Breaking Rainbow Takes a Weekend on a Laptop

Ward Beullens

IBM Research, Zurich, Switzerland
wbe@zurich.ibm.com

Abstract. This work introduces new key recovery attacks against the Rainbow signature scheme, which is one of the three finalist signature schemes still in the NIST Post-Quantum Cryptography standardization project. The new attacks outperform previously known attacks for all the parameter sets submitted to NIST and make a key-recovery practical for the SL 1 parameters. Concretely, given a Rainbow public key for the SL 1 parameters of the second-round submission, our attack returns the corresponding secret key after on average 53 hours (one weekend) of computation time on a standard laptop.

1 Introduction

The Rainbow signature scheme [8], proposed by Ding and Schmidt in 2005, is one of the oldest and most studied signature schemes in multivariate cryptography. Rainbow is based on the (unbalanced) Oil and Vinegar signature scheme [16, 11], which, for properly chosen parameters, has withstood all cryptanalysis since 1999. In the last decade, there has been a renewed interest in multivariate cryptography, because it is believed to resist attacks from quantum adversaries. The goal of this paper is to improve the cryptanalysis of Rainbow, which is an important objective because Rainbow is currently one of three finalist signature schemes in the NIST Post-Quantum Cryptography standardization project.

Related Work. The cryptanalysis of Rainbow and its predecessors was an active area of research for some years in the early 2000s. Attacks from this era include the MinRank attack, HighRank attack, the Billet-Gilbert attack, UOV reconciliation attack, and the Rainbow Band Separation Attack [12, 18, 5, 10, 9]. After 2008 the cryptanalysis seemed to have stabilized, until the participation of Rainbow in the NIST PQC project motivated more cryptanalysis. During the second round of the NIST project, Bardet *et al.* proposed a new algorithm for solving the MinRank problem [3]. This drastically improved the efficiency of the

* Ward Beullens holds Junior Post-Doctoral fellowship 1S95620N from the Research Foundation Flanders (FWO).

MinRank attack, although not enough to threaten the parameters submitted to NIST. A more memory-friendly version of this algorithm was proposed by Baena *et al.* [2]. Perlner and Smith-Tone tightened the analysis of the Rainbow Band Separation attack, showing that the attack was more efficient than previously assumed [17]. This prompted the Rainbow team to increase the parameters slightly for the third round. During the third round, Beullens introduced new attacks [4] which reduced the security level of Rainbow by a factor of 2^{20} for the SL 1 parameters. The Rainbow team argued that despite the new attacks, the Rainbow parameters still meet the NIST requirements [1].

Contributions. This paper introduces two new key-recovery attacks. Recall that if $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ is a Rainbow public key, then the corresponding secret key contains, among some other information, a subspace $O_2 \subset \mathbb{F}_q^n$, such that $\mathcal{P}(O_2) = 0$.

Our attacks are based on the simple observation that for a randomly chosen $\mathbf{x} \in \mathbb{F}_q^n$, the differential

$$D_{\mathbf{x}} : \mathbb{F}_q^n \to \mathbb{F}_q^m : \mathbf{y} \mapsto \mathcal{P}(\mathbf{x}+\mathbf{y}) - \mathcal{P}(\mathbf{x}) - \mathcal{P}(\mathbf{y})$$

(which is a linear map) has a kernel vector in O_2 with probability $\approx 1/q$. Given this observation, we first propose the following simple attack: Guess a vector $\mathbf{x}$, and try to solve for a vector $\mathbf{o}$ such that

$$\begin{cases} D_{\mathbf{x}}\mathbf{o} = 0 \\ \mathcal{P}(\mathbf{o}) = 0 \end{cases} .$$

If we find such a solution $\mathbf{o}$, then with high probability we found a vector in O_2, after which a full key recovery is easy. Otherwise, if no solution exists, we try again with a different guess for $\mathbf{x}$. In fields of odd characteristic, we find that the quadratic systems that arise in the attack behave exactly like random systems. In fields of characteristic 2 (which includes all the parameters submitted to NIST in the second and third rounds), the systems have some structure that can be exploited to solve the systems slightly more efficiently. This simple attack is efficient enough to do a key recovery attack in practice for the SL1 parameter set from the second-round submission to the NIST PQC project. For a single guess of $\mathbf{x}$, it takes only 3 hours and 32 minutes to solve the resulting system, and a guess is good with a probability of approximately 1/15.06, so on average, a full attack takes $15.06 \cdot 3.53 \approx 53$ hours. We estimate that a key recovery for the SL 1 parameter set of the third-round submission requires only a factor 2^8 more effort (see Table 1).

For the parameter sets targeting NIST security levels 3 and 5, we find that the attack can be improved by combining the new technique with the rectangular MinRank attack of Beullens [4]. The combined attack chooses a random $\mathbf{x}$ and essentialy restricts $\mathcal{P}$ to the kernel of $D_{\mathbf{x}}$ and runs the rectangular MinRank on this smaller system, which will succeed with a probability of approximately $1/q$. Estimates of the complexities of the simple and combined attacks against the Rainbow parameter sets submitted to NIST are given in Table 1.

Table 1. An overview of the cost of our attack versus known attacks for the six Rainbow parameter sets submitted to the second round and the finals of the NIST PQC standardization project. Complexities are given as $\log_2$ of the estimated gate count. The complexities of the known attacks are taken from [4].

Parameter set		(q, n, m, o_2)	Simple attack	Combined attack	Known attacks
Second round	SL I	$(16, 96, 64, 32)$	<u>61</u>	93	123
	SL III	$(256, 140, 72, 36)$	186	<u>131</u>	151
	SL V	$(256, 188, 96, 48)$	246	<u>164</u>	191
Finals	SL I	$(16, 100, 64, 32)$	<u>69</u>	99	127
	SL III	$(256, 148, 80, 48)$	160	<u>157</u>	177
	SL V	$(256, 196, 100, 64)$	257	<u>206</u>	226

2 Preliminaries

Notation. Let $\mathbb{F}_q$ be the finite field with q elements, and let $\mathcal{P} = \{p_i\}_{i=1}^m$ be a sequence of m multivariate quadratic polynomials in n variables over $\mathbb{F}_q$. We identify $\mathcal{P}$ with the function $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ defined as $\mathcal{P}(\mathbf{x}) = \{p_i(\mathbf{x})\}_{i=1}^m$. We define the differential $\mathcal{P}'(\mathbf{x}, \mathbf{y})$ (sometimes called polar the form of $\mathcal{P}$) as $\mathcal{P}'(\mathbf{x}, \mathbf{y}) := \mathcal{P}(\mathbf{x} + \mathbf{y}) - \mathcal{P}(\mathbf{x}) - \mathcal{P}(\mathbf{y}) + \mathcal{P}(0)$. It is easily checked that $\mathcal{P}'(\mathbf{x}, \mathbf{y})$ is symmetric and bilinear.

Solving multivariate systems. Our attacks use (in a black-box way) a subroutine that given a homogeneous multivariate quadratic map $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$, finds a non-zero solution $\mathbf{x}$ such that $\mathcal{P}(\mathbf{x}) = 0$, if such a solution exists. We instantiate this subroutine with the block Wiedemann XL algorithm [14, 7, 15, 6]. This algorithm constructs a large but very sparse system of linear equations and solves it with the block Wiedemann algorithm to take advantage of the sparsity. For the experimental validation of our attacks we used the optimized implementation of Block Wiedemann XL by Cheng, Chou, Niederhagen, and Yang [6]. The gate complexity of this algorithm on an instance with m random equations in n variables can be estimated as the cost of

$$3\binom{n-1+D}{D}^2\binom{n+1}{2}$$

field multiplications, where D is the *operating degree* of XL, which is chosen to be the smallest integer such that the coefficient of the t^D term in the power series expansion of

$$\frac{(1-t^2)^m}{(1-t)^n}$$

is non-positive.

Example 1. Suppose we want to find a solution to a system of 63 homogeneous quadratic equations in 31 variables. We have

$$\frac{(1-t^2)^{63}}{(1-t)^{31}} = 1 + 31t + 433t^2 + 3503t^3 + 17081t^4 + 41447t^5 - 44919t^6 + O(t^7)\,,$$

so we can run XL at degree $D = 6$, with an estimated cost of

$$3\binom{31-1+6}{6}^2\binom{31+1}{2} \approx 2^{52.3}$$

field multiplications.

Solving MinRank problems. Our attacks will also make use of an algorithm to solve the MinRank problem. An instance of this problem is a list of matrices $L_1, \ldots, L_k \in \mathbb{F}_q^{n\times m}$, and a target rank r. And the task is to find a non-zero linear combination of the matrices whose rank is at most r. This NP-hard problem often appears in the cryptanalysis of multivariate and rank metric code-based cryptosystems [13, 9], and has therefore been studied relatively well.

Our attacks use the support-minors algorithm of Bardet, Bros, Cabarcas, Gaborit, Perlner, Smith-Tone, Tillich, and Verbel [3]. This algorithm translates the rank condition to a large sparse system of bilinear equations and solves this system using linearization and sparse linear algebra methods. The complexity of this algorithm can be estimated as

$$3(k-1)(r+1)\binom{m}{r}^2\binom{k+b-2}{b}^2\,,$$

where b is the operating degree of the algorithm, which is chosen to be the smallest positive integer such that

$$\binom{m}{r}\binom{K+b-2}{b} - 1 \leq \sum_{i=1}^{b}(-1)^{i+1}\binom{m}{r+i}\binom{n+i-1}{i}\binom{K+b-i-2}{b-i}\,. \quad (1)$$

It is sometimes beneficial to ignore some columns of the L_i matrices; one can choose to truncate the L_i matrices to their first $m' \leq m$ columns, for some optimal value of m' in the range $[r+1, m]$. It might seem wasteful to not use all the columns, but current MinRank algorithms can unfortunately not always use all the columns efficiently. (Similar to how LWE solving algorithms often cannot make good use of all their LWE samples.)

Example 2. Suppose we are given $k = 92$ matrices with $n = 187$ rows and $m = 96$ columns each, and we know there is a non-zero linear combination of the matrices with rank $r = 48$, which we want to find. Plugging our parameters into inequality (1), we find that can work at degree $b = 1$ as longs as we keep at least 72 columns, we can work at $b = 2$ is we keep at least 68 columns, at $b = 3$ if we keep 65 columns and at $b = 4$ if we keep 63 columns etc. It turns out that we get the most efficient algorithm if we keep $m' = 65$ columns and work at degree $b = 3$. The estimated cost of the algorithm is then

$$3(92-1)(48+1)\binom{65}{48}^2\binom{92+3-2}{3}^2 \approx 2^{149.1}$$

field multiplications.

The Rainbow trapdoor. We present the Rainbow trapdoor as described by Beullens [4]. A Rainbow instance is parameterized by four parameters:

- q, the size of the finite field,
- n, the number of variables,
- m, the number of equations in the public key, and
- o_2, the dimension of the subspaces $O_2 \subset \mathbb{F}_q^n$ and $W \subset \mathbb{F}_q^m$.

The public key is then a multivariate quadratic map $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$, and the secret key consists of three linear subspaces O_1, O_2, W, such that (see Figure 1):

1. $O_2 \subset O_1 \subset \mathbb{F}_q^n$, and $W \subset \mathbb{F}_q^m$,
2. $\dim(O_2) = \dim(W) = o_2$, and $\dim(O_1) = m$,
3. for all $\mathbf{o}_2 \in O_2$ and $\mathbf{x} \in \mathbb{F}_q^n$ we have $\mathcal{P}(\mathbf{o}_2) = 0$ and $\mathcal{P}'(\mathbf{x}, \mathbf{o}_2) \in W$, and
4. for all $\mathbf{o}_1 \in O_1$, we have $\mathcal{P}(\mathbf{o}_1) \in W$.

The key generation algorithm chooses the subspaces $O_2 \subset O_1 \subset \mathbb{F}_q^n$ and $W \subset \mathbb{F}_q^m$ of the correct dimension, and produces a public key $\mathcal{P}$ that is distributed uniformly among all the $\mathcal{P}$ that behave properly on O_2, O_1, W. How to do key generation efficiently, and how to use the trapdoor structure to sample preimages for $\mathcal{P}$ is irrelevant for our attacks, so we refer to [4] for the details.

3 Simple Attack

Let $(\mathsf{pk} = \mathcal{P}, \mathsf{sk} = (O_2, O_1, W))$ be a Rainbow key pair. For any vector $\mathbf{x} \in \mathbb{F}_q^n$, and any vecor $\mathbf{o}_2 \in O_2$, we have by construction (see Section 2) that $\mathcal{P}'(\mathbf{x}, \mathbf{o}_2) \in W$. So for any $\mathbf{x}$ we can consider the differential

$$D_{\mathbf{x}} : \mathbb{F}_q^n \to \mathbb{F}_q^m : \mathbf{y} \mapsto \mathcal{P}'(\mathbf{x}, \mathbf{y}) \,,$$

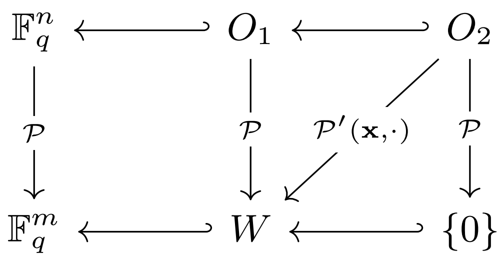

Fig. 1. The structure of a Rainbow public key. The differential $\mathcal{P}'(\mathbf{x},\cdot)$ maps O_2 to W for every $\mathbf{x} \in \mathbb{F}_q^n$.

which is a linear map from $\mathbb{F}_q^n$ to $\mathbb{F}_q^m$, that moreover sends O_2 to W. For any fixed non-zero $\mathbf{x}$ the differential $D_{\mathbf{x}}|_{O_2}$ restricted to O_2 is a uniformly random linear map from O_2 to W (over the random bits of the key generation algorithm). Note that $\dim(O_2) = \dim(W) = o_2$, so the probability that $D_{\mathbf{x}}$ has a kernel vector in O_2 is exactly the probability that a random o_2-by-o_2 matrix over $\mathbb{F}_q$ is singular. A matrix is non-singular if the first row is non-zero, and for each $i < o_2$, the $i+1$-th row is not in the span of the first i rows (which happens with probability q^{i-1-o_2}), so the probability of being singular is

$$1 - \prod_{i=0}^{o_2-1} \left(1 - q^{i-o_2}\right) ,$$

which is close to $1/q$ for sufficiently large q, regardless of o_2. For example, with $q = 16, o_2 = 32$, the probability is approximately $1/15.06$.

Our attack is now to simply pick a random (non-zero) $\mathbf{x}$, hope that the kernel of $D_{\mathbf{x}}$ intersects O_2 non-trivially, and then try to solve for a vector $\mathbf{o}$ in this intersection. Since $\mathcal{P}(\mathbf{o}) = 0$ for all $\mathbf{o} \in O_2$, we propose to do this by solving the following system

$$\begin{cases} D_{\mathbf{x}}\mathbf{o} = 0 \\ \mathcal{P}(\mathbf{o}) = 0 \end{cases}$$

This is a system of m homogeneous linear equations, and m homogeneous quadratic quations in the n variables of $\mathbf{o}$. If we use the m linear equations to eliminate m of the variables from the quadratic equations, we end up with a system of m homogeneous equations in $n-m$ variables. Concretely, let $B \in F_q^{n\times(n-m)}$ be a matrix whose columns form a basis for $\ker(D_{\mathbf{x}})$, then we are looking for a solution $\mathbf{x} \in \mathbb{F}_q^{n-m}$ to $\tilde{\mathcal{P}}(\mathbf{x}) = 0$, where $\tilde{\mathcal{P}}(\mathbf{x}) := \mathcal{P}(B\mathbf{x})$.

Attack in fields of odd characteristic. Our experiments (see Appendix A) show that when q is odd, $\tilde{\mathcal{P}}$ behaves like a random system of m homogeneous quadratic equation in $n-m$ variables in the XL algorithm. The ranks of the XL systems exactly match the ranks of XL systems of systems or random quadratic equations at each operation degree D. In particular, if a solution to $\mathcal{P}(\mathbf{x}) = 0$ exists we can find it with an estimated gate cost of

$$3\binom{n-m-1+D}{D}^2\binom{n-m+1}{2}$$

field multiplications, where D is the smallest positive integer such that the t^D coefficient of the power series expansion of $(1-t^2)^m/(1-t)^{m-n}$ (see Section 2.)

Attack in fields of even characteristic. Our experiments show that for even q, the rank of the XL systems does not match that of random systems, and just applying the XL as in the case of odd characteristic sometimes fails. The reason is that $\mathcal{P}'(\mathbf{x},\mathbf{x}) = 2\mathcal{P}(\mathbf{x})$ vanishes in characteristic 2, so $\mathbf{x} \in \ker(D_{\mathbf{x}})$. This means there is a $\tilde{\mathbf{x}} \in \mathbb{F}_q^{n-m}$ (known to the attacker) such that $\tilde{\mathcal{P}}(\tilde{\mathbf{x}}+\mathbf{y}) = \tilde{\mathcal{P}}(\tilde{\mathbf{x}}) + \tilde{\mathcal{P}}(\mathbf{y})$ for all $\mathbf{y} \in \mathbb{F}_q^{n-m}$, which is not something that usually happens for random $\tilde{\mathcal{P}}$.

Luckily for us, this is not a problem for the attack, in fact we can even exploit this property to make the attack slightly more efficient: We want to find $\mathbf{x}$ such that $\tilde{\mathcal{P}}(\mathbf{x}) = 0$. Let $Y \subset \mathbb{F}_q^{n-m}$ be any subspace of dimension $n-m-1$ that does not contain $\tilde{\mathbf{x}}$, such that $\langle \tilde{\mathbf{x}} \rangle + Y = \mathbb{F}_q^{n-m}$. Then it suffices to find $\mathbf{y} \in Y$ such that $\tilde{\mathcal{P}}(\mathbf{y}) = \alpha\tilde{\mathcal{P}}(\tilde{\mathbf{x}})$ for some $\alpha \in \mathbb{F}_q$, because then $\mathbf{x} = \tilde{\mathbf{x}} + \alpha^{-1/2}\mathbf{y}$ is a solution to $\tilde{\mathcal{P}}(\mathbf{x}) = 0$, (recall that evey element has a square root in fields of characteristic 2, so $\alpha^{-1/2}$ exists), because

$$\tilde{\mathcal{P}}(\tilde{\mathbf{x}} + \alpha^{-1/2}\mathbf{y}) = \tilde{\mathcal{P}}(\tilde{\mathbf{x}}) + \alpha^{-1}\tilde{\mathcal{P}}(\mathbf{y}) = 0\,.$$

To find this $\mathbf{y} \in Y$, we restrict $\tilde{\mathcal{P}}$ to Y, and look for a solution to the $m-1$ homogeneous quadratic equations

$$\hat{\mathcal{P}} := \{\tilde{p}_1 a_i - \tilde{p}_i a_1\}_{i=2}^m\ ,$$

where $\mathbf{a} = \tilde{\mathcal{P}}(\tilde{\mathbf{x}})$, and we assume with loss of generality that $a_1 \neq 0$.

By restricting to Y, we remove the problematic vector $\tilde{\mathbf{x}}$, so it should not be a surprise that our rank experiments show that the new system $\hat{\mathcal{P}}$ behaves like a system of $m-1$ random homogeneous quadratic equations in $n-m-1$ variables (see the rank experiments in Appendix A). Therefore, if a solution exists, we can find it with an estimated cost of

$$3\binom{n-m-2+D}{D}^2\binom{n-m}{2}$$

field multiplications, where D is the smallest positive integer such that the t^D coefficient of the power series expansion of $(1-t^2)^{m-1}/(1-t)^{m-n-1}$.

Example 3. The SL1 parameter set of the second-round NIST submission is $q = 16, n = 96, m = 64, o_2 = 32$. If we apply our attack to this parameter set we need to solve systems of $m - 1 = 63$ homogeneous quadratic equations in $n - m - 1 = 31$ variables, so the estimated cost of solving each system is $2^{52.3}$ multiplications (see Example 1). On average we need to try 15.06 systems. If the cost of one $\mathbb{F}_{16}$-multiplication is 36 gates, then we can estimate that the total average gate cost of the attack is $2^{52.3} \cdot 15.06 \cdot 36 \approx 2^{61.4}$, as reported in Table 1.

Completing the attack. It is well known that once a vector in O_2 is found, a full secret key (O_2, O_1, W) can be recovered very efficiently. See e.g., Section 5.3 of [4]. Given a single vector $\mathbf{o} \in O_2$, one can first compute

$$\langle \mathcal{P}'(\mathbf{o}, \mathbf{e}_1), \ldots, \mathcal{P}'(\mathbf{o}, \mathbf{e}_n) \rangle \subset W \,,$$

which will with overwhelming probability be an equality. Let ϕ_W be the quotient map $\mathbb{F}_q^m \to \mathbb{F}_a^m / W$. Then O_2 is found as the kernel of the linear map

$$\mathbf{o} \mapsto \begin{pmatrix} \phi_W(\mathcal{P}'(\mathbf{e}_1, \mathbf{o})) \\ \cdots \\ \phi_W(\mathcal{P}'(\mathbf{e}_n, \mathbf{e})) \end{pmatrix} .$$

This kernel contains O_2, and with overwhelming probability, the kernel is exactly equal to O_2. Finally, to compute O_1, note that $\phi_W \circ \mathcal{P}$ is a multivariate quadratic map in n variables that vanishes on O_1 (which has dimension m). Since for Rainbow parameters we have $n < 2m$ we can use the Kipnis-Shamir attack [12] to recover O_1 in polynomial time.

4 Combination with rectangular MinRank attack

Even though the simple attack from the previous section is very efficient for the NIST SL 1 parameter sets of Rainbow (because $n - m$ is small), we see in Table 1 that for the SL 3 and SL 5 parameter sets, the new attack does not always outperform the rectangular MinRank attack of Beullens [4]. In this section, we first summarize how the rectangular MinRank attack works, and then we show that it can be made more efficient by combining it with our "guess-$D_{\mathbf{x}}$" technique.

Rectangular MinRank attack. Let $\mathbf{e}_1, \cdots, \mathbf{e}_n$ be a basis for $\mathbb{F}_q^n$, and let $\mathcal{P}$ be a Rainbow public key. Then we define n rectangular matrices $L_i \in \mathbb{F}_q^{n \times m}$ as

$$L_i := \begin{pmatrix} \mathcal{P}'(\mathbf{e}_1, \mathbf{e}_i) \\ \cdots \\ \mathcal{P}'(\mathbf{e}_n, \mathbf{e}_i) \end{pmatrix} \,,$$

for all i from 1 to n. Let $\mathbf{o} \in \mathbb{F}_q^n$ be a vector, then since $\mathcal{P}'$ is bilinear, we have that

$$\sum_{i=1}^{n} o_i L_i = \begin{pmatrix} \mathcal{P}'(\mathbf{e}_1, \mathbf{o}) \\ \dots \\ \mathcal{P}'(\mathbf{e}_n, \mathbf{o})\,, \end{pmatrix}$$

which has rank at most $\dim(W) = o_2$ if $\mathbf{o} \in O_2$, because all the rows of the matrix are in W.

We have n public matrices of dimensions n-by-m, and we know there exist linear combinations of these matrices that have exceptionally low rank $\leq \dim(W)$, so we have an instance of the MinRank problem. We can now use generic MinRank solvers, such as the algorithms by Bardet *et al.* [3], to find a linear combination $\mathbf{o} \in \mathbb{F}_q^n$, such that $\sum o_i L_i$ has rank at most o_2. If $\mathbf{o}$ is such a solution, then with overwhelming probability $\mathbf{o} \in O_2$.

Note that every $\mathbf{o} \in O_2$ is a solution to the MinRank problem. Therefore, we can discard $o_2 - 1$ of the matrices, and the span of the remaining $n - o_2 + 1$ matrices will still contain a non-zero matrix of low rank. This is useful because reducing the number of matrices in the MinRank problem reduces the cost of finding a solution.

Once a solution $\mathbf{o} \in O_2$ is found, a full key recovery can be done efficiently, as explained at the end of section Section 3.

Remark 4. We have extra information about the solution $\mathbf{o}$ the MinRank problem, namely that $\mathcal{P}(\mathbf{o}) = o$. Beullens [4] shows that the MinRank solving algorithm of Bardet *et al.* [3] can be adapted to take advantage of this extra information. This reduces the cost of the attack by a small factor between 2^2 and 2^9 for the Rainbow parameters submitted to NIST.

Combined attack. The combined attack is straightforward. We choose a random $\mathbf{x} \in \mathbb{F}_q^n$, and then we solve for a vector $\mathbf{o} \in \ker(D_\mathbf{x})$, such that $\sum o_i L_i$ has rank at most o_2. We can use the $D_\mathbf{x}\mathbf{o} = 0$ equations to reduce the number of matrices in the MinRank problem by m. Concretely, let $\mathbf{b}_1, \dots, \mathbf{b}_{n-m}$ be a basis for $\ker(D_\mathbf{x})$, then we consider the $n - m$ matrices

$$\tilde{L}_i := \sum_{j=1}^{n} b_{ij} L_j = \begin{pmatrix} \mathcal{P}'(\mathbf{e}_1, \mathbf{b}_i) \\ \dots \\ \mathcal{P}'(\mathbf{e}_n, \mathbf{b}_i) \end{pmatrix} ,$$

for all i from 1 to $n - m$. Now $\mathbf{o} = \sum x_i \mathbf{b}_i \in \ker(D_\mathbf{x})$ is a solution to the original MinRank problem if and only if $\mathbf{x}$ is a solution to the new MinRank problem with $n - m$ matrices $\tilde{L}_1, \dots, \tilde{L}_{n-m}$.

The advantage of this approach is that we now have a MinRank problem with only $n - m$ matrices, which makes finding the solution much easier compared to the original rectangular MinRank attack, where we had $n - o_2 + 1$ matrices. This comes at the cost of having to repeat the attack on average approximately q times, until $\ker(D_{\mathbf{x}}) \cap O_2 \neq \{0\}$.

Experiments (see Appendix A) reveal that the MinRank instance $\tilde{L}_1, \ldots, \tilde{L}_{n-m}$ does not behave like a random MinRank instance. Upon inspection we see that this is because for all $\tilde{L}_i$, we have

$$\mathbf{x}\tilde{L}_i = \mathcal{P}'(\mathbf{x}, \mathbf{b}_i) = D_{\mathbf{x}}\mathbf{b}_i = 0.$$

That is, there is a common linear dependency shared by all the $\tilde{L}_i$ matrices. This means that one of the rows is not contributing any information to the MinRank problem. For example, if $x_1 \neq 0$, then the first row of $\sum o_i \tilde{L}_i$ is just a linear combination of the other rows, which means we can safely delete this first row without affecting the rank of $\sum o_i \tilde{L}_i$. After deleting a row from the $\tilde{L}_i$ we get a MinRank problem with $n - m$ matrices of size $(n-1)$-by-m, and for which there exists a solution of rank o_2 if the guess of $D_{\mathbf{x}}$ was good. Our rank experiments show that this system behaves exactly like a random MinRank instance in fields of odd characteristic. In fields of characteristic two, we occasionally observe some rank defects (see Appendix A). Since the observed defects are small, we believe that the complexity of solving random MinRank instances is a good estimate for the complexity of solving the MinRank instances coming from a Rainbow public key. We leave the investigation of the rank defects and quantifying how much is gained by adding the $\mathcal{P}(\mathbf{o}) = 0$ equations for future work.

Example 5. We estimate the cost of the combined attack against the SL 5 parameter set from the second-round submission to NIST. This paramater set is $q = 256, n = 188, m = 96, o_2 = 48$. This means that after guessing a good $D_{\mathbf{x}}$ (which happens with probability of approximately 1/255), we get a MinRank instance of $n - m = 92$ matrices with $n - 1 = 187$ rows and $m = 96$ columns, whose span contains a non-zero matrix of rank $o_2 = 48$. Solving this MinRank instance with the algorithm of Bardet *et al.* costs $2^{149.1}$ field multiplications (see Example 2). If the gate cost of a $\mathbb{F}_{256}$-multiplication is 128, then the total expected gate cost of the attack is $2^{149.1} \cdot 128 \cdot 255 \approx 2^{164.1}$, as reported in Table 1. This is an improvement by a factor 2^{27} over previously known attacks.

5 Experimental Results and Conclusion

To validate our attack and showcase that the attack is efficient enough to be performed in practice, we implemented a Sage script that generates a Rainbow public key, guesses a vector $\mathbf{x} \in \mathbb{F}_q^n$, and constructs (in fields of odd characteristic) the system $\tilde{\mathcal{P}}$ as described in Section 3, and writes it to a file in the format readable by the optimized implementation of the block Wiedemann XL algorithm by Cheng, Chou, Niederhagen, and Yang [6]. In fields of characteristic two, the script instead constructs and stores the slightly smaller $\hat{\mathcal{P}}$ system. We then then run the block Wiedemann XL algorithm on the stored systems, and find that it indeed finds solutions to $\tilde{\mathcal{P}}(\mathbf{x}) = 0$ (resp. $\hat{\mathcal{P}}(\mathbf{x}) = 0$) if the solutions exist.

The SL 1 parameter set of the second-round Rainbow submission is $(q = 16, n = 96, m = 64, o_2 = 32)$. For these parameters solving $\hat{\mathcal{P}}(\mathbf{x}) = 0$ takes three hours and 32 minutes on a laptop using the 8 cores of an Intel i9-10885H CPU, running at 2.5 GHz. The block Wiedemann XL algorithm reports on the rate at which it does $\mathbb{F}_{16}$-multiplications, which fluctuates between 130 and 200 multiplications per cycle. This is consistent with the estimate that solving the system takes $2^{52.3}$ multiplications (Example 1). Solving the system only uses 1.1 GB of memory. Since each guess $\mathbf{x}$ leads to a key recovery with a probability of 1/15.06, to total expected running time of the attack is $15.06 \cdot 3.53 \approx 53$ hours.

We can use the knowledge of the secret key to determine if a guess for $\mathbf{x}$ is good (i.e., if $\ker(D_{\mathbf{x}}) \cap O_2 \neq \{0\}$) without doing the expensive system-solving computation. This allows us to try a large number of guesses and count how often a guess is good. We made 4000 guesses and found that 242 of them are good, which is consistent with the null hypothesis of 1/15.06 (with a one-sided p-value of 0.085).

The sage implementation of our attack and scripts for reproducing the rank experiments of Appendix A are available at

https://github.com/WardBeullens/BreakingRainbow

We can conclude that the cost and success probability of the attack in practice agree very well with what the theory predicts. Moreover, we demonstrated that a key-recovery against the SL 1 parameter set of the second-round submission of Rainbow can be performed in practice by anyone with a decent laptop and some patience (or luck). A key-recovery attack against the SL 1 parameter set of the third-round Rainbow submission is expected to be more costly by only a factor 2^8, so this should be feasible for an attacker with a moderate amount of resources.

In principle, it would be possible to move to larger parameters to protect against the attacks presented in this paper, at the cost of larger key sizes and signature sizes. E.g., the SL III parameters of the third-round submission seem to provide enough security for SL I, but those parameters have signatures and public keys that are larger by a factor 2.5 and 4.4 respectively compared to the SL I parameters. However, there seems to be some room for improvement for the attacks in Section 4, so more cryptanalysis would be required before we can have confidence in the security of Rainbow. Moreover, the resulting Rainbow signature scheme would be less efficient than the Oil and Vinegar scheme. So there is seemingly no reason to prefer Rainbow over the Oil and Vinegar scheme [16], on which Rainbow is based, and which is older, simpler, and has a strictly smaller attack surface in comparison to Rainbow. (E.g., none of the attacks in this paper seem to apply to the Oil and Vinegar scheme).

References

[1] Response to recent paper by Ward Beullens. https://troll.iis.sinica.edu.tw/by-publ/recent/response-ward.pdf, 2020. 1

[2] John Baena, Pierre Briaud, Daniel Cabarcas, Ray Perlner, Daniel Smith-Tone, and Javier Verbel. Improving support-minors rank attacks: applications to G*e*MSS and rainbow. Cryptology ePrint Archive, Report 2021/1677, 2021. https://eprint.iacr.org/2021/1677. 1

[3] Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray A. Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier A. Verbel. Improvements of algebraic attacks for solving the rank decoding and MinRank problems. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 507–536. Springer, Heidelberg, December 2020. 1, 2, 4, 4

[4] Ward Beullens. Improved cryptanalysis of UOV and rainbow. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 348–373. Springer, Heidelberg, October 2021. 1, 1, 2, 2, 3, 4, 4

[5] Olivier Billet and Henri Gilbert. Cryptanalysis of Rainbow. In Roberto De Prisco and Moti Yung, editors, *SCN 06*, volume 4116 of *LNCS*, pages 336–347. Springer, Heidelberg, September 2006. 1

[6] Chen-Mou Cheng, Tung Chou, Ruben Niederhagen, and Bo-Yin Yang. Solving quadratic equations with XL on parallel architectures. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 356–373. Springer, Heidelberg, September 2012. 2, 5

[7] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 392–407. Springer, Heidelberg, May 2000. 2

[8] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *ACNS 05*, volume 3531 of *LNCS*, pages 164–175. Springer, Heidelberg, June 2005. 1

[9] Jintai Ding, Bo-Yin Yang, Chia-Hsin Owen Chen, Ming-Shing Chen, and Chen-Mou Cheng. New differential-algebraic attacks and reparametrization of Rainbow. In Steven M. Bellovin, Rosario Gennaro, Angelos D. Keromytis, and Moti Yung, editors, *ACNS 08*, volume 5037 of *LNCS*, pages 242–257. Springer, Heidelberg, June 2008. 1, 2

[10] Louis Goubin and Nicolas Courtois. Cryptanalysis of the TTM cryptosystem. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 44–57. Springer, Heidelberg, December 2000. 1

[11] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 206–222. Springer, Heidelberg, May 1999. 1

[12] Aviad Kipnis and Adi Shamir. Cryptanalysis of the oil & vinegar signature scheme. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 257–266. Springer, Heidelberg, August 1998. 1, 3

[13] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 19–30. Springer, Heidelberg, August 1999. 2

[14] Daniel Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In *European Conference on Computer Algebra*, pages 146–156. Springer, 1983. 2

[15] Wael Said Abdelmageed Mohamed, Jintai Ding, Thorsten Kleinjung, Stanislav Bulygin, and Johannes Buchmann. Pwxl: A parallel wiedemann-xl algorithm for solving polynomial equations over gf (2). In *Conference on Symbolic Computation and Cryptography*, page 89, 2010. 2

[16] Jacques Patarin. The oil and vinegar signature scheme. In *Dagstuhl Workshop on Cryptography September, 1997*, 1997. 1, 5

[17] Ray Perlner and Daniel Smith-Tone. Rainbow band separation is better than we thought. Cryptology ePrint Archive, Report 2020/702, 2020. https://eprint.iacr.org/2020/702. 1

[18] Bo-Yin Yang and Jiun-Ming Chen. Building secure tame-like multivariate public-key cryptosystems: The new TTS. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 05*, volume 3574 of *LNCS*, pages 518–531. Springer, Heidelberg, July 2005. 1

A Rank experiments

Simple Attack. For some rainbow parameter sets over $\mathbb{F}_{31}$ we construct some $\tilde{\mathcal{P}}(\mathbf{x}) = 0$ systems as in our simple attack, and compute the ranks of Macaulay matrices of these systems at various degrees. These ranks are displayed in Table 2. Similarly, for some rainbow parameters over $\mathbb{F}_{16}$, we construct some $\hat{\mathcal{P}}(\mathbf{x}) = 0$ systems, and we displayed the ranks of their Macaulay matrices in Table 3. We observe in both cases that the ranks are identical to the ranks of systems of uniformly random quadratic equations with the same dimensions. I.e., if $\tilde{\mathcal{P}}(\mathbf{x})$ (or $\hat{\mathcal{P}}(\mathbf{x})$) has m equations and n variables, then the rank of its Macaulay matrix at degree D is equal to the coefficient of t^D in the power series expansion of

$$(1-t)^n(1-(1-t^2)^m)\,,$$

if this coefficient is positive. Otherwise, the system has a kernel of dimension 1, which corresponds to the 1-dimensional space of solutions. This is evidence that the $\tilde{\mathcal{P}}(\mathbf{x}) = 0$ and $\hat{\mathcal{P}}(\mathbf{x}) = 0$ systems do not have special properties that make them easier or harder to solve in comparison with random systems.

Combined Attack. Table 4 reports on some of our rank experiments for the combined attack. For some small Rainbow parameter sets, we executed the combined attack from Section 4 to derive a MinRank instance with $n-m$ matrices with $n-1$ rows and m columns (of which we keep m'). Then we constructed the linearized systems as they appear in the MinRank solving algorithm of Bardet *et al.* at several bi-degrees $(b, 1)$, and we compute their ranks. We found that in odd characteristic, the rank of the Macaulay matrices always matches those of random MinRank instances with the same parameters. In contrast, we sometimes observe a small rank defect in characteristic two (they are underlined in the Table 4).

Table 2. The rank and the number of columns of the Macaulay matrices for the $\tilde{\mathcal{P}}(\mathbf{x}) = 0$ system of equations of simple attack over $\mathbb{F}_{31}$. Ranks of the Macaulay matrix of degree D is given in boldface if the system can be solved at that degree.

Rainbow parameters			$\tilde{\mathcal{P}}$ size			Rank of Macaulay matrix at degree D		
n	m	o_2	m	n		$D=2$	$D=3$	$D=4$
30	20	10	20	10	rank	20	200	**714**
					columns	55	220	715
45	30	15	30	15	rank	30	450	**3059**
					columns	120	680	3060
60	40	20	40	20	rank	40	800	7620
					columns	210	1540	8855

Table 3. The rank and the number of columns of the Macaulay matrices for the $\hat{\mathcal{P}}(\mathbf{x}) = 0$ system of equations of simple attack over $\mathbb{F}_{16}$. Ranks of the Macaulay matrix of degree D is given in boldface if the system can be solved at that degree.

Rainbow parameters			$\hat{\mathcal{P}}$ size			Rank of Macaulay matrix at degree D		
n	m	o_2	m	n		$D=2$	$D=3$	$D=4$
30	20	10	19	9	rank	19	**164**	
					columns	45	165	
36	24	12	23	11	rank	23	253	**1000**
					columns	66	286	1001
42	28	14	27	13	rank	27	351	**1819**
					columns	91	455	1820

Table 4. The rank and the number of columns of the Macaulay matrices for the MinRank problems from the combined attack over $\mathbb{F}_{31}$ and $\mathbb{F}_{16}$. Ranks of the Macaulay matrix at bi-degree $(b, 1)$ is given in boldface if the system can be solved at that bi-degree.

Rainbow parameters			MinRank parameters			Rank of Macaulay matrix at bi-degree $(b,1)$		
n	m	o_2	k	m'		$b = 1$	$b = 2$	$b = 3$
					rank in $\mathbb{F}_{31}$	**279**		
15	10	5	5	8	rank in $\mathbb{F}_{16}$	**279**		
					columns	280		
					rank in $\mathbb{F}_{31}$	98	**314**	
15	10	5	5	7	rank in $\mathbb{F}_{16}$	98	**314**	
					columns	105	315	
					rank in $\mathbb{F}_{31}$	78	533	**1799**
14	6	4	8	6	rank in $\mathbb{F}_{16}$	78	<u>527</u>	**1799**
					columns	120	540	1800