Key Exchange and Authenticated Key Exchange with Reusable Keys Based on RLWE Assumption

Jintai Ding\textsuperscript{1}, Pedro Branco\textsuperscript{*2}, and Kevin Schmitt\textsuperscript{1}

\textsuperscript{1}University of Cincinnati
\textsuperscript{2}SQIG-IT / IST-Universidade de Lisboa

Abstract

Key Exchange (KE) is, undoubtedly, one of the most used cryptographic primitives in practice. Its authenticated version, Authenticated Key Exchange (AKE), avoids man-in-the-middle-based attacks by providing authentication for both parties involved. It is widely used on the Internet, in protocols such as TLS or SSH. In this work, we provide new constructions for KE and AKE based on ideal lattices in the Random Oracle Model (ROM). The contributions of this work can be summarized as follows:

• It is well-known that RLWE-based KE protocols are not robust for key reuses since the signal function leaks information about the secret key. We modify the design of previous RLWE-based KE schemes to allow key reuse in the ROM. Our construction makes use of a new technique called \textit{pasteurization} which enforces a supposedly RLWE sample sent by the other party to be indeed indistinguishable from a uniform sample and, therefore, ensures no information leakage in the whole KE process.

• We build a new AKE scheme based on the construction above. The scheme provides implicit authentication (that is, it does not require the use of any other authentication mechanism, like a signature scheme) and it is proven secure in the Bellare-Rogaway model with weak Perfect Forward Secrecy in the ROM. It improves previous designs for AKE schemes based on lattices in several aspects. Our construction just requires sampling from only one discrete Gaussian distribution and avoids rejection sampling and noise flooding techniques, unlike previous proposals (Zhang \textit{et al.}, EUROCRYPT 2015). Thus, the scheme is much more efficient than previous constructions in terms of computational and communication complexity.

Since our constructions are provably secure assuming the hardness of the RLWE problem, they are considered to be robust against quantum adversaries and, thus, suitable for post-quantum applications.

\textsuperscript{*}Email: pmbranco@math.tecnico.ulisboa.pt
1 Introduction

Key Exchange (KE) is a cryptographic primitive that allows two parties to agree on a shared key while a third party eavesdropping the communication gets no information about the shared key. Subsequently, the shared key can be used to securely communicate or provide authentication. This cryptographic primitive was presented in the seminal paper by Diffie and Hellman [22], which marked the birth of modern cryptography, and is, undoubtedly, one of the most used primitives in both theoretical and real applications.

However, the standard KE primitive is not robust to man-in-the-middle (or impersonation) attacks: an adversary controlling the network can easily modify the communication between two parties, making them believe that they are privately communicating with each other while, in fact, the conversation is being controlled by the adversary. Authenticated Key Exchange (AKE) is a flavor of KE which provides authentication to the parties involved and, thus, it avoids man-in-the-middle attacks. Due to its robustness to this type of attacks, AKE is used in a wide range of applications such as SSL [32] and TLS [21].

Authentication for KE can be achieved explicitly (that is, by explicitly using other primitives that provide authentication, like signature schemes) or implicitly (that is, without requiring the explicit use of other primitives). The idea of using implicit authentication for KE was presented in [48] and has been intensively studied since then. In these protocols, a key is shared by means of static and ephemeral keys belonging to the parties involved in the protocol; authentication is guaranteed by the static key while the ephemeral key usually provides Perfect Forward Secrecy (PFS) [1]. The efficiency in terms of both communication and computational complexity achieved by this kind of protocols, as well as its simplicity and elegance, has led to their massive standardization by institutions all over the world. Examples of such schemes are the MQV-like protocols (that is, MQV [49, 43], HMQV [41] and OAKE [55]), which have been extensively used as standard [38], and NAXOS [42].

All the above schemes are based on the Diffie-Hellman protocol, hence, their security is based on the discrete logarithm assumption. With the possible advent of a large-scale quantum computer, these protocols will become obsolete [54]. This fact has led the National Institute of Standards and Technology (NIST) to announce a call to define the next post-quantum standard protocols for KE and digital signatures to be used by national institutions in the USA [1]. Hence, the development of post-quantum KE schemes and its variants (such as AKE schemes) is of high priority and we should expect these types of protocols to be used in the near future.

1.1 Lattice-based Key Exchange

One of the first post-quantum KE was presented in [28], where a KE protocol based on the Ring Learning with Errors (RLWE) assumption [46] was pre-

Recall that PFS guarantees that in case a static secret key is revealed to an adversary, previously established session keys are not compromised.
sented. The scheme of [28] is also our starting point. Hence, to ease the presentation of our results, we recall the scheme of [28] which we refer to as Ding’s KE. Let \( R_q = \mathbb{Z}_q[X]/(X^n + 1) \), for some prime \( q \), and \( \chi_\alpha \) be a discrete Gaussian distribution. Let \( P_i \) and \( P_j \) be two parties that want to exchange a key. User \( P_i \) (resp. \( P_j \)) has a secret key \( s_i \) (resp. \( s_j \)) sampled from \( \chi_\alpha \) and the public key is a RLWE sample of the form \( as_i + 2e_i \) (resp. \( as_j + 2e_j \)) where \( e_i \) (resp. \( e_j \)) is an error vector sampled from \( \chi_\alpha \). Party \( P_i \) starts by sending \( x_i = as_i + 2e_i \). Party \( P_j \) computes \( y_j = as_j + 2e_j \) and \( k_j = x_is_j + 2g_j \) where \( g_j \) is sampled from \( \chi_\alpha \). Now, party \( P_j \) computes the signal \( w_j \) of \( k_j \), using a function \( \text{Sig} \) that just tells if each coefficient of \( k_j \) is within an interval or not. Party \( P_j \) sends \( y_j \) and the signal \( w_j \) of \( k_j \) to \( P_i \). Both parties can now agree on a shared key using an extractor function \( \text{Mod}_2 \).

Recent results have noticed that information is leaked by the signal function, in RLWE-based KE protocols [31, 23, 25]. In particular, if the value \( x_i \) sent by party \( P_i \) is not computed honestly (that is, if it is not an RLWE sample), then party \( P_i \) can recover information about party \( P_j \)’s secret key. This can be done by noticing the behavior of the signal sent by \( P_j \) after several executions of the scheme. Hence, one cannot reuse the same keys in several executions of the protocol, risking itself of having its secret key exposed to someone else.

After the introduction of Ding’s KE [28], several other lattice-based KE [51, 56, 14, 5] were proposed. However, most of these schemes do not provide authentication by themselves. So, authentication is guaranteed by means of an explicit mechanism (such as signature schemes). As far as we are aware, the only lattice-based AKE that provides implicit authentication is the scheme of [56]. Unfortunately, the use of techniques such as rejection sampling and noise flooding may raise implementation issues [30]. They also turn the parameters of the scheme large and it is required the use of more than one discrete Gaussian distribution.

Another different approach to exchange keys is to use Key Encapsulation Mechanism (KEM), for which several lattice-based proposals have been made in recent years [15, 33, 34, 13]. However, KEMs require the use of a decryption algorithm, which is usually more computationally expensive than using a KE protocol. Also, KEMs usually do not provide PFS, that is, all the previously established session secret keys are compromised in case the secret key of a user is exposed. Recall that, in a nutshell, a KEM is a Public-Key Encryption (PKE) used as a KE; hence, once the secret key of a party is revealed, every message that was encrypted using the corresponding public key is also revealed (in the case of a KEM, this corresponds to established session keys).

1.2 Contributions and techniques

In this work, we present post-quantum solutions for KE and AKE that allow for key reuse. We base the security of our schemes in the RLWE assumption [46].

Another prominent line of research in post-quantum Key Exchange adopts the supersingular isogeny-based approach [39].
a well-established assumption in cryptography that enjoys an average-case reduction from worst-case lattice problems. Schemes based on this assumption usually provide post-quantum security and are asymptotically more efficient than their discrete log-based counterpart.

1.2.1 Key Exchange with reusable keys

First, we remark that if \( P_i \)'s message is an RLWE sample, then the value \( k_j \) computed by \( P_j \) is indistinguishable from a uniformly chosen value and, thus, the signal of \( k_j \) is also indistinguishable from a uniformly chosen value. Hence, we just have to force each party to behave honestly in the protocol.

We use a technique, which we call pasteurization, to force the parties involved in the KE scheme to behave honestly. The technique was previously introduced in [26] in the context of zero-knowledge proofs. The idea of this technique is the following: after receiving \( x_i \) from \( P_i \), the party \( P_j \) pasteurizes \( x_i \), i.e., it computes

\[
\bar{x}_i = x_i + aH(x_i) + 2f_j,
\]

where \( H \) is a random oracle whose outputs are sampled from \( \chi_\alpha \) and \( f_j \) is sampled from \( \chi_\alpha \). If \( x_i \) is indeed an RLWE sample, then the pasteurization \( \bar{x}_i \) is also an RLWE sample, for which \( P_i \) knows the secret. However, when \( x_i \) is not an RLWE sample, then \( \bar{x}_i \) looks pseudorandom to \( P_i \). Thus, the signal of \( k_j \) is also pseudorandom and \( P_i \) cannot extract information about \( P_j \)'s secret key from it. We conclude that party \( P_i \) gains nothing by not following the protocol.

To guarantee that party \( P_i \) can also reuse its key in several executions of the protocol, we make it pasteurize \( y_j \) sent from \( P_j \). A scheme of the protocol is presented in Figure 1.

![Figure 1: Ding’s Ke with reusable keys](image)

\( \chi_\alpha \) is a discrete Gaussian distribution over \( \mathbb{R}_q \) with standard deviation \( \alpha \), \( H_1 : \{0,1\}^* \to \chi_\alpha \) is a hash function whose outputs are sampled from \( \chi_\alpha \) and \( \text{Sig} \) and \( \text{Mod}_2 \) are the signal and the extraction functions (respectively), as defined in [28].
In the Diffie-Hellman KE \cite{22}, exchanged messages from both parties are supposed to be in a group $\mathbb{G}$. We avoid possible attacks by making the parties verify if all the exchanged values are in $\mathbb{G}$, which can be done in polynomial time. However, in the RLWE-based KE, since the exchange messages are RLWE samples, it is impossible to straightforwardly check if they are honestly computed. The pasteurization technique can be seen as the analog of checking if the exchanged messages are in $\mathbb{G}$ in the Diffie-Hellman KE, since the technique also enforces good behavior by the parties involved.

1.2.2 New Authenticated Key Exchange scheme

Our major contribution is the design of a new AKE scheme based on the RLWE assumption. At the heart of our construction is the RLWE-based KE described above. The scheme can be found in Figure \cite{22}.

![Figure 2: The new AKE protocol: $\chi_\alpha$ is a discrete Gaussian distribution over $R_q$ with standard deviation $\alpha$, $H_1 : \{0,1\}^* \rightarrow \chi_\alpha$ is a hash function whose outputs are sampled from $\chi_\alpha$, $H_2 : \{0,1\}^* \rightarrow \{0,1\}^\kappa$ is a $\kappa$-bit key derivation function and $\text{Sig}$ and $\text{Mod}_2$ are the signal and the extraction functions (respectively), as defined in \cite{28}.

Protocol idea. Let $(\text{spk}_i, \text{ssk}_i)$ (resp. $(\text{spk}_j, \text{ssk}_j)$) and $(\text{epk}_i, \text{esk}_i)$ (resp. $(\text{epk}_j, \text{esk}_j)$) be pairs of static and ephemeral public and secret keys of party $P_i$ (resp. $P_j$). Symbolically, the key $k_i$ that party $P_i$ computes can be viewed as the sum of the shared keys between $(\text{ssk}_i, \text{epk}_j)$ (static secret key of $P_i$ with ephemeral public key of $P_j$), $(\text{esk}_i, \text{spk}_j)$ (ephemeral secret key of $P_i$ with static public key of $P_j$) and $(\text{esk}_i, \text{epk}_j)$ (ephemeral secret key of $P_i$ with ephemeral public key of $P_j$). Similarly, $P_j$ computes the sum of the shared keys between $(\text{ssk}_j, \text{epk}_j)$, $(\text{esk}_j, \text{spk}_j)$ and $(\text{esk}_j, \text{epk}_j)$. 

\[ \text{Party } P_i \]

Static $\text{spk}_i$: $p_i = \alpha s_i + 2e_i$
Static $\text{ssk}_i$: $s_i, e_i \leftarrow \chi_\alpha$

\[ r_i, f_i \leftarrow \chi_\alpha \]
\[ x_i = ar_i + 2f_i \]
\[ c \leftarrow H_1(P_i, P_j, x_i) \]
\[ d \leftarrow H_1(P_i, P_j, x_i, y_j) \]
\[ g_j, h_j \leftarrow \chi_\alpha \]
\[ y_j = y_j + ad + 2g_i \]
\[ k_i = (p_i + y_j)(s_i + r_i + c) - p_i s_i + 2h_i \]
\[ \sigma_i \leftarrow \text{Mod}_2(k_i, w_j) \]
\[ \text{sk}_i \leftarrow H_2(P_i, P_j, x_i, y_j, w_j, \sigma_i) \]

\[ x_i \]

\[ y_j, w_j \]

\[ \text{Party } P_j \]

Static $\text{spk}_j$: $p_j = \alpha s_j + 2e_j$
Static $\text{ssk}_j$: $s_j, e_j \leftarrow \chi_\alpha$

\[ r_j, f_j \leftarrow \chi_\alpha \]
\[ y_j = ar_j + 2f_j \]
\[ c \leftarrow H_1(P_i, P_j, x_i) \]
\[ d \leftarrow H_1(P_i, P_j, x_i, y_j) \]
\[ g_j, h_j \leftarrow \chi_\alpha \]
\[ x_i = x_i + ac + 2g_j \]
\[ k_j = (p_i + x_i)(s_j + r_j + d) - p_j s_j + 2h_j \]
\[ w_j \leftarrow \text{Sig}(k_j) \]
\[ \sigma_j \leftarrow \text{Mod}_2(k_j, w_j) \]
\[ \text{sk}_j \leftarrow H_2(P_i, P_j, x_i, y_j, w_j, \sigma_j) \]
More precisely, let \( \text{spk}_i = p_i = a s_i + 2 e_i \) and \( \text{epk} = x_i = a r_i + 2 f_i \) (resp. \( \text{spk}_j = p_j = a s_j + 2 e_j \) and \( \text{epk} = y_j = a r_j + 2 f_j \)) be the static and ephemeral public keys of \( P_i \) (resp. \( P_j \)). The key \( k_i \) computed by \( P_i \) is equal to \( (p_j + \bar{y}_j)(s_i + r_i + c) - p_j s_i \), which is the sum of all possible combinations between static and ephemeral keys of both parties, minus the key resulting of the exchange between static keys. As in the previous construction, we pasteurize \( y_j \) to avoid any leakage of information. Similarly, the key \( k_j \) computed by \( P_j \) is equal to \( (p_i + \bar{x}_i)(s_j + r_j + d) - p_i s_j \). Of course \( k_i \) and \( k_j \) are just approximately equal, hence the functions \( \text{Sig} \) and \( \text{Mod}_2 \) are used to agree on a shared value.

Similarities with NAXOS protocol. We recall the NAXOS AKE protocol of [42] which is based on the Diffie-Hellman protocol. Let \( G \) be a group and let \( (\text{spk}_i, \text{ssk}_i) \) (resp. \( (\text{spk}_j, \text{ssk}_j) \)) and \( (\text{epk}_i, \text{esk}_i) \) (resp. \( (\text{epk}_j, \text{esk}_j) \)) be pairs of static and ephemeral public and secret keys of party \( P_i \) (resp. \( P_j \)). Party \( P_i \) computes the shared key as \( H(\text{epk}_j^{\text{ssk}_i}, \text{spk}_j^{\text{esk}_i}, \bar{\text{epk}}_j^{\text{esk}_i}) \) where \( H \) is a random oracle.

Our AKE protocol shares similarities with the NAXOS protocol. However, we compute the shared key in one operation while NAXOS computes three keys individually. This allows saving a couple of multiplications in the ring \( R_q \), improving the efficiency.

However, we were not able to prove security in the (extended) Canetti-Krawczyk (eCK) model [42], as in NAXOS. This is due to the fact that, given two session transcripts \( (x_i, (y_j, w_j)) \) and \( (x_i, (y_j, w_j')) \), these two sessions have the same state (that is, \( k_j \)) and the eCK model allows the adversary to get the state of parties in a session.

Comparison with scheme of Zhang et al. As far as we are aware, the only RLWE-based AKE scheme with implicit authentication was presented in [56] (we refer to it as ZZD+ scheme, for convenience). We compare our scheme with this one.

In terms of computational complexity, the ZZD+ scheme requires ten multiplications in the ring \( R_q \), five for each party. This is due to the use of the rejection sampling technique [44, 45], in which each party has to check if the ephemeral key leaks information. Although for a proper choice of parameters, the rejection happens rarely, the test has to be done in every execution. The scheme also requires to sample six times from a discrete Gaussian distribution (three for each user), half the number of samples compared with our protocol. However, the ZZD+ scheme requires three distributions \( \chi_\alpha \) and \( \chi_\beta \) with \( \beta >> \alpha \) and \( \chi_\tau \), since the noise flooding technique is used. This turns the

\[3\]The scheme of [56] was not analyzed in the Canetti-Krawczyk (CK) model for the same reason.

\[4\]Rejection sampling is needed in the ZZD+ scheme since the scheme (implicitly) resorts to signatures to provide authentication, just like HMQV which (implicitly) relies on Schnorr signatures [53]. Our design does not require signatures as authentication is provided by the shared keys between ephemeral secret keys and static public keys.

\[5\]The noise flooding technique is used in ZZD+ in order for the ephemeral secret key to
implementation of the scheme way more complicated than the implementation of our scheme, which only needs the distribution $\chi_\alpha$.

<table>
<thead>
<tr>
<th></th>
<th>Rounds</th>
<th>Multiplications in $R_q$</th>
<th>Samples</th>
<th>Required distributions</th>
<th>Rejection Sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td>[58]</td>
<td>2</td>
<td>10</td>
<td>6</td>
<td>$\chi_\alpha$, $\chi_\beta$ ($\beta &gt;&gt; \alpha$), $\chi_\tau$</td>
<td>✓</td>
</tr>
<tr>
<td>Ours</td>
<td>2</td>
<td>6(+2 offline)</td>
<td>8</td>
<td>$\chi_\alpha$</td>
<td>×</td>
</tr>
</tbody>
</table>

Table 1: Comparison with other AKE schemes with implicit authentication.

The ZZD+ scheme requires two elements of $R_q$ and the signal to be sent during the execution of the protocol. Hence, it achieves the same communication complexity as our scheme. However, the use of several discrete Gaussian distributions (in particular, the use of $\chi_\beta$ with $\beta >> \alpha$) implies the use of a larger $q$ than in our scheme for the same security level. This fact leads to much larger parameters to be used, comparing to our proposal, for the same security parameter. We elaborate more on this in Section 6.

1.3 Other previous work

Prominent work on security models for AKE schemes was presented in [11, 19, 42]. Here, we work on the BR-model, which is the most common model used and it is believed to be enough for practical purposes since it also provides composability [18].

The idea of sanitizing the other’s party message in the context of KE with lattices was already employed in [35]. However, the strategy used in [35] consists in multiplying the key obtained by $P_j$ by a small value and then revealing it to $P_i$, so that $P_i$ can also compute the key. Although the authors of [35] give arguments on why their construction is robust to the key reuse attacks of [31, 23], no proofs of security are presented. Contrarily to the construction of [35], we can prove robustness for key reuse for our scheme.

Identity-based Encryption (IBE) schemes can also be used as KEM that provide authentication and several constructions for IBE based on lattices have been proposed before [36, 2, 20, 29]. However, the efficiency of these schemes is too cumbersome to be used in practice.

Password-authenticated key-exchange is yet another flavor of AKE. Previous work on RLWE-based PAKE were presented in [40] (via public-key encryption scheme) and [24] (via KE).

2 Preliminaries

Let $D$ be an algorithm. By $y \leftarrow D(x)$ we denote the output $y$ after running $D$ on input $x$. If $S$ is a set and $\rho$ a distribution over $S$ we denote by $x \leftarrow S$ the obliterate every information about the static secret key and, thus, to allow for key reuse. Our approach for the key reuse problem is to pasteurize RLWE samples, as explained previously.
element \( x \) sampled uniformly at random from \( S \) (if \( S \) is finite) and by \( x \leftarrow \rho \) the element \( x \) sampled from \( S \) according to \( \rho \).

Let \( n \) be a power of 2. For a prime \( q \), let \( R_q = \mathbb{Z}_q[X]/(X^n + 1) \). Notice that \( R_q \) can be embedded into \( \mathbb{R}^n \). In this work, we consider the coefficient embedding where each polynomial \( a(X) = a_0 + a_1 X + \cdots + a_{n-1} X^{n-1} \) is mapped to the vector \( (a_0, \ldots, a_{n-1}) \in \mathbb{R}^n \). For \( a \in R_q \), \( \|a\| \) denotes the usual \( \ell_2 \) norm of the embedded vector \( (a_0, \ldots, a_{n-1}) \in \mathbb{R}^n \) and \( \|a\|_\infty \) denotes its \( \ell_\infty \) norm.

We define the statistical distance between two random variables \( X \) and \( Y \) by

\[
d(X, Y) = \frac{1}{2} \sum_r |\Pr[X = r] - \Pr[Y = r]|.
\]

Let \( \mathcal{X} = \{X_\kappa\} \) and \( \mathcal{Y} = \{Y_\kappa\} \) two probability distributions. We say that \( \mathcal{X} \) and \( \mathcal{Y} \) are statistically close to uniform if \( d(X_\kappa, Y_\kappa) \leq \text{negl}(\kappa) \).

2.1 BR security model

We describe the Bellare-Rogaway (BR) model \cite{BR} adapted to two-pass AKE schemes, which was also used in \cite{56}. In this model, the adversary has full control of the network, which means that it can read, modify, intercept, and inject messages in the network. It is also allowed to reveal session keys that have been established, to modeled possible leaks of information in the real-world use of a protocol, and to reveal the static secret keys of users, in order to capture PFS. We briefly survey the security model.

In this work, an execution of an AKE scheme \( \Pi \) is performed by two parties, the initiator \( I \) and the responder \( R \). Let \( N \) be the number of users using the AKE protocol. Each user has a pair of static public and secret keys. As usual, we assume that static public keys of the users are validated either by a Certificate Authority (CA) or using some other mechanism.

**Session.** A session \( \text{id} = (\Pi, I, P_i, P_j, X, Y) \) (or \( \text{id} = (\Pi, R, P_j, P_i, X, Y) \)) is a single execution of \( \Pi \), where \( I \) (or \( R \)) denotes the role of the session owner, \( P_i \) and \( P_j \) are the parties involved in the session (the first one being the owner of the session), \( X \) is the message sent from \( P_i \) to \( P_j \) and \( Y \) is the message sent from \( P_j \) to \( P_i \). A session has a owner which is the party that activates it. A session is said to be completed when a party computes a session key. A session \( \text{id} = (\Pi, I, P_i, P_j, X, Y) \) has a matching session if \( \widetilde{\text{id}} = (\Pi, R, P_j, P_i, X, Y) \) exists (and vice-versa).

A session can be activated by a message of the form \( (\Pi, I, P_i, P_j) \) (when the session belongs to the initiator) or of the form \( (\Pi, R, P_j, P_i, X) \) (when the session belongs to the responder). In the first case, we say that \( P_i \) is the initiator and it should output a message \( X \). After receiving a message of the form \( (\Pi, R, P_j, P_i, X) \), \( P_j \) takes the role of the responder and should output a message \( Y \) and computes the shared session key. Finally, upon receiving a message of the form \( (\Pi, I, P_i, P_j, X, Y) \), \( P_i \) computes the shared session key, which will be the same as the one computed by \( P_j \).
Oracles. The adversary $A$ has access to the following oracles:

- **Initiate**($\Pi, I, P_i, P_j$): party $P_i$ is activated as the initiator. **Initiate** returns $X_i$, a message intended for party $P_j$.
- **Respond**($\Pi, R, P_j, P_i, X_i$): party $P_j$ is activated as the responder. **Respond** returns $Y_j$, a message intended for $P_i$.
- **Complete**($\Pi, I, P_i, P_j, X_i, Y_j$): the message $Y_j$ is sent to $P_i$ to complete a session previously activated by an **Initiate** query, which outputted $X_i$.
- **skReveal**(sid): it returns the session key of session sid, if it exists.
- **Corrupt**(P_i): it returns the static secret key of party $P_i$.
- **Test**(sid): it chooses $b \leftarrow \{0, 1\}$. If $b = 0$, it returns a uniformly chosen key. Else, it returns the session key of session sid.

A party that has its key revealed (by querying **Corrupt**) is called dishonest. We just allow **Test** to be called once and on a fresh session to avoid trivial attacks. The definition of fresh session is presented below.

Security of AKE. First, we define the concept of a fresh session.

**Definition 1** (Fresh session). Let sid be a completed session and let $\tilde{\text{sid}}$ be the matching session (if it exists). We say that sid is fresh if:

1. **skReveal** was not queried on sid nor on $\tilde{\text{sid}}$;
2. **Corrupt** was not queried on $P_i$ nor on $P_j$, if $\tilde{\text{sid}}$ does not exist.

Weak Perfect Forward Secrecy (wPFS) means that it is infeasible for an adversary to recover a session key that was established without its intervention [41]. This should hold even when the attacker knows the static secret keys of both parties involved in the key exchange. Restricting the adversary to query the **Test** oracle on a fresh session captures the notion of wPFS. Recall that wPFS is the strongest type of PFS that a two-pass KE protocol can achieve [41].

Let $\Pi$ be an AKE scheme and $\kappa$ a security parameter. Consider the following security game: $A$ can query a polynomial number of times the oracles described above, except for **Test** oracle, which is queried only once on a fresh session. Let $b$ be the bit chosen by the oracle **Test**. The game ends with $A$ outputting $b'$, a guess of $b$. We define the advantage of $A$ as $\text{Adv}_{\Pi, A}(\kappa) = \Pr[b' = b] - 1/2$.

**Definition 2.** Let $\Pi$ be a AKE scheme and $\kappa$ be the security parameter. We say that $\Pi$ is secure if $\text{Adv}_{\Pi, A}(\kappa) \leq \text{negl}(\kappa)$, for any adversary $A$. 

9
2.2 Ring-Learning with Errors

In this section, we present the Ring Learning with Errors (RLWE) problem \cite{46}, a famous variant of the Learning with Errors (LWE) problem, firstly presented by Regev \cite{52}.

Let $\rho_{v,\alpha}(a) = \frac{1}{\alpha \sqrt{2\pi}} \exp\left(\frac{-\|a-v\|^2}{2\alpha^2}\right)$ be the probability distribution of the Gaussian distribution over $\mathbb{R}^n$ centered at $v \in \mathbb{R}^n$ and with standard deviation $\alpha$. We define the discrete Gaussian distribution over $\mathbb{R}_q$ centered at $v \in \mathbb{R}_q$ and with standard deviation $\alpha$ by the probability distribution $\chi_{v,\alpha}(a) = \frac{\rho_{v,\alpha}(a)}{\rho_{v,\alpha}(R_q)}$ for all $a \in \mathbb{R}_q$. The subscript $v$ is omitted when it is equal to zero.

We recall some basic facts about the $\ell_2$ and $\ell_{\infty}$ norms: for all $a, b \in \mathbb{R}_q$, $\|a \cdot b\| \leq \sqrt{n} \|a\| \cdot \|b\|$ and $\|a\|_{\infty} \leq \|a\|$.

**Lemma 3** (\cite{50}). For any $\alpha = \omega(\sqrt{\log n})$, we have

$$\text{Pr}[\|x\| \geq \alpha \sqrt{n} : x \leftarrow \chi_\alpha] \leq 2^{-n+1}.$$  

Let $s \leftarrow \mathbb{R}_q$. The RLWE distribution $D_{RLWE}^{s,\chi_\alpha}$ samples $a \leftarrow \mathbb{R}_q$ and $e \leftarrow \chi_\alpha$ and outputs $(a, as + e)$.

**Definition 4** (Ring Learning with Errors). The decision version of the RLWE problem, denoted by RLWE$_q,\chi_\alpha$, asks to distinguish samples $(a, as + e) \leftarrow D_{RLWE}^{s,\chi_\alpha}$ from samples $(a, u) \leftarrow \mathbb{R}_q \times \mathbb{R}_q$.

It was shown that solving the RLWE assumption on average is at least as hard as solving worst-case lattice problems (namely the Approximate Shortest Independent Vector Problem), which is assumed to be hard for classical and quantum computers \cite{46}.

It is well-known that the RLWE problem is still hard when the secret $s$ is sampled from the error distribution $\chi_\alpha$, instead of being chosen uniformly from $\mathbb{R}_q$ \cite{8} \cite{46} \cite{47}. This is usually called the Hermite Normal Form-RLWE (we will denote it by HNF-RLWE$_q,\chi_\alpha$) and it is proven to be as hard as RLWE$_q,\chi_\alpha$.

It is also known that the RLWE problem is still hard when we scale the error of the sample by a constant $t$ (which is co-prime with $q$), that is, $as + te$ \cite{16}. Moreover, it is straightforward to prove that the RLWE problem is still hard when $s$ is sampled from $\chi_{k\alpha}$ for $k = 2$ or $k = 3$, instead of being sampled from $\chi_\alpha$. We will use this fact to guarantee the security of our scheme.

2.3 Signal Function

We define the signal function which was firstly presented in \cite{28} and has found numerous applications such as in key exchange (and its variants) \cite{28} \cite{56} \cite{24}, zero-knowledge proof \cite{26} or oblivious transfer \cite{17}.  

10
Let $E = \{\lfloor \frac{q}{4} \rfloor, \ldots, \lfloor \frac{q}{4} \rfloor \} \subset \mathbb{Z}_q$ and $E + 1 = \{-\lfloor \frac{q}{4} \rfloor + 1, \lfloor \frac{q}{4} \rfloor + 1\}$. We define the functions $\text{Sig}_0, \text{Sig}_1 : \mathbb{Z}_q \to \{0, 1\}$ as

$$
\text{Sig}_0(a) = \begin{cases} 0, & \text{if } a \in E \\ 1, & \text{if } a \notin E \end{cases} \quad \text{and} \quad \text{Sig}_1(a) = \begin{cases} 0, & \text{if } a \in E + 1 \\ 1, & \text{if } a \notin E + 1 \end{cases}
$$

The randomized function $\text{Sig}_*(a)$ is given by choosing $b \leftarrow \{0, 1\}$ and outputting $\text{Sig}_i(a)$. We extend $\text{Sig}_*$ to a function $\text{Sig} : \mathbb{R}_q \rightarrow \mathbb{R}_2$ on $\mathbb{R}_q$ given by $\text{Sig}(a) = (\text{Sig}_1(a_0), \ldots, \text{Sig}_1(a_{n-1}))$, where $a = a_0 + a_1X + \ldots a_{n-1}X^{n-1} \in \mathbb{R}_q$.

**Lemma 5.** Let $k \leftarrow \mathbb{R}_q$ and $w \leftarrow \text{Sig}(k)$. Then $w$ follows a uniform distribution over $\mathbb{R}_2$.

**Proof.** If $k$ is sampled uniformly from $\mathbb{R}_q$, then each of its coefficients is sampled uniformly from $\mathbb{Z}_q$. Then, this means that each coefficient of $k$ has a $1/2$ probability of being in the set $E$. Thus, the signal $w$ of $k$ is a uniformly chosen polynomial from $\mathbb{R}_2$.

We also define the extractor function $\text{Mod}_2 : \mathbb{R}_q \times \mathbb{R}_2 \rightarrow \mathbb{R}_2$ as

$$
\text{Mod}_2(a, w) = \left( a + w \frac{q - 1}{2} \mod q \right) \mod 2.
$$

**Lemma 6 ([28]).** Let $q > 8$, $u, v \in \mathbb{R}_q$ such that $\|u - v\|_\infty < q/4$ and $w \leftarrow \text{Sig}(v)$. Then

$$
\text{Mod}_2(u, w) = \text{Mod}_2(v, w).
$$

Recall that the min-entropy of a random variable $V$ is defined by

$$
- \log \left( \max_{s \in S} \Pr[V = s] \right)
$$

and says that it is infeasible for an adversary (even with unlimited computational power) to guess $v$ chosen uniformly at random from $V$ with probability greater than $2^{-\log(\max_{s \in S} \Pr[V = s])}$.

**Lemma 7 ([56]).** Let $q$ be an odd prime and $\mathbb{R}_q$ be as above. For any $b \in \mathbb{R}_2$ and any $v' \in \mathbb{R}_q$, the output distribution of $\text{Mod}_2(v + v', b)$ given $\text{Sig}(v)$ has min-entropy of at least

$$
-n \log \left( \frac{1}{2} + \frac{1}{|E| - 1} \right)
$$

where $v \leftarrow \mathbb{R}_q$.

By the Lemma above, we have that, when $q > 203$, then $-n \log \left( \frac{1}{2} + \frac{1}{|E| - 1} \right) > 0.97n$ [56].
3 Key exchange with reusable keys

Several attacks have been found on RLWE-based key exchanges [31, 23, 25]. All of these attacks rely on the fact that the signal function leaks information about the secret key. Hence, we want to show that there is no leakage of information in our protocol. In particular, we want to show that there is no leakage of information about the static secret key of each party. This can be proven due to the fact that, by pasteurizing the message sent by the other party, the signal function will look completely random.

In this section, we present a variant of the KE of [28] which allows for key reuse.

3.1 The protocol

Let \( a \leftarrow R_q \). Let \( H_1 : \{0,1\}^* \rightarrow \chi_\alpha \) be a random oracle whose outputs are sampled from \( \chi_\alpha \).

1. \( P_i \) does the following:
   - It samples \( s_i, e_i \leftarrow \chi_\alpha \) and computes \( x_i = as_i + 2e_i \).
   - It sends \( x_i \) to \( P_j \).

2. Upon receiving \( x_i \) from \( P_i \), \( P_j \) does the following:
   - It samples \( s_j, e_j \leftarrow \chi \) and computes \( y_j = as_j + 2e_j \).
   - It computes \( c \leftarrow H_1(P_i, P_j, x_i) \) and \( d \leftarrow H_1(P_i, P_j, x_i, y_j) \).
   - It samples \( f_j \leftarrow \chi_\alpha \) and computes \( \bar{x}_i = x_i + ac + 2f_j \).
   - It samples \( g_j \leftarrow \chi_\alpha \) and computes \( k_j = \bar{x}_j(s_j + d) + 2g_j \).
   - It computes \( w_j \leftarrow \text{Sig}(k_j) \) and sets the session key as \( \text{sk}_j \leftarrow \text{Mod}_2(k_j, w_j) \).
   - It sends \( (y_j, w_j) \) to \( P_i \).

3. Upon receiving \( (y_j, w_j) \) from \( P_j \), \( P_i \) does the following:
   - It computes \( c \leftarrow H_1(P_i, P_j, x_i) \) and \( d \leftarrow H_1(P_i, P_j, x_i, y_j) \).
   - It samples \( f_i \leftarrow \chi_\alpha \) and computes \( \bar{y}_j = y_j + ad + 2f_i \).
   - It samples \( g_i \leftarrow \chi_\alpha \) and computes \( k_i = \bar{y}_j(s_i + c) + 2g_i \).
   - It sets the the session key as \( \text{sk}_i \leftarrow \text{Mod}_2(k_i, w_j) \).

\(^6\)Note that such a hash function \( H_1 \) can be trivially implemented by means of a usual hash function \( \tilde{H}_1 \) and an algorithm \( S \) that samples according to \( \chi_\alpha \), and by using the value \( H(x) \) as the seed in \( S \).
Discussion. The main idea of this new protocol is that an adversary gains nothing by sending something that is not an RLWE sample. To see this, assume that party $P_i$ is dishonest, and controlled by an adversary $A$, (as in the setup of the attacks of [31, 23]) and sends $x_i$ to $P_i$. If $x_i$ is an HNF-RLWE sample, then $\bar{x}_i$ is also a HNF-RLWE sample for which $P_i$ has the corresponding secret. However, when $x_i$ is not an HNF-RLWE sample, but rather a value that follows some other arbitrary distribution over $R_q$, then we prove that $\bar{x}_i$ follows a distribution that is statistically close to the uniform distribution from the point of view of $A$. This happens because $H$ is modeled as a random oracle and, thus, $A$ has no control over it. In particular, it has no control over the value $aH(x_i) + e$ since $H(x_i)$ is independent of $x_i$ (since $H$ is a random oracle) and $e$ is sampled by the other party.

Also, we remark that the probability of $A$ finding $x_i$ such that $\bar{x}_i$ is some particular value is equal to the probability of honestly sampling HNF-RLWE samples and getting the same particular value (which should be negligible).

Correctness. We prove that the scheme is correct with overwhelming probability.

Lemma 8. Suppose that $q > 16(4\alpha^2n^{3/2} + \alpha\sqrt{n})$. Then $sk_i = sk_j$, except with negligible probability.

Proof. By Lemma 2 we have to show that $\|k_i - k_j\|_\infty < q/4$. First, note that

$$k_i = a\tilde{s} + 2\tilde{e}_i \quad \text{and} \quad k_j = a\tilde{s} + 2\tilde{e}_j$$

where

$$\tilde{s} = (s_i + c)(s_j + d)$$

$$\tilde{e}_i = e_is_i + ejc + f_is_i + f_ic + g_i$$

$$\tilde{e}_j = e_is_j + ejd + f_is_j + f_jd + g_j.$$  

Recall that $\|a.b\| \leq \sqrt{n} \|a\| \cdot \|b\|$ for any $a, b \in R_q$. Plugging this fact together with the triangular inequality and Lemma 3 we have that

$$\|k_i - k_j\|_\infty < 2(8\alpha^2n^{3/2} + 2\alpha\sqrt{n}).$$

Since, by assumption, we have that

$$q > 8(8\alpha^2n^{3/2} + 2\alpha\sqrt{n}),$$

then $\|k_i - k_j\|_\infty < q/4$ and correctness of the protocol follows. □

3.2 Security against passive adversaries

Let $A$ be an adversary. Consider the following security game for KE protocols. $A$ is given a honestly generated transcript of the KE. Then, a random bit $b$ is chosen uniformly at random. If $b = 0$, then $A$ is given a uniformly chosen key
and, if \( b = 1 \), then \( A \) is given the actual session key \( k \). Finally, \( A \) must output a bit \( b' \), which a guess of \( b \). We define the advantage of \( A \) to be \( \text{Adv}_{\text{KE},A}^{\text{pas}}(\kappa) = \Pr[b = b'] - 1/2 \) and say that the KE scheme is secure against passive adversaries if \( \text{Adv}_{\text{KE},A}^{\text{pas}}(\kappa) \leq \text{negl}(\kappa) \), for any adversary \( A \).

**Theorem 9.** The scheme is secure against passive adversary, given that \( q \) is a prime as in Lemma 8 and \( \text{HNF-RLWE}_{q,\chi}^{\alpha} \) is hard.

**Proof.** The security proof of the scheme for passive adversaries follows the same line as the proof in [28]. We omit it here. \( \square \)

### 3.3 Robustness of the scheme to key reuse

We prove that the scheme is robust to key reuse. That is, it is infeasible for an adversary to get information about the other party’s secret key \( s \), even when the same keys are reused in several executions of the protocol.

We say that a KE scheme is robust to key reuse if it is robust to key reuse for both parties involved in the protocol (we formally define robustness for each party below).

**Lemma 10** ([26]). Let \( \phi \) be an arbitrary distribution over \( R_q \) and let \( \psi \) be a distribution over \( R_q \) which is statistically close to the uniform distribution over \( R_q \). Let \( x, y \in R_q \) such that \( x \leftarrow \phi \) and \( y \leftarrow \psi \). Then, the distribution of \( \bar{x} = x + y \) is statistically close to uniform.

**Proof.** The proof is presented in [26], however we present it here for completeness. For any \( r \in R_q \), we have that

\[
\Pr[x + y = r] = \sum_{i \in R_q} \Pr[x = r - i] \Pr[y = i]
\]

\[
= \sum_{i \in R_q} \left( \frac{1}{q^n} + \text{negl}(\kappa) \right) \Pr[x = r - i]
\]

\[
= \frac{1}{q^n} \sum_{i \in R_q} \Pr[x = r - i] + \sum_{i \in R_q} \text{negl}(\kappa) \Pr[x = r - i]
\]

\[
\leq \frac{1}{q^n} + \sum_{i \in R_q} \text{negl}(\kappa)
\]

and the sum \( \sum_{i \in R_q} \text{negl}(\kappa) \) is a negligible value. Hence, the distribution of \( \bar{x} = x + y \) is statistically close to the uniform distribution. \( \square \)

**Lemma 11.** Let \( s \leftarrow \chi_\alpha \). Given \( (a, y = as + e) \leftarrow \mathcal{D}_{s,\chi_\alpha}^{\text{RLWE}} \), the probability \( \Pr[y = r] \) for any \( r \in R_q \) is less or equal to \( 1/q^n \).

**Proof.** First, note that, if \( a \in R_q \) is uniformly chosen, then the probability that \( \Pr[as = r] \) for some \( r \in R_q \) is \( 1/q^n \). To see this, note that the product \( a_0s_0 \) is uniform in \( \mathbb{Z}_q \), for \( a_0, s_0 \in \mathbb{Z}_q \) (since \( q \) is prime, then \( \mathbb{Z}_q \) forms a field). Hence,
each coefficient of the product of two polynomials
\[ a = a_0 + a_1 X + \cdots + a_{n-1} X^{n-1} \]
and
\[ s = s_0 + s_1 X + \cdots + s_{n-1} X^{n-1} \]
in \( R_q \) is nothing but a sum of (some of) the coefficients of \( a \) and \( s \). Since the coefficients of \( a \) are uniformly chosen from \( \mathbb{Z}_q \), then the product \( a_k s_\ell \) is also uniform in \( \mathbb{Z}_q \), for any \( k, \ell = 1, \ldots, n-1 \). Therefore \( as \) is uniform in \( R_q \) since its coefficients are sums of uniformly chosen values from \( \mathbb{Z}_q \).

Now, using Lemma 10, we conclude that the value \( as + e \) follows a distribution statistically close to the uniform distribution over \( R_q \). We conclude that
\[ \Pr[as + e = r] = 1/q^n + \text{negl}(\kappa). \]

From the lemma presented above, we immediately get the following corollary.

**Corollary 12.** The distribution \( D_{s, \chi_\alpha}^{RLWE} \) is statistically close to the uniform distribution over \( R_q \).

**Proof.** This is a direct consequence of the previous lemma.

**Corollary 13.** Let \( H \) be a random oracle whose outputs are sampled from \( \chi_\alpha \) and let \( x \leftarrow \phi \), where \( \phi \) is a distribution over \( R_q \), different from \( D_{s, \chi_\alpha}^{RLWE} \), where \( s \leftarrow \chi_\alpha \). Then, the distribution of \( \bar{x} = x + aH(x) + e \), where \( e \leftarrow \chi_\alpha \), is statistically close to the uniform distribution over \( R_q \), in the ROM.

**Proof.** The proof follows from Lemma 10, Corollary 12 and by noting that the distribution of \( H(x) \) is independent of the distribution of \( x \) (because \( H \) is a random oracle).

**Key reuse for party \( P_j \).** Let \( A \) be an adversary. Consider the following security game: \( A \) is allowed to open as many sessions as it wants with party \( P_j \) (always playing the role of \( P_i \)). At some point, a key is exchanged between \( P_i \) and \( P_j \) with \( A \) passively observing. Observe that \( A \) is not given access to the secret key of \( P_j \). Then, a random bit \( b \) is chosen uniformly at random. If \( b = 0 \), then \( A \) is given a uniformly chosen key and, if \( b = 1 \), then \( A \) is given the actual session key \( k \) (computed by \( P_j \)) between \( P_i \) and \( P_j \). Finally, \( A \) must output a bit \( b' \), which a guess of \( b \). We define the advantage of \( A \) to be
\[ \text{Adv}_{\text{KE}, A}(\kappa) = \Pr[b = b'] - 1/2 \]
and say that the KE scheme is robust to key reuse for \( P_j \) if \( \text{Adv}_{\text{KE}, A}(\kappa) \leq \text{negl}(\kappa) \), for any adversary \( A \).

**Theorem 14.** Let \( q \) be as in Lemma 8. The proposed KE scheme is robust to key reuse for party \( P_j \) in the ROM, given that the HNF-RLWE \( q, \chi_\alpha \) is hard.

**Proof.** First, we show that, whatever the strategy used by the adversary \( A \), it cannot get any information on the secret key \( s_j \) of party \( P_j \). When interacting with \( P_j \), \( A \) sends a value \( x \in R_q \). There are two cases to consider:

1. \( (a, x) \leftarrow D_{s, \chi_\alpha}^{RLWE} \), where \( s \leftarrow \chi_\alpha \).

2. The value \( x \) follows some other distribution \( \phi \) over \( R_q \), different from \( D_{s, \chi_\alpha}^{RLWE} \), where \( s \leftarrow \chi_\alpha \).
The case 1 reduces to the case of passive security. Hence, in this case, the adversary $A$ cannot get information about $s_j$ (the secret of $P_j$).

In the second case, suppose that $A$ sends $x$ sampled from an arbitrary distribution $\phi$. Then, by Lemma 13, the distribution of $\tilde{x} = x + a\mathcal{H}(x) + e$, where $e$ is sampled from $\chi_\alpha$ by $P_j$, is statistically close to the uniform distribution. Hence, by the HNF-RLWE assumption, the key $k_j$ is indistinguishable from a uniformly chosen value of $R_q$ from the point-of-view of $A$. Then, by Lemma 5, the signal $w_j$ is uniform in $R_2$. By a simple hybrid argument, we can replace the key $k_j$ and the signal $w_j$ of each of these sessions by random elements of $R_q$ and $R_2$, by Lemma 5. We conclude that, in this case, it is infeasible for $A$ to get $s_j$, except with negligible probability.

If it is infeasible for $A$ to get information about $s_j$ when interacting with $P_j$, then the case where $A$ passively observes the execution the protocol between $P_i$ and $P_j$ falls in the case of Theorem 9. Thus, it follows that $A$ has a negligible advantage in the game and, therefore, the scheme is robust to key reuse for party $P_j$.

**Key reuse for party $P_i$.** Similarly, we define the concept of robustness to key reuse for $P_i$.

Consider the following security game for any adversary $A$: $A$ is allowed to open as many sessions as it wants with party $P_i$ (always playing the role of $P_j$). At some point, a key is exchanged between $P_i$ and $P_j$ with $A$ watching passively. Observe that $A$ is not given access to the secret key of $P_j$. Similarly to the previous case, a random bit $b$ is chosen uniformly at random. If $b = 0$, then $A$ is given a uniformly chosen key and, if $b = 1$, then $A$ is given the actual session key $k$ (computed by party $P_i$) between $P_i$ and $P_j$. Finally, $A$ must output a bit $b'$, which a guess of $b$. We define the advantage of $A$ to be $\text{Adv}_{\text{KE},A}^{\text{nri}}(\kappa) = \Pr[b = b'] - 1/2$ and say that the KE scheme is robust to key reuse for $P_i$ if $\text{Adv}_{\text{KE},A}^{\text{nri}}(\kappa) \leq \text{negl}(\kappa)$, for any adversary $A$.

**Theorem 15.** Let $q$ be as in Lemma 8. The proposed KE scheme is robust to key reuse for party $P_i$ in the ROM, given that the HNF-RLWE$_{q,\chi_\alpha}$ is hard.

**Proof.** The analysis is similar to the proof of Theorem 14. We omit it for briefness.

### 3.4 Efficiency and comparison

We compare our scheme with Ding’s KE [28] (based on the RLWE assumption) in terms of computational complexity. The comparison is presented in Table 2. Our proposal maintains the same communication complexity: The same amount of information is exchanged in the same number of rounds. As we can see in Table 2 we obtain a small computational overhead in order to guarantee robustness to key reuse.
We specify it in full detail: these functions are modeled as random oracles.

Similarly for party $P_j$, its static public key is $p = as_j + 2e_j$ and its static secret key is $s_j$. Let $H_1 : \{0, 1\}^* \rightarrow \chi_\alpha$ be a hash function whose outputs are sampled from $\chi_\alpha$ and let $H_2 : \{0, 1\}^* \rightarrow \{0, 1\}^\epsilon$ be a key derivation function. Both of these functions are modeled as random oracles.

The protocol is composed by three algorithms: **Initiate**, **Respond** and **Complete**. We specify it in full detail:

1. **Initiate**: $P_i$ does the following:
   - It samples $r_i, f_i \leftarrow \chi_\alpha$ and computes $x_i = ar_i + 2f_i \mod q$.
   - It sends $x_i$ to $P_j$.

2. **Respond**: Upon receiving $x_i$, $P_j$ does the following:
   - It samples $r_j, f_j \leftarrow \chi_\alpha$ and computes $y_j = ar_j + 2f_j \mod q$.
   - It computes $c \leftarrow H_1(P_i, P_j, x_i)$ and $d = H_1(P_i, P_j, x_i, y_j)$.
   - It samples $g_j \leftarrow \chi_\alpha$ and computes $\bar{x}_j = x_i + ac + 2g_j$.
   - It samples $h_j \leftarrow \chi_\alpha$ and computes
     $$k_j = (p_i + \bar{x}_j)(s_j + r_j + d) - p_i s_j + 2h_j.$$  
   - It computes $w_j \leftarrow \text{Sig}(k_j)$ and $\sigma_j \leftarrow \text{Mod}_2(k_j, w_j)$.
   - It sets $sk_j \leftarrow H_2(P_i, P_j, x_i, y_j, w_j, \sigma_j)$ as the shared key.
   - It sends $(y_j, w_j)$ to $P_i$.

3. **Complete**: Upon receiving $(y_j, w_j)$, $P_i$ does the following:
   - It sets $c \leftarrow H_1(P_i, P_j, x_i)$ and $d = H_1(P_i, P_j, x_i, y_j)$
   - It samples $g_i \leftarrow \chi_\alpha$ and computes $\bar{y}_j = y_j + ad + 2g_i$.
   - It samples $h_i \leftarrow \chi_\alpha$ and computes
     $$k_i = (p_j + \bar{x}_i)(s_i + r_i + c) - p_i s_i + 2h_i.$$  
   - It computes $\sigma_i \leftarrow \text{Mod}_2(k_i, w_j)$.
   - It sets $sk_i \leftarrow H_2(P_i, P_j, x_i, y_j, w_j, \sigma_i)$ as the shared key.
The following lemma proves the correctness of the scheme, that is, parties $P_i$ and $P_j$ end up with the same key after executing the protocol.

**Lemma 16** (Correctness). If $q > 8 \left(16\alpha^2 n^{3/2} + 2\alpha\sqrt{n}\right)$ then $sk_i = sk_j$, except with negligible probability.

**Proof.** To prove the correctness of the scheme it is enough to show that $\sigma_i = \sigma_j$. By Lemma 6 we have to show that $\|k_i - k_j\|_\infty < q/4$. First, note that $k_i = a\tilde{s} + 2\tilde{e}_i$ and $k_j = a\tilde{s} + 2\tilde{e}_j$ where

\[
\tilde{s} = r_js_i + ds_i + s_jr_i + r_jr_i + dr_i + s_jc + r_jc + dc
\]

\[
\tilde{e}_i = f_js_i + g_js_i + e_jr_i + f_jr_i + g_ir_i + e_jc + f_jc + g_ic + h_i
\]

\[
\tilde{e}_j = f_is_j + g_js_j + e_ir_j + f_ir_j + g_ir_j + e_id + f_id + g_jd + h_j.
\]

Recall that $\|a.b\| \leq \sqrt{n} \|a\| \cdot \|b\|$ for any $a, b \in R_q$. Plugging this fact together with the triangular inequality and Lemma 3 we have that

\[
\|k_i - k_j\|_\infty \leq 2 \left(16\alpha^2 n^{3/2} + 2\alpha\sqrt{n}\right).
\]

Since, by assumption, we have that

\[
q > 8 \left(16\alpha^2 n^{3/2} + 2\alpha\sqrt{n}\right),
\]

then $\|k_i - k_j\|_\infty < q/4$ and the correctness of the protocol follows. \qed

## 5 Security proof for the AKE scheme

Before proving security of the scheme in the BR-model, remark that no information about the secret static keys of each party is leaked during the execution of each session. This is guaranteed by Theorem 14 and Theorem 15.

We present the result that guarantees the security of the proposed scheme in the BR-model.

**Theorem 17.** Let $\kappa$ be the security parameter. Suppose that $n$ is a power of 2 such that $0.97n \geq 2\kappa$, $q$ is a prime such that $q > 203$ and $q > 8 \left(16\alpha^2 n^{3/2} + 2\alpha\sqrt{n}\right)$, and the HNF-RLWE$_{q,\chi,\alpha}$ is hard. Then, the proposed AKE scheme is secure in the BR-model in the ROM.

The rest of this section is dedicated to the proof of this Theorem, which follows from Lemmas 18, 25, 32, 39 and 16.

First, note that the test session may have a matching session or not. When it has a matching session, then the adversary can corrupt parties and recover their static secret key, by the definition of fresh session (Definition 1). We enumerate the several types of adversaries:
• Adversary chooses a test session that has a matching session:
  
  - **Type A₁**: Let $\text{sid} = (\Pi, I, P_i^*, P_j^*, x_i^*, (y_j^*, w_j^*))$ be the test session where $y_j^*$ was outputted by $\text{Respond}(\Pi, R, P_j^*, P_i^*, x_i^*)$.
  
  - **Type A₂**: Let $\text{sid} = (\Pi, R, P_j^*, P_i^*, x_i^*, (y_j^*, w_j^*))$ be the test session where $x_i^*$ was outputted by $\text{Initiate}(\Pi, I, P_i^*, P_j^*)$ and $P_i^*$ either completes the session with $y_j^*$ or it never completes it.

• Adversary chooses a test session that does not have a matching session:

  - **Type A₃**: Let $\text{sid} = (\Pi, R, P_j^*, P_i^*, x_i^*, (y_j^*, w_j^*))$ be the test session where $x_i^*$ was not outputted by $\text{Initiate}(\Pi, I, P_i^*, P_j^*)$.
  
  - **Type A₄**: Let $\text{sid} = (\Pi, I, P_j^*, P_i^*, x_i^*, (y_j^*, w_j^*))$ be the test session where $(y_j^*, w_j^*)$ was not outputted by $\text{Respond}(\Pi, R, P_j^*, P_i^*, x_i^*)$.
  
  - **Type A₅**: Let $\text{sid} = (\Pi, R, P_j^*, P_i^*, x_i^*, (y_j^*, w_j^*))$ be the test session where $x_i^*$ was not outputted by $\text{Initiate}(\Pi, I, P_i^*, P_j^*)$ but $P_i^*$’s session is completed with $y_j^* \neq y_j^*$.

Weak perfect forward secrecy (wPFS) is obtained from **Type A₁** and **Type A₂** adversaries, since these types of adversaries can corrupt parties involved in the test session. Observe that **Type A₃**, **Type A₄** and **Type A₅** adversaries cannot query **Corrupt** on either $P_i^*$ or $P_j^*$, since $\text{sid}^*$ has no matching session.

The idea of the proof is very simple. Consider, for example, a session belonging to party $P_i^*$, when interacting with $P_j^*$. Either $P_j^*$’s static public key $p_j^*$ or ephemeral public key $y_j^*$ are indistinguishable from a uniformly chosen value to the adversary: When the test session has a matching session, the adversary is allowed to get the static secret keys of both parties but it is not allowed to modify the messages exchanged between them (by the BR-model). Therefore, in this case the ephemeral public key of $P_j^*$ can be replaced by a uniformly chosen value. Thus, the key obtained by $P_i^*$ is indistinguishable from a uniformly chosen value by the HNF-RLWE assumption. When the test session does not have a matching session, then the adversary is not allowed to get static secret keys. Thus, the static public key of $P_j^*$ is indistinguishable from a uniformly random value and, by the same reasoning, the shared key obtained by $P_i^*$ is also indistinguishable from a uniformly chosen value. In this case, we just have to show that the simulator is able to simulate the execution of sessions involving $P_j^*$ which it can since it knows all the static secret keys of all other parties.

### 5.1 Test session has a matching session

By the definition of freshness, when the test session has a matching session, then the adversary is allowed to corrupt both parties, get their static secret key and eavesdrop the communication.

Here, we have two possible types of adversaries: one that uses a test session belonging to the initiator and other that uses a test session belonging to the responder.
5.1.1 Adversary Type $A_1$

**Lemma 18.** For any adversary $A$ of Type $A_1$, the advantage $\text{Adv}_{\Pi, A}$ is negligible in the ROM, given that $0.97n \geq 2\alpha$, $q$ is a prime such that $q > 203$ and $q > 8(16a^3n^{3/2} + 2\alpha\sqrt{n})$, and the HNF-RLWE$_{q, x_0}$ is hard.

The proof of this lemma follows from the sequence of games $G_{1,0}, \ldots, G_{1,4}$.

**Game $G_{1,0}$**. The simulator chooses $a \leftarrow R_q$ and creates static public keys for each user, following the protocol. The simulator $S$ chooses $(I, I, P_1^*, P_j^*, x_1^*, (y_j^*, w_j))$ as the test session where $P_1^* \leftarrow \{P_1, \ldots, P_N\}$, $s_1^*, s_j^* \leftarrow \{1, \ldots, m\}$, $x_i^*$ is outputted by a query $\text{Initiate}(I, I, P_i^*, P_j^*)$ on the $s_i^*$-th session of $P_i^*$, $y_j^*$ is outputted by $\text{Respond}(I, R, P_j^*, P_i^*, x_i^*)$ on the $s_j^*$-th session of $P_j^*$. $S$ runs internally $A$ and simulates the oracles in the following way:

- $H_1$ and $H_2$: let $L_1$ and $L_2$ be two lists of pairs $(q, h)$ (i.e., query made to the random oracles and their respective response). As usual, if the query $q$ is made to $H_1$, $S$ checks if there is a pair $(q, h)$ in $L_1$. If there is, it returns $h$, else $S$ samples $h \leftarrow \chi$, returns $h$ and keeps $(q, h) \in L_1$. If the query $q$ is made to $H_2$, $S$ checks if there is a pair $(q, h)$ in $L_2$. If there is, it returns $h$, else $S$ chooses $h \leftarrow R_q$, returns $h$ and keeps $(q, h) \in L_2$.

- $\text{Initiate}$, $\text{Respond}$ and $\text{Complete}$ are simulated following the AKE protocol.

- $\text{skReveal}$ and $\text{Corrupt}$ as described in Section 2.1.

- $\text{Test}(\text{sid})$ : Let $\text{sid} = (I, I, P_i, P_j, x_i, (y_j, w_j))$ be the test session queried by $A$. If $(P_i, P_j) \neq (P_i^*, P_j^*)$ or $x_i$ and is not outputted in the $s_i^*$-th session of $P_i^*$ or $y_j$ is not outputted in the $s_j^*$-session of $P_j^*$, $S$ aborts the execution. Else, it chooses $b \leftarrow \{0, 1\}$ and returns either a random key $\text{sk}_i^* \leftarrow \{0, 1\}^\kappa$, if $b = 0$, or the session key $\text{sk}_i$ of session $\text{sid}$, if $b = 1$.

**Claim 19.** The probability that $S$ aborts in $G_{1,0}$ is $\frac{1}{m^rN^r}$.

**Game $G_{1,1}$**. $S$ simulates the oracles as in $G_{1,0}$ except for $\text{Complete}$:

- When it is queried on $\text{Complete}(I, I, P_i, P_j, x_i, (y_j, w_j))$, $S$ does the following: If $(P_i, P_j) = (P_i^*, P_j^*)$ and it is the $s_i^*$-th session of $P_i$ and $(y_j, w_j)$ was outputted on the $s_j^*$-th session of $P_j$, then $S$ sets $\text{sk}_i = \text{sk}_j$.

**Claim 20.** For every adversary $A$, the probability that $A$ distinguishes between games $G_{1,0}$ and $G_{1,1}$ is negligible.

**Proof.** Note that completeness of the protocol still holds in game $G_{1,1}$. Hence, there is no difference between the games $G_{1,0}$ and $G_{1,1}$. □
Game $G_{1.2}$. $S$ simulates the oracles as in $G_{1.1}$, except for $\text{Initiate}$:

- When it is queried $\text{Initiate}(II, I, P_i, P_j)$, $S$ does the following: If $(P_i, P_j) = (P_i^*, P_j^*)$ and it is the $s_i^*$-th session of $P_i$, then $S$ samples $x_i \leftarrow R_q$ (instead of computing $x_i = ar_i + 2f_i$).

Claim 21. For every adversary $A$, the probability that $A$ distinguishes between games $G_{1.1}$ and $G_{1.2}$ is negligible, given that HNF-RLWE$_{q,\chi}$ is hard.

Proof. It is straightforward to construct an algorithm that decides the HNF-RLWE problem, if there is an algorithm that can distinguish both games. \hfill $\square$

Game $G_{1.3}$. $S$ simulates the oracles as in $G_{1.1}$, except for $\text{Complete}$:

- When it is queried $\text{Complete}(II, I, P_i, P_j, x_i, (y_j, w_j))$, $S$ does the following:
  - If $(P_i, P_j) = (P_i^*, P_j^*)$ and it is the $s_i^*$-th session of $P_i$ and $(y_j, w_j)$ was not outputted on the $s_j^*$-th session of $B$, then $S$ samples $k_i \leftarrow R_q$.

Claim 22. For every adversary $A$, the probability that $A$ distinguishes between games $G_{1.2}$ and $G_{1.3}$ is negligible, given that HNF-RLWE$_{q,\chi}$ is hard.

Proof. In this case, we do not know which is the distribution of the value $y_j$ since it was not outputted by the oracle $\text{Respond}$. However, by Corollary [13] we have that the pasteurization $\bar{y}_j$ of $y_j$ is statistically close to a uniformly chosen value, independently of the distribution of $y_j$. Hence, we consider the key computed by $P_i$ which is

\[ k_i = (p_j + \bar{y}_j)(s_i + r_i + c) - p_j s_i + 2h_i. \]

Rewriting the expression, we have that

\[ k_i = \bar{y}_j (s_i + r_i + c) + 2h_i + p_j (r_i + c). \]

By the HNF-RLWE assumption, we have that $\bar{y}_j (s_i + r_i + c) + 2h_i$ is indistinguishable from a uniformly chosen value in $R_q$, since $\bar{y}_j$ is uniform in $R_q$, and $(s_i + r_i + c)$ and $h_i$ are discrete Gaussian samples. Observe that the HNF-RLWE assumption still holds when the secret is chosen from $\chi_{\sqrt{\alpha}}$ and the error from the distribution $\chi_{\sqrt{\alpha}}$.\footnote{It is trivial to build the reduction: Given a HNF-RLWE sample $(a, y = as + e)$ where $s \leftarrow \chi_{\alpha}$ and $e \leftarrow \chi_{\alpha}$, just choose $s', s'' \leftarrow \chi_{\alpha}$ and $e \leftarrow \chi_{\alpha}$, and compute $y' = y + a(s' + s'') + e$.}

Hence, consider the following hybrid game $G_{1.2}'$, where $S$ chooses $r_i \leftarrow R_q$ and computes $k_i = r_i + p_j (r_i + c)$. From the reasoning above, it is infeasible for any adversary $A$ to distinguish $G_{1.2}$ from the hybrid game $G_{1.2}'$, given that the HNF-RLWE assumption holds.

Since $r_i$ is uniform in $R_q$ then, by Lemma [10] we have that $k_i$ is also uniform in $R_q$. Hence $G_{1.2}'$ and $G_{1.3}$ are indistinguishable from the point of view of the adversary $A$.\hfill $\square$
Game $G_{1,4}$. $S$ simulates the oracles as in $G_{1,3}$, except for Respond:

- When it is queried on $\text{Respond}(\Pi, R, P_i, P_j, x_i)$, $S$ does the following: if $(P_i, P_j) = (P_i^*, P_j^*)$ and $x_i$ was outputted in the $s_i^*$-th session of $P_i^*$ and it is the $s_j^*$-th session of $P_j^*$, $S$ chooses $y_j \leftarrow R_q$ and the key $k_j \leftarrow R_q$. It sends $(y_j, w_j)$ where $w_j \leftarrow \text{Sig}(k_j)$. Else, it simulates $\text{Respond}$ as in $G_{1,3}$.

Claim 23. For every adversary $A$, the probability that $A$ distinguishes between games $G_{1,3}$ and $G_{1,4}$ is negligible, given that $\text{HNF-RLWE}_{q,\kappa}$ is hard.

Proof. The key computed by $P_j$ is equal to

$$k_j = (p_i + \bar{x}_i)(s_j + r_j + d) - p_is_j + 2h_j.$$ 

Using the same argument as in the proof of Claim 22, we conclude that games $G_{1,3}$ and $G_{1,4}$ are indistinguishable from the point of view of the adversary. \qed

Finally, we prove that the advantage of any adversary in the game $G_{1,4}$ is negligible.

Claim 24. For any adversary $A$, the advantage $\text{Adv}_{\Pi, A}$ in game $G_{1,4}$ is negligible, given that $0.97n > 2\kappa$.

Proof. Since $k_i$ is chosen uniformly from $R_q$, we have that $\sigma_i$ has high min-entropy, even when $\sigma_j$ is given, by Lemma 7. In particular, when $0.97n > 2\kappa$ then the probability that $A$ queries $H_2$ on input $(P_i, P_j, x_i, y_j, w_j, \sigma_j)$ is at most $2^{-0.97n} + \text{negl}(\kappa)$. \qed

5.1.2 Adversary Type $A_2$

Lemma 25. For any adversary $A$ of Type $A_2$, the advantage $\text{Adv}_{\Pi, A}$ is negligible in the ROM, given that $0.97n \geq 2\kappa$, $q$ is a prime such that $q > 203$ and $q > 8(16\alpha^2 n^{3/2} + 2\alpha\sqrt{n})$, and the $\text{HNF-RLWE}_{q,\kappa}$ is hard.

The proof of this lemma follows from the sequence of games $G_{2,0}, \ldots, G_{2,4}$.

Game $G_{2,0}$. Similar to $G_{1,0}$ but now $S$ chooses $\text{sid} = (\Pi, R, P_i^*, P_j^*, x_i^*, (y_j^*, w_j^*))$ as the test session where $P_i^*, P_j^* \leftarrow \{P_1, \ldots, P_N\}$, $s_i^*, s_j^* \leftarrow \{1, \ldots, m\}$, $x_i^*$ is outputted by a query $\text{Initiate}(\Pi, I, P_i^*, P_j^*)$ on the $s_i^*$-th session of $P_i^*$ and $y_j^*$ is outputted by $\text{Respond}(\Pi, R, P_i^*, P_j^*, x_i^*)$ on the $s_j^*$-th session of $P_j^*$. $S$ runs internally $A$ and simulates the oracles as in $G_{1,0}$ except for $\text{Test}$:

- $\text{Test(sid)}$: Let $\text{sid} = (\Pi, R, P_j, P_j, x_i, (y_j, w_j))$ be the test session queried by $A$. If $(P_i, P_j) \neq (P_i^*, P_j^*)$ or $x_i$ and is not outputted in the $s_i^*$-th session of $P_i^*$ or $y_j$ is not outputted in the $s_j^*$-th session of $P_j^*$, $S$ aborts the execution.

Else, it chooses $b \leftarrow \{0, 1\}$ and returns either a random key $sk_b^i \leftarrow \{0, 1\}^\kappa$, if $b = 0$, or the session key $sk_i$ of session $\text{sid}$, if $b = 1$.

Claim 26. The probability that $S$ aborts in $G_{2,0}$ is $\frac{1}{m^2N^2}$. 22
Claim 27. For every adversary \( A \), the probability that \( A \) distinguishes games \( G_{2,0} \) and \( G_{2,1} \) is negligible.

Proof. Similar to the proof of Claim 20 \( \square \)

Game \( G_{2,1} \). \( S \) simulates the oracles as in \( G_{2,0} \), except for Complete:

- When it is queried Complete\((\Pi, I, P_i, P_j, x_i, (y_j, w_j))\), \( S \) does the following:
  - If \((P_i, P_j) = (P_i^*, P_j^*)\) and it is the \( s_j^* \)-th session of \( P_i^* \) and \((y_j, w_j)\) was not outputted on the \( s_j^* \) session of \( P_j \), \( S \) sets \( sk_i = sk_j \). Else, it simulates the oracles as in \( G_{2,0} \).

Claim 28. For every adversary \( A \), the probability that \( A \) distinguishes games \( G_{2,1} \) and \( G_{2,2} \) is negligible, given that HNF-RLWE\(_{q,\alpha} \) is hard.

Proof. Similar to the proof of Claim 21 \( \square \)

Game \( G_{2,2} \). \( S \) simulates the oracles as in \( G_{2,1} \), except for Initiate:

- When it is queried Initiate\((\Pi, I, P_i, P_j)\), \( S \) does the following:
  - If \((P_i, P_j) = (P_i^*, P_j^*)\) and it is the \( s_i^* \)-th session of \( P_i^* \) and \((y_j, w_j)\) was not outputted in the \( s_j^* \)-th session of \( B \), then \( S \) samples \( k_i \leftarrow R_q \), instead of computing \( x_i = a r_i + 2 f_i \).

Claim 29. For every adversary \( A \), the probability that \( A \) distinguishes between games \( G_{1,2} \) and \( G_{1,3} \) is negligible, given that HNF-RLWE\(_{q,\alpha} \) is hard.

Proof. Similar to the proof of Claim 22 \( \square \)

Game \( G_{2,3} \). \( S \) simulates the oracles as in \( G_{2,2} \), except for Complete:

- When it is queried Complete\((\Pi, I, P_i, P_j, x_i, (y_j, w_j))\), \( S \) does the following:
  - If \((P_i, P_j) = (P_i^*, P_j^*)\) and it is the \( s_i^* \)-th session of \( P_i \) and \((y_j, w_j)\) was not outputted on the \( s_j^* \) session of \( B \), then \( S \) samples \( k_i \leftarrow R_q \).

Claim 30. For every adversary \( A \), the probability that \( A \) distinguishes between games \( G_{1,3} \) and \( G_{1,4} \) is negligible, given that HNF-RLWE\(_{q,\alpha} \) is hard.

Proof. Similar to the proof of Claim 23 \( \square \)

Claim 31. For any adversary \( A \), the advantage \( \text{Adv}_{\Pi, A} \) in game \( G_{2,4} \) is negligible, given that the HNF-RLWE assumption holds and \( 0.97 n \geq 2 \kappa \).

Proof. Similar to the proof of Claim 24 \( \square \)
5.2 Test session does not have a matching session

When the test session does not have a matching session, then the adversary is not allowed to corrupt parties involved in the test session.

However, here we cannot replace the messages sent by the parties by random values since these messages do not have to be created according to the protocol. Since the adversary cannot ask \texttt{Corrupt} for none of these parties, then we can replace their static public key by random values. It should be infeasible for an adversary to notice that the public key of a party was replaced a random value by the HNF-RLWE assumption. When the static public key of a party is replaced by a random value, then the key computed by another party interacting with the first is also indistinguishable from uniformly random value, by the HNF-RLWE.

So, we may conclude that the advantage of an adversary is negligible.

5.2.1 Adversary Type $A_3$

**Lemma 32.** For any adversary $A$ of Type $A_3$, the advantage $\text{Adv}_{G_{A},A}$ is negligible in the ROM, given that $0.97n \geq 2\kappa$, $q$ is a prime such that $q > 203$ and $q > 8(16\alpha^2n^{3/2} + 2\alpha\sqrt{n})$, and the HNF-RLWE$_{q,x_0}$ is hard.

The proof of this lemma follows from Claims 33, ..., 38.

**Game $G_{3,0}$.** Similar to $G_{1,0}$ but now $S$ chooses $\text{sid} = (\Pi, R, P^*_j, P^*_i, x^*_i, (y^*_j, w^*_j))$ as the test session where $P^*_i, P^*_j \leftarrow \{P_1, \ldots, P_N\}$, $s^*_j \leftarrow \{1, \ldots, m\}$, $y^*_j$ is outputted by \texttt{Respond}(\Pi, R, P^*_j, P^*_i, x^*_i) on the $s^*_j$-th session of $P^*_j$. $S$ runs internally $A$ and simulates the oracles as in $G_{1,0}$ except for Test:

- Test($\text{sid}$): Let $\text{sid} = (\Pi, R, P_j, P_i, x_i, (y_j, w_j))$ be the test session queried by $A$. If $(P_i, P_j) \neq (P^*_i, P^*_j)$ or $y_j$ is not outputted in the $s^*_j$ session of $P^*_j$, $S$ aborts the execution. Else, it chooses $b \leftarrow \{0, 1\}$ and returns either a random key $sk_i \leftarrow \{0, 1\}^\kappa$, if $b = 0$, or the session key $sk_i$ of session $\text{sid}$, if $b = 1$.

**Claim 33.** The probability that $S$ aborts in $G_{3,0}$ is $\frac{1}{mN^2}$.

*Proof.* The probability of choosing the right session, out of $m$ possible values, and the right parties, out of $N$ possibilities is $1/(mN^2)$. \hfill $\square$

**Game $G_{3,1}$.** $S$ simulates the oracles as in $G_{3,0}$, except for \texttt{Initiate} and \texttt{Complete}:

- When it is queried \texttt{Complete}(\Pi, I, P_i, P_j, x_i, (y_j, w_j)), $S$ does the following:
  - If $P_j = P^*_i$ and it is the $s^*_j$-th session of $P^*_i$, $S$ computes the key
    $$k_i = p_j(r_i + c) + \bar{y}_j(s_j + r_i + c) + 2h_i.$$ 
  Else, it simulates as in $G_{3,0}$.

**Claim 34.** For every adversary $A$, the probability that $A$ distinguishes games $G_{3,0}$ and $G_{3,1}$ is negligible.
Proof. Note that \( S \) knows all the static secret keys. So, by setting \( k_j = p_j(r_j + c) + \bar{y}_j(s_j + r_i + c) + 2h_i \), we guarantee the correctness of the scheme in the simulation.

Game \( G_{3,2} \). \( S \) simulates the oracles as in \( G_{3,1} \), except for \textbf{Respond}:

- When it is queried \textbf{Respond}(\( \Pi, R, P_j, P_i, x_i \)), \( S \) does the following: if \( P_j = P_i^* \) and it is the \( s_i^* \)-th session of \( P_i^* \), \( S \) computes
  \[
  k_j = p_i(r_j + d) + \bar{x}_i(s_i + r_j + d) + 2h_j.
  \]

  Else it simulates \textbf{Respond} as in \( G_{3,1} \).

\textbf{Claim 35.} For every adversary \( A \), the probability that \( A \) distinguishes games \( G_{3,1} \) and \( G_{3,2} \) is negligible.

\textbf{Proof.} Again, note that \( S \) knows all the static secret keys. Hence, by setting \( k_j = p_i(r_j + d) + \bar{x}_i(s_i + r_j + d) + 2\bar{h}_j \), we guarantee the correctness of the scheme in the simulation.

Game \( G_{3,3} \). \( S \) simulates the oracles as in \( G_{3,2} \), except for:

- It replaces \( p_i^* \) (the static public key of \( P_i^* \)) by \( u_i \leftarrow R_q \)

\textbf{Claim 36.} For every adversary \( A \), the probability that \( A \) distinguishes games \( G_{3,2} \) and \( G_{3,3} \) is negligible, given that HNF-RLWE\( _{q,x} \) is hard.

\textbf{Proof.} Since the value \( p_i^* \) is a HNF-RLWE sample, then it is indistinguishable from a uniformly random value given that the HNF-RLWE assumption holds.

Game \( G_{3,4} \). \( S \) simulates the oracles as in \( G_{3,4} \), except for:

- When it is queried \textbf{Respond}(\( \Pi, R, P_j, P_i, x_i \)), \( S \) does the following: If \( (P_i, P_j) = (P_i^*, P_j^*) \) and it is the \( s_i^* \)-th session of \( P_i^* \) and \( x_i \) not was outputted by \textbf{Initiate}(\( \Pi, I, P_i^*, P_j^* \)), \( S \) chooses \( k_j \leftarrow R_q \). Else, it simulates \textbf{Respond} as in \( G_{3,3} \).

\textbf{Claim 37.} For every adversary \( A \), the probability that \( A \) distinguishes games \( G_{3,3} \) and \( G_{3,4} \) is negligible, given that HNF-RLWE\( _{q,x} \) is hard.

\textbf{Proof.} Remark that the key \( k_j \) is computed as

\[
  k_j = p_i^*(r_j + d) + \bar{x}_i(s_i + r_j + d) + 2h_j.
\]

Remark that the term \( p_i^*(r_j + d) + 2\bar{h}_j \) is a HNF-RLWE sample since \( p_i^* \leftarrow R_q \), \( r_j + d \leftarrow \chi_{\sqrt{q}} \), and \( h_j \) is an error term sampled from \( \chi_{\alpha} \).

Therefore, we define a hybrid game \( G'_{3,3} \), where the simulator chooses \( t_j \leftarrow R_q \) and computes \( k_j = t_j + \bar{x}_i(s_i + r_j + d) \). Games \( G_{3,3} \) and \( G'_{3,3} \) are indistinguishable by the HNF-RLWE assumption.

Now, since \( t_j \) is uniform, \( k_j \) is also uniform in \( R_q \) by Lemma 10. Hence, games \( G'_{3,3} \) and \( G_{3,4} \) are indistinguishable.
Claim 38. For any adversary $A$, the advantage $\text{Adv}_{\Pi, A}$ in game $G_{3,4}$ is negligible, given that the HNF-RLWE assumption holds and $0.97n \geq 2\kappa$.

Proof. Similar to the proof of Claim 24.

5.2.2 Adversary Type $A_4$

Lemma 39. For any adversary $A$ of Type $A_4$, the advantage $\text{Adv}_{\Pi, A}$ is negligible in the ROM, given that $0.97n \geq 2\kappa$, $q$ is a prime such that $q > 203$ and $q > 8(16\alpha^2n^{3/2} + 2\alpha\sqrt{n})$, and the HNF-RLWE$_{q,x}$ is hard.

The proof of this lemma follows from Claims 40, 41.

Game $G_{4,0}$. Similar to $G_{1,0}$ but now $S$ chooses $\text{sid} = (\Pi, I, P_i^*, P_j^*, x_i^*, (y_j^*, w_j^*))$ as the test session where $P_i^*, P_j^* \leftarrow \{P_1, \ldots, P_N\}$, $s_i^* \leftarrow \{1, \ldots, m\}$, $x_i^*$ is output by $\text{Initiate}(\Pi, I, P_i^*, P_j^*)$ on the $s_i^*$-th session of $P_i^*$. $S$ runs internally $A$ and simulates the oracles as in $G_{1,0}$ except for $\text{Test}$:

- $\text{Test}(\text{sid})$: Let $\text{sid} = (\Pi, I, P_i, P_j, x_i, (y_j, w_j))$ be the test session queried by $A$. If $(P_i, P_j) \neq (P_i^*, P_j^*)$ or $x_i$ is not outputted in the $s_i^*$ session of $P_i^*$, $S$ aborts the execution. Else, it chooses $b \leftarrow \{0, 1\}$ and returns either a random key $sk_i' \leftarrow \{0, 1\}^\kappa$, if $b = 0$, or the session key $sk_i$ of session $\text{sid}$, if $b = 1$.

Claim 40. The probability that $S$ aborts in $G_{4,0}$ is $\frac{1}{mN^2}$.

Game $G_{4,1}$. $S$ simulates the oracles as in $G_{4,0}$, except for $\text{Respond}$:

- When it is queried $\text{Respond}(\Pi, R, P_j, P_i, x_i)$, $S$ does the following: if $P_j = P_j^*$ and it is the $s_j^*$-th session of $P_j^*$, $S$ computes
  $$k_j = p_j(r_j + d) + \bar{x}_i(s_i + r_j + d) + 2h_j.$$
  Else it simulates $\text{Respond}$ as in $G_{3,1}$.

Claim 41. For every adversary $A$, the probability that $A$ distinguishes games $G_{4,0}$ and $G_{4,1}$ is negligible.

Proof. The proof is similar to the proof of Claim 35.

Game $G_{4,2}$. $S$ simulates the oracles as in $G_{4,1}$, except for $\text{Initiate}$ and $\text{Complete}$:

- When it is queried $\text{Complete}(\Pi, I, P_i, P_j, x_i, (y_j, w_j))$, $S$ does the following:
  If $P_i = P_i^*$ and it is the $s_i^*$-th session of $P_i^*$, $S$ computes the key
  $$k_i = p_i(r_i + c) + \bar{y}_j(s_j + r_i + d) + 2h_i.$$
  Else, it simulates as in $G_{4,1}$.

Claim 42. For every adversary $A$, the probability that $A$ distinguishes games $G_{4,1}$ and $G_{4,2}$ is negligible.

Proof. The proof is similar to the proof of Claim 34.
Game $G_{4,3}$. $S$ simulates the oracles as in $G_{4,2}$, except for:

- It replaces $p^*_j$ (the static public key of $P^*_j$) by $v_j \leftarrow R_q$

Claim 43. For every adversary $A$, the probability that $A$ distinguishes games $G_{4,2}$ and $G_{4,3}$ is negligible, given that HNF-RLWE$_{q,\chi}$ is hard.

Proof. The proof is similar to the proof of Claim 36.

Game $G_{4,4}$. $S$ simulates the oracles as in $G_{4,4}$, except for:

- When it is queried Complete$(\Pi, R, P_j, P_i, x_i)$, $S$ does the following: If $(P_i, P_j) = (P^*_i, P^*_j)$ and it is the $s^*_i$-th session of $P^*_i$ and $y_j$ not was outputted by Respond$(\Pi, R, P_j, P_i, x_i)$, $S$ chooses $k_i \leftarrow R_q$. Else, it simulates Complete as in $G_{4,3}$.

Claim 44. For every adversary $A$, the probability that $A$ distinguishes games $G_{4,3}$ and $G_{4,4}$ is negligible, given that HNF-RLWE$_{q,\chi}$ is hard.

Proof. The proof is similar to the proof of Claim 37.

Claim 45. For any adversary $A$, the advantage $\text{Adv}_{\Pi, A}$ in game $G_{4,4}$ is negligible, given that $0.97n \geq 2\kappa$.

Proof. Similar to the proof of Claim 24.

5.2.3 Adversary Type $\mathcal{A}_5$

Lemma 46. For any adversary $A$ of Type $\mathcal{A}_5$, the advantage $\text{Adv}_{\Pi, A}$ is negligible in the ROM, given that $0.97n \geq 2\kappa$, $q$ is a prime such that $q > 203$ and $q > 8 \left(16\kappa^2 n^{3/2} + 2\sqrt{\kappa n}\right)$, and the HNF-RLWE$_{q,\chi}$ is hard.

The proof of the lemma follows the sequence of games $G_{5,0}, \ldots, G_{5,3}$.

Game $G_{5,0}$. Similar to $G_{1,0}$ but now $S$ chooses $sid = (\Pi, R, P^*_i, P^*_j, x^*_i, (y^*_j, w^*_j))$ as the test session where $P^*_i, P^*_j \leftarrow (P_1, \ldots, P_N)$, $s^*_i, s^*_j \leftarrow \{1, \ldots, m\}$, $x^*_i$ is outputted by a query Initiate$(\Pi, I, P^*_i, P^*_j)$ on the $s^*_i$-th session of $P^*_i$ and $y^*_j$ is outputted by Respond$(\Pi, R, P^*_j, P^*_i, x^*_j)$ on the $s^*_j$-th session of $P^*_j$.

Claim 47. The probability that $S$ aborts in $G_{5,0}$ is $\frac{1}{m^2 N^2}$.

Game $G_{5,1}$. $S$ simulates the oracles as in $G_{5,0}$ except for Complete:

- When it is queried on Initiate$(\Pi, I, P_i, P_j)$, $S$ does the following: If $(P_i, P_j) = (P^*_i, P^*_j)$ and it is the $s^*_i$-th session of $P_i$, then $S$ samples $x_i \leftarrow R_q$.

Else, it simulates the oracles as in game $G_{5,0}$.

Claim 48. For every adversary $A$, the probability that $A$ distinguishes games $G_{5,0}$ and $G_{5,1}$ is negligible, given that HNF-RLWE$_{q,\chi}$ is hard.

Proof. The proof is the same as the proof of Claim 21.
Game $G_{5,2}$. $S$ simulates the oracles as in $G_{5,1}$, except for Respond:

- When it is queried \texttt{Complete}(II, I, P_i, P_j, x_i, (y_j, w_j)), $S$ does the following:
  - if $(P_i, P_j) = (P_i^*, P_j^*)$ and it is the $s_i^*$-th session of $P_i^*$ and $(y_j, w_j)$ was not outputted on the $s_j^*$-th session of $P_j^*$, $S$ samples $k_i \leftarrow R_q$.

\textbf{Claim 49.} For every adversary $A$, the probability that $A$ distinguishes games $G_{5,1}$ and $G_{5,2}$ is negligible, given that $\text{HNF-RLWE}_{q,\alpha}$ is hard.

\textit{Proof.} The proof is the same as the proof of Claim 22. \hfill $\square$

Game $G_{5,3}$. $S$ simulates the oracles as in $G_{5,2}$, except for Respond:

- When it is queried \texttt{Complete}(II, I, P_i, P_j, x_i, (y_j, w_j)), $S$ does the following:
  - if $(P_i, P_j) = (P_i^*, P_j^*)$ and it is the $s_j^*$-th session of $P_j^*$, $S$ samples $k_j \leftarrow R_q$.

\textbf{Claim 50.} For every adversary $A$, the probability that $A$ distinguishes games $G_{5,2}$ and $G_{5,3}$ is negligible, given that $\text{HNF-RLWE}_{q,\alpha}$ is hard.

\textit{Proof.} The proof is the same as the proof of Claim 23. \hfill $\square$

\textbf{Claim 51.} For any adversary $A$, the advantage $\text{Adv}_{\text{II},A}^\text{II}$ in game $G_{5,3}$ is negligible, given that $0.97n \geq 2\kappa$.

\textit{Proof.} The proof is the same as the proof of Claim 24. \hfill $\square$

6 Efficiency of the AKE scheme and comparison

\textbf{Communication complexity.} The messages exchanged are $x_i$, carrying $n \log q$ bits of information, and $(y_j, w_j)$ carrying $n + n \log q$ bits of information. Hence, the total number of bits exchange during one execution of the protocol is $n + 2n \log q$.

\textbf{Computational complexity.} First, note that party $P_i$ can perform the multiplication $p_j s_i$ offline and save this value for later use. Hence the scheme requires 6 multiplication in the ring $R_q$, 3 for each party. One execution of the protocol requires to sample 8 times from discrete Gaussian distributions, 4 for each party.

\textbf{Proposed parameters.} For a security of (at least) 128 bits, the proposed parameters of [27] are $n = 512$, $\alpha = 4.19$ and $q = 12033$. These parameters of [27] were estimated based the attacks of [3, 37, 7, 9]. However, we cannot use these parameters in our scheme, since correctness is not guaranteed, by Lemma 16. Hence, we consider $q = 26038273$, which is the minimum prime that satisfies $q \equiv 1 \mod 2n$ and Lemma 16. We require that $q \equiv 1 \mod 2n$ because of efficiency purposes [4, 27]. Note that, by increasing the value of $q$, the security can only increase. For a security level of (at least) 256 bits, we propose the parameters $n = 1024$, $\alpha = 2.6$ and $q = 28434433$. These parameters
were chosen in a similar way as the previous ones. With these parameters, the probability of failure is negligible, according to Lemma 16.

Table 3 presents a comparison with the ZZD+ scheme of the proposed parameters. Table 4 presents a comparison with the ZZD+ scheme of the keys and messages size (expressed in kyloBytes). As we can see, our scheme achieves smaller keys and smaller exchanged messages. As discuss in the introduction, this is due to the fact that our scheme avoids the use of the rejection sampling technique. Since the scheme of ZZD+ requires rejection sampling and several discrete Gaussian distributions (in particular, it is required to use a discrete Gaussian distribution $\chi_{\beta}$ with $\beta >> \alpha$), the value of $q$ needs to be larger than usual. This results in larger keys and exchanged messages.

<table>
<thead>
<tr>
<th></th>
<th>$n$</th>
<th>$\alpha$</th>
<th>$\tau$</th>
<th>$\log \beta$</th>
<th>$q$</th>
<th>Sec. level (bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZZD+ [56]</td>
<td>1024</td>
<td>3.397</td>
<td>12</td>
<td>16.1</td>
<td>$2^{15}$</td>
<td>80</td>
</tr>
<tr>
<td>Ours (I)</td>
<td>512</td>
<td>4.19</td>
<td>-</td>
<td>-</td>
<td>26,038,273</td>
<td>128</td>
</tr>
<tr>
<td>Ours (II)</td>
<td>1024</td>
<td>2.6</td>
<td>-</td>
<td>-</td>
<td>28,434,433</td>
<td>256</td>
</tr>
</tbody>
</table>

Table 3: Comparison of parameters with other AKE schemes with implicit authentication.

<table>
<thead>
<tr>
<th></th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Static)</td>
</tr>
<tr>
<td>ZZD+ [56]</td>
<td>5.625 KB</td>
</tr>
<tr>
<td>Ours (I)</td>
<td>1.577 KB</td>
</tr>
<tr>
<td>Ours (II)</td>
<td>3.153 KB</td>
</tr>
</tbody>
</table>

Table 4: Comparison of size with other AKE schemes with implicit authentication.

7 Conclusion and open problems

We propose an efficient technique that allows for key reuse in lattice-based KE and which we have called pasteurization. We believe that this technique may be of independent interest, as it can be used to sanitize RLWE samples.

We also present a new AKE scheme using the above technique and that surpasses state-of-the-art AKE with implicit authentication in both communication and computational complexity.

Note that our proofs are in the ROM [10]. We leave as future work to present a proof of security for the AKE in the Quantum Random Oracle Model (QROM) [12] as, by now, the lack of proofs techniques makes this problem quite hard. However, we remark that very few constructions are known to be secure in the ROM, but insecure in the QROM [6]. Thus, we strongly believe that the

---

*We also remark that the attack of [6] is quite impractical in real-life.
scheme is secure in the QROM since we are not aware of any quantum attack that breaks the scheme, although we present no proof for this.

Acknowledgments

Pedro Branco thanks the support of DP-PMI and FCT (Portugal) through the grant PD/BD/135181/2017.

References


