_k \leftarrow \{\boldsymbol{u} \in \mathcal{R}_q \mid \mathsf{Dim}\,(\boldsymbol{u}) = [i-1]\}$, $\boldsymbol{e}_k \leftarrow \mathbb{Z}_{1+2\alpha'\sqrt{n}}$, and $\boldsymbol{w}_k, \boldsymbol{z}_k \in \mathbb{B}^n$ are to be determined later.

Recall that, on the query $(\texttt{id}, \boldsymbol{p},\ \boldsymbol{x}_k = k\boldsymbol{c}_i + \boldsymbol{h}_k + 2\boldsymbol{e}_k, \boldsymbol{w}_k, \boldsymbol{z}_k)$ where $\boldsymbol{h}_k = \sum_{r\in[i-1]} h_{k,r}\boldsymbol{c}_r, h_{k,r} \leftarrow \mathbb{F}_q^\times$, the oracle $\mathcal{M}_0$ first samples $\boldsymbol{g} \leftarrow D_{\mathbb{Z}^n,\alpha}$, then compute $\boldsymbol{c} \triangleq H_1(\texttt{id}, \texttt{id}^*, \boldsymbol{x}_k)$ and

$$\begin{aligned}
\boldsymbol{v}_k &\triangleq (\boldsymbol{p}^*\boldsymbol{c} + \boldsymbol{x}_k)\boldsymbol{s} + q_0 \cdot \boldsymbol{w}_k + 2\boldsymbol{c}\cdot\boldsymbol{g} \\
&= k \cdot \boldsymbol{s} \cdot \boldsymbol{c}_i + \boldsymbol{p}_0^*\boldsymbol{c}\boldsymbol{s} + \boldsymbol{s}\cdot\boldsymbol{h}_k + q_0\cdot\boldsymbol{w}_k + 2\cdot(\boldsymbol{e}_k\boldsymbol{s} + \boldsymbol{e}_0^*\boldsymbol{c}\boldsymbol{s} + \boldsymbol{c}\boldsymbol{g}) \\
&= k\varDelta s_i\boldsymbol{c}_i + (k\tilde{s}_i\boldsymbol{c}_i + \boldsymbol{p}_0^*\boldsymbol{c}\boldsymbol{s} + \boldsymbol{s}\cdot\boldsymbol{h}_k + q_0\cdot\boldsymbol{w}_k) + 2\varepsilon_k \quad (\varepsilon_k \triangleq \boldsymbol{e}_k\boldsymbol{s} + \boldsymbol{e}_0^*\boldsymbol{c}\boldsymbol{s} + \boldsymbol{c}\boldsymbol{g} \sim [\varepsilon_{k,j}]_{j\in[n]}) \\
&= \begin{bmatrix} \varDelta s_i \cdot kc_{i,1} + \Big(k\tilde{s}_i c_{i,1} + \sum_{r\in I}\eta_r(\boldsymbol{p}_0^*\boldsymbol{c}\boldsymbol{s})c_{r,1} + \sum_{r\in[i-1]} s_r h_{k,r} c_{r,1} + q_0 w_{k,1}\Big) \\ \vdots \\ \varDelta s_i \cdot kc_{i,n} + \Big(k\tilde{s}_i c_{i,n} + \sum_{r\in I}\eta_r(\boldsymbol{p}_0^*\boldsymbol{c}\boldsymbol{s})c_{r,n} + \sum_{r\in[i-1]} s_r h_{k,r} c_{r,n} + q_0 w_{k,n}\Big) \end{bmatrix} + 2\cdot\begin{bmatrix}\varepsilon_{k,1} \\ \vdots \\ \varepsilon_{k,n}\end{bmatrix},
\end{aligned}$$

where $\varDelta s_i \triangleq s_i - \tilde{s}_i$; Finally, $\mathcal{M}_0$ computes $\sigma_k := \mathsf{Parity}\,(\boldsymbol{v}_k)$, and returns 1 if and only if $\sigma_k = \boldsymbol{z}_k$. Notice that $\mathsf{Dim}\,(\boldsymbol{p}_0^*\boldsymbol{c}\boldsymbol{s}) \subseteq \mathsf{Dim}\,(\boldsymbol{p}_0^*) = I$, making it possible for $\mathcal{A}_0$ to pre-compute every $\eta_r(\boldsymbol{p}_0^*\boldsymbol{c}\boldsymbol{s}), r \in I$, before issuing the query.

It is not hard to verify the correctness of the following lemma.

Lemma 15. *Let g denote a primitive element of $\mathbb{F}_q^\times$, and define $d \triangleq \frac{q-1}{m}$, $S_g \triangleq \{g^r \mid r \in [d]\}$. With the notations defined previously, if $I \subseteq [i-1]$ and $s_1, \cdots, s_{i-1}$ have been given and $q > 1 + 8(n\alpha\alpha' + n\alpha\gamma + n^2\alpha^2\gamma)$, then* except with negligible probability, *$\varDelta s_i = 0$ if and only if $\mathcal{M}_0$ returns 1 on every query in*

$$\mathcal{Q}_i(\tilde{s_i}) = \left\{ \big(\texttt{id}^*, \boldsymbol{p}^*, k\boldsymbol{c}_i + \boldsymbol{h}_k + 2\boldsymbol{e}_k, [w_{k,j}]_{j\in[n]}, [z_{k,j}]_{j\in[n]}\big) \;\middle|\; \begin{array}{c} k \in S_g,\ j \in [n],\ h_{k,1}, \cdots, h_{k,i-1} \leftarrow \mathbb{F}_q^\times, \\ \boldsymbol{h}_k = \sum_{r\in[i-1]} h_{k,r}\boldsymbol{c}_r,\ \boldsymbol{e}_k \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}, \\ u_{k,j} = k\tilde{s}_i c_{i,j} + \sum_{r\in I}\eta_r(\boldsymbol{p}_0^*\boldsymbol{c}\boldsymbol{s})c_{r,j} + \sum_{r\in[i-1]} s_r h_{k,r} c_{r,j}, \\ w_{k,j} = \mathsf{Cha}\,(u_{k,j})\,, z_{k,j} = \mathsf{Mod}\,(u_{k,j}, w_{k,j}) \end{array} \right\}.$$

In particular, for every query in $\mathcal{Q}_i(\tilde{s_i})$, its $\boldsymbol{x}$-entry is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$, where $\mathsf{Dim}\,(\boldsymbol{x}_0) = [i]$ and $\boldsymbol{e} \in \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$. □

The correctness and the efficiency of $\mathcal{A}_0$ can be easily verified. In sum, the efficient attacker $\mathcal{A}_0$ implies the existence of the desired efficient attacker against the honest party j in Π_1, as the following theorem indicates.

Theorem 16. *With the notations defined previously, when $q > 1 + 8(n\alpha\alpha' + n\alpha\gamma + n^2\alpha^2\gamma)$, there exists an efficient adversary $\mathcal{A}$ that can recover the static private key of the honest party j in Π_1* w.o.p. *In particular, for every query made by $\mathcal{A}$, its $\boldsymbol{x}$-entry is always of the form $\boldsymbol{x}_0 + 2\boldsymbol{e}$, where $|\mathsf{Dim}\,(\boldsymbol{x}_0)| \geq \delta$ and $\boldsymbol{e} \in \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$.* □

In [ZZDS14,ZZD+15], four groups of suggested parameters for Π_1 are proposed; Also, $\gamma := \alpha$ is suggested. We should stress that all of these four groups of parameters satisfy the parameter requirement in Theorem 16.

Remarks By the ring-LWE assumption, the distribution of the static public key of an *honest* player is computationally indistinguishable from the uniform distribution over $\mathcal{R}_q$. Therefore, from the viewpoint of the honest party j, every element in $\mathcal{R}_q$ is *equally* likely to be the static public key of an honest initiator.

Similar to $(\mathcal{V}/\mathcal{A}_1')_\delta$, the efficient adversary $\mathcal{A}$ against Π_1 can make its session queries to the honest party j as random-looking as possible by choosing an appropriate δ, making it *hard in practice* for party j to identify (and hence reject) those session queries made by $\mathcal{A}$.

7 Analysis on the Two-pass AKE Protocol Π_2

In Π_2, the static private key of party i is $(\boldsymbol{s}_i \leftarrow D_{\mathbb{Z}^n,\alpha}, \boldsymbol{e}_i \leftarrow D_{\mathbb{Z}^n,\alpha})$, and the associate static public key is $\boldsymbol{p}_i \triangleq \boldsymbol{a}\boldsymbol{s}_i + 2\boldsymbol{e}_i \in \mathcal{R}_q$, where $\boldsymbol{a} \in \mathcal{R}_q$ denotes a global parameter of Π_2 that is drawn from the uniform distribution over $\mathcal{R}_q$. Similar notations, $(\boldsymbol{s}_j, \boldsymbol{e}_j), \boldsymbol{p}_j$, carry over to party j in Π_2.

Here is a brief review of the two-pass AKE scheme Π_2 proposed in [ZZDS14,ZZD$^+$15].

Initiation First, party i acts as follows:

1. Pick $\boldsymbol{r}_i, \boldsymbol{f}_i \leftarrow D_{\mathbb{Z}^n,\beta}$, and compute $\boldsymbol{x}_i \triangleq \boldsymbol{a}\boldsymbol{r}_i + 2\boldsymbol{f}_i$.
2. Compute $\boldsymbol{c} \triangleq H_1(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i)$, $\hat{\boldsymbol{r}}_i \triangleq \boldsymbol{s}_i\boldsymbol{c} + \boldsymbol{r}_i$ and $\hat{\boldsymbol{f}}_i \triangleq \boldsymbol{e}_i\boldsymbol{c} + \boldsymbol{f}_i$.
3. Let $\boldsymbol{z} \in \mathbb{Z}^{2n}$ be the coefficient vector of $\hat{\boldsymbol{r}}_i$ concatenated with that of $\hat{\boldsymbol{f}}_i$, and $\boldsymbol{z}_1$ be the coefficient vector of $\boldsymbol{s}_i\boldsymbol{c}$ concatenated with that of $\boldsymbol{e}_i\boldsymbol{c}$. Repeat steps 1-3 with probability $1 - \min\left(1, \frac{D_{\mathbb{Z}^{2n},\beta}(\boldsymbol{z})}{M \cdot D_{\mathbb{Z}^{2n},\beta,\boldsymbol{z}_1}(\boldsymbol{z})}\right)$.
4. Send $\boldsymbol{x}_i$ to party j.

Response On message $\boldsymbol{x}_i$, party j works as follows:

1. Pick $\boldsymbol{r}_j, \boldsymbol{f}_j \leftarrow D_{\mathbb{Z}^n,\beta}$, and compute $\boldsymbol{y}_j \triangleq \boldsymbol{a}\boldsymbol{r}_j + 2\boldsymbol{f}_j$.
2. Compute $\boldsymbol{d} \triangleq H_1(\mathtt{id}_j, \mathtt{id}_i, \boldsymbol{y}_j, \boldsymbol{x}_i)$, $\hat{\boldsymbol{r}_j} \triangleq \boldsymbol{s}_j\boldsymbol{d} + \boldsymbol{r}_j$ and $\hat{\boldsymbol{f}}_j \triangleq \boldsymbol{e}_j\boldsymbol{d} + \boldsymbol{f}_j$.
3. Let $\boldsymbol{z} \in \mathbb{Z}^{2n}$ be the coefficient vector of $\hat{\boldsymbol{r}_j}$ concatenated with that of $\hat{\boldsymbol{f}}_j$, and $\boldsymbol{z}_1$ be the coefficient vector of $\boldsymbol{s}_j\boldsymbol{d}$ concatenated with that of $\boldsymbol{e}_j\boldsymbol{d}$. Repeat steps 1-3 with probability $1 - \min\left(1, \frac{D_{\mathbb{Z}^{2n},\beta}(\boldsymbol{z})}{M \cdot D_{\mathbb{Z}^{2n},\beta,\boldsymbol{z}_1}(\boldsymbol{z})}\right)$.
4. Pick $\boldsymbol{g}_j \leftarrow D_{\mathbb{Z}^n,\beta}$, and compute $\boldsymbol{k}_j \triangleq (\boldsymbol{p}_i\boldsymbol{c} + \boldsymbol{x}_i) \cdot \hat{\boldsymbol{r}_j} + 2\boldsymbol{c}\boldsymbol{g}_j$.
5. Let $\boldsymbol{w}_j \triangleq \mathsf{Cha}\left(\boldsymbol{k}_j\right) \in \mathbb{B}^n$, and send $(\boldsymbol{y}_j, \boldsymbol{w}_j)$ to party i.
6. Compute $\sigma_j \triangleq \mathsf{Mod}\left(\boldsymbol{k}_j, \boldsymbol{w}_j\right)$, and derive the session key $\mathtt{sk}_j \triangleq H_2(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{y}_j, \boldsymbol{w}_j, \sigma_j)$.

Finish On message $(\boldsymbol{y}_j, \boldsymbol{w}_j)$, party i proceeds as follows:

1. Pick $\boldsymbol{g}_i \leftarrow D_{\mathbb{Z}^n,\beta}$, and compute $\boldsymbol{k}_i \triangleq (\boldsymbol{p}_j\boldsymbol{d} + \boldsymbol{y}_j) \cdot \hat{\boldsymbol{r}}_i + 2\boldsymbol{d}\boldsymbol{g}_i$.
2. Compute $\sigma_i \triangleq \mathsf{Mod}\left(\boldsymbol{k}_i, \boldsymbol{w}_j\right)$, and derive the session key $\mathtt{sk}_i \triangleq H_2(\mathtt{id}_i, \mathtt{id}_j, \boldsymbol{x}_i, \boldsymbol{y}_j, \boldsymbol{w}_j, \sigma_i)$.

7.1 Claim 16, and the Underlying Games $G_{2,4}, G_{2,5}$

In the security proof of Π_2 [ZZDS14], the set of PPT adversaries is partitioned into *five* types, according to the internal structures of the test session. We are interested in the Type-II adversaries w.r.t. the test session denoted $(\Pi_2, I, \mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, (\boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}))$, in which the adversary $\mathcal{A}$ impersonates the honest user j^* but $\boldsymbol{y}_{j^*}$ is *not* sent by j^* upon receiving $\boldsymbol{x}_{i^*}$ from i^*; In other words, the test session has no matching session in this case. Claim 16, together with other claims, is devoted to establishing the provable security regarding Type-II adversary, and two games are involved in Claim 16: $G_{2,4}$ and $G_{2,5}$. Roughly speaking, the difference between $G_{2,4}$ and $G_{2,5}$ lies in the simulation of the test-session, as summarized below. Please refer to [ZZDS14] for the full detail.

Denote by $\mathsf{sid}^* \triangleq (\Pi_2, I, \mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, (\boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}))$ the test session, where $\boldsymbol{x}_{i^*}$ is output by honest party i^* with intended honest party j^*. In $G_{2,4}$, the simulator $\mathcal{S}$ maintains two tables L_1, L_2 for the random oracles $H_1(\cdot), H_2(\cdot)$, respectively. Now, $\mathcal{S}$ chooses $\boldsymbol{u}_0, \boldsymbol{u}_1 \leftarrow \mathcal{R}_q$, and sets the global parameter $\boldsymbol{a}$ to be $\boldsymbol{u}_0$, and sets the static public key of party j^* to be $\boldsymbol{u}_1$, *i.e.*, $\boldsymbol{a} := \boldsymbol{u}_0, \boldsymbol{p}_{j^*} := \boldsymbol{u}_1$. Moreover, $\mathcal{S}$ prepares the table

$$T \triangleq \left\{ \left(\hat{\boldsymbol{r}}_k, \hat{\boldsymbol{f}}_k, \boldsymbol{g}_k, \boldsymbol{v}_{0,k}, \boldsymbol{v}_{1,k}\right) \;\middle|\; \begin{array}{c} 1 \le k \le \ell, \hat{\boldsymbol{r}}_k, \hat{\boldsymbol{f}}_k, \boldsymbol{g}_k \leftarrow D_{\mathbb{Z}^n,\beta}, \\ \boldsymbol{v}_{0,k} = \boldsymbol{u}_0\hat{\boldsymbol{r}}_k + 2\hat{\boldsymbol{f}}_k, \\ \boldsymbol{v}_{1,k} = \boldsymbol{u}_1\hat{\boldsymbol{r}}_k + 2\boldsymbol{g}_k \end{array} \right\}.$$

Finally, for the test session, $\mathcal{S}$ answers oracle queries made by the efficient adversary $\mathcal{A}$ as follows:

- Upon receiving $\mathsf{Send}_0(\Pi_2, I, \mathtt{id}_{i^*}, \mathtt{id}_{j^*})$ w.r.t. the test session, $\mathcal{S}$ proceeds as follow:
  1. Sample an invertible element $\boldsymbol{c}^* \leftarrow D_{\mathbb{Z}^n,\gamma}$, and choose the *first unused* tuple in T, say the ℓ^*-th tuple, and set $\hat{\boldsymbol{x}}_{i^*} := \boldsymbol{v}_{0,\ell^*}$.
  2. Define $\boldsymbol{x}_{i^*} := \hat{\boldsymbol{x}}_{i^*} - \boldsymbol{p}_{i^*}\boldsymbol{c}^*$.
  3. Repeat steps 1-2 with probability $1 - 1/M$, where M is a sufficiently large positive integer.

4. Abort if there is a tuple $((\mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}), *) \in L_1$. Else, add $((\mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}), \boldsymbol{c}^*)$ into L_1, and return $\boldsymbol{x}_{i^*}$ to the adversary $\mathcal{A}$.

- Upon receiving $\mathsf{Send}_2(\Pi_2, I, \mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, (\boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}))$ w.r.t. the test-session, $\mathcal{S}$ proceeds as follow:
  5. Set $\boldsymbol{d}^* \leftarrow H_1(\mathtt{id}_{j^*}, \mathtt{id}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{x}_{i^*})$, and compute
$$\boldsymbol{k}_{i^*} := \boldsymbol{d}^* \boldsymbol{v}_{1,\ell^*} + \boldsymbol{y}_{j^*} \hat{\boldsymbol{r}}_{\ell^*} = (\boldsymbol{p}_{j^*} \boldsymbol{d}^* + \boldsymbol{y}_{j^*}) \hat{\boldsymbol{r}}_{\ell^*} + 2\boldsymbol{d}^* \boldsymbol{g}_{\ell^*}.$$
  6. Compute $\sigma_{i^*} := \mathsf{Mod}\,(\boldsymbol{k}_{i^*}, \boldsymbol{w}_{j^*})$, and derive the session key $\mathtt{sk}_{i^*} \leftarrow H_2(\mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}, \sigma_{i^*})$.
- Upon the query of $\mathsf{Test}(\Pi_2, I, \mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, (\boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}))$, $\mathcal{S}$ chooses $b \leftarrow \mathbb{B}$, and $\mathtt{sk}'_{i^*} \leftarrow \mathbb{B}^\lambda$; If $b = 0$, $\mathtt{sk}'_{i^*}$ is returned, or else $\mathtt{sk}_{i^*}$ is returned.

Game $G_{2,5}$ is pretty similar to $G_{2,4}$, and hence only the differences are listed below. In $G_{2,5}$, $\mathcal{S}$ generates the table
$$T' \triangleq \left\{ (\boldsymbol{v}_{0,k}, \boldsymbol{v}_{1,k}) \,\middle|\, 1 \le k \le \ell, \boldsymbol{v}_{0,k}, \boldsymbol{v}_{1,k} \leftarrow \mathcal{R}_q. \right\}$$
Moreover, the only difference between $G_{2,4}$ and $G_{2,5}$ is that, in $G_{2,5}$:

- Upon the query of $\mathsf{Send}_2(\Pi_2, I, \mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, (\boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}))$, $\mathcal{S}$ proceeds as follow:
  1. Randomly choose $\boldsymbol{k}_{i^*} \leftarrow \mathcal{R}_q$;
  2. Compute $\sigma_{i^*} := \mathsf{Mod}\,(\boldsymbol{k}_{i^*}, \boldsymbol{w}_{j^*})$, and derive the session key $\mathtt{sk}_{i^*} \leftarrow H_2(\mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}, \sigma_{i^*})$.

Note that, in $G_{2,5}$, when $\mathcal{S}$ answers the Send_0 oracle query for the test session, the value $\hat{\boldsymbol{x}}_{i^*} := \boldsymbol{v}_{0,\ell^*}$ follows the uniform distribution over $\mathcal{R}_q$ according to the table T' set by $\mathcal{S}$.

7.2 Analysis on the Proof of Claim 16

The proof of Claim 16 (in accordance with our communications with the corresponding author) considers any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, which could be divided into two *consecutive* stages $\mathcal{A}_1$ and $\mathcal{A}_2$. $\mathcal{A}_1$ denotes the actions of $\mathcal{A}$ until it just gets the RO-answer $\boldsymbol{d}^* \leftarrow H_1(\mathtt{id}_{j^*}, \mathtt{id}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{x}_{i^*})$. Let $\mathsf{view}_1 \triangleq (\boldsymbol{x}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{d}^*, \boldsymbol{p}_{j^*}, \boldsymbol{p}_{i^*}, \boldsymbol{c}^*, \boldsymbol{a}, \boldsymbol{tr}, \boldsymbol{st})$ be the output of $\mathcal{A}_1$ (to $\mathcal{A}_2$), where $\boldsymbol{tr}$ denotes the view of $\mathcal{A}_1$ in other sessions other than the test session, and $\boldsymbol{st}$ denotes some state information. On input of view_1, $\mathcal{A}_2$ performs the remaining actions of $\mathcal{A}$.

Thanks to deep discussions with the author in charge of the proof of Claim 16 [ZZDS14], we could figure out more detailed explanations for the proof and reach some consensus as follows.

- Under the ring-LWE assumption, the output (*i.e.*, view_1) of $\mathcal{A}_1$ in $G_{2,4}$ is computationally indistinguishable from that of $\mathcal{A}_1$ in $G_{2,5}$.
- In the RO model, in order for $\mathcal{A}_2$ to succeed, it has to make the RO-query $H_2(\mathtt{id}_{i^*}, \mathtt{id}_{j^*}, \boldsymbol{x}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{w}_{j^*}, \sigma_{i^*})$ with non-negligible probability. This means that *except with negligible probability*, a successful $\mathcal{A}_2$ has to compute $\sigma_{i^*} := \mathsf{Mod}\,(\boldsymbol{k}_{i^*}, \boldsymbol{w}_{j^*})$.
- The actions of $\mathcal{A}_2$ essentially makes no essential contribution to its ability of computing $\sigma_{i^*} := \mathsf{Mod}\,(\boldsymbol{k}_{i^*}, \boldsymbol{w}_{j^*})$, even if $\mathcal{A}_2$ could maliciously set $\boldsymbol{w}_{j^*}$ in its stage. In other words, for a successful $\mathcal{A}_2$, its ability of computing σ_{i^*} mainly stems from the output view_1 of $\mathcal{A}_1$.
- The probability that $\mathcal{A}_2$ could compute σ_{i^*} in $G_{2,5}$ is negligible, as in the RO model $\boldsymbol{k}_{i^*}$ is a random value independent of the view of $\mathcal{A}$ in $G_{2,5}$.
- In the proof of Claim 16 in [ZZDS14], it also uses the forking lemma to argue some extra properties. According to our discussions with the corresponding author of Claim 16, the use of forking lemma is actually unnecessary for proving Claim 16, and thus the reasoning related to forking lemma can be removed.

Now comes the divarication. The corresponding author of Claim 16 suggests that the above facts have already been sufficient to reach the final conclusion: the probability that $\mathcal{A}_2$ outputs $\sigma_{i^*} := \mathsf{Mod}\,(\boldsymbol{k}_{i^*}, \boldsymbol{w}_{j^*})$ in $G_{2,4}$ is also negligible.

From our view, to reach this final conclusion, we need to additionally argue that the joint distribution of $(\mathsf{view}_1, \boldsymbol{k}_{i^*})$ in $G_{2,4}$ and that in $G_{2,5}$ are computationally indistinguishable, because the event in question is defined over both view_1 and $\boldsymbol{k}_{i^*}$. In particular, we need to at least argue the computational indistinguishability between $\boldsymbol{k}_{i^*}$ defined over $G_{2,4}$ and the uniform distribution over $\mathcal{R}_q$(that is just the distribution of $\boldsymbol{k}_{i^*}$ in $G_{2,5}$), which is, however, explicitly claimed to be unnecessary by the author of Claim 16.

Key differences between $G_{2,4}$ and $G_{2,5}$, and subtleties buried To make the analysis clearer, we would like to highlight some key differences between $G_{2,4}$ and $G_{2,5}$ explicitly.

Firstly, in game $G_{2,5}$, the random variables $\boldsymbol{k}_{i^*}$ and $\boldsymbol{x}_{i^*}$ are *independent*. However this is not the case in game $G_{2,4}$, since $\boldsymbol{x}_{i^*} = \boldsymbol{a}\hat{\boldsymbol{r}}_{\ell^*} + 2\hat{\boldsymbol{f}}_{\ell^*} - \boldsymbol{p}_{i^*}\boldsymbol{c}^*$ and $\boldsymbol{k}_{i^*} = \boldsymbol{d}^*(\boldsymbol{p}_{j^*}\hat{\boldsymbol{r}}_{\ell^*} + 2\boldsymbol{g}_{\ell^*}) + \boldsymbol{y}_{j^*}\hat{\boldsymbol{r}}_{\ell^*}$ are related by $\hat{\boldsymbol{r}}_{\ell^*}$, making them *dependent*. In other words, in game $G_{2,4}$, when $\boldsymbol{x}_{i^*}$ is given to the efficient adversary $\mathcal{A}$, it might be possible for $\mathcal{A}$ to extract some information regarding $\boldsymbol{k}_{i^*}$.

Secondly, given that in $G_{2,4}$ we have $\boldsymbol{x}_{i^*} = \boldsymbol{a}\hat{\boldsymbol{r}}_{\ell^*} + 2\hat{\boldsymbol{f}}_{\ell^*} - \boldsymbol{p}_{i^*}\boldsymbol{c}^*$ and $\boldsymbol{k}_{i^*} := \boldsymbol{d}^*\boldsymbol{v}_{1,\ell^*} + \boldsymbol{y}_{j^*}\hat{\boldsymbol{r}}_{\ell^*} = (\boldsymbol{p}_{j^*}\boldsymbol{d}^* + \boldsymbol{y}_{j^*})\hat{\boldsymbol{r}}_{\ell^*} + 2\boldsymbol{d}^*\boldsymbol{g}_{\ell^*}$, the values $(\boldsymbol{x}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{d}^*, \boldsymbol{p}_{j^*}, \boldsymbol{p}_{i^*}, \boldsymbol{c}^*, \boldsymbol{a})$ essentially determine the value $\boldsymbol{k}_{i^*} - 2\boldsymbol{d}^*\boldsymbol{g}_{\ell^*}$. In view that $2\boldsymbol{d}^*\boldsymbol{g}_{\ell^*}$ is small, the value $\boldsymbol{k}_{i^*}$ is essentially committed to $(\boldsymbol{x}_{i^*}, \boldsymbol{y}_{j^*}, \boldsymbol{d}^*, \boldsymbol{p}_{j^*}, \boldsymbol{p}_{i^*}, \boldsymbol{c}^*, \boldsymbol{a})$ *up to a "small" and even noise*.

Finally, notice that in $G_{2,4}$, for the equation $\boldsymbol{k}_{i^*} = \boldsymbol{d}^*\boldsymbol{v}_{1,\ell^*} + \boldsymbol{y}_{j^*}\hat{\boldsymbol{r}}_{\ell^*}$, the two terms on the right-hand side are not independent either. Thus, it is not that easy to analyze the *exact* distribution of $\boldsymbol{k}_{i^*}$.

These highlighted differences indicate that the above facts *by consensus* are insufficient to establish the indistinguishability between $\boldsymbol{k}_{i^*}$ defined over $G_{2,4}$ and the uniform distribution over $\mathcal{R}_q$. Actually, we even do not know how to *formally* prove, with a reducibility argument, a seemingly easier goal: given $\boldsymbol{x}_{i^*} = \boldsymbol{a}\hat{\boldsymbol{r}}_{\ell^*} + 2\hat{\boldsymbol{f}}_{\ell^*} - \boldsymbol{p}_{i^*}\boldsymbol{c}^*$, it is infeasible for any efficient A_2 to successfully recover $\boldsymbol{h}_{i^*} \triangleq (\boldsymbol{p}_{j^*}\boldsymbol{d}^* + \boldsymbol{y}_{j^*})\hat{\boldsymbol{r}}_{\ell^*}$ by maliciously setting $\boldsymbol{y}_{j^*}$ and $\boldsymbol{w}_{j^*}$ (recall that $\boldsymbol{k}_{i^*} = \boldsymbol{h}_{i^*} + 2\boldsymbol{d}^*\boldsymbol{g}_{\ell^*}$ where $2\boldsymbol{d}^*\boldsymbol{g}_{\ell^*}$ is even and "small"). Notice that, conditioned on A_2 recovers $\boldsymbol{h}_{i^*}$, by setting $\boldsymbol{w}_{j^*} := \mathsf{Parity}(\boldsymbol{h}_{i^*})$ it can then determine the value of σ_{i^*} and thus break the security, since the following holds *w.o.p.*:

$$\sigma_{i^*} = \mathsf{Mod}(\boldsymbol{k}_{i^*}, \boldsymbol{w}_{j^*}) = \mathsf{Mod}(\boldsymbol{h}_{i^*} + 2\boldsymbol{d}^*\boldsymbol{g}_{\ell^*}, \mathsf{Parity}(\boldsymbol{h}_{i^*})) = \mathsf{Mod}(\boldsymbol{h}_{i^*}, \mathsf{Parity}(\boldsymbol{h}_{i^*}))\,.$$

The above clarifications also indicate a fundamental difference between the security proof of HMQV [Kra05] and that of its RLWE-based analogue [ZZD$^+$15]. In the security proof of HMQV [Kra05], the simulator can *directly* obtain, from RO-query, the key material (corresponding to $\boldsymbol{k}_{i^*}$ in [ZZD$^+$15]) in order to reach reduction contradiction to the underlying gap Diffie-Hellman assumption. However, in the security proof of RLWE-based HMQV-analogue [ZZD$^+$15], the simulator gets, also from RO-query, only the value $\sigma_{i^*} := \mathsf{Mod}(\boldsymbol{k}_{i^*}, \boldsymbol{w}_{j^*})$, where the key material $\boldsymbol{k}_{i^*}$ is hidden and $\boldsymbol{w}_{j^*}$ may be maliciously set only in the stage of $\mathcal{A}_2$. From our view, these buried subtleties need to be dealt with explicitly in a formal analysis.

Justification with simplified analogous games To further justify our observations, we consider a pair of simplified analogous games (G_0, G_1) w.r.t. a special PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. These simplified games are artificial and are actually unrelated to $G_{2,4}$ and $G_{2,5}$, which are introduced for easier logic explanation. In both games, $\mathcal{A}$ gets access to a random oracle $H : \{0,1\}^\lambda \to \{0,1\}^\lambda$ that is maintained and simulated by a PPT simulator S, where λ is the security parameter. Denote by Com a computationally hiding commitment scheme used also in these games.

In game G_0 (resp., G_1), S simulates the random oracle H with another random oracle $H' : \{0,1\}^{2\lambda} \to \{0,1\}^\lambda$ as follows. Before the game starts, it sets $pre = 0^\lambda$ (resp., $pre \leftarrow \{0,1\}^\lambda$, *i.e.*, pre is a value taken uniformly at random from $\{0,1\}^\lambda$ in G_1). Whenever the adversary makes an RO-query with $x \in \{0,1\}^\lambda$, it returns the value $y = H'(pre||x)$. Notice that, from the view of $\mathcal{A}$, the RO simulation of S is perfect. At the end of $\mathcal{A}_1$, S computes $C = Com(0^\lambda)$ (resp., $C = Com(pre')$, where $pre' \leftarrow \{0,1\}^\lambda$ is independent of pre), and gives C to A_1. Define $\mathsf{view}_1 = (C, \boldsymbol{tr}, \boldsymbol{st})$ the output of $\mathcal{A}_1$, on which $\mathcal{A}_2$ proceeds further. Finally, $\mathcal{A}_2$ just simply outputs 0^λ.

For the above simplified analogous games, we have: (1) The output of $\mathcal{A}_1$ in G_0 is computationally indistinguishable from that in G_1;[4] (2) The actions of $\mathcal{A}_2$ make no contribution to its ability of computing pre; (3) Clearly, in G_2, the probability that $\mathcal{A}_2$ correctly outputs pre with probability of just $2^{-\lambda}$ in the RO model, as pre is a random value actually independent of $\mathcal{A}$'s view in this case. However, we couldn't reach the conclusion that $\mathcal{A}_2$ will also output pre with negligible probability in G_0. Actually, the success probability of $\mathcal{A}_2$ in G_0 is 1. The reason is just that the success event of $\mathcal{A}_2$ is defined not only over its view but also over the "hidden" value pre, which is 0^λ in G_0 clearly distinguishable from a random value in G_2.

Acknowledgement We thank some authors of [ZZD$^+$15,ZZDS14] for helpful discussions. We are indebted to Daniel J. Bernstein for many insightful suggestions, which have significantly improved this work. We also would like to thank the anonymous PQCrypto'17 reviewers for their valuable comments.

References

ADPS16. E. Alkim, Léo Ducas, T. Pöppelmann and P. Schwabe. Post-quantum key exchange - a new hope. 25th USENIX Security Symposium, USENIX Security 16, pages 327-343. August 10-12, 2016.

[4] Note that the random oracle $H : \{0,1\}^\lambda \to \{0,1\}^\lambda$ can be perfectly simulated without using H' or the knowledge of pre. Also, if C is not given to $\mathcal{A}_1$, $\mathcal{A}$'s view in G_0 and that in G_1 are identical.

BR93. M. Bellare and P. Rogaway. Entity authentication and key distribution. CRYPTO 1993, LNCS Vol. 773, pages 110-125. Springer, 1993.

BCNS15. Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Postquantum key exchange for the tls protocol from the ring learning with errors problem. IEEE S&P 2015: 553-570.

CK01. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. EUROCRYPT 2001, LNCS Vol. 2045, pages 453-474. Springer, 2001.

DARF16. J. Ding, S. Alsayigh, S. RV, and S. Fluhrer. Leakage of Signal function with reused keys in RLWE key exchange. *IACR Cryptology ePrint Archive*, 2016/1176, 2016.

DD12. L. Ducas and A. Durmus. Ring-LWE in polynomial rings. PKC 2012, LNCS Vol. 7293, pages 34-51. Springer, 2012.

DDLL13. L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal gaussians. CRYPTO 2013, LNCS Vol. 8042, pages 40-56. Springer, 2013. CRYPTO 2013, LNCS Vol. 8042. Springer, 2013.

DH76. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644-654, 1976.

DXL12. J. Ding, X. Xie, and X. Lin. A simple provably secure key exchange scheme based on the learning with errors problem. *IACR Cryptology ePrint Archive*, 2012/688, 2012.

Flu16. Fluhrer, S. Cryptanalysis of ring-LWE based key exchange with key share reuse. *IACR Cryptology ePrint Archive*, 2016/085, 2016.

Gen09. C. Gentry. Fully homomorphic encryption using ideal lattices. STOC 2009, pages 169 – 178. ACM, 2009.

GGH13a. S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. EUROCRYPT 2013. LNCS Vol. 7881, pages 1-17. Springer, 2013.

GGH$^+$13b. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. FOCS 2013, pages 40 – 49.

HK11. S. Halevi and H. Krawczyk. One-pass HMQV and asymmetric key-wrapping. PKC 2011. LNCS Vol. 6571, pages 317-334. Springer, 2011.

Kra05. H. Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. CRYPTO 2005, LNCS Vol. 3621, pages 546-566. Springer, 2005.

LMQ$^+$03. L. Law, A. Menezes, M. Qu, J. A. Solinas, and S. A. Vanstone. An efficient protocol for authenticated key agreement. *Des. Codes Cryptography*, 28, 119-134, 2003. *Des. Codes Cryptography*, 2003.

LPR13a. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43, 2013.

LPR13b. V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-LWE cryptography. EUROCRYPT 2013. LNCS Vol. 7881, pages 35-54. 2013.

LS15. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. Designs, Codes and Cryptography, 75(3):565 – 599, 2015.

Lyu12. V. Lyubashevsky. Lattice signatures without trapdoors. EUROCRYPT 2012, LNCS Vol. 7237, pages 738-755. Springer, 2012.

Pei14. C. Peikert. Lattice cryptography for the internet. PQCrypto 2014, LNCS Vol. 8772, pages 197 – 219. Springer, 2014.

Pei16. C. Peikert. A Decade of Lattice Cryptography. Foundations and Trends in Theoretical Computer Science 10(4), pages 283-424, 2016.

Reg09. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

SS11. D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. EUROCRYPT 2011, LNCS Vol. 6632, pages 27 – 47. Springer, 2011.

ZZD$^+$15. J. Zhang, Z. Zhang, J. Ding, M. Snook, and Ö. Dagdelen. Authenticated key exchange from ideal lattices. EUROCRYPT 2015, LNCS Vol. 9057, pages 719-751. Springer, 2015.

ZZDS14. J. Zhang, Z. Zhang, J. Ding, and M. Snook. Authenticated key exchange from ideal lattices. *IACR Cryptology ePrint Archive*, 2014/589, 2014.

A Communications with the Corresponding Authors of [ZZD$^+$15,ZZDS14] about the Security Proof of Π_2

We have made deep discussions with the corresponding authors of [ZZD$^+$15,ZZDS14] *in regard to the security proof of* Π_2. Here, we briefly report the results from the communications, *which is only for the reference of referees*.

The technical details of the small field attack against Π_1 are examined by one corresponding author and his students, and the validity is explicitly confirmed.

Unfortunately, we could not reach a final consensus from the communications with the corresponding author of Claim 16, *i.e.*, an intermediate claim in the security proof of Π_2. To be frank, the analysis of Claim 16 in [ZZDS14] is over sketchy, and the logic is elusive there. Moreover, according to our communications, the parts related to forking lemma in the analysis of Claim 16 is unnecessary and can be removed there (as explicitly acknowledged by the corresponding author), which results in a further sketched proof. Thanks to the deep discussions, we still reach some important consensus with the corresponding author on the proof details of Claim 16, as presented in Section 7. However, we have a fundamental diverge in reaching the final conclusion. It is unfortunate that, after we made clear of the subtleties related to the divarication, the corresponding author refused further communications. As a consequence, we could not explicitly state our final conclusion on the security proof of Claim 16, but present our observations and clarifications and leave drawing the final conclusion to the reader.

B The Algorithms $\mathcal{A}_1'$ and $\mathcal{V}$ in Pseudocode

B.1 The Algorithm $\mathcal{A}_1'$

Algorithm 1 The algorithm $\mathcal{A}_1'$

```
procedure $(\mathcal{A}_1')^{\mathcal{M}_1}(i, \{s_1, \cdots, s_{i-1}\})$
    Choose $g \in \mathbb{F}_q^\times$ such that $g$ is a primitive element of $\mathbb{F}_q$;
    $d := q_0/n$;
    for every $\tilde{s_i} \in \mathbb{F}_q$ do
        for every $k \in \{g^r \mid r \in [d]\}$ do
            for every $r \in [i-1]$ do
                $h_{k,r} \leftarrow \mathbb{F}_q^\times$;          ▷ Generate nonzero $h_{k,r} \in \mathbb{F}_q$ randomly
            end for
            $\boldsymbol{h}_k := \sum_{r\in[i-1]} h_{k,r} \cdot \boldsymbol{c}_r$;          ▷ Generate the random $\boldsymbol{h}_k$
            $\boldsymbol{e}_k \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{n}}$;          ▷ Generate the small noise $\boldsymbol{e}_k$
            for every $j \in [n]$ do
                $u_{k,j} := \tilde{s_i} \cdot kc_{i,j} + \sum_{r\in[i-1]} s_r h_{k,r} c_{r,j}$;
                $w_{k,j} := \mathsf{Cha}\,(u_{k,j})$;          ▷ Set the desired $\boldsymbol{w}$-entry
                $z_{k,j} := \mathsf{Mod}\,(u_{k,j}, w_{k,j})$;          ▷ Set the desired $\boldsymbol{z}$-entry
            end for
            $b_k \leftarrow \mathcal{M}_1\left(k \cdot \boldsymbol{c}_i + \boldsymbol{h}_k + 2\boldsymbol{e}_k, [w_{k,j}]_{j\in[n]}, [z_{k,j}]_{j\in[n]}\right)$;          ▷ Invoke $\mathcal{M}_1$
        end for
        if $b_k = 1$ for every $k \in \{g^r \mid r \in [d]\}$ then
            return $\tilde{s_i}$;          ▷ This implies the guess $s_i = \tilde{s_i}$ is correct
        end if
    end for
    return $\perp$;
end procedure
```

B.2 The Algorithm $\mathcal{V}$

Algorithm 2 Verification of $(s_1, \cdots, s_n) \overset{?}{=} (\tilde{s_1}, \cdots, \tilde{s_n})$

```
procedure $\mathcal{V}(\tilde{s_1}, \tilde{s_2}, \cdots, \tilde{s_n})$
    $J := \emptyset$;
    repeat
        Choose $k \in [n]$ such that $\{i \in [n] \mid c_{i,k} \neq 0\} \not\subseteq J$;
        Choose an $\mathbb{F}_q$-basis for $\mathcal{U}_k^{\perp}$ randomly, say $\boldsymbol{F}_k = \left\{\boldsymbol{f}_{k,1}, \cdots, \boldsymbol{f}_{k,n-1}\right\} \subseteq \mathbb{F}_q^n$;
        for every $y \in [q_0]$ do
            for every $\boldsymbol{u} \in \boldsymbol{F}_k$ do
                for every $r \in [n]$ do
                    $u_r := y \cdot A_r(\boldsymbol{u})$;
                    $\sigma_r := \mathsf{Cha}\,(u_r)$;   $z_r := \mathsf{Mod}\,(u_r, \sigma_r)$;
                end for
                $b_1(y, \boldsymbol{u}) \leftarrow \mathcal{M}_1\left(\tau(y, \boldsymbol{u}), [\sigma_r]_{r \in [n]}, [z_r]_{r \in [n]}\right)$;
                if $b_1(y, \boldsymbol{u}) \neq 1$ then
                    return 0;        ▷ $\boldsymbol{b}^{(k)} \notin \mathcal{U}^{(k)}$, and hence $(s_1, \cdots, s_n) = (\tilde{s_1}, \cdots, \tilde{s_n})$
                end if
            end for
        end for
        Choose $\boldsymbol{u}^* \in \mathbb{F}_q^n$ randomly such that $A_k(\boldsymbol{u}^*) = 1$;
        for every $y \in [q_0]$ do
            for every $r \in [n]$ do
                $u_r := y \cdot A_r(\boldsymbol{u}^*)$;
                $\sigma_r := \mathsf{Cha}\,(u_r)$;
                $z_r := \mathsf{Mod}\,(u_r, \sigma_r)$;
            end for
            $b_2(y) \leftarrow \mathcal{M}_1\left(\tau(y, \boldsymbol{u}^*), [\sigma_r]_{r \in [n]}, [z_r]_{r \in [n]}\right)$;
        end for
        if $b_2(y) = 0$ for some $y \in [q_0]$ then
            return 0;        ▷ $(s_1, \cdots, s_n) \neq (\tilde{s_1}, \cdots, \tilde{s_n})$
        end if
        $J := J \cup \{i \in [n] \mid c_{i,k} \neq 0\}$;
    until $J == [n]$
    return 1;        ▷ $(s_1, \cdots, s_n) = (\tilde{s_1}, \cdots, \tilde{s_n})$
end procedure
```