Modified Parameters of Rainbow in Response to a Refined Analysis of the Rainbow Band Separation Attack by the NIST Team and the Recent New MinRank attacks

Rainbow Team

Updated September 18, 2020

Recently, the Rainbow team was notified by Ray Perlner and Daniel Smith-Tone from the NIST PQC team of a new refined analysis of the Rainbow Band Separation (RBS) Attack using the natural variable partition of the polynomial system generated by the attack.

(A) Cryptology ePrint Archive: Report 2020/702 Rainbow Band Separation is Better than we Thought, Ray Perlner and Daniel Smith-Tone

This new analysis makes full use of the polynomial structure and therefore gains a few bits in terms of attack efficiency. Since the parameters of the submitted Rainbow instance were chosen almost exactly according to the NIST requirement, we have to modify the parameters of the scheme slightly to address this new refined analysis.

Furthermore, the Rainbow team has been paying close attention to the recent development of a new method to improve the MinRank attack. This line of work started with a paper coauthored by the NIST team, too:

(1) Javier A. Verbel, John Baena, Daniel Cabarcas, Ray A. Perlner, Daniel Smith-Tone: On the Complexity of "Superdetermined" Minrank Instances. PQCrypto 2019: 167-186.

Before of that, the methods used to evaluate the complexity of the MinRank attack mainly relied on the paper:

(2) Jean-Charles Faugere, Francoise Levy-dit-Vehel, Ludovic Perret: Cryptanalysis of MinRank. CRYPTO 2008: 280-296,

in which the authors claimed that they proved the best attacking method is to use the minors of the matrix generated by a linear combination of given matrices.

Since the cost of a MinRank attack was very high according to this estimate, the impact of the MinRank attack against the Rainbow cryptosystem was believed to be very small.

However the paper **(1)** clearly shows that the proof in the Crypto 2008 was completely wrong and furthermore using the idea of by this work, another paper:

(3) Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, Vincent Neiger, Olivier Ruatta, Jean-Pierre Tillich: An Algebraic Attack on Rank Metric Code-Based Cryptosystems. EUROCRYPT (3) 2020: 64-93,

breaks the Minrank based Code-based schemes in the NIST second round submission.

Then in 2020, there was a new development in this direction:

(4) Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray A. Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, Javier A. Verbel: Algebraic attacks for solving the Rank Decoding and MinRank problems without Gröbner basis. CoRR abs/2002.08322 (2020),

which further demonstrates the complete failure of the proof in (2), and destroyed the MinRank based Code-based schemes in the NIST second round submission. In this paper, the authors further applied their excellent methods to attack Rainbow, which reduced the attack complexity tremendously. However, the complexity of a MinRank attack against Rainbow is still above the NIST security requirements.

We have been studying the paper **(4)** very carefully, and we believe it gives the best possible attack using the MinRank method from the theoretical perspective. Furthermore the authors shed new lights on how we should select better parameters such that we can improve the security without much cost. Therefore, we would like to adjust our parameters with full consideration of these new attacks as well.

Overall, we would like to point out that the development of these new attacks does not affect the fundamentals of the security analysis of Rainbow and it actually increases our confidence in the overall design of Rainbow. The modifications have very small impact on efficiency of the Rainbow signature scheme.

We will recap first the work by NIST PQC team on the Rainbow Band Separation attack using the Wiedemann XL algorithm and how to address this problem (see Section 1 and 2). Secondly, in Section 3, we will address the new MinRank attacks and our conclusions from it. Finally we will present in Section 4 the new modified parameters for the Rainbow signature scheme.

1 Bipartite XL Attack and Parameters

We will for brevity call the specialized XL-like attack on equations of bi-degree $(2,0)$ and $(1,1)$ a "Bipartite XL Attack".

Note: By a bi-degree (a,b) polynomial, we mean a polynomial with a partition of variables into two subsets X and Y, such that the highest degree in X is a and the highest degree in Y is b. Given a collection of terms and equations, we mean by N "missing equations" or "residual degrees of freedom" that, after performing Gaussian Elimination on the equations, N of the terms still remain uneliminated.

1.1 Basic Facts about Bipartite XL

1. The NIST PQC team proposed a generating function expression

   $$h(\alpha,\beta;n_x,n_y;m_x,m_{xy}) := [t^\alpha s^\beta]\left(\frac{(1-t^2)^{m_x}(1-ts)^{m_{xy}}}{(1-t)^{1+n_x}(1-s)^{1+n_y}}\right)$$

   to characterize the remaining missing equations (e.g. residual degrees of freedom) for bi-degrees less than (α,β) in a Bipartite XL algorithm, for a generic system with n_x X-variables, n_y Y-variables, m_x bi-degree $(2,0)$ equations, and m_{xy} bi-degree $(1,1)$ equations, in sufficiently large fields, as long as it is greater than zero. In this Bipartite XL algorithm, we multiply the quadratic system by monomials such that the maximum degree allowed is a given bi-degree (α,β).

   When this expression is nonpositive, the system can be solved using Bipartite XL.

2. We want to use Block Wiedemann.

   Let $T = T^{(\alpha,\beta)}$ be the #terms with bi-degree less than (α,β).

   In our range of parameters we have $T^{(1,1)} < T^{(2,0)}$, so we will be using

   $$T' = T - h(\alpha,\beta;n_x,n_y;m_x,m_{xy})$$

   equations generated from bi-degree $(1,1)$ equations, and $T-T'$ equations generated from bi-degree $(2,0)$ equations.

   The total #terms in these T equations is

   $$K' = \left(T^{(1,1)}T' + T^{(2,0)}(T-T')\right),$$

   and a lower bound on the number of multiplications used in the Bipartite XL is thus $3T^2K'$.

If we set

$$R = R^{(\alpha,\beta)} = m_x T^{(\alpha-2,\beta)} + m_{xy} T^{(\alpha-1,\beta-1)}$$

to be the total #equations generated up to bi-degree (α, β), and

$$K = m_x T^{(2,0)} T^{(\alpha-2,\beta)} + m_{xy} T^{(1,1)} T^{(\alpha-1,\beta-1)}$$

to be the total #terms in those R equations, an upper bound on the number of multiplications is $3RK$.

This new refined analysis of the NIST PQC team is natural but very clever and makes full use of the bipartite structure of the polynomial system to derive a more precise analysis of the complexity.

1.2 RBS Complexities in View of Bipartite XL

From the refined analysis, it is very clear to us that to address the security parameter modification, we only need to slightly increase the vinegar variables on the first Rainbow layer.

Armed with the formulas from above, we may recompute the complexities required for the Rainbow Band Separation (RBS) attack for proposed Rainbow instances in Round 2 of the NIST PQC Competition below in Table 1.

Scheme	bi-degree	mult	gates	AES/SHA3	Required	Level
Rainbow-Ia(32,32,32;16)	$(12,4)$	134	139	124	128 (AES)	I
Rainbow-$I'a$(34,32,32;16)	$(14,3)$	139	144	129		
Rainbow-$I''a$(36,32,32;16)	$(13,4)$	141	147	132		
Rainbow-$IIIc$(68,36,36;256)	$(15,9)$	197	204	186	192(SHA3)	III
Rainbow-$III'c$(72,36,36;256)	$(15,10)$	203	210	192		
Rainbow-$III''c$(76,36,36;256)	$(18,7)$	208	215	197		
Rainbow-$III\dagger c$(68,32,48;256)	$(14,12)$	210	217	199		
Rainbow-Vc(92,48,48;256)	$(19,12)$	256	263	245	256(SHA3)	V
Rainbow-$V'c$(100,48,48;256)	$(21,11)$	267	274	256		
Rainbow-$V''c$(104,48,48;256)	$(20,13)$	271	278	260		
Rainbow-$V\dagger c(96, 36, 64; 256)$	$(21,12)$	274	281	263		

Table 1: Bipartite XL in RBS and Rainbow Instances

We should notice that these attacks assume a somewhat impractical constant time memory access, since the memory requirements for these XL attacks are huge. However, in the NIST Call for Proposals document about Category 1, NIST says that every attack “must require computational resources comparable to or greater than those required for key search on a block cipher with a 128-bit key (e.g. AES128)”. This requirement must be met in ”*all* metrics that NIST deems to be potentially relevant to practical security”.

One of the metrics already mentioned in the call is “classical gates”; NIST estimates that AES-128 key search uses about 2^{143} “classical gates”.

In view of these revised estimates of this known attack, to which we agree, Rainbow Ia, IIIc, and Vc do not meet their proposed security levels by a few bits (4, 6, 9 respectively) and needs to be amended.

1.3 Modified Parameters in View of Bipartite XL

For each parameter set $(v, o_1, o_2; q)$, we increased v until the complexity of the RBS attack using Bipartite XL is at least equal to the required level. We then continued to increase v until we get at least 2^4 times the required level. Our new parameters will keep the oil variable size intact.

Besides increasing the complexity of Bipartite-XL attacks, increasing the parameter v increases the complexity of the UOV attack significantly. Furthermore, it slightly increases the complexities of the Rank attacks. On the downside, increasing v might have a negative effect on the complexity of direct attacks. If the number of variables n in the public system exceeds the number of equations m by a factor of two, we can use a method of Thomae to transform the public system into an equivalent system of $m-1$ equations in $m-1$ variables. For the parameters for NIST security level I, we are far away from this case, so the complexity of direct attacks is not changed by increasing the parameter v. For the higher security levels we find that, after increasing v, the number of variables now exceeds twice the number of equations, which decreases the complexity of direct attacks against these schemes. However we find that that the decrease in complexity is roughly 1 bit, which implies that the complexity of direct attacks against these schemes is still well beyond the required security levels.

Overall these updates make essentially no difference in other standard Rainbow attacks. RBS remains the most potent classical attack and is the decisive factor in our parameter choices.

2 The Putative Bipartite XL2 Attack

The NIST PQC team also presented a security analysis based on the first fall degree, which is also called the mutant degree. At this degree, due to a non-trivial syzygy, new lower degree polynomial, mutants, are produced.

We agree that a modified XL-style algorithm will produce a degree fall and therefore mutants at the lowest bi-degree (a, b) for which

$$\overline{h}(a, b; n_x, n_y; m_x, m_{xy}) := [t^a s^b] \left(\frac{(1-t^2)^{m_x} (1-ts)^{m_{xy}}}{(1-t)^{n_x} (1-s)^{n_y}} \right)$$

is negative. This means among other things that the first fall degree in a F_4 algorithm would be no higher than $a+b$. However, we *present a refined analysis on whether this will lead to a valid, faster attack.* We show:

1. The attack using these mutants would happen under slightly different conditions than what the NIST PQC team depicted; *and*

2. The attack using these mutants, even when it works, actually has a higher cost than the full Wiedemann Bipartite XL attack in general.

We will take as an illustrative example the RBS system created when attacking Rainbow-$I''a(36, 32, 32; 16)$. In this attack, we need to solve a system with 68 X-variables, 32 Y-variables, 64 bi-degree $(2, 0)$-equations and 99 bi-degree $(1, 1)$ equations. We have

$$\begin{aligned} h(9, 8; 68, 32; 64, 99) &= 11622781589490522 \gtrsim 2^{53} > 0 \\ \overline{h}(9, 8; 68, 32; 64, 99) &= -2564665346260512 \gtrsim -2^{51.2} < 0 \end{aligned}$$

Therefore, Bipartite XL is not expected to result in a solution at bi-degree $(9, 8)$. The reason for this is that there are $h(9, 8; 68, 32; 64, 99) \gtrsim 2^{53}$ remaining missing equations, while there are $\overline{h}(9, 8; 68, 32; 64, 99) \lesssim 2^{51.2}$ mutants at a lower degree (≤ 16) to be found whose highest order terms are in general of degree 16, at bi-degree (8, 8) or degree (9,7).

The NIST team's FirstFall attack suggests that a block Wiedemann on a system with $\overline{T}^{(9,8)} = 87650838664469273200 \lesssim 2^{63}$ variables (where $\overline{T}^{(a,b)} =$ #terms at exactly bi-degree (a, b), as opposed to at that or lower), and at least 68×32 terms per column, which takes around $3\left(\overline{T}^{(9,8)}\right)^2(68 \times 32) \gtrsim 2^{138.5}$, can eventually lead to a solution. We will show

1. that the first fall at bi-degree $(9, 8)$ doesn't lead to a solution, although the first fall at bi-degree $(10, 7)$ (or $(11, 6)$, or $(12, 5)$) does; *and*

2. that finding enough mutants costs more than direct Bipartite XL; *and*

3. that going from the mutants through a final elimination also takes more effort than the direct Bipartite XL.

2.1 Illustrative "First Fall" (Non-Bipartite) XL2

We will first recap a case where the Wiedemann XL2 idea seems to work. Let there be $n = 22$ variables, and $m = 44$ equations. The Wiedemann XL attack happens at degree $D = 6$ because

$$\begin{aligned} h(6;22;44) = [t^6]\left(\frac{(1-t^2)^{44}}{(1-t)^{22+1}}\right) &= -33208 < 0, \\ h(5;22;44) = [t^5]\left(\frac{(1-t^2)^{44}}{(1-t)^{22+1}}\right) &= 1288 > 0. \end{aligned}$$

But, we are able to conduct a Wiedemann XL2 attack at $D = 5$ because

$$\bar{h}(5;22;44) = [t^5]\left(\frac{(1-t^2)^{44}}{(1-t)^{22}}\right) = -2464 < 0.$$

To be more precise, let $\bar{T}^{(D)}(n) = [t^D](1-t)^{-n}$ and $T^{(D)}(n) = [t^D](1-t)^{-(n+1)}$ be the number of terms at and up to degree D with n variables respectively. If we randomly select $\bar{T}^{(5)}(22) - \bar{h}(5;22;44) = 68244$ equations from the $\bar{R}^{(5)}(22;44) = 44 \times \bar{T}^{(3)}(22) = 89056$ relations we obtain by multiplying all degree-3 terms to the 44 original equations, we can likely obtain all $-\bar{h}(5;22;44)$ nontrivial degree-falls at degree 5 by solving for vanishing linear combinations of rows of the Macaulay matrix.

Note: the initial block Wiedemann takes about $3 \times \left(\bar{T}^{(5)}(22)\right)^2 \times \bar{T}^{(2)}(22) = 3284199375600 \approx 2^{41.5}$ multiplications. If we can somehow run this initial block Wiedemann with a block size of 64 (or repeat the block Wiedemann a few times with a block size of 16 or 32), then we can obtain 64 mutants each with $T^{(4)}(22) = 14950$ terms. By multiplying each of these mutants with each variable, we obtain $22 \times 64 = 1408$ equations, totalling $1408 \times 14950 = 21049600 = 2^{24.3}$ terms in all. The Macaulay-like system at degree 5 had $R^{(5)}(22;44) = 44 \times T^{(3)}(22) = 101200$ equations for $101200 \times T^{(2)}(22) = 27931200 \approx 2^{24.7}$ terms, totalling $48980800 = 2^{25.5}$ terms in the system. We can now run a second Wiedemann with $T^{(5)}(22) = 80730$ variables and complexity $3 \times 80730 \times 48980800 = 11862659952000 \approx 2^{43.5}$ multiplications, versus the direct block Wiedemann XL at degree 6 using $3 \times \left(T^{(6)}(22)\right)^2 \times T^{(2)}(22) = 243253449757720561500 = 2^{46.7}$ multiplications.

Criteria for Wiedemann XL2: The reasons why this works, which we shall verify to not hold for the putative Wiedemann Bipartite XL2, are:

1. One (or few) run of block Wiedemann finds enough nontrivial degree-fall linear combinations (mutants) to make up for the missing equations;

2. mutants multiplied by variables add new equations without adding new terms we must cater to (*else the system size explodes*); *and*

3. Few mutants and new equations need to be added (*else the system becomes much bigger, and sparse matrix methods becomes impossible*).

This means that the system can only be solved using sparse XL if we have not too many mutants of high degree.

2.2 The "First Fall" Bipartite XL2?

Now let's go back to the RBS system system created when attacking Rainbow-$I''a(36, 32, 32; 16)$. There are 68 X-variables, 32 Y-variables, 64 bi-degree $(2, 0)$-equations and 99 bi-degree $(1, 1)$ equations. We multiply the 64 bi-degree $(2, 0)$-equations by all bi-degree $(7, 8)$ monomials and the 99 bi-degree $(1, 1)$ equations by all bi-degree $(8, 7)$ monomials, and have sufficiently many equations to eliminate all bi-degree $(9, 8)$ monomials to get degree-16 mutants.

But these mutants have leading terms both of bi-degree $(9, 7)$ *and* $(8, 8)$. If we plan to multiply the result by an X-variable, we would get terms of bi-degree $(10, 7)$; if we multiply by a Y-variable, we get terms of bi-degree $(8, 9)$. *In either case, our collection of terms to eliminate has gotten larger.* With that, the number of extra equations we need to generate goes up, by $\overline{T}^{(10,7)}(68, 32) \gtrsim 2^{63}$ or $\overline{T}^{(9,8)}(68, 32) \gtrsim 2^{62}$, and these equations will all be *dense* not sparse equations once we generate them, with $T^{(9,8)}(68, 32) - \overline{T}^{(9,8)}(68, 32) \gtrsim 2^{61.6}$ terms per equations. Surely 2^{123} terms in memory is not tractable.

2.2.1 X-Mutants and Y-Mutants

We say an equation is an X-mutant from a collection of bi-degree (a, b) equations, if all terms of bi-degree (a, i) have been eliminated. We can safely multiply an X-mutant by an X-variable and get a new equation of bi-degree (a, b) or lower. We define a Y-mutant similarly.

Can we find any X or Y mutants from bi-degree $(9, 8)$?

Let's first take all equations at bi-degrees $(9, i)$ and try to eliminate all terms of bi-degrees $(9, i)$ for $i \leq 8$. We have $-\overline{h}(9, 8; 68, 32; 64, 99) - \overline{h}(9, 7; 68, 32; 64, 99) = 56509572816576 \approx 2^{45.7}$. This means that if we take all equations at bi-degree $(9, 8)$ or lower and try to eliminate the bi-degree $(9, 8)$ and $(9, 7)$ terms, we find $56509572816576 \approx 2^{45.7}$ equations remaining, which is insufficient to eliminate all bi-degree $(9, 6)$ terms, since we still need $\overline{h}(9, 6; 68, 32; 64, 99) = 1729160624511936 \approx 2^{50.6}$ extra equations at bi-degree $(9, 6)$. So there should be no X-mutants. *If we multiply any mutant equation by an X-variable, we find bi-degree* $(10, 6)$ *or higher terms. XL2 therefore cannot continue this way without going again up in degree.*

As $\overline{h}(8,8;32,64;64,99) = 3415567098484728 \approx 2^{51.6} > -h(9,8;32,64;64,99)$, the mutants from the equations at bi-degree $(9,8)$ are not enough to eliminate all terms of bi-degree $(8,8)$, so similarly there should be no Y-mutants.

2.2.2 Criteria for Finding X- and Y-Mutants

Under what condition do we see an X-mutant at a bi-degree (a,b)?

This should happen when the number of mutants is sufficient to complete the elimination of all terms of bi-degree (a,i) for $i < b$ from the original set of equations at degree (a,b) or lower. One expects that this happens if the number of mutants is greater than the sum of residual degrees of freedom in the terms of bi-degree (a,i) in the equations of bi-degree (a,i), or when

$$-\overline{h}(a,b;n_x,n_y;m_x,m_{xy}) \geq \sum_{i=0}^{b-1} \overline{h}(a,i;n_x,n_y;m_x,m_{xy}).$$

We can re-write this as $0 \geq \sum_{i=0}^{b} \overline{h}(a,i;n_x,n_y;m_x,m_{xy})$, or,

$$\sum_{i=0}^{b}\left[[t^a s^i]\left(\frac{(1-t^2)^{m_x}(1-ts)^{m_{xy}}}{(1-t)^{n_x}(1-s)^{n_y}}\right)\right] = [t^a s^b]\left(\frac{(1-t^2)^{m_x}(1-ts)^{m_{xy}}}{(1-t)^{n_x}(1-s)^{n_y+1}}\right) \leq 0,$$

using generating function expressions, and similarly, for Y-mutants we have

$$[t^a s^b]\left(\frac{(1-t^2)^{m_x}(1-ts)^{m_{xy}}}{(1-t)^{n_x+1}(1-s)^{n_y}}\right) \leq 0.$$

We see that both these conditions hold for $(n_x,n_y;m_x,m_{xy}) = (68,32;64,99)$ and $(a,b) = (10,7)$ but not $(9,8)$. Therefore, *for the RBS system attacking Rainbow-I$''$a$(36,32,32)$, we believe that the possible Bipartite XL2 attack suggested by the NIST team doesn't happen at bi-degree* $(9,8)$. For bi-degree $(10,7)$, see the analysis below on the number of mutants needed.

2.3 The Task of Finding Enough Mutants

In the following we assume that the putative "First-Fall" Bipartite XL2 attack of the previous subsection somehow works as suggested by the NIST team. The same computations would hold for bi-degrees $(10,7)$ instead of $(9,8)$. In that case each large number mentioned below will be replaced by one of comparable magnitude.

2.3.1 An Initial Wiedemann Run

A (block) Wiedemann on the transpose of the Macaulay-like matrix for terms at bi-degree $(9,8)$ can reveal random nontrivial linear combinations from the $\overline{R}^{(9,8)} = 1428708670234491531600 \approx 2^{70}$ relations in the Macaulay-like matrix ($\overline{R}^{(a,b)} = \#$equations at exactly bi-degree (a,b)), up to $\overline{h}(9,8;68,32;64,99) \lesssim 2^{51.2}$ in all, such all bi-degree $(9,8)$ terms are eliminated.

At least $\overline{T}^{(9,8)} - \overline{h}(9,8;68,32;64,99) = 8767648531815533712 \lesssim 2^{63}$ random equations out of the $\overline{R}^{(9,8)}$ are necessary for running this (block) Wiedemann to find $-\overline{h}(9,8;68,32;64,99)$ mutants. (This number is not much bigger than $\overline{T}^{(9,8)}$)

In general, a block Wiedemann uses the same $3 \times (\#\text{columns}) \times (\#\text{terms})$ multiplications independent of the block width if it is small, but at some point the block Berlekamp-Massey will become painful, and we can estimate a practical cap on the width. Let us say that a computing node in a cluster running the block Wiedemann has 128GB $= 2^{40}$ bits of memory. If the block is wider than 2^{20}, the node cannot hold the matrix, which is dense.

Notice that $h(9,8;68,32;64,99)/-\overline{h}(9,8;68,32;64,99) \approx 4.5$, and that $h(9,9;68,32;64,99) = -14405095991599302 \approx -2^{54} < 0$. So we might envision that if we multiply the mutants by each of the 32 Y-variables, we can get a new nonredundant equation almost every time and complete the elimination the moment we have sufficiently many equations. However, this means we are going to need at least $h(9,8;68,32;64,99)/32 \approx 2^{48}$ mutants out of $-\overline{h}(9,8;68,32;64,99) \approx 2^{51.2}$ in our initial block Wiedemann runs.

Conclusion: To get all the nontrivial degree-16 reduced equations we need, we must run the block Wiedemann at least $2^{48}/2^{20} = 2^{28}$ times.

Even assuming we need only 2^{14} (square root of 2^{28}) block Wiedemanns, this will take $\geq 2^{14} \times 2^{138.5} > 2^{152}$ multiplications, wiping out all the advantages of having a smaller system to solve initially.

For reference, we give the corresponding numbers for bi-degree $(10,7)$ — note that for bi-degrees $(11,6)$ and $(12,5)$ the Bipartite XL works, but takes more time than bi-degree $(13,4)$.

$$\begin{aligned} h(10,7;68,32;64,99) &= 11622781589490522 \gtrsim 2^{51} > 0 \\ \overline{h}(10,7;68,32;64,99) &= -9349017047523648 \gtrsim -2^{53.1} < 0 \end{aligned}$$

2.3.2 If We don't use (Block) Wiedemann

A common argument that we see is to argue that we don't need to use (block) Wiedemann, instead using some dense form of Gaussian Elimination and further say that it runs in N^{ω} multiplications with $\omega \approx 2.37$ (where N is the number of variables). This is absurd because the correct asymptotic is $N^{\omega+o(1)}$. However, there is a proportional constant c_{ω} which is astronomical when we take $\omega \approx 2.37$, such that no one ever uses Coppersmith-Winograd or even more complex methods of matrix multiplication in practice. A practical asymptotic is $7N^{\log_2 7} \approx 7N^{2.8} \approx 2^{179}$ which again wipes out any gains of the putative Bipartite XL2 over Bipartite XL. Of course we note that even the absurdly overoptimistic $\left(\overline{T}^{(9,8)}\right)^{2.37} \approx 2^{149}$ is too large to show a profit.

2.4 After the Initial Elimination

The result of the initial elimination at bi-degree $(9,8)$ is a collection of first-fall results (mutants), which are *dense* equations at degree 16. Each such equation has $T^{(9,8)} - \overline{T}^{(9,8)} = 3641376753349367175 \gtrsim 2^{61}$.

At bi-degree $(9,8)$, the linear system still has $h(9,8;68,32;64,99) \approx 2^{48}$ missing equations. We need to generate approximately that many equations. Note that Bipartite XL would terminate at bi-degree $(10,8)$ or $(9,9)$. So if we take the mutants and multiply them by variables to generate new equations, we expect the process to end with a solution. Unfortunately, the collection of such mutant-generated equations has approximately $\left(T^{(9,8)} - \overline{T}^{(9,8)}\right) \times h(9,8;68,32;64,99) \approx 2^{115}$ terms, which rules out any sparse matrix methods. Multiplications we would need in the follow-up elimination would be at least (using the abovementioned wildly optimistic estimate) in the realm of $\left(\overline{T}^{(9,8)}\right)^{2.37} \approx 2^{149}$, which again clearly outnumbers what one would have paid for a straight Bipartite XL with block Wiedemann.

2.5 Rainbow $Ia(32,32,32)$

We will take as a further illustrative example the RBS system created when attacking Rainbow-$Ia(32,32,32;16)$. For this attack, we need to solve a system with 64 X-variables, 32 Y-variables, 64 bi-degree $(2,0)$-equations and 95 bi-degree $(1,1)$ equations. We have

$$\begin{aligned} h(14,2;64,32;64,95) &= 997579892898379 \approx 2^{50} > 0, \text{ but} \\ \overline{h}(14,2;64,32;64,95) &= -4222561761072 \approx -2^{42} < 0. \end{aligned}$$

Thus, the NIST team suggested that it might be possible to conduct an XL2-like attack starting at bi-degree $(14,2)$.

X- and Y-Mutants: We can first check that there are no X- and Y-mutants at bi-degree $(14, 2)$ by checking that

$$\begin{aligned}
[t^{14}s^2]\left(\frac{(1-t^2)^{64}(1-ts)^{95}}{(1-t)^{64}(1-s)^{33}}\right) &= 327013949718576 \approx 2^{48.2} > 0 \\
[t^{14}s^2]\left(\frac{(1-t^2)^{64}(1-ts)^{95}}{(1-t)^{65}(1-s)^{32}}\right) &= 493968247129017 \approx 2^{48.8} > 0
\end{aligned}$$

Because these numbers are large, we know that attempting to run XL2 at bi-degree $(14, 2)$ will generate lots of extra monomials.

Number of Mutants Needed: $h(14, 2; 64, 32; 64, 95)/(64 + 32) \approx 2^{43}$. The same for $(13, 3)$ would be $2^{41.4}$, both needing $> 2^{20}$ Wiedemann runs.

Resulting System after X- or Y-extension: The resulting system has $T^{(13,3)} \approx T^{(14,2)} \approx 2^{60}$ variables, and has at least 2^{48} dense equations.

So, there is no hope for Wiedemann and the overoptimistic Coppersmith-Winograd estimate gives 2^{142}, which is more than what the Wiedemann Bipartite XL attack at $(12, 4)$ will cost. Thus Bipartite XL2 doesn't work well.

2.6 Rainbow $III''c(76, 36, 36)$

We will take as a further illustrative example the RBS system created when attacking Rainbow-$III''c(76, 36, 36)$. For this attack, we need to solve a system with 112 X-variables, 36 Y-variables, 72 bi-degree $(2, 0)$-equations and 147 bi-degree $(1, 1)$ equations. We have

$$\begin{aligned}
h(14, 11; 112, 36; 72, 147) &= 1377541749643608527503980 \approx 2^{80} > 0, \text{ but} \\
\overline{h}(14, 11; 64, 32; 64, 95) &= -1038927173849655588825776 \approx -2^{76.5} < 0.
\end{aligned}$$

Thus, if we follow the suggestion of the NIST team, it might be possible to conduct an XL2-like attack starting at bi-degree $(14, 11)$.

X- and Y-Mutants: We can first check that there are no X- and Y-mutants at bi-degree $(14, 11)$ by checking that

$$\begin{aligned}
[t^{14}s^{11}]\left(\frac{(1-t^2)^{72}(1-ts)^{147}}{(1-t)^{112}(1-s)^{37}}\right) &= 602315797005838403236840 \approx 2^{79} > 0 \\
[t^{14}s^{11}]\left(\frac{(1-t^2)^{72}(1-ts)^{147}}{(1-t)^{113}(1-s)^{36}}\right) &= 260464711394354060327643 \approx 2^{77.8} > 0
\end{aligned}$$

Because these numbers are large, we know that attempting to run XL2 at bi-degree $(14, 11)$ will generate lots of extra monomials.

Number of Mutants Needed: $h(14, 11; 112, 36; 72, 147)/(148) \approx 2^{73}$. needing $> 2^{53}$ Wiedemann runs.

Resulting System after X- or Y-extension: The resulting system has $T^{(14,11)} \approx 2^{94}$ variables, and has at least 2^{80} dense equations.

Again, there is no hope for Wiedemann and the overoptimistic Coppersmith-Winograd estimate gives 2^{223}, more than what the Wiedemann Bipartite XL attack at $(18, 7)$ will cost. Thus Bipartite XL2 doesn't work well, and fundamentally this is again because elimination leads to dense equations.

2.7 Rainbow $III^{\dagger}c(68, 32, 48)$

We will take as a further illustrative example the RBS system created when attacking Rainbow-$III^{\dagger}c(68, 32, 48)$. For this attack, we need to solve a system with 100 X-variables, 48 Y-variables, 80 bi-degree $(2, 0)$-equations and 147 bi-degree $(1, 1)$ equations. We have

$$\begin{aligned} h(20, 5; 100, 48; 80, 147) &= 71345380836754462750007 85 \approx 2^{82.5} > 0, \text{ but} \\ \overline{h}(20, 5; 100, 48; 80, 147) &= -47924175520423534156608 0 \approx -2^{78.7} < 0. \end{aligned}$$

Thus, if we follow the suggestion of the NIST team, it might be possible to conduct an XL2-like attack starting at bi-degree $(20, 5)$.

X- and Y-Mutants: We can first check that there are no X- and Y-mutants at bi-degree $(20, 5)$ by checking that

$$\begin{aligned} [t^{20}s^{5}]\left(\frac{(1-t^2)^{80}(1-ts)^{147}}{(1-t)^{101}(1-s)^{48}}\right) &= 247337171181471742349286 0 \approx 2^{81} > 0 \\ [t^{20}s^{5}]\left(\frac{(1-t^2)^{80}(1-ts)^{147}}{(1-t)^{100}(1-s)^{49}}\right) &= 241615389570567391516504 5 \approx 2^{81} > 0 \end{aligned}$$

Because these numbers are large, we know that attempting to run XL2 at bi-degree $(20, 5)$ will generate lots of extra monomials.

Number of Mutants Needed: $h(20, 5; 100, 48; 80, 147)/(148) \approx 2^{75.4}$. needing $> 2^{55}$ Wiedemann runs.

Resulting System after X- or Y-extension: The resulting system has $T^{(20,5)} \approx 2^{96}$ variables, and has at least 2^{82} dense equations.

Again, there is no hope for Wiedemann and the overoptimistic Coppersmith-Winograd estimate gives $\approx 2^{227}$, more than what the Wiedemann Bipartite XL attack at $(14, 12)$ will cost. Thus Bipartite XL2 doesn't work well.

2.8 Rainbow $V''c(104, 48, 48)$

We will take as a further illustrative example the RBS system created when attacking Rainbow-$V''c(104, 48, 48)$. For this attack, we need to solve a system with 152 X-variables, 48 Y-variables, 96 bi-degree $(2, 0)$-equations and 199 bi-degree $(1, 1)$ equations. We have

$$\begin{aligned} h(19,13;152,48;96,199) &= 33283451421253255630824071181 2488 \approx 2^{108} > 0, \text{ but} \\ \overline{h}(19,13;152,48;96,199) &= -1600687433035468626342834076240 \approx -2^{100} < 0. \end{aligned}$$

Thus, if we follow the suggestion of the NIST team, it might be possible to conduct an XL2-like attack starting at bi-degree $(19, 13)$.

X- and Y-Mutants: We can first check that there are no X- and Y-mutants at bi-degree $(19, 13)$ by checking that

$$\begin{aligned} [t^{19}s^{13}]\left(\frac{(1-t^2)^{96}(1-ts)^{199}}{(1-t)^{153}(1-s)^{48}}\right) &= 8020012471958500358681256016 2920 \approx 2^{106} > 0 \\ [t^{19}s^{13}]\left(\frac{(1-t^2)^{96}(1-ts)^{199}}{(1-t)^{152}(1-s)^{49}}\right) &= 16193979646610821053769333211 9990 \approx 2^{107} > 0 \end{aligned}$$

Because these numbers are large, we know that attempting to run XL2 at bi-degree $(19, 13)$ will generate lots of extra monomials.

Number of Mutants Needed: $h(19, 13; 152, 48; 96, 199)/(200) \approx 2^{100.4}$. needing $> 2^{80}$ Wiedemann runs.

Resulting System after X- or Y-extension: The resulting system has $T^{(19,13)} \approx 2^{125}$ variables, and has at least 2^{108} dense equations.

There is no hope for Wiedemann and the overoptimistic Coppersmith-Winograd estimate gives $\approx 2^{297}$, more than what the Wiedemann Bipartite XL attack at $(19, 13)$ will cost. We thus see that Bipartite XL2 doesn't work well.

2.9 Rainbow $V^{\dagger}c(96, 36, 64)$

We will take as a further illustrative example the RBS system created when attacking Rainbow-$V^{\dagger}c(96, 36, 64)$. For this attack, we need to solve a system with 132 X-variables, 64 Y-variables, 100 bi-degree $(2, 0)$-equations, 195 bi-degree $(1, 1)$ equations. We have

$$\begin{aligned} h(18, 15; 132, 64; 100, 195) &= 2099369496623115234879186961755620 \approx 2^{110.7} > 0, \text{ but} \\ \overline{h}(18, 15; 132, 64; 100, 195) &= -371675427752064100013163901852480 \approx -2^{108} < 0. \end{aligned}$$

Thus, if we follow the NIST team suggestion, it might be possible to conduct an XL2-like attack starting at bi-degree $(18, 15)$.

X- and Y-Mutants: We can first check that there are no X- and Y-mutants at bi-degree $(18, 15)$ by checking that

$$\begin{aligned} [t^{18}s^{15}]\left(\frac{(1-t^2)^{100}(1-ts)^{195}}{(1-t)^{133}(1-s)^{64}}\right) &= 493658677133282151171730895365520 \approx 2^{108.6} > 0 \\ [t^{18}s^{15}]\left(\frac{(1-t^2)^{100}(1-ts)^{195}}{(1-t)^{132}(1-s)^{65}}\right) &= 536178001671049801324155671099180 \approx 2^{108.7} > 0 \end{aligned}$$

Because these numbers are large, we know that attempting to run XL2 at bi-degree $(18, 15)$ will generate lots of extra monomials.

Number of Mutants Needed: $h(18, 15; 132, 64; 100, 195)/296 \approx 2^{102.5}$. needing $> 2^{82.5}$ Wiedemann runs.

Resulting System after X- or Y-extension: The resulting system has $T^{(18,15)} \approx 2^{128}$ variables, and has at least 2^{110} dense equations.

There is no hope for Wiedemann and the overoptimistic Coppersmith-Winograd estimate gives $\approx 2^{304}$, more than what the Wiedemann Bipartite XL attack at $(19, 13)$ will cost. We thus see that Bipartite XL2 doesn't work well.

3 The new MinRank Attacks

Here we will recap briefly the new attack method in **(4)**.

Suppose we are given $k + 1$ matrices $Y, M_1, ..., M_k$ of size $m \times n$ and we want to find a linear combination

$$M = Y + \sum_{i=1}^{k} x_i M_i$$

of rank $\leq R$.

Kipnis and Shamir proposed a method to solve this problem by introducing a new matrix K of size $n \times r$, which is a basis of the kernel of M. This gives us the condition:

$$MK = 0.$$

Then we will try to solve this new equation but with much more additional variables from K.

The work of **(2)** claims that they proved that the best method is actually using the $(r+1) \times (r+1)$ minors of the matrix M to solve the system. But the work of **(1)** gives new analysis along the direction of the Kipnis-Shamir attack to show that the claim in **(2)** is wrong. The best attack of this new direction is that of **(4)**. Their idea has some new tools.

- The first is that they suggest that instead of the kernel, we use a new matrix C, which is a basis of the row space of M. Here C is an $r \times n$ matrix. Then we look at the new matrix $\binom{m_i}{C}$, where m_i is the i-th row of M and derive equations from the $(r+1) \times (r+1)$ minors in this new matrix.

- The second is that they suggest that we consider the $(r \times r)$ minors of C as new variables. Then we solve the so derived bi-partite equations in x_i and the minors of C using Wiedemann.

The combination of these two, in particular, the application of Wiedemann XL, changes the MinRank attack significantly.

Our new analysis derived 3 new insights which were not explained in **(4)**. we can prove the following:

- All the new equations can actually be derived directly from the KS equations.

- When $m = n = k + 1$, the new attack and KS attack are identical.

This, to us, shows that the new MinRank attack is nothing but a new efficient and very clever way of organizing the known equations through the formal minors instead of the new set of variables in the kernel (and/or row basis) of the desired MinRank matrix. We believe this is really the best we can do in terms of improving the MinRank attack from the theoretical perspective since minors are the only algebraic structure we can explore to speed up the computations.

Remark *In the originial document submitted to NIST on July 22, 2020, one of the statements here was incorrect. We stated that "The new equations also contain equations that can not be derived trivially from the KS equations without getting through finding mutants." We actually found out that J. Ding made a mistake in his toy-example programming test. What we can prove now is that actually all the new equations can be derived directly from the KS equations. We will put out soon a new paper to give the details.*

The Rainbow parameters proposed in the NIST submission had very high security estimates against the MinRank attack and therefore can withstand the new attack. But this attack makes us think further if our design is indeed the best with desired performance.

A key formula in the analysis of **(4)** is:

$$m\binom{n}{r+1} \geq (k+1)\binom{n}{r} - 1,$$

where the left side counts the number of equations and the right side counts the number of variables. If this inequality holds, we can solve the resulting system efficiently using a Wiedemann approach. For Rainbow we find that the inequality even holds if we don't consider all columns of the matrices $\binom{m_i}{C}$ (thus replacing n by some $n' < n$). This gives us an additional reduction of the complexity of the attack.
Since the resulting linear system is sufficiently sparse, we can solve it using the Wiedemann algorithm using

$$\mathcal{O}\left(\left((k+1)\cdot\binom{n'}{r}\right)^2 ((r+1)\cdot(k+1))\right).$$

operations.
Our analysis shows that the number k, which corresponds to o_2 in the Rainbow design, is extremely important in terms of the impact of the new attack. In our new parameter recommendations, we therefore choose $o_2 > o_1$. This increase of k will increase tremendously the attack complexity.

4 The Hybrid MinRank Attack

In this section we analyze the question if we can improve the attack by guessing some of the k variables x_i. When guessing a of the k variables, the complexity of the attack turns out to be

$$comp = q^a \cdot 3 \cdot \binom{n'(a)}{r}^2 (k-a+1) \cdot (r+1).$$

Here, the notation $n'(a)$ denotes that the optimal value n' depends on the number a of guessed variables.

We performed a number of experiments for the Rainbow instances I, III and V. We have

- I: $(q, v, o_1, o_2) = (16, 36, 32, 32) \Rightarrow r = v + o_1 = 68$, $k = o_2 = 32$
- III: $(q, v, o_1, o_2) = (256, 68, 32, 48) \Rightarrow r = v + o_1 = 100$, $k = o_2 = 48$
- V: $(q, v, o_1, o_2) = (256, 96, 36, 64) \Rightarrow r = v + o_1 = 132$, $k = o_2 = 64$

For each of the three parameter sets and $a \in \{0, \ldots, 9\}$, we computed the smallest value n' such that

$$n \cdot \binom{n'}{r+1} \geq (k-a+1) \cdot \binom{n'}{r} - 1$$

is fulfilled. After that, we computed the complexity of the attack as

$$comp = q^a \cdot 3 \cdot \binom{n'}{r}^2 \cdot ((r+1) \cdot (k-a+1)).$$

The result is shown by Tables 1, 2 and 3.
In the tables, we multiplied the above number of field multiplications with the factor

$$2 \cdot \log_2(q)^2 + \log_2 q.$$

As the tables show, the value n' reduces only very slowly when increasing the number a of guessed variables, due to which the complexity of the attack increases for higher a. Therefore, guessing variables seems not to be a good idea in the given scenario.

Furthermore, we computed the complexity when guessing all of the $k = o_2$ variables x_i. In this case we have to check in every iteration, if the rank of the resulting matrix indeed is $\leq r$. Therefore, the complexity of this attack is given as

$$comp = q^{o_2+1} \cdot r^{\omega}.$$

We find that the complexity of such an attack against the Rainbow instances I, III and V is 152, 415 and 544 bits respectively, which is well beyond the required security levels.

a	$k-a$	$n'(a)$	complexity (bit)
0	33	91	168.1
1	32	91	172.0
2	31	90	171.8
3	30	89	171.6
4	29	89	175.5
5	28	88	175.2
6	27	87	174.7
7	26	86	174.2
8	25	86	178.0
9	24	85	177.3

Table 2: Complexity of the Attack against the Rainbow instance I (depending on the number a of guessed variables)

a	$k-a$	$n'(a)$	complexity (bit)
0	49	134	242.3
1	48	133	246.2
2	47	133	254.1
3	46	132	258.0
4	45	131	261.8
5	44	131	269.7
6	43	130	273.5
7	42	129	277.1
8	41	128	280.7
9	40	128	288.6

Table 3: Complexity of the Attack against the Rainbow instance III (depending on the number a of guessed variables)

a	$k-a$	$n'(a)$	complexity (bit)
0	65	177	314.0
1	64	176	318.0
2	63	175	321.9
3	62	175	329.9
4	61	174	333.8
5	60	173	337.6
6	59	173	345.5
7	58	172	349.3
8	57	171	353.0
9	56	170	356.7

Table 4: Complexity of the Attack against the Rainbow instance V (depending on the number a of guessed variables)

5 Experiments with higher values of b

In many cases one gets better results when following a somewhat more general approach:

1. Use the technique described above to generate multivariate equations which are bilinear in the variables x_i and the $r \times r$ minors of the matrix C.

2. Multiply the equations found in 1) with monomials of degree $b-1$ in the variables λ_i. By doing so, one obtains equations of degree $b+1$ which are linear in the $r \times r$ minors of C and of degree b in the variables x_i.

The number of monomials in the so obtained equations is

$$\binom{k+b}{b} \cdot \binom{n}{r}.$$

The number of (linearly independent) equations is

$$\sum_{i=1}^{b} (-1)^{i+1} \binom{n}{r+i} \cdot \binom{m+i-1}{i} \cdot \binom{k+b-i}{b-i}.$$

Therefore, we can solve the system by linearization if and only if

$$\binom{k+b}{b} \cdot \binom{n}{r} - 1 \leq \sum_{i=1}^{b} (-1)^{i+1} \binom{n}{r+i} \cdot \binom{m+i-1}{i} \cdot \binom{k+b-i}{b-i}$$

holds.

The resulting complexity is then

$$\left(\binom{k+b}{b}\cdot\binom{n'}{r}\right)^2\cdot((r+1)\cdot(k+1))$$

field operations.

In the following we determine the optimal value n' (number of columns of C needed) such that the above equation is fulfilled and the complexity of the attack. For the complexity, we multiply the number of field multiplications with the factor

$$2\cdot\log_2(q^2)+\log_2 q.$$

We do this both for the parameter sets used in the second submission (Ia, IIIc, and Vc) and for the new parameter sets (I, III and V).

- parameter set Ia $(q,v,o_1,o_2)=(16,32,32,32)\Rightarrow m=96,\ k=33,\ r=64$

b	1	2	3	4
n'	87	84	82	81
complexity	167	164	163	165

- parameter set I $(q,v,o_1,o_2)=(16,36,32,32)\Rightarrow m=100,\ k=33,\ r=68$

b	1	2	3	4
n'	91	89	87	86
complexity	171	171	170	172

- parameter set IIIc $(q,v,o_1,o_2)=(256,68,36,36)\Rightarrow m=140,\ k=37,\ r=104$

b	1	2	3	4
n'	132	130	129	127
complexity	228	227	230	227

- parameter set III $(q,v,o_1,o_2)=(256,68,32,48)\Rightarrow m=148,\ k=37,\ r=100$

b	1	2	3	4	5	6
n'	134	131	128	127	125	124
complexity	251	248	244	247	244	246

- parameter set Vc $(q, v, o_1, o_2) = (256, 92, 48, 48) \Rightarrow m = 188,\ k = 49,\ r = 140$

b	1	2	3	4
n'	177	174	172	171
complexity	294	290	288	291

- parameter set V $(q, v, o_1, o_2) = (256, 96, 36, 64) \Rightarrow m = 196,\ k = 65,\ r = 132$

b	1	2	3	4	5	6
n'	177	173	170	168	166	165
complexity	322	317	313	312	311	314

5.1 New Quantum attacks

As for the quantum attack along the line of RBS attack, one way is still to do Grovers algorithm search, since the numbers of variables becomes much bigger than the direct attack, the cost will be much higher than the direct attack. The same can be applied to the New MinRank attack.

Recently, there are some works by Gao etc on using HHL to speed up the XL algorithms. However the works of Ding, Gheorghiu and Gilyen

`https://simons.berkeley.edu/talks/overview-attacks-elliptic-curve-isogenies-based-systems`

`https://simons.berkeley.edu/talks/overview-quantum-algorithmic-tools`

show that it is not efficient at all due to the large condition number in the system.

In order to use the method in Gao etc for the RBS attack or the New MinRank attack, one must first covert the system into a system of GF(2), then convert the equation into equations over real numbers. This first will make the systems much larger with much more variables. In the case of RBS attack, the bi-degree argument will fail due to the new variables and the solving degree will become much higher. In the case of the new MInRank attack, the number of new variables need to be added is even much larger as the original system, which will again make the original argument of using

XL linear fail. From what we can see now, using the argument by Ding, Gheorghiu and Gilyen etc, we can show that this again will have much higher cost that the original direct quantum attack. The detailed analysis requires much more space, which we will address in a separate paper.

6 Parameters and Performance

Based on our observations from the previous sections, we propose the following three parameter sets for Rainbow.

- $(GF(16), 36, 32, 32)$ for security level I,
- $(GF(256), 68, 32, 48)$ for security level III and
- $(GF(256), 96, 36, 64)$ for security level V.

The complexity of known attacks against these Rainbow instances can be computed as shown below. Note that the behavior of the RBS attack against the parameter sets for level III and V was already analyzed in Sections 2.7 and 2.9. Furthermore note that the complexity estimates of all known attacks are well beyond the NIST security requirements. In this sense, the modified parameters are chosen in a much more conservative way than the 2nd round parameters.

- Level 1: (GF(16),36,32,32) Requirement 143 / 74 (classic / quantum)
  Direct attack: 164 / 122
  UOV attack: 157 / 91
  HighRank: 150 / 86
  New MinRank attack: 162
  New RBS attack: 147
- Level III: (GF(256),68,32,48) Requirement 207 / 137
  Direct Attack: 234 /200
  UOV attack: 437 / 233
  High Rank: 410 / 218
  New MinRank attack: 228
  New RBS attack: 217

Level	parameters	public key size (kB)	private key size (kB)	signature size (bit)
I	(GF(16),36,32,32)	157.8	101.2	528
III	(GF(256),68,32,48)	861.4	611.3	1,312
V	(GF(256),96,36,64)	1,885.4	1,375.7	1,632

Table 5: Key and Signature Sizes for Standard Rainbow. The private key can be generated from a small seed.

Level	parameters	public key size (kB)	private key size (kB)	signature size (bit)
I	(GF(16),36,32,32)	58.8	101.2 (99.0)	528
III	(GF(256),68,32,48)	258.4	611.3 (603.0)	1,312
V	(GF(256),96,36,64)	523.5	1,375.7 (1,361.8)	1,696

Table 6: Key and Signature Sizes for Cyclic Rainbow. The numbers in brackets give the private key size if the linear maps $\mathcal{S}$ and $\mathcal{T}$ are generated from a 256 bit seed

- Level V: (GF(256), 96,36,64) Requirement 272 / 202

  Direct Attack: 285 / 243

  UOV attack: 567 /299

  High Rank: 539 / 283

  New MinRank attack: 296

  New RBS attack: 281

Key and signature sizes of the modified Rainbow instances are shown in Tables 2 and 3.

It is clear from the performance perspective that these new changes do not have significant impact. For the first parameter set, the only difference is the slightly higher number of vinegar variables, which leads to a small increase of the signing time. The cost of key generation and signature verification basically stays the same. Since we have an extremely low signing cost, the small slow down does not really matter much.

For our new parameters for Level III and V, the number of equations in the system is slightly increased, which leads to a moderate slow down of the key generation, signing and verification algorithms.