Overview of post-quantum public-key cryptosystems for key exchange

Annabell Kuldmaa
Supervised by Ahto Truu

December 15, 2015

Abstract

In this report we review four post-quantum cryptosystems: the ring learning with errors key exchange, the supersingular isogeny key exchange, the NTRU and the McEliece cryptosystem. For each protocol, we introduce the underlying mathematical assumption, give overview of the protocol and present some implementation results. We compare the implementation results on 128-bit security level with elliptic curve Diffie-Hellman and RSA.

1 Introduction

The aim of post-quantum cryptography is to introduce cryptosystems which are not known to be broken using quantum computers. Most of today's public-key cryptosystems, including the Diffie-Hellman key exchange protocol, rely on mathematical problems that are hard for classical computers, but can be solved on quantum computers using Shor's algorithm. In this report we consider replacements for the Diffie-Hellmann key exchange and introduce several quantum-resistant public-key cryptosystems.

In Section 2 the ring learning with errors key exchange is presented which was introduced by Peikert in 2014 [1]. We continue in Section 3 with the supersingular isogeny Diffie–Hellman key exchange presented by De Feo, Jao, and Plut in 2011 [2]. In Section 5 we consider the NTRU encryption scheme first described by Hoffstein, Piphe and Silvermain in 1996 [3]. We conclude in Section 6 with the McEliece cryptosystem introduced by McEliece in 1978 [4]. As NTRU and the McEliece cryptosystem are not originally designed for key exchange, we also briefly explain in Section 4 how we can construct key exchange from any asymmetric encryption scheme.

For each scheme we first introduce notations and fundamental concepts that are needed to describe the cryptosystem and understand the underlying assumption which is considered to be hard for a quantum adversary. We present the problem and give a description for the cryptosystem. Note that we only give the most important definitions and concepts and refer the reader to relevant sources for more details. This is followed by some results on performance of these protocols on classical computers. We mainly focus on 128-bit security level in order to compare the results with other currently widely used public-key cryptosystems such as RSA and elliptic curve Diffie-Hellman.

2 Ring Learning with Errors Key Exchange

In this Section the notation and terminology is mostly as in [1] and [5] . Also, we refer the reader to [6] for fundamentals of finite fields.

2.1 The Ring Learning with Errors Problem

Let $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ be the quotient ring for any integer $q \geq 1$. Let $R = \mathbb{Z}[x]/\Phi_m(x)$ denote the m-th cyclotomic ring, i.e. $\Phi_m(x) = x^n + 1 \in \mathbb{Z}[x]$ is the m-th cyclotomic polynomial where $n = 2^l, \quad l > 0$ and $m = 2n$. As above, let $R_q = R/qR$ denote the quotient ring for any integer $q \geq 1$.

Let χ be a probability distribution over R_q, then $x \leftarrow \chi$ denotes sampling $x \in R_q$ according to χ. Also, for a set S, let $\mathcal{U}(S)$ denote the uniform distribution on S and $x \leftarrow \mathcal{U}(S)$ sampling x uniformly at random from S.

We now define ring learning (R-LWE) distribution and the decisional version of the problem.

Definition 2.1 (R-LWE Distribution) *[1] For an $s \in R$ and a distribution χ over R, a sample from the R-LWE distribution $A_{s,\chi}$ over $R_q \times R_q$ is generated by choosing $a \leftarrow \mathcal{U}(R_q)$ uniformly at random, choosing $e \leftarrow \chi$, and outputting $(a, a \cdot s + e)$.*

Definition 2.2 (Decisional R-LWE) *[1] The decision version of the R-LWE problem is to distinguish with non-negligible advantage between independent samples from $A_{s,\chi}$, where χ is chosen once and for all, and the same number of uniformly random and independent samples from $R_q \times R_q$.*

The problem is presented in normal form, i.e. the secret s is chosen from the error distribution instead of the uniform distribution over R_q. The hardness proof for the problem above can be found in [1]. The general idea of the proof is to reduce the R-LWE problem to the Shortest Vector Problem in an Ideal Lattice which is considered to be $\mathcal{NP}$-hard.

2.2 The Key Exchange

We start with defining the rounding and reconciliation functions and conclude with describing the key exchange protocol based on the R-LWE problem.

Let $\lfloor \cdot \rceil : \mathbb{R} \to \mathbb{Z}$, denote the common rounding of reals to integers, i.e. $\lfloor x \rceil = z$ for $x \in [z - \frac{1}{2}, z + \frac{1}{2})$.

Definition 2.3 *Let q be a positive integer. Define the modular rounding function*

$$\lfloor \cdot \rceil_{q,2} : \mathbb{Z}_q \to \mathbb{Z}_2, \quad x \mapsto \lfloor x \rceil_{q,2} = \lfloor \frac{2}{q} x \rceil \mod 2,$$

and the cross-rounding function

$$\langle \cdot \rangle_{q,2} : \mathbb{Z}_q \to \mathbb{Z}_2, \quad x \mapsto \langle x \rangle_{q,2} = \lfloor \frac{4}{q} x \rceil \mod 2.$$

Note that both functions defined above can be extended to elements of R_q coefficient-wise, i.e. for $f(x) = f_{n-1}x^{n-1} + \ldots + f_1 x + f_0$, define

$$\lfloor f(x) \rceil_{q,2} = (\lfloor f_{n-1} \rceil_{q,2}, \ldots, \lfloor f_1 \rceil_{q,2}, \lfloor f_0 \rceil_{q,2}),$$
$$\langle f(x) \rangle_{q,2} = (\langle f_{n-1} \rangle_{q,2}, \ldots, \langle f_1 \rangle_{q,2}, \langle f_0 \rangle_{q,2}).$$

Note that if the modulus q is odd, in order to avoid bias in the derived bits, the reconciliation mechanism must work in $\mathbb{Z}_{2q}$ instead of $\mathbb{Z}_q$. Thus, as we follow [5] where q is defined to be odd, we introduce the randomized doubling function.

Let $\mathrm{dbl}(v) : \mathbb{Z}_q \to \mathbb{Z}_{2q}, \quad x \mapsto \mathrm{dbl}(x) = 2x - e$, where e is sampled from $\{-1, 0, 1\}$ with probabilities $P_{-1} = P_1 = \frac{1}{4}$ and $P_0 = \frac{1}{2}$.

Next, let us define reconciliation function to recover $\lfloor v \rceil_{q,2}$ from an element $w \in \mathbb{Z}_q$, given w and $\langle v \rangle_{q,2}$. Define $I_0 = \{0, 1, \ldots, \lfloor \frac{q}{2} \rceil - 1\}$, $I_1 = \{\lfloor \frac{q}{2} \rceil, \ldots, -1\}$ and $E = [-\frac{q}{4}, \frac{q}{4})$. The reconciliation function rec : $\mathbb{Z}_{2q} \times \mathbb{Z}_2 \to \mathbb{Z}_2$ is defined as follows:

$$\mathrm{rec}(w, b) = \begin{cases} 0 & \text{if } xw \in I_b + E \mod 2q; \\ 1 & \text{else.} \end{cases}$$

The unauthenticated key exchange protocol based on R-LWE is presented as follows:

Public parameters : $a \leftarrow \mathcal{U}(R_q)$	q, n, χ	
Alice		Bob
$s, e \leftarrow \chi$		$s', e' \leftarrow \chi$
$b \leftarrow as + e \in R_q$		$b' \leftarrow as' + e' \in R_q$
	$\xrightarrow{b}$	
		$e'' \leftarrow \chi$
		$v \leftarrow bs' + e'' \in R_q$
		$v^* \leftarrow \mathrm{dbl}(v) \in R_{2q}$
		$c \leftarrow \langle v^* \rangle_{2q,2} \in \{0,1\}^n$
	$\xleftarrow{b', c}$	
$k_A \leftarrow \mathrm{rec}(2b's, c) \in \{0,1\}^n$		$k_B \leftarrow \lfloor v^* \rceil_{2q,2} \in \{0,1\}^n$

The correctness and detailed description of the protocol presented can be found from [5].

Note that in order to have an exact key exchange protocol, error correction is applied. Also, the distribution χ is usually a discrete Gaussian distribution on R with parameter σ. In [5] it is advised to use the following parameters to provide at least 128-bit security against non-quantum adversary: $n = 1024$, $q = 2^{32} - 1$, $\sigma = \frac{8}{\sqrt{2\pi}}$.

The resilience against quantum adversaries is not that clear. Recall that the security lies in finding a short vector of certain length in a corresponding lattice and the Grover's search algorithm is considered to give a square-root speed up for that. If this is the case, quantum adversary would require approximately 2^{80} steps to solve that problem given the parameters above.

2.3 Implementation and Performance

In [5] the protocol was implemented in C. The tests performed ran on two computers. The client computer had an Intel i5 processor with four cores running at 2.7 GHz each and the server computer had an Intel Core 2 Duo processor with two cores running at 2.33 GHz each. The implementation is aimed at the 128-bit level against classical adversaries to be comparable with the currently most widely used elliptic curve in TLS. We refer the interested reader to [5] for detailed description of implementation and standalone mathematical operations and in the following give a brief overview of protocol's overall performance. Note that by far the most expensive operation is sampling from the error distribution taking approximately 1,041,700 cycles.

The total R-LWE key exchange protocol running time integrated into OpenSSL is 1.4 ms on the client and 2.1 ms on the server. Recall that the parameters used were as follows: $n = 1024$, $q = 2^{32} - 1$ and $\sigma = \frac{8}{\sqrt{2\pi}}$.

3 Supersingular Isogeny Key Exchange

This Section mostly relies on [2] where interested readers can find further information. Also, the fundamentals of elliptic curves and isogenies are well covered in [7].

3.1 The Supersingular Decision Diffie-Hellman Problem

Let E be an elliptic curve defined over a finite field $\mathbb{F}_\mathrm{q}$. If $\mathrm{char}(\mathbb{F}_\mathrm{q}) \notin \{2,3\}$, an elliptic curve E can be written in the following form:

$$E : y^2 = x^3 + ax + b.$$

This will always be the case for the curves considered in the rest of this Section.

Note that two elliptic curves are isomorphic over $\mathbb{F}_\mathrm{q}$ if and only if they have the same j-invariant where j-invariant of E is defined by

$$j(E) = 1728 \frac{4a^3}{4a^3 + 27b^2}.$$

The n-torsion group $E[n]$ is the set of all points $P \in \bar{\mathbb{F}}_q$ such that nP is the identity where $\bar{\mathbb{F}}_q$ is the algebraic closure of $\mathbb{F}_\mathrm{q}$.

Let E_1 and E_2 be elliptic curves defined over a finite field $\mathbb{F}_\mathrm{q}$. An isogeny from E_1 to E_2 is a map $\phi : E_1 \to E_2$ that is defined using rational functions and that sends the zero point on E_1 to the zero point on E_2. Isogenies from E_1 to E_2 are determined up to isomorphism by their kernels, i.e. for any isogeny ϕ from E_1 to E_2, the group $\ker \phi$ is a finite subgroup of E_1. Finite subgroups of E_1 can be specified by identifying the set of generators and from which the corresponding isogeny can be computed using Vélu's formulas [8].

Note that the key exchange uses supersingular elliptic curves. An elliptic curve is supersingular if the rank of its endomorphism ring is 4.

Let $p = l_A^{k_A} l_B^{k_B} f \pm 1$ be a prime. Fix a supersingular elliptic curve E_0 over $\mathbb{F}_{\mathrm{p}^2}$ with bases $\{P_A, Q_A\}$ of $E_0[l_A^{k_A}]$ and $\{P_B, Q_B\}$ of $E_0[l_B^{k_B}]$.

Let $\phi_A : E_0 \rightarrow E_A$ be an isogeny with $\ker \phi_A = \langle m_A P_A + n_A Q_A \rangle$ and $\phi_B : E_0 \rightarrow E_B$ be an isogeny with $\ker \phi_B = \langle m_B P_B + n_B Q_B \rangle$ where m_A, n_A m_B, n_B are chosen at random from $\mathbb{Z}/l_A^{k_A}\mathbb{Z}$ and $\mathbb{Z}/l_B^{k_B}\mathbb{Z}$, respectively, and $m_A \nmid l_A$, $n_A \nmid l_A$, $m_B \nmid l_B, n_B \nmid l_B$.

Definition 3.1 (Supersingular Decision Diffie-Hellman (SSDDH) problem) *[2]*
The decision version of the SSDDH problem is to distinguish with non-negligible advantage between samples from the following two distributions:

$$(E_A, E_B, \phi_A(P_B), \phi_A(Q_B), \phi_B(P_A), \phi_B(Q_A), E_{AB})$$
$$(E_A, E_B, \phi_A(P_B), \phi_A(Q_B), \phi_B(P_A), \phi_B(Q_A), E_C)$$

where $E_{AB} \cong E_0/\langle m_A P_A + n_A Q_A, m_B P_B + n_B Q_B \rangle$ and $E_C \cong E_0/\langle m'_A P_A + n'_A Q_A, m'_B P_B + n'_B Q_B \rangle$ where m'_A, n'_A m'_B, n'_B are chosen following the analogue of m_A, n_A m_B, n_B.

The SSDDH problem relies on computing an isogeny between isogenous supersingular elliptic curves. The hardness of the security assumption is further discussed in [2]. Note that the best known attack for the problem above is $O(\sqrt[4]{p})$ with a classical computer and $O(\sqrt[6]{p})$ with a quantum computer.

3.2 The Key Exchange

The high-level description of the key exchange protocol using isogenies on supersingular curves is presented as follows:

Public parameters :	$E_0, \{P_A, Q_A\}, \{P_B, Q_B\}$	
Alice		Bob
$m_A, n_A \leftarrow \mathbb{Z}/l_A^{k_A}\mathbb{Z}$		$m_B, n_B \leftarrow \mathbb{Z}/l_B^{k_B}\mathbb{Z}$
$\phi_A \leftarrow E_0/\langle m_A P_A + n_A Q_A \rangle$		$\phi_B \leftarrow E_0/\langle m_B P_B + n_B Q_B \rangle$
	$\xrightarrow{E_A, \phi_A(P_B), \phi_A(Q_B)}$	
	$\xleftarrow{E_B, \phi_B(P_A), \phi_B(Q_A)}$	
$E_{AB} \leftarrow E_B/\langle m_A \phi_B(P_A) +$ $+ n_A \phi_B(Q_A) \rangle$		$E_{BA} \leftarrow E_A/\langle m_B \phi_A(P_B)$ $+ n_B \phi_A(Q_B) \rangle$
$K_A \leftarrow j(E_{AB})$		$K_B \leftarrow j(E_{BA})$

The details of the protocol and the computations done in each step can be found from [2]. Following the hardness of the underlying assumption, it is suggested to use a 768-bit prime p for 128-bit security.

Note that one of the main advantages of the protocol is that it provides forward secrecy.

3.3 Implementation and Performance

The details of the original implementation of the key exchange protocol based on supersingular elliptic curve isogeny can be found in [2]. In the following we also consider the results introduced in [9]. The improved results come from a pure C implementation which is approximately 20% faster than original one in C/Cython/Python/Sage.

The tests ran on Mac OS Macbook Pro Intel Core i5 2.4 GHz. Using the original implementation the key-exchange for $l_A = 2$ and $l_B = 3$ at 128-bit security level with $p = 2 \cdot 2^{386} \cdot 3^{242} - 1$ took 303 ms. The pure C version took 226 ms, which was further reduced to 217 ms using assembly-language implementations of field addition and multiplication.

4 Key Exchange from Asymmetric Public-Key Cryptography

Let $Enc()$ denote the asymmetric public-key encryption algorithm, $Dec()$ the corresponding decryption algorithm and $KeyGen()$ the key generation algorithm.

The key can be exchanged securely by encrypting the secret key with receiver's public key. The high-level description can be presented as the following protocol.

Alice		Bob
$(sk_A, pk_A) \leftarrow KeyGen()$		
	$\xrightarrow{pk_A}$	
		$k_B \leftarrow \{0,1\}^n$
		$c \leftarrow Enc(pk_A, k_B)$
	$\xleftarrow{c}$	
$k_A \leftarrow Dec(sk_A, c)$		

5 NTRU

The original construction of the NTRU cryptosystem NTRUEncrypt is patented and accepted to IEEE P1363 standards under the specifications for lattice based cryptography [10].

In the following we describe a provably secure variant of the public key cryptosystem NTRU which was introduced in 2011 by Stehle and Steinfield [11].

5.1 Provably Secure NTRU

Define $\Phi = x^n + 1$ with $n = 2^k \geq 8$. Then we have $R = \mathbb{Z}[x]/\Phi$ and $R_q = R/qR$ where $q \geq 5$ is a prime such that Φ has n distinct linear factors modulo q.

n, p, q, σ and α are the parameters of the encryption scheme. $p \in R_q^*$, where R_q^* denotes the set of invertible elements of R_q, defines the message space as $M = R/pR$. For example, we can have $p = 2$ or $p = 3$. Parameter α is defined to be the R-LWE noise parameter and σ is the standard deviation of the discrete Gaussian distribution. For simplicity, in the following we consider α as a parameter of some distribution ρ described in [11].

Let χ be the discrete Gaussian distribution with parameter σ. Recall that here $x \leftarrow \chi$ denotes sampling $x \in R_q$ according to χ.

Key Generation

- Pick a random polynomial $f' \leftarrow \chi$ and compute $f = p \cdot f' + 1$. Resample if $f \bmod q \notin R_q^*$.
- Pick a random polynomial $g \leftarrow \chi$. Resample if $g \bmod q \notin R_q^*$.
- Compute $h \equiv pgf^{-1} \pmod{q}$.
- The public key is h and the private key is f.

Note that in the original construction f and g are sampled uniformly at random.

Encryption

- To encrypt a plaintext $m \in M$, choose random $r, r' \leftarrow \rho$ and compute the ciphertext using public key h as follows

$$c \equiv hr + pr' + m \pmod{q}.$$

Decryption

- To decrypt the ciphertext c, compute

$$m \equiv f \cdot c \pmod{p}.$$

We refer the interested reader to [11] for complete description of the cryptosystem including analysis of the security and comparison with the original NTRUEncrypt.

5.2 The Underlying Assumption

The underlying hard mathematical assumption for the original NTRU encryption scheme is considered to be the problem of finding the shortest vector in a lattice. Unfortunately, there is no complete reduction proof available. Thus, we consider the provable variant of NTRU encryption scheme.

The provable variant of NTRU cryptosystem is IND-CPA secure under decisional R-LWE problem 2.2. The proof for that can be found from [11]. Note that in the security proof the R-LWE problem is defined in a modified form.

It can be shown that if sampling f and g form discrete Gaussian distribution with a large parameter σ, the public key h is statistically close to uniform.

5.3 Implementation and Performance

We begin with some results on performance of NTRU cryptosystem published in the eBATS benchmarking system [12] in order to compare them with the McEliece cryptosystem in Section 6.3.

The measurements were done on a computer which had Intel Core i5-2400 processor with four cores running at 3.1 GHz each. Note that the measured NTRU implementation was not pure NTRU encryption, but hybrid encryption. Further details and the source code is available [13]. For 128-bit security, the following parameters were used: $n = 439$ and $q = 2048$.

Consider encryption of 59-byte message. The key generation took 631,232, encryption 71,124 and decryption 75,292 cycles. Thus, implementing the key exchange would take at least 0.251 ms. The length of the secret key is 659 bytes and the public key is 609 bytes.

For provable variant of NTRU cryptosystem we present the results published in [14]. All experiments performed ran on a Sun XFire 4440 server with 16 Quad-Core AMD Opteron Processor 8356 CPUs running at 2.3 GHz each. The experiments used only one of the cores.

The running time of provably secure variant of NTRU is significantly longer compared to the original version. Consider the following parameters: $n = 2048$, $\log q = 77.28$, $\log \sigma = 53.63$ and $\alpha = \sqrt{\frac{2n}{\pi}}$. These parameters provide security level 2^{144}. The running time of key generation, encryption and decryption were 1731 ms, 12.09 ms and 9.51 ms, respectively.

6 McEliece Cryptosystem

In this section we briefly survey the McEliece cryptosystem. We mainly follow [15] and [16]. For the basic concepts of coding theory, we refer the reader to [17].

6.1 McEliece Encryption Algorithm

The public-key cryptosystem by McEliece introduced in [4] is based on error-correcting codes. The public key is a hidden generator matrix of a binary linear code of length n and dimension k with error correcting capability t. We present the cryptosystem using binary Goppa Codes as in the original construction by McEliece.

Let C be a binary linear code of length n and dimension k, denoted $[n, k]$ in the following. A generator matrix of C is a $k \times n$ matrix G such that $C = \{xG : \ x \in \mathbb{F}_2^{\mathrm{k}}\}$. Also, recall that the weight of an element $c \in \mathbb{F}_2^{\mathrm{n}}$ is the number of non-zero entries of c and distance between two codewords is the Hamming distance between them.

Let $\mathbb{F}_{2^{\mathrm{m}}}$ be a binary field. Choose a labeling $L = \{\alpha_i : \ 0 \leq i \leq 2^m - 1\}$ of the field and an irreducible polynomial $f(x) \in \mathbb{F}_{2^{\mathrm{m}}}[x]$ of degree t. Then, $c = (c_0, \ldots, c_{2^m-1})$ is an irreducible Goppa code $\Gamma(L, f(x))$ if and only if

$$\sum_{i=0}^{2^m-1} c_i(x - \alpha_i)^{-1} \equiv 0 \pmod{f(x)}.$$

Since $\mathbb{F}_{2^{\mathrm{m}}}[x]/f(x)$ is a field, $(x - \alpha_i)^{-1}$ always exists.

Key Generation

- Pick a random $[n, k]$-linear code over $\mathbb{F}_2$ having an efficient decoding algorithm D that can correct up to t errors.

- Compute a $k \times n$ generator matrix G of C.
- Generate a random $k \times k$ non-singular matrix S and $n \times n$ permutation matrix P.
- Compute the matrix $H = SGP$.
- The public key is the matrix H and the private key is (S, G, P, D).

Encryption

- To encrypt a plaintext $m \in \{0,1\}^k$, choose a random $r \in \{0,1\}^n$ of weight t and compute the ciphertext using the public key H as follows
$$c = mH + r.$$

Decryption

- To decrypt the ciphertext c, first compute cP^{-1} and apply decoding algorithm D to obtain $c' = mS$. Then, the message is recovered by computing
$$m = c'S^{-1}.$$

In[18] it is proposed to use for 128-bit security following parameters: $m = 13$ and $t = 29$. Thus, $n = 2^m = 8192$ and $k = n - mt = 7815$. Note that the key size using these parameters is 2,946,255 bits. The considerably large key size is one of the main disadvantages of using the McEliece cryptosystem in practice.

A number of variations of the original McEliece cryptosystem have been proposed. The most well known is the Niederreiter cryptosystem [19] which is very similar to the McEliece cryptosystem, but uses a parity-check matrix instead of a generator matrix.

6.2 Underlying Security Assumptions

The security of McEliece cryptosystem is considered to be related to the following computational problems in coding theory.

Let C be an $[n, k]$ linear code over $\mathbb{F}$.

Definition 6.1 (General Decoding Problem) *Let $y \in \mathbb{F}^n$. The general decoding problem of linear codes is to find a codeword $c \in C$ such that distance between c and y is minimal.*

Definition 6.2 (Finding a Codeword of Length l) *Let $l \in \mathbb{N}$. The problem of finding a codeword of length l is to find a codeword $c \in C$ of weight l.*

The problems defined above are considered to be $\mathcal{NP}$-hard. This does not directly imply that attacking the McEliece cryptosystem is that difficult as the construction uses only binary Goppa codes [20]. Therefore, one may assume that the security of the original McEliece cryptosystem relies on the assumption that the public key is indistinguishable from a random matrix.

6.3 Implementation and Performance

Although the McEliece cryptosystem dates back to late 1970s, the implementation efficiency has not been surveyed as thoroughly as one might expect.

We begin with the results on performance of the McEliece cryptosystem published in the eBATS benchmarking system [12] measured on the same computer as described in Section 5.3. The details of the implementation can be found from [18]. Note this is not pure implementation of the original McEliece cryptosystem. Compared to the original cryptosystem, the size of the public key is reduced by making use of a generator matrix in row echelon form. Also, information rate is increased by putting some data in the error pattern.

The parameters used were as follows: $m = 11$ and $t = 32$. These parameters provide 88-bit security. The key generation took 34,215,116 cycles in total. The encryption of 59-byte message took 69,596 and decryption 1,174,628 cycles. Thus, considering the scheme described in Section 4, the lower bound for key exchange would be at least 11.4 ms. The length of the secret key is 137,282 bytes and the public key is 81,408 bytes.

In [15] Bernstein, Chou and Schwabe have surveyed the decryption algorithm of Niederreiter variation and proposed their improved implementation. At 128-bit security level decryption took 60,493 cycles on a 4-core 3.4 GHz Intel Core i5-3570 CPU. This is approximately 20 times faster than the eBACS implementation. Unfortunately, results for key generation and encryption have not been published, but the decryption results give hope that these operations can be significantly optimized too.

7 Conclusions

The implementation results associated with the cryptosystems studied in this report are presented as Table 1. The table also includes some of the public key cryptosystems widely used today such as RSA and elliptic curve Diffie-Hellman. For elliptic curve Diffie-Hellman and RSA the results are taken form the eBATS benchmarking system [12] and measured on Intel Core i5-2400 with four cores running at 3.1 GHz for elliptic curve Diffie-Hellman and on Intel Core i5-6600 with four cores running at 3.31 GHz for RSA.

The comparison takes place at the 128-bit security level, except for the McEliece cryptosystem for which 90-bit security level is used due to lack of implementation results available. The numbers presented in the table are rough estimates, but should give sufficiently accurate information to make some conclusions.

Undoubtedly the fastest post-quantum cryptosystem is the NTRU, but we see clearly that the cost of provable security is very high. Despite that, the NTRU is accepted as IEEE P1363 standard and due to its performance and low memory requirements can be used in various applications such as mobile devices and smart cards.

If excluding the key generation process, the Niederreiter variant of the McEliece cryptosystem is roughly as fast as the NTRU. But here we have to take into consideration the key size of the McEliece cryptosystem which is very large in comparison with the NTRU.

Cryptosystem	Time (ms)	Cycles
R-LWE key exchange [a]	1.4 + 2.1	1,780,000 + 4,893,000
Supersingular isogeny DH key exchange	217	520,800,000
NTRU [b]	0.204 + 0.023 + 0.024	631,200 + 71,100 + 75,300
Provably secure NTRU [b]	1731 + 12.09 + 9.5	3,981,300,000 + 27,807,000 + 21,850,000
McEliece cryptosystem [b] [c]	11.03 + 0.02 + 0.38	34,215,116 + 69,600 + 1,174,600
Niederreiter scheme [bd]	11.03 + 0.02 + 0.02	34,215,116 + 69,600+ 60,500
Elliptic curve DH (curve 25519) [e]	0.066 + 0.063	204,600 + 193,900
RSA (3072-bit)[b]	240.10[f] + 0.04 + 2.50	794,832,800[f] + 118,800 +8,262,200

Table 1: A Comparison of Public Key Cryptosystems at the 128 Bit Security Level

[a]Client + server (Open SSL)
[b]Key generation + encryption + decryption
[c]90-bit security level
[d]Key generation and encryption estimated to be same as for McEliece
[e]Key generation + sharing
[f]Measurements indicated large variance

When considering the protocols originally designed for post-quantum key exchange, the R-LWE key exchange has an advantage over the supersingular isogeny Diffie-Hellman key exchange. Although, we have to remind that the R-LWE key exchange was introduced in 2014, thus it must be cryptanalyzed thoroughly before using it in practice.

In conclusion, it is not straight-forward which post-quantum cryptosystem studied in the report is best for key exchange. The choice depends on the concrete application. Furthermore, one must keep in mind that these underlying mathematical problems presented in the report are not studied as much as, for example, factoring and discrete logarithms, thus the attacks might improve and suggested parameter sizes change.

Acknowledgements

The author of this report has received the Skype and IT Academy Master's Scholarship for the academic year 2015/16, funded by Estonian Information Technology Foundation and Skype Technologies OÜ.

References

[1] C. Peikert, "Lattice cryptography for the internet," in *Post-Quantum Cryptography*, ser. Lecture Notes in Computer Science, M. Mosca, Ed. Springer International Publishing, 2014, vol. 8772, pp. 197–219. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11659-4_12

[2] D. Jao and L. De Feo, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," in *Post-Quantum Cryptography*, ser. Lecture Notes in Computer Science, B.-Y. Yang, Ed. Springer Berlin Heidelberg, 2011, vol. 7071, pp. 19–34. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-25405-5_2

[3] J. Hoffstein, J. Pipher, and J. Silverman, "NTRU: a new high speed public key cryptosystem." Preprint; presented at the rump session of CRYPTO 1996, 1996.

[4] R. J. McEliece, "A Public-Key Cryptosystem Based On Algebraic Coding Theory," ser. DSN Progress Report, 1978, vol. 42, no. 44, pp. 114–116. [Online]. Available: http://www.cs.colorado.edu/~jrblack/class/csci7000/f03/papers/mceliece.pdf

[5] J. W. Bos, C. Costello, M. Naehrig, and D. Stebila, "Post-quantum key exchange for the TLS protocol from the ring learning with errors problem," 2014, http://eprint.iacr.org/.

[6] R. Lidl and H. Niederreiter, *Finite fields*, ser. Encyclopaedia of mathematics and its applications. New York: Cambridge University Press, 1997, cette réimpression est la version 2008. [Online]. Available: http://opac.inria.fr/record=b1127735

[7] I. Blake, G. Seroussi, N. Smart, and J. W. S. Cassels, *Advances in Elliptic Curve Cryptography (London Mathematical Society Lecture Note Series)*. New York, NY, USA: Cambridge University Press, 1999.

[8] D. Moody and D. Shumow, "Analogues of Velu's formulas for isogenies on alternate models of elliptic curves," 2011, http://eprint.iacr.org/.

[9] D. Fishbein, "Machine-level software optimization of cryptographic protocols," Master's thesis, University of Waterloo, Waterloo, Canada, 2014.

[10] "IEEE P1363 standard specifications for public key cryptography," 2008.

[11] D. Stehlé and R. Steinfeld, "Making NTRU as secure as worst-case problems over ideal lattices," in *Advances in Cryptology—EUROCRYPT 2011*, ser. Lecture Notes in Computer Science, K. G. Paterson, Ed. Springer Berlin Heidelberg, 2011, vol. 6632, pp. 27–47. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-20465-4_4

[12] "eBACS: ECRYPT benchmarking of cryptographic systems," D. J. Bernstein and T. Lange, Eds. [Online]. Available: http://bench.cr.yp.to

[13] V. Kumar, "NTRU open source project.NTRUEncrypt-ebatc." GitHub. [Online]. Available: https://github.com/NTRUOpenSourceProject/NTRUEncrypt-ebats

[14] D. Cabarcas, P. Weiden, and J. Buchmann, "On the efficiency of provably secure NTRU," in *Post-Quantum Cryptography*, ser. Lecture Notes in Computer Science, M. Mosca, Ed. Springer International Publishing, 2014, vol. 8772, pp. 22–39. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11659-4_2

[15] D. J. Bernstein, T. Chou, and P. Schwabe, "McBits: Fast constant-time code-based cryptography," in *Cryptographic Hardware and Embedded Systems - CHES 2013*, ser. Lecture Notes in Computer Science, G. Bertoni and J.-S. Coron, Eds. Springer Berlin Heidelberg, 2013, vol. 8086, pp. 250–272. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40349-1_15

[16] D. J. Bernstein, T. Lange, and C. Peters, "Attacking and defending theMcEliece cryptosystem," in *Post-Quantum Cryptography*, ser. Lecture Notes in Computer Science, J. Buchmann and J. Ding, Eds. Springer Berlin Heidelberg, 2008, vol. 5299, pp. 31–46. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-88403-3_3

[17] R. Roth, *Introduction to Coding Theory*. New York, NY, USA: Cambridge University Press, 2006.

[18] B. Biswas and N. Sendrier, "McEliece cryptosystem implementation: Theory and practice," in *Post-Quantum Cryptography*, ser. Lecture Notes in Computer Science, J. Buchmann and J. Ding, Eds. Springer Berlin Heidelberg, 2008, vol. 5299, pp. 47–62. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-88403-3_4

[19] H. Niederreiter, "Knapsack-type cryptosystems and algebraic coding theory," in *Problems of Control and Information Theory 15*, 1986, pp. 159–166.

[20] E. Berlekamp, R. McEliece, and H. Van Tilborg, "On the inherent intractability of certain coding problems," ser. Information Theory, IEEE Transactions on. IEEE, 1978, vol. 24, no. 3, pp. 384–386.