NewHope without reconciliation

Erdem Alkim[1], Léo Ducas[2⋆], Thomas Pöppelmann[3], and Peter Schwabe[4] ⋆⋆

[1] Department of Mathemathics, Ege University, Turkey
erdemalkim@gmail.com
[2] Centrum voor Wiskunde en Informatica, Amsterdam, The Netherlands
leo.ducas@cwi.nl
[3] Infineon Technologies AG, Munich, Germany
thomas.poeppelmann@infineon.com
[4] Digital Security Group, Radboud University, The Netherlands
peter@cryptojedi.org

Abstract. In this paper we introduce NewHope-Simple, a variant of the NewHope Ring-LWE-based key exchange that is using a straight-forward transformation from Ring-LWE encryption to a passively secure KEM (or key-exchange scheme). The main advantage of NewHope-Simple over NewHope is simplicity. In particular, it avoids the error-reconciliation mechanism originally proposed by Ding. The explanation of his method, combined with other tricks, like unbiasing the key following Peikert's tweak and using the quantizer D_4 to extract one key bit from multiple coefficients, takes more than three pages in the NewHope paper.
The price for that simplicity is small: one of the exchanged messages increases in size by 6.25% from 2048 bytes to 2176 bytes. The security of NewHope-Simple is the same as the security of NewHope; the performance is very similar.
Keywords. Post-quantum key exchange, NewHope, code simplicity.

1 Introduction

Two approaches to (Ring)-LWE Key Exchange

Ring-LWE-based key exchange has recently gotten quite some attention, if nothing else because Google deployed the NewHope scheme [ADPS16] in a post-quantum TLS experiment [Bra16]. In this short paper we revisit one aspect of passively secured Ring-LWE-based key encapsulation (KEM) or key exchange, namely how exactly the second component of the encapsulation is computed (and consequently how precisely Alice and Bob agree on the shared key). Throughout the paper we will use Ring-LWE (i.e., polynomial) notation, but the discussion applies just as well to general LWE and some references date back to before Ring-LWE was introduced.

Let us first fix some notation and then review what the schemes discussed in this paper have in common. Throughout the paper $\mathcal{R}_q$ will denote the ring $\mathbb{Z}_q[X]/(X^n+1)$ for some integers q and n and χ will denote a noise distribution. Sampling from χ yields elements of $\mathcal{R}_q$ that have "small" coefficients. We will denote polynomials in boldface characters. What all schemes discussed in this paper have in common is the following: Alice and Bob agree on a system parameter $\mathbf{a} \in \mathcal{R}_q$. Alice samples $(\mathbf{s}, \mathbf{e})$ from χ and sends a public key $\mathbf{b} = (\mathbf{as}+\mathbf{e})$ to Bob. Bob samples $(\mathbf{s}', \mathbf{e}')$ from χ and sends $\mathbf{u} = (\mathbf{as}' + \mathbf{e}')$ to Alice. This allows Bob to compute $\mathbf{v} = (\mathbf{as} + \mathbf{e})\mathbf{s}' = \mathbf{ass}' + \mathbf{es}'$ and Alice to compute $\mathbf{v}' = (\mathbf{as}' + \mathbf{e}')\mathbf{s} = \mathbf{ass}' + \mathbf{e}'\mathbf{s}$. Note at this point that in both approaches discussed in this paper, Bob

⋆ This work has been supported by a grant from CWI from budget for public-private-partnerships and in part by a grant from NXP Semiconductors.
⋆⋆ This work has furthermore been supported by TÜBITAK under 2214-A Doctoral Research Program Grant, by the European Commission through the ICT program under contract ICT-645622 (PQCRYPTO), and by the Netherlands Organisation for Scientific Research (NWO) through Veni 2013 project 13114 and through a Free Competition Grant. Permanent ID of this document: 0462d84a3d34b12b75e8f5e4ca032869. Date: 2017-11-08

adds an additional noise term $\mathbf{e}''$, sampled from χ, into $\mathbf{v}$, so we actually have $\mathbf{v} = \mathbf{ass}' + \mathbf{es}' + \mathbf{e}''$. As $\mathbf{s}, \mathbf{s}', \mathbf{e}, \mathbf{e}'$ and $\mathbf{e}''$ are "small", the values $\mathbf{v}$ and $\mathbf{v}'$ are both *approximately* $\mathbf{ass}'$. This observation is the basis of lattice-based encryption schemes and key-exchange schemes. This approximate lattice-based key exchange was listed as a special instance of "Noisy Diffie Hellman" by Gaborit (presenting joint work with Aguilar, Lacharme, Schrek, and Zémor) in his talk at PQCrypto 2010 [Gab10, Slides 4 and 6]. It was also described by Lindner and Peikert as "(approximate) key agreement" [LP11, Sec. 3.1] and was already mentioned vaguely in an invited lecture of Peikert at TCC 2009 [Pei09c, Slide 14].

In order to agree on an *exact* (instead of an approximate) shared key, Bob needs to send some additional information; this is where the two approaches discussed in this paper differ. The first approach is straight-forwardly derived from (Ring-)LWE-based encryption schemes, for example the ones described in [LP11] (moved to the Ring-LWE setting) and in [LPR10a] (not in the original conference version of the paper, but in the slides of the presentation at Eurocrypt 2010 [LPR10b] and in full journal version [LPR13]). In this approach, Bob picks a secret ν (in the original encryption schemes, this would be the message), encodes it into the high bits of the coefficients of a polynomial $\mathbf{k} = \mathsf{Encode}(\nu)$, and computes and sends $\mathbf{c} = \mathbf{v} + \mathbf{k}$. Alice, after computing $\mathbf{v}'$, computes $\mathbf{k}' = \mathbf{c} - \mathbf{v}' \approx \mathbf{k}$ and recovers the high bits. Note that in the context of public-key encryption, the encoding function Encode has to be injective so that Alice can recover the original message. In the context of key exchange this is not required: Alice and Bob can agree on a key from the high bits of the coefficients of $\mathbf{k}$, for example by simply hashing those bits; there is no need for Alice to recover ν. We will in the following refer to this approach as the *encryption-based* approach for RLWE-based key exchange (see Figure 1a).

In [Din12b], Ding[5] described an approach to exact keys from an approximate key exchange that does not require Bob to choose and encode the additional secret k. The idea is to use a "reconciliation mechanism" to extract an exact shared value from noisy data, which is essentially the idea of a fuzzy extractor [DRS04], known, for example, from physically unclonable functions. See, for example, [BGS+08, HKM+12]. In the remainder of this paper we will refer to this approach to (Ring-)LWE based key exchange as the "reconciliation-based approach" (see Figure 1b).

In Ding's reconciliation mechanism, the additive noise terms $\mathbf{e}$, $\mathbf{e}'$, and $\mathbf{e}''$ are multiplied by 2, so that (except for modular wrap-around), the approximate shared values $\mathbf{v}$ and $\mathbf{v}'$ agree on the least significant bit[6]. The reconciliation information sent by Bob aims at fixing (with very high probability) the problem that Alice's value $\mathbf{v}'$ and Bob's value $\mathbf{v}$ have a different modular wrap-around. For each coefficient v_i of $\mathbf{v}'$, Bob sends one bit of information stating whether the coefficient is in $[-\lfloor q/4 \rfloor, \lfloor q/4 \rfloor]$ or not. For each coefficient that is in this interval it is very unlikely that Alice's and Bob's value have a different wrap-around and they simply extract the least significant bit. For each coefficient that is *not* in this interval, both Alice and Bob add $q/2$ before extracting the least significant bit. A small issue with this approach was pointed out and fixed by Peikert in [Pei14]: Ding's approach inevitably generates biased keys; Peikert's reconciliation mechanism avoids this by using a "randomized doubling" technique. For details see [Pei14, Section 3]. A very similar randomization technique was adopted by Ding, Xie, and Lin the latest version of their manuscript [DXL14] to eliminate the bias in the key.

A comparison of the two approaches to Ring-LWE-based key exchange is given in Figure 1b and Figure 1a. The function $\mathsf{HelpRec}$ in Figure 1b is a (in some cases randomized) algorithm computing the reconciliation information $\mathbf{r}$; the function Rec is an algorithm that computes a common key given a noisy key and the reconciliation information. A straight-forward way to instantiate the encoding function used in Figure 1a to encode the binary secret value k of length n into a polynomial $\mathbf{v} \in \mathcal{R}_q$ given in [LPR10a, LP11] is defined as

$$\mathbf{k} = \mathsf{Encode}(\nu \in \{0,1\}^n) = \sum_{i=0}^{n-1} k_i \cdot \left\lfloor \frac{q}{2} \right\rfloor \cdot X^i.$$

[5] Later and revised versions of the manuscript also list Lin [DL13] and Xie and Lin [DXL14] as authors.

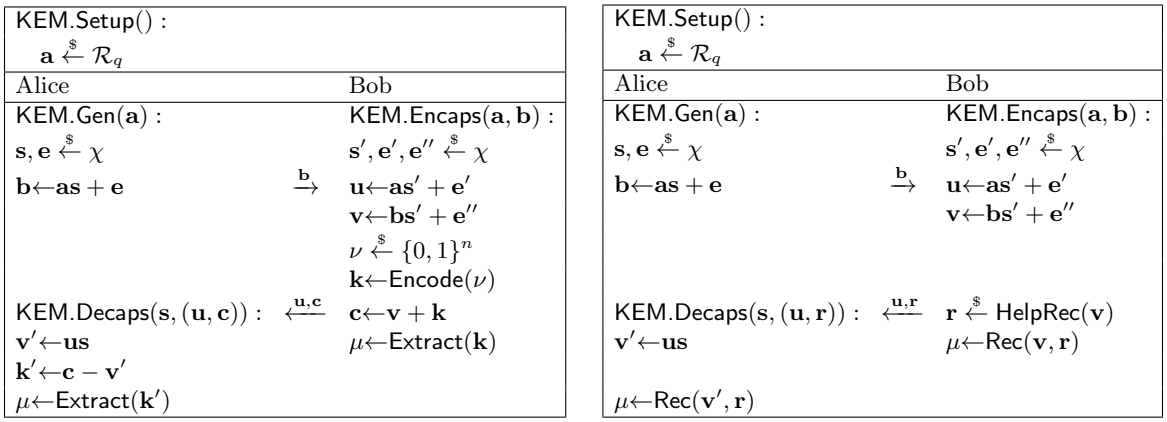
[6] Ding also considers values other than 2 to extract more than one bit per coefficient at the cost of higher failure probability or lower security; we focus on the most relevant case in the context of Ring-LWE-based key exchange.

The corresponding straight-forward way to implement the extraction of the shared key is through decoding as follows:

$$\mu = \mathsf{Extract}(\mathbf{k}') = \mathsf{Decode}(\mathbf{k}') = \left\{ \begin{array}{l} \mu_i = 1 \text{ iff } \nu_i \in \left[-\left\lfloor \frac{q}{4} \right\rfloor, \left\lfloor \frac{q}{4} \right\rfloor \right) \\ \mu_i = 0 \text{ otherwise} \end{array} \right\}, \text{for } i = 0, \ldots, n-1.$$

Note that with these Encode and Extract = Decode functions, the scheme in Figure 1a is exactly the Ring-LWE-based public-key encryption scheme described in [LPR10b] and [LPR13] phrased as key encapsulation. Several implementations of this scheme have been provided since then, e.g., on FPGAs [GFS+12, RVM+14, PG13] or microcontrollers [LSR+15, POG15]. An interesting observation by Stehlé [Ste10, Slide 19] is that Ring-LWE based encryption is basically ElGamal (with noise). We would also like to mention that Yao, Huang, Khanna, Shelat, Calhoun, Lach, and Evans already provided a hardware implementation of a ideal-lattice based public key encryption scheme [YHK+11]. In [RS10, Appendix A.2.] Rückert and Schneider give variants denoted as Multi-bit LWE, Dual Ring-LWE, Multi-bit Ring-LWE, and Dual-LWE.

The main reason for the reconciliation-based approach is a reduced bandwidth requirement. For example, [Pei14] advertises "nearly halving the ciphertext size". This estimate comes from the fact that (in a naive version of the encryption-based approach) the polynomial $\mathbf{c}$ in Figure 1a has coefficients in $\{0, \ldots, q-1\}$ whereas the coefficients of $\mathbf{r}$ in Figure 1b are in $\{0, 1\}$ (for [Din12b, DL13, DXL14] and [Pei14]) or in $\{0, 1, 2, 3\}$ for NewHope.



(a) Encryption-based KEM

(b) Reconciliation-based KEM

Fig. 1: Two approaches to construct a passively secure lattice-based KEM or key-exchange scheme

The NewHope scheme

Recently, in [ADPS16], we proposed NewHope, an instantiation (with some new tricks) of a lattice-based KEM following the reconciliation-based approach. The reconciliation technique of NewHope is generalizing and improving the previous approaches and extracts a single key bit from multiple polynomial coefficients. It relies on non-trivial *lattice-codes* and *lattice-quantizers* [CN88]. It is efficient, but fairly complex. The description of just this mechanism in the paper takes more than 3 pages and a preliminary version of the paper (and accompanying software) contained a mistake in precisely this mechanism brought to our attention by Hamburg [Ham15].

This work

In this work we investigate the characteristics of the encryption-based approach instantiated with the parameters and arithmetic proposed for NewHope in [ADPS16]. We call this variant of NewHope derived from encryption NewHope-Simple. NewHope-Simple has the same security properties as NewHope, similar performance, and, most importantly, is considerably simpler, because it does not need the reconciliation mechanism of NewHope. The difference in bandwidth requirements for NewHope and NewHope-Simple is much smaller than one might expect. Specifically, the message from Bob to Alice (the ciphertext) in NewHope-Simple requires only 2176 bytes (compared to 2048 bytes in NewHope); the message from Alice to Bob (the public key) has the same size (1824 bytes) for both variants. We obtain this result by carefully analyzing and optimizing a technique that is known since at least [Pei09a, Sec. 4.2] and has also been used in [PG13], for lattice-based signatures in [GLP12], in the context of fully homomorphic encryption in [BV11, Sec. 4.2] and [BLLN13, Sec. 5.4] and for lattice-based PRFs in [BPR12]. The idea of this technique is that the low bits of each coefficient of $\mathbf{c}$ mainly carry noise and contribute very little to the successful recovery of the plaintext. One can thus decide to "discard" (i.e, not transmit) those bits and thus shorten the length of $\mathbf{c}$. This can also be seen as switching to a smaller modulus and is therefore also called "modulus switching".

We combine this technique with a simple technique to encode one key bit into 4 coefficients that was first described by Güneysu and Pöppelmann in [PG13] and minimize the ciphertext size under the constraint that the failure probability of NewHope-Simple does not exceed the failure probability of NewHope.

Concurrent related work

In independent concurrent work, also Jin and Zhao describe key exchange following the encryption-based approach. In particular they derive what they call AKCN-RLWE-4:1, which is precisely the approach that we briefly discuss in the paragraph "The true cost of the encryption-based approach". Jin and Zhao posted an early version of their paper in November 2016 on the arXiv [JZ16]; we became aware of their work only when it was posted it on IACR's eprint archive in October 2017 [JZ17]. Already in January 2016, Ducas described a general framework for reconciliation and applications to LWE encryption and key exchange in a blog post [Duc16].

The true cost of the encryption-based approach

If one would rather optimize bandwidth than simplicity, while still opting for the encryption-based approach, we feel compelled to clarify what should be done: the bandwidth overhead can be brought down to exactly 256 bits. This is only natural, since those 256 bits are chosen by one party rather than derived from the protocol. The 1024 bits of overhead of NewHope-Simple over NewHope is a consequence of a deliberate choice of simplification.

To achieve such bandwidth, one should note that the use of a lattice-code/quantizer D_4 in NewHope (or any other similar scheme) is *independent* of the reconciliation-based approach. The typical modulus switching (discarding lower order bits) is to be interpreted as a quantizing algorithm, to the trivial lattice $\frac{q}{4}\mathbb{Z}^4$ (2 higher order bits per coordinate). The strategy developed in NewHope consist in replacing this trivial lattice by $\frac{q}{4}D_4$, which is twice denser that $\frac{q}{4}\mathbb{Z}^4$: one extra bit per 4 coordinates would be required in encryption mode. This extra bit is exactly what is removed by using the reconciliation trick of the protocol by Ding [Din12a] or Peikert [Pei14].

For the sake of simplicity, in NewHope-Simple we instead downgraded our scheme back to a trivial quantizer, namely $\frac{q}{8}\mathbb{Z}^4$: the price is one extra bit per coordinate (1024 bits in total), which we deem a worthy trade-off, when read as relative bandwidth overhead of 6.25%.

2 Scheme and Analysis

2.1 Notation

In the description of NewHope-Simple we use the same notation as in [ADPS16]. In particular, $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n+1)$ and ψ_{16}^n is the centered binomial distribution with parameter 16. In the definitions of NHSDecode and NHSCompress we assume that all coefficients of the input are in $\{0, \ldots, q-1\}$. The operator $\lceil \cdot \rfloor$ denotes rounding to the nearest integer.

2.2 Protocol description

In Figure 1 we provide the full description of the NewHope-Simple protocol; for the definition of the NTT and error distribution χ we refer to [ADPS16]. We use the same parameters $q = 12289$, $n = 1024$, $k = 16$ as NewHope and only replace the Rec and HelpRec functions and modify the encodeB function.

As previously stated, the difference in NewHope-Simple is that Bob chooses a random 256-bit key $\nu \xleftarrow{\$} \{0,1\}^{256}$. Each bit of the key ν is then encoded into four coefficients by NHSEncode (see Algorithm 1). The decoding function NHSDecode (see Algorithm 2) maps from four coefficients back to the original key bit. The idea of this mapping is the same as in [PG13]: First map the four coefficients from the range $\{0, \ldots, q-1\}$ into the range $\{\lfloor -q/2 \rfloor, \ldots, \lfloor q/2 \rfloor\}$, accumulate their absolute values, and then set the key bit to 1 if this sum is larger than q and to 0 otherwise. The computation given in Algorithm 2 follows a slightly different (but equivalent) approach, which is more efficient when protecting against timing attacks. Take four coefficients (each in the range $\{0, \ldots, q-1\}$, subtract $\lfloor q/2 \rfloor$ from each of them, accumulate their absolute values, and set the key bit to 0 if the sum is larger than q or to 1 otherwise.

The compression and decompression functions simply perform coefficient-wise modulus switching between modulus q and modulus 8 by multiplying by the new modulus and then performing a rounding division by the old modulus.

Additionally, we would like to note that the first hashing of the key (SHA3-256(ν)) is the common protection against a random-number generator leaking internal state information through its output. The final hashing performed by Alice and Bob is required so that Theorem 4.1 from [ADPS16] still holds (Renyi divergence argument supporting the use of the binomial distribution).

Algorithm 1 NewHope-Simple key encoding

function NHSEncode($\nu \in \{0,1\}^{256}$)
  $\mathbf{k} \leftarrow 0$
  for i **from** 0 **to** 255 **do**
    $k_i \leftarrow \nu_i \cdot \lfloor q/2 \rfloor$
    $k_{i+256} \leftarrow \nu_i \cdot \lfloor q/2 \rfloor$
    $k_{i+512} \leftarrow \nu_i \cdot \lfloor q/2 \rfloor$
    $k_{i+768} \leftarrow \nu_i \cdot \lfloor q/2 \rfloor$
  return k

Algorithm 2 NewHope-Simple key decoding

function NHSDecode($\mathbf{k}' \in \mathcal{R}_q$)
  $\nu \leftarrow 0$
  for i **from** 0 **to** 255 **do**
    $t \leftarrow \sum_{j=0}^{3} |v_{i+256 \cdot j} - \lfloor q/2 \rfloor|$
    if $t < q$ **then**
      $\nu_i \leftarrow 1$
    else
      $\nu_i \leftarrow 0$
  return ν

Note that in our protocol the key is entirely chosen by Bob; the protocol is thus not contributory. One might be concerned that the transported key might be weak if Bob's random-number generator is weak as the key only depends on Bob's input (contrary to the reconciliation approach). However, if *either party* has a weak random number generator in the reconciliation-based approach, this party will not be able to securely generate random noise polynomials (e.g., $\mathbf{s}', \mathbf{e}', \mathbf{e}'' \xleftarrow{\$} \psi_{16}^n$), rendering the

Algorithm 3 NewHope-Simple ciphertext compression

function NHSCompress($\mathbf{c} \in \mathcal{R}_q$)
  for i **from** 0 **to** 1023 **do**
    $\bar{c}_i \leftarrow \lceil (c'_i \cdot 8)/q \rfloor \mod 8$
  return $\bar{\mathbf{c}}$

Algorithm 4 NewHope-Simple ciphertext decompression

function NHSDecompress($\bar{\mathbf{c}} \in \{0, \ldots, 7\}^{1024}$)
  for i **from** 0 **to** 1023 **do**
    $c'_i \leftarrow \lceil (\bar{c}_i \cdot q)/8 \rfloor$
  return $\mathbf{c}$'

reconciliation-based approach insecure as well. Furthermore, applications that insist on a contributory key agreement and want to use the encryption-based approach (e.g., NewHope-Simple), can simply hash (parts of) Alice's public key into the final shared key.

Parameters: $q = 12289 < 2^{14},\ n = 1024$ Error distribution: ψ_{16}^n		
Alice (server)		**Bob (client)**
$seed \xleftarrow{\$} \{0, \ldots, 255\}^{32}$		
$\hat{\mathbf{a}} \leftarrow \mathsf{Parse}(\mathsf{SHAKE\text{-}128}(seed))$		
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \psi_{16}^n$		$\mathbf{s}', \mathbf{e}', \mathbf{e}'' \xleftarrow{\$} \psi_{16}^n$
$\hat{\mathbf{s}} \leftarrow \mathsf{NTT}(\mathbf{s})$		
$\hat{\mathbf{b}} \leftarrow \hat{\mathbf{a}} \circ \hat{\mathbf{s}} + \mathsf{NTT}(\mathbf{e})$	$\xrightarrow[1824 \text{ Bytes}]{m_a = \mathsf{encodeA}(seed, \hat{\mathbf{b}})}$	$(\hat{\mathbf{b}}, seed) \leftarrow \mathsf{decodeA}(m_a)$
		$\hat{\mathbf{a}} \leftarrow \mathsf{Parse}(\mathsf{SHAKE\text{-}128}(seed))$
		$\hat{\mathbf{t}} \leftarrow \mathsf{NTT}(\mathbf{s}')$
		$\hat{\mathbf{u}} \leftarrow \hat{\mathbf{a}} \circ \hat{\mathbf{t}} + \mathsf{NTT}(\mathbf{e}')$
		$\nu \xleftarrow{\$} \{0, 1\}^{256}$
		$\nu' \leftarrow \mathsf{SHA3\text{-}256}(\nu)$
		$\mathbf{k} \leftarrow \mathsf{NHSEncode}(\nu')$
		$\mathbf{c} \leftarrow \mathsf{NTT}^{-1}(\hat{\mathbf{b}} \circ \hat{\mathbf{t}}) + \mathbf{e}'' + \mathbf{k}$
$(\hat{\mathbf{u}}, \bar{\mathbf{c}}) \leftarrow \mathsf{decodeB}(m_b)$	$\xleftarrow[2176 \text{ Bytes}]{m_b = \mathsf{encodeB}(\hat{\mathbf{u}}, \bar{\mathbf{c}})}$	$\bar{\mathbf{c}} \xleftarrow{\$} \mathsf{NHSCompress}(\mathbf{c})$
$\mathbf{c}' \leftarrow \mathsf{NHSDecompress}(\bar{\mathbf{c}})$		$\mu \leftarrow \mathsf{SHA3\text{-}256}(\nu')$
$\mathbf{k}' = \mathbf{c}' - \mathsf{NTT}^{-1}(\hat{\mathbf{u}} \circ \hat{\mathbf{s}})$		
$\nu' \leftarrow \mathsf{NHSDecode}(\mathbf{k}')$		
$\mu \leftarrow \mathsf{SHA3\text{-}256}(\nu')$		

Protocol 1: Our proposed NewHope-Simple protocol including NTT and NTT^{-1} computations and sizes of exchanged messages; $\circ$ denotes pointwise multiplication; elements in NTT domain are denoted with a hat ($\hat{}$)

2.3 Failure probability and Bandwidth requirements

The rounding error in each coordinate is at most $\lceil q/16 \rceil$, so for every 4-dim chunk, this rounding error $r \in \mathbb{Z}^4$ verifies $\|r\|_1 \leq q/4 + 4$. This is enough to conclude that it is possible to use the same analysis as NewHope which leads to an error rate bounded by 2^{-60}. In our proposal the bandwidth requirement for the message from Alice to Bob is still 1824 bytes but Bob now has to send 2176 bytes to Alice (an increase from 2048 to 2176 bytes). This stems from the necessity to now send three bits ($1024 \cdot (14 + 3)$ bits = 2176 bytes) for each coefficient of the second element compared to two bits

$(1024 \cdot (14+2)$ bits $= 2048$ bytes) in NewHope. Thus, overall 4000 bytes have to be transferred in a NewHope-Simple key exchange.

2.4 Outlook and Future Work

The implementation in this work is not limited to microprocessors and the changes to the reconciliation can be straightforwardly plugged into microcontroller or hardware implementations of NewHope or Ring-LWE encryption. The flexible implementation of Ring-LWE encryption given in [PG13] can be turned into a NewHope-Simple implementation by setting the appropriate parameters n, q as well as roots of unity for the NTT, a Gaussian parameter that gives an error distribution close to χ, and by implementing the proper encoding and decoding functions[7].

We would also like to remind that the encryption approach re-described in this work can also be applied to standard LWE-based key exchange schemes (e.g., [BCD+16]) in a straightforward manner. Moreover, a combination of the analog error correction used by NewHope and NewHope-Simple with digital error correcting codes (e.g., BCH codes [MS88]) is a possible future work[8]. In [GFS+12] Göttert, Feller, Schneider, Buchmann, and Huss already suggested the usage of Viterbi codes. The study and implementation of lattice-codes for lattice based cryptography was also initiated in [vP16], where the Leech lattice decoder [CN88] is applied to improve by $10\% \sim 20\%$ the bandwidth of a Matrix-LWE encryption scheme [LP11], similar in parameters to Frodo [BCD+16].

References

ADPS15. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange – a new hope. IACR Cryptology ePrint Archive report 2015/1092, 2015. https://eprint.iacr.org/2015/1092/20160329:201913. 8

ADPS16. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange – a new hope. In *Proceedings of the 25th USENIX Security Symposium*. USENIX Association, 2016. http://eprint.iacr.org/2015/1092. 1, 3, 4, 5

BCD+16. Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE. In *Proceedings of the 23rd ACM Conference on Computer and Communications Security (CCS) 2016*. ACM, 2016. http://eprint.iacr.org/2016/659. 7

BGS+08. Christoph Bösch, Jorge Guajardo, Ahmad-Reza Sadeghi, Jamshid Shokrollahi, and Pim Tuyls. Efficient helper data key extractor on FPGAs. In Elisabeth Oswald and Pankaj Rohatgi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2008*, volume 5154 of *LNCS*, pages 181–197. Springer, 2008. 2

BLLN13. Joppe W. Bos, Kristin E. Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In Martijn Stam, editor, *Cryptography and Coding*, volume 8308 of *LNCS*, pages 45–64. Springer, 2013. 4

BPR12. Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, 2012. https://eprint.iacr.org/2011/401. 4

Bra16. Matt Braithwaite. Experimenting with post-quantum cryptography. Posting on the Google Security Blog, 2016. https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html. 1

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. IACR Cryptology ePrint Archive report 2011/344, 2011. Preprint of [BV14]. http://eprint.iacr.org/2011/344. 4

[7] Similar modifications are most likely also possible for other implementations like [CMV+15, RVM+14].

[8] In this case an option would be to map one message bit to two coefficients (instead of four), which would give a message space of 512 bits. Then Bob can choose a 256-bit key and can use up-to 256 bits for redundancy information to allow error correction. This way a $[n, k]$-BCH code with parameters $[511, 259]$ would allow to send a 259-bit key and allow to correct up to 30 errors on Alice's side. We leave the analysis of this approach and fine tuning (e.g., different codes, encoding, and parameters) as future work.

BV14. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM Journal on Computing*, 43(2):831–871, 2014. 7

CMV+15. Donald Donglong Chen, Nele Mentens, Frederik Vercauteren, Sujoy Sinha Roy, Ray C. C. Cheung, Derek Pao, and Ingrid Verbauwhede. High-speed polynomial multiplication architecture for Ring-LWE and SHE cryptosystems. *IEEE Trans. on Circuits and Systems*, 62-I(1):157–166, 2015. 7

CN88. John Conway and Neil J.A. Sloane Neil. *Sphere packings, lattices and groups*, volume 290 of *Grundlehren der mathematischen Wissenschaften*. Springer, 1988. 3, 7

Din12a. Jintai Ding. New cryptographic constructions using generalized learning with errors problem. IACR Cryptology ePrint Archive report 2012/387, 2012. http://eprint.iacr.org/2012/387. 4

Din12b. Jintai Ding. A simple provably secure key exchange scheme based on the learning with errors problem. IACR Cryptology ePrint Archive report 2012/688, version 20121210:115748, 2012. http://eprint.iacr.org/2012/688. 2, 3

DL13. Jintai Ding and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. IACR Cryptology ePrint Archive report 2012/688, versions 20130303:142425 and 20130303:142813, 2013. http://eprint.iacr.org/2012/688. 2, 3

DRS04. Yevgeniy Dodis, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540. Springer, 2004. 2

Duc16. L'eo Ducas. Newhope's reconciliation mechanism explained. Blog post, 2016. https://homepages.cwi.nl/~ducas/weblog/NewhopeRec/index.html. 4

DXL14. Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. IACR Cryptology ePrint Archive report 2012/688, version 20140729:180116, 2014. http://eprint.iacr.org/2012/688. 2, 3

Gab10. Philippe Gaborit. Noisy diffie-hellman protocols. Slides of a talk in the recent results session at the Third International Workshop on Post-Quantum Cryptography – PQCRYPTO 2010, 2010. https://pqc2010.cased.de/rr/03.pdf. 2

GFS+12. Norman Göttert, Thomas Feller, Michael Schneider, Johannes A. Buchmann, and Sorin A. Huss. On the design of hardware building blocks for modern lattice-based encryption schemes. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems - CHES 2012*, volume 7428 of *LNCS*, pages 512–529. Springer, 2012. http://www.iacr.org/archive/ches2012/74280511/74280511.pdf. 3, 7

GLP12. Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems – CHES 2012*, volume 7428 of *LNCS*, pages 530–547. Springer, 2012. https://www.sha.rub.de/media/sh/veroeffentlichungen/2013/02/28/lattice_signature.pdf. 4

GPV08a. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. How to use a short basis: Trapdoors for hard lattices and new cryptographic constructions. In *STOC '08 Proceedings of the fortieth annual ACM symposium on Theory of computing* [GPV08b], pages 197–206. http://web.eecs.umich.edu/~cpeikert/pubs/trap_lattice.pdf (full version of [GPV08b]). 8

GPV08b. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC '08 Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206. ACM, 2008. see also full version [GPV08a]. 8

Ham15. Mike Hamburg. Personnal communication to the authors of [ADPS15], December 2015. 3

HKM+12. Anthony Van Herrewege, Stefan Katzenbeisser, Roel Maes, Roel Peeters, Ahmad-Reza Sadeghi, Ingrid Verbauwhede, and Christian Wachsmann. Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs. In Angelos D. Keromytis, editor, *Financial Cryptography and Data Security*, volume 7397 of *LNCS*, pages 374–389. Springer, 2012. 2

JZ16. Zhengzhong Jin and Yunlei Zhao. Optimal key consensus in presence of noise. arXiv report 1611.06150v1, 2016. https://arxiv.org/abs/1611.06150v1. 4

JZ17. Zhengzhong Jin and Yunlei Zhao. Optimal key consensus in presence of noise. IACR Cryptology ePrint Archive report 2017/1058, 2017. https://eprint.iacr.org/2017/1058. 4

LP11. Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer, 2011. https://eprint.iacr.org/2010/613/. 2, 7

LPR10a. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *LNCS*,

pages 1–23. Springer, 2010. www.iacr.org/archive/eurocrypt2010/66320288/66320288.pdf, see also full version [LPR13]. 2

LPR10b. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. Slides of the talk given by Chris Peikert at Eurocrypt 2010, 2010. http://crypto.rd.francetelecom.com/events/eurocrypt2010/talks/slides-ideal-lwe.pdf. 2, 3

LPR13. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM (JACM)*, 60(6):43:1–43:35, 2013. http://www.cims.nyu.edu/~regev/papers/ideal-lwe.pdf. 2, 3, 9

LSR+15. Zhe Liu, Hwajeong Seo, Sujoy Sinha Roy, Johann Großschädl, Howon Kim, and Ingrid Verbauwhede. Efficient Ring-LWE encryption on 8-bit AVR processors. In Tim Güneysu and Helena Handschuh, editors, *Cryptographic Hardware and Embedded Systems - CHES 2015*, volume 9293 of *LNCS*, pages 663–682. Springer, 2015. https://eprint.iacr.org/2015/410/. 3

MS88. Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error correcting codes*. North-Holland Mathematical Lib, 1988. 7

Pei09a. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem, 2009. https://web.eecs.umich.edu/~cpeikert/pubs/svpcrypto.pdf (full version of [Pei09b]). 4, 9

Pei09b. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC '09 Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 333–342. ACM, 2009. see also full version [Pei09a]. 9

Pei09c. Chris Peikert. Some recent progress in lattice-based cryptography. Slides of an invited tutorial at the Sixth IACR Theory of Cryptography Conference – TCC 2009, 2009. http://www.cc.gatech.edu/fac/cpeikert/pubs/slides-tcc09.pdf. 2

Pei14. Chris Peikert. Lattice cryptography for the Internet. In Michele Mosca, editor, *Post-Quantum Cryptography*, volume 8772 of *LNCS*, pages 197–219. Springer, 2014. http://web.eecs.umich.edu/~cpeikert/pubs/suite.pdf. 2, 3, 4

PG13. Thomas Pöppelmann and Tim Güneysu. Towards practical lattice-based public-key encryption on reconfigurable hardware. In Tanja Lange, Kristin Lauter, and Petr Lisoněk, editors, *Selected Areas in Cryptography – SAC 2013*, volume 8282 of *LNCS*, pages 68–85. Springer, 2013. https://www.ei.rub.de/media/sh/veroeffentlichungen/2013/08/14/lwe_encrypt.pdf. 3, 4, 5, 7

POG15. Thomas Pöppelmann, Tobias Oder, and Tim Güneysu. High-performance ideal lattice-based cryptography on 8-bit ATxmega microcontrollers. In Kristin E. Lauter and Francisco Rodríguez-Henríquez, editors, *Progress in Cryptology – LATINCRYPT 2015*, volume 9230 of *LNCS*, pages 346–365. Springer, 2015. 3

RS10. Markus Rückert and Michael Schneider. Estimating the security of lattice-based cryptosystems. IACR Cryptology ePrint Archive report 2010/137, 2010. http://eprint.iacr.org/2010/137. 3

RVM+14. Sujoy Sinha Roy, Frederik Vercauteren, Nele Mentens, Donald Donglong Chen, and Ingrid Verbauwhede. Compact Ring-LWE cryptoprocessor. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems – CHES 2014*, volume 8731 of *LNCS*, pages 371–391. Springer, 2014. https://eprint.iacr.org/2013/866/. 3, 7

Ste10. Damien Stehlé. Introduction to modern lattice-based cryptography (part ii), June 2010. http://www1.spms.ntu.edu.sg/~ccrg/documents/LBC_Part2_for_printer.pdf. 3

vP16. Alex van Poppelen. Cryptographic decoding of the leech lattice. Master's thesis, Mathematical Institute, Utrecht University, 2016. https://github.com/avanpo/leech-decoding. 7

YHK+11. Yu Yao, Jiawei Huang, Sudhanshu Khanna, Abhi Shelat, Benton Highsmith Calhoun, John Lach, and David Evans. A sub-0.5 V lattice-based public-key encryption scheme for RFID platforms in 130nm CMOS. In Tieyan Li, Chao-Hsien Chu, Ping Wang, and Guilin Wang, editors, *Radio Frequency Identification System Security*, volume 6 of *Cryptology and Information Security Series*, pages 96–113. IOS Press, 2011. http://www.cs.virginia.edu/~evans/pubs/rfidsec11/rfidsec.pdf. 3