3WISE: Cubic Element-Wise trapdoor based MPKC cryptosystem

2023-05-31 Submission

Principal Submitter

This submission is from the following team, listed in alphabetical order:

- Borja Gómez Rodríguez

E-mail address: borja.gomez@develrox.com - borgomez026@gmail.com

Telephone: +34 626 30 78 32 - +56 981 25 25 68

Postal Address:
Borja Gómez Rodríguez
PZ Landabaso 8º - 5A
Bilbao, 48015
Spain

Auxiliary submitters: There are no auxiliary submitters. The principal submitter is the team listed above.

Inventors/developers: The inventors/developers of this submission are the same as the principal submitter. Relevant prior work is credited below where appropriate.

Owner: Same as submitter.

Signature:

See also printed version of "Statement by Each Submitter"

Contents

1 Introduction

This paper introduces 3WISE, a new cryptosystem based on Multivariate Public Key Cryptography (MPKC) . The idea behind 3WISE is to hide the cubic element-wise multiplication of a given vector over a prime p, where 3 has a multiplicative inverse in $\mathcal{Z}_{p-1}$ in order to be able to sign or decipher.

2 Algorithm Specification (part of 2.B.1)

2.1 Tensor Algebra

The Public Key of the schemes proposed in Multivariate Cryptography are composed of m equations in n variables having quadratic or cubic monomials. These equations can be expressed in terms of quadratic forms or tensors. Quadratic schemes are simply m quadratic forms of size $n \times n$. In the cubic case, we have n quadratic forms per equation thus m equations involving n quadratic forms of size $n \times n$, which adds to a rectangular matrix $m \times n^3$. As every quadratic form can be expressed as a tensor $q_i \cdot (x \otimes x)$ then the Public Key of a MPKC scheme can be viewed as a rectangular matrix times the tensor product of the input vector.
This will give us a rectangular $m \times n^2$ (quadratic case) or $m \times n^3$ (cubic case).
These representations can be encoded up to a matrix of size $m \times \binom{n+d-1}{d}$ reducing the size of the Public Key matrix. For $d = 2$ we have $m \times \frac{n(n+1)}{2}$ and for $d = 3$ the matrix has size $m \times \frac{1}{6}n(1+n)(2+n)$

2.2 Hadamard product and Face-Splitting product

The Hadamard product of vectors is equivalent to the element-wise operation between their components. Here the idea is to take the Hadamard product of a vector $x = (x_1, \ldots, x_n) \in F_p^n$ three times such that $x \circ x \circ x = (x_1^3, \ldots, x_n^3)$. However, dealing with symbolic expressions is non convenient as a the cryptosystem would use the expression Public Key $P(x) = T \cdot (Sx \circ Sx \circ Sx)$. If we expand the expression symbolically, let $z = S(x)$ then $T \cdot (z \circ z \circ z)$ so the computation of every symbolic expression $z_i^3 = (\sum_{j=1}^{n} S_{i,j} x_j)^3$ is an expensive task as $n \geq 64$.
Here the approach is to use the fact that the d-th Hadamard product $z \circ z \circ z$ is equivalent to the d-th Face Splitting product of their transformation matrix S times the d-th tensor product of the input vector x where d is the degree, here $d = 3$.

$$P(x) = T \cdot (S \bullet S \bullet S) \cdot (x \otimes x \otimes x) = T \cdot (Sx \circ Sx \circ Sx)$$

The Face-Splitting product of matrices is represented as the tensor between their rows. Given $M = [r_1, \ldots, r_n] \in F_p^{n \times n}$ in row format, the operation gives $M \bullet M = [r_1 \otimes r_1, \ldots, r_n \otimes r_n] \in F_p^{n \times n^2}$ in row format.
Then the cubic Face-Splitting product is $M \bullet M \bullet M = [r_1 \otimes r_1 \otimes r_1, \ldots, r_n \otimes r_n \otimes r_n] \in F_p^{n \times n^3}$. Further we show how to eliminate redundancies from the public key reducing from a rectangular $n \times n^3$ to a $n \times \binom{n+2}{3}$ matrix.

2.3 3WISE Scheme

In MPKC we are interested on functions $P(X) = T \circ F \circ S(X)$ such that T, S are linear/affine maps and $F(X)$ is a quadratic or cubic set of equations. This is because $P(X)$ is non-linear thanks to the internal structure of $F(X)$. However, we need $F(X)$ to be a trapdoor function since recovering X from the public key $P^{-1}(Y) = X$ is considered hard.

To build such trapdoor functions a new family of private polynomials $F(X)$ is presented. The construction guarantees that is easy to evaluate the map but hard to recover the original point, in theory. Let's give a detailed description

2.3.1 Description

The base field prime p is an odd-prime where the order of the multiplicative group of units Z_p^* is coprime with 3, this is $\gcd(p-1, 3) = 1$. This guarantees that 3 has a multiplicative inverse $p-1$ this is $3 \cdot d \equiv 1 \pmod{p-1}$.
This fact enables us to decipher a ciphertext or sign a message, depending on the selected mode of the cryptosystem, as $y_i^d = (z_i^3)^d = z_i$

2.3.2 Trapdoor Permutation

Matrices T, S being invertible and the condition $\gcd(3, p-1) \neq 1$ guarantee that the scheme is a Trapdoor Permutation. As exponentiation by 3 has an inverse d then $z = x^3 = x \circ x \circ x \circ$ is recoverable as $z^d = x^{3 \cdot d} = x^1 = x$. All the operations are computed through bijective $F_p \mapsto F_p$ maps.

2.3.3 Key Generation

For building the Private and Public key we need first to select invertible matrices $S, T \in F_p^{n \times n}$. With that in mind, le'ts define both key generation procedures:

2.3.3.1 Public Key Construction

In order to generate a public key we must set-up the parameters of the scheme:

- Select an odd-prime where $\gcd(p-1,3)=1$, for example $p=17$. Here 3 is a unit in Z^*_{p-1} .
- Select invertible linear matrices $T,S \in F_p^{n\times n}$ which are used to hide the private polynomial map $F(X)$.
- Compute the cubic Face-Splitting product $M = S \bullet S \bullet S = [r_1 \otimes r_1 \otimes r_1, \ldots, r_n \otimes r_n \otimes r_n]$
- Compute $P = T \cdot M$ where $\cdot$ is the dot product.
- Now $P \cdot (x \otimes x \otimes x)$ is a representation of $T \cdot (Sx \circ Sx \circ Sx)$
- Reduce Public Key's size from a $n \times n^3$ matrix to a $n \times \binom{n+2}{3}$ as further seen.

2.3.3.2 Private Key Construction

The owner must retain matrices $T^{-1}, S^{-1} \in F_p^{n\times n}$

2.3.4 Signing process

2.3.4.1 Message Signing

- For signing a message m of any size compute the digest of the message via a Hash Function $H(m) = y$. Each numeric coefficient of the hash must be less than p. For example, in parametrization we select $p = 17$ so each coefficient is in hexadecimal (0 to 15).
- Compute $T^{-1} \cdot y = y' \in F_p^n$.
- Compute vector z by exponentiating every $y_i'^d \quad \forall 1 \leq i \leq n$ such that $y_i'^d = (z_i^3)^d = z_i$.
- Recover the signature as $x = S^{-1} \cdot z$.
- Send the signature-message pair (x, m) to the requester.

2.3.4.2 Message Verification

- The verifier must posses the Public Key in Tensor form $P(x) = A \cdot (x \otimes x) = y$. Reduced tensor version is used so the input vector is of $\binom{n+2}{3}$ length instead of n^3.
- Verifier receives a triplet (H, x, m) where H is the Hash function, x is the signature and m the message to be validated.
- Verifier computes $P(x) = H(m) = y$ and if its correct the signature is trusted as only the owner of the private key can issue valid signatures.

2.3.5 Key Sizes

The advantage of MPKC over other PQC candidates is the reduced signature bit length. However, it's been widely commented that MPKC has notorious trade-off between the signature bit length and public/private key pairs (specially HFE variants). Let's examine these cases:

2.3.5.1 Public Key

Recall that the Tensor representation of the Public Key is $P(x) = T \cdot (S \bullet S \bullet S) \cdot (x \otimes x \otimes x) = A \cdot (x \otimes x \otimes x)$. The matrix $A \in F_q^{n \times n^3}$ is the un-reduced public key, thus after reduction it takes $log_2 p \times n \times \binom{n+2}{3}$ size.

2.3.5.2 Reducing Public Key size

As the base field F_p is commutative we can reduce the tensor representation as tensor products of vectors $x \otimes x \otimes x$ have monomials $x_i x_j x_k$ that can be categorized into multiple cases:

- Case where $i \neq j \neq k$: Here we have 6 possible combinations, these are $ijk, ikj, jik, jki, kij, kji$. So we must sum up the coefficients of those matrices i.e $M_{i,jk} + \ldots + M_{k,j,i}$
- Case where $i = j \neq k$: Here we have 3 possible combinations, being: iik, iki, kii
- Case where $i \neq j = k$: 3 possible combinations: ijj, jij, jji

Every row of the un-reduced Public Key matrix of size $n \times n^3$ has n matrices of size $n \times n$, this is, every subrow of length n^2 is one of those matrices.
This method can be expanded to other multi-dimensional size like quadratic ($d = 2$) degree, where $i \neq j$ then $x_i x_j = x_j x_i$ are equal so we must sum up quadratic form coefficients into just 1 coefficient, thus reducing public key's size.

2.3.5.3 Private Key

The private key is comprised of matrices $T^{-1}, S^{-1} \in F_p^{n \times n}$ having $\log_2 p \times 2 \times n \times n$ bit size.

2.3.5.4 Signature Length

The signature $x = (x_1, \ldots, x_n) \in F_p^n$ has bit size equal to $\log_2 p \times n$.

2.3.6 Key Encoding and Decoding

NIST proposed a template for KAT values where Public and Private keys are represented as an unsigned char vector, this is, a byte vector. In the Parametrization section we define prime modulus $p < 256$ so every number in F_p fits in a byte (unsigned char).
For decoding, take an unsigned char that ranges from $0 - 255$ and convert it to an integer modulo p. By the condition stated above, as $p < 255$ there's no need to reduce modulo p, and the original encoded coefficient is preserved.

2.3.7 Signature encoding

As $p < 256$ the signature $x = (x_1, \ldots, x_n) \in F_p^n$ fits in a byte (unsigned char) vector of length n. This is, every element x_i is encoded as an integer from $0 - 255$.

3 List of parameter sets (part of 2.B.1)

- p is an odd prime defining F_p. The order of F_p^* must be coprime with 3. Here $p = 17$ is used as every coefficient of the hash of the message is viewed as an hexadecimal digit 0-15. The signature has characters ranging 0-16. This guarantees that every coefficient of the signature and message tuple are encoded as an integer into a byte.

- n is the number of variables and equations. The value is fixed depending on the utilized Hash Function, for example SHA-256 has 32 bytes or 64 hex digits. Truncated SHA-256 to 128 has 16 Bytes or 32 hex digits, for 192 we have 24 Bytes or 48 hex digits. Here every hex digit is taken as a whole byte i.e AF is taken as 10 15, thus every digit is taken as an integer, which is set to a byte variable.

- | pk | is the reduced Public Key size so it has **byte** size equal to $n \times \binom{n+2}{3}$

Scheme	Security Level	p	n	| pk | (KB)	| sk | (KB)	sign (B)
3WISE-128	2	17	32	187	2	32
3WISE-192	4	17	48	918.75	4.5	48
3WISE-256	5	17	64	2860	8	64

Table 1: Parameter list for Security Levels KB:Kilo-Byte, B:Byte

4 Design rationale (part of 2.B.1)

The field of Multivariate Public Key Cryptography counts with a taxonomy that categorizes schemes into families: BigField, MediumField, Stepwise, Oil-Vinegar [WP]. Schemes based on these families play an important role in the development and study of Post-Quantum schemes as there is a need for strengthening key exchange found in the Internet (i.e: SSL/TLS) and for digital signatures. It's theorized that in next decades Quantum computers will be ready to break the schemes that we use today, as they're based in common problems found in commutative cryptography where the security relies in the Discrete Logarithm Problem and Integer Factorization.
With the introduction of the C* scheme by Matsumoto-Imai the field of MPKC started to gain attention [TH]. The C* scheme was broken in the work of Patarin [Pata] by a Differential attack. Attacker gathers plaintext and ciphertext pairs and mounts a linear equation system that recovers the coefficient of the quadratic equations obtained by the Differential. With these equations Patarin demonstrated that plaintext recovery is doable for C*. Other variations were done like the Perturbation modifier.
After this breakthrough HFE [Patb] gained attention, which is a Dembowski-Ostrom private polynomial, that is represented as a quadratic set (system of quadratic equations). HFE has its weakness on decipher stage, where the Degree of the central polynomial $F(X)$ must be small to apply root finding (e.g: Berlekamp's Trace). Kipnis and Shamir published a work [KS] demonstrating that private polynomial computation is feasible solving the Minrank problem by solving a multivariate system of equations using the relinearization technique. This is because the rank of the private polynomial is considered small.
Variations of HFE appeared to protect from these key recovery attacks. (Gui, HFEv-, GeMMs, QUARTZ), that nowadays are considered not secure as it's been proved that are not resistant to recent discoveries [STV] [TV]. In addition, the underlying problems of MPKC have been broadly studied: PoSSo, Minrank [Bus], Isomorphism of Polynomials (IP2) [Pata] .

With that all in mind the design of **3WISE** is based on concepts that have not been extensively covered as cubic degree schemes has not attracted the same level of attention as quadratic scheme did. For the perspective of security are believed to be stronger than quadratic ones, but in terms of data representation (storage) and performance it seems that perform worse than quadratics. The advantage of 3WISE being cubic is that it's internal operations are quite simpler than other schemes that require a full matrix tensor to be represented like HFE based schemes. So being cubic is not a non-stopper for building the public key faster than a quadratic scheme for the same parametrization m, n, q. This has been demonstrated as the Face Splitting product is faster than the full tensor approach.

5 Detailed performance analysis (part of 2.B.2)

5.1 Testing Platform

The reference and optimized implementation have been tested on a single platform. KAT values have been generated on the same workstation too.

Computer	Processor	Frequency (MHz)	Max freq. (MHz)
Workstation	AMD Ryzen 3700x	3600	4200
Embedded	Raspberry PI 4b	1500	1500

Table 2: Description of the testing platform.

Computer	OS	Kernel	RAM
Workstation	Arch Linux	6.1.12	32GB
Embedded	OpenBSD 7.3	GENERIC.MP	4GB

Table 3: Description of OS and RAM

5.2 Third Party Open Source Libraries

The reference and optimized implementation make use of the C library FLINT, which is used for implementing all low level Linear Linear algebra operations related to compressed cubic kronecker products. FLINT has native support for some operations related to matrix multiplication, matrix inverses, rank, random number generation, finite field operations, etc. Other references to libraries and function calls should be POSIX compatible.

5.2.1 Differences between Operating Systems

There are major differences between the packaging found among Unix-like OS and GNU/Linux distributions.

5.2.1.1 GNU/LINUX

In Arch Linux and Kali (Debian based) FLINT is at version 2.9.0 which guarantees that some primitives like *fmpz_mod_mat_rank(), fmpz_mod_mat_inv* do exist. The implementations do rely on these primitives which are not present for example in Ubuntu 22.04. Manual installation of FLINT 2.9.0 is mandatory in those cases since compilation with GCC will throw errors as it cannot find those function calls.

5.2.1.2 BSD

In the case of other Unix-Like OS, OpenBSD has been tested on a Raspberry PI 4b (aarch64). FLINT was manually compiled and installed with gmake ang gcc. The implementations compiled by passing the argument "-I /usr/local/includes" so the compiler locates FLINT's header files. Here the Clang C compiler has been used.

5.3 Reference vs Optimized implementation

The reference and optimized implementations are the same at code level.

5.3.1 Benchmark

The MAKEFILE has an available benchmark build option called the "Fast Test" which is compiled with optimizations with the "-Ofast" compiler flag. The results indicate that signing is faster than verifying as the operations involved on signing are faster than the linear algebra done on the tensor reduction of $x \otimes x \otimes x$ for the cubic degree case.

Scheme	Gen	Sign	Verify
3WISE-128	34.11ms	38μs	1.11ms
3WISE-192	255.51ms	60μs	5.22ms
3WISE-256	958.42ms	93μs	15.33ms

Table 4: Median time of distinct Parametrizations in 20 rounds in *Workstation*

Scheme	Gen	Sign	Verify
3WISE-128	263.1ms	170μs	9.14ms
3WISE-192	1.5s	339μs	58.5ms
3WISE-256	5.3s	569μs	165.31ms

Table 5: Median time of distinct Parametrizations in 20 rounds in *Embedded*

5.4 Workstation vs Embedded

Embedded machines are great candidates for running the scheme in all modes (Sign, Verify and KeyGen). In general, tested machines run quite fast in every parametrization case: 128, 192 and 256.

6 Expected Strength (part of 2.B.4)

6.1 EUF-CMA security

The presented scheme must be analyzed from the perspective of the **EUF-CMA** security model applied to digital signatures. The model proposes the following conditions:

- Challenger $\mathcal{C}$ generates a pair of public-private keys (pk, sk) and sends the public key pk to the adversary $\mathcal{A}$.
- The adversary $\mathcal{A}$ has access to the oracle and queries for the message m.
- The oracle returns the signature $\theta \leftarrow \text{Sign}(sk, m)$ and stores the message m into the message list $\mathcal{Q}$, so every submitted message by $\mathcal{A}$ has to be not repeated or it will be discarded by the Oracle and/or the Challenger $\mathcal{C}$.
- $\mathcal{A}$ wins when finds a valid pair (m^*, θ^*) where $\text{Verify}_{pk}(m^*, \theta^*) = 1$ and $m^* \notin \mathcal{Q}$, this is, the message m^* must not be submitted to the oracle and θ^* is a valid signature for m^*.

6.1.1 Signature Forgery

The adversary $\mathcal{A}$ has the public key and at most 2^{64} valid message and signature pairs generated by the Oracle. With this information, $\mathcal{A}$ cannot generate a linear attack as the scheme is non-linear. In addition, the scheme consists a bijective map (trapdoor permutation), given $m^* \notin \mathcal{Q}$ there exists an unique signature θ^* such that its image on the public key gives the digest of m^*, this is: $P(\theta^*) = H(m^*)$. Such unique relation can only be found by inverting the trapdoor, which would require to solve an instance of the PoSSo (Polynomial System Solving) problem over F_p or the IP2 problem, which is the Isomorphism of Polynomials, where T and S are to be found, or equivalent ones such that the public key generated by T' and S' gives the relation $P(X) = P'(X)$. In the case of solving PoSSo, it can be addressed by direct Gröbner bases attack which would involve n cubic equations in n variables. Other technique is to compute the Multivariate Differential of the cubic map $P(X)$, which is a quadratic system of n equations in n variables. Both techniques are computationally infeasible for the given parameter list. The internal structure of the scheme must be exploited in order to cut off a significant magnitude on the general time complexity of these attacks.

7 Analysis of known attacks (part of 2.B.5)

The proposed scheme consists of n cubic equations in n variables, which is not a common area of research in the field of MPKC. The most remarkable schemes are ABC cryptosystem [PW] and the cubic variant of HFE [Ea], which both have vulnerabilities in their internal structure [PST] that eventually lead to rupture. Therefore, do not take the following analysis as the representation of the possible reality of the required attack complexity.

7.1 Index & Degree Regularity

There are distinct cases for estimating attacks using Gröbner in multivariate polynomial equation systems. For example, under some conditions, over-defined systems are easier to solve than under-determined or systems where $m = n$ which is the case of **3WISE**. The goal is to demonstrate that a system belongs to the *worst-case family* of polynomials such that the computed Gröbner basis has (almost) maximal degree of regularity, this is $d_{reg} \leq \#MB = \sum_{i=1}^{n}(d_i - 1) + 1$, where MB is the *Macaulay Bound*, an upper bound that defines the highest degree that a term can have in the resulting Gröbner basis.
In general, for regular systems where $m = n$ the index of regularity plus one coincides with the upper bound MB so we conclude that $d_{reg} \leq i_{reg} + 1$. The *index of regularity*, i_{reg} is the degree of $HS_{\mathcal{I}}(t) = \frac{\prod_{i=1}^{n}(1-t^{degp_i})}{(1-t)^n}$, the Hilbert Series polynomial of the Ideal generated by the polynomials of the system $P(X) - Y = 0$.

7.2 Gröbner Bases

The Polynomial System Solving (PoSSo) problem in MPKC is based on finding the set of roots over F_p from the Tensor representation of the public key $P(x) - y = 0$. The normal approach is to solve it by using Gröbner Basis, which finds an Algebraic Variety that contains the roots (solution) of the system.
Testing has been conducted on instances of 3WISE where $5 \leq n \leq 10$ using Wolfram Mathematica for generating the symbolic public key and SAGE for computing Gröbner basis, i_{reg} and d_{reg}. Cubic equations are placed in a text file, then loaded via Sage to generate a Gröbner basis. d_{reg} is the highest degree found among the polynomials in the basis. For computing i_{reg}, the program finds the degree of the Hilbert series of the Ideal generated by the leading monomial terms of the basis. In all instances d_{reg} is close to i_{reg} or $i_{reg} + 1$ which equals to the Macaulay's bound, theorizing that generated instances are in the "worst-case" space of multivariate equations. Monomial ordering selected is **degree reverse lexicographic**. **MB** stands for *Macaulay's Bound*.

7.2.1 Results

Every test has the nomenclature $nX - Y$ where X is the number of variables and Y the test number, for example, n6-1 is the first test of a random instance of 3WISE with $n = 6$. Each test is available in Github and can be replicated with the code found in the same repository. Gröbner basis computation with *giac:gbasis* method should provide same values for the degree of regularity and number of equations, but this may vary depending the version of libraries found in the system.

$\#Test$	d_{reg}	i_{reg}	MB	$\#npolys$	$\#Test$	d_{reg}	i_{reg}	MB	$\#npolys$
n6-1	12	12	13	172	n7-1	14	14	15	496
n6-2	12	12	13	171	n7-2	15	14	15	558
n6-3	13	12	13	205	n7-3	15	14	15	558
n6-4	13	12	13	197	n7-4	14	14	15	470
n6-5	13	12	13	207	n7-5	14	14	15	469

Table 6: d_{reg} for 5 instances of 3WISE with $n = 6, 7$

$\#Test$	d_{reg}	i_{reg}	MB	$\#npolys$	$\#Test$	d_{reg}	i_{reg}	MB	$\#npolys$
n8-1	17	16	17	1526	n9-1	17	18	19	4224
n8-2	16	16	17	1288	n9-2	17	18	19	4224
n8-3	17	16	17	1524	n9-3	17	18	19	4221
n8-4	17	16	17	1525	n9-4	18	18	19	3591
n8-5	17	16	17	1526	n9-5	18	18	19	3591

Table 7: d_{reg} for 5 instances of 3WISE with $n = 8, 9$

7.2.2 Complexity of parameter list

Data indicates that d_{reg} is close to MB for small n. However, it wouldn't be prudent to extrapolate this fact to other instances with bigger n as for the parameter list $n = 32, 48, 64$. For example, d_{reg} has a difference of two with MB at $n = 9$. Taking this into account, the complexity of Gröbner basis computation over F_p is divided into two categories: Bound and Conservative. In bound d_{reg} equals to *Macaulay's bound* (worst-case scenario) and Conservative has $d_{reg} = MB - 4$ as the degree of regularity tends to differ from MB as n increases, for some random instances. This is, there are instances where $d_{reg} = MB$ however there are some that not, this is why conservative offset of four has been selected. The nº of field operations required for computing a Gröbner basis can be estimated as:

$$\mathcal{O}(p \cdot \binom{n + d_{reg}}{d_{reg}}^{w})$$

Where $w = 2$ is set up conservatively and $p = 17$ is the selected odd-prime for the scheme.

Scheme	MB	Conservative
3WISE-128	2^{175}	2^{170}
3WISE-192	2^{263}	2^{258}
3WISE-256	2^{350}	2^{345}

Table 8: Complexity of Gröbner basis computation

7.3 MinRank

The MinRank problem for quadratic schemes MR(m, n, r) consists of finding a linear combination of m matrices of size $n \times n$ to obtain a $n \times n$ matrix of rank r. MinRank problem is NP-complete [Bus]. However, the approach employed in **3WISE** should be different as the degree is cubic. Matrices are of type $M_i \in F_p^{n \times n \times n}$ so the rank condition has to be extended to a three dimensional tensor space. In the cryptosystem ABC column band separation was derived from the Differential of public key equations [PST]. Other practices require to express the Differential of a cubic scheme as a quadratic set of equations, where MinRank is applied. In [Ea] the KS attack is applied to a cubic variant of HFE.
Attacking a scheme via MinRank requires that the internal structure of the underlying algorithm to be vulnerable to algebraic relations that result in efficient separation of equations or low-rank matrices that permit recovering matrices T, S or equivalent T', S' such that $T \circ F \circ S = T' \circ F \circ S'$.

8 Advantages and limitations (part of 2.B.6)

8.1 Advantages

- **Small Signatures:** Schemes based on Multivariate Cryptography are well known for their small signature size. Signatures are sent along with the message to the verifier, so it doesn't take much bandwith over a network.
- **Key Generation:** Cubic Face Splitting product is way better than a cubic complete Kronecker Product of matrices. It has a speeduf factor of n. For example a cubic HFE would be overkill using all the exposed concepts in the paper.
- **Fast Verification:** The verification of a signature is really fast for all the covered parameters using compressed tensor forms.
- **Fast Signing:** Message signing has demonstrated to be the fastest procedure between generation and verification.
- **Arithmetic:** The operations done by the scheme are easily handled by any electronic device as the scheme mainly relies in Linear Algebra over F_{17}.

8.2 Limitations

- **Uncommon area:** MPKC schemes where equations have cubic degree are uncommon area of study. This is because quadratic degree schemes offer lower bit size in terms of public key. Here we can notice that the trade-off of changing to $d = 3$ doesn't impact that much on the performance and the bit size of involved key material.

References

[WP] Christopher Wolf and Bart Preneel. *Taxonomy of Public Key Schemes based on the problem of Multivariate Quadratic equations*. URL: https://eprint.iacr.org/2005/077.pdf.

[TH] T.Matsumoto and H.Imai. *Public Quadratic Polynomial-tuples for eficient sigriature-verification and message encryption*. URL: https://link.springer.com/chapter/10.1007/3-540-45961-8_39.

[Pata] Jacques Patarin. *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'98*. URL: https://link.springer.com/article/10.1023/A:1008341625464.

[Patb] Jacques Patarin. *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms*. URL: http://www.minrank.org/hfe.pdf.

[KS] Aviad Kipnis and Adi Shamir. *Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization*. URL: https://link.springer.com/chapter/10.1007/3-540-48405-1_2.

[STV] John Baena , Pierre Briaud , Daniel Cabarcas , Ray Perlner , Daniel Smith-Tone and Javier Verbel. *Improving Support-Minors Rank Attacks: Applications to GeMSS and Rainbow*. URL: https://eprint.iacr.org/2021/1677.pdf.

[TV] Magali Bardet , Maxime Bros , Daniel Cabarcas , Philippe Gaborit , Ray Perlner , Daniel Smith-Tone , Jean-Pierre Tillich and Javier Verbel. *Improvements of Algebraic Attacks for solving the Rank Decoding and MinRank problems*. URL: https://arxiv.org/pdf/2002.08322.pdf.

[Bus] J.O. Shallit , G.S. Frandsen , J.F. Buss. *The computational complexity of some problems of linear algebra*. URL: https://www.brics.dk/RS/96/33/BRICS-RS-96-33.pdf.

[PW] J.Ding , A. Petzoldt and L. Wang. *The cubic simple matrix encryption scheme*. URL: https://link.springer.com/chapter/10.1007/978-3-319-69453-5_29.

[Ea] J.Baena , D. Cabarcas , D. Escudero and et al. *Rank analysis of cubic multivariate cryptosystems*. URL: https://eprint.iacr.org/2018/110.pdf.

[PST] Dustin Moody , Ray Perlner and Daniel Smith-Tone. *Key Recovery Attack on the Cubic ABC Simple Matrix Multivariate Encryption Scheme*. URL: https://link.springer.com/chapter/10.1007/978-3-319-69453-5_29.

9 2.D.1 Statement by Each Submitter

I, Borja Gómez Rodríguez, of PZ Landabaso 8 - 5ºA, Bilbao 48015, Spain, do hereby declare that the cryptosystem, reference implementation, or optimized implementations that I have submitted, known as 3WISE, is my own original work, or if submitted jointly with others, is the original work of the joint submitters. I further declare that (check one):

☑ a. *I do not hold and do not intend to hold any patent or patent application with a claim or that could be amended to include a claim that may cover the cryptosystem, reference implementation, or optimized implementations that I have submitted, known as 3WISE; OR (check one or both of the following):*

☐ b. *to the best of my knowledge, the practice of the cryptosystem, reference implementation, or optimized implementations that I have submitted, known as 3WISE may be covered by the following U.S. and/or foreign patents: (describe and enumerate or state "none" if applicable)*
None

☐ c. *to the best of my knowledge, the following pending U.S. and/or foreign patent applications may cover the practice of my submitted cryptosystem, reference implementation or optimized implementations: (describe and enumerate or state "none" if applicable)*
None

I do hereby acknowledge and agree that my submitted cryptosystem will be provided to the public for review and will be evaluated by NIST, and that it might not be selected for standardization by NIST. I further acknowledge that I will not receive financial or other compensation from the U.S. Government for my submission. I certify that, to the best of my knowledge, I have fully disclosed all patents and patent applications which may cover my cryptosystem, reference implementation or optimized implementations. I also acknowledge and agree that the U.S. Government may, during the public review and the evaluation process, and, if my submitted cryptosystem is selected for standardization, during the lifetime of the standard, modify my submitted cryptosystem's specifications (e.g., to protect against a newly discovered vulnerability)

I acknowledge that NIST will announce any selected cryptosystem(s) and proceed to publish the draft standards for public comment. I do hereby agree to provide the statements required by Sections 2.D.2 and 2.D.3, below, for any patent or patent application identified to cover the practice of my cryptosystem, reference implementation or optimized implementations and the right to use such implementations for the purposes of the public review and evaluation process.

I acknowledge that, during the post-quantum algorithm evaluation process, NIST may remove my cryptosystem from consideration for standardization. If my cryptosystem (or the derived cryptosystem) is removed from consideration for standardization or withdrawn from consideration by all submitter(s) and owner(s), I understand that rights granted and assurances made under Sections 2.D.1, 2.D.2 and 2.D.3, including use rights of the reference and optimized implementations, may be withdrawn by the submitter(s) and owner(s), as appropriate.

Signed: Borja Gómez Rodríguez

Title: 3WISE
Date: 31/05/2023
Place: Santiago, Chile

10 2.D.3 Statement by Reference/Optimized Implementation's Owner

I, Borja Gómez Rodríguez, PZ Landabaso 8 - 5ºA, Bilbao 48015, Spain, am the owner or authorized representative of the owner of the submitted reference implementation and optimized implementations and hereby grant the U.S. Government and any interested party the right to reproduce, prepare derivative works based upon, distribute copies of, and display such implementations for the purposes of the post-quantum algorithm public review and evaluation process, and implementation if the corresponding cryptosystem is selected for standardization and as a standard, notwithstanding that the implementations may be copyrighted or copyrightable.

Signed: Borja Gómez Rodríguez

Title: 3WISE
Date: 31/05/2023
Place: Santiago, Chile