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 [ZZDS14,ZZD15]. Similar to HMQV, this ring-LWE based analogue of HMQV proposed there consist of two variants: a two-pass protocol $\Pi_2$, as well as a one-pass protocol $\Pi_1$ that implies a signcryption scheme (named as “deniable encryption” in [ZZD15]). 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 $\Pi_1$ 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 $\Pi_1$ 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 $R_q$ 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 $\Pi_2$ is then revisited. We are stuck at Claim 16 in [ZZDST14], 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 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-Kravitz (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 highcreption (aka the deniable encryption in [ZZDST14,ZZD15]), 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 $K \equiv \mathbb{Q}[x]/\langle f \rangle$ 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,GGH13b,Lyu12,DDL11]. 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 $q \equiv 1 \pmod{2^n}$, and $f(x) = \Phi_{2^n}(x) \in \mathbb{Z}[x]$ 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,DDL12].

RLWE-based Diffie-Hellman and HMQV. We briefly review the abstract basic structure of key exchange over RLWE [ZZDS14,ZZD15,DXL12,Pei14,BCNS15,ADPS16]. Let Alice and Bob denote the two involved parties for simplicity. To Alice (party $i$), the static private key is $(s_i, e_i) \in R_q \times R_q$ (both are “small”), the static public

* Communications with the corresponding authors of [ZZD15,ZZDST14] are briefly summarized in Section A.
key is $p_i = as_i + c \cdot e_i$, where $a \in \mathbb{R}_q$ denotes the public system parameter; the ephemeral private key is $(r_i, f_i) \in \mathbb{R}_q \times \mathbb{R}_q$ (both are “small”), and the ephemeral public key is $x_i = ar_i + c \cdot f_i$. Similar notations $(s_j, e_j), (p_j, r_j, f_j), x_j$ apply to Bob (party $j$). The value $c \in \mathbb{F}_q^*$ 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 $x_i$ to Bob; Then Bob replies to Alice with $(x_j, w_j = g(x_i, r_i, f_j))$, 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 $K_i(x_j, r_i, f_i, w_j) = K_j(x_i, r_j, f_j, w_j)$. Briefly speaking, the underlying key derivation mechanism is based on the bilinear map $(r_i, r_j) \rightarrow d = r_1 \cdot r_2$ of $\mathbb{R}_q \times \mathbb{R}_q$ into $\mathbb{R}_q$, where $d \in \mathbb{R}_q$ denotes the dominant value from which the session-key is derived. However, neither Alice nor Bob could directly compute the dominant value $d$ due to the small noises $f_j$ and $f_i$ 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 $d$. This basic key exchange protocol is thus analogous to Diffie-Hellman, which in turn is based on bilinear map over cyclic group $\mathbb{G}$. In two-pass RLWE-based HMQV analogue, we have $w_j = g(p_j, x_i, s_j, e_j, r_j, f_j)$, and $K_i(p_j, x_j, s_i, e_i, r_i, f_i, w_j) = K_j(p_i, x_j, s_j, e_j, r_j, f_j, w_j)$. Conversely, in one-pass RLWE-based HMQV analogue where the values $(x_i, p_j)$ sent by Bob in the second round is waived, Alice sends $x_i$ as well as the signal $w_i = g(p_j, s_i, e_i, r_i, f_j)$. In this case, $K_i$ and $K_j$ are defined to be: $K_i(p_j, s_i, e_i, r_i, f_j, w_i) = K_j(p_i, x_j, s_j, e_j, w_i)$.

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 $\Pi_2$ and one-pass protocol $\Pi_1$ were proposed. Both $\Pi_2$ and $\Pi_1$ 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 $\Pi_1$, it is claimed to be provably secure “in a weak model similar to [Kna03] which avoids some reasonable insufficiencies for one-pass protocol” ([ZZD+15], page 744). However, the exact security model for $\Pi_1$ 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 $\Pi_1$, 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 $\Pi_2$ is presented in Section 7. Below, we briefly review $\Pi_1$ in Figure 4.1, in accordance with our abstract protocol structure above.

The one-pass AKE scheme $\Pi_1$ The scheme $\Pi_1$, proposed in [ZZDS14,ZZD+15], is built upon the ring-LWE assumption. In $\Pi_1$, $a \leftarrow \mathbb{R}_q$ is the global public parameter, and $M > 0$ is a constant that is sufficiently large. As a two-party AKE protocol, users in $\Pi_1$ are represented by party $i$ and party $j$. For party $i$: the static private key is $(s_i, e_i)$, where $s_i, e_i \leftarrow D_{\mathbb{Z}_{q}},$ its associated public key is $p_i \triangleq as_i + 2e_i \in \mathbb{R}_q$; And its identity issued by the Certificate Authority (CA) is $\mathbb{id}_i$. Similar notations, $s_j, e_j \leftarrow D_{\mathbb{Z}_q}, p_j \triangleq as_j + 2e_j \in \mathbb{R}_q$ and $\mathbb{id}_j$, apply to party $j$. Let $H_1 : \{0,1\}^* \rightarrow D_{\mathbb{Z}_q}$, be a hash function that outputs invertible elements in $\mathbb{R}_q$, and $H_2 : \{0,1\}^* \rightarrow \{0,1\}^n$ be the key derivation function. Both $H_1$ and $H_2$ are regarded as random oracles in $\Pi_1$.

The following functions are essential for the definition of $\Pi_1$. First, When the prime $q$ is clear from the context, define the function $\text{Parity} : \mathbb{F}_q \rightarrow \{0,1\}$, where $\text{Parity} (u) \triangleq u (\text{mod} 2) \in \{0,1\}$. Moreover, we define the function $\text{Mod} : \mathbb{F}_q \times \{0,1\} \rightarrow \{0,1\}$, where $\text{Mod} (u, w) \triangleq \text{Parity} ((u + w \cdot q_0) \text{ mod } q) \in \{0,1\}$. Finally, define the function $\text{Cha} : \mathbb{F}_q \rightarrow \{0,1\}$, such that $\text{Cha} (u) = 0$ if and only if $u \in \{-q^{-1}, \ldots, q^{-1}\}$; Otherwise, $\text{Cha} (u) = 1$. All these functions could be easily generalized to the $n$-dimensional case in the component-wise manner: For instance, $\text{Cha} (v) \triangleq \left[ \text{Cha} (v_j) \right]_{j \in [n]} \in \{0,1\}^n$ for every $v = [v_j]_{j \in [n]} \in \mathbb{F}_q^n$, and it is understood that $\text{Cha} (u) \triangleq \left[ \text{Cha} (u_j) \right]_{j \in [n]}$ for $u = \sum_{j \in [n]} u_j q^{j-1} \in \mathbb{R}_q$.

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 $\Pi_1$, 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 $\Pi_1$ with overwhelming
Fig. 1. Description of $\Pi_1$. Note that in the full description in [ZZDS14, ZZD15], party $i$ applies the rejection sampling operation to generate a “good” $(x_i, w_i)$ 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. 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 $\Pi_1$ and may be of independent interest.

We also notice that the SFA attack may not violate the security claim for $\Pi_1$ made in [ZZD15], as with SFA the malicious user needs to register its public key on its own (while in the security model of [ZZD15], 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 $\Pi_1$ could be proved “secure” within.

Then, the security analysis of the two-pass AKE scheme $\Pi_2$ in [ZZDS14, ZZD15] is revisited. Loosely speaking, the security proof of $\Pi_2$ considers five types of adversaries, Type-I through Type-V; Only Type-I is analyzed in [ZZD15], 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 $(\text{view}_1, \text{view}_2)$ of a PPT adversary $\mathcal{A}$, consider the ability that $\mathcal{A}$ could successfully output a value $\sigma_1, t \in \{1, 2\}$, where $\sigma_2$ is a random value independent of view2 but $\sigma_1$ is essentially committed to view1 and is infeasible to be efficiently computed from view1. The corresponding author of Claim 16 [ZZDS14] concludes that: if view1 and view2 are computationally indistinguishable and $\mathcal{A}$ could not output $\sigma_2$ from view2 with non-negligible probability, then $\mathcal{A}$ should also not be able to output $\sigma_1$ from view1 with non-negligible probability; In particular, whether $\sigma_1$ is distinguishable from $\sigma_2$ does not affect the conclusion. However,
from our view, to get the conclusion, we need to prove that the joint distribution of \((\text{view}_1, \sigma_1)\) is computationally indistinguishable from that of \((\text{view}_2, \sigma_2)\); At least we need to argue the indistinguishability between \(\sigma_1\) and \(\sigma_2\), as the adversarial event is defined not only on the views \((\text{view}_1, \text{view}_2)\) but also on the hidden values \((\sigma_1, \sigma_2)\). 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 \(R_q\) 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 \(R_q\)) first proposed in [LPR13a]. Its basic properties can be summarized as follows. First, the CRT basis for \(R_q\) is unique, and could be found efficiently [LPR13a]. Furthermore, the CRT basis \(\{c_1, \cdots, c_n\}\) is an \(F_q\)-basis for \(R_q\), when \(R_q\) is seen as an \(F_q\)-module of rank \(n\) in the natural way; For instance, every element \(u \in R_q\) could be uniquely written as \(u = \sum_{i \in [n]} u_i c_i, u_i \in F_q\); For simplicity, in this work let \(\eta_i(u) \triangleq u_i \in F_q\) denote the \(i\)-th CRT-coefficient of \(u \in R_q\), and let \(\text{Dim} \ (u) \triangleq \{i \in [n] \mid \eta_i(u) \neq 0\}\). Finally, the equality \(uv = \sum_{i \in [n]} \eta_i(u)\eta_i(v) \cdot c_i\) holds in the ring \(R_q\) for every \(u, v \in R_q\).

In addition to these basic ones, some interesting property regarding \(\{c_1, \cdots, c_n\}\) is further developed in this work. To be precise, by assuming \(c_i = \sum_{j \in [n]} c_{i,j} q^{j-1} \in R_q, c_{i,j} \in F_q\) for every \(i \in [n]\), we shall prove in Lemma 3.2 that for each \(i\), the coefficients \(c_{1,i}, c_{2,i}, \cdots, c_{n,i} \in F_q\) form a geometric sequence (in \(F_q\)). This new property is essential for our SFA against \(\Pi_1\), and may be of independent value.

**The small field attack (SFA) against \(\Pi_1\)** Our analysis on \(\Pi_1\) can be briefly summarized as follows, as is depicted in Figure 2.

- In Section 4, we first review \(\Pi_1\), and then abstract part of valid functionalities of party \(j\) as an oracle \(M_0\) with private key. To efficiently recover the static private key of party \(j\) in \(\Pi_1\), it suffices to show how to recover the private key of \(M_0\) efficiently. To do so, we then define a simplified variant of \(M_0, i.e., M_1\) with secret, which corresponds to the special case when the static public key of the attacker against \(M_0\) is \(0 \in R_q\) by default. Finally, a basic yet efficient attacker \(A_1\) is constructed that can recover the secret of \(M_1\). As a concrete instance of SFA, \(A_1\) implies that there exists an efficient attacker with static public key \(0 \in R_q\) who can break party \(j\) in \(\Pi_1\). This basic attacker already demonstrates the insecurity of \(\Pi_1\) in practice, which is, however, can be easily detected and prevented.

- In Section 5, we continue our analysis on \(M_1\) and construct an efficient attacker \((\mathcal{V}/A_1')_j\) that can break \(M_1\); And, the queries made by \((\mathcal{V}/A_1')_j\) are well designed such that it is hard for \(M_1\) to distinguish those malicious queries from honestly generated ones in practice. The attacker \((\mathcal{V}/A_1')_j\) implies that there exists an efficient attacker, still with static public key \(0 \in R_q\), who can break party \(j\) in \(\Pi_1\); 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 \(M_0\), and design an “undetectable” attacker \(A_0\) against \(M_0\), which is very similar to \((\mathcal{V}/A_1')_j\) and can break \(M_0\) via a sequence of random-looking queries; Furthermore, motivated by the construction of \((\mathcal{V}/A_1')_j\), the static public key of \(A_0\) can be well formed such that it is hard in practice for \(M_0\) to distinguish the static public key of \(A_0\) from that of an honest user. It follows immediately from the construction of \(A_0\) that we can design an “undetectable” attacker \(A\) that can break party \(j\) in \(\Pi_1\), and it is hard in practice for party \(j\) to identify either the static public key of \(A\) or those malicious queries made by \(A\)\(^4\).

**More details about our attack** The vulnerability of \(\Pi_1\) is demonstrated as follows. First, we abstract some of the valid functionalities of the honest party \(j\) in \(\Pi_1\) as an oracle \(M_0\) with private key, where the private key of \(M_0\) corresponds to the static private key of the honest party \(j\). Each query made to \(M_0\) consists of the caller’s public key, the message msg for \(M_0\) to create a new session, and the session key sk\(j\) of the matching session. On each query, \(M_0\) first creates a new session associated with msg, computes its session key sk\(j\), and finally returns 1 if sk\(j\) = 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, ZZD\(^+\)14]. Moreover, for one-pass protocol \(\Pi_1\) 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 \(\Pi_2\) in [ZZDS14, ZZD\(^+\)15], since the static private key of party \(j\) is protected by its ephemeral private key in \(\Pi_2\).
A simplified variant of \(A_0\) against \(M_0\) is first constructed in Section 4.3. Observe that if the \(q\)-th round is devoted to the recovery of \(i\). In this work, \(\text{Parity}(x)\) represents the parity of \(x \in \mathbb{F}_q = \{-\frac{n-1}{2}, \ldots, \frac{n-1}{2}\}\). Moreover, for \(u = \sum_{j=1}^n u_j \xi^{j-1} \in \mathbb{R}_q\), it is understood \(\text{Parity}(u) = [\text{Parity}(u_j)]_{j \in [n]} \in \{0, 1\}^n\).

A simple efficient attacker \(A_1\) against \(M_1\) is first constructed in Section 4.3. Observe that if the \(x\)-entry of any query is of the form \(k \cdot c_i, k \in \mathbb{F}_q, i \in [n]\), then the product \(xs = k\eta_i(s) \cdot c_i\), always belongs to a “small” set \(\{k' \cdot c_i | k' \in \mathbb{F}_q\}\), which is a subfield of the ring \(\mathbb{R}_q\) and of size \(q = \text{poly}(\lambda) \ll q^n = |\mathbb{R}_q|\), making it possible for us to recover \(\eta_i(s) \in \mathbb{F}_q\) efficiently. This explains how our small field attack bears its name. Such observation implies the general structure of the desired attacker \(A_1\): the main body of \(A_1\) is an \(n\)-round loop, and the \(i\)-th round is devoted to the recovery of \(s_i = \eta_i(s) \in \mathbb{F}_q, i \in [n]\). In the \(i\)-th round, given \(s_1, \ldots, s_{i-1} \in \mathbb{F}_q\) and oracle access to \(M_1\), first picks \(s_i \in \mathbb{F}_q\) randomly, guesses \(s_i = \tilde{s}_i \in \mathbb{F}_q\), and then verifies the correctness of this guess via a set \(Q_i(\tilde{s}_i)\) of queries to \(M_1\). The set \(Q_i(\tilde{s}_i)\) is carefully chosen such that the \(x\)-entry is always of the form \(ke_i, k \in \mathbb{F}_q^*\), and the distribution of those query replies under the condition \(s_i = \tilde{s}_i\) is computationally distinguishable from that under the condition \(s_i \neq \tilde{s}_i\). In this manner, \(\tilde{s}_i\) 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 \(s = \sum_{i\in [n]} s_i \cdot c_i\) is recovered.

Of every query made by \(A_1\), its \(w\)- and \(z\)-entries are “random” enough, but its \(x\)-entry is easy to recognize, since \(x \in \{k\eta_i | k \in \mathbb{F}_q^*, i \in [n]\}\) always holds. Equivalently, \(|\dim(x)| = 1\). As a result, it is easy for \(M_1\) to prevent \(A_1\) by requiring that \(|\dim(x)| \geq 2\) for each incoming query. Such restriction is reasonable in the sense that the set \(\{k\eta_i | k \in \mathbb{F}_q^*, i \in [n]\}\) is of “small” size compared with \(\mathbb{R}_q\). Now the first motivating question arises: how to improve \(A_1\) so that we can still recover the secret of \(M_1\), even if the foregoing requirement on the \(x\)-entry is imposed?

Actually, it is not hard to construct an improved variant of \(A_1\), i.e., \(A'_1\), to resolve this problem, as we shall see in Section 5.3. First, given that in the \(i\)-th round, the CRT-coefficients \(s_1, \ldots, s_{i-1}\) has already been recovered successfully, we can make full use of these known CRT-coefficients to re-design the \(x\)-entry. Moreover, since \(s \leftarrow D_{\mathbb{Z}_{\alpha\cdot n}}\) is “small”, it can be proven that the product \(se \in \mathbb{R}_q\) is “small” as well for \(e \leftarrow [-\alpha\sqrt{n}, \ldots, \alpha\sqrt{n}]\). These two observations are essential for us to design the desired \(A'_1\); its general structure is almost the same as that of \(A_1\), except that in its \(i\)-th round, the \(x\)-entry every query in \(Q_i(\tilde{s}_i)\) is always of the form \(x = k \cdot c_i + h + 2e\), where \(k \in \mathbb{F}_q^*, h \leftarrow \{u \in \mathbb{R}_q | \dim(u) = [i - 1]\}\), and \(e \leftarrow \mathbb{Z}_{\alpha\sqrt{n} + 2\alpha\sqrt{n}} = [-\alpha\sqrt{n}, \ldots, \alpha\sqrt{n}]\). By re-designing the \(w\)- and \(z\)-entries appropriately, the efficient attacker \(A'_1\) can be proven to recover every CRT-coefficient \(s_i \in \mathbb{F}_q\) with overwhelming probability.

---

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 \(\Pi_1\), it is sufficient to construct an efficient attacker \(A_0\) that can recover the private key of \(M_0\).

The precise construction of \(A_0\) involves too many details. Fortunately, the following simplified analysis implies how \(A_0\) against \(M_0\) works. For the moment, we assume that the public key of \(A_0\) is \(0 \in \mathbb{R}_q\) (this assumption could be dropped finally). In such simplified setting, we can define an oracle \(M_1\) with secret \(s \leftarrow D_{\mathbb{Z}_{\alpha\cdot n}}\), which could be seen as a simplified variant of \(M_0\). Here, \(s\) corresponds to the static private key of party \(j \in \Pi_1\), and could be seen as an element of \(\mathbb{R}_q\) in the natural way, provided \(q\) is sufficiently large. On input \((x, w, z = [\sigma_j]_{j \in [n]} \in \mathbb{R}_q \times \{0, 1\}^n \times \{0, 1\}^n\), the oracle \(M_1\) first generates a small error \(e \leftarrow \mathbb{Z}_{\alpha\sqrt{n} + 2\alpha\sqrt{n}} = [-\theta, \ldots, \theta]\), and then computes \(\sigma \triangleq \text{Parity}(x + q_0w + 2e) = [\sigma_j]_{j \in [n]}\), and finally returns 1 if and only if \([\sigma_j]_{j \in [n]} = [\sigma_j]_{j \in [n]}\). In this work, \(\text{Parity}(x)\) represents the parity of \(x \in \mathbb{F}_q = \{-\frac{n-1}{2}, \ldots, \frac{n-1}{2}\}\). Moreover, for \(u = \sum_{j=1}^n u_j \xi^{j-1} \in \mathbb{R}_q\), it is understood \(\text{Parity}(u) = [\text{Parity}(u_j)]_{j \in [n]} \in \{0, 1\}^n\).

---

\(j\) in \(\Pi_1\) (in §4.1) | oracle \(M_0\) (in §4.2): an abstraction of party \(j\) in \(\Pi_1\) | oracle \(M_1\) (in §4.3): a simplified variant of \(M_0\) | basic attacker \(A_1\) (in §4.3) against \(M_1\) | improved attacker \(A'_1\) (in §5.1) against \(M_1\) | an attacker \(A\) (in §6) against party \(j\) in \(\Pi_1\) | attacker \(A_0\) (in §6) against \(M_0\) | hybrid attacker \((V/A'_1)_s\) (in §5.2) against \(M_1\)
Clearly \( \dim(k \cdot c_i + h) = [i] = \{1, 2, \ldots, i\} \), and hence the more CRT-coefficients of \( s \) 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 \( \mathcal{A}_1' \), \( h = 0 \in \mathcal{R}_q \) and hence the \( x \)-entry of the form \( x = k e_0 + 2e \). To protect its secret, \( \mathcal{M}_1 \) may reject those queries for which the \( x \)-entries are of the form \( x = k e_0 + 2e \) such that \( k \in \mathbb{F}_q, i \in [n] \) and \( |e|_\infty \) 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 \( x \)-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 \( x \)-entry of every query made by \( \mathcal{V} \) to \( \mathcal{M}_1 \) is of the form \( x_0 + 2e \), where \( \dim (x_0) = I \) and \( e \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{\pi}} \).

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 \( \delta \) of moderately large, and an index set \( I \subseteq [n] \) of size \( \delta \) randomly. Then, feed \( \mathcal{V} \) with \( q^k \) 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 \( x \)-entry of every query made in this phase is always of the form \( x_0 + 2e \), where \( \dim (x_0) = I \) and \( e \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{\pi}} \). Given the randomness of \( I \), it is hard in practice 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 \( s \), as is done in \( \mathcal{A}_1' \).

In particular, the \( x \)-entry of every query made in this phase is always of the form \( x_0 + 2e \) where \( \dim (x_0) \supseteq I \) and \( e \leftarrow \mathbb{Z}^n_{1+2\alpha'\sqrt{\pi}} \), making it more difficult for \( \mathcal{M}_1 \) to identify them.

The notation \( \mathcal{V}(\mathcal{A}_1')_I \) 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 \( \mathcal{V}(\mathcal{A}_1')_I \). 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 \( \mathcal{V}(\mathcal{A}_1')_I \) 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 \( \Pi_1 \)**

It is noteworthy that [ZZDS14, ZZD+15] do not describe precise definitions and proofs of its security claims for \( \Pi_1 \). Instead, some properties of this security model are discussed. For instance, it is claimed in [ZZDS14, ZZD+15] that \( \Pi_1 \) cannot provide forward secrecy and does not stop replays, just as the one-pass HMQR, and that \( \Pi_1 \) “can be proven in a weak model similar to [Kra05]”; What is more, \( \Pi_1 \) “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 \( \Pi_1 \) are unclear to us, as far as we can see, the natural interpretation of the foregoing CCA-security claim about \( \Pi_1 \) is disproved by our SFA attack. We argue that, no matter what the precise interpretations of original security claims were, our attack shows that \( \Pi_1 \) 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 using 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 \( \Pi_1 \) 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 \( \Pi_1 \). Actually, as discussed in [DARFT16], no existing concrete one-pass AKE scheme was affected with attack considered in [Flu16].

2 Preliminaries

Let \( \lambda \) denote the security parameters throughout this work. Let \( B \triangleq \{0, 1\} \). For an odd integer \( p > 0 \), let \( \mathbb{Z}_p \triangleq \left\{ -\frac{p-1}{2}, \ldots, \frac{p-1}{2} \right\} \); For instance, \( \mathbb{Z}_4 \triangleq \{-1, 0, 1\} \). For every positive integer \( k \), let \( [k] \) denote the finite set \( \{1, 2, \ldots, k\} \subseteq \mathbb{Z} \).

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 \( \lambda \); 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 \( \{-q_0, \ldots, q_0\} \). Define \( \mathbb{F}_q^\times \triangleq \mathbb{F}_q \setminus \{0\} \), and \( \text{zone}_0 \triangleq \{ -q_0/2, \ldots, +q_0/2 \} \subseteq \mathbb{F}_q \). And, when comparison is carried out between \( a, b \in \mathbb{F}_q \), both \( a, b \) are regarded as \textit{real numbers}.

Unless otherwise stated, vectors are represented in \textbf{column} form in this work, and the \( n \)-dimensional vector \((u_1, \ldots, u_n)^\top \) is usually abbreviated as \([u_j]_{j \in [n]}\). Let \( 0 \triangleq (0, \ldots, 0)^\top \in \mathbb{F}_q^n \). When either \( u \in \mathbb{F}_q^n \) or \( u \in \mathbb{Z}^n \), let \( \|u\|_2 \) and \( \|u\|_\infty \) denote the \( \ell_2 \)- and \( \ell_\infty \)-norms of \( u \); 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 \( \mu_j : \mathbb{F}_q \to \mathbb{F}_q \) for every \( j \in [n] \) such that \( \mu_j ([u_j]_{j \in [n]}) \triangleq u_j \).

For an event \( E \), the notation \( \Pr \left[ E \right] = \text{negl} (\lambda) \) indicates the probability that \( E \) occurs is negligible in \( \lambda \). Conversely, if \( \Pr \left[ E \right] = 1 - \text{negl} (\lambda) \), we say that \( E \) occurs with \textit{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 \( x_1, x_2, \ldots, x_k \), this process is usually written as \( y := A (x_1, x_2, \ldots, x_k) \). Conversely, for a \textit{randomized} algorithm \( B \), which returns \( y \) on input \( x_1, x_2, \ldots, x_k \) as well as the random nonce \( r \), this process is usually abbreviated as \( y := B (x_1, x_2, \ldots, x_k; r) \), or simply \( y \leftarrow B (x_1, x_2, \ldots, x_k) \). The notation \( \Pr \left[ R_1, \ldots, R_k : E \right] \) denotes the probability of the event \( E \) after the \textit{ordered} execution of random processes \( R_1, \ldots, R_k \). For a \textit{finite} set \( S \), let \( x \leftarrow S \) denote a sample drawn from the \textit{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 \( \Pi_1 \) 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].

\textbf{The rings} \( \mathbb{R} \) \textbf{and} \( \mathcal{R}_q \). Let \( \zeta \) 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 \( \mathbb{R} \triangleq \mathbb{Z}[\zeta] \) be the ring of integers of the number field \( \mathbb{Q}(\zeta)/\mathbb{Q} \). Moreover, define the principal ideal \( (\zeta) = q\mathbb{R} \) and its associated quotient ring \( \mathcal{R}_q \triangleq \mathbb{R}/q\mathbb{R} \). In this work, each coset of \( q\mathbb{R} \) in \( \mathbb{R} \) is naturally represented by a \textit{unique} element in the set \( \left\{ \sum_{j \in [n]} u_j \zeta^{j-1} \left| u_j \in \mathbb{F}_q \right. \right\} \).

Since \( \zeta^0, \ldots, \zeta^{n-1} \) constitute an \( \mathbb{F}_q \)-basis for the free \( \mathbb{F}_q \)-module \( \mathcal{R}_q \) of rank \( n \), every \( u = \sum_{j \in [n]} u_j \zeta^{j-1} \in \mathcal{R}_q \) could be identified with the (column) vector \([u_1, \ldots, u_n]^\top = [u_j]_{j \in [n]} \in \mathbb{F}_q^n \), \textit{and vice versa}. Such identification is denoted as \( u \sim [u_j]_{j \in [n]} \) 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, \textit{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 \( \mu(u) \triangleq [\mu_j(u)]_{j \in [n]} \), which induces an \( \mathbb{F}_q \)-module isomorphism. It is understood that \( \|u\|_2 \triangleq \|\mu(u)\|_2 \) and \( \|u\|_\infty \triangleq \|\mu(u)\|_\infty \) for every \( u \in \mathcal{R}_q \). 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 \( 0 \) denote both the vector \( (0, \ldots, 0)^\top \in \mathbb{F}_q^n \) 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 $D_{\mathbb{Z},\alpha}(x) = \rho_\alpha(x)/\rho_\alpha(0)$, $x \in \mathbb{Z}$. 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 $\Pr[\|\varepsilon\|_2 \leq \alpha \sqrt{n}] = 1 - \negl(\lambda)$ [Reg09,LR13a]. 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, $\varepsilon$ 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 $n = \poly(\lambda) \geq 16$ be a power-of-two. If $\alpha_i = \omega(\sqrt{\log n})$ and $e_i \leftarrow D_{\mathbb{Z^n},\alpha_i}$ for every $i \in \{1, 2, 3\}$, then every $e_i$ could be regarded as an element of $\mathcal{R}$ in the natural way with overwhelming probability; Moreover, the following inequalities hold with overwhelming probability:

$$\|e_1e_2\|_2 \leq n \cdot \alpha_1\alpha_2, \quad \|e_1e_2e_3\|_\infty \leq n^2 \cdot \alpha_1\alpha_2\alpha_3.$$  

\[\square\]

The ring-LWE (RLWE) problem  The following reviews a special case [DD12] of the original ring-LWE problem [LR13a], since this special case, instead of the original one, serves as the underlying hard problem of Definition 2 [LPR13a,LPR13b,DD12]. Furthermore, the hardness of (search/decisional) ring-LWE problem is implied by the following Theorem.

Each ring-LWE instance is parameterized by $n = m(\lambda)$, $q = q(\lambda)$ and $\alpha = \alpha(\lambda)$, where $n \geq 16$ is a power-of-two, $q$ is a positive rational prime such that $q \equiv 1 \pmod{2n}$, and $\alpha \geq 0$. For every (fixed) $s \in \mathcal{R}_q$, we define the ring-LWE distribution $A_{n,q,o,s}$ over $\mathcal{R}_q \times \mathcal{R}_q$; a sample drawn from $A_{n,q,o,s}$ is generated by first choosing $a \leftarrow \mathcal{R}_q$, $\varepsilon \leftarrow D_{\mathbb{Z^n},\alpha}$, and then outputting the ring-LWE sample $(a, b \triangleq as + \varepsilon) \in \mathcal{R}_q \times \mathcal{R}_q$ or equivalently $(a, b = as + 2\varepsilon)$ as in [ZZD14,ZZD15].

Definition 2 [LR13a,LR13b,DD12]. For the ring-LWE problem, its search variant is defined as follows: given access to arbitrarily many independent samples drawn from $A_{n,q,o,s}$ for some arbitrary $s \in \mathcal{R}_q$, the problem asks to recover $s \in \mathcal{R}_q$. In contrast, the decisional variant asks to distinguish, with non-negligible advantage, between arbitrarily many independent samples from $A_{n,q,o,s}$ for a random $s \leftarrow \mathcal{R}_q$, 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 [LR13a,DD12]. Furthermore, the hardness of (search/decisional) ring-LWE problem is implied by the following Theorem.

Theorem 3 [LR13a,DD12]. For the RLWE problem defined in Definition 2 if there is an efficient algorithm that can distinguish, with $1/poly(\lambda)$ advantage, between $\ell$ samples drawn from the uniform distribution over $\mathcal{R}_q \times \mathcal{R}_q$ and $\ell$ samples drawn from $A_{n,q,o,s}$, where $\beta = \sqrt{n \log(q(\lambda))}/\log(q(\lambda))$, then there exists an efficient quantum algorithm that runs in time $O(q \cdot poly(m))$ and solves the approximate SVP problem to within a factor $O(\sqrt{2n}/\alpha)$ in any ideal of $\mathbb{Z}[\zeta]$.

\[\square\]

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 [LR13a]. 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 $\{\omega_1, \cdots, \omega_n\} \subseteq \mathbb{F}_q^n \setminus \{\pm 1\}$ be the set of elements in $\mathbb{F}_q^n$ that are of multiplicative order $2n$. Since the polynomial $\Phi_{2n}(x) = \prod_{\omega \in [n]}(x - \omega)$ (mod $q$) is separable over $\mathbb{F}_q$ by the assumption on parameters, the principal ideal $q\mathcal{R}$ of $\mathcal{R}$ could be factored as $q\mathcal{R} = \prod_{\omega \in [n]}q\mathcal{R}_\omega$, where every nonzero prime ideal $q_\omega = (q, \zeta - \omega)$ by a suitable ordering, and the norm of every $q_\omega$ in $\mathcal{R}$ is $|q\mathcal{R}_\omega| = q^{\deg(x - \omega)} = q$. Hence, for every quotient ring $\mathcal{R}/q_\omega$ is a finite field of prime order $q$, indicating that $\mathcal{R}/q_\omega \cong \mathbb{F}_q$. Hence, the ring $\mathcal{R}_q$ could be identified with the direct product of $n$ small finite field, each of prime order $q = poly(\lambda)$. This explains how our notion of small field attack bears its name.

As the distinct nonzero prime ideals $q_1, \cdots, q_n$ are necessarily pairwise coprime, it follows from the Chinese remainder theorem that $\mathcal{R}_q = \mathcal{R}/q\mathcal{R} \cong \prod_{i \in [n]} \mathcal{R}/q_i$ under the natural ring isomorphism $u + q\mathcal{R} \mapsto (u +
Fact 4 \([LPR13a,LPR13b]\). For the CRT basis \(\{c_1, \ldots, c_n\}\) for \(R_q\),

(a) Given \(n\) and \(q\) (in unary form), the CRT basis for \(R_q\) could be found efficiently.
(b) Every \(u \in R_q\) can be written uniquely as \(u = \sum_{i \in [n]} u_i \cdot c_i\), where \(u_i \in F_q\).
(c) Let \(u = \sum_{i \in [n]} u_i \cdot c_i\) and \(v = \sum_{i \in [n]} v_i \cdot c_i\). Then for every \(k \in F_q\), we have:

\[ k \cdot u + v = \sum (u_i + v_i) \cdot c_i. \]

Throughout this work, when \(n, q\) are clear from the context, let \(\{c_1, \ldots, c_n\}\) denote the CRT basis for \(R_q\).

Lemma 5. With the notations defined previously, we have \(c_{i,j} = c_{i,n} \cdot \omega_i^{n-j} \neq 0 \in F_q\) for every \(i, j \in [n]\).

Proof. Fix \(i \in [n]\). For the element \(\zeta \in R\), we have

\[ \zeta + q_i = \zeta + \{q_i, \zeta - \omega_i\} = \omega_i + \{q, \zeta - \omega_i\} \in R/q_i. \]

It follows that in the ring \(R_q\), \(\zeta \equiv \omega_i \cdot c_i \pmod{q_i}\).

On the one hand, by Fact 4 we have that in \(R_q\),

\[ \zeta \cdot c_i = \omega_i \cdot c_i = \sum_{j \in [n]} (\omega_i \cdot c_{i,j}) \zeta^{j-1} \in R_q. \]

On the other hand, the following equality holds in \(R_q\):

\[ \zeta \cdot c_i = \zeta \left( c_{i,1} + c_{i,2} \zeta + \cdots + c_{i,n} \zeta^{n-1} \right) = c_{i,n} \zeta^n + \sum_{j \in [n-1]} c_{i,j} \zeta^j, \]

where the equality \(0 = \Phi_{2n}(\zeta) = \zeta^n + 1 \in R_q\) is implicitly applied.

As \(\zeta, \zeta, \ldots, \zeta^{n-1}\) form an \(F_q\)-basis for the \(F_q\)-module \(R_q\), we have

\[ c_{i,1} = \omega_1 c_{i,2}, \ldots, c_{i,n-1} = \omega_1 c_{i,n}, \quad -c_{i,n} = \omega_1 c_{i,1}. \]

Equivalently, \(c_{i,j} = c_{i,n} \cdot \omega_i^{n-j} \in F_q\) for every \(j \in [n]\).

If \(c_{i,n} = 0\), then \(c_{i,j} = 0\) for every \(j \in [n]\), making \(c_i = 0\), contradictory to the definition of the CRT basis for \(R_q\). 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 $\Pi_1$: a Warm-up

We first review the one-pass AKE scheme $\Pi_1$ [ZZDS14,ZZD+15] in Section 4.1. Then in Section 4.2 the honest party $j$ in $\Pi_1$ is abstracted as an oracle $M_0$ with private key, such that to efficiently recover the static private key of party $j$ in $\Pi_1$, it suffices to show how to recover the private key of $M_0$ efficiently, provided that adversaries are allowed to register public keys on behalf of dishonest users.

4.1 The One-Pass AKE Scheme $\Pi_1$

The one-pass AKE scheme $\Pi_1$ [ZZDS14,ZZD+15] is built upon the RLWE problem in Definition 2 where every RLWE sample is of the form $(a, as + 2e)$. In $\Pi_1$, $a \leftarrow R_q$ is a global public parameter, and $M > 0$ is a sufficiently large constant. As a two-party AKE scheme, users in $\Pi_1$ 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 $(s_i, e_i)$, where $s_i, e_i \leftarrow D_{Z^{n,\alpha}}$; its associated static public key is $p_i \triangleq as_i + 2e_i \in R_q$; and its identity issued by the Certificate Authority (CA) is $id_i$. Similar notations, $s_j, e_j \leftarrow D_{Z^{n,\alpha}}, p_j \triangleq as_j + 2e_j \in R_q$, and $id_j$, apply to party $j$. Let $H_1 : \{0,1\}^* \rightarrow D_{Z^{n,\alpha}}$ be a hash function that outputs invertible elements in $R_q$, and $H_2 : \{0,1\}^* \rightarrow \{0,1\}^n$ be the key derivation function. Both $H_1$ and $H_2$ are regarded as random oracles in $\Pi_1$.

The following functions are essential for the definition of $\Pi_1$. First, when the prime $q$ is clear from the context, define the function $Parity : F_q \rightarrow \{0,1\}$, where $Parity (u) \triangleq u(\text{mod } 2) \in \{0,1\}$.

For every $u \in F_q$,

(a) We always have $v \triangleq u + Cha (u) \cdot q0 \in \{-q0/2, \cdots , +q0/2\} \subseteq F_q$.
(b) The value $Parity (v) = Mod (u, Cha (u)) \in B$ is immune to a short even noise in the sense that

$$Parity (v) = Parity (v + 2e)$$

for every $-q0/4 < e < q0/4$.

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

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

for every $-q0/4 < e < q0/4$.


\[\square\]

The one-pass scheme $\Pi_1$ can be roughly described in the following Figure 3 [ZZDS14,ZZD+15].

Correctness analysis of $\Pi_1$: Roughly speaking, the correctness of this scheme states that $sk_i = sk_j$ holds $w.o.p.$ On the one hand, since $H_2(\cdot)$ is modeled as a random oracle and $id_i, id_j, x_i, u_j$ 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

$$k_i - k_j = 2 \cdot (e_i e_j + r_i e_j + g_i - c e_i s_j - f_i s_j - e g_j).$$

By Lemma 4 $\|k_i - k_j\|_\infty \leq 2 (\alpha \beta + \beta \sqrt{\pi} + 2n \alpha \beta + 2n^2 \alpha^2 \gamma)$ holds $w.o.p.$ Thus, when $q$ is sufficiently large, $\|k_i - k_j\|_\infty < q0/2$ holds $w.o.p.$ It follows from Fact 6 (b) that $\sigma_i = Mod (k_i, Cha (k_i)) = Mod (k_j, Cha (k_j)) = \sigma_j$ holds $w.o.p.$ This finishes the correctness analysis of $\Pi_1$.

Clearly Lemma 4 is essential for the correctness of $\Pi_1$. However, when $n$ is not a power-of-two, or $q \neq 1 \mod 2n$, inequalities in Lemma 4 may no longer hold, which implies the underlying hard problem of $\Pi_1$ must be Definition 2 a special case of the original ring-LWE problem defined in LPR13a [LPR13b].

Security analysis of $\Pi_1$: It is claimed in [ZZDS14,ZZD+15] that when $n$ is a power-of-two, $0.97n > 2\lambda$, $\beta = \omega(\alpha \gamma \sqrt{n} \log n)$, the prime $q > 203$ satisfies $q \equiv 1 \mod 2n$, if the associated ring-LWE problem is “hard”, then $\Pi_1$ is secure. In addition, four groups of parameters are suggested in [ZZD+15,ZZDS14] to instantiate $\Pi_1$. For instance, in one typical group of suggested parameters, $n = 1024, q \approx 2^{30}, \alpha = 3.397, \beta \approx 2^{16.1}, \gamma = \alpha$. 

sk: \( r_i, f_i \sim \mathcal{D}_{Z^n, \alpha} \)
ephemeral pk: \( x_i = ar_i + 2f_i \)
\( c = H_1(\text{id}_i, \text{id}_j, x_i) \)
\( k_i = p_j(s_i c + r_i) + 2g_j \)
\( w_i = \text{Cha}(k_i) \)
\( \sigma_i = \text{Mod}(k_i, w_i) \)
\( \text{sk}_i = H_2(\text{id}_i, \text{id}_j, x_i, w_i, \sigma_i) \)

\( \text{pk}_i = a s_i + 2e_i \in \mathcal{R}_q \)

\( \text{pk}_j = a s_j + 2e_j \in \mathcal{R}_q \)

(\( x_i, w_i \))

\( c = H_1(\text{id}_i, \text{id}_j, x_i); g_j \sim \mathcal{D}_{Z^n, \alpha} \)
\( k_j = (p_j c + x_i)s_j + 2cg_j \)
\( \sigma_j = \text{Mod}(k_j, w_i) \)
\( \text{sk}_j = H_2(\text{id}_i, \text{id}_j, x_i, w_i, \sigma_j) \)

\[ \text{sk}(s_i \sim \mathcal{D}_{Z^n, \alpha}, e_i \sim \mathcal{D}_{Z^n, \alpha}) \]
\[ \text{pk}: p_i = a s_i + 2e_i \in \mathcal{R}_q \]

Fig. 3. General description of \( \Pi_i \). Note that in the full description of \( \Pi_i \) in [ZZD14,ZZD+15], party \( i \) applies the rejection sampling to generate a “good” \((x_i, w_i)\) 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 \( \Pi_1 \), 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 \( \Pi_1 \), 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 \( \Pi_1 \).

First, notice that in \( \Pi_1 \), the session key derivation function \( H_2(\cdot) \) is modeled as a random oracle. Moreover, notice that every time party \( j \) generates its session key by invoking \( \text{sk}_j \triangleq H_2(\text{id}_i, \text{id}_j, x_i, w_i, \sigma_j) \), all the input values except \( \sigma_j \) are known to party \( i \). It follows that except with negligible probability, if party \( i \) is able to figure out the session key \( \text{sk}_j \) 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 \( \sigma_j \) 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 \( \Pi_1 \). The private key of \( \mathcal{M}_0 \) is \((s, e)\) and the associated public key is \( p \), where \( s, e \sim \mathcal{D}_{Z^n, \alpha} \) and \( p \triangleq a \cdot s + 2e \in \mathcal{R}_q \) (recall that \( a \leftarrow \mathcal{R}_q \) is a global parameter in \( \Pi_1 \)). Moreover, as a simulator of party \( j \), the identifier of \( \mathcal{M}_0 \) is denoted by \( \text{id} \). On input \((\text{id}^*, p^*, x, w, z)\), where \( x \in \mathcal{R}_q, z, w \in \mathbb{B}^n \), and \( \text{id}^* \) denotes the identifier of the initiator with static public key \( p^* \in \mathcal{R}_q \), \( \mathcal{M}_0 \) does the following: it first samples \( g \sim \mathcal{D}_{Z^n, \alpha} \), then computes \( c \leftarrow H_1(\text{id}^*, \text{id}, x) \), \( k := (p^* c + x)s + q_0 w + 2cg \), and \( \sigma := \text{Parity}(k) \in \mathbb{B}^n \); finally, \( \mathcal{M}_0 \) returns 1 if and only if \( z = \sigma \).

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 4.3, an efficient attacker \( \mathcal{A}_0 \) that can recover the private key \((s, e)\) 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 \( \Pi_1 \).

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 \( p^* = 0 \in \mathcal{R}_q \). By definition, for the input \((\text{id}^*, p^* = 0, x, w, z)\) made by this attacker, \( \mathcal{M}_0 \) first samples \( g \sim \mathcal{D}_{Z^n, \alpha} \), then computes \( c \leftarrow H_1(\text{id}^*, \text{id}, x) \), \( k := (p^* c + x)s + q_0 w + 2cg = x s + q_0 w + 2cg \), and \( \sigma := \text{Parity}(k) \in \mathbb{B}^n \); finally, \( \mathcal{M}_0 \) returns 1 if and only if \( z = \sigma \). Notice that \( \|cg\|_\infty \leq n \cdot \alpha \gamma \) by Lemma 1.
This simplified analysis motivates us to define the oracle $M_1$ with secret $s \leftarrow D_{Z^n, \alpha}$: on input $(x, w, z = [z_j]_{j \in [n]}) \in \mathcal{R}_q \times \mathbb{Z}^n \times \mathbb{Z}^n$, the oracle $M_1$ first generates a small error $\varepsilon \leftarrow \mathbb{Z}_{1+2\theta}^n = \{ -\theta, \cdots, \theta \}^n$, and then computes $\sigma \triangleq \text{Parity}(xs + q_0w + 2\varepsilon) = [\sigma_j]_{j \in [n]}$, 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, $M_1$ could be seen as a simplified variant of $M_0$ with $\theta = n\alpha \gamma$. And, the rest of this section is devoted to the construction of a basic yet efficient attacker $A_1$ that, given oracle access to $M_1$, can recover the secret of $M_1$ w.o.p.

**The CRT basis w.r.t. $A_1$** The construction of $A_1$ could be seen as a simple application of the CRT basis $\{e_1, \cdots, e_n\}$ for $\mathbb{Z}_q$. Recall that with $\{e_1, \cdots, e_n\}$ that is efficiently computable, every element $u \in \mathcal{R}_q$ can be uniquely written as $u = \sum_{i \in [n]} \eta_i(u) \cdot e_i$. Moreover, we have $uv = \sum_{i \in [n]} \eta_i(u) \eta_i(v) \cdot e_i$ for every $u, v \in \mathcal{R}_q$.

Thus, to recover $s \in \mathcal{R}_q$, it suffices to recover all of its CRT-coefficients $s_i \triangleq \eta_i(s) \in \mathbb{Z}_q, i \in [n]$. Furthermore, notice that for every query $(x, u, z)$ made to $M_1$, if $x = k \cdot c_i$ for some $k \in \mathbb{Z}_q, i \in [n]$, then we have $xs = ks_i \cdot c_i$. In such setting, the product $xs$ always falls into a subgroup $\{kc_i | k \in \mathbb{Z}_q\}$ of $\mathcal{R}_q$, which is of “small” size $q = \text{poly}(\lambda)$, making it possible for us to recover every $s_i \in \mathbb{F}_q$ in a sequential manner.

**General structure of $A_1$** By properties of the CRT basis for $\mathcal{R}_q$, to recover the secret $s \in \mathcal{R}_q$ of $M_1$, it suffices to recover every $s_i \triangleq \eta_i(s) \in \mathbb{F}_q$. 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 $A_1$ as follows: the main body of $A_1$ consist of an $n$-round loop, and the $i$-th round is devoted to the recovery of $s_i = \eta_i(s) \in \mathbb{Z}_q, i \in [n]$: In the $i$-th round, given $s_1, \cdots, s_{i-1}$ oracle access to $M_1$, it first picks $\tilde{s}_i \in \mathbb{F}_q$ randomly, guesses $s_i = \tilde{s}_i \in \mathbb{F}_q$, and then verifies the correctness of this guess via a set $Q_i(\tilde{s}_i)$ of queries to $M_1$; The set $Q_i(\tilde{s}_i)$ 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 $Q_i(\tilde{s}_i)$** For the moment, assume $s_1, \cdots, s_{i-1}$ have been recovered successfully and $\tilde{s}_i \in \mathbb{F}_q$ is fixed. We are to constructing the recovery of the guess $s_i = \tilde{s}_i$ by designing the desired query set $Q_i(\tilde{s}_i)$.

Jumping ahead, $Q_i(\tilde{s}_i) = \left\{ Q_k \triangleq (x_k = kc_i, w_k = [w_{k,j}]_{j \in [n]}, z_k = [z_{k,j}]_{j \in [n]}) \mid k \in S_q \subseteq \mathbb{F}_q^\times \right\}$, where $S_q$ is a nonempty proper subset of $\mathbb{F}_q^\times$ to be defined later. It remains to specify the $w$- and $z$-entries of every query in $Q_i(\tilde{s}_i)$.

For the moment, we assume $s_1 = \tilde{s}_1$. In such case, for the query $Q_k = (kc_i, w_k, z_k) \in Q_i(\tilde{s}_1)$, if $w_k = \text{Cha}(\mu_j(k\tilde{s}_i, c_i))$ for every $j \in [n]$, then every $\mu_j(kc_i, s + q_0w_k)$ into the “secure zone” zone$_0$ by Fact 6(a); Moreover, when every $\mu_j(kc_i, s + q_0w_k) \in \text{zone}_0$, Fact 6(b) guarantees that the small and even noise $2\varepsilon$ does not affect the parity values, provided that $\theta$ is “small” relative to $q$. Therefore, $M_1$ would return 1 on $Q_k$ if $z_k = \text{Mod}(kc_i, w_k) = \text{Parity}(kc_i(s + q_0w_k))$. 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 $S_q \triangleq \{ g^r \mid r \in [d] \}$ and $d \triangleq \frac{q-1}{2\theta}$. Define $Q_i(\tilde{s}_i) = \left\{ Q_k = (kc_i, [w_{k,j}]_{j \in [n]}, [z_{k,j}]_{j \in [n]}) \mid k \in S_q, j \in [n], w_{k,j} = \tilde{s}_i \cdot kc_i, z_{k,j} = \text{Cha}(w_{k,j}), [z_{k,j}]_{j \in [n]} = \text{Mod}(w_{k,j}, w_k) \right\}$.

**If $q > 1 + \max \{ 8\theta, \ 2\sqrt{n} \}$, then except with negligible probability, $s_i = \tilde{s}_i$ if and only if $M_1$ returns 1 on every query in $Q_i(\tilde{s}_i)$.**

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

Recall that for the query $Q_k = (x_k = k \cdot c_i, w_k = [w_{k,j}]_{j \in [n]}, z_k = [z_{k,j}]_{j \in [n]}) \in Q_i(\tilde{s}_i)$, $M_1$ first generates $\varepsilon_k \leftarrow \mathbb{Z}_{1+2\theta}$, and then computes $v_k \triangleq x_k \cdot s + q_0 \cdot w + 2\varepsilon_k = ks_i \cdot c_i + q_0 \cdot w_k + 2\varepsilon_k$

\[ \begin{bmatrix} s_i \cdot k \cdot c_{i,1} + q_0 \cdot w_{k,1} \\ \vdots \\ s_i \cdot k \cdot c_{i,n} + q_0 \cdot w_{k,n} \end{bmatrix} + \begin{bmatrix} 2\varepsilon_{k,1} \\ \vdots \\ 2\varepsilon_{k,n} \end{bmatrix} (\varepsilon_{k,j} \triangleq \mu_j(\varepsilon_k)) \]
Finally, $M_1$ returns 1 if $\text{Parity}(v_{k,j}) = z_{k,j}$ for every $j \in [n]$, where $v_{k,j} \triangleq \mu_j(v_k)$; Otherwise, $M_1$ returns 0.

Notice that we always have $u_{k,j} + \text{Cha}(u_{k,j}) \cdot q_0 \in \text{zone}_0$ by Fact 6(a). Also, it follows from the inequality $q > 1 + \sqrt{2\log n}$ 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 $M_1$ returns 1 on every $Q_b$ by definition.

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

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

and we claim that $\{-1,1\} \cap \text{offset}(\Delta s_i) \neq \emptyset$, i.e., either $1 \in \text{offset}(\Delta s_i)$ or $-1 \in \text{offset}(\Delta s_i)$. 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 $\Delta s_i c_{i,n} = g^{r*}$ where $e^* \in [g-1]$. Clearly there exists a $r^* \in [d]$ such that $d | (e^*+r^*)$, and $(e^*+r^*)/d \in [2n]$. Then

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

It is easy to see that 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 $\Delta s_i \cdot c_{i,n} \cdot k' \cdot \omega_i^{n-j} = \omega_i^0 = k \in \mathbb{F}_q$.

When $\Delta s_i \cdot k' \cdot c_{i,j} = \pm 1$, it is easy to verify that $z_{k,j} \neq \text{Parity}(v_{k,j})$ by Fact 6(c). Equivalently, the associated $j^*$-th equality of $Q_{k^*}$ does not hold, and $M_1$ returns 0 on the query $Q_{k^*} \in Q_i(x_i)$.

The following theorem follows immediately from Lemma 7.

**Theorem 8.** When $q > 1 + \max \{ 8\theta, 2\sqrt{n} \}$, the efficient attacker $A_1$ defined previously can recover the secret $s$ of $M_1$ with overwhelming probability, by making at most $n \cdot q \cdot \frac{1}{2n} = \text{poly}(\lambda)$ queries to $M_1$. □

**Remarks** First, the proof of Lemma 7 implies that the exact distribution form of the noise term $\varepsilon$ generated by $M_1$ does not affect the correctness of $A_1$; Only its support does. This explains why the noise term in $M_1$ is simply defined to be drawn from the uniform distribution over its support $Z_2^n$. Moreover, it is not hard to see that the index set $S_y = \{ g^r | r \in [d] \}$ could be replaced by any complete system of representatives of cosets in the quotient group $\mathbb{F}_q^* / G$, where $G \triangleq \langle \omega \rangle$ is the unique subgroup of the cyclic $\mathbb{F}_q^*$ satisfying $|G| = 2n$. All these observations show that $s$ a concrete instance of SFA, the basic attacker $A_1$ is versatile.

The success of $A_1$ relies heavily on the notion of the CRT basis $c_1, \ldots, c_n$ for $R_q$ as well as its property, which explains why our attackers ($A_1$ as well as its improved variants) are called small field attackers. Computer experiments have justified the correctness of our small field attacker $A_1$ against $M_1$; In particular, $A_1$ succeeds when the oracle $M_1$ is instantiated with these four groups of suggested parameters in [ZZD+13][ZZDS14] ($\theta := n\alpha\gamma$).

The existence of $A_1$ implies that there exists an efficient attacker with public key $0 \in R_q$ that can recover the private key of $M_0$ w.o.p. Hence, $A_1$ itself suffices to show the vulnerability of $\Pi_1$.

In essence, our small field attack against $\Pi_1$ is equivalent to solving a problem that is much harder than the adaptive variant of RLWE problem, where attacker can adaptively choose the alpha-part in each sample [a, b]. Although the adaptive RLWE can be solved without the CRT basis of $R_q$, the CRT basis is necessary and plays an essential role in our SFA.

Although we assume $s_1, \ldots, s_{i-1} \in \mathbb{F}_q$ are known before the $i$-th round of $A_1$, such assumption is unnecessary for $A_1$. Nevertheless, this assumption is essential for us to construct an improved variant of $A_1$, as we shall see in Section 5.
5 Making Small Field Attack “Undetectable”

In this section, we continue our analysis on the oracle $M_1$, and show that we can construct an efficient hybrid attacker $(V/A'_1)_q$ against $M_1$, whose malicious queries are as “random-looking” as possible.

5.1 The Improved Attacker $A'_1$ Against $M_1$

Motivating question #1 The success of $A_1$ relies on the assumption that $M_1$ imposes no restrictions on the incoming queries. Intuitively, for every query made by $A_1$, the $w$- and $z$-entries seem “random” enough; However, the algebraic structure of $x$-entry is rather simple, as the $x$-entry always falls into the set $\{kc_i \mid k \in \mathbb{F}, i \in [n]\}$ with size $nq \ll q^n$. Hence, it is easy for $M_1$ to identify those malicious queries made by $A_1$, and the attacker $A_1$ will no longer work if the oracle $M_1$ additionally requires that $x \notin \{kc_i \mid k \in \mathbb{F}, i \in [n]\}$. Such additional requirement seems reasonable, given that $nq \ll q^n$.

Thus, our first motivating question arises: can we improve $A_1$ so that it can still recover the secret of $M_1$, even if the aforementioned requirement is imposed by $M_1$? The answer is affirmative, as we shall see later.

5.2 Construction of $A'_1$

Nevertheless, we can construct an improved variant of $A_1$, i.e., $A'_1$, to resolve the issue proposed by the motivating question #1. Its general structure is similar to that of $A_1$, and the only difference lies in the definition of the query set in each round.

The design of query set in $A'_1$ relies on the following two observations. First, recall that in the $i$-th round of $A_1$, $s_1, \ldots, s_{i-1} \in \mathbb{F}_q$ are assumed to be known already. Thus, we can use these known CRT-coefficients of $s \in \mathcal{R}_q$ to re-design the $x$-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 $x$-entry, provided that $q$ is sufficiently large. In sum, in the $i$-th round, for every query made by $A'_1$ to $M_1$, the $x$-entry is of the form

$$x = kc_i + h + 2e,$$

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

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

The following lemma characterizes the query set $Q'_i(s_i)$ used by $A'_1$ in its $i$-th round.

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

$$Q'_i(s_i) = \left\{ Q'_k = \left( k \cdot c_i + h_k + 2e_k, [w_{k,j}]_{j \in [n]}, [z_{k,j}]_{j \in [n]} \right) \mid k \in S_q, j \in [n], h_{k,1}, \ldots, h_{k,i-1} \leftarrow \mathbb{Z}_q^n, h_k = \sum_{r \in [i-1]} h_{k,r} \cdot e_r + \mathbb{Z}_q^{1+2\sqrt{n}}, u_{k,j} = s_i \cdot k c_i + \sum_{r \in [i-1]} s_i h_{k,r} c_r, w_{k,j} = \text{Cha}(u_{k,j}), z_{k,j} = \text{Mod}(u_{k,j}, w_{k,j}) \right\}.$$

If $q > 1 + 8(\theta + n\alpha')$, then except with negligible probability, $s_i = s_i$ if and only if $M_1$ returns 1 on every query in $Q'_i(s_i)$.

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

For the query $Q'_k = (x_k = k \cdot c_i + h_k + 2e_k, w_k = [w_{k,j}]_{j \in [n]}, z_k = [z_{k,j}]_{j \in [n]} \in Q'_i(s_i), M_1$ first generates $e_k \leftarrow \mathbb{Z}_q^{1+2\theta} = \{-\theta, \ldots, \theta\}^n$, and then computes

$$v_k \triangleq s \cdot x_k + q_0 w_k + 2\varepsilon_k$$

$$= s \cdot (k c_i + h_k + 2e_k) + q_0 w_k + 2\varepsilon_k$$

$$= k \Delta s_i c_i + \left( k s_i c_i + \sum_{r \in [i-1]} s_i h_{r,c_r} + q_0 w_k \right) + 2(s e_k + \varepsilon_k)$$

14
5.3 The “Undetectable” Attacker \((\mathcal{V}/\mathcal{A}_1)_\delta\) 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 \(x\)-entry. And the \(h\)- and \(e\)-parts were introduced to complicate the algebraic structure of \(x\). Clearly, the more CRT coefficients of \(s\) we get, the more difficult for \(\mathcal{M}_1\) to distinguish those queries made by \(\mathcal{A}_1\) from ordinary ones.

However, \(\mathcal{M}_1\) will still be a problem: if \(\mathcal{A}_1\) seeks to recover the first CRT-coefficient of \(s\) in its first round, \(h = 0\) and hence the \(x\)-entry falls into the set \(\{k \cdot c_i + 2e \mid k \in \mathbb{F}_q, i \in [n], e \in \mathbb{Z}_{q^{1/2}}\}\) \(\subseteq \mathcal{R}_q\), 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 \(s\).

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 \(I \subseteq [n]\), \([s_i]_{i \in I} \in \mathbb{F}_q^I\) 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 \(s_i = \eta_i(s)\), where \(s\) 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 \((I, [s_i]_{i \in I})\) of \(\mathcal{P}_2\), for every query made by \(\mathcal{V}\) to \(\mathcal{M}_1\), the \(x\)-entry is always of the form \(x_0 + 2e\), where \(\text{Dim} (x_0) = I\) and \(e \in \mathbb{Z}_{q^{1/2} + 2\alpha' \sqrt{\pi}}\).

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 \(\delta\) of moderately large, and an index set \(I \subseteq [n]\) of size \(\delta\) randomly. Then, feed \(\mathcal{V}\) with \(q^2\) instances, each of the form \((I, [s_i]_{i \in I})\), \(s_i \in \mathbb{F}_q^I\). In this manner, the CRT-coefficients \(s_i, i \in I\), would be recovered successfully when \([s_i]_{i \in I}\) runs over the set \(\mathbb{F}_q^I\).

In particular, the \(x\)-entry of every query made in this phase is always of the form \(x_0 + 2e\), where \(\text{Dim} (x_0) = I\), and \(e \leftarrow \mathbb{Z}_{q^{1/2} + 2\alpha' \sqrt{\pi}}\). Given the randomness of \(I\), it is hard in practice for \(\mathcal{M}_1\) to identify those queries made by \(\mathcal{V}\).

15
The notation $(V/A_1)_s$ is applied to emphasize the structure of this attacker.

First come some notations. Choose $k \leftarrow [n]$ randomly. Let $\Delta s_i \triangleq s_i - \bar{s}_i$ for every $i \in [n]$. For every $j \in [n]$, define $a_j \triangleq [\bar{s}_j \cdot c_j]_{i \in [n]} \in \mathbb{F}^n_q$, and $b_j \triangleq [\Delta s_i \cdot c_j]_{i \in [n]} \in \mathbb{F}^n_q$; Moreover, define $A_i(u) \triangleq \langle u, a_j \rangle \in \mathbb{F}_q$, and $B_j(u) \triangleq \langle u, b_j \rangle \in \mathbb{F}_q$, where $u \in \mathbb{F}^n_q$. Define the $\mathbb{F}_q$-vector space $U_k \triangleq \{ r \cdot a_k \mid r \in \mathbb{F}_q \} \subseteq \mathbb{F}^n_q$. 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 $U^\perp_k \triangleq \{ v \in \mathbb{F}_q^n \mid A_k(v) = 0 \in \mathbb{F}_q \} \subseteq \mathbb{F}_q^n$. Choose an $\mathbb{F}_q$-basis for $U^\perp_k$ randomly, say $F_k \triangleq \{ f_{k,1}, \ldots, f_{k,n-1} \} \subseteq U^\perp_k$ such that every entry of every $f_{k,i}$ is non-zero. With $U_k$, the set $\mathbb{F}_q^n$ is partitioned into three parts: $S_1 \triangleq \mathbb{F}_q^n \setminus U_k$, $S_2 \triangleq U_k \setminus \{0\}$, and $S_3 \triangleq \{0\}$. Finally, for every $(t,u) \in \mathbb{F}_q \times \mathbb{F}_q^n$, let $\tau(t,u)$ denote $\sum_{i\in [n]} t_{\mu_i}(u) \cdot c_i \in \mathbb{R}_q$.

Some remarks are in order. First, notice that for every $i,j \in [n]$, $s_i, \Delta s_i$, and $b_j$ are unknown to us. Moreover, although the map $A_i(\cdot)$ is efficiently computable, it is not true for $B_j(\cdot)$. Finally, recall that every $c_{i,j} \neq 0$ by Lemma\textsuperscript{5} so the guess $[s_i]_{i \in [n]} = [\bar{s}_i]_{i \in [n]}$ is correct if and only if $b_k = 0 \in \mathbb{F}_q^n$. Since $0 \in S_3 \subseteq U_k$ trivially, a necessary yet insufficient condition for $b_k = 0$ is: $b_k \in U_k = S_2 \cup S_3$, or equivalently, $0 = \langle f_{k,i}, b_k \rangle = B_k(f_{k,i})$ for every $i \in [n-1]$.

The general idea behind $V$ is simple: it first makes the guess, i.e., $[s_i]_{i \in [n]} = [\bar{s}_i]_{i \in [n]}$, and then verifies the correctness of the guess via a set $Q \triangleq Q_1 \cup Q_2$ of queries to $M_1$ such that except with negligible probability, the guess is correct if and only if $M_1$ returns 1 on every query in $Q$.

In more detail, $V$ consists of two successive phases: Phase 1 and Phase 2.

- By issuing a set $Q_1$ of queries to $M_1$, Phase 1 is devoted to deciding whether $b_k \in U_k = S_2 \cup S_3$ or not;
- Conditioned on $b_k \notin U_k$ and hence $b_k = r_0 \cdot a_k$ for some $r_0 \in \mathbb{F}_q$, Phase 2 is to decide whether $r_0 = 0$ or not, by a set $Q_2$ of queries to $M_1$.

It remains to design $Q = Q_1 \cup Q_2$. Jumping ahead, for every query $(x,w,z) \in \mathcal{Q}$, the $x$-entry is always of the form $x_0 + 2e$, where $\dim (x_0) = I = [n]$, and $e \leftarrow \mathbb{Z}_t^{n+2\alpha\sqrt{\pi}}$. Similar to that of $A'_1$, the $e$-part is introduced here to make those queries made by $V$ as “random-looking” as possible.

**Design of Phase 1** Jumping ahead, the query set $Q_1 = Q_1(F_k)$ is

$$Q_1(F_k) \triangleq \{ Q_{t,i} = (\tau(t, f_{k,i}) + 2e_{t,i}, w_{t,i}, z_{t,i}) \mid t \in [g_0], i \in [n-1], e_{t,i} \leftarrow \mathbb{Z}_t^{n+2\alpha\sqrt{\pi}} \}.$$

It remains to set the $w_{t,i}$- and $z_{t,i}$-entries.

Observe that for the query $Q_{t,i} = (\tau(t, f_{k,i}) + 2e_{t,i}, w_{t,i}, z_{t,i}) = (w_{t,i,j})_{j \in [n]}, z_{t,i} = (z_{t,i,j})_{j \in [n]} \in Q_1(F_k)$, $M_1$ first generates $\varepsilon_{t,i} \leftarrow \mathbb{Z}_t^{n+2\alpha}$, and then computes

$$v_{t,i} \triangleq s \cdot (\tau(t, f_{k,i}) + 2e_{t,i}) + g_0 \cdot w_{t,i} + 2\varepsilon_{t,i}$$

$$= s \cdot (\tau(t, f_{k,i}) + g_0 \cdot w_{t,i} + 2(\varepsilon_{t,i} + \varepsilon'_{t,i})) \ (\varepsilon'_{t,i} \triangleq s \cdot e_{t,i} \sim (\varepsilon'_{t,i,j})_{j \in [n]}$$

$$$$

**Phase 2** This phase consists of $n - \delta$ rounds, each devoted to recovering one of the remaining $n - \delta$ CRT-coefficients of $s$, as is done in $A'_1$.

In particular, the $x$-entry of every query made in this phase is always of the form $x_0 + 2e$ where $\dim (x_0) \supseteq I$ and $e \leftarrow \mathbb{Z}_t^{n+2\alpha\sqrt{\pi}}$, making it harder in practice for $M_1$ to identify them.
Design of Phase 2 explains the necessity of Phase 2.

Lemma 11. holds It remains to set the expressed in terms of Proof. When \( q > 1 + 8(\theta + n\alpha\alpha') \), the noise \( (\epsilon_{t, i} + \epsilon'_{t, i}) \) is “short” in the sense that \( \|\epsilon_{t, i} + \epsilon'_{t, i}\|_\infty < \frac{q-1}{n} \) holds w.o.p.

And this definition is justified by the following lemma.

**Lemma 11. Define**

\[
Q_1(F_k) \triangleq \left\{ Q_{t, i} = \left( \tau(t, f_{k,i}) + 2\epsilon_{t,i}, [w_{t, i,j}]_{j\in[n]}, [z_{t, i,j}]_{j\in[n]} \right) \mid t \in [q_0], i \in [n-1], j \in [n], \epsilon_{t,i} \sim Z_{1+2n\alpha\sqrt{n}} \right\}.
\]

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

(a) If \( M_1 \) returns 0 on some queries in \( Q_1(F_k) \), then \( b_k \neq 0 \) and hence \([s_i]_{i\in[n]} \neq \widetilde{[s_i]}_{i\in[n]}\);

(b) If \( M_1 \) returns 1 on every query in \( Q_1(F_k) \), then \( b_k \in U_k = S_2 \cup S_3 \).

**Proof.** First, if \( b_k \in S_3 = \{0\} \), then our guess is correct, every \( b_j = 0 \) and hence every \( B_j(\cdot) = 0 \); By Fact [8], \( M_1 \) returns 1 w.o.p. for every query in \( Q_1(F_k) \).

Moreover, if \( b_k \in S_1 = F_0 \setminus U_k \), then there exists \( f_{k,i} \in F_k \) such that \( B_k(f_{k,i}) \neq 0 \); Moreover, there exists a \( b^* \in [q_0] \) such that \( t^* \cdot B_k(f_{k,i}) = \pm 1 \). By Fact [8], \( M_1 \) returns 0 w.o.p. for at least one query in \( Q_1(F_k) \). \( \square \)

It should be noted that, in Phase 1, it is difficult to analyze the distribution of query replies when \( b_k \in S_2 \), which explains the necessity of Phase 2.

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

Choose \( u^* \sim R_q \) randomly such that \( A_k(u^*) = 1 \) and every entry of \( u^* \) is non-zero. It follows \( t \cdot A_k(u^*) + t \cdot B_k(u^*) = t(1 + r_0) \) for every \( t \in F_q^n \). Jumping ahead, the set \( Q_2 = Q_2(u^*) \)

\[ Q_2(u^*) \triangleq \left\{ Q'_{t,i} = (\tau(t, u^*) + 2\epsilon_{t,i}, w_t, z_t) \mid t \in [q_0], \epsilon_{t,i} \sim Z_{1+2n\alpha\sqrt{n}} \right\}. \]

It remains to set the \( w_t \)- and \( z_t \)-entries.

Observe that for every query \( Q'_{t,i} = (\tau(t, u^*) + 2\epsilon_{t,i}, w_t, z_t) = [w_{t, i,j}]_{j\in[n]}, [z_{t, i,j}]_{j\in[n]} \), \( M_1 \) first generates \( \epsilon_{t,i} \sim Z_{1+2n\alpha} \), and then computes

\[
\epsilon'_{t,i} \sim s \cdot (\tau(t, u^*) + 2\epsilon_{t,i}) + q_0 \cdot w_t + 2\epsilon_t
\]

\[
\sim t \cdot A_1(u^*) + t \cdot B_1(u^*) + 2(\epsilon_{t,i} + \epsilon'_{t,i})
\]

\[
\sim t \cdot A_n(u^*) + t \cdot B_n(u^*) + 2(\epsilon_{t,i} + \epsilon'_{t,i})
\]

Again, when \( q > 1 + 8(\theta + n\alpha\alpha') \), the inequality \( \|\epsilon_{t,i} + \epsilon'_{t,i}\|_\infty < \frac{q-1}{n} \) holds w.o.p. With this in mind, we can define

\[
Q_2(u^*) \triangleq \left\{ Q'_{t,i} = (\tau(t, u^*) + 2\epsilon_{t,i}, [w_{t, i,j}]_{j\in[n]}, [z_{t, i,j}]_{j\in[n]} \mid t \in [q_0], \epsilon_{t,i} \sim Z_{1+2n\alpha\sqrt{n}} \right\}
\]

And the definition is justified by the following lemma.
Lemma 12. With the notations defined previously, if \( q > 1 + 8(\theta + n\alpha\tau) \) and \( b_k \in U_k \) are guaranteed, then except with negligible probability, we have \([s_i]_{i \in [n]} = [\hat{s}_i]_{i \in [n]}\) if and only if \( M_1 \) returns 1 on every query in \( Q_2(u^*)\).

Proof. First, if \([s_i]_{i \in [n]} = [\hat{s}_i]_{i \in [n]}\), then our guess is correct, \( r_0 = 0 \), and every \( B_j(\cdot) = 0 \). By Fact 6(b), \( M_1 \) returns 1 on every query in \( Q_2(u^*)\).

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 \( Q_{r^*} = \tau(t^*, u^*) + 2e_τ, w_τ, z_τ) \in Q_2(u^*) \), its associated \( k \)-th equality does not hold w.o.p. By definition, \( M_1 \) returns 0 w.o.p. on this specific query \( Q_{r^*} \in Q_2(u^*) \).

This finishes the construction of \( V \), as well as its correctness analysis, for the special case when the index set \( I = [n] \). Clearly, it takes at most \( n \cdot q_0 = \text{poly}(\lambda) \) queries for \( V \) to solve this special case of \( P_2 \), indicating that \( V \) runs in polynomial time. See Appendix B.2 for the pseudocode of \( V \).

Moreover, it is easy to generalize the foregoing construction such that \( V \) could be applied to solve the more general case of \( P_2 \), i.e., \( \emptyset \neq I \subseteq [n] \). In general, the number of queries made by \( V \) is upper-bounded by \( q_0 \cdot |I| = q_0 \cdot \text{poly}(\lambda) \).

Theorem 13. Assume \( q > 1 + 8(\theta + n\alpha\tau) \). With the notations defined previously, given \( \emptyset \neq I \subseteq [n] \), \([s_i]_{i \in I}\) and oracle access to \( M_1 \), it takes at most \( q_0 \cdot |I| = q_0 \cdot \delta \) queries for \( V \) to decide whether \([s_i]_{i \in [n]} = [\hat{s}_i]_{i \in [n]}\) or not; except with negligible probability, the equality holds if and only if \( M_1 \) returns 1 on every query in \( Q = Q_1 \cup Q_2 \). In particular, for every query in \( Q \), its \( x \)-entry could be written as \( x = x_0 + 2e \) satisfying \( \text{Dim}(x_0) = I \) and \( e \in \mathbb{Z}^{2n}_1 + 2\alpha \sqrt{\pi} \).

Theorem 14. When \( q > 1 + 8(\theta + n\alpha\tau) \), there exists an efficient attacker \((V,A'_1)\) that can recover the secret of \( M_1 \) w.o.p., after making \( q_0 \cdot q^\delta \cdot (n - \delta) \cdot q \cdot \frac{q-1}{q} = \text{poly}(\lambda) \) queries to \( M_1 \). In particular, for every query made by \((V,A'_1)\) to \( M_1 \), it \( x \)-entry is always of the form \( x = x_0 + 2e \) where \( |\text{Dim}(x_0)| \geq \delta \) and \( e \in \mathbb{Z}^{2n}_1 + 2\alpha \sqrt{\pi} \).

Remarks. To us, the best way for \( M_1 \) to decide whether an incoming query is made by our small field attacker or not is to check the algebraic structure of its \( x \)-entry, i.e., whether \( x \) can be written in the form \( x_0 + 2e \) such that \( \text{Dim}(x_0) \) is “small” and \( e \) 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 \( M_1 \) should decide whether an incoming query is malicious or not in a timely manner; Moreover, such restriction imposed by \( M_1 \) 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 \((V,A'_1)\), the best way is to identify those queries made by \((V,A'_1)\) in its Phase 1. Hence, when \( \delta \) 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 \( M_1 \) to protect its secret \( s \) from our efficient attacker \((V,A'_1)\), even if the foregoing restrictions are imposed by \( M_1 \) on incoming queries.

6 The Small Field Attack Against \( \Pi_1 \)

In this section, we switch our analysis back to the oracle \( M_0 \) (defined in Section 4.2), and show how to construct an efficient attacker \( A_0 \) which can recover the static private key of \( M_0 \) (defined in Section 4.2), after a set of queries made to \( M_0 \); Moreover, both the static public key of \( A_0 \) and those queries it makes to \( M_0 \) are as “random-looking” as possible. The existence of \( A_0 \) implies \( \Pi_1 \) is vulnerable to our small field attack.

Construction of \( A_0 \). As \( p = as + 2e \in R_q \) is made public, to recover the private key \((s,e)\) of \( M_0 \), it suffices for \( A_0 \) to recover \( s \in R_q \). Moreover, it suffices for \( A_0 \) to recover every CRT-coefficient \( s_i = \eta_i(s) \) of \( s \in R_q \). Recall that the “undetectable” and efficient attacker \((V,A'_1)\) against \( M_1 \) constructed in Section 5.3 implies we can already construct an efficient attacker against \( M_0 \) with static public key \( 0 \in R_q \), whose queries made to \( M_0 \) are as “random-looking” as possible. Thus, the difficulty of designing \( A_0 \) lies in how to design its static public key \( p^* \) that looks “random”. It turns out the construction of \( V \) already indicates that how to choose a “random-looking” \( p^* \) for \( A_0 \).

The general structure of \( A_0 \) is defined as follows. \( A_0 \) consists of three consecutive phases: Phase 0, Phase 1, and Phase 2. In Phase 0, \( A_0 \) generates static its public/private key pair as follows: first, it chooses a positive integer \( \delta \) that is moderately large, say \( \delta \); Then, it chooses a proper index set \( I \subseteq [n] \) of size \( \delta \) uniformly at random; After that, it samples \( p_0^* \leftarrow \{u \in R_q \mid \text{Dim}(u) = I \} \) and \( e_0^* \leftarrow \mathbb{Z}^{2n}_1 + 2\alpha \sqrt{\pi} \) uniformly at random; The static public
key of $A_0$ is $p^* \triangleq p_0^* + 2e_0^*$, and the associated static private key of $A_0$ is $(s^*, e^*)$, where $s^* \leftarrow \mathcal{R}_q$ is drawn randomly and $e^* \triangleq 2^{-s}(p^* - as^*)$. It should be stressed that the static public/private key pair of $A_0$ is not honestly generated.

The Phase 1 and Phase 2 of $A_0$ are devoted to the recovery of every $s_i = \eta_i(s)$, $i \in [n]$, and their functionalities are similar to those of $(\mathcal{V}/A'_i)_g$, respectively: Phase 1 is devoted to recovering $\delta$ CRT-coefficients of $s$, i.e., $\{s_i, i \in I\}$ where $I \subseteq [n]$ is of size $\delta$, and Phase 2 is to recover the others ones.

The Phase 1 of $A_0$ is very close to the design of $(\mathcal{V}/A'_i)_g$ 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 $A_0$ is fully described in the sequel. For the moment, we assume that $I \subseteq [i-1]$ and the CRT-coefficients $s_1, \ldots, s_{i-1} \in \mathbb{F}_q$ have already been recovered successfully, and we are about to see how $A_0$ recovers a new CRT-coefficient of $s$, say $s_i \in \mathbb{F}_q$, via a set of queries to $\mathcal{M}_0$. The general strategy is simple: first pick $\tilde{s}_i \in \mathbb{F}_q$ randomly, and guess $s_i = \tilde{s}_i$; Then conduct a set $Q_i(\tilde{s}_i)$ 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 $Q_i(\tilde{s}_i)$; 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 $Q_i(\tilde{s}_i)$ is of the form $(\tilde{d}^*, p^*, x_k = kc_i + h_k + 2e_k, w_k, z_k)$, where $k \in [q_0]$, $h_k \leftarrow \{u \in \mathcal{R}_q \mid \text{Dim}(u) = [i-1]\}$, $e_k \leftarrow \mathbb{Z}_{1+2n^\alpha \sqrt{n}}$, and $w_k, z_k \in \mathbb{B}^n$ are to be determined later.

Recall that, on the query $(\tilde{d}, p, x_k = kc_i + h_k + 2e_k, w_k, z_k)$ where $h_k = \sum_{r \leq i-1} h_{k,r}c_r, h_{k,r} \leftarrow \mathbb{F}_q^*$, the oracle $\mathcal{M}_0$ first samples $g \leftarrow D_{\mathbb{Z}_n, \alpha}$, then compute $c \leftarrow H_1(\tilde{d}, \tilde{d}^*, x_k)$ and

$$v_k \triangleq (p^*c + x_k)s + q_0 \cdot w_k + 2e \cdot g$$

$$= k \cdot s \cdot c + p_0^*c + s \cdot h_k + q_0 \cdot w_k + 2(e_k s + e_0^* cs + eg)$$

$$= k \Delta s_i c_i + (k\tilde{s}_i c_i + p_0^* c + s \cdot h_k + q_0 \cdot w_k) + 2e_k (e_k \triangleq e_k s + e_0^* cs + eg \sim [e_{k,j}]_{j \in [n]})$$

$$= \begin{bmatrix} \Delta s_i \cdot k c_{i,1} + (k\tilde{s}_i c_{i,1} + \sum_{r \leq i-1} \eta_r(p_0^* c_s) c_{r,1} + \sum_{r \leq i-1} s_i h_{k,r} c_{r,1} + q_0 w_{k,1}) \vline \vdots \vline \sum_{r \leq i-1} s_i h_{k,r} c_{r,n} + q_0 w_{k,n} \end{bmatrix} + 2^\varepsilon \begin{bmatrix} \varepsilon_{k,1} \vline \vdots \vline \varepsilon_{k,n} \end{bmatrix},$$

where $\Delta s_i \triangleq s_i - \tilde{s}_i$; Finally, $\mathcal{M}_0$ computes $\sigma_k := \text{Parity}(v_k)$, and returns 1 if and only if $\sigma_k = z_k$. Notice that $\text{Dim}(p_0^*cs) \subseteq \text{Dim}(p_0^*) = I$, making it possible for $A_0$ to pre-compute every $\eta_r(p_0^*cs), r \in I$, 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 $\mathbb{F}_q^*$, 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 $s_1, \ldots, s_{i-1}$ have been given and $q > 1 + 8(n \alpha \gamma + n^2 \alpha^2 \gamma)$, then except with negligible probability, $\Delta_{s_i} = 0$ if and only if $\mathcal{M}_0$ returns 1 on every query in

$$Q_i(\tilde{s}_i) = \left\{ \left( \tilde{d}^*, p^*, kc_i + h_k + 2e_k, (w_{k,j})_{j \in [n], [z_{k,j}]_{j \in [n]}} \right) \mid \begin{array}{c} k \in S_n, j \in [n], h_{k,1}, \ldots, h_{k,i-1} \leftarrow \mathbb{F}_q^*, h_k = \sum_{r \leq i-1} h_{k,r} c_r, e_k \leftarrow \mathbb{Z}_{1+2n^\alpha \sqrt{n}}, u_{k,j} = k\tilde{s}_i c_{i,j} + \sum_{r \leq i-1} \eta_r(p_0^* cs) c_{r,j} + \sum_{r \leq i-1} s_i h_{k,r} c_{r,j}; w_{k,j} = \text{Chg}(u_{k,j}), z_{k,j} = \text{Mod}(u_{k,j}, q) \end{array} \right\}.$$  

In particular, for every query in $Q_i(\tilde{s}_i)$, its $x$-entry is always of the form $x_0 + 2e$, where $\text{Dim}(x_0) = [i]$ and $e \in \mathbb{Z}_{1+2n^\alpha \sqrt{n}}^n$. 

The correctness and the efficiency of $A_0$ can be easily verified. In sum, the efficient attacker $A_0$ implies the existence of the desired efficient attacker against the honest party $j$ in $\Pi_1$, as the following theorem indicates.

Theorem 16. With the notations defined previously, when $q > 1 + 8(n \alpha \gamma + n^2 \alpha^2 \gamma)$, there exists an efficient adversary $A$ that can recover the private static key of the honest party $j$ in $\Pi_1$ w.o.p. In particular, for every query made by $A$, its $x$-entry is always of the form $x_0 + 2e$, where $\text{Dim}(x_0) \geq \delta$ and $e \in \mathbb{Z}_{1+2n^\alpha \sqrt{n}}^n$. 

In [ZDS14, ZD+15], four groups of suggested parameters for $\Pi_1$ 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 \((V/A'_1)\), the efficient adversary \(A\) against \(\Pi_1\) can make its session queries to the honest party \(j\) as random-looking as possible by choosing an appropriate \(\delta\), making it hard in practice for party \(j\) to identify (and hence reject) those session queries made by \(A\).

7 Analysis on the Two-pass AKE Protocol \(\Pi_2\)

In \(\Pi_2\), the static private key of party \(i\) is \((s_i \leftarrow D_{Z^n}, \alpha e_i \leftarrow D_{Z^n})\), and the associate static public key is \(p_i = \alpha s_i + 2e_i \in \mathcal{R}_q\), where \(\alpha \in \mathcal{R}_q\) denotes a global parameter of \(\Pi_2\) that is drawn from the uniform distribution over \(\mathcal{R}_q\). Similar notations, \((s_j, e_j), p_j\), carry over to party \(j\) in \(\Pi_2\).

Here is a brief review of the two-pass AKE scheme \(\Pi_2\) proposed in [ZZDS14, ZZD+15].

**Initialization** First, party \(i\) acts as follows:
1. Pick \(r_i, f_i \leftarrow D_{Z^n}, \beta\), and compute \(x_i = \alpha r_i + 2f_i\).
2. Compute \(c \leftarrow H_1(\hat{x}_i, i, x_i)\), \(r_i = s_i c + r_i\), and \(f_i = e_i c + f_i\).
3. Let \(z_i \in Z^{2n}\) be the coefficient vector of \(r_i\) concatenated with that of \(f_i\), and \(z_1\) be the coefficient vector of \(s_i c\) concatenated with that of \(e_i c\).
4. Send \(x_i\) to party \(j\).

**Response** On message \(x_i\), party \(j\) works as follows:
1. Pick \(r_j, f_j \leftarrow D_{Z^n}, \beta\), and compute \(y_j = \alpha r_j + 2f_j\).
2. Compute \(d \leftarrow H_1(\hat{y}_j, i, y_j, x_i)\), \(r_j = s_j d + r_j\), and \(f_j = e_j d + f_j\).
3. Let \(z_j \in Z^{2n}\) be the coefficient vector of \(r_j\) concatenated with that of \(f_j\), and \(z_1\) be the coefficient vector of \(s_j d\) concatenated with that of \(e_j d\).
4. Send \(y_j\) to party \(i\).

**Finish** On message \((y_j, w_j)\), party \(i\) proceeds as follows:
1. Pick \(g_i \leftarrow D_{Z^n}, \beta\), and compute \(k_i = (p_i c + x_i) \cdot r_j + 2e_j g_i\).
2. Let \(w_j \leftarrow \text{Cha}(k_i)\), and send \((y_j, w_j)\) to party \(i\).

**Claim 16** and the Underlying Games \(G_{2,4}, G_{2,5}\)

In the security proof of \(\Pi_2\) [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 \((\Pi_2, I, i.d., i.d., i.d., x_i, (y_j, w_j))\), in which the adversary \(A\) impersonates the honest user \(j^*\) but \(y_j\) is not sent by \(j^*\) upon receiving \(x_i\) 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 \(\text{sid}^* \triangleq (\Pi_2, I, i.d., i.d., i.d., x_i, (y_j, w_j))\) the test session, where \(x_i\) is output by honest party \(i^*\) with intended honest party \(j^*\). In \(G_{2,4}\), the simulator \(S\) maintains two tables \(L_1, L_2\) for the random oracles \(H_1(\cdot), H_2(\cdot)\), respectively. Now, \(S\) chooses \(u_0, u_1 \leftarrow \mathcal{R}_q\) and sets the global parameter \(\alpha\) to be \(u_0\), and then sets the static public key of party \(j^*\) to be \(u_1\), i.e., \(\alpha := u_0, p_{j^*} := u_1\). Moreover, \(S\) prepares the table

\[
T \triangleq \left\{ (\hat{r}_k, \hat{f}_k, g_k, v_{0,k}, v_{1,k}) \mid 1 \leq k \leq \ell, \hat{r}_k, \hat{f}_k, g_k \leftarrow D_{Z^n, \beta}, v_{0,k} = u_0 \hat{r}_k + 2\hat{f}_k, v_{1,k} = u_1 \hat{r}_k + 2g_k \right\}.
\]

Finally, for the test session, \(S\) answers oracle queries made by the efficient adversary \(A\) as follows:

- Upon receiving Send_4(\(\Pi_2, I, i.d., i.d., i.d.)\) w.r.t. the test session, \(S\) proceeds as follows:
  1. Sample an invertible element \(e^* \leftarrow D_{Z^n, \gamma}\), and choose the first unused tuple in \(T\), say the \(\ell^*\)-th tuple, and set \(\hat{x}_{i^*} := v_{0,\ell^*}\).
  2. Define \(x_{i^*} := \hat{x}_{i^*} - p_{i^*} e^*\).
  3. Repeat steps 1-2 with probability \(1 - 1/M\), where \(M\) is a sufficiently large positive integer.
Moreover, the only difference between $G_A$ and $PPT$ adversary $R$ is that, in $G_A$, the use of forking lemma is actually unnecessary for proving Claim 16, and thus the reasoning related to forking lemma can be removed.

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.

Note that, in $G_A$, when $S$ answers the Send$_d$ oracle query for the test session, the value $\hat{x}_i := v_{0,i}$ follows the uniform distribution over $\mathcal{R}_q$ according to the table $T'$ set by $S$.

### 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 $A = (A_1, A_2)$, which could be divided into two consecutive stages $A_1$ and $A_2$. $A_1$ denotes the actions of $A$ until it just gets the RO-answer $d' \leftarrow H_1(\text{id}_i, \text{id}_j, y_j, x_i)$. Let $v_{0,i} \triangleq (x_i, y_j, d', p_i, p_j, c_i, a, t, r, s)$ be the output of $A_1$ (to $A_2$), where $t, r, s$ denotes the views of $A_1$ in other sessions other than the test session, and $st$ denotes some state information. On input of $v_{0,i}, A_2$ performs the remaining actions of $A$.

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 $A_1$ in $G_A$ is computationally indistinguishable from that of $A_1$ in $G_{2,5}$.
- In the RO model, in order for $A_2$ to succeed, it has to make the RO-query $H_2(\text{id}_i, \text{id}_j, y_j, x_i)$. Let $v_{0,i} \triangleq (x_i, y_j, d', p_i, p_j, c_i, a, t, r, s)$ be the output of $A_1$ (to $A_2$), where $t, r, s$ denotes the views of $A_1$ in other sessions other than the test session, and $st$ denotes some state information. On input of $v_{0,i}, A_2$ performs the remaining actions of $A$.

In the proof of Claim 16 [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 $A_2$ outputs $\sigma_i := \text{Mod}(k_i, w_j)$ 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 $(\text{view}_1, k_i)$ in $G_{2,4}$ and that in $G_{2,5}$ are computationally indistinguishable, because the event in question is defined over both $\text{view}_1$ and $k_i$. In particular, we need to at least argue the computational indistinguishability between $k_i$ defined over $G_{2,4}$ and the uniform distribution over $\mathcal{R}_q$ (that is just the distribution of $k_i$ 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 $k_i^*$ and $x_i^*$ are independent. However this is not the case in game $G_{2.4}$, since $x_i^* = \alpha \hat{r}_{i^*} + 2 \tilde{f}_{i^*} - p_i, c^*$ and $k_i^* = d^*(p_j, \hat{r}_{j^*} + 2y_{i^*}) + y_{i^*}, \hat{r}_{i^*}$ are related by $\hat{r}_{i^*}$, making them dependent. In other words, in game $G_{2.4}$, when $x_i^*$ is given to the efficient adversary $A$, it might be possible for $A$ to extract some information regarding $k_i^*$.

Secondly, given that in $G_{2.4}$, we have $x_i^* = \alpha \hat{r}_{i^*} + 2 \tilde{f}_{i^*} - p_i, c^*$ and $k_i^* := d^*v_{1,i^*} + y_j, \hat{r}_{i^*} = (p_j, d^* + y_j, \hat{r}_{i^*})$, the values $(x_i^*, y_j, d^*, p_j, p_i, c^*, a)$ essentially determine the value $k_i^* - 2d^* g_{i^*}$. In view that $2d^* g_{i^*}$ is small, the value $k_i^*$ is essentially committed to $(x_i^*, y_j, d^*, p_j, p_i, c^*, a)$ up to a “small” and even noise.

Finally, notice that in $G_{2.4}$, for the equation $k_i^* = d^*v_{1,i^*} + y_j, \hat{r}_{i^*}$, the two terms on the right-hand side are not independent either. Thus, it is not that easy to analyze the exact distribution of $k_i^*$.

These highlighted differences indicate that the above facts by consensus are insufficient to establish the indistinguishability between $k_i^*$ defined over $G_{2.4}$ and the uniform distribution over $R_q$. Actually, we even do not know how to formally prove, with a reducibility argument, a seemingly easier goal: given $x_i^* = \alpha \hat{r}_{i^*} + 2 \tilde{f}_{i^*} - p_i, c^*$, it is infeasible for any efficient $A_2$ to successfully recover $h_i^* \equiv (p_j, d^* + y_j, \hat{r}_{i^*})$ by maliciously setting $y_j$ and $w_j$, (recall that $k_i^* = h_i^* + 2d^* g_{i^*}$, where $2d^* g_{i^*}$ is even and “small”). Notice that, conditioned on $A_2$ recovers $h_i^*$, by setting $w_j := \text{Parity}(h_i^*)$ it can then determine the value of $\sigma_i^*$ and thus break the security, since the following holds w.r.o.p.:

\[
\sigma_i^* = \text{Mod}(k_i^*, w_j^*), \quad \text{Mod}(h_i^* + 2d^* g_{i^*}, \text{Parity}(h_i^*)) = \text{Mod}(h_i^*, \text{Parity}(h_i^*)).
\]

The above clarifications also indicate a fundamental difference between the security proof of HMQV [Kra05] and that of its RLWE-based analogue [ZZD15]. In the security proof of HMQV [Kra05], the simulator can directly obtain, from RO-query, the key material (corresponding to $k_i^*$ in [ZZD15]) in order to reach reduction contradiction to the underlying gap Diffie-Hellman assumption. However, in the security proof of RLWE-based HMQV-analogue [ZZD15], the simulator gets, also from RO-query, only the value $\sigma_i^* := \text{Mod}(k_i^*, w_j^*)$, where the key material $k_i^*$ is hidden and $w_j^*$ may be maliciously set only in the stage of $A_2$. 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 $A = (A_1, A_2)$. 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, $A$ gets access to a random oracle $H : \{0,1\}^\lambda \rightarrow \{0,1\}^\lambda$ that is maintained and simulated by a PPT simulator $S$, where $\lambda$ is the security parameter. Denote by $\text{Com}$ 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 $H' : \{0,1\}^{2\lambda} \rightarrow \{0,1\}^\lambda$ as follows. Before the game starts, it sets $\text{pre} = 0^\lambda$ (resp., $\text{pre} \leftarrow \{0,1\}^\lambda$, i.e., $\text{pre}$ 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 $y = H'(\text{pre}||x)$. Notice that, from the view of $A$, the RO simulation of $S$ is perfect. At the end of $A_1$, $S$ computes $C = \text{Com}(\text{pre}^*)$ (resp., $C = \text{Com}(\text{pre})$, where $\text{pre}^* \leftarrow \{0,1\}^\lambda$ is independent of $\text{pre}$), and gives $C$ to $A_1$. Define $\text{view}_1 = (C, \text{tr}, s)$ the output of $A_1$, on which $A_2$ proceeds further. Finally, $A_2$ just simply outputs $\text{pre}^*$.

For the above simplified analogous games, we have: (1) The output of $A_1$ in $G_0$ is computationally indistinguishable from that in $G_1$ [2]. The actions of $A_2$ make no contribution to its ability of computing $\text{pre}$. (2) Clearly, in $G_2$, the probability that $A_2$ correctly outputs $\text{pre}$ with probability of just $2^{-\lambda}$ in the RO model, as $\text{pre}$ is a random value actually independent of $A_1$’s view in this case. However, we couldn’t reach the conclusion that $A_2$ will also output $\text{pre}$ with negligible probability in $G_0$. Actually, the success probability of $A_2$ in $G_0$ is 1. The reason is just that the success event of $A_2$ is defined not only over its view but also over the “hidden” value $\text{pre}$, which is $0^\lambda$ in $G_0$ clearly distinguishable from a random value in $G_2$.

**Acknowledgement** We thank some authors of [ZZD15, 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**


\[4\] Note that the random oracle $H : \{0,1\}^\lambda \rightarrow \{0,1\}^\lambda$ can be perfectly simulated without using $H'$ or the knowledge of $\text{pre}$.

Also, if $C$ is not given to $A_1$, $A$‘s view in $G_0$ and that in $G_1$ are identical.
We have made deep discussions with the corresponding authors of \([ZZD^{+}15, ZZDS14]\) in regard to the security proof of \(\Pi_2\). 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 \(\Pi_1\) 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 \(\Pi_2\). 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 $A'_1$ and $V$ in Pseudocode

B.1  The Algorithm $A'_1$

Algorithm 1 The algorithm $A'_1$

1: procedure $(A'_1)^{M_1}(i, \{s_1, \cdots, s_{i-1}\})$
2:   Choose $g \in \mathbb{F}_q^*$ such that $g$ is a primitive element of $\mathbb{F}_q$;
3:   $d := q_0/n$;
4:   for every $\tilde{s}_i \in \mathbb{F}_q$ do
5:      for every $k \in \{g^r | r \in [d]\}$ do
6:         for every $r \in [i-1]$ do
7:            $h_{k,r} \leftarrow F_q$; $\triangleright$ Generate nonzero $h_{k,r} \in \mathbb{F}_q$ randomly
8:         $e_k \leftarrow Z^{d+2n'}_{1+\sqrt{n}}$; $\triangleright$ Generate the random $e_k$
9:      for every $j \in [n]$ do
10:         $u_{k,j} := \tilde{s}_i \cdot k c_{i,j} + \sum_{r \in [i-1]} s_r h_{k,r} r_{c,j}$;
11:         $w_{k,j} := \text{Cha}(u_{k,j});$ $\triangleright$ Generate the small noise $e_k$
12:      $z_{k,j} := \text{Mod}(u_{k,j}, w_{k,j});$ $\triangleright$ Set the desired $w$-entry
13:      $b_k \leftarrow M_1 (k \cdot c_i + h_k + 2 e_k, [w_{k,j}]_{j \in [n]}, [z_{k,j}]_{j \in [n]});$ $\triangleright$ Invoke $M_1$
14:   end for
15: end for
16: if $b_k = 1$ for every $k \in \{g^r | r \in [d]\}$ then
17:   return $\tilde{s}_i$; $\triangleright$ This implies the guess $s_i = \tilde{s}_i$ is correct
18: end if
19: return $\bot$;
20: end procedure
B.2 The Algorithm $\mathcal{V}$

Algorithm 2 Verification of $(s_1, \ldots, s_n) \equiv (\tilde{s}_1, \ldots, \tilde{s}_n)$

1: procedure $\mathcal{V}(\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_n)$
2: $J := \emptyset$;
3: repeat
4: Choose $k \in [n]$ such that $\{i \in [n] | c_{i,k} \neq 0\} \not\subseteq J$;
5: Choose an $F_q$-basis for $U_k^r$ randomly, say $F_k = \{f_{k,1}, \ldots, f_{k,n-1}\} \subseteq \mathbb{F}_q^n$;
6: for every $y \in [q_0]$ do
7: for every $u \in F_k$ do
8: $u_r := y \cdot A_r(u)$;
9: $\sigma_r := \text{Cha}(u_r)$; $z_r := \text{Mod}(u_r, \sigma_r)$;
10: end for
11: $b_1(y, u) \leftarrow \mathcal{M}_1(\tau(y, u), [\sigma_r]_{r \in [n]}, [z_r]_{r \in [n]})$;
12: if $b_1(y, u) \neq 1$ then
13: return 0; \quad \triangleright \text{ $b^{(k)} \not\subseteq U^{(k)}$, and hence $(s_1, \ldots, s_n) \neq (\tilde{s}_1, \ldots, \tilde{s}_n)$}
14: end if
15: end for
16: end for
17: Choose $u^* \in \mathbb{F}_q^n$ randomly such that $A_k(u^*) = 1$;
18: for every $y \in [q_0]$ do
19: for every $r \in [n]$ do
20: $u_r := y \cdot A_r(u^*)$;
21: $\sigma_r := \text{Cha}(u_r)$;
22: $z_r := \text{Mod}(u_r, \sigma_r)$;
23: end for
24: $b_2(y) \leftarrow \mathcal{M}_1(\tau(y, u^*), [\sigma_r]_{r \in [n]}, [z_r]_{r \in [n]})$;
25: end for
26: if $b_2(y) = 0$ for some $y \in [q_0]$ then
27: return 0; \quad \triangleright (s_1, \ldots, s_n) \neq (\tilde{s}_1, \ldots, \tilde{s}_n)$
28: end if
29: $J := J \cup \{i \in [n] | c_{i,k} \neq 0\}$;
30: until $J = [n]$
31: return 1; \quad \triangleright (s_1, \ldots, s_n) = (\tilde{s}_1, \ldots, \tilde{s}_n)$
32: end procedure