Journal of Linear and Topological Algebra
Vol. 11*, No.* 01*,* 2022*,* 69- 75
DOR: 20.1001.1.22520201.2022.11.01.7.8
DOI: 10.30495/jlta.2022.690518

Application of algebraic-ring in key exchange protocol

J. Sharafi[a], H. Daghigh[a,*]

[a]*Department of Mathematics, Kashan University, Kashan, Iran.*

Received 28 February 2022; Revised 12 April 2022; Accepted 13 April 2022.
Communicated by Ghasem Soleimani Rad

Abstract. In this article, we present a non-interactive key exchange protocol with a faster run time, which is based on a Module-LWE. The Structure of protocol is designed just by relating the error vectors of both sides, without any use of a reconciliation mechanism. The idea is that as error vectors get closer to each other the success probability of the protocol increases. The innovation in this scheme is the use of high-order bits in the keys computed by both sides. Compared to the existing lattice-based key-exchange protocols, this scheme leads to lower computational complexity and longer parameters.

Keywords: Diffie-Hellman key exchange, lattice-based cryptography, learning with error, ideal lattice.

2010 AMS Subject Classification: 94A60.

1. Introduction

Nowadays billions of internet communications are being protected by public-key cryptography. In every secure communication protocol such as TLS, public-key cryptography is used to generate a cryptography primitive which is called "key agreement". In communication channels, this primitive is used for the public key agreement between both sides. Afterward, the obtained public key is used in other primitives such as symmetric cryptography, or authentication codes. The first and the most well-known key-agreement primitive, which is known as the Diffie-Hellman protocol [4] was introduced in 1976. The public parameters of this protocol contain the multiplicative algebraic group $\mathbb{Z}_p^*$ (where p is a prime number) associated with a unique primitive root g of $\mathbb{Z}_p^*$. The structure of this protocol is given in Table 1 (in this paper we call the sides of the communication as Alice and Bob).

*Corresponding author.
E-mail address: javadsharafi@grad.kashanu.ac.ir (J. Sharafi); hassan@Kashanu.ac.ir (H. Daghigh).

Print ISSN: 2252-0201
Online ISSN: 2345-5934

Alice		Bob
$a \in \{0, 1, \ldots, p-1\}$		$b \in \{0, 1, \ldots, p-1\}$
$\mathbf{A} = g^a (\mathrm{mod}\, p)$		$\mathbf{B} = g^b (\mathrm{mod}\, p)$
	$\mathbf{A} \longrightarrow$	
	$\longleftarrow \mathbf{B}$	
$s = \mathbf{B}^a (\mathrm{mod}\, p)$		$s = \mathbf{A}^b (\mathrm{mod}\, p)$

Table 1.: Diffie-Hellman key exchange protocol [4]

On the one hand, the security of most public-key cryptosystems (such as Diffie-Hellman Protocol) is based on the hardness of a mathematical problem such as factorization of a large integer, or computation of a discrete logarithm in a specific group. Also, since the best well-known classic attacks against these schemes are executed in exponential or sub-exponential time, quantum computers can able to solve some of these problems a lot faster than classic computers. For example, using the Shor algorithm [14], one can decompose large integers and calculate discrete logarithms in a polynomial time. In this way, all of the public-key cryptosystems based on these problems will be breakable.

2. Preliminaries

In this section, we bring some essential concepts frequently used in lattice based cryptography which is needed for the rest of the paper. For more information, the reader is referred to [8, 10].

Given n linearly independent vectors $b_1, b_2, \ldots, b_n \in \mathbb{R}^d$, the *lattice* $\mathcal{L}$ is defined as

$$\mathcal{L} = \{a_1 b_1 + \cdots + a_n b_n \,|\, a_i \in \mathbb{Z}\}, \tag{1}$$

where the set of $B = \{b_1, \ldots, b_n\}$ is called a *basis* of the lattice. The integers n, d respectively are called the *rank* and *dimension* of $\mathcal{L}$. If $n = d$, $\mathcal{L}$ is called a *full rank* (or *full dimension*) lattice in $\mathbb{R}^d$; this kind of lattice is very common to use in lattice-based cryptography. The determinant $\det(\mathcal{L})$ of a lattice $\mathcal{L}$ is $|\det(B)|$ for any basis B of $\mathcal{L}$.

Shortest Vector Problem (SVP)

There are three variants of the SVP [7] that can be changed to each other. The first is to find the shortest nonzero vector; the second is to find the length of the shortest nonzero vector; and the third is to determine if the shortest nonzero vector is shorter than a given real number.

Let B be a lattice basis for $\mathcal{L}$; non-zero vector $v \in \mathcal{L}(B)$ is defined as the shortest vector in lattice $\mathcal{L}(B)$ such that

$$||v|| = \lambda_1(\mathcal{L}(B)) := \min\{||w|| \,:\, 0 \neq w \in \mathcal{L}(B)\}.$$

(Notice that SVP can be defined under any norm (here we use ℓ_2-norm i.e the Euclidean norm).

Shortest Independent Vectors Problem (SIVP)

Consider B a lattice basis and q a prime integer; *SIVP* is defined as finding n linearly independent lattice vectors:

$$\{v_1, \ldots, v_n : v_i \in \mathcal{L}(B)\}$$

that minimize $||v|| := \max_i ||v_i||$. Given an approximate factor $\gamma \geqslant 1$, the γ-approximate $SIVP_\gamma$ is defined as finding n-linearly independent vectors:

$$\{v_1, \ldots, v_n : v_i \in \mathcal{L}(B)\}$$

that minimize $\max_i ||v_i|| \leqslant \lambda_n(\mathcal{L}(B))$, where λ_n is the n-th *success minimum* for an n-dimensional lattice. The i-th success minimum λ_i is the radius of the smallest ball that contains i linearly independent lattice vectors. The *decision* version of SIVP is called $GAP\ SIVP_\gamma$ and is defined by $d < \lambda_n(\mathcal{L}(B)) \leqslant (\gamma)(d)$, where d is a positive real number.

The Problem of Module-LWE

Suppose $M = R^d$. The problem of Module-LWE (shortened as $MLWE^{(M)}_{mq\Psi}(D)$) is distinguishing m samples of $U((R_q)^d \times \mathbb{T}_{R^\nu})$ with respect to $A^{(M)}_{q,s,\Psi}$, where Ψ is an arbitrary distribution in Ψ. A search copy of Module-LWE problem (shortened as $MLWE^{(M)}_{mq\Psi}$) is the problem of finding hidden element S.

Closest Vectors Problem (CVP)

Let $\mathcal{L}$ be an $n-$dimensional lattice and B is its basis; the *Closest Vector Problem* (CVP) is to find the closest lattice vector $(B.x) \in \mathcal{L}$ to a target vector $t \in \mathbb{Z}^n$. In other words, the problem is to find a vector $x \in \mathbb{Z}^n$ such that for all $y \in \mathbb{Z}^n$, we have $\|(B.x) - t\| \leqslant \|(B.y) - t\|$.

Remark 1 *In [15] NP-hardness of CVP and SVP under $l_\infty-$norm, are proved. In addition, Ajtai [1] proved that SVP is also an NP-hard problem under l_2-norm. In [7, 16] some results that compare the hardness of SVP and CVP. In [5] there is a review of the existing results about the algorithms for solutions of SVP and CVP. Few efficient algorithms exist that can solve these problems (in fact, no algorithm is defined in polynomial time, for solving these problems). This is the reason that computer scientists only consider the approximated version of these problems [8, Chapter 1].*

2.1 *Learning With Errors Problem*

A large part of cryptography constructions which are based on lattices are built under the average-case *learning with errors* (LWE) problem [12, 13], or it's more efficient variant that is defined over rings (*Ring-LWE*) [6]. these are families of problems that are represented by choosing a ring, an integer modulus, and a distribution for error.

Following [12] Regev, the Learning with Errors (LWE) problem became well-known; he showed that under quantum reduction, solving a random LWE instance is as hard as solving some of the worst-case instances of certain lattice problems. The LWE problem can be seen as a generalization of the "learning parity with noise (LPN)" problem, and it is related to hard decoding problems [9]. In general, to solve the LWE problem, one has to recover a secret vector $s \in \mathbb{Z}_q^n$ by a sequence of approximate random linear equations on s. Also, non-quantum reductions to variants of the LWE problem, based on variants of the shortest vector problem, have been shown in [11]. The LWE problem is usually used to build primitives such as *CPA*, *CCA*-secure public-key encryption, identity-based

encryption (IBE), and fully-homomorphic encryption schemes [13]. It can be defined as a search problem (sLWE) where the task is to recover the secret vector s or as a decision problem (dLWE) that asks to distinguish LWE samples from uniformly random samples.

The LWE problem is parameterized by positive integers n and q, and an error distribution χ over $\mathbb{Z}$. χ is usually taken to be a discrete Gaussian of width αq with $\alpha < 1$, which is often called the relative "*error rate*".

3. Key exchange algorithm with ring-lwe

Here we introduce a key exchange scheme based on the learning with errors problem with security against chosen main-text attacks (IND-CPA). In this scheme we use a simple method to remove error vector and reach a public key. Instead of using the compilation mechanism or coding [2, 3], here the parameters are indicated so that *high priority bits* of the calculated values from both sides become equal. Remark that using the compilation mechanism or coding, increases the complexity of calculations in implementation. So the complexity of calculating for this proposed protocol (and also the number of run cycles) are levels lower than key agreement schemes while at the same time, the generated keys are longer. So it is safe to say that this protocol and other similar protocols (with longer keys and fewer run cycles) are better choices to implement in *compilation cryptosystems*.

3.1 *Algorithm structure*

In general it can be said that the structure of key-agreement schemes based on ring learning with error problem have one thing in common: after generating polynomial $\mathbf{a}$, Alice generates random samples $\mathbf{s}_A, \mathbf{e}_A \leftarrow_R \chi$, and sends the key $\mathbf{b}_A := \mathbf{a}.\mathbf{s}_A + \mathbf{e}_A$ to Bob. Meanwhile Bob generates random samples $\mathbf{s}_B$, $\mathbf{e}_B$ and sends the key $\mathbf{b}_B := \mathbf{a}.\mathbf{s}_B + \mathbf{e}_B$ to Alice. Alice and Bob respectively compute $\mathbf{s}_A.\mathbf{b}_B$ and $\mathbf{s}_B.\mathbf{b}_A$ by using their private keys. These two values are almost, but not exactly, equal:

$$\mathbf{a}.\mathbf{s}_B.\mathbf{s}_A + \mathbf{e}_B.\mathbf{s}_A \approx \mathbf{a}.\mathbf{s}_A.\mathbf{s}_B + \mathbf{e}_A.\mathbf{s}_B.$$

The Compilation mechanism or the coding method is used in the process of removing error elements, to reach a public key. This leads to exchanging higher values between the two sides. For example, in the key agreement algorithm NewHope-USENIX [3] reconciliation mechanism and an auxiliary function are used, which increases the complexity of computation in the implementation. As another example, in the algorithm NewHope-SIMPLE [2] the encoding and mining method is used so that both sides send an extra encoded value and hence a longer message is made.

The idea of the proposed key exchange scheme is to remove the reconciliation mechanism or the encoding method; hence reducing computation complexity.

The parameters used in this scheme, contain n (as the dimension of the lattice), modulus q and the rings $\frac{\mathbb{Z}[x]}{<x^n+1>}$ and $\frac{\mathbb{Z}_q[x]}{<x^n+1>}$. The main structure of the proposed algorithm is shown in table (2). In this scheme the polynomial $\mathbf{a} \in R_q$ is sampled uniformly. Alice and Bob sample their private keys and error polynomials respectively $\{\mathbf{s}', \mathbf{e}'\}$ and $\{\mathbf{s}, \mathbf{e}\}$, by a discrete Gaussian distribution χ over R_q with parameter σ. Afterward Alice and Bob respectively compute the public keys $\mathbf{b} := \mathbf{a}.\mathbf{s} + \mathbf{e}$ and $\mathbf{b}' := \mathbf{a}.\mathbf{s}' + \mathbf{e}'$ and send them to each other. In the final phase, they consider *high order bits* of respectively $(\mathbf{b}' - \mathbf{e}).\mathbf{s}$ and $(\mathbf{b} - \mathbf{e}').\mathbf{s}'$, as the public key; see Table 2 below (the parameter γ will be introduced in Lemma 3.1):

Alice		Bob
$\mathbf{a} \in R_q$		$\mathbf{a} \in R_q$
$\mathbf{s}, \mathbf{e} \leftarrow_\chi R_q$		$\mathbf{s}', \mathbf{e}' \leftarrow_\chi R_q$
$\mathbf{b} := \mathbf{a}.\mathbf{s} + \mathbf{e}$		$\mathbf{b}' := \mathbf{a}.\mathbf{s}' + \mathbf{e}'$
	$\longrightarrow^{\mathbf{b}}$	
	$\longleftarrow^{\mathbf{b}'}$	
$\mathbf{w} = (\mathbf{b}' - \mathbf{e}).\mathbf{s}$		$\mathbf{w}' = (\mathbf{b} - \mathbf{e}').\mathbf{s}'$
$\mu = Highbits(\mathbf{w}, 2\gamma)$		$\mu = Highbits(\mathbf{w}', 2\gamma)$

Table 2.: Key exchange with LWE-problem

Remark 2 *μ is a vector containing the highest bits of the all the coefficients* $\mathbf{w}$ *and* $\mathbf{w}'$.

3.2 *Correctness*

Although $\mathbf{w} \neq \mathbf{w}'$, we show that a lot of their bits (high order bits) are equal to each other. To do that we minimize the effect of $\mathbf{e}$ and $\mathbf{e}'$ in calculating $\mathbf{w}$ and $\mathbf{w}'$; also we calculate the failure rate in the correct way of executing this algorithm. This is directly related to the choice of suitable parameters so that these error polynomials just effect low level bits hence we reach a common secret key μ Notice that the computed primary keys in the communication channel are

$$\mathbf{w} = \mathbf{a}.\mathbf{s}' - \mathbf{e}''.\mathbf{s} \quad , \quad \mathbf{w}' = \mathbf{a}.\mathbf{s}.\mathbf{s}' - \mathbf{e}''.\mathbf{s},$$

where $\mathbf{e}'' := \mathbf{e} - \mathbf{e}'$ noticeable point in a protocol 2 is the relation between the two error vectors $\mathbf{e}$ and $\mathbf{e}'$ so that as these two error vectors come closer to each other, $\mathbf{e}''$ gets smaller and $\mathbf{w}$ and $\mathbf{w}'$ come closer together.

Lemma 3.1 Suppose that $q = q_l.2^l + q_{l-1}.2^{l-1} + \cdots + q_1 2 + q_0$ is the representation of q in the 2-base form, and $\max\{\|\mathbf{e}''.\mathbf{s}'\|_\infty, \|\mathbf{e}''.\mathbf{s}\|_\infty\} \leqslant \gamma$. Then the probability of protocol (2) being true, is at least $1 - \frac{1}{2^k}$ where $\frac{q}{l} \leqslant 2^{k+1}$, (which is the same probability as $l-1$ bits of $\mathbf{w}$ and $\mathbf{w}'$ being equal).

Proof. Polynomials $\mathbf{w}$ and $\mathbf{w}'$ have small bounded coefficients with bound γ. Since $q \leqslant 2^1.2^k\gamma$, all the coefficients have at most $1 + k + \log(\gamma) \geqslant \log(q)$ bits. In one hand just the last $\log(\gamma)$ bits at the end of each coefficient effect the error polynomials. $(l-1)$th bit of each coefficient changes just when all the previous bits are equal to 1 and therefore these bits (from the first to $(l-2)$th one) indicate the multiplication result. The probability of these k bits ($\log(q) - \gamma - 1 = k$) being equal to 1, is 2^{-k}. Therefore the probability that two coefficient share the highest bit, is $1 - \frac{1}{2^k}$. ■

4. Proposed Key exchange algorithm with Module-LWE

In this section, using the Diffie-Hellmann key exchange, the Module-LWE, and the key exchange algorithm in a Table 2, we present a lattice-based key exchange algorithm based on Module-LWE. the public key is obtained using high-order bits. Of course, in the proposed key exchange algorithm, the error vectors of the parties are also used, and because these vectors are finite, the probability of success increases.

4.1 *Algorithm structure*

The set of parameters of the proposed key exchange algorithm, including module q, lattice dimension n and the polynomial ring is $R_q = \frac{\mathbb{Z}[x]}{<x^n+1>}$. To generate the public key $A \in R_q^{k\times l}$, one of the parties of the key exchange algorithm, for example, Alice generates a random seed ξ and sends it to the other party, for example, Bob. The parties receive the public key A, which is in matrix form, using the **Expand**A algorithm based on the hash function such as SHAKE-128.
To generate the private key and the error polynomials, each party generates a random seed and using the collision resistant hash function (e.g., H) calculates the hidden polynomials and the error vector. In the last step, after calculating and sending the public keys sampled based on the Module-LWE distribution, each party multiplies the difference between received public key and the polynomial error to its secret key. Afterward, by calculating the abstracted important bits or High-order bits reaches a common 256-bit string.

Alice	Bob
	$\xi \in \{0,1\}^{256}$
	$\xleftarrow{\xi}$
$\xi \in \{0,1\}^{256}$	
$\rho := H(\xi)$	$\rho := H(\xi)$
$A \in R_q^{k\times l} := ExpandA(\xi)$	$A \in R_q^{k\times l} := ExpandA(\xi)$
$seed_B \in \{0,1\}^{256}$	$seed_A \in \{0,1\}^{256}$
$(s'_1, s'_2) \in S_\eta^\ell . S_\eta^k := H(seed_B)$	$(s_1, s_2) \in S_\eta^\ell . S_\eta^k := H(seed_A)$
$t' := As'_1 + s'_2$	$t := As_1 + s_2$
	$\xleftarrow{t}$
$\xrightarrow{t'}$	
$w' = (t - s'_2).s'_1$	$w = (t' - s_2).s_1$
$\mu' = HighBites_q(w', 2a)$	$\mu = HighBits_q(w, 2a)$
$ss = SHAKE128(\mu') \in \{0,1\}^{256}$	$ss = SHAKE128(\mu) \in \{0,1\}^{256}$

Table 3.: Proposed Key exchange algorithm with Module-LWE

4.2 *Security*

The main problem in this section is to find parameters such that the success probability increases and moreover the security level improves. In this algorithm, module q and lattice n are defined as $q = 2^{23} - 2^{13} + 1$ and $n = 256$. The complete set of proposed parameters for 3-level security 108 bit, 160 bit and 226 bit are provided in Table 4.

5. Conclusions

In this paper, we compare three key exchange algorithms. Among the other two algorithms, the Module-LWE algorithm is faster due to having fewer circular runs. Moreover, smaller parameters (module q and lattice n) render the algorithm much more efficient than the other algorithms.

Parameters					Security level				
n	q	η	k	α	decoding attack	distinguish attack	dual attack	primal attack	SVP attack
256	2^{23}-$2^{13}+1$	2	4	$\frac{2^{23}-2^{13}}{88}$	120	125	112	107	108
256	2^{23}-$2^{13}+1$	4	2	$\frac{2^{23}-2^{13}}{32}$	163	167	166	165	160

Table 4.: Proposed parameters with security level estimation results when we use proposed Module-LWE algorithm.

References

[1] M. Ajtai, The shortest vector problem in L_2 is NP-hard for randomized reductions, Proceedings of the 30th ACM Symposium on the Theory of Computing. (1998), 10-19.
[2] E. Alkim, L. Ducas, T. Pöppelmann, P. Schwabe, New Hope Without Reconciliation, 2016.
[3] E. Alkim, L. Ducas, T. Pöppelmann, P. Schwabe, Post-Quantum Key Exchange-A New Hope, Proceedings of the 25th USENIX Security Symposium, 2016.
[4] W. Diffie, M. Hellman, New directions in cryptography, IEEE transactions on Information Theory. 22 (1976), 644-654.
[5] G. Hanrot, X. Pujol, D. Stehlé, Algorithms for the Shortest and Closest Lattice Vector Problems, Coding and Cryptology, Third International Workshop, IWCC, 6639 of LNCS, 2011.
[6] V. Lyubashevsky, C. Peikert, O. Regev, On Ideal Lattices and Learning with Errors over Rings, Advances in Cryptology-Eurocrypt. (2010), 1-23.
[7] D. Micciancio, S. Goldwasser, Complexity of Lattice Problems: A Cryptographic Perspective, The Kluwer International Series in Engineering and Computer Science, 671, 2002.
[8] D. Micciancio, S. Goldwasser, Complexity of Lattice Problems: A Cryptographic Perspective, Springer, 2012.
[9] D. Micciancio, O. Regev, Lattice-based cryptography, Post-quantum Cryptography. (2009), 147-191.
[10] C. Peikert, A decade of lattice cryptography, Foundations and Trends in Theoretical Computer Science. 10 (2016), 283-424.
[11] C. Peikert, Public-key cryptosystems from the worst-case shortest vector problem, Proceedings of the 49th ACM Symposium on Theory of Computing. (2009), 333-342.
[12] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, Proceedings of the 37th ACM Symposium on Theory of Computing. (2005), 84-93.
[13] O. Regev, The learning with errors problem (invited survey), In Proceedings of the 25th Annual IEEE Conference on Computational Complexity-CCC. (2010), 191-204.
[14] P. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, 35th FOCS IEEE Computer Society Press. (1994), 124-134.
[15] P. Van Emde Boas, Another NP-Complete Partition Problem and the Complexity of Computing Short Vectors in a Lattice, Technical Report, Universiteit van Amsterdam, Mathematisch Institut, 1981.
[16] K. Xagawa, Cryptography with Lattices, PhD thesis, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2010.