Post-quantum Key Exchange for the Internet and the Open Quantum Safe Project

Douglas Stebila[1](✉) and Michele Mosca[2,3,4]

[1] Department of Computing and Software, McMaster University,
Hamilton, ON, Canada
stebilad@mcmaster.ca

[2] Institute for Quantum Computing and Department of Combinatorics
and Optimization, University of Waterloo, Waterloo, ON, Canada
mmosca@uwaterloo.ca

[3] Perimeter Institute for Theoretical Physics, Waterloo, ON, Canada

[4] Canadian Institute for Advanced Research, Toronto, ON, Canada

Abstract. Designing public key cryptosystems that resist attacks by quantum computers is an important area of current cryptographic research and standardization. To retain confidentiality of today's communications against future quantum computers, applications and protocols must begin exploring the use of quantum-resistant key exchange and encryption. In this paper, we explore post-quantum cryptography in general and key exchange specifically. We review two protocols for quantum-resistant key exchange based on lattice problems: BCNS15, based on the ring learning with errors problem, and Frodo, based on the learning with errors problem. We discuss their security and performance characteristics, both on their own and in the context of the Transport Layer Security (TLS) protocol. We introduce the Open Quantum Safe project, an open-source software project for prototyping quantum-resistant cryptography, which includes liboqs, a C library of quantum-resistant algorithms, and our integrations of liboqs into popular open-source applications and protocols, including the widely used OpenSSL library.

1 Introduction

All Internet security protocols that use cryptography, such as the Transport Layer Security (TLS, a.k.a. the Secure Sockets Layer (SSL)) protocol [18] have the same basic structure: *public key cryptography* is used to authenticate the communicating parties to each other and to establish a shared secret key, which

Based on the Stafford Tavares Invited Lecture by D. Stebila.

D. Stebila—Supported in part by Australian Research Council (ARC) Discovery Project grant DP130104304, Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery grant RGPIN-2016-05146, and an NSERC Discovery Accelerator Supplement grant.

M. Mosca—Supported by NSERC, CFI, and ORF. IQC and the Perimeter Institute are supported in part by the Government of Canada and the Province of Ontario.

R. Avanzi and H. Heys (Eds.): SAC 2016, LNCS 10532, pp. 14–37, 2017.
https://doi.org/10.1007/978-3-319-69453-5_2

is then used in *symmetric cryptography* to provide confidentiality and integrity to their communication. The security of most public key cryptosystems depends on the difficulty of solving some mathematical problem, such as factoring large numbers or computing discrete logarithms in finite field or elliptic curve groups. The best known solutions to these problems run in exponential (or sub-exponential) time, making it infeasible for attackers to break the schemes.

Quantum mechanics allows for devices that operate on quantum bits, known as *qubits*, which are two-state quantum systems that can be in any quantum superposition of 0 and 1. Such devices are called quantum computers, and could solve certain types of problems much faster than "classical" (non-quantum) computers. Shor's algorithm [48] could efficiently (i.e., in polynomial time) factor large numbers and compute discrete logarithms, breaking all widely deployed public key cryptosystems. Most symmetric key schemes, such as the Advanced Encryption Standard (AES) cipher, would not be broken by quantum algorithms, although would generally need bigger keys. While large-scale quantum computers do not yet exist, building quantum computers is an active area of research. And Schoelkopf [17] identify seven stages in the development of quantum computers: so far, physicists can perform operations on single and multiple physical qubits, perform non-destructive measurements for error correction, and are making progress on constructing logical memories with longer lifetime than physical qubits; to achieve large-scale quantum computation, we will require the ability to perform operations on single and multiple logical qubits with fault-tolerant computation. Regarding the million-dollar question of when a large-scale quantum computer will be built, in 2015 Mosca [38] stated "I estimate a 1/7 chance of breaking RSA-2048 by 2026 and a 1/2 chance by 2031."

Any attacker who records present-day communications would be able to decrypt it once a quantum computer is built; and there is evidence that governments are storing vast quantities of encrypted Internet traffic. This motivates the urgent use of cryptography that is designed to be safe against quantum attackers—called "post-quantum" or "quantum-safe" or "quantum-resistant" cryptography. In August 2015, the United States National Security Agency (NSA) issued a memo regarding its Suite B cryptographic algorithms for government use, advising that it plans to "transition to quantum resistant algorithms in the not too distant future" [39]. In August 2016, the United States National Institute of Standards and Technology (NIST) launched its post-quantum crypto project[1], a multi-year process with the goal of evaluating and standardizing one or more quantum-resistant public key cryptosystems.

Post-quantum Cryptography. There are several classes of mathematical problems that are conjectured to resist attacks by quantum computers and have been used to construct public key cryptosystems, several of which date from the early days of public key cryptography. These include:

- *Code-based cryptography.* The McEliece public key encryption scheme [36] was one of the first public key schemes, and is based on error-correcting codes, in

[1] http://www.nist.gov/pqcrypto.

particular, the hardness of decoding a general linear code. Niederreiter [40] subsequently proposed a digital signature scheme based on error correcting codes.

- *Hash-based cryptography.* Merkle [37] first proposed the use of hash functions for digitally signing documents; Lamport [30] and Winternitz then showed how to convert Merkle's one-time signature scheme into a many-time signature scheme. These schemes are based entirely on standard hash function properties, and thus are believed to be among the most quantum-resistant. Modern variants include SPHINCS [7] and XMSS [13].
- *Multivariate cryptography.* These cryptosystems are based on the difficulty of solving non-linear, usually quadratic, polynomials, over a field [35,41].
- *Lattice-based cryptography.* Ajtai [1] proposed the first cryptographic schemes directly based on lattices. Regev [46] then introduced the related *learning with errors (LWE) problem*, the security of which is based on lattice problems, and which now forms the basis of a variety of public key encryption and signature schemes [31]. The *ring learning with errors (ring-LWE)* problem [33] uses additional structure which allows for smaller key sizes. Another scheme whose security relates to lattices is the NTRU scheme [26], which also allows for fairly small key sizes.
- *Supersingular elliptic curve isogenies.* One of the newest candidates for quantum-resistant public key cryptography is based on the difficulty of finding isogenies between supersingular elliptic curves [20].

In addition, quantum information can be used directly to create cryptosystems; this is called *quantum cryptography.* For example, quantum key distribution allows two parties to establish a shared secret key using quantum communication and an authenticated classical channel. While this can provide very strong security, it is not yet a candidate for widespread usage since it requires physical infrastructure capable of transmitting quantum states reliably over long distances, so in the rest of this paper we focus solely on quantum-resistant cryptography using classical (non-quantum) computers.

Existing quantum-resistant schemes generally have several limitations. Compared with traditional RSA, finite field, and elliptic curve discrete logarithm schemes, all quantum-resistant schemes have either larger public keys, larger ciphertexts/signatures, or slower runtime. Many quantum-resistant schemes are also based on mathematical problems that are, from a cryptographic perspective, quite new, and thus have received comparably less cryptanalysis. There remain many open questions in post-quantum cryptography, making it an exciting and active research area: the design of better public key encryption and signature schemes with smaller keys and ciphertexts/signatures; improved cryptanalysis leading to better parameter estimates; development of fast, secure implementations suitable for high-performance servers and small embedded devices; and integration into existing network infrastructure and applications.

(It is worth noting that research into post-quantum cryptography is valuable even if large-scale quantum computers are never built: it is possible that the factoring, RSA, or discrete logarithms problems will be solved by some (non-quantum) mathematical breakthrough. Having a diverse family of cryptography assumptions on which we can base public key cryptography protects against such a scenario. Furthermore, the cryptographic agility that will help prepare for a transition to yet-to-be-determined quantum-resistant cryptographic algorithms will also enable the ability to respond quickly to other unexpected weaknesses in cryptographic algorithms.)

This Paper. In this paper, we discuss two research projects in the area of lattice-based key exchange: the “BCNS15” protocol [10] based on the ring-LWE problem, and the “Frodo” protocol [9] based on the LWE problem. We will explain the basic mathematics of these protocols, and our results on the performance of these protocols and their integration into the TLS protocol. We will introduce the Open Quantum Safe project, an open-source software project designed for evaluating post-quantum cryptography candidates and prototyping their use in applications and protocols such as TLS.

This line of work focuses initially on key exchange, with digital signatures to follow closely. As noted above, any attacker who records present-day communications protected using non-quantum-resistant cryptography would be able to decrypt it once a quantum computer is built. This implies that information that needs to remain confidential for many years needs to be protected with quantum-resistant cryptography even before quantum computers exist. In communication protocols like TLS, digital signatures are used to authenticate the parties and key exchange is used to establish a shared secret, which can then be used in symmetric cryptography. This means that, for security against a future quantum adversary, authentication in today’s secure channel establishment protocols can still rely on traditional primitives (such as RSA or elliptic curve signatures), but we should incorporate post-quantum key exchange to provide quantum-resistant long-term confidentiality. This has the benefit of allowing us to introduce new post-quantum ciphersuites in TLS while relying on the existing RSA-based public key infrastructure for certificate authorities. However, applications which require long-term integrity, such as signing contracts and archiving documents, will need to begin considering quantum-resistant signature schemes.

Notation. Let χ be a distribution; $a \overset{\$}{\leftarrow} \chi$ denotes sampling a randomly according to χ. The uniform distribution is denoted by $\mathcal{U}$. Vectors are denoted in lower-case bold, like $\mathbf{a}$; matrices are denoted in upper-case bold, like $\mathbf{A}$. The inner product of two vectors $\mathbf{a}$ and $\mathbf{b}$ is $\langle \mathbf{a}, \mathbf{b} \rangle$. Sampling each component of the length-n vector $\mathbf{a}$ independently at random from χ is denoted by $\mathbf{a} \overset{\$}{\leftarrow} \chi^n$. If $\mathcal{A}$ is a probabilistic algorithm, then $y \overset{\$}{\leftarrow} \mathcal{A}(x)$ denotes running $\mathcal{A}$ on input x with fresh randomness and storing the output in variable y, and $y \overset{\$}{\leftarrow} \mathcal{A}^O(x)$ denotes running $\mathcal{A}$ with oracle access to procedure O.

2 Lattice-Based Cryptography and the LWE Problems

In a seminal 1996 work, Ajtai [1] first proposed a cryptographic construction (in that case, a hash function) that relied on the hardness of a computational problem on lattices (the Short Integer Solution (SIS) problem). A subsequent work by Ajtai and Dwork [2] presented a public key encryption scheme based on another lattice problem. Concurrently, Hoffstein, Pipher, and Silverman [26] created the NTRU public key encryption scheme with can be viewed as involving algebraically structured lattices. A variety of research on the use of lattices in constructing cryptosystems continued during that era, and a detailed chronology is outside the scope of this paper; see one of the many surveys of lattice-based cryptography, such as Peikert's [44].

2.1 The Learning with Errors Problem

In 2005, Regev [46] introduced the *learning with errors (LWE) problem*, showed that LWE is related to the hardness of a lattice problem (the Gap Shortest Vector Problem (GapSVP)), and gave a public key encryption scheme based on LWE. Being a more abstract algebraic problem, LWE can be easier to work with in terms of building cryptosystems, and a large amount of research into the hardness of LWE and its use in cryptography has followed; again, see a survey such as [44] for a detailed chronology.

The search learning with errors problem is like a noisy version of solving a system of linear equations: given a matrix $\mathbf{A}$ and a vector $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$, find $\mathbf{s}$.

Definition 1 (Search LWE problem). *Let n, m, and q be positive integers. Let χ_s and χ_e be distributions over $\mathbb{Z}$. Let $\mathbf{s} \xleftarrow{\$} \chi_s^n$. Let $\mathbf{a}_i \xleftarrow{\$} \mathcal{U}(\mathbb{Z}_q^n), e_i \xleftarrow{\$} \chi_e$, and set $b_i \leftarrow \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \mod q$, for $i = 1, \ldots, m$. The* search LWE problem *for $(n, m, q, \chi_s, \chi_e)$ is to find $\mathbf{s}$ given $(\mathbf{a}_i, b_i)_{i=1}^m$. In particular, for algorithm $\mathcal{A}$, define the advantage*

$$\mathrm{Adv}^{\mathrm{lwe}}_{n,m,q,\chi_s,\chi_e}(\mathcal{A}) = \Pr\big[\mathbf{s} \xleftarrow{\$} \chi_s^n; \mathbf{a}_i \xleftarrow{\$} \mathcal{U}(\mathbb{Z}_q^n); e_i \xleftarrow{\$} \chi_e;$$
$$b_i \leftarrow \langle \mathbf{a}_i, \mathbf{s}_i \rangle + e \bmod q : \mathcal{A}((\mathbf{a}_i, b_i)_{i=1}^m) = \mathbf{s})\big].$$

For appropriate distributions χ_s and χ_e, not only is it conjectured to be hard to find the secret vector $\mathbf{s}$, it is even conjectured that LWE samples $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$ look independent and random: this is the decision LWE problem.

Definition 2 (Decision LWE problem). *Let n and q be positive integers. Let χ_s and χ_e be distributions over $\mathbb{Z}$. Let $\mathbf{s} \xleftarrow{\$} \chi_s^n$. Define the following two oracles:*

- *$O_{\chi_e,\mathbf{s}}$: $\mathbf{a} \xleftarrow{\$} \mathcal{U}(\mathbb{Z}_q^n), e \xleftarrow{\$} \chi_e$; return $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e \bmod q)$.*
- *U: $\mathbf{a} \xleftarrow{\$} \mathcal{U}(\mathbb{Z}_q^n), u \xleftarrow{\$} \mathcal{U}(\mathbb{Z}_q)$; return $(\mathbf{a}, u)$.*

The decision LWE problem *for (n, q, χ_s, χ_e) is to distinguish $O_{\chi,\mathbf{s}}$ from U. In particular, for algorithm $\mathcal{A}$, define the advantage*

$$\mathrm{Adv}^{\mathrm{dlwe}}_{n,q,\chi_s,\chi_e}(\mathcal{A}) = \left|\Pr(\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n : \mathcal{A}^{O_{\chi_e,\mathbf{s}}}() = 1) - \Pr(\mathcal{A}^U() = 1)\right|.$$

Choice of Distributions. The error distribution χ_e is usually a discrete Gaussian distribution of width αq for "error rate" $\alpha < 1$.

LWE was originally phrased involving a uniform distribution on the secret $\mathbf{s}$ ($\chi_s^n = \mathcal{U}(\mathbb{Z}_q^n)$). Applebaum et al. [5] showed that the *short secrets* (or "normal form") variant, in which $\chi_s = \chi_e$, has a tight reduction to the original uniform secrets variant. In what follows, we use the short secrets variant throughout, and abbreviate to a single error distribution χ (using shorthand notation $\mathrm{Adv}^{\mathrm{lwe}}_{n,m,q,\chi}$ and $\mathrm{Adv}^{\mathrm{dlwe}}_{n,q,\chi}$).

Difficulty. Difficulty of both search and decision LWE problems depends on the size of n, m, and q, as well as the distributions χ_s and χ_e. Regev [46] showed that, for appropriate parameters, search and decision LWE are worst-case hard assuming the (average case) hardness of a lattice problem. In particular, he showed first that search-LWE is at least as hard as solving the worst-case lattice problems GapSVP_γ and SIVP_γ (for a parameter γ depending on n and α) using a quantum reduction; then that decision-LWE is at least as hard as the search version using a classical reduction. A sequence of later results have improved various aspects (making the first reduction classical, not quantum; handling more moduli); see Peikert's survey [44, Sect. 4.2.2] for a list.

Extracting Secret Bits. The decision LWE problem effectively yields an element $\langle \mathbf{a}, \mathbf{s} \rangle + e \in \mathbb{Z}_q$ that is indistinguishable from random. Parties using LWE to establish a shared secret for public key encryption (like in Regev's scheme) or key agreement (as we will see in the next section) will only approximately agree on the same value modulo q, so they will have to apply some reconciliation function and extracting a small number of bits (maybe even just 1 bit) from a single element of $\mathbb{Z}_q$. In order to establish a multi-bit shared secret with LWE, the parties will hence need to send many samples, which we can then think of in matrix form: a matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$, and an error $\mathbf{e} \xleftarrow{\$} \chi^n$, to obtain $\mathbf{b} \leftarrow \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m$. This increases communication sizes m-fold, and requires approximately $O(mn \log q)$ bits of communication to obtain an m-bit secret. To reduce communication sizes, one could try to introduce some structure to the matrix $\mathbf{A}$, for example making each row the cyclic shift of the previous row. However, rather than working in matrix form, we can shift our representation to a polynomial ring, leading us to the ring-LWE problem.

2.2 The Ring Learning with Errors Problem

In 2010, Lyubashevsky et al. [34] introduced the ring-LWE problem. Let $R = \mathbb{Z}[X]/\langle X^n + 1 \rangle$, where n is a power of 2. Let q be an integer, and define $R_q = R/qR$, i.e., $R_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$. In other words, R_q consists of polynomials of degree at most $n-1$, with coefficients in $\mathbb{Z}_q$, and the wrapping rule that $X^n \equiv -1 \bmod q$. The search and decision ring-LWE problems are analogues of the corresponding LWE problems, except with ring elements rather than vectors.

Definition 3 (Search ring-LWE problem). *Let n and q be positive integers. Let χ_s and χ_e be distributions over R_q. Let $s \xleftarrow{\$} \chi_s$. Let $a \xleftarrow{\$} \mathcal{U}(R_q), e \xleftarrow{\$} \chi_e$, and set $b \leftarrow as + e$. The* search ring-LWE problem *for (n, q, χ_s, χ_e) is to find s given (a, b). In particular, for algorithm $\mathcal{A}$ define the advantage*

$$\mathrm{Adv}^{\mathrm{rlwe}}_{n,q,\chi_s,\chi_e}(\mathcal{A}) = \Pr\left[s \xleftarrow{\$} \chi_s; a \xleftarrow{\$} \mathcal{U}(R_q); e \xleftarrow{\$} \chi_e; b \leftarrow as + e : \mathcal{A}(a, b) = s\right].$$

Again, for appropriate distributions χ_s and χ_e, not only is it conjectured to be hard to find the secret s, it is even conjectured that ring-LWE samples $(a, as+e)$ look independent and random: this is the decision ring-LWE problem.

Definition 4 (Decision ring-LWE problem). *Let n and q be positive integers. Let χ_s and χ_e be distributions over R_q. Let $s \xleftarrow{\$} \chi_s$. Define the following two oracles:*

- *$O_{\chi_e,s}$: $a \xleftarrow{\$} \mathcal{U}(R_q), e \xleftarrow{\$} \chi_e$; return $(a, as + e)$.*
- *U: $a, u \xleftarrow{\$} \mathcal{U}(R_q)$; return (a, u).*

The decision ring-LWE problem *for (n, q, χ_s, χ_e) is to distinguish $O_{\chi_e,s}$ from U. In particular, for algorithm $\mathcal{A}$, define the advantage*

$$\mathrm{Adv}^{\mathrm{drlwe}}_{n,q,\chi_s,\chi_e}(\mathcal{A}) = \left|\Pr(s \xleftarrow{\$} R_q : \mathcal{A}^{O_{\chi_e,s}}() = 1) - \Pr(\mathcal{A}^{U}() = 1)\right|.$$

Choice of Distributions. The error distribution χ_e is usually a discretized Gaussian distribution in the canonical embedding of R; for an appropriate choice of parameters, we can sample ring elements from χ_e by sampling each coefficient of the polynomial independently from a related distribution.

As with LWE, ring-LWE can be formulated using either a uniform secret ($\chi_s = \mathcal{U}(R_q)$) or with short secrets ($\chi_s = \chi_e$), which has a tight reduction to the original secrets variant. In what follows, we use the short secrets variant throughout, and abbreviate to a single error distribution χ (using shorthand notation $\mathrm{Adv}^{\mathrm{rlwe}}_{n,q,\chi}$ and $\mathrm{Adv}^{\mathrm{drlwe}}_{n,q,\chi}$).

Difficulty. Difficulty of both search and decision ring-LWE depends on the parameters n and q and the distributions χ_s and χ_e. Lyubashevsky et al. [34] showed that search ring-LWE as hard as quantumly solving approximate shortest vector problem on an *ideal* lattice in R; and then the classical search-to-decision reduction applies.

Because of the additional structure present in ring-LWE, the choice of n and q requires greater care than the unstructured LWE problem [45]. There is also the risk that the ring-LWE problem may be easier than the LWE problem. Currently, the best known algorithms for solving hard problems in ideal lattices [14,29] are the same as those for regular lattices (ignoring small polynomial speedups); and in some sieving algorithms, the ideal case enables one to save a small constant factor of time or space [11,47]. Very recently Cramer et al. [16] gave a quantum polynomial time algorithm algorithm for ideal-SVP with certain parameters, but this is not currently applicable to ring-LWE. In summary, some view LWE as a more "conservative" security choice than ring-LWE, though there is no appreciable security difference at present.

Extracting Secret Bits. The decision ring-LWE problem effectively yields a ring element that is indistinguishable from random. Being an element of $R_q = \mathbb{Z}_q[X]/\langle X^n + 1\rangle$, we have n coefficients each of which is an element of $\mathbb{Z}_q$. As with LWE, cryptographic constructions using this will need to reconcile approximately equal shared secrets, and thus can extract only a small number of bits (maybe even just 1 bit) from each coefficient. But since there are n (independent-looking) coefficients, one can extract n random-looking bits from a single ring element. Thus, one needs approximately $O(n \log q)$ bits of communication to obtain an n-bit secret, a substantial reduction compared to LWE. Thus, in practice, one must decide between the decreased communication of ring-LWE versus the potentially more conservative security of LWE.

3 Key Exchange Protocols from LWE and Ring-LWE

Regev [46] was the first to give a public key encryption scheme from the learning with errors problem, and Lyubashevsky et al. [33] were the first to give a public key encryption scheme from ring-LWE. Like ElGamal public key encryption, both these schemes implicitly contain a key encapsulation mechanism and then one-time-mask the KEM shared secret with (an encoded form of) the message. Peikert [42] describes a corresponding approximate LWE key agreement protocol. In 2010, Lindner and Peikert [31] gave an improved LWE-based public key encryption scheme, and a ring-LWE analogue, and described how to view it as an approximate key agreement protocol. This was followed by detailed LWE- and ring-LWE-based key agreement protocols by Ding et al. [19] (including a single-bit reconciliation mechanism to obtain exact key agreement); a sketch of an LWE-based key agreement scheme by Blazy et al. [8, Figs. 1, 2]; and detailed ring-LWE-based key encapsulation mechanisms by Fujioka et al. [22, Sect. 5.2] and Peikert [43] (with an alternative single-bit reconciliation mechanism). In addition to basic unauthenticated key exchange, there have been works on using LWE to create *password-authenticated* key exchange [28] and using ring-LWE to create *authenticated* key exchange [49] (though the security proof of the latter is questioned [24]).

In this section, we will examine two unauthenticated key agreement protocols in which this paper's first author was involved. Frodo [9], an LWE-based key exchange protocol, is an instantiation of the Lindner–Peikert LWE approximate key agreement scheme using a generalization of Peikert's reconciliation mechanism in which multiple bits are extracted from a single element of $\mathbb{Z}_q$. BCNS15 [10], a ring-LWE-based key exchange protocol, is an instantiation of the key exchange scheme corresponding to the KEM in the Lyubashevsky–Piekert–Regev public key encryption scheme from ring-LWE using Peikert's reconciliation mechanism.

3.1 Common Tools: Reconciliation

In both Frodo and BCNS15, the parties will establish an approximately equal shared secret, then exchange some "hints" that allow them to perform a reconciliation operation on the approximately equal shared secret to extract some secret bits that are, with high probability, the same for both parties. The reconciliation technique of Ding et al. [19] sends a single bit "hint" for each key bit and relies on the low-order bits of the shared secret; Peikert's technique [43] also sends a single bit hint but relies on the high-order bits of the shared secret. The explanation below generalizes Peikert's approach [43] to extract multiple bits.

Let $B \in \mathbb{N}$ be the number of bits we aim to extract from one element of $\mathbb{Z}_q$. Assume $B < (\log_2 q) - 1$. Let $\overline{B} = \lceil \log_2 q \rceil - B$. Let $v \in \mathbb{Z}_q$, represented canonically as an integer in $[0, q)$. Define the *rounding* function

$$\lfloor \cdot \rceil_{2^B} : \mathbb{Z}_q \to \mathbb{Z}_{2^B} : v \mapsto \left\lfloor 2^{-\overline{B}} v \right\rceil \bmod 2^B ,$$

where $\lfloor \cdot \rceil : \mathbb{R} \to \mathbb{Z}$ rounds real number x to the closest integer. When q is a multiple of 2^B, $\lfloor \cdot \rceil_{2^B}$ outputs the B most significant bits of $(v + 2^{\overline{B}-1}) \bmod q$, thereby partitioning $\mathbb{Z}_q$ into 2^B intervals of integers with the same B most significant bits (up to a cyclic shift of the values that centres these intervals around 0).

Define the *cross-rounding* function

$$\langle \cdot \rangle_{2^B} : \mathbb{Z}_q \to \mathbb{Z}_2 : v \mapsto \left\lfloor 2^{-\overline{B}+1} v \right\rfloor \bmod 2,$$

where $\lfloor \cdot \rfloor : \mathbb{R} \to \mathbb{Z}$ takes the floor of the real number x. When q is a multiple of 2^{B+1}, $\langle \cdot \rangle_{2^B}$ partitions $\mathbb{Z}_q$ into two intervals based according to their $(B+1)$th most significant bit.

On input of $w \in \mathbb{Z}_q$ and $c \in \{0, 1\}$, the *reconciliation* function $\mathrm{rec}_{2^B}(w, c)$ outputs $\lfloor v \rceil_{2^B}$, where v is the *closest* element to w such that $\langle v \rangle_{2^B} = c$.

If Alice and Bob have approximately equal values $v, w \in \mathbb{Z}_q$, they can use the following process to derive B bits that are, with high probability, equal. Suppose q is a multiple of 2^B. Bob computes $c \leftarrow \langle v \rangle_{2^B}$ and sends c to Alice. Bob computes $k' \leftarrow \lfloor v \rceil_{2^B}$. Alice computes $k \leftarrow \mathrm{rec}_{2^B}(w, c)$.

Security of this technique follows from the following fact: if $v \in \mathbb{Z}_q$ is uniformly random, then $\lfloor v \rceil_{2^B}$ is uniformly random given $\langle v \rangle_{2^B}$.

Correctness follows if v and w are sufficiently close. Namely, if $|v - w \pmod q| < 2^{\overline{B}-2}$, then $\mathrm{rec}_{2^B}(w, \langle v \rangle_{2^B}) = \lfloor v \rceil_{2^B}$. Parameters must be chosen so that v and w are sufficiently close.

For our parameters in the ring setting, we will want to extract 1 bit from each element of $\mathbb{Z}_q$, but q will not be a multiple of 2. Peikert suggested the following technique: Bob computes $\overline{v} \xleftarrow{\$} \mathrm{dbl}(v)$, where $\mathrm{dbl} : \mathbb{Z}_q \to \mathbb{Z}_{2q} : x \mapsto 2x - e$, where e is sampled from $\{-1, 0, 1\}$ with probabilities $p_{-1} = p_1 = \frac{1}{4}$ and $p_0 = \frac{1}{2}$. Bob computes $c \leftarrow \langle \overline{v}/2 \rangle_2$ and sends c to Alice. Bob computes $k' \leftarrow \lfloor \overline{v}/2 \rceil_2$. Alice computes $k \leftarrow \mathrm{rec}_2(2w, c)$.

For ring-LWE, these functions are extended from $\mathbb{Z}_q$ to the ring $R_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ coefficient-wise. For matrix forms of LWE, these functions can be extended to vectors component-wise.

3.2 Ring-LWE-Based Key Exchange: BCNS15

Protocol. The BCNS15 protocol [10], based on the ring-LWE problem, is shown in Fig. 1. Alice and Bob exchange ring-LWE samples $b = as + e$ and $b' = as' + e'$. They can then compute an approximately equal shared secret:

$$sb' = sas' + se' \approx sas' + s'e = bs' \in R_q = \mathbb{Z}_q[X]\langle X^n + 1\rangle .$$

From each coefficient of the approximately equal shared secret, they extract a single secret bit.

Public parameters		
Decision ring-LWE parameters n, q, χ		
$a \xleftarrow{\$} \mathcal{U}(R_q)$		
Alice		**Bob**
$s, e \xleftarrow{\$} \chi$		
$b \leftarrow as + e \in R_q$	$\xrightarrow{b}$	$s', e' \xleftarrow{\$} \chi$
		$b' \leftarrow as' + e' \in R_q$
		$e'' \xleftarrow{\$} \chi$
		$v \leftarrow bs' + e'' \in R_q$
		$\overline{v} \xleftarrow{\$} \mathrm{dbl}(v) \in R_{2q}$
	$\xleftarrow{b', c}$	$c \leftarrow \langle \overline{v}/2 \rangle_2 \in \{0,1\}^n$
$k_A \leftarrow \mathrm{rec}_2(2b's, c) \in \{0,1\}^n$		$k_B \leftarrow \lfloor \overline{v}/2 \rceil_2 \in \{0,1\}^n$

Fig. 1. BCNS15: unauthenticated Diffie–Hellman-like key exchange from ring-LWE

Security. Assuming the decision ring-LWE problem is hard for the parameters chosen, the BCNS15 key exchange protocol is a secure unauthenticated key exchange protocol. The argument follows [31,43] by using two applications of the decision ring-LWE assumption: first, on Alice's computations involving s (so b becomes independent from s), and second on Bob's computations involving s' (so b' and v become independent from s'). This makes the approximately equal shared secret v uniformly random from the adversary's perspective, and as noted above the hint c reveals no information about extracted key k'.

Parameters. The BCNS15 protocol is instantiated with $n = 1024$ and $q = 2^{32} - 1$. The error distribution χ is a discrete Gaussian distribution; because n is a power of 2, this can be achieved by sampling each coefficient from a discrete Gaussian $D_{\mathbb{Z},\sigma}$ with has $D_{\mathbb{Z},\sigma}(x) = \frac{1}{S} e^{-x^2/(2\sigma^2)}$ where $S = 1 + 2\sum_{k=1}^{\infty} e^{-k^2/(2\sigma^2)}$. With these parameters, the probability that reconciliation yields $k \neq k'$ is much less than 2^{-128}. Total communication required for two parties to establish a shared secret is 8,320 bytes.

Based on hardness estimates by Albrecht et al. [3], breaking the system with these parameters would require $2^{163.8}$ operations on a classical computer with at least $2^{94.4}$ memory usage. Assuming a square-root speedup for quantum computers via Grover's algorithm (though it is not known how to achieve a full square-root speedup), this suggests at least $2^{81.9}$ quantum security. Based on the same difficulty estimates for the subsequent NewHope protocol [4], Alkim et al. list BCNS15 as having 86-bit classical security and 78-bit quantum security.

Subsequent Works. Alkim et al. [4] subsequently published the so-called "NewHope" protocol, making several improvements to the BCNS15 protocol. NewHope uses different parameters and a different error distribution (which was easier to sample), resulting in substantially improved performance and smaller communication (3,872 bytes). NewHope also uses a pseudorandomly generated a, rather than a fixed public parameter. In July 2016, Google announced that they were deploying a two-year experiment in the alpha version of their Chrome web browser (called "Canary") that uses the NewHope key exchange protocol in a hybrid ciphersuite with elliptic curve Diffie–Hellman [12]. Further improvements to NewHope have been given by several papers [25,32].

3.3 LWE-Based Key Exchange: Frodo

Protocol. The Frodo key exchange protocol [9], based on the LWE problem, is shown in Fig. 2. It uses a matrix form of the LWE problem: Alice uses m secrets $\mathbf{s}_1, \ldots, \mathbf{s}_m$, represented as a matrix $\mathbf{S}$; similarly for Bob. Alice and Bob exchange matrix LWE samples $\mathbf{B} = \mathbf{AS} + \mathbf{E}$ and $\mathbf{B}' = \mathbf{S}'\mathbf{A}' + \mathbf{E}'$. They can then compute an approximately equal shared secret:

$$\mathbf{B}'\mathbf{S} = \mathbf{S}'\mathbf{AS} + \mathbf{S}'\mathbf{E} \approx \mathbf{S}'\mathbf{AS} + \mathbf{SE}' = \mathbf{S}'\mathbf{B} \in \mathbb{Z}_q^{m \times m}.$$

From each entry of the approximately equal shared secret, they extract B secret bits. Frodo follows NewHope's idea of using a pseudorandomly generated $\mathbf{A}$.

Security. Assuming the decision LWE problem is hard for the parameters chosen, and PRF is a pseudorandom function, the Frodo key exchange protocol is a secure unauthenticated key exchange protocol. A hybrid argument goes from the standard decision-LWE problem to a matrix form of it, then the same argument as for BCNS15 above yields the indistinguishability of the session key.

Parameters. The Frodo paper contains several parameter sets, including a "recommended" parameter set, which uses $n = 752, q = 2^{15}, m = 8$, and $B = 4$. The error distribution χ is a concrete distribution specified in the paper, which is close in Renyi divergence to a rounded continuous Gaussian distribution (but requires fewer bits to sample). With these parameters, the probability that reconciliation yields $\mathbf{k} \neq \mathbf{k}'$ is $2^{-38.9}$. Total communication required for two parties to establish a shared secret is 8,320 bytes. The claimed security level is 140 bits of security against a classical adversary, and 130 bits against a quantum adversary. The paper also includes a higher-security "paranoid" parameter set, which conjectures a certain lower bound on lattice sieving for any adversary.

Public parameters		
Decision LWE parameters n, q, χ; integer m		
Alice		**Bob**
$seed \xleftarrow{\$} \{0,1\}^{\lambda}$		
$\mathbf{A} \leftarrow \mathrm{PRF}(seed) \in \mathbb{Z}_q^{n\times n}$		
$\mathbf{S}, \mathbf{E} \xleftarrow{\$} \chi(\mathbb{Z}_q^{n\times m})$		
$\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E} \in \mathbb{Z}_q^{n\times m}$	$\xrightarrow{b, seed}$	$\mathbf{A} \leftarrow \mathrm{PRF}(seed) \in \mathbb{Z}_q^{n\times n}$
		$\mathbf{S}', \mathbf{E}' \xleftarrow{\$} \chi(\mathbb{Z}_q^{m\times n})$
		$\mathbf{B}' \leftarrow \mathbf{S}'\mathbf{A} + \mathbf{E}' \in \mathbb{Z}_q^{m\times n}$
		$\mathbf{E}'' \xleftarrow{\$} \chi(\mathbb{Z}_q^{m\times m})$
		$\mathbf{V} \leftarrow \mathbf{S}'\mathbf{B} + \mathbf{E}'' \in \mathbb{Z}_q^{m\times m}$
	$\xleftarrow{\mathbf{B}', \mathbf{C}}$	$\mathbf{C} \leftarrow \langle \mathbf{V} \rangle_{2^B} \in \mathbb{Z}_{2^B}^{m\times m}$
$\mathbf{k} \leftarrow \mathrm{rec}_{2^B}(\mathbf{B}'\mathbf{S}, \mathbf{C}) \in \mathbb{Z}_{2^B}^{m}$		$\mathbf{k}' \leftarrow \lfloor \mathbf{V} \rceil_{2^B} \in \mathbb{Z}_{2^B}^{m}$

Fig. 2. Frodo: unauthenticated Diffie–Hellman-like key exchange from LWE

3.4 Performance of Post-quantum Key Exchange

Table 1 (copied from [9]) shows the performance characteristics of several post-quantum key exchange protocols:

- BCNS ring-LWE key exchange, C implementation [10];
- NewHope ring-LWE key exchange, C implementation [4];
- NTRU public key encryption key transport using parameter set `EES743EP1`, C implementation;[2] and
- SIDH (supersingular isogeny Diffie–Hellman) key exchange, C implementation [15].

The table also includes non-quantum-secure algorithms at the 128-bit classical security level for comparison: OpenSSL's implementation of ECDH (on the `nistp256` curve) and RSA with a 3072-bit modulus. Results were measured on a single hardware hyper-thread on a 2.6 GHz Intel Xeon E5 (Sandy Bridge); see [9] for details. Although some implementations included optimizations using the AVX2 instruction set, the computer used for measurements did not support AVX2.

In the table, Alice0 denotes Alice's procedure for constructing her outgoing message, and Alice1 is her procedure for processing Bob's incoming message and deriving the shared secret.

The NewHope protocol has the best computational performance of the post-quantum key exchange algorithms tested, even outperforming traditional RSA and ECDH. However, all structured lattice schemes (ring-LWE and NTRU) have larger communication than RSA and ECDH, around 2–8 KiB round-trip.

[2] https://github.com/NTRUOpenSourceProject/ntru-crypto.

Table 1. Performance of standalone cryptographic operations, showing mean runtime in milliseconds of standalone cryptographic operations, communication sizes (public key/messages) in bytes, and claimed security level in bits. Table from [9].

Scheme	Alice0 (ms)	Bob (ms)	Alice1 (ms)	Communication (bytes)		Claimed security	
				A→B	B→A	Classical	Quantum
RSA 3072-bit	—	0.09	4.49	387/0*	384	128	—
ECDH `nistp256`	0.37	0.70	0.33	32	32	128	—
BCNS	1.01	1.59	0.17	4,096	4,224	86	78
NewHope	0.11	0.16	0.03	1,824	2,048	229	206
NTRU `EES743EP1`	2.00	0.28	0.15	1,027	1,022	256	128
Frodo recomm.	1.13	1.34	0.13	11,377	11,296	144	130
Frodo paranoid	1.25	1.64	0.15	13,057	12,976	177	161
SIDH	135	464	301	564	564	192	128

*In TLS, the RSA public key is already included in the server's certificate message, so RSA key transport imposes no additional communication from server to client.

Unstructured lattice schemes (LWE) also achieve good performance, on the order of 1 ms, but require even more communication, around 22 KiB round-trip. Supersingular isogeny Diffie–Hellman has much smaller keys (1 KiB round-trip, not much larger than RSA 3072), but orders of magnitude slower performance. (Note, however, that the AVX2 optimized implementation of SIDH was an order of magnitude faster than its C implementation). No code-based post-quantum protocol was included in the tests above. In particular, the implementation of Bernstein et al.'s "McBits" high-speed code-based cryptosystem [6] was not publicly available at the time of writing, but their paper reports speeds of 0.005ms (on a 3.4 GHz CPU) for decryption at the 128-bit quantum security level, but at the cost of 216 KiB public keys.

These trade-offs leave no clear post-quantum winner: the smallest key sizes come from SIDH but it has slow performance (though performance usually improves!); ring-LWE gives a decent tradeoff with fast performance and not-too-big keys; LWE's performance remains good, and avoids the use of a structured lattice, but requires larger communication. Though these larger public keys may be too big for embedded devices, it should be remembered that the average webpage is over 1 MB: if we had to switch the Internet to post-quantum cryptography today, the communication costs from post-quantum key exchange would not be much more than an extra emoticon on a webpage.

3.5 From Unauthenticated to Authenticated Key Exchange

Both the BCNS15 and Frodo protocols are for unauthenticated key exchange: they assume the adversary is passive. Of course in practice one must achieve security against an active network adversary. Peikert [43] noted the challenges that are faced in securing LWE and ring-LWE based protocols against an active adversary, and Fluhrer [21] described an explicit attack on ring-LWE protocols that reuse ephemeral key shares against an active adversary. Peikert suggested the use of a transform such as the Fujisaki–Okamoto transform [23] which converts a passively secure (IND-CPA) key encapsulation mechanism (KEM) into an actively secure (IND-CCA) KEM. For integration with TLS, there is also the possibility of using signatures in a signed-DH-like protocol to first authenticate the keyshares; see [10].

4 Integrating Post-quantum Key Exchange into TLS

All the post-quantum key exchange candidates explored in the previous section incur some penalty (either slower computation, or bigger communication, or both) compared to existing RSA or elliptic curve public key cryptography. It is therefore important to understand the impact of these penalties in a practical setting. Both the BCNS15 and Frodo papers integrate the corresponding key exchange scheme into the Transport Layer Security (TLS) protocol, the dominant protocol used securing Internet communications. In particular, they create new TLS version 1.2 ciphersuites which use traditional RSA or ECDSA certificates for signature, but use post-quantum key exchange to derive a shared secret, and then continue to use standard TLS authenticated encryption constructions (e.g., AES in GCM mode). (Due to the message flow in the TLS 1.2 handshake, the TLS server plays the role of "Alice" in the key exchange, and the TLS client plays the role of "Bob".) This is achieved by modifying OpenSSL, a common open-source library for SSL/TLS, which is used by applications such as the Apache httpd web server for securing web server communication.

Hybrid Ciphersuites. The experiments involving post-quantum ciphersuites in TLS also included hybrid ciphersuites, where the TLS handshake uses two key exchange algorithms: one post-quantum algorithm, and one traditional algorithm (in this case, ECDH). While the use of two key exchange algorithms does impact performance, it allows early adopters to retain the (current) security of traditional algorithms like ECDHE while obtaining (potential) security against quantum computers: since many post-quantum algorithms have had comparatively less cryptanalysis, there is an increased chance that parameter sizes for post-quantum algorithms will evolve more rapidly over the next few years in the face of new classical or quantum cryptanalytic advances. Interestingly, Google, in its recent NewHope experiment in Chrome, decided to use solely hybrid ciphersuites [12].

Security. As noted above, BCNS15 and Frodo were shown to be secure *unauthenticated* key exchange protocols, i.e., assuming a passive adversary. For security against an active adversary, we showed in the BCNS paper [10] how to achieve the standard security notion for TLS ("authenticated and confidential channel establishment" (ACCE) [27]) if the server signs both the client and server key share. Note that this would require reordering some of the messages in TLS. An alternative, as noted above, is to use a KEM transform to obtain an actively-secure key exchange protocol.

Table 2. Performance of Apache httpd web server, measured in connections per second, connection time in milliseconds, and handshake size in bytes. Table from [9].

Ciphersuite		Connections/second			Connection time (ms)		Handshake size (bytes)
Key exchange	Signature	1B	10 KiB	100 KiB	w/o load	w/load	
ECDHE `nistp256`	ECDSA	1187	1088	961	14.2	22.2	1, 264
	RSA	814	790	710	16.1	24.7	1,845
BCNS15	ECDSA	922	893	819	18.8	35.8	9,455
	RSA	722	716	638	20.5	36.9	9,964
NewHope	ECDSA	1616	1351	985	12.1	18.6	5,005
	RSA	983	949	771	13.1	20.0	5,514
NTRU `EES743EP1`	ECDSA	725	708	612	20.0	27.2	3,181
	RSA	553	548	512	19.9	29.6	3,691
Frodo Recomm.	ECDSA	923	878	843	18.3	31.5	23,725
	RSA	703	698	635	20.7	32.7	24,228
Hybrid ciphersuites							
BCNS15+ECDHE	ECDSA	736	728	664	23.1	37.7	9,595
	RSA	567	559	503	24.6	40.2	10,177
NewHope+ECDHE	ECDSA	1095	1017	776	16.5	25.2	5,151
	RSA	776	765	686	18.1	28.0	5,731
NTRU+ECDHE	ECDSA	590	578	539	22.5	34.3	3,328
	RSA	468	456	424	24.2	36.8	3,908
Frodo Rec.+ECDHE	ECDSA	735	701	667	22.9	36.4	23,859
	RSA	552	544	516	24.5	39.9	24,439

All TLS ciphersuites used `AES256-GCM` authenticated encryption with `SHA384` in the MAC and KDF. Note that different key exchange methods are at different security levels; see Table 1 for details.

4.1 Performance of Post-quantum Key Exchange in TLS

Table 2 (copied from [9]) shows the performance of a TLS-protected Apache web server using various key exchange mechanisms and signature schemes. It measures:

- *throughput* (connections/second): number of connections per second at the server before server latency spikes, measured with requests of different payload sizes (1 B, 10 KiB, 100 KiB);
- *handshake latency* (milliseconds): time from when client sends first TCP packet till client receives first application data packet, measured on an unloaded server and a loaded server (with sufficiently many connections to achieve 70% CPU load).

Performance was measured on a 4-CPU server with the same CPU as in Sect. 3.4. See [9] for the detailed methodology.

Unsurprisingly, the additional overhead of other cryptographic and network operations in a TLS connection mutes the performance differences between key exchange protocols. For example, while the standalone performance of NewHope is 9× better than that of Frodo recommended, throughput of a NewHope-based ciphersuite is only 1.75× better than Frodo recommended when the server returns 1 byte of application data, and the gap narrows further to just 1.12× when the server returns 100 KiB of application data. Similarly, the latency of a Frodo-based ciphersuite is only 1.5× slower than a NewHope-based ciphersuite. When hybrid ciphersuites are used, the performance difference between slow and fast post-quantum ciphersuites narrows even further.

5 Interlude: Programming is Hard

In the BCNS15 work on ring-LWE-based key exchange, we did a performance evaluation at two levels: the standalone cryptographic operations of the ring-LWE key exchange protocol, and its performance when run in the TLS protocol. The first is a fairly common practice in cryptographic research: implement your algorithms in C, then use some cycle counting or microsecond-accurate timing code to determine the runtime of your algorithms.

Evaluating performance in the TLS protocol is less common due in part to the difficulty of doing so: either one has to implement a network protocol from scratch (which is painful and usually not the main purpose of the research), or integrate the cryptographic algorithms into an existing cryptographic library, such as OpenSSL. These libraries are often quite complex. When we wanted to add our BCNS15 ring-LWE key exchange protocol to OpenSSL for testing purposes, we had to first "wrap" our core ring-LWE library inside of OpenSSL's data structures inside the `crypto` directory, then modify OpenSSL's `ssl` directory to make use of those new data structures. Table 3 shows the number of files and lines of code that were added or changed. While the core ring-LWE library consisted of only 6 (standalone) C files totalling just under 900 lines of code, integrating it into OpenSSL required touching 23 files and changing or adding another 2143 lines of code.

Table 3. Source code changes to add BCNS15 ring-LWE key exchange to OpenSSL

Component	New files	Existing files	Lines of code*
Core ring-LWE library	6	0	896
Ring-LWE "wrapper" for OpenSSL	6	5	1229
SSL integration	0	12	914

*Lines of code excludes Makefiles and automatically generated files, but includes comments and whitespace, and counts both lines added and deleted. Calculated from https://github.com/dstebila/openssl-rlwekex/commit/f80719bf.

For the Frodo work on LWE-based key exchange, we again wanted to evaluate the performance of our algorithms both in a standalone setting and in the context of TLS, but we also wanted to compare with other post-quantum key exchange candidates. Writing 2100 lines of wrapper/integration code for each algorithm we wanted to add was an unappealing prospect. For the Frodo project, we developed an intermediate API that allowed us to more easily integrate different post-quantum key exchange algorithms into OpenSSL for performance comparison. This not-publicly-released intermediate API was the predecessor of and partial motivation for some of the features added to the Open Quantum Safe framework.

6 Open Quantum Safe: A Software Framework for Post-quantum Cryptography

The goal of our Open Quantum Safe (OQS) project (https://openquantumsafe.org) is to support the development and prototyping of quantum-resistant cryptography. OQS consists of two main lines of work: liboqs, an open source C library for quantum-resistant cryptographic algorithms; and prototype integrations into protocols and applications, including the widely used OpenSSL library.

As an example of where the OQS framework can assist with the grand challenge of moving quantum-resistant cryptography towards reliable widespread deployment, consider a small- or medium-sized enterprise that understands the need to integrate quantum-resistant cryptography into its products. Perhaps their products protect information that requires long-term confidentiality. Perhaps their products will be deployed in the field for many years with no easy opportunity for changing the cryptographic algorithms later. Or perhaps they or their customers are worried about the small but non-negligible chance that today's algorithms will be broken, by quantum computers or otherwise, much earlier than expected.

Whatever their reason for wishing to integrate quantum-safe cryptography into their products sooner rather than later, this would not be an easy path for them to take. In-house implementation of quantum-safe primitives requires advanced specialized expertise in order to understand the research literature, choose a suitable scheme, digest the new mathematics, choose suitable parameters, and develop robust software or hardware implementations. This is an enormous, expensive, and risky endeavour to undertake on one's own, especially for a small- or medium-sized enterprise.

Commercially available alternatives, especially back in 2014 when this project started taking shape, were few, and also potentially problematic from a variety of perspectives: cost, patents, transparency, maintenance, degree of external scrutiny, etc.

Companies who would like to offer a quantum-safe option today do not have an easy or robust path for doing so.

OQS gives such organizations the option of prototyping an available quantum-resistant algorithm in their applications. Since these are still largely experimental algorithms that have not yet received the intense scrutiny of the global cryptographic community, our recommendation is to use one of the available post-quantum algorithms in a "hybrid" fashion with a standard algorithm that has received intense scrutiny with respect to classical cryptanalysis and robust implementation.

Since we fully expect that ongoing developments and improvements in the design, cryptanalysis, and implementation of quantum-safe algorithms, OQS is designed so improvements and changes in the post-quantum algorithm can be adopted without major changes to application software.

Organizations who do not wish or need to use open source in their products can still benefit from:

- reference implementations that will guide them in their own implementations
- benchmark information that will guide their choice of algorithm
- the ability to test alternatives in their products before deciding which algorithms to choose.

OQS was thus designed with the goal of both facilitating the prototyping and testing of quantum-resistant algorithms in a range of applications, and of driving forward the implementation, testing, and benchmarking of quantum-resistant primitives themselves.

The high-level architecture of the OQS software project is shown in Fig. 3.

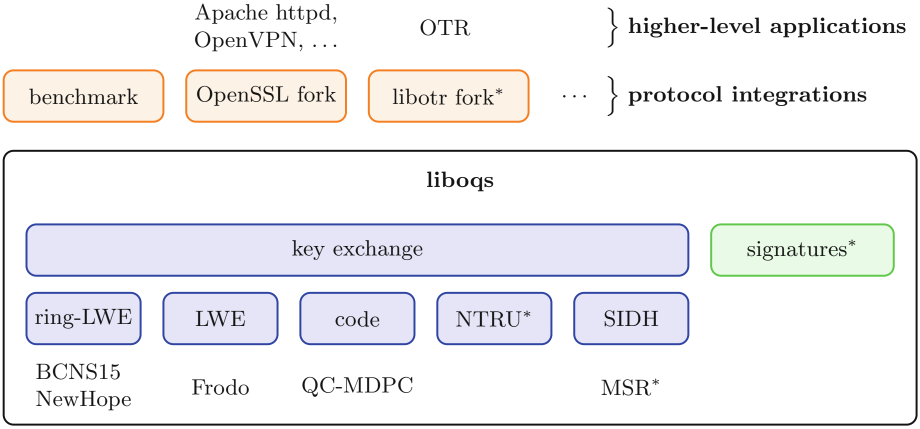

Fig. 3. Architecture of the Open Quantum Safe project. (* denotes future plans.)

6.1 liboqs

liboqs (https://github.com/open-quantum-safe/liboqs) provides a common interface for key exchange and digital signature schemes, as well as implementations of a variety of post-quantum schemes. Some implementations are based on existing open source implementations, either adapting the implementation or putting a thin "wrapper" around the implementation. Other implementations have been written from scratch directly for the library. As of writing, liboqs includes key exchange based on:

- ring-LWE using the BCNS15 protocol (adaptation of existing implementation) [10];
- ring-LWE using the NewHope protocol (wrapper around existing implementation) [4];
- LWE using the Frodo protocol (adaptation of existing implementation) [9];
- error correcting codes – quasi-cyclic medium-density parity-check codes using the Niederreiter cryptosystem (new implementation).

liboqs also includes common routines available to all liboqs modules, including a common random number generator and various symmetric primitives such as AES and SHA-3.

liboqs includes a benchmarking program that enables runtime comparisons of all supported implementations. The library and benchmarking program build and have been tested on Mac OS X 10.11.6, macOS 10.12, and Ubuntu 16.04.1 (using clang or gcc), and Windows 10 (using Visual Studio).

6.2 Application/Protocol Integrations

The OQS project also includes prototype integrations into protocols and applications. Our first integration is into the OpenSSL library,[3] which is an open source cryptographic library that provides both cryptographic functions (`libcrypto`) and an SSL/TLS implementation (`libssl`). OpenSSL is used by many network applications, including the popular Apache httpd web server and the OpenVPN virtual private networking software.

Our OpenSSL fork (https://github.com/open-quantum-safe/openssl) integrates post-quantum key exchange algorithms from liboqs into OpenSSL's `speed` command, and provides TLS 1.2 ciphersuites using post-quantum key exchange based on primitives from liboqs. For each post-quantum key exchange primitive supported by liboqs, there are ciphersuites with AES-128 or AES-256 encryption in GCM mode (with either SHA-256 or SHA-384, respectively), and authentication using either RSA or ECDSA certificates. (We use experimental ciphersuite numbers.)

Each of these four ciphersuites is also mirrored by another four *hybrid* ciphersuites which use both elliptic curve Diffie–Hellman (ECDHE) key exchange *and* the post-quantum key exchange primitive.

Our OpenSSL integration also includes *generic* ciphersuites. liboqs includes interfaces for each key exchange algorithm so it can be selected by the caller at runtime, but it also includes a generic interface that can be configured *at compile time*. Our OpenSSL integration does include ciphersuites for each individual key exchange algorithm in liboqs, but it also includes a set of ciphersuites that call the generic interface, which will then use whatever key exchange algorithm was specified at compile time. This means that a developer can add a new algorithm to liboqs and immediately prototype its use in SSL/TLS without changing a single line of code in OpenSSL, simply by using the generic OQS ciphersuites in OpenSSL and compiling liboqs to use the desired algorithm.

[3] https://www.openssl.org.

6.3 Case Study: Adding NewHope to liboqs and OpenSSL

As mentioned earlier, one of the goals of the Open Quantum Safe project is to make it easier to prototype post-quantum cryptography. It should be easy to add a new algorithm to liboqs, and then easy to use that algorithm in an application or protocol that already supports liboqs.

Recently, we added the NewHope ring-LWE-based key exchange to liboqs and our OpenSSL fork. It is interesting to compare the amount of work required to add NewHope to liboqs and our OpenSSL fork with the figures in Table 3 on adding BCNS15 directly to OpenSSL.

In liboqs, the wrapper around NewHope is 2 new files, totalling 163 lines of code, and requires 5 lines of code to be changed in 2 other files (plus changes in the Makefile).

As noted above, liboqs includes a "generic" key exchange method which can be hard-coded at compile time to any one of its implementations, and our OpenSSL fork already includes a "generic OQS" key exchange ciphersuite that calls liboqs' generic key exchange method. Thus, once NewHope has been added to liboqs, it is possible to test NewHope in OpenSSL with zero changes to the OpenSSL fork via the generic key exchange method and recompiling. However, to explicitly add named NewHope ciphersuites to OpenSSL, we are able to reuse existing data structures, resulting in a diff that touches 10 files and totals 222 lines of code. Moreover, the additions can very easily follow the pattern from previous diffs,[4] making adding a new OQS-based ciphersuite a 15-min job.

7 Conclusion and Outlook

The next few years will be an exciting time in the area of post-quantum cryptography. With the forthcoming NIST post-quantum project, and with continuing advances in quantum computing research, there will be increasing interest from government, industry, and standards bodies in understanding and using quantum-resistant cryptography. Lattice-based cryptography, in the form of the learning with errors and the ring-LWE problems, is particularly promising for quantum-resistant public key encryption and key exchange, offering high computation efficiency with reasonable key sizes. More cryptanalytic research will be essential to increase confidence in any standardized primitive. Since each post-quantum candidate to date has trade-offs between computational efficiency and communication sizes compared to existing primitives, it is also important to understand the how applications and network protocols behave when using different post-quantum algorithms. The Open Quantum Safe project can help rapidly compare post-quantum algorithms and prototype their use in existing protocols and applications, and experiments like Google's use of NewHope in its Chrome Canary browser will give valuable information about how post-quantum cryptosystems behave in real-world deployments.

[4] https://github.com/open-quantum-safe/openssl/commit/cb91c708 and https://github.com/open-quantum-safe/openssl/commit/3a04b822.

For cryptographers interested in designing new public key encryption, digital signature schemes, and key exchange protocols—for cryptanalysts looking to study new mathematical problems—for cryptographic engineers building new systems—and for standards bodies preparing for the future—exciting times lie ahead!

Acknowledgements. Research on LWE and ring-LWE based key exchange discussed in this paper includes joint work with Joppe W. Bos (NXP Semiconductors), Craig Costello (Microsoft Research), Léo Ducas (CWI), Ilya Mironov (Google), Michael Naehrig (Microsoft Research), Valeria Nikolaenko (Stanford University), and Ananth Raghunathan (Google) and was published as [9,10].

The Open Quantum Safe project grew out of discussions with a number of colleagues over the past few years, especially during early 2014, around the challenges of taking post-quantum cryptography closer to widespread practical application. These colleagues include: Scott Vanstone and Sherry Shannon Vanstone (Trustpoint); Matthew Campagna (Amazon Web Services); Alfred Menezes, Ian Goldberg, and Guang Gong (University of Waterloo); William Whyte and Zhenfei Zhang (Security Innovation); as well as the research colleagues in the paragraph above; we are grateful for all the valuable discussions. The Open Quantum Safe project has been supported in part by the Tutte Institute for Mathematics and Computing. Contributors to OQS are listed at https://github.com/open-quantum-safe/liboqs/graphs/contributors and as of writing include Jennifer Fernick, David Jao, Tancrède Lepoint, Shravan Mishra, Christian Paquin, Alex Parent, John Schanck, and Sebastian Verschoor.

D.S. thanks the chairs of SAC 2016 for the invitation to give the Stafford Tavares Invited Lecture at Canada's annual cryptography conference.

References

1. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: 28th ACM STOC, pp. 99–108. ACM Press, May 1996
2. Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: 29th ACM STOC, pp. 284–293. ACM Press, May 1997
3. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Cryptology ePrint Archive, Report 2015/046 (2015). http://eprint.iacr.org/2015/046
4. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - a new hope. In: USENIX Security 2016. USENIX Association, August 2016. https://eprint.iacr.org/2015/1092
5. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 595–618. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03356-8_35
6. Bernstein, D.J., Chou, T., Schwabe, P.: McBits: fast constant-time code-based cryptography. In: Bertoni, G., Coron, J.-S. (eds.) CHES 2013. LNCS, vol. 8086, pp. 250–272. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40349-1_15
7. Bernstein, D.J., et al.: SPHINCS: practical stateless hash-based signatures. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 368–397. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46800-5_15

8. Blazy, O., Chevalier, C., Ducas, L., Pan, J.: Exact smooth projective hash function based on LWE. Cryptology ePrint Archive, Report 2013/821 (2013). http://eprint.iacr.org/2013/821
9. Bos, J.W., Costello, C., Ducas, L., Mironov, I., Naehrig, M., Nikolaenko, V., Raghunathan, A., Stebila, D.: Frodo: take off the ring! practical, quantum-secure key exchange from LWE. In: ACM CCS 2016. ACM Press, October 2016. https://eprint.iacr.org/2016/659
10. Bos, J.W., Costello, C., Naehrig, M., Stebila, D.: Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In: 2015 IEEE Symposium on Security and Privacy, pp. 553–570. IEEE Computer Society Press, May 2015
11. Bos, J.W., Naehrig, M., van de Pol, J.: Sieving for shortest vectors in ideal lattices: a practical perspective. Cryptology ePrint Archive, Report 2014/880 (2014). http://eprint.iacr.org/2014/880
12. Braithwaite, M.: Google security blog: experimenting with post-quantum cryptography, July 2016. https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html
13. Buchmann, J., Dahmen, E., Hülsing, A.: XMSS - a practical forward secure signature scheme based on minimal security assumptions. In: Yang, B.-Y. (ed.) PQCrypto 2011. LNCS, vol. 7071, pp. 117–129. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25405-5_8
14. Chen, Y., Nguyen, P.Q.: BKZ 2.0: better lattice security estimates. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 1–20. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25385-0_1
15. Costello, C., Longa, P., Naehrig, M.: Efficient algorithms for supersingular Isogeny Diffie-Hellman. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 572–601. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53018-4_21
16. Cramer, R., Ducas, L., Wesolowski, B.: Short Stickelberger class relations and application to Ideal-SVP. Cryptology ePrint Archive, Report 2016/885 (2016). http://eprint.iacr.org/2016/885
17. Devoret, M.H., Schoelkopf, R.J.: Superconducting circuits for quantum information: an outlook. Science **339**(6124), 1169–1174 (2013)
18. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. http://www.ietf.org/rfc/rfc5246.txt
19. Ding, J., Xie, X., Lin, X.: A simple provably secure key exchange scheme based on the learning with errors problem. Cryptology ePrint Archive, Report 2012/688 (2012). http://eprint.iacr.org/2012/688
20. Feo, L.D., Jao, D., Plût, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. J. Math. Cryptol. **8**(3), 209–247 (2014)
21. Fluhrer, S.: Cryptanalysis of ring-LWE based key exchange with key share reuse. Cryptology ePrint Archive, Report 2016/085 (2016). http://eprint.iacr.org/2016/085
22. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In: Chen, K., Xie, Q., Qiu, W., Li, N., Tzeng, W.G. (eds.) ASIACCS 13, pp. 83–94. ACM Press, May 2013
23. Fujisaki, E., Okamoto, T.: How to enhance the security of public-key encryption at minimum cost. In: Imai, H., Zheng, Y. (eds.) PKC 1999. LNCS, vol. 1560, pp. 53–68. Springer, Heidelberg (1999). doi:10.1007/3-540-49162-7_5

24. Gong, B., Zhao, Y.: Small field attack, and revisiting RLWE-based authenticated key exchange from Eurocrypt'15. Cryptology ePrint Archive, Report 2016/913 (2016). http://eprint.iacr.org/2016/913
25. Gueron, S., Schlieker, F.: Speeding up R-LWE post-quantum key exchange. Cryptology ePrint Archive, Report 2016/467 (2016). http://eprint.iacr.org/2016/467
26. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: a ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998). doi:10.1007/BFb0054868
27. Jager, T., Kohlar, F., Schäge, S., Schwenk, J.: On the security of TLS-DHE in the standard model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 273–293. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32009-5_17
28. Katz, J., Vaikuntanathan, V.: Smooth projective hashing and password-based authenticated key exchange from lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 636–652. Springer, Heidelberg (2009). doi:10.1007/978-3-642-10366-7_37
29. Laarhoven, T.: Sieving for shortest vectors in lattices using angular locality-sensitive hashing. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 3–22. Springer, Heidelberg (2015). doi:10.1007/978-3-662-47989-6_1
30. Lamport, L.: Constructing digital signatures from a one-way function. Technical report SRI-CSL-98, SRI International Computer Science Laboratory, October 1979
31. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011). doi:10.1007/978-3-642-19074-2_21
32. Longa, P., Naehrig, M.: Speeding up the number theoretic transform for faster ideal lattice-based cryptography. Cryptology ePrint Archive, Report 2016/504 (2016). http://eprint.iacr.org/2016/504
33. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). doi:10.1007/978-3-642-13190-5_1
34. Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-LWE cryptography. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 35–54. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38348-9_3
35. Matsumoto, T., Imai, H.: Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In: Barstow, D., et al. (eds.) EUROCRYPT 1988. LNCS, vol. 330, pp. 419–453. Springer, Heidelberg (1988). doi:10.1007/3-540-45961-8_39
36. McEliece, R.: A public-key cryptosystem based on algebraic coding theory. DSN Progress Report 42–44, January and February 1978. http://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF
37. Merkle, R.C.: A certified digital signature. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 218–238. Springer, New York (1990). doi:10.1007/0-387-34805-0_21
38. Mosca, M.: Cybersecurity in an era with quantum computers: will we be ready? Cryptology ePrint Archive, Report 2015/1075 (2015). http://eprint.iacr.org/2015/1075
39. National Security Agency: NSA Suite B cryptography: Cryptography today, August 2015. https://www.nsa.gov/ia/programs/suiteb_cryptography/
40. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Prob. Control Inf. Theory. Problemy Upravlenija i Teorii Informacii **15**, 159–166 (1986)

41. Ong, H., Schnorr, C.P.: Signatures through approximate representation by quadratic forms. In: Chaum, D. (ed.) CRYPTO 1983, pp. 117–131. Plenum Press, New York (1983)
42. Peikert, C.: Some recent progress in lattice-based cryptography. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 72–72. Springer, Heidelberg (2009). doi:10.1007/978-3-642-00457-5_5
43. Peikert, C.: Lattice cryptography for the internet. In: Mosca, M. (ed.) PQCrypto 2014. LNCS, vol. 8772, pp. 197–219. Springer, Cham (2014). doi:10.1007/978-3-319-11659-4_12
44. Peikert, C.: A decade of lattice cryptography. Found. Trends Theor. Comput. Sci. **10**(4), 283–424 (2016)
45. Peikert, C.: How (not) to instantiate ring-LWE. Cryptology ePrint Archive, Report 2016/351 (2016). http://eprint.iacr.org/2016/351
46. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC, pp. 84–93. ACM Press, May 2005
47. Schneider, M.: Sieving for shortest vectors in ideal lattices. In: Youssef, A., Nitaj, A., Hassanien, A.E. (eds.) AFRICACRYPT 2013. LNCS, vol. 7918, pp. 375–391. Springer, Heidelberg (2013). doi:10.1007/978-3-642-38553-7_22
48. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: 35th FOCS, pp. 124–134. IEEE Computer Society Press, November 1994
49. Zhang, J., Zhang, Z., Ding, J., Snook, M., Dagdelen, Ö.: Authenticated key exchange from ideal lattices. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 719–751. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46803-6_24