# Post-quantum Authenticated Key Exchange from Ideal Lattices

Jiang Zhang[1], Zhenfeng Zhang[1,*], Jintai Ding[2,*], Michael Snook[2], and Özgür Dagdelen[3]

[1] Institute of Software, Chinese Academy of Sciences, China
[2] University of Cincinnati, Cincinnati, USA
[3] Darmstadt University of Technology, Germany
jiangzhang09@gmail.com, zfzhang@tca.iscas.ac.cn, jintai.ding@gmail.com,
snookml@mail.uc.edu, oezguer.dagdelen@cased.de
* Corresponding Authors

**Abstract.** In this paper, we present a practical and provably secure two-pass AKE protocol from ideal lattices, which is conceptually simple and has similarities to the Diffie-Hellman based protocols such as HMQV (CRYPTO 2005) and OAKE (CCS 2013). Our protocol does not rely on other cryptographic primitives—in particular, it does not use signatures—simplifying the protocol and resting the security solely on the hardness of the ring learning with errors problem. The security is proven in the Bellare-Rogaway model with weak perfect forward secrecy. We also give a one-pass variant of our two-pass protocol, which might be appealing in specific applications. Several concrete choices of parameters are provided, and a proof-of-concept implementation shows that our protocols are indeed practical.

## 1 Introduction

Key Exchange (KE) is a fundamental cryptographic primitive, allowing two parties to securely generate a common secret key over an insecure network. Because symmetric cryptographic tools (*e.g.* AES) are reliant on both parties having a shared key in order to securely transmit data, KE is one of the most used cryptographic tools in building secure communication protocols (*e.g.* SSL/TLS, IPSec, SSH). Following the introduction of the Diffie-Hellman (DH) protocol [25], cryptographers have devised a wide selection of KE protocols with various use-cases. One such class is Authenticated Key Exchange (AKE), a class of KE protocols where each party is able to verify the other's identity, so that an adversary cannot impersonate one party in the conversation.

For an AKE protocol, each party has a pair of *static keys*: a *static secret key* and a corresponding *static public key*. The static public key is certified to belong to its owner using a public-key or ID-based infrastructure. For each run of the protocol, the parties involved generate *ephemeral secret keys* and use these to generate *ephemeral public keys* that they exchange. Then all the keys are used along with the transcripts of the session to create a shared *session state*, which is then passed to a *key derivation function* to obtain the final session key. Intuitively, such a protocol is secure if no efficient adversary is able to extract any information about the session key from the publicly exchanged messages. More formally, Bellare and Rogaway [7] introduced an indistinguishability-based security model for AKE, the BR model, which captures key authentication such as *implicit mutual key authentication* and *confidentiality of agreed session keys*. The most prominent alternatives stem from Canetti and Krawczyk [14] and LaMacchia *et al.*[46], that also accounts for scenarios in which the adversary is able to obtain information about a static secret key or a session state other than the state of the target session. In practice, AKE protocols are usually required to have a property, Perfect Forward Secrecy (PFS), that an adversary cannot compromise session keys after a completed session, even if it obtains the parties' static secret keys (*e.g.*, via a heartbleed attack[4]). As shown in [44], no two-pass AKE protocol based on public-key authentication can achieve PFS. Thus, the notion of weak PFS (wPFS) is usually considered for two-pass AKE protocols, which states that the session key of an honestly run session remains private if the static keys are compromised after the session is finished [44].

One approach for achieving authentication in KE protocols is to explicitly authenticate the exchanged messages between the involved parties by using some cryptographic primitives (*e.g.*, signatures, or MAC), which usually incurs additional computation and communication overheads with respect to the basic KE protocol, and complicates the

[4] http://heartbleed.com/

understanding of the KE protocol. This includes several well-known protocols such as IKE [37,42], SIGMA [43], SSL [30], TLS [24,45,54,34,11], as well as the standard in German electronic identity cards, namely EAC [13,21], and the standardized protocols OPACITY [22] and PLAID [23]. Another line of designing AKEs follows the idea of MQV [55,39,44,66] (which has been standardized by ISO/IEC and IEEE, and recommended by NIST and NSA Suite B) by making good use of the algebraic structure of DH problems to achieve implicit authentication. All the above AKEs are based on classic hard problems, such as factoring, the RSA problem, or the computational/decision DH problem. Since these hard problems are vulnerable to quantum computers [62] and as we are moving into the era of quantum computing, it is very appealing to find other counterparts based on problems believed to be resistant to quantum attacks. For instance, post-quantum AKE is considered of high priority by NIST [16]. Due to the potential benefits of lattice-based constructions such as asymptotic efficiency, conceptual simplicity, and worst-case hardness assumptions, it makes perfect sense to build lattice-based AKEs.

### 1.1 Main Contributions

In this paper, we propose an efficient AKE protocol based on the Ring Learning With Errors (Ring-LWE), which in turn is as hard as some lattice problems (*e.g.*, SIVP) in the worst case on ideal lattices [52,28]. Our method avoids introducing extra cryptographic primitives, thus simplifying the design and reducing overhead. In particular, the communicating parties are not required to either encrypt any messages with the other's public key, nor sign any of their own messages during key exchange. Furthermore, by having the key exchange as a self-contained system, we reduce the security assumptions needed, and are able to rely directly and solely on the hardness of Ring-LWE.

By utilizing many useful properties of Ring-LWE problems and discrete Gaussian distributions, we establish an approach to combine both the static and ephemeral public/secret keys, in a manner similar to HMQV [44]. Thus, our protocol not only enjoys many nice properties of HMQV such as two-pass messages, implicit key authentication, high efficiency, and without using any explicit entity authentication techniques (*e.g.*, signatures), but also has many properties of lattice-based cryptography, such as asymptotic efficiency, conceptual simplicity, worst-case hardness assumption, as well as resistance to quantum computer attacks. However, there are also several shortcomings inherited from lattice-based cryptography, such as "handling of noises" and large public/secret keys. Besides, unlike HMQV which works on "nice-behaving" cyclic groups, the security of our protocol cannot be proven in the CK model [14] due to the underlying noise-based algebraic structures. Fortunately, we prove the security in the BR model, which is the most common model considered as it is usually strong enough for many practical applications and it comes with composability [12]. In addition, our protocol achieves weak PFS property, which is known as the best PFS notion achievable by two-pass protocols [44].

As MQV [55] and HMQV [44], we present a one-pass variant of our basic protocol (*i.e.*, the two parties can only exchange a single message in order to derive a shared session key), which might be useful in client-server based applications. Finally, we select concrete choices of parameters and construct a proof-of-concept implementation to examine the efficiency of our protocols. Through the implementation has not undergone any real optimization, the performance results already indicate that our protocols are practical.

We note that none of the techniques we use prevent us from instantiating our AKE protocol based on standard lattices. One just has to keep in mind that key sizes and performance eventually become worse.

### 1.2 Techniques, and Relation to HMQV

Our AKE protocol is inspired by HMQV [44], which makes our protocol share some similarities to HMQV. However, there are also many differences between our protocol and HMQV due to the different underlying algebraic structures. To better illustrate the commons and differences between our AKE protocol and HMQV, we first briefly recall the HMQV protocol [44]. Let $\mathbb{G}$ be a cyclic group with generator $g \in \mathbb{G}$. Let $(P_i = g^{s_i}, s_i)$ and $(P_j = g^{s_j}, s_j)$ be the static public/secret key pairs of party $i$ and party $j$, respectively. During the protocol, both parties exchange ephemeral public keys, *e.g.*, party $i$ sends $X_i = g^{r_i}$ to party $j$, and party $j$ sends $Y_j = g^{r_j}$ to party $i$. Then, both parties compute the same key material $k_i = (P_j^d Y_j)^{s_i c + r_i} = g^{(s_i c + r_i)(s_j d + r_j)} = (P_i^c X_i)^{s_j d + r_j} = k_j$ where $c = H_1(j, X)$ and $d = H_1(i, Y)$ are computed by using a function $H_1$, and use it as input of a key derivation function $H_2$ to generate a common session key, *i.e.*, $\mathrm{sk}_i = H_2(k_i) = H_2(k_j) = \mathrm{sk}_j$.

As mentioned above, HMQV has many nice properties such as only two-pass messages, implicit key authentication, high efficiency, and without using any explicit entity authentication techniques (*e.g.*, signatures). Our main goal is to construct a lattice-based counterpart such that it not only enjoys all those nice properties of HMQV, but also belongs to post-quantum cryptography, *i.e.*, the underlying hardness assumption is believed to hold even against quantum computer. However, such a task is highly non-trivial since the success of HMQV extremely relies on the nice property of cyclic groups such as commutativity (*i.e.,* $(g^a)^b = (g^b)^a$) and perfect (and public) randomization (*i.e.* $g^a$ can be perfectly randomized by computing $g^a g^r$ with a uniformly chosen $r$ at random).

Fortunately, as noticed in [26,59,8], the Ring-LWE problem actually supports some kind of "approximate" commutativity, and can be used to build passive-secure key exchange protocol. Specifically, let $R_q$ be a ring, and $\chi$ be a Gaussian distribution over $R_q$. Then, given two Ring-LWE tuples with both secret and errors choosing from $\chi$, *e.g.,* $(a, b_1 = as_1 + e_1)$ and $(a, b_2 = as_2 + e_2)$ for randomly chosen $a \leftarrow_r R_q, s_1, s_2, e_1, e_2 \leftarrow_r \chi$, the approximate equation $s_1 b_2 \approx s_1 a s_2 \approx s_2 b_1$ holds with overwhelming probability for proper parameters. By the same observation, we construct an AKE protocol (as illustrated in Fig. 1), where both the static and ephemeral public keys are actually Ring-LWE elements corresponding to a globally public element $a \in R_q$. In order to overcome the inability of "approximate" commutativity, our protocol has to send a signal information $w_j$ computed by using a function $\mathsf{Cha}$. Combining this with another useful function $\mathsf{Mod}_2$, both parties are able to compute the same key material $\sigma_i = \sigma_j$ (from the approximately equal values $k_i$ and $k_j$) with a guarantee that $\sigma_j = \mathsf{Mod}_2(k_j, w_j)$ has high min-entropy even conditioned on the partial information $w_j = \mathsf{Cha}(k_j)$ of $k_j$ (thus it can be used to derive a uniform session key $\mathrm{sk}_j$).

| Party $i$ | | Party $j$ |
|---|---|---|
| Public Key: $p_i = as_i + 2e_i \in R_q$ | | Public Key: $p_j = as_j + 2e_j \in R_q$ |
| Secret Key: $s_i \in R_q$ | | Secret Key: $s_j \in R_q$ |
| where $s_i, e_i \leftarrow_r \chi_\alpha$ | | where $s_j, e_j \leftarrow_r \chi_\alpha$ |
| $x_i = ar_i + 2f_i \in R_q$ | $\xrightarrow{x_i}$ | $y_j = ar_j + 2f_j \in R_q$ |
| where $r_i, f_i \leftarrow_r \chi_\beta$ | | $k_j = (p_i c + x_i)(s_j d + r_j) + 2cg_j$ |
| | | where $r_j, f_j, g_j \leftarrow_r \chi_\beta$ |
| $k_i = (p_j d + y_j)(s_i c + r_i) + 2dg_i$ | $\xleftarrow{y_j, w_j}$ | $w_j = \mathsf{Cha}(k_j) \in \{0,1\}^n$ |
| where $g_i \leftarrow_r \chi_\beta$ | | $\sigma_j = \mathsf{Mod}_2(k_j, w_j) \in \{0,1\}^n$ |
| $\sigma_i = \mathsf{Mod}_2(k_i, w_j) \in \{0,1\}^n$ | | $\mathrm{sk}_j = H_2(i, j, x_i, y_j, w_j, \sigma_j)$ |
| $\mathrm{sk}_i = H_2(i, j, x_i, y_j, w_j, \sigma_i)$ | | |

$$c = H_1(i, j, x_i) \in R, d = H_1(j, i, y_j, x_i) \in R$$

**Fig. 1.** Our AKE protocol based on Ring-LWE, where $R_q = \mathbb{Z}_q/(x^n + 1)$ is a ring, $\chi_\alpha$ and $\chi_\beta$ are two Gaussian distributions over $R_q$. The two functions $\mathsf{Cha}$ and $\mathsf{Mod}_2$ provide that $\sigma_i = \mathsf{Mod}_2(k_i, w_j) = \mathsf{Mod}_2(k_j, w_j) = \sigma_j$.

However, the strategy of sending out the information $w_j = \mathsf{Cha}(k_j)$ inherently brings an undesired byproduct. Specifically, unlike HMQV, the security of our AKE protocol cannot be proven in the CK model which allows the adversaries to obtain the session state $k_j$ via *session state reveal queries*. This is because in a traditional definition of session identifier that consists of all the exchanged messages, the two "different" sessions $\mathrm{sid} = (i, j, x_i, y_j, w_j)$ and $\mathrm{sid}' = (i, j, x_i, y_j, w_j')$ in our protocol have the same session state, *i.e.,* $k_i$ at party $i$.[5] This also means that we cannot directly use $\sigma_i = \sigma_j$ as the session key, because the binding between the value of $\sigma_i$ and the session identifier is too loose (especially for the signal part, $w_j$'s). Since both sessions $\mathrm{sid}$ and $\mathrm{sid}'$ have the same session state $k_i$, the value $\sigma_i' = \mathsf{Mod}_2(k_i, w_j')$ corresponding to $\mathrm{sid}'$ is simply a shift of $\sigma_i = \mathsf{Mod}_2(k_i, w_j)$ corresponding to $\mathrm{sid}$ (by the definition of the $\mathsf{Mod}_2$ function). We prevent the adversary from utilizing this weakness by setting the session key as the output of the hash function $H_2$ (which is modeled as a random oracle) which tightly binds the session identifier $\mathrm{sid}$ and the key material $\sigma_i$ (*i.e.*, $\mathrm{sk}_i = H_2(\mathrm{sid}, \sigma_i)$). Our technique works due to another useful property of $\mathsf{Mod}_2$, which

[5] We remark that this problem might not exist if we consider a different definition of session identifier, *e.g.*, the one that was uniquely determined at the beginning of each execution of the protocol.

guarantees that $\sigma'_i = \mathsf{Mod}_2(k_i, w'_j)$ preserves the high min-entropy property of $k_i$ for any $w'_j$ (and thus is enough to generate a secure session key by the property of random oracle $H_2$).[6]

In order to finally get a security proof of our AKE protocol in the BR model with weakly perfect forward secrecy, we have to make use of the following property of Gaussian distributions namely some kind of "public randomization". Specifically, let $\chi_\alpha$ and $\chi_\beta$ be two Gaussian distributions with standard deviation $\alpha$ and $\beta$, respectively. Then, the summation of the two distributions is still a Gaussian distribution $\chi_\gamma$ with standard deviation $\gamma = \sqrt{\alpha^2 + \beta^2}$. In particular, if $\beta \gg \alpha$ (*e.g.*, $\beta/\alpha = 2^{\omega(\log \kappa)}$ for some security parameter $\kappa$), we have that the distribution $\chi_\gamma$ is statistically close to $\chi_\beta$. This technique is also known as "noise flooding" and has been applied, for instance, in proving robustness of the LWE assumption [35]. [7] Using this technique allows to statistically hide the distribution of $\chi_\alpha$ in a bigger distribution $\chi_\beta$. The security proof of our protocol is based on this observation, and for now let us keep it in mind that a large distribution will be used to hide a small one.

To better illustrate our technique, we take party $j$ as an example, who combines his static and ephemeral secret keys by computing $\hat{r}_j = s_j d + r_j$ where $d = H_1(j, i, y_j, x_i)$. We notice that the value $\hat{r}_j$ actually behaves like a "signature" on the messages that party $j$ knows so far. In other words, it should be difficult to compute $\hat{r}_j$ if we do not know the corresponding "signing key" $s_j$. Indeed, this combination is necessary to provide the implicit entity authentication. However, it also posts an obstacle to get a security proof since the simulator may also be unaware of $s_j$. Fortunately, if the randomness $r_j$ is chosen from a big enough Gaussian distribution, then the value $\hat{r}_j$ almost obliterates all information of $s_j$. More specifically, the simulator can directly choose $\hat{r}_j$ such that $\hat{r}_j = s_j d + r_j$ for some unknown $r_j$ by computing $y_j = (a\hat{r}_j + 2\hat{f}_j) - p_j d$, and programming the random oracle $d = H_1(j, i, y_j, x_i)$ correspondingly. Combining the properties of Gaussian distributions and the random oracle $H_1$, we have that $y_j$ is almost identically distributed as that in the real run of the protocol. Now, we check the randomness of $k_j = (p_i c + x_i)\hat{r}_j + 2cg_j$. Note that for the test session, we can always guarantee that at least one of $p_i$ and $x_i$ is honestly generated (and thus is computationally indistinguishable from uniformly distributed element under the Ring-LWE assumption), or else there is no "secrecy" to protect at all if both $p_i$ and $x_i$ are chosen by the adversary. That is, the value $p_i c + x_i$ is always uniformly distributed if $c$ is invertible in $R_q$. Again, by programming $c = H_1(i, j, x_i)$, the simulator can actually replace $p_i c + x_i$ with $\hat{x}_i = c^{-1} u_i$ for a uniformly distributed ring element $u_i$. In this case, we have that $k_j = \hat{x}_i \hat{r}_j + 2cg_j = c(u_i \hat{r}_j + 2g_j)$ should be computationally indistinguishable from a uniformly distributed element under the Ring-LWE assumption. In other words, $k_j$ can be used to derive a high min-entropy key material $\sigma_j$ as required by using the $\mathsf{Mod}_2$ function.

Unfortunately, directly using "noise flooding" has a significant drawback, *i.e.*, the requirement of a super-polynomially big standard deviation $\beta$, which may lead to a nightmare for practical performance due to a super-polynomially big modulus $q$ for correctness and a very large ring dimension $n$ for the hardness of the underlying Ring-LWE problems. Fortunately, we can somehow reduce the big cost by further employing the rejection sampling technique [50]. Rejection sampling is a crucial technique in signature schemes to make the distribution of signatures independent of the signing key. Since [50] it has been applied in many other lattice-based signature schemes [36,29,3,38].

In our case the combination of the static and ephemeral secret keys, $\hat{r}_j = s_j d + r_j$, at party $j$ is essentially a signature on all the public messages under party $j$'s public key (we again take party $j$ as an example, but note that similar analysis also holds for party $i$). Thus, we can freely use the rejection sampling technique to relax the requirement on a super-polynomially big $\beta$. In other words, we can use a much smaller $\beta$, but require party $j$ to use $r_j$ if $\hat{r}_j = s_j d + r_j$ follows the distribution $\chi_\beta$, and to resample a new $r_j$ otherwise. We note that by deploying rejection sampling in our AKE it is the first time that rejection sampling is used beyond signature schemes. As for signatures, rejection sampling is done locally, and thus will not affect the interaction between the two parties, *i.e.*, two-pass messages. Even though the computational performance of each execution might become worse with certain (small) probability (due to rejection and repeated sampling), the average computational cost is much better than the setting of using a super-polynomially big $\beta$.

[6] We remark that this is also the reason why the nice reconciliation mechanism in [59] cannot be used in our protocol. Specifically, it is unclear whether the reconciliation function $\mathsf{rec}(\cdot,\cdot)$ in [59] could also preserve the high min-entropy property of the first input (*i.e.*, which might not be uniformly random) for any (maliciously chosen) second input.

[7] Actually, noise flooding works conditioned on the size of the random variable, and thus does not require to be distributed according to $\chi_\alpha$.

### 1.3 Related Work, Comparison and Discussion

In the past few years, many cryptographers have put effort into constructing different kinds of KE protocols from lattices. At Asiacrypt 2009, Katz and Vaikuntanathan [41] proposed the first password-based authenticated key exchange protocol that can be proven secure based on the LWE assumption. Ding *et al.* [26] proposed a passive-secure KE protocol based on (Ring-)LWE. Like the standard DH protocol, the protocol in [26] could not provide authentication—*i.e.*, it is not an AKE protocol—and is thus weak to man-in-the-middle attacks. Lei *et al.* [47] presented a KE protocol based on NTRU encryption and a new "NTRU-KE" assumption.

**Table 1.** Comparison of Lattice-based AKEs (CCA$^*$ means CCA-security with high min-entropy keys [31], and EUF-CMA means existential unforgeability under chosen message attacks)

| Protocols | KEM/PKE | Signature | Message-pass | Model | RO? | Num. of $R_q$ |
|---|---|---|---|---|---|---|
| FSXY12 [31] | CCA$^*$ | - | 2-pass | CK | $\times$ | $\gg 7$ |
| FSXY13 [32] | OW-CCA | - | 2-pass | CK | $\surd$ | 7 |
| Peikert14 [59] | CPA | EUF-CMA | 3-pass | SK-security | $\surd$ | $> 2$ [a] |
| BCNS14 [8] | CPA | EUF-CMA | 4-pass | ACCE | $\surd$ | 2 for KEM [b] |
| Ours | - | - | 2-pass | BR with wPFS | $\surd$ | 2 |

[a] The actual number of ring elements depends on the choice of the concrete lattice-based signatures.

[b] Since the protocol uses traditional signatures to provide authentication, it does not contain any other ring elements.

To the best of our knowledge, there are four papers focusing on designing AKEs from lattices [31,59,32,8]. In general, all known lattice-based AKE protocols work by following generic transformations from key encapsulation mechanisms (KEM) to AKEs and explicitly using signatures to provide authentication. Fujioka *et al.* [31] proposed a generic construction of AKE from KEMs, which can be proven secure in the CK model. Informally, they showed that if there is a CCA secure KEM with high min-entropy keys and a family of pseudorandom functions (PRF), then there is a secure AKE protocol in the standard model. Instantiated with lattice-based CCA secure KEMs such as [60,57], it is possible to construct lattice-based AKE protocols in the standard model. However, as the authors commented, their construction was just of theoretic interest due to huge public keys and the lack of an efficient and direct construction of PRFs from (Ring-)LWE. Following [31], the paper [32] tried to get a practical AKE protocol, and gave a generic construction from any one-way CCA-secure KEM in the random oracle model. The two protocols in [31,32] share some similarities such as having two-pass messages, and involving three times encryptions (*i.e.*, two encryptions under each party's static public keys and one encryption under an ephemeral public key). For concreteness, instantiated with the CPA-secure encryption from Ring-LWE [52] (*i.e.*, by first transforming it into a CCA-secure one using the Fujisaki-Okamoto (FO) transformation in the random oracle model), the protocol in [32] requires to exchange seven ring elements in total.

Recently, Peikert [59] presented an efficient KEM based on Ring-LWE, which was then transformed into an AKE protocol by using the same structure as SIGMA [43]. The resulting protocol involved one encryption, and two signatures and two MACs for explicit entity authentication. As the SIGMA protocol, the protocol in [59] has three-pass messages and was proven SK-secure [15] in the random oracle model. Bos *et al.* [8] treated Peikert's KEM as a DH-like KE protocol, and integrated it into the Transport Layer Security (TLS) protocol. Thus, their AKE protocol also employed signatures to provide explicit authentication. In fact, they used the traditional digital signatures such as RSA and ECDSA to provide authentication (*i.e.,* it is not a pure post-quantum AKE protocol). The security of their protocol was proven in the authenticated and confidential channel establishment (ACCE) security model [40], which is based on the BR model, but has many differences to capture entity authentication and channel security.

Since the lack of concrete security analysis and parameter choices in the literature, we only give a theoretical comparison of lattice-based AKEs in Table 1. In summary, our protocol only has two-pass messages (about two ring elements) and does not use signatures/MACs at al, and its security solely relies on the hardness of Ring-LWE. To the best of our knowledge there is not a single post-quantum authenticated key exchange protocol (until this work) which solely relies on a quantum-hard computational problem and does not make use of explicit cryptographic primitives except hash functions.

### 1.4 Roadmap

In the preliminaries section, we recall the BR model and several useful tools on lattices. Then, we give a two-pass AKE protocol from ideal lattices in Section 3, and prove its security based on Ring-LWE problems in Section 4. In Section 5, we present the one-pass variant of our protocol. The concrete choices of parameters and timings are given in Section 6.

## 2 Preliminaries

### 2.1 Notation

Let $\kappa$ be the natural security parameter, and all quantities are implicitly dependent on $\kappa$. Let $\mathrm{poly}(\kappa)$ denote an unspecified function $f(\kappa) = O(\kappa^c)$ for some constant $c$. The function $\log$ denotes the natural logarithm. We use standard notation $O, \omega$ to classify the growth of functions. If $f(\kappa) = O(g(\kappa) \cdot \log^c \kappa)$, we denote $f(\kappa) = \tilde{O}(g(\kappa))$. We say a function $f(\kappa)$ is negligible if for every $c > 0$, there exists a $N$ such that $f(\kappa) < 1/\kappa^c$ for all $\kappa > N$. We use $\mathrm{negl}(\kappa)$ to denote a negligible function of $\kappa$, and we say a probability is overwhelming if it is $1 - \mathrm{negl}(\kappa)$.

The set of real numbers (integers) is denoted by $\mathbb{R}$ ($\mathbb{Z}$, resp.). We use $\leftarrow_r$ to denote randomly choosing an element from some distribution (or the uniform distribution over some finite set). Vectors are in column form and denoted by bold lower-case letters (*e.g.*, $\mathbf{x}$). The $\ell_2$ and $\ell_\infty$ norms we designate by $\|\cdot\|$ and $\|\cdot\|_\infty$. The ring of polynomials over $\mathbb{Z}$ ($\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$, resp.) we denote by $\mathbb{Z}[x]$ ($\mathbb{Z}_q[x]$, resp.).

Let $X$ be a distribution over finite set $S$. The min-entropy of $X$ is defined as

$$H_\infty(X) = -\log(\max_{s \in S} \Pr[X = s]).$$

Intuitively, the min-entropy says that if we (privately) choose $x$ from $X$ at random, then no (unbounded) algorithm can guess the value of $x$ correctly with probability greater than $2^{-H_\infty(X)}$.

### 2.2 Security Model for AKE

We now recall the Bellare-Rogaway security model [7], restricted to the case of two-pass AKE protocol.

*Sessions* We fix a positive integer $N$ to be the maximum number of honest parties that use the AKE protocol. Each party is uniquely identified by an integer $i$ in $\{1, 2, \ldots, N\}$, and has a static key pair consisting of a static secret key $\mathrm{sk}_i$ and static public key $\mathrm{pk}_i$, which is signed by a Certificate Authority (CA). A single run of the protocol is called a *session*. A session is activated at a party by an incoming message of the form $(\Pi, I, i, j)$ or the form $(\Pi, R, j, i, X_i)$, where $\Pi$ is a protocol identifier; $I$ and $R$ are role identifiers; $i$ and $j$ are party identifiers. If party $i$ receives a message of the form $(\Pi, I, i, j)$, we say that $i$ is the session initiator. Party $i$ then outputs the response $X_i$ intended for party $j$. If party $j$ receives a message of the form $(\Pi, R, j, i, X_i)$, we say that $j$ is the session responder; party $j$ then outputs a response $Y_j$ to party $i$. After exchanging these messages, both parties compute a session key.

If a session is activated at party $i$ with $i$ being the initiator, we associate with it a *session identifier* $\mathrm{sid} = (\Pi, I, i, j, X_i)$ or $\mathrm{sid} = (\Pi, I, i, j, X_i, Y_j)$. Similarly, if a session is activated at party $j$ with $j$ being the responder, the session identifier has the form $\mathrm{sid} = (\Pi, R, j, i, X_i, Y_j)$. For a session identifier $\mathrm{sid} = (\Pi, *, i, j, *[, *])$, the third coordinate—that is, the first party identifier—is called the owner of the session; the other party is called the peer of the session. A session is said to be *completed* when its owner computes a session key. The *matching session* of $\mathrm{sid} = (\Pi, I, i, j, X_i, Y_j)$ is the session with identifier $\widetilde{\mathrm{sid}} = (\Pi, R, j, i, X_i, Y_j)$ and vice versa.

*Adversarial Capabilities* We model the adversary $\mathcal{A}$ as a probabilistic polynomial time (PPT) Turing machine with full control over all communications channels between parties, including control over session activations. In particular, $\mathcal{A}$ can intercept all messages, read them all, and remove or modify any desired messages as well as inject its own messages. We also suppose $\mathcal{A}$ is capable of obtaining hidden information about the parties, including static secret keys and session keys to model potential leakage of them in genuine protocol executions. These abilities are formalized by providing $\mathcal{A}$ with the following oracles (we split the $\mathsf{Send}$ query in [14] into $\mathsf{Send}_0$, $\mathsf{Send}_1$ and $\mathsf{Send}_2$ queries for the case of two-pass protocols):

- $\mathsf{Send}_0(\Pi, I, i, j)$: $\mathcal{A}$ activates party $i$ as an initiator. The oracle returns a message $X_i$ intended for party $j$.
- $\mathsf{Send}_1(\Pi, R, j, i, X_i)$: $\mathcal{A}$ activates party $j$ as a responder using message $X_i$. The oracle returns a message $Y_j$ intended for party $i$.
- $\mathsf{Send}_2(\Pi, R, i, j, X_i, Y_j)$: $\mathcal{A}$ sends party $i$ the message $Y_j$ to complete a session previously activated with a $\mathsf{Send}_0(\Pi, I, i, j)$ query that returned $X_i$.
- $\mathsf{SessionKeyReveal}(\mathrm{sid})$: The oracle returns the session key associated with the session $\mathrm{sid}$ if it has been generated.
- $\mathsf{Corrupt}(i)$: The oracle returns the static secret key belonging to party $i$. A party whose key is given to $\mathcal{A}$ in this way is called *dishonest*; a party not compromised in this way is called *honest*.
- $\mathsf{Test}(\mathrm{sid}^*)$: The oracle chooses a bit $b \leftarrow_r \{0,1\}$. If $b = 0$, it returns a key chosen uniformly at random; if $b = 1$, it returns the session key associated with $\mathrm{sid}^*$. Note that we impose some restrictions on this query. We only allow $\mathcal{A}$ to query this oracle once, and only on a fresh (see Definition 1) session $\mathrm{sid}^*$.

**Definition 1 (Freshness).** *Let* $\mathrm{sid}^* = (\Pi, I, i^*, j^*, X_i, Y_j)$ *or* $(\Pi, R, j^*, i^*, X_i, Y_j)$ *be a completed session with initiator party* $i^*$ *and responder party* $j^*$*. If the matching session exists, denote it* $\widetilde{\mathrm{sid}}^*$*. We say that* $\mathrm{sid}^*$ *is fresh if the following conditions all hold:*

- *$\mathcal{A}$ has not made a $\mathsf{SessionKeyReveal}$ query on $\mathrm{sid}^*$.*
- *$\mathcal{A}$ has not made a $\mathsf{SessionKeyReveal}$ query on $\widetilde{\mathrm{sid}}^*$ (if it exists).*
- *Neither party $i^*$ nor $j^*$ is dishonest if $\widetilde{\mathrm{sid}}^*$ does not exist. I.e., $\mathcal{A}$ has not made a $\mathsf{Corrupt}$ query on either of them.*

Recall that in the original BR model [7], no corruption query is allowed. In the above freshness definition, we allow the adversary to corrupt both parties of $\mathrm{sid}^*$ if the matching session exists, *i.e.*, the adversary can obtain the parties's secret key in advance and then passively eavesdrops the session $\mathrm{sid}^*$ (and thus $\widetilde{\mathrm{sid}}^*$). We remark that this is actually stronger than what is needed for capturing wPFS [44], where the adversary is only allowed to corrupt a party after an honest session $\mathrm{sid}^*$ (and thus $\widetilde{\mathrm{sid}}^*$) has been completed.

*Security Game* The security of a two-pass AKE protocol is defined in terms of the following game. The adversary $\mathcal{A}$ makes any sequence of queries to the oracles above, so long as only one $\mathsf{Test}$ query is made on a fresh session, as mentioned above. The game ends when $\mathcal{A}$ outputs a guess $b'$ for $b$. We say $\mathcal{A}$ wins the game if its guess is correct, so that $b' = b$. The advantage of $\mathcal{A}$, $\mathbf{Adv}_{\Pi,\mathcal{A}}$, is defined as $\Pr[b' = b] - 1/2$.

**Definition 2 (Security).** *We say that an AKE protocol $\Pi$ is secure if the following conditions hold:*

- *If two honest parties complete matching sessions then they compute the same session key with overwhelming probability.*
- *For any PPT adversary $\mathcal{A}$, the advantage $\mathbf{Adv}_{\Pi,\mathcal{A}}$ is negligible.*

### 2.3 The Gaussian Distributions and Rejection Sampling

For any positive real $\alpha \in \mathbb{R}$, and vectors $\mathbf{c} \in \mathbb{R}^m$, the continuous Gaussian distribution over $\mathbb{R}^m$ with standard deviation $\alpha$ centered at $\mathbf{v}$ is defined by the probability function $\rho_{\alpha,\mathbf{c}}(\mathbf{x}) = (\frac{1}{\sqrt{2\pi\sigma^2}})^m \exp\left(\frac{-\|\mathbf{x}-\mathbf{v}\|^2}{2\sigma^2}\right)$. For integer vectors $\mathbf{c} \in \mathbb{R}^n$, let $\rho_{s,\mathbf{c}}(\mathbb{Z}^m) = \sum_{\mathbf{x}\in\mathbb{Z}^m} \rho_{s,\mathbf{c}}(\mathbf{x})$. Then, we define the discrete Gaussian distribution over $\mathbb{Z}^m$ as $D_{\mathbb{Z}^m,s,\mathbf{c}}(\mathbf{x}) = \frac{\rho_{s,\mathbf{c}}(\mathbf{x})}{\rho_{s,\mathbf{c}}(\mathbb{Z}^m)}$, where $\mathbf{x} \in \mathbb{Z}^m$. The subscripts $s$ and $\mathbf{c}$ are taken to be $1$ and $\mathbf{0}$ (respectively) when omitted. The following lemma says that for large enough $\alpha$, almost all the samples from $D_{\mathbb{Z}^m,\alpha}$ are small.

**Lemma 1** **([56]).** *Letting real* $\alpha = \omega(\sqrt{\log m})$*, constant* $d > 1/\sqrt{2\pi}$*, then* $\Pr_{\mathbf{x} \leftarrow_r D_{\mathbb{Z}^m,\alpha}}[\|\mathbf{x}\| > d \cdot \alpha\sqrt{m}] \leq \frac{1}{2} D^n$*, where* $D = d\sqrt{2\pi e} \cdot e^{-\pi \cdot d^2}$*. In particular, we have* $\Pr_{\mathbf{x} \leftarrow_r D_{\mathbb{Z}^m,\alpha}}[\|\mathbf{x}\| > \alpha\sqrt{m}] \leq 2^{-m+1}$.

Now, we recall rejection sampling in Theorem 1 from [50], which will be used in the security proof of our AKE protocol. Informally, the rejection sampling theorem says that for large enough $\alpha$, the distributions $D_{\mathbb{Z}^m,\alpha,\mathbf{c}}$ and $D_{\mathbb{Z}^m,\alpha}$ are statistically indistinguishable even given vector $\mathbf{c} \in \mathbb{Z}$.

**Theorem 1** **(Rejection Sampling [50]).** *Let* $V$ *be a subset of* $\mathbb{Z}^m$ *in which all the elements have norms less than* $T$*,* $\alpha = \omega(T\sqrt{\log m})$ *be a real, and* $\psi : V \to \mathbb{R}$ *be a probability distribution. Then there exists a constant* $M = O(1)$ *such that the distribution of the following algorithm* $\mathsf{Samp}_1$ :

*1:* $\mathbf{c} \leftarrow_r \psi$
*2:* $\mathbf{z} \leftarrow_r D_{\mathbb{Z}^m,\alpha,\mathbf{c}}$
*3: output* $(\mathbf{z}, \mathbf{c})$ *with probability* $\min\left(\frac{D_{\mathbb{Z}^m,\alpha}(\mathbf{z})}{M D_{\mathbb{Z}^m,\alpha,\mathbf{c}}(\mathbf{z})}, 1\right)$.

*is within statistical distance* $\frac{2^{-\omega(\log m)}}{M}$ *of the distribution of the following algorithm* $\mathsf{Samp}_2$ :

*1:* $\mathbf{c} \leftarrow_r \psi$
*2:* $\mathbf{z} \leftarrow_r D_{\mathbb{Z}^m,\alpha}$
*3: output* $(\mathbf{z}, \mathbf{c})$ *with probability* $1/M$.

*Moreover, the probability that* $\mathsf{Samp}_1$ *outputs something is at least* $\frac{1-2^{-\omega(\log m)}}{M}$*. More concretely, if* $\alpha = \tau T$ *for any positive* $\tau$*, then* $M = e^{12/\tau + 1/(2\tau^2)}$ *and the output of algorithm* $\mathsf{Samp}_1$ *is within statistical distance* $\frac{2^{-100}}{M}$ *of the output of* $\mathsf{Samp}_2$*, and the probability that* $\mathcal{A}$ *outputs something is at least* $\frac{1-2^{-100}}{M}$.

### 2.4 Ring Learning with Errors

Let the integer $n$ be a power of 2, and consider the ring $R = \mathbb{Z}[x]/(x^n + 1)$. For any positive integer $q$, we define the ring $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ analogously. For any polynomial $y(x)$ in $R$ (or $R_q$), we identify $y$ with its coefficient vector in $\mathbb{Z}^n$ (or $\mathbb{Z}_q^n$). Then we define the norm of a polynomial to be the norm of its coefficient vector.

**Lemma 2.** *For any* $s, t \in R$*, we have* $\|s \cdot t\| \leq \sqrt{n} \cdot \|s\| \cdot \|t\|$ *and* $\|s \cdot t\|_\infty \leq n \cdot \|s\|_\infty \cdot \|t\|_\infty$.

Besides, the discrete Gaussian distribution over the ring $R$ can be naturally defined as the distribution of ring elements whose coefficient vectors are distributed according to the discrete Gaussian distribution over $\mathbb{Z}^n$, *e.g.*, $D_{\mathbb{Z}^n,\alpha}$ for some positive real $\alpha$. Letting $\chi_\alpha$ be the discrete Gaussian distribution over $\mathbb{Z}^n$ with standard deviation $\alpha$ centered at $\mathbf{0}$, *i.e.*, $\chi_\alpha := D_{\mathbb{Z}^n,\alpha}$, we now adopt the following notational convention: since bold-face letters denote vectors, $\mathbf{x} \leftarrow_r \chi_\alpha$ means we sample the vector $\mathbf{x}$ from the distribution $\chi_\alpha$; for normal weight variables (*e.g.* $y \leftarrow_r \chi_\alpha$) we sample an element of $R$ whose coefficient vector is distributed according to $\chi_\alpha$.

Now we come to the statement of the Ring-LWE assumption; we will use a special case detailed in [52]. Let $R_q$ be defined as above, and $s \leftarrow_r R_q$. We define $A_{s,\chi_\alpha}$ to be the distribution of the pair $(a, as + x) \in R_q \times R_q$, where $a \leftarrow_r R_q$ is uniformly chosen and $x \leftarrow_r \chi_\alpha$ is independent of $a$.

**Definition 3** **(Ring-LWE Assumption).** *Let* $R_q$ *and* $\chi_\alpha$ *be defined as above, and let* $s \leftarrow_r R_q$*. The Ring-LWE assumption RLWE*$_{q,\alpha}$ *states that it is hard for any PPT algorithm to distinguish* $A_{s,\chi_\alpha}$ *from the uniform distribution on* $R_q \times R_q$ *with only polynomially many samples.*

The following lemma says that the hardness of the Ring-LWE assumption can be reduced to some hard lattice problems such as the Shortest Independet Vectors Problem (SIVP) over ideal lattices.

**Proposition 1** **(A special case of [52]).** *Let* $n$ *be a power of* $2$*, let* $\alpha$ *be a real number in* $(0, 1)$*, and* $q$ *a prime such that* $q \bmod 2n = 1$ *and* $\alpha q > \omega(\sqrt{\log n})$*. Define* $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ *as above. Then there exists a polynomial time quantum reduction from* $\tilde{O}(\sqrt{n}/\alpha)$*-SIVP in the worst case to average-case RLWE*$_{q,\beta}$ *with* $\ell$ *samples, where* $\beta = \alpha q \cdot (n\ell/\log(n\ell))^{1/4}$.

It has been proven that the Ring-LWE assumption still holds even if the secret $s$ is chosen according to the error distribution $\chi_\beta$ rather than uniformly [1,52]. This variant is known as the *normal form*, and is preferable for controlling the size of the error term [10,9]. The underlying Ring-LWE assumption also holds when scaling the error by a constant $t$ relatively prime to $q$ [10], *i.e.*, using the pair $(a_i, a_i s+tx_i)$ rather than $(a_i, a_i s+x_i)$. Several lattice-based cryptographic schemes have been constructed based on this variant [10,9]. In our case, we will fix $t = 2$. Besides, recall that the RLWE$_{q,\beta}$ assumption guarantees that for some prior fixed (but randomly chosen) $s$, the tuple $(a, as + 2x)$ is computationally indistinguishable from the uniform distribution over $R_q \times R_q$ if $a \leftarrow_r R_q$ and $x \leftarrow \chi_\beta$. In this paper, we will use a matrix form ring-LWE assumption. Formally, let $B_{\chi_\beta,\ell_1,\ell_2}$ be the distribution of $(\mathbf{a}, \mathbf{B} = (b_{i,j})) \in R_q^{\ell_1} \times R_q^{\ell_1\times\ell_2}$, where $\mathbf{a} = (a_0, \dots, a_{\ell_1-1}) \leftarrow_r R_q^{\ell_1}$, $\mathbf{s} = (s_0, \dots, s_{\ell_2-1}) \leftarrow_r R_q^{\ell_2}$, $e_{i,j} \leftarrow_r \chi_\beta$, and $b_{i,j} = a_i s_j + 2e_{i,j}$ for $i \in \{0, \dots, \ell_1-1\}$ and $j \in \{0, \dots, \ell_2-1\}$. For polynomially bounded $\ell_1$ and $\ell_2$, one can show that the distribution of $B_{\chi_\beta,\ell_1,\ell_2}$ is pseudorandom based on the RLWE$_{q,\beta}$ assumption [60].

## 3 Authenticated Key Exchange from Ring-LWE

We now introduce some notation before presenting our protocol. For odd prime $q > 2$, denote $\mathbb{Z}_q = \{-\frac{q-1}{2}, \dots, \frac{q-1}{2}\}$ and define the subset $E := \{-\lfloor\frac{q}{4}\rfloor, \dots, \lfloor\frac{q}{4}\rceil\}$ as the middle half of $\mathbb{Z}_q$. We also define $\mathsf{Cha}$ to be the characteristic function of *the complement of* $E$, so $\mathsf{Cha}(v) = 0$ if $v \in E$ and 1 otherwise. Obviously, for any $v$ in $\mathbb{Z}_q$, $v + \mathsf{Cha}(v) \cdot \frac{q-1}{2} \bmod q$ belongs to $E$. We define an auxiliary modular function, $\mathsf{Mod}_2\colon \mathbb{Z}_q \times \{0,1\} \to \{0,1\}$:

$$\mathsf{Mod}_2(v, b) = (v + b \cdot \frac{q-1}{2}) \bmod q \bmod 2.$$

In the following lemma, we show that given the bit $b = \mathsf{Cha}(v)$, and a value $w = v + 2e$ with sufficiently small $e$, we can recover $\mathsf{Mod}_2(v, \mathsf{Cha}(v))$. In particular, we have $\mathsf{Mod}_2(v, b) = \mathsf{Mod}_2(w, b)$.

**Lemma 3.** *Let $q$ be an odd prime, $v \in \mathbb{Z}_q$ and $e \in \mathbb{Z}_q$ such that $|e| < q/8$. Then, for $w = v + 2e$, we have $\mathsf{Mod}_2(v, \mathsf{Cha}(v)) = \mathsf{Mod}_2(w, \mathsf{Cha}(v))$.*

*Proof.* Note that $w + \mathsf{Cha}(v)\frac{q-1}{2} \bmod q = v + \mathsf{Cha}(v)\frac{q-1}{2} + 2e \bmod q$. Now, $v + \mathsf{Cha}(v)\frac{q-1}{2} \bmod q$ is in $E$ as we stated above; that is, $-\lfloor\frac{q}{4}\rfloor \le v + \mathsf{Cha}(v)\frac{q-1}{2} \bmod q \le \lfloor\frac{q}{4}\rceil$. Thus, since $-q/8 < e < q/8$, we have $-\lfloor\frac{q}{2}\rfloor \le v+\mathsf{Cha}(v)\frac{q-1}{2} \bmod q+2e \le \lfloor\frac{q}{2}\rceil$. Therefore, we have $v+\mathsf{Cha}(v)\frac{q-1}{2} \bmod q+2e = v+\mathsf{Cha}(v)\frac{q-1}{2}+2e \bmod q = w + \mathsf{Cha}(v)\frac{q-1}{2} \bmod q$. Thus, $\mathsf{Mod}_2(w, \mathsf{Cha}(v)) = \mathsf{Mod}_2(v, \mathsf{Cha}(v))$.

Now, we extend the functions $\mathsf{Cha}$ and $\mathsf{Mod}_2$ to ring $R_q$ by applying them coefficient-wise to ring elements. Namely, for ring element $v = (v_0, \dots, v_{n-1}) \in R_q$ and binary-vector $\mathbf{b} = (b_0, \dots, b_{n-1}) \in \{0,1\}^n$, define $\widetilde{\mathsf{Cha}}(v) = (\mathsf{Cha}(v_0), \dots, \mathsf{Cha}(v_{n-1}))$ and $\widetilde{\mathsf{Mod}_2}(v, \mathbf{b}) = (\mathsf{Mod}_2(v_0, b_0), \dots, \mathsf{Mod}_2(v_{n-1}, b_{n-1}))$. For simplicity, we slightly abuse the notations and still use $\mathsf{Cha}$ and $\mathsf{Mod}_2$ to denote $\widetilde{\mathsf{Cha}}$ and $\widetilde{\mathsf{Mod}_2}$, respectively. Clearly, the result in Lemma 3 still holds when extending to ring elements.

In our AKE protocol, the two involved parties will use $\mathsf{Cha}$ and $\mathsf{Mod}_2$ to derive a common key material. Concretely, the responder will publicly send the result of $\mathsf{Cha}$ on his own secret ring element to the initiator in order to compute a shared key material from two “closed” ring elements (by applying the $\mathsf{Mod}_2$ function). Ideally, for uniformly chosen element $v$ from $R_q$ at random, we hope that the output of $\mathsf{Mod}_2(v, \mathsf{Cha}(v))$ is uniformly distributed $\{0,1\}^n$. However, this can never happen when $q$ is a odd prime. Fortunately, we can show that the output of $\mathsf{Mod}_2(v, \mathsf{Cha}(v))$ conditioned on $\mathsf{Cha}(v)$ has high min-entropy, thus can be used to extract an (almost) uniformly session key. Actually, we can prove a stronger result.

**Lemma 4.** *Let $q$ be any odd prime and $R_q$ be the ring defined above. Then, for any $\mathbf{b} \in \{0,1\}^n$ and any $v' \in R_q$, the output distribution of $\mathsf{Mod}_2(v+v', \mathbf{b})$ given $\mathsf{Cha}(v)$ has min-entropy at least $-n\log(\frac{1}{2}+\frac{1}{|E|-1})$, where $v$ is uniformly chosen from $R_q$ at random. In particular, when $q > 203$, we have $-n\log(\frac{1}{2}+\frac{1}{|E|-1}) > 0.97n$.*

*Proof.* Since each coefficient of $v$ is independently and uniformly chosen from $\mathbb{Z}_q$ at random, we can simplify the proof by focusing on the first coefficient of $v$. Formally, letting $v = (v_0, \dots, v_{n-1})$, $v' = (v'_0, \dots, v'_{n-1})$ and $\mathbf{b} = (b_0, \dots, b_{n-1})$, we condition on $\mathsf{Cha}(v_0)$:

– If $\mathsf{Cha}(v_0) = 0$, then $v_0 + v_0' + b_0 \cdot \frac{q-1}{2}$ is uniformly distributed over $v_0' + b_0 \cdot \frac{q-1}{2} + E \bmod q$. This shifted set has $(q+1)/2$ elements, which are either consecutive integers—if the shift is small enough—or two sets of consecutive integers—if the shift is large enough to cause wrap-around. Thus, we must distinguish a few cases:

- If $|E|$ is even and no wrap-around occurs, then the result of $\mathsf{Mod}_2(v_0 + v_0', b_0)$ is clearly uniform on $\{0, 1\}$. Namely, the result of $\mathsf{Mod}_2(v_0 + v_0', b_0)$ has no bias.
- If $|E|$ is odd and no wrap-around occurs, then the result of $\mathsf{Mod}_2(v_0 + v_0', b_0)$ has a bias with probability $\frac{1}{2|E|}$ over $\{0, 1\}$. In other words, the $\mathsf{Mod}_2(v_0 + v_0', b_0)$ will output either 0 or 1 with probability exactly $\frac{1}{2} + \frac{1}{2|E|}$.
- If $|E|$ is odd and wrap-around does occur, then the set $v_0' + b_0 \cdot \frac{q-1}{2} + E \bmod q$ splits into two parts, one with an even number of elements, and one with an odd number of elements. This leads to the same situation as with no wrap-around.
- If $|E|$ is even and wrap-around occurs, then our sample space is split into either two even-sized sets, or two odd sized sets. If both are even, then once again the result of $\mathsf{Mod}_2(v_0 + v_0', b_0)$ is uniform. If both are odd, it is easy to calculate that the result of $\mathsf{Mod}_2(v_0 + v_0', b_0)$ has a bias with probability $\frac{1}{|E|}$ over $\{0, 1\}$.

– If $\mathsf{Cha}(v_0) = 1$, $v_0 + v_0' + b_0 \cdot \frac{q-1}{2}$ is uniformly distributed over $v_0' + b_0 \cdot \frac{q-1}{2} + \tilde{E}$, where $\tilde{E} = \mathbb{Z}_q \setminus E$. Now $|\tilde{E}| = |E| - 1$, so by splitting into the same cases as $\mathsf{Cha}(v_0) = 0$, the result of $\mathsf{Mod}_2(v_0 + v_0', b)$ has a bias with probability $\frac{1}{|E|-1}$ over $\{0, 1\}$.

In all, we have that the result of $\mathsf{Mod}_2(v_0 + v_0', b_0)$ conditioned on $\mathsf{Cha}(v_0)$ has min-entropy at least $-\log(\frac{1}{2} + \frac{1}{|E|-1})$. Since the bits in the result of $\mathsf{Mod}_2(v + v', \mathbf{b})$ are independent, we have that given $\mathsf{Cha}(v)$, the min-entropy $H_\infty(\mathsf{Mod}_2(v + v', \mathbf{b})) \geq -n \log(\frac{1}{2} + \frac{1}{|E|-1})$. This completes the first claim. The second claim directly follows from the fact that $-\log(\frac{1}{2} + \frac{1}{|E|-1}) > -\log(0.51) > 0.97$ when $q > 203$. □

*Remark 1 (On Uniformly Distributed Keys).* It is known that randomness extractor can be used to obtain an almost uniformly distributed key from a biased bit-string with high min-entropy [18,64,65,27,4]. In practice, as recommended by NIST [5], one can actually use the standard cryptographic hash functions such as SHA-2 to derive a uniformly distributed key if the source string has at least $2\kappa$ min-entropy, where $\kappa$ is the length of the cryptographic hash function.

### 3.1 The Protocol

We now describe our protocol in detail. Let $n$ be a power of 2, and $q$ be an odd prime such that $q \bmod 2n = 1$. Take $R = \mathbb{Z}[x]/(x^n + 1)$ and $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ as above. For $\gamma \in \mathbb{R}^+$, let $H_1 \colon \{0,1\}^* \to \chi_\gamma = D_{\mathbb{Z}^n,\gamma}$ be a hash function that always output invertible elements in $R_q$.[8] Let $H_2 \colon \{0,1\}^* \to \{0,1\}^\kappa$ be the key derivation function, where $\kappa$ is the bit-length of the final shared key. We model both functions as random oracles [6]. Let $\chi_\alpha, \chi_\beta$ be two discrete Gaussian distributions with parameters $\alpha, \beta \in \mathbb{R}^+$. Let $a \in R_q$ be the global public parameter uniformly chosen from $R_q$ at random, and $M$ be a constant determined by Theorem 1. Let $p_i = as_i + 2e_i \in R_q$ be party $i$'s static public key, where $(s_i, e_i)$ is the corresponding static secret key; both $s_i$ and $e_i$ are taken from the distribution $\chi_\alpha$. Similarly, party $j$ has static public key $p_j = as_j + 2e_j$ and static secret key $(s_j, e_j)$.

**Initiation** Party $i$ proceeds as follows:

1. Sample $r_i, f_i \leftarrow_r \chi_\beta$ and compute $x_i = ar_i + 2f_i$;
2. Compute $c = H_1(i, j, x_i)$, $\hat{r}_i = s_i c + r_i$ and $\hat{f}_i = e_i c + f_i$;
3. Letting $\mathbf{z} \in \mathbb{Z}^{2n}$ be the coefficient vector of $\hat{r}_i$ concatenated with the coefficient vector of $\hat{f}_i$, and $\mathbf{z}_1 \in \mathbb{Z}^{2n}$ be the coefficient vector of $s_i c$ concatenated with the coefficient vector of $e_i c$, repeat the steps $1 \sim 3$ with probability $1 - \min\left(\frac{D_{\mathbb{Z}^{2n},\beta}(\mathbf{z})}{M D_{\mathbb{Z}^{2n},\beta,\mathbf{z}_1}(\mathbf{z})}, 1\right)$.
4. Send $x_i$ to party $j$.

**Response** After receiving $x_i$ from party $i$, party $j$ proceeds as follows:

1′. Sample $r_j, f_j \leftarrow_r \chi_\beta$ and compute $y_j = ar_j + 2f_j$;

[8] In practice, one can first use a hash function such as SHA-2 to obtain a uniformly random string, and then use it to sample from $D_{\mathbb{Z}^n,\gamma}$. The algorithm output a sample only if it is invertible in $R_q$, otherwise, it tries another sample and repeats. By Lemma 10 in [63], we can have a good probability to sample an invertible element in each trial for an appropriate choice of $\gamma$.

2′. Compute $d = H_1(j, i, y_j, x_i)$, $\hat{r}_j = s_j d + r_j$ and $\hat{f}_j = e_j d + f_j$;
3′. Letting $\mathbf{z} \in \mathbb{Z}^{2n}$ be the coefficient vector of $\hat{r}_j$ concatenated with the coefficient vector of $\hat{f}_j$, and $\mathbf{z}_1 \in \mathbb{Z}^{2n}$ be the coefficient vector of $s_j d$ concatenated with the coefficient vector of $e_j d$, repeat the steps $1' \sim 3'$ with probability $1 - \min\left(\frac{D_{\mathbb{Z}^{2n},\beta}(\mathbf{z})}{M D_{\mathbb{Z}^{2n},\beta,\mathbf{z}_1}(\mathbf{z})}, 1\right)$;
4′. Sample $g_j \leftarrow_r \chi_\beta$ and compute $k_j = (p_i c + x_i)\hat{r}_j + 2cg_j$ where $c = H_1(i, j, x_i)$;
5′. Compute $w_j = \mathsf{Cha}(k_j) \in \{0,1\}^n$ and send $(y_j, w_j)$ to party $i$;
6′. Compute $\sigma_j = \mathsf{Mod}_2(k_j, w_j)$ and derive the session key $\mathrm{sk}_j = H_2(i, j, x_i, y_j, w_j, \sigma_j)$.

**Finish** Party $i$ receives the pair $(y_j, w_j)$ from party $j$, and proceeds as follows:
5. Sample $g_i \leftarrow_r \chi_\beta$ and compute $k_i = (p_j d + y_j)\hat{r}_i + 2dg_i$ where $d = H_1(j, i, y_j, x_i)$;
6. Compute $\sigma_i = \mathsf{Mod}_2(k_i, w_j)$ and derive the session key $\mathrm{sk}_i = H_2(i, j, x_i, y_j, w_j, \sigma_i)$.

In the above protocol, both parties will make use of rejection sampling, *i.e.,* they will repeat the first three steps with certain probability. By Theorem 1, the probability that each party will repeat the steps with probability about $\frac{1}{M}$ for some constant $M$ and appropriately chosen $\beta$. Thus, one can hope that both parties will send something to each other after an averaged $M$ times repetitions of the first three steps. In the following subsection, we will show that once they send something to each other, both parties will finally compute a shared session key.

### 3.2 Correctness

To show the correctness of our AKE protocol, *i.e.*, that both parties compute the same session key $\mathrm{sk}_i = \mathrm{sk}_j$, it suffices to show that $\sigma_i = \sigma_j$. Since $\sigma_i$ and $\sigma_j$ are both the output of $\mathsf{Mod}_2$ with $\mathsf{Cha}(k_j)$ as the second argument, we need only to show that $k_i$ and $k_j$ are sufficiently close by Lemma 3. Note that the two parties will compute $k_i$ and $k_j$ as follows:

$$\begin{aligned} k_i &= (p_j d + y_j)\hat{r}_i + 2dg_i \\ &= a(s_j d + r_j)\hat{r}_i + 2(e_j d + f_j)\hat{r}_i + 2dg_i \\ &= a\hat{r}_i\hat{r}_j + 2\widetilde{g}_i \end{aligned} \qquad \begin{aligned} k_j &= (p_i c + x_i)\hat{r}_j + 2cg_j \\ &= a(s_i c + r_i)\hat{r}_j + 2(e_i c + f_i)\hat{r}_j + 2cg_j \\ &= a\hat{r}_i\hat{r}_j + 2\widetilde{g}_j \end{aligned}$$

where $\widetilde{g}_i = \hat{f}_j\hat{r}_i + dg_i$, and $\widetilde{g}_j = \hat{f}_i\hat{r}_j + cg_j$. Then $k_i = k_j + 2(\widetilde{g}_i - \widetilde{g}_j)$, and we have $\sigma_i = \sigma_j$ if $\|\widetilde{g}_i - \widetilde{g}_j\|_\infty < q/8$ by Lemma 3.

## 4 Security

**Theorem 2.** *Let $n$ be a power of 2 satisfying $0.97n \geq 2\kappa$, prime $q > 203$ satisfying $q = 1 \mod 2n$, $\beta = \omega(\alpha\gamma n\sqrt{n\log n})$. Then, if RLWE$_{q,\alpha}$ is hard, the proposed AKE is secure with respect to Definition 2 in the random oracle model.*

The intuition behind our proof is quite simple. Since the public element $a$ and the public key of each party (*e.g.*, $p_i = as_i + 2e_i$) actually consist of a RLWE$_{q,\alpha}$ tuple with Gaussian parameter $\alpha$ (scaled by 2), the parties' static public keys are computationally indistinguishable from uniformly distributed elements in $R_q$ under the Ring-LWE assumption. Similarly, both the exchanged elements $x_i$ and $y_j$ are also computationally indistinguishable from uniformly distributed elements in $R_q$ under the RLWE$_{q,\beta}$ assumption. Since the proof is very technical and too long, we refer the readers the full version online.

□

## 5 One-Pass Protocol from Ring-LWE

As MQV [55] and HMQV [44], our AKE protocol has a one-pass variant, which only consists a single message from one party to the other. Let $a \in R_q$ be the global public parameter uniformly chosen from $R_q$ at random, and $M$ be a constant. Let $p_i = as_i + 2e_i \in R_q$ be party $i$'s static public key, where $(s_i, e_i)$ is the corresponding static secret key; both $s_i$ and $e_i$ are taken from the distribution $\chi_\alpha$. Similarly, party $j$ has static public key $p_j = as_j + 2e_j$ and static secret key $(s_j, e_j)$. The other parameters and notations used in this section are the same as before.

**Initiation** Party $i$ proceeds as follows:
1. Sample $r_i, f_i \leftarrow_r \chi_\beta$ and compute $x_i = ar_i + 2f_i$;
2. Compute $c = H_1(i, j, x_i)$, $\hat{r}_i = s_i c + r_i$ and $\hat{f}_i = e_i c + f_i$;
3. Letting $\mathbf{z} \in \mathbb{Z}^{2n}$ be the coefficient vector of $\hat{r}_i$ concatenated with the coefficient vector of $\hat{f}_i$, and $\mathbf{z}_1 \in \mathbb{Z}^{2n}$ be the coefficient vector of $s_i c$ concatenated with the coefficient vector of $e_i c$, repeat the steps $1 \sim 3$ with probability $1 - \min\left(\frac{D_{\mathbb{Z}^{2n},\beta}(\mathbf{z})}{MD_{\mathbb{Z}^{2n},\beta,\mathbf{z}_1}(\mathbf{z})}, 1\right)$.
4. Sample $g_i \leftarrow_r \chi_\beta$ and compute $k_i = p_j \hat{r}_i + 2g_i$ where $c = H_1(i, j, x_i)$;
5. Compute $w_i = \mathsf{Cha}(k_i) \in \{0,1\}^n$ and send $(y_i, w_i)$ to party $j$;
6. Compute $\sigma_i = \mathsf{Mod}_2(k_i, w_i)$ and derive the session key $\mathrm{sk}_i = H_2(i, j, x_i, w_i, \sigma_i)$.

**Finish** Party $j$ receives the pair $(x_i, w_i)$ from party $i$, and proceeds as follows:

1′. Sample $g_j \leftarrow_r \chi_\alpha$ and compute $k_j = (p_i c + x_i)s_j + 2cg_j$ where $c = H_1(i, j, x_i)$;

2′. Compute $\sigma_j = \mathsf{Mod}_2(k_j, w_i)$ and derive the session key $\mathrm{sk}_j = H_2(i, j, x_i, w_i, \sigma_j)$.

The correctness of the protocol simply follows as before. The security of the protocol cannot be proven in the BR model with party corruption. However, we can prove it in a weak model similar to [44]. This one-pass protocol can essentially be used as a KEM, and can be transformed into a CCA 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.

## 6 Concrete Parameters and Timings

In this section, we present concrete choices of parameters, and the timings in a proof-of-concept implementation. Our selection of parameters for our AKE protocols can be found in Table 2. Those parameters were chosen such that the correctness property is satisfied with high probability and with the choice of different levels of security.

For correctness we must satisfy that the error term $\|\widetilde{g}_i - \widetilde{g}_j\|_\infty < q/8$. Note that $\widetilde{g}_i = (e_j d + f_j)(s_i c + r_i) + dg_i$, and $\widetilde{g}_j = (e_i c + f_i)(s_j d + r_j) + cg_j$, where $e_i, e_j \leftarrow_r \chi_\alpha$, $c, d \leftarrow_r \chi_\gamma$, and $f_i, f_j, r_i, r_j, g_i, g_j \leftarrow_r \chi_\beta$. Due to the symmetry, we only estimate the size of $\|\widetilde{g}_i\|_\infty$. At this point, we use the following fact about the product of two Gaussian distributed random values (as stated in [8]). Let $x \in R$ and $y \in R$ be two polynomials whose coefficients are distributed according to a discrete Gaussian distribution with standard deviation $\sigma$ and $\tau$, respectively. The individual coefficients of the product $xy$ are then (approximately) normally distributed around zero with standard deviation $\sigma\tau\sqrt{n}$ where $n$ is the degree of the polynomial.

In our case, it means that we have $\|(e_j d + f_j)(s_i c + r_i)\|_\infty \le 6\beta^2\sqrt{n}$ and $\|dg_i\|_\infty \le 6\gamma\beta\sqrt{n}$ with overwhelming probability (since erfc(6) is about $2^{-55}$). Note that the distributions of $e_j d + f_j$ and $s_i c + r_i$ are both according to $\chi_\beta$ since we use the rejection sampling in the protocol. Now, to choose an appropriate $\beta$ we set $d = 1/2$ in Lemma 1 such that $\|e_j d\|, \|s_i c\| \le 1/2\alpha\gamma n$ with probability at most $2 \cdot 0.943^{-n}$. Hence, for $n \ge 1024$, we get a potential decryption error with only a probability about $2^{-87}$. In order to make the rejection sampling work, it is sufficient to set $\beta \ge \tau * 1/2\alpha\gamma n = 1/2\tau\alpha\gamma n$ for some constant $\tau$ (which is much better than the worst-case bound $\beta = \omega(\alpha\gamma\sqrt{n \log n})$ in Theorem 1). For instance, if $\tau = 12$, we have an expect number of rejection sampling about $M = 2.72$ and a statistical distance about $\frac{2^{-100}}{M}$ by Theorem 1. For such a choice of $\beta$, we can safely assume that $\|\widetilde{g}_i\|_\infty \le 6\beta^2\sqrt{n} + 6\gamma\beta\sqrt{n} \le 7\beta^2\sqrt{n}$. Thus, it is enough to set $16 * 7\beta^2\sqrt{n} < q$ for correctness of the protocol.

Though the Ring-LWE problem enjoys a worst-case connection to some hard problems (*e.g.*, SIVP [52]) on ideal lattices, the connection as summarized in Proposition 1 seems less powerful to estimate the actual security for concrete choices of parameters. In order to assess the concrete security of our parameters, we use the approach of [20], which investigates the two most efficient ways to solve the underlying (R)LWE problem, namely the embedding and decoding attacks. As opposed to [20], the decoding attack is more efficient against our instances because in RLWE with $m \ge 2n$ one typically is close to the optimal attack dimension for the corresponding attacks. The decoding attack first uses a lattice reduction algorithm, such as BKZ [61] / BKZ 2.0 [17] and then applies a decoding algorithm, such as Babai's nearest plane [2], Lindner and Peikert's nearest planes [48], or Liu and Nguyen's pruned enumeration approach [49]. Finally, the closest vector is returned which coincides with the error polynomial, and the secret polynomial is recovered.

As recommended in [48,33], it is enough to set the Gaussian parameter $\alpha \ge 3.2$ so that the discrete Gaussian $D_{\mathbb{Z}^n,\alpha}$ approximates the continuous Gaussian $D_\alpha$ extremely well[9]. In our experiment, we fix $\alpha = 3.397$ for a better

[9] Only $\alpha$ is considered because $\beta \gg \alpha$, and the (R-)LWE problem becomes harder as $\alpha$ grows bigger (for a fixed modulus $q$).

**Table 2.** Choices of Parameters (The bound $6\alpha$ with $\mathrm{erfc}(6) \approx 2^{-55}$ is used to estimate the size of secret keys)

| Protocol | Choice of Parameters | $n$ | Security | $\alpha$ | $\tau$ | $\log \beta$ | $\log q$ (bits) | Size (KB) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | pk | sk (expt.) | init. msg | resp. msg |
| Two-pass | $\mathrm{I}_1$ | 1024 | 80 bits | 3.397 | 12 | 16.1 | 45 | 5.625 KB | 1.5 KB | 5.625 KB | 5.75 KB |
| | $\mathrm{I}_2$ | | 75 bits | 3.397 | 24 | 17.1 | 47 | 5.875 KB | 1.5 KB | 5.875 KB | 6.0 KB |
| | $\mathrm{II}_1$ | 2048 | 230 bits | 3.397 | 12 | 17.1 | 47 | 11.75 KB | 3.0 KB | 11.75 KB | 12.0 KB |
| | $\mathrm{II}_2$ | | 210 bits | 3.397 | 36 | 18.7 | 50 | 12.50 KB | 3.0 KB | 12.50 KB | 12.75 KB |
| One-pass | $\mathrm{III}_1$ | 1024 | 160 bits | 3.397 | 12 | 16.1 | 30 | 3.75 KB | 1.5 KB | 3.75 KB | 3.875 KB |
| | $\mathrm{III}_2$ | | 140 bits | 3.397 | 36 | 17.7 | 32 | 4.0 KB | 1.5 KB | 4.0 KB | 4.125 KB |
| | $\mathrm{IV}_1$ | 2048 | 360 bits | 3.397 | 12 | 17.1 | 32 | 8.0 KB | 3.0 KB | 8.0 KB | 8.25 KB |
| | $\mathrm{IV}_2$ | | 350 bits | 3.397 | 36 | 18.7 | 33 | 8.25 KB | 3.0 KB | 8.25 KB | 8.5 KB |

performance of the Gaussian sampling algorithm in [29]. As for the choices of $\gamma$, we set $\gamma = \alpha$ for simplicity (actually such a choice in our experiments works very well: no rejection happened for 1000 times hash evaluations). In Table 1, we set all other parameters $\beta, n, q$ for our two-pass protocol to satisfy the correctness condition. We also give the parameter choices of our one-pass protocol (in this case, we can save a factor of $\beta$ in $q$ due to the asymmetry). Note that $n$ is required to be a power of 2 in our protocol (*i.e.*, it is very sparsely distributed[10]), we present several candidate choices of parameters for $n = 1024, 2048$, and estimate the sizes of public keys, secret keys, and communication overheads in Table 2.

**Table 3.** Timings of proof-of-concept implementations in ms.

| Protocol | Parameters | $\tau$ | Initiation | Response | Finish |
|---|---|---|---|---|---|
| Two-pass | $\mathrm{I}_1$ | 12 | 22.05 ms | 30.61 ms | 4.35 ms |
| | $\mathrm{I}_2$ | 24 | 14.26 ms | 19.18 ms | 4.41 ms |
| | $\mathrm{II}_1$ | 12 | 49.77 ms | 60.31 ms | 9.44 ms |
| | $\mathrm{II}_2$ | 36 | 25.40 ms | 36.96 ms | 9.59 ms |

| Protocol | Parameters | $\tau$ | Initiation | Finish |
|---|---|---|---|---|
| One-pass | $\mathrm{III}_1$ | 12 | 26.17 ms | 3.64 ms |
| | $\mathrm{III}_2$ | 36 | 14.57 ms | 3.70 ms |
| | $\mathrm{IV}_1$ | 12 | 53.78 ms | 7.75 ms |
| | $\mathrm{IV}_2$ | 36 | 32.28 ms | 7.94 ms |

We implement our AKE protocol by using the NTL library compiled with the option NTL_GMP_LIP=on (*i.e.*, building NTL using the GNU Multi-Precision package). The implementations are written in C++ without any parallel computations or multi-threads programming techniques. The program is run on a Dell Optiplex 780 computer with Ubuntu 12.04 TLS 64-bit system, equipped with a 2.83GHz Intel Core 2 Quad CPU and 3.8GB RAM. We use a $n$-dimensional Fast Fourier Transform (FFT) for the multiplications of two ring elements [19,51]. We use the CDT algorithm [58] as a tool for hashing to $D_{\mathbb{Z}^n,\gamma}$ and sampling from $D_{\mathbb{Z}^n,\alpha}$, but use the DDLL algorithm [29] for sampling from $D_{\mathbb{Z}^n,\beta}$ (because the CDT algorithm has to store large precomputed values for a big $\beta$). In Table 3, we present the timings of each operation, and the figures represent the averaged timing (in millisecond, ms) for 1000 executions. Since our protocols also allow some kind of precomputations such as sampling Gaussian distributions offline, the timings can be greatly reduced if one consider it in practice. Finally, we note that our implementation has not undergone any real optimization, and it can much improved in practice.

## References


1. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO*, pages 595–618. 2009.
2. László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.


[10] We remark such a choice of $n$ is not necessary, but it gives a simple analysis and implementation. In practice, one might use the techniques for Ring-LWE cryptography in [53] to give a tighter choice of parameters for desired security levels.

3. Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In *CT-RSA*, pages 28–47, 2014.
4. B. Barak, R. Impagliazzo, and A. Wigderson. Extracting randomness using few independent sources. *SIAM Journal on Computing*, 36(4):1095–1118, 2006.
5. E. Barker and A. Roginsky. Recommendation for the entropy sources used for random bit generation. Draft NIST Special Publication 800-90B, August 2012.
6. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS*, pages 62–73, 1993.
7. Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *CRYPTO*, volume 773, pages 232–249. 1994.
8. Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. Cryptology ePrint Archive, Report 2014/599, 2014.
9. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. Fully homomorphic encryption without bootstrapping. *Innovations in Theoretical Computer Science, ITCS*, pages 309–325, 2012.
10. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from Ring-LWE and security for key dependent messages. In *CRYPTO*, pages 505–524. 2011.
11. Christina Brzuska, Marc Fischlin, Nigel P. Smart, Bogdan Warinschi, and Stephen C. Williams. Less is more: relaxed yet composable security notions for key exchange. *Int. J. Inf. Sec.*, 12(4):267–297, 2013.
12. Christina Brzuska, Marc Fischlin, Bogdan Warinschi, and Stephen C. Williams. Composability of bellare-rogaway key exchange protocols. In *CCS*, pages 51–62, 2011.
13. BSI. Advanced security mechanism for machine readable travel documents extended access control (eac). Technical Report (BSI-TR-03110) Version 2.05 Release Candidate, Bundesamt fuer Sicherheit in der Informationstechnik (BSI), 2010.
14. Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *EUROCRYPT*, pages 453–474. 2001.
15. Ran Canetti and Hugo Krawczyk. Security analysis of IKEs signature-based key-exchange protocol. In *CRYPTO*, pages 143–161. 2002.
16. Lily Chen. Practical impacts on qutumn computing. Quantum-Safe-Crypto Workshop at the European Telecommunications Standards Institute, 2013. http://docbox.etsi.org/Workshop/2013/201309_CRYPTO/S05_DEPLOYMENT/NIST_CHEN.pdf.
17. Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *ASIACRYPT*, pages 1–20, 2011.
18. Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. In *FOCS*, pages 429–442, 1985.
19. Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, Clifford Stein, et al. *Introduction to algorithms*, volume 2. MIT press Cambridge, 2001.
20. Özgür Dagdelen, Rachid El Bansarkhani, Florian Göpfert, Tim Güneysu, Tobias Oder, Thomas Pöppelmann, Ana Helena Snchez, and Peter Schwabe. High-speed signatures from standard lattices. In *to appear at LATINCRYPT*. 2014.
21. Özgür Dagdelen and Marc Fischlin. Security analysis of the extended access control protocol for machine readable travel documents. In *ISC*, pages 54–68, 2010.
22. Özgür Dagdelen, Marc Fischlin, Tommaso Gagliardoni, Giorgia Azzurra Marson, Arno Mittelbach, and Cristina Onete. A cryptographic analysis of OPACITY - (extended abstract). In *ESORICS*, pages 345–362, 2013.
23. Jean Paul Degabriele, Victoria Fehr, Marc Fischlin, Tommaso Gagliardoni, Felix Günther, Giorgia Azzurra Marson, Arno Mittelbach, and Kenneth G. Paterson. Unpicking PLAID - a cryptographic analysis of an ISO-standards-track authentication protocol. Cryptology ePrint Archive, Report 2014/728, 2014.
24. Tim Dierks. The transport layer security (TLS) protocol version 1.2. 2008.
25. W. Diffie and M. Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644 – 654, nov 1976.
26. Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. Cryptology ePrint Archive, Report 2012/688, 2012.
27. Yevgeniy Dodis, Rosario Gennaro, Johan Håstad, Hugo Krawczyk, and Tal Rabin. Randomness extraction and key derivation using the CBC, Cascade and HMAC modes. In *CRYPTO*, pages 494–510. 2004.
28. Léo Ducas and Alain Durmus. Ring-LWE in polynomial rings. In *PKC*, pages 34–51, 2012.
29. Lo Ducas, Alain Durmus, Tancrde Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal Gaussians. In *CRYPTO*, pages 40–56. 2013.
30. Alan Freier. The SSL protocol version 3.0. *http://wp. netscape. com/eng/ssl3/draft302. txt*, 1996.
31. Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. In *PKC*, pages 467–484. 2012.
32. Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama. Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In *ASIACCS*, pages 83–94, 2013.

33. Craig Gentry, Shai Halevi, and NigelP. Smart. Homomorphic evaluation of the AES circuit. In *CRYPTO*, pages 850–867. 2012.
34. Florian Giesen, Florian Kohlar, and Douglas Stebila. On the security of TLS renegotiation. In *CCS*, pages 387–398, 2013.
35. Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In *Innovations in Computer Science*, pages 230–240, 2010.
36. Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In *CHES*, pages 530–547, 2012.
37. Dan Harkins, Dave Carrel, et al. The internet key exchange (IKE). Technical report, RFC 2409, november, 1998.
38. Jeffrey Hoffstein, Jill Pipher, John M. Schanck, Joseph H. Silverman, and William Whyte. Practical signatures from the partial fourier recovery problem. In *ACNS*, pages 476–493, 2014.
39. ISO/IEC. 11770-3:2008 information technology – security techniques – key management – part 3: Mechanisms using asymmetric techniques.
40. Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of TLS-DHE in the standard model. In *CRYPTO*, pages 273–293. 2012.
41. Jonathan Katz and Vinod Vaikuntanathan. Smooth projective hashing and password-based authenticated key exchange from lattices. In *ASIACRYPT*, pages 636–652. 2009.
42. Charlie Kaufman, Paul Hoffman, Yoav Nir, and Pasi Eronen. Internet key exchange protocol version 2 (IKEv2). Technical report, RFC 5996, September, 2010.
43. Hugo Krawczyk. SIGMA: The 'SIGn-and-MAc' approach to authenticated Diffie-Hellman and its use in the IKE protocols. In *CRYPTO*, pages 400–425. 2003.
44. Hugo Krawczyk. HMQV: A high-performance secure Diffie-Hellman protocol. In *CRYPTO*, pages 546–566. 2005.
45. Hugo Krawczyk, KennethG. Paterson, and Hoeteck Wee. On the security of the TLS protocol: A systematic analysis. In *CRYPTO*, pages 429–448. 2013.
46. Brian A. LaMacchia, Kristin E. Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In *ProvSec*, pages 1–16, 2007.
47. Xinyu Lei and Xiaofeng Liao. NTRU-KE: A lattice-based public key exchange protocol. Cryptology ePrint Archive, Report 2013/718, 2013.
48. Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In*CT-RSA*, pages 319–339. 2011.
49. Mingjie Liu and Phong Q. Nguyen. Solving BDD by enumeration: An update. In *CT-RSA*, pages 293–309. 2013.
50. Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors,*EUROCRYPT*, pages 738–755. 2012.
51. Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. SWIFFT: A modest proposal for FFT hashing. In *FSE*, pages 54–72. 2008.
52. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, pages 1–23. 2010.
53. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for Ring-LWE cryptography. In *EUROCRYPT*, pages 35–54. 2013.
54. Nikos Mavrogiannopoulos, Frederik Vercauteren, Vesselin Velichkov, and Bart Preneel. A cross-protocol attack on the TLS protocol. In *CCS*, pages 62–72, 2012.
55. A. Menezes, M. Qu, and S. Vanstone. Some new key agreement protocols providing mutual implicit authentication. In *SAC*, 1995.
56. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37:267–302, 2007.
57. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In*STOC*, pages 333–342, 2009.
58. Chris Peikert. An efficient and parallel Gaussian sampler for lattices. In *CRYPTO*, pages 80–97. 2010.
59. Chris Peikert. Lattice cryptography for the Internet. Cryptology ePrint Archive, Report 2014/070, 2014.
60. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *STOC*, pages 187–196, 2008.
61. Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.*, 66:181–199, 1994.
62. P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.
63. Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *EUROCRYPT*, pages 27–47. 2011.
64. L. Trevisan and S. Vadhan. Extracting randomness from samplable distributions. In *FOCS*, pages 32–, 2000.
65. Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, July 2001.
66. Andrew Chi-Chih Yao and Yunlei Zhao. OAKE: A new family of implicitly authenticated Diffie-Hellman protocols. In *CCS*, pages 1113–1128, 2013.