Multidimentional ModDiv public key exchange protocol

Samir Bouftass

E-mail : crypticator@gmail.com

June 4, 2021

Abstract

This paper presents Multidimentional ModDiv public key exchange protocol which security is based on the hardness of an LWR problem instance consisting on finding a secret vector X in $\mathbb{Z}_r^n$ knowing vectors A and B respectively in $\mathbb{Z}_s^m$ and $\mathbb{Z}_t^l$, where elements of vector B are defined as follows : $B(i) = (\sum_{j=1}^{j=n} A(i+j) \times X(j))\ Mod(2^p)Div(2^q)$.
Mod is integer modulo operation, Div is integer division operation, p and q are known positive integers satisfying $p > 2 \times q$. Size in bits of s equals p, size in bits of r equals q, and size in bits of t equals $p - q$, $m > 2 \times n$ and $l = m - n$.

Keywords : Diffie Hellman key exchange protocol, Post Quantum cryptography, Lattice based cryptography, Closest vector problem, Learn with rounding problem.

1 Introduction :

Since its invention by Withfield Diffie and Martin Hellman [1], public key cryptography has imposed itself as the necessary and indispensable building block of every IT security architecture.
In the last decades, it has been proven that public key cryptosystems based on number theory problems are not immune againt quantum computing attacks [2], urging the necessity of inventing new algorithms not based on classical problems namely factoring, discret log over multiplicative groups or elliptic curves.

In [3] is presented a one dimentional ModDiv public key exchange protocol which security have been shown in Barcau et all [4] to be based on CVP problem.
Y Zang [5] have proven that one dimentional ModDiv security problem can be reduced to a CVP problem in 2 dimentional lattice.

Present paper proposes Multidimentional ModDiv public key exchange protocol which security is based on an instance of learn with rounding problem first proposed by Banerjee et all [6] .
In section 2, we describe one dimentional ModDiv public key exchange protocol and provide a proof of its corectness .
In section 3, we describe Multidimentional ModDiv public key exchange protocol.

2 One dimentional ModDiv public key exchange protocol :

2.1 Notations:

$mdv2_{(p,q)}(A) = A\ Mod(2^p)Div(2^q)$. (A being an integer, Mod modulo operation, and Div integer division).

$\parallel A \parallel$: size in bits of A .

2.1.1 Public parameters :

Integer A, positive integers p, q and S, where $p > 2 \times q$.

A is pseudorandom and $\parallel A \parallel = p$.

S is exchanged key maximum size, which is equal to $p - (2 \times q)$.

2.1.2 Private Computations :

- Alice generates pseudorandomly a q bit number X, and calculates $U = mdv2_{(p,q)}(A \times X)$.

- Bob generates pseudorandomly a q bit number Y, and calculates $V = mdv2_{(p,q)}(A \times Y)$.

2.1.3 Publicly exchanged values :

- Alice sends U to Bob.

- Bob sends V to Alice.

2.1.4 Further Private Computations :

- Alice calculates $Wa = mdv2_{(p-q,q)}(X \times V)$.

- Bob calculates $Wb = mdv2_{(p-q,q)}(Y \times U)$.

Bob and Alice know that :

$Wa = Wb$ or $\mid Wa - Wb \mid = 1$.

2.2 Proof of correctness :

Lemma 1. *Let A be an integer, p and q positive integers.*

If $(p > q)$, then $mdv2_{(p,q)}(A \times 2^q) = A\ mod(2^{p-q})$.

Proof :

$$mdv2_{(p,q)}(A \times 2^q) = (A \times 2^q)\ Mod(2^p) Div(2^q).$$

Observe least significant q bits of $N = (A \times 2^q)\ Mod(2^p)$ are zeros whereas its most significant $p-q$ bits are the least significant $p-q$ bits of A, dividing then N by 2^q implies :

$$(A \times 2^q)\ Mod(2^p) Div(2^q) = A\ Mod(2^{p-q}) = mdv2_{(p,q)}(A \times 2^q)\ .$$

Theorem 1. *Let A, X, Y, p and q be integers where $p > q$, $\| A \| = p$, $\| X \| = \| Y \| = q$.*

$Wa = mdv2_{(p-q,q)}(X \times mdv_{(p,q)}(A \times Y))$.

$Wb = mdv2_{(p-q,q)}(Y \times mdv_{(p,q)}(A \times X))$.

There is two possibilities :

1 - $Wa = Wb$.

2 - $| Wa - Wb | = 1$.

Proof :

Let $H1$ and $H2$ be integers such as :

$$U_1 = mdv2_{(p,q)}(A \times X) \times 2^q = (A \times X - H1)\ Mod(2^p).$$

$$V_1 = mdv2_{(p,q)}(A \times Y) \times 2^q = (A \times Y - H2)\ Mod(2^p).$$

The fact that the least significant q bits of U_1 and V_1 are zeroes, implies $\| H_1 \| = \| H_2 \| = q$.

Let's calculate :

$$Wa1 = (X \times V_1)\ Mod(2^p) = ((X \times Y \times A) - (X \times H_2))\ Mod(2^p)\ (1)$$

$$Wb1 = (Y \times U_1)\ Mod(2^p) = ((Y \times X \times A) - (Y \times H_1))\ Mod(2^p)\ (2)$$

$\| X \|=\| Y \|=\| H_1 \|=\| H_2 \|= q$ implies $\| X \times H_2 \|=\| Y \times H_1 \|= 2 \times q$, we have then :

$$Wa1\ Div(2^{2\times q}) = (X \times Y \times A)\ Mod(2^p)Div(2^{2\times q}) - E_a$$

$$Wb1\ Div(2^{2\times q}) = (Y \times X \times A)\ Mod(2^p)Div(2^{2\times q}) - E_b$$

where E_a and E_b are respectively the $2 \times q$'th borrows of binary substractions (1) and (2) .

E_a and E_b being bits, they can have then for values 0 or 1, implying :

if $E_a = E_b$ we have $Wa1\ Div(2^{2\times q}) = Wb1\ Div(2^{2\times q})$.

if $| E_a - E_b |= 1$ we have $| Wa1\ Div(2^{2\times q}) - Wb1\ Div(2^{2\times q}) |= 1$.

Now we'll show that :

$$Wa = Wa1\ Div(2^{2\times q}) \text{ and } Wb = Wb1\ Div(2^{2\times q}).$$

ending thus theorem's proof .

$$Wa1 = (X \times V_1)\ Mod(2^p) = (X \times mdv2_{(p,q)}(A \times Y) \times 2^q)\ Mod(2^p).$$

$$Wa1\ Div(2^q) = (X \times V_1)\ Mod(2^p)Div(2^q) = (X \times mdv2_{(p,q)}(A \times Y) \times 2^q)\ Mod(2^p)Div(2^q).$$

Applying Lemma 1, we get :

$$Wa1\ div(2^q) = (X \times V_1)\ Mod(2^p)Div(2^q) = (X \times mdv_{(p,q)}(A \times Y))\ Mod(2^{p-q}).$$

$$Wa1\ Div(2^{2\times q}) = (X \times mdv2_{(p,q)}(A \times Y))\ Mod(2^{p-q})Div(2^q).$$

$$Wa1\ Div(2^{2\times q}) = mdv2_{(p-q,q)}(X \times mdv_{(p,q)}(A \times Y)).$$

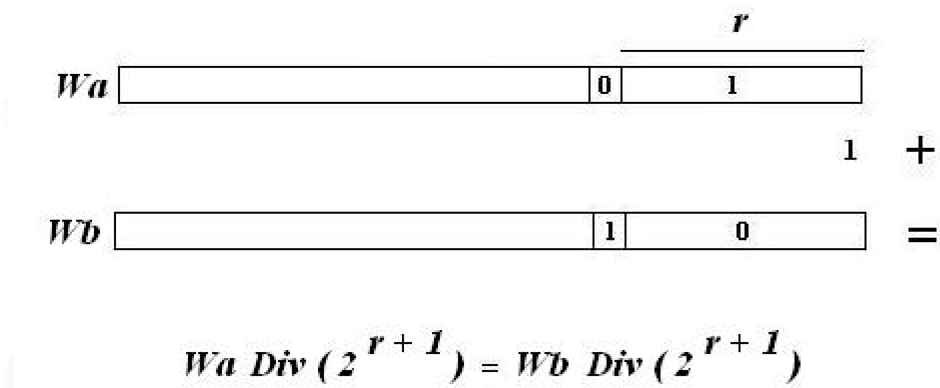
$$Wa1\ Div(2^{2\times q}) = Wa.$$

Likewise we can prove :

$$Wb1\ Div(2^{2\times q}) = Wb.$$

Observe, $\text{Max}(S) = \| Wa \| = \| Wb \| = p - (2 \times q)$.

Figure 1:



But there is one drawback, Alice or Bob wouldn't get precisely the same value, they only know that $Wa = Wb$ or $|\ Wa - Wb\ | = 1$. That is if Alice encrypts a message M with key Wa and sends resulting cipher text C to Bob who in order to get M, he should decrypt C with Wb, $Wb + 1$ and $Wb - 1$ and gets 3 plausible plain texts. To decide which one is correct, Alice should priorly hash M and joins computed digest as header to M before encryption. Bob can then decide which plain text is correct by hashing obtained plain text and compares it to joined hash value, but decryption can be then three times slower than encryption. Now lets suppose that computed values Wa and Wb are uniformly distributed : the probability that their least r significant bits are ones is $(1/2)^r$. To get M, Bob can perform only one decryption if he and Alice agreed on r. Alice would then encrypt with $Wa\ Div(2^r)$, Bob would decrypt with $Wb\ Div(2^r)$ (Figure 1).

We have experimentaly observed that $Pr[\ Wa = Wb\ ] = 2/3$, implying :

$$Pr[\ Wa\ Div(2^r) = Wb\ Div(2^r)\ ] = 1 - ((1/3) \times (1/2)^r). \ (1)$$

Consequently Bob can decrypt C only one time, but the price going with it, is r bits security and a probability of $1 - ((1/3) \times (1/2)^r)$ to get plain text right.

Although introducing fixing parameter r, helps to know exact key size and increases probability of key exchange corectness. End goal remains geting a key exchange protocol functioning all the time. Figure 1 shows that as long next significant bit is the same, (1 in case of addition, 0 in case of substraction) bit carry/borrow propagates. The fact that $Wa = Wb$ or $Wa = Wb + 1$ or $Wa = Wb - 1$, means that in order to get the same key, it suffices to know least significants bits number of Wb affected by addition or substraction by 1, or how much bit carries/borrows of addition or substraction by 1 propagates to most significants bits. All Bob and Alice have to do is right shifting their obtained numbers as long as least significant bit value is the same as the formers, then right shift again. In conclusion they would have an all the time functioning key exchange protocol, but they wouldn't know shared key exact size. Key sizes would have a statistical distibution derived from (1).

3 Multidimentional ModDiv public key exchange protocol :

3.1 Notations:

polyDiv : Polynomial division operation, polyMod : Polynomial modulo operation.

$pmdv_{(m,n)}(\mathbf{A}) = (\mathbf{A})\ polyMod(x^m) polyDiv(x^n)$.

3.1.1 Public parameters :

Integers m, n, p, q where $m > (2 \times n)$ and $p > (2 \times q) + log_2(n)$.

An m degree polynomial **A**, which coefficients sizes in bits equals $\| 2^p \|$.

A coefficients are pseudorandomly generated.

3.1.2 Private Computations :

- Alice chooses an n degree polynomial **X**, which coefficient sizes in bits equals $\| 2^q \|$. **X** coefficients are pseudorandomly generated.

- Alice calculates $\mathbf{U} = pmdv_{(m,n)}(\mathbf{A} \times \mathbf{X})$.

- For each **U** coefficient $U(i)$, she calculates $U1(i) = mdv2_{(p,q)}(U(i))$, getting thus polynomial **U1**.

- Bob chooses an n degree polynomial **Y**, which coefficient sizes in bits equal $\| 2^q \|$. **Y** coefficients are pseudorandomly generated.

- Bob calculates $\mathbf{V} = pmdv_{(m,n)}(\mathbf{A} \times \mathbf{Y})$.

- For each **V** coefficient $V(i)$, he calculates $V1(i) = mdv2_{(p,q)}(V(i))$, getting thus polynomial **V1**.

3.1.3 Publicly exchanged values :

- Alice sends **U1** coefficients to Bob.

- Bob sends **V1** coefficients to Alice.

3.1.4 Further Private Computations :

- Alice calculates polynomial $\mathbf{Wa}= pmdv_{(m-n,n)}(\mathbf{X}\times\mathbf{V1})$.

- For each **Wa** coefficient $Wa(i)$:

  – She calculates r(i), $Wa(i)$ consecutive least significant bits with the same value, number + 1.

  – She calculates $Wa1(i) = mdv2_{(p-q,q)}(Wa(i))\ Div(2^{r(i)+log_2(n)})$

- Bob calculates polynomial $Wb = pmdv_{(m-n,n)}(\mathbf{X}\times\mathbf{U1})$.

- For each **Wb** coefficient $Wb(i)$:

  – He calculates r(i), $Wb(i)$ consecutive least significant bits with the same value, number + 1.

  – He calculates $Wb1(i) = mdv2_{(p-q,q)}(Wb(i))\ Div(2^{r(i)+log_2(n)})$

Bob and Alice know that for each coefficients $Wa1(i)$ and $Wb1(i)$:

Shared key sise in bits they would get equals :

$\sum_{i=1}^{i=m-(2\times n)}(m-(2\times n))\times(p-((2\times q)+r(i)+log_2(n))$.

3.2 Proof of Correction :

Proof of correction straitly follows from :

- Polynomial multiplication commutativity.

- Theorem 1.

- Size in bits of the result of adding n, s bits numbers is $s\times log_2(n)$.

3.3 Generalized Multidimentional Moddiv public key exchange protocol:

3.3.1 Public parameters :

Integers m, n, p, q, P, Q where $m > 2 \times n$, $p > (2 \times q) + log_2(n)$, $\| P \| = \| 2^p \|$ and $\| Q \| = \| 2^q \|$.

An m degree polynomial **A**, which coefficients sizes in bits equal $\| 2^p \|$.

P, Q and **A** coefficients are pseudorandomly generated.

3.3.2 Private Computations :

- Alice chooses an n degree polynomial **X**, which coefficient sizes in bits equal $\| 2^q \|$. **X** coefficients are pseudorandomly generated.

- Alice calculates $\mathbf{U} = pmdv_{(m,n)}(\mathbf{A} \times \mathbf{X})$.

- For each **U** coefficient $U(i)$:

  – she calculates $U1(i) = U(i)\ Mod(P)Div(Q)$, getting thus polynomial **U1**.

- Bob chooses an n degree polynomial **Y**, which coefficient sizes in bits equal $\| 2^q \|$. **Y** coefficients are pseudorandomly generated.

- Bob calculates $\mathbf{V} = pmdv_{(m,n)}(\mathbf{A} \times \mathbf{Y})$.

- For each **V** coefficient $V(i)$:

  – He calculates $V1(i) = V(i)\ Mod(P)Div(Q)$, getting thus polynomial **V1**.

3.3.3 Publicly exchanged values :

- Alice sends **U1** coefficients to Bob.

- Bob sends **V1** coefficients to Alice.

3.3.4 Further Private Computations :

- Alice calculates polynomial $\mathbf{Wa} = pmdv_{(m-n,n)}(\mathbf{X} \times \mathbf{V1})$.

- For each **Wa** coefficient $Wa(i)$:

  – She calculates r(i), $Wa(i)$ consecutive least significant bits with the same value, number + 1.
  – She calculates $Wa1(i) = Wa(i)\ Mod(P\ Div(Q))\ Div(2^{q+r(i)+log_2(n)})$.

- Bob calculates polynomial $\mathbf{Wb} = pmdv_{(m-n,n)}(\mathbf{X} \times \mathbf{U1})$.

- For each **Wb** coefficient $Wb(i)$:

  – He calculates r(i), $Wb(i)$ consecutive least significant bits with the same value, number + 1.
  – He calculates $Wb1(i) = Wb(i)\ Mod(P\ Div(Q))\ Div(2^{q+r(i)+log_2(n)})$.

Shared key sise in bits they could get equals :

$\sum_{i=1}^{i=m-(2\times n)}(m-(2\times n))\times(p-((2\times q)+r(i)+log_2(n))$.

We have experimentally observed that generalized variant of proposed public key exchange protocol holds.

3.4 Security :

To attack proposed public key exchange protocol, adversary Eve knows polynomials **A**, **U1**, **V1** and parameters p, q, m, n satistifying the following conditions :

- **A** is of degree m.
- **U1** and **V1** are of degree m-n.
- **A** coefficients sizes in bits are p.
- **U1** and **V1** coefficients sizes in bits are p-q.
- $m > 2 \times n$ and $p > 2 \times q$.

She equaly knows that there exists polynomials **U**, **V**, **X** and **Y** Satisfying :

- **X** and **Y** are of degree n.
- **U** and **V** are of degree m-n.
- **X** and **Y** coefficients sizes in bits are q.
- **U** and **V** coefficients sizes in bits are $p + q + log_2(n)$.
- $\mathbf{U} = (\mathbf{X} \times \mathbf{A})\ polyMod(x^m)polyDiv(x^n)$.
- $\mathbf{V} = (\mathbf{Y} \times \mathbf{A})\ polyMod(x^m)polyDiv(x^n)$.

Coefficients of **U** and **V** are respectively related to those of **U1** and **V1** by following equations :

- $U1(i) = U(i)\ Mod(2^p)Div(2^q)$.
- $V1(i) = V(i)\ Mod(2^p)Div(2^q)$.

Meaning to know secret polynomials **X** and **Y**, Eve have to solve the following equations set :

For $1 \leq$ i $\leq$ m - n

$U1(i)$ =($(\sum_{j=1}^{j=n} A(j+i) \times X(j))$ $Mod(2^p) Div(2^q)$.

$V1(i)$ =($(\sum_{j=1}^{j=n} A(j+i) \times Y(j))$ $Mod(2^p) Div(2^q)$.

To Attack Generalized Multidimentional ModDiv public key, Eve had to solve the following equations set, knowing parameters P and Q satisfy $\parallel P \parallel = \parallel 2^p \parallel$ and $\parallel Q \parallel = \parallel 2^q \parallel$

For $1 \leq$ i $\leq$ m - n

$U1(i)$ =($(\sum_{j=1}^{j=n} A(j+i) \times X(j))$ $Mod(P) Div(Q)$.

$V1(i)$ =($(\sum_{j=1}^{j=n} A(j+i) \times Y(j))$ $Mod(P) Div(Q)$.

3.4.1 ModDiv learn with rounding problem:

Basically underlying security problem of proposed public key exchange protocol is an instance of learn with rounding problem first proposed by Banerjee et all [6], which we would define as ModDiv learn with rounding problem.

Said instance is caracterized by the following features :

- Rounding is done from $\mathbb{Z}_{2^p}$ to $\mathbb{Z}_{2^{(p-q)}}$.

- In Generalized Multidimentional ModDiv, rounding is done from $\mathbb{Z}_P$ to $\mathbb{Z}_{P/Q}$.

- Public vectors are not random, they reflect polynomial mutiplication algebraic structure : If vector **A1** [A(1), A(2) .... A(n)] is given as public, vector **A2** [R(1),A(2),.... A(n-1)]. is also given as public. Element R(1) is pseudorandomly generated.

- Secret vector elements size is the same as derandomized error vector elements induced by rounding, which is q bits.

Seemingly Multidimentional ModDiv learn with rounding problem is at most as hard as its generalized counterpart, but public key cryptosystem based on the first is easier to implement, the modulis are powers of two : bit shifting operations could be used instead of actual integer modulo and division operations.

Learn with rounding problem is considered to be at least as hard as Learn with error problem, and was used recently to construct public key cryptosystems like Saber [7] one of NIST third round finalists.

The open question is how hard ModDiv learn with rounding problem, actually is ?.

4 Conclusion :

In this paper we have presented Multidimentional ModDiv public key exchange protocol, the multidimentional version of public key exchange protocol presented in [3].

We have shown that its security is based on an instance of learn with rounding problem, we defined as ModDiv learn with rounding problem.

One can construct public key encryption and digital signature ElGamal schemes based on presented Key exchange protocols, it is also quite possible to device hash functions and pseudo random numbers generators, based on introduced problems.

References

[1] Whitfield Diffie, Martin E.Hellman. *New Directions in cryptography, IEEE Trans. on Info. Theory, Vol. IT-22, Nov. 1976 (1976)*

[2] Daniel J Bernstein, Johannes Buchmann, Erik Dahman. *Post-Quantum Cryptography*, (2009), Springer Verlag , Berlin Heidelberg .

[3] A Azhari, S Bouftass : On a new fast public key cryptosystem. *IACR Cryptology eprint Archive 2014:946(2014).*

[4] Mugurel Barcau, Vicentiu Pasol, Cezar Plesca, and Mihai Togan : On a Key Exchange Protocol *SECITC 2017.*

[5] Y Zhang : A practical attack to Bouftasss crypto system *arXiv:1605.00987 [cs.CR].*

[6] Abhishek Banerjee, Chris Peikert, Alon Rosen : Pseudorandom Functions and Lattices *IACR Cryptology eprint Archive 2011:401(2011).*

[7] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren Saber: Module-LWR based key exchange,CPA-secure encryption and CCA-secure KEM *IACR Cryptology eprint Archive 2018:230(2018).*