Selecting Parameters for the Rainbow Signature Scheme

Albrecht Petzoldt[1], Stanislav Bulygin[2], and Johannes Buchmann[1,2]

[1] Technische Universität Darmstadt, Department of Computer Science
Hochschulstraße 10, 64289 Darmstadt, Germany
{apetzoldt,buchmann}@cdc.informatik.tu-darmstadt.de
[2] Center for Advanced Security Research Darmstadt - CASED
Mornewegstraße 32, 64293 Darmstadt, Germany
{johannes.buchmann,Stanislav.Bulygin}@cased.de

Abstract. Multivariate public key cryptography is one of the main approaches to guarantee the security of communication in a post-quantum world. One of the most promising candidates in this area is the Rainbow signature scheme, which was first proposed by J. Ding and D. Schmidt in 2005. In this paper we develop a model of security for the Rainbow signature scheme. We use this model to find parameters which, under certain assumptions, guarantee the security of the scheme for now and the near future.

Keywords: Multivariate cryptography, Rainbow signature scheme, parameters.

1 Introduction

To guarantee the security of communication it is important to have fast and secure signature schemes. One major field of application for them is the authenticity of data and information, for example software updates.

One of the most promising candidates in this area is the Rainbow signature scheme, which was presented by J. Ding and D. Schmidt in [DS05]. Similarly to other multivariate schemes like $3iC^-p$ [DW07] and Projected Flash [PC01], [DY07] it is very efficient and provides fast signature generation and verification. In opposite to classical schemes, e.g. RSA or ECDSA, Rainbow is believed to be secure against attacks with quantum computers [BB08].

In the last years a lot of work has been done to study the security of multivariate schemes and many attacks were proposed. Among these are direct attacks on which a lot of work was done [YC07], [Fa99] as well as rank attacks which were introduced in [CS94] by Coppersmith and Stern to attack the Birational Permutation Scheme and later improved by a number of other researchers [YC05], [BG06]. A good overview of these attacks can be found in [GC00]. Special attacks on Rainbow-like schemes were proposed by Ding and Yang in [DY08]. There have also been some attempts to derive appropriate parameters from the complexities of these attacks [CC08]. However, it is still an open problem how

N. Sendrier (Ed.): PQCrypto 2010, LNCS 6061, pp. 218–240, 2010.

we have to adapt the parameters of multivariate schemes to future developments in cryptanalysis and computing power.

In this paper we try to answer this question for the Rainbow signature scheme. We start with the security model of Lenstra and Verheul [LV00] to compute necessary security levels for the years 2010 to 2050. After that we look at the known attacks against the Rainbow signature scheme and observe that the security of the scheme is mainly determined by two of them, namely the direct attack and the Rainbow-Band-Separation attack. We study the complexity of these two attacks, both with data from the literature and with own experiments, for which we use two of the most efficient available software packages, namely MAGMA and Singular. Finally, we use the results of these experiments to find appropriate parameters for Rainbow such that it fulfills the given security levels, as well as Rainbow schemes for limited hash sizes. One of our main results here is, that we need at least 26 equations to achieve the necessary security level for 2010. So, the often proposed scheme Rainbow(18,12,12) does not provide adequate security.

The structure of the paper is as follows: In Section 2 we describe the Rainbow signature scheme. Section 3 describes our model of security for the Rainbow scheme. In Section 4 we take a closer look at the complexities of the direct and the Rainbow-Band-Separation attack and present the results of our experiments. Section 5 gives secure parameter sets optimized for small public key sizes as well as Rainbow schemes for limited hash sizes. Finally, Section 6 concludes the paper.

2 Multivariate Public Key Cryptography

Multivariate Public Key Cryptography is one of the main approaches for secure communication in a post-quantum world. The principle idea is to choose a multivariate system F of quadratic polynomials which can be easily inverted (central map). After that one chooses two affine linear invertible maps S and T to hide the structure of the central map. The public key of the cryptosystem is the composed map $P = S \circ F \circ T$ which is difficult to invert. The private key consists of S, F and T and therefore allows to invert P.

There are several ways to build the central map F. One approach are the so called BigField-Schemes like Matsumoto-Imai [MI88] and HFE [Pa96] with many variations and improvements [BB08], [Di04], [PC01]. On the other hand, we have the so called SingleField family with schemes like UOV [KP99] and Rainbow [DS05]. Recently, a third family called MediumField has been proposed which contains schemes like ℓ-iC [DW07].

2.1 The Principle of Oil and Vinegar (OV)

One way to create easily invertible multivariate quadratic systems is the principle of Oil and Vinegar, which was first proposed by J. Patarin in [Pa97].

Let K be a finite field (e.g. $K = GF(2^8)$). Let o and v be two integers and set $n = o + v$. Patarin suggested to choose $o = v$. After this original scheme was broken by Kipnis and Shamir in [KS98], it was recommended in [KP99] to

choose $v > o$ (Unbalanced Oil and Vinegar (UOV)). In this Section we describe the more general approach UOV.

We set $V = \{1, \ldots, v\}$ and $O = \{v+1, \ldots, n\}$. Of the n variables $x_1, \ldots, x_n$ we call $x_1, \ldots, x_v$ the Vinegar variables and $x_{v+1}, \ldots, x_n$ Oil variables. We define o quadratic polynomials
$f_k(\mathbf{x}) = f_k(x_1, \ldots, x_n)$ by

$$f_k(\mathbf{x}) = \sum_{i \in V,\ j \in O} \alpha_{ij}^{(k)} x_i x_j + \sum_{i,j \in V,\ i \leq j} \beta_{ij}^{(k)} x_i x_j + \sum_{i \in V \cup O} \gamma_i^{(k)} x_i + \eta^{(k)} \ (k \in O)$$

Note that Oil and Vinegar variables are not fully mixed, just like oil and vinegar in a salad dressing.

The map $F = (f_{v+1}(\mathbf{x}), \ldots, f_n(\mathbf{x}))$ can be easily inverted. First, we choose the values of the v Vinegar variables $x_1, \ldots, x_v$ at random. Such we get a system of o linear equations in the o variables $x_{v+1}, \ldots, x_n$ which can be solved by Gaussian Elimination. (If the system doesn't have a solution, choose other values of $x_1, \ldots, x_v$ and try again).

2.2 The Rainbow Signature Scheme

In [DS05] J. Ding and D. Schmidt proposed a new signature scheme called Rainbow, which is based on the idea of Oil and Vinegar.

Let K be a finite field (e.g. $K = GF(2^8)$) and S be the set $\{1, \ldots, n\}$. Let $v_1, \ldots, v_{u+1}, u \geq 1$ be integers such that $0 < v_1 < v_2 < \cdots < v_u < v_{u+1} = n$ and define the sets of integers $S_i = \{1, \ldots, v_i\}$ for $i = 1, \ldots, u$. We set $o_i = v_{i+1} - v_i$ and $O_i = \{v_i + 1, \ldots, v_{i+1}\}$ $(i = 1, \ldots, u)$. The number of elements in S_i is v_i and we have $|O_i| = o_i$. For $k = v_1 + 1, \ldots, n$ we define multivariate quadratic polynomials in the n variables $x_1, \ldots, x_n$ by

$$f_k(\mathbf{x}) = \sum_{i \in O_l,\ j \in S_l} \alpha_{i,j}^{(k)} x_i x_j + \sum_{i,j \in S_l,\ i \leq j} \beta_{i,j}^{(k)} x_i x_j + \sum_{i \in S_l \cup O_l} \gamma_i^{(k)} x_i + \eta^{(k)},$$

where l is the only integer such that $k \in O_l$. Note that these are Oil and Vinegar polynomials with $x_i,\ i \in S_l$ being the Vinegar variables and $x_j,\ j \in O_l$ being the Oil variables.

The map $F(\mathbf{x}) = (f_{v_1+1}(\mathbf{x}), \ldots, f_n(\mathbf{x}))$ can be inverted as follows: First, we choose $x_1, \ldots, x_{v_1}$ at random. Hence we get a system of o_1 linear equations (given by the polynomials f_k $(k \in O_1)$) in the o_1 unknowns $x_{v_1+1}, \ldots, x_{v_2}$, which can be solved by Gaussian Elimination. The so computed values of x_i $(i \in O_1)$ are put into the polynomials $f_k(\mathbf{x})$ $(k > v_2)$ and a system of o_2 linear equations (given by the polynomials f_k $(k \in O_2)$) in the o_2 unknowns x_i $(i \in O_2)$ is obtained. By repeating this process we can get values for all the variables x_i $(i = 1, \ldots, n)$ [1].

[1] It may happen, that one of the linear systems does not have a solution. If so, one has to choose other values of $x_1, \ldots x_{v_1}$ and try again.

The Rainbow signature scheme is defined as follows:

Key Generation. The private key consists of two invertible affine maps $L_1 : K^m \to K^m$ and $L_2 : K^n \to K^n$ and the map $F = (f_{v_1+1}(\mathbf{x}), \ldots, f_n(\mathbf{x}))$. Here, $m = n - v_1$ is the number of components of F.

The public key consists of the field K and the composed map $P(\mathbf{x}) = L_1 \circ F \circ L_2(\mathbf{x}) : K^n \to K^m$.

Signature Generation. To sign a document d, we use a hash function $\mathbf{h} : K^* \to K^m$ to compute the value $\mathbf{h} = \mathbf{h}(d) \in K^m$. Then we compute recursively $\mathbf{x} = L_1^{-1}(\mathbf{h})$, $\mathbf{y} = F^{-1}(\mathbf{x})$ and $\mathbf{z} = L_2^{-1}(\mathbf{y})$. The signature of the document is $\mathbf{z} \in K^n$. Here, $F^{-1}(\mathbf{x})$ means finding one (of the possibly many) pre-image of $\mathbf{x}$.

Verification. To verify the authenticity of a signature, one simply computes $\mathbf{h}' = P(\mathbf{z})$ and the hashvalue $\mathbf{h} = \mathbf{h}(d)$ of the document. If $\mathbf{h}' = \mathbf{h}$ holds, the signature is accepted, otherwise rejected.

The size of the public key is (for $K = GF(2^8)$)

$$\text{size(public key)} = m \cdot \left(\frac{n \cdot (n+1)}{2} + n + 1 \right) = m \cdot \frac{(n+1) \cdot (n+2)}{2} \text{ bytes,} \quad (1)$$

the size of the private key

$$\begin{aligned} \text{size(private key)} = \; & m \cdot (m+1) + n \cdot (n+1) \\ & + \sum_{l=1}^{u} o_l \cdot \left(v_l \cdot o_l + \frac{v_l \cdot (v_l+1)}{2} + v_{l+1} + 1 \right) \text{ bytes.} \quad (2) \end{aligned}$$

The length of the needed hash value is m bytes, the length of the signature is n bytes.

The scheme is denoted by Rainbow$(v_1, o_1, \ldots, o_u)$. For $u = 1$ we get the original UOV scheme.

3 Our Model of Security

In this Section we describe the model underlying our parameter choices below. We base on the approach of Lenstra and Verheul [LV00].

3.1 The Model

In [LV00] Lenstra and Verheul developed a security model, which they used to find appropriate parameters for symmetric cryptography and some asymmetric schemes. The main points of their model are:

1. Security margin: a definition of the term "adequate security".
2. Computing environment: the expected change in computational resources available to attackers.
3. Cryptanalytic development: the expected development in cryptanalysis.

In the following we take a closer look at these items.

Security margin. To decide, whether a given scheme offers adequate security, one has to define the term "adequate security". [LV00] defines it by the security offered by DES in 1982. That is, in 1982 a computational effort of $5 \cdot 10^5$ MIPS years provided an adequate security. We follow this definition.

Computing environment. Here [LV00] use a slightly modified version of Moore's law, which states that the amount of computing power and random access memory one gets for 1 dollar doubles every t months. Our default setting of t is 18, see [LV00].

Another thing we have to take into account, is the budget of an attacker, which might increase over time. The variable $b > 0$ is defined as the number of years it takes on average for an expected two-fold increase of a budget. Statistical data says, that the US Gross National product (in today's prices) doubles about every ten years. So our default setting for b is 10.

Cryptanalytic Development. The number $r > 0$ is defined to be the number of months it is expected to take on average for cryptanalytic developments affecting Multivariate Public Key Cryptosystems to become twice as effective.

Under the assumption, that the pace of cryptanalytic findings in the area of multivariate cryptography will not vary dramatically from those in the field of classical cryptosystems, our default setting for r is $r = 18$.

After having developed concrete security levels based on these three items, Lenstra and Verheul analyzed known attacks against several schemes to get concrete parameter sets.

Analogous to [LV00], we will use "Infeasible number of MIPS years" (IMY) to define security requirements for the Rainbow signature scheme. Given that breaking DES takes $5 \cdot 10^5$ MIPS years, which was infeasible to do in year 1982, we get the number of MIPS years that are infeasible to break in the year y by the formula

$$IMY(y) = 5 \cdot 10^5 \cdot 2^{12(y-1982)/t} \cdot 2^{(y-1982)/b} \quad \text{MIPS years.} \tag{3}$$

With our default settings we get

$$IMY(y) = 2^{\frac{23}{30} \cdot y - 1500.6} \text{ MIPS years} \tag{4}$$

So far, we have not considered the possible advances in cryptanalysis. To cover these, we have to adapt the upper formula slightly. So, a cryptosystem, which shall be secure in the year y, must reach the security level

$$\text{Security level}(y) \geq IMY(y) \cdot 2^{12(y-2009)/r} \text{ MIPS years} \overset{r=18}{=} 2^{\frac{43}{30} \cdot y - 2839.9} \text{ MIPS years} \tag{5}$$

3.2 Security Level of Rainbow

In this subsection we look at the known attacks against the Rainbow signature scheme. We will find, that the security of the scheme is mainly given by the complexities of two attacks, namely the direct and the Rainbow-Band-Separation attack and therefore can be said to be the minimum of those two complexities.

The known attacks against the Rainbow Signature Scheme are:

1. direct attacks [BB08], [Ya07]: Direct attacks use equation solvers like XL and its derivatives as well as Gröbner Basis algorithms: Buchberger, F_4, and F_5. The complexity is approximately given as

$$C_{\text{direct}}(q, m, n) = C_{MQ(q,m,n)}, \tag{6}$$

   where $C_{MQ(q,m,n)}$ denotes the complexity of solving a "generic" system of m quadratic equations in n variables over a field with q elements.
2. MinRank attack [GC00], [YC05]

$$C_{\text{MR}}(q, m, n, v_1) = [q^{v_1+1} \cdot m \cdot (n^2/2 - m^2/6)] \text{ m} \tag{7}$$

3. HighRank attack [GC00], [DY08]

$$C_{\text{HR}}(q, n, o_u) = [q^{o_u} \cdot n^3/6] \text{ m} \tag{8}$$

4. UOV attack [KP99]

$$C_{\text{UOV}}(q, n, o_u) = [q^{n-2\cdot o_u-1} \cdot o_u^4] \text{ m} \tag{9}$$

5. UOV-Reconciliation attack [BB08], [DY08]

$$C_{\text{UOVR}}(q, m, n, o_u) = C_{MQ(q,m,n-o_u)} \tag{10}$$

6. Rainbow-Band-Separation attack [DY08]

$$C_{\text{RBS}}(q, m, n) = C_{MQ(q,m+n-1,n)} \tag{11}$$

Here, m stands for the number of field multiplications needed during the attack. Defending a Rainbow scheme against the attacks from the items 2 to 4 is relatively easy:

Proposition 1: A Rainbow instance over $GF(2^a)$ with parameters $v_1, o_1, \ldots, o_u$ (see Section 2.2) and $n \geq m \geq 10$, for which the items

1. $v_1 \geq \frac{\ell}{a} - 1$
2. $o_u \geq \frac{\ell}{a}$
3. $n - 2 \cdot o_u \geq \frac{\ell}{a} + 1$

hold, has a security level of ℓ bits against the MinRank, the HighRank and the UOV attack.

Proof: The proof of Proposition 1 can be found in the appendix of this paper.

Table 1 shows how we have to adapt the parameters of Rainbow over time according to Proposition 1.

Table 1. Parameter restrictions according to Proposition 1

years	MinRank $v_1 \geq$	HighRank $o_u \geq$	UOV-Attack $n - 2o_u \geq$
2010	9	10	11
2011-2015	10	11	12
2016-2021	11	12	13
2022-2027	12	13	14
2028-2032	13	14	15
2033-2038	14	15	16
2039-2043	15	16	17
2044-2049	16	17	18
2050-2055	17	18	19

To defend the scheme against the UOV-Reconciliation attack, we need $v_1 \geq o_u$. Then, the algebraic part of the attack leads to an underdetermined system of quadratic equations which is as difficult to solve as a direct attack against the original scheme.

In the following, we look at Rainbow instances which are secure against these four attacks. So, the security of a Rainbow scheme depends only on the complexities of the direct and the Rainbow-Band-Separation (RBS) attack and therefore can be said to be the minimum of those two complexities. Hence, it depends only on the number of equations m and the number of variables n. The security of a scheme is therefore given by

$$\mathrm{Sec}(m, n) = \min\{C_{\mathrm{direct}}(q, m, n), C_{\mathrm{RBS}}(q, m, n)\} \tag{12}$$

In the next Section, we will take a closer look at these two complexities.

4 Complexity Estimations for the Direct and RBS Attacks

In this Section we look at the complexities of the direct and the RBS attack against Rainbow schemes. For both of these two attacks we have to solve systems of quadratic equations. We look at several ways to do this step, namely XL-Wiedemann, the implementation of Faugere's F4 algorithm contained in MAGMA and Singular's implementation of Buchberger's algorithm. Unfortunately, our analysis does not contain the F5 algorithm, because no implementation of it is publicly available. Throughout this section, we look at Rainbow Schemes defined over a field of 256 elements.

4.1 XL-Wiedemann

In this subsection we look at XL-Wiedemann, which it is the best analyzed general method for solving systems of quadratic equations. The complexity of

XL-Wiedemann for an overdetermined system of m quadratic equations in n variables ($m \geq n$) over a field with q elements is given by the formula [BB08]

$$\mathrm{C}_{\mathrm{XL-Wiedemann}} = [3 \cdot T^2 \cdot \tau] \text{ m}, \tag{13}$$

where T is the number of monomials up to a certain degree D reduced mod $x^q - x$ ($T = \binom{n+D}{D}$ for larger fields), and τ stands for the average number of terms per equation, which in our case is given by $\tau = \frac{(n+1)\cdot(n+2)}{2}$. The minimal degree D_0, for which XL works is given by

$D_0 = \min\{D : [t^D]((1-t)^{m-n-1}(1+t)^m) \leq D\}$, where $[t^D](p)$ denotes the coefficient of t^D in p.

By guessing some of the variables before applying the algorithm it might be possible to decrease the necessary degree D. This can speed up the time needed to solve the system even if one has to run the algorithm several times with different guesses (see also the discussion of FXL in [Ya07]).

Direct attacks on Rainbow Schemes. When attacking a Rainbow instance directly we have to solve an underdetermined system of quadratic equations. Since XL-Wiedemann doesn't work for underdetermined systems, we have to guess at $n - m$ variables to create a determined system before applying the algorithm[2]. We computed the complexity of solving determined systems for $10 \leq m \leq 50$ equations by formula (13) with guessing at 1, 2, 3 or 4 variables[3]. We found, that for $m \leq 30$ we get the best results when guessing at 2 variables and then try to solve the 256^2 overdetermined systems with m equations in $m - 2$ variables. For $m \geq 31$ it seems to be even better to guess at 3 variables before applying the algorithm to all of the 256^3 overdetermined systems (see figure 1 in the appendix of this paper).

When guessing at more variables, the slope of the corresponding graphs is even less, but due to the guessing part we have to run the algorithm so often, that this does not help much. By guessing at 2 resp. 3 variables, we get the following approximation for the complexity of solving an (under-)determined system by XL-Wiedemann

$$\begin{aligned} C_{\mathrm{XL-Wiedemann}}(m) &= \left[2^{2.84\cdot m+17.6}\right] \text{m} \quad (10 \leq m \leq 30) \\ C_{\mathrm{XL-Wiedemann}}(m) &= \left[2^{2.62\cdot m+24.3}\right] \text{m} \quad (31 \leq m \leq 50) \end{aligned} \tag{14}$$

Complexity of the RBS attack. The algebraic part of the RBS attack leads to a system of $m + n - 1$ quadratic equations in n variables. So we have to solve an overdetermined system of equations. We can use XL-Wiedemann on this system without guessing at any variables. Figure 2 (in the appendix) shows the complexity of this attack for some fixed values of m.

[2] It might occur that the determined system one gets after the guessing part does not have a solution. If this happens, one has to choose other values for the variables to be guessed and try again. However, this occurs very rarely, so that we do not take this case into further consideration.

[3] Without guessing, XL runs at degree q, which is impractical for $q = 256$ [Ya07].

While the number of equations we need for the security of the scheme is given by the complexity of the direct attack, the necessary number of variables is determined by the complexity of the RBS attack.

We define $\bar{n}(m)$ to be the minimum number of variables, such that the RBS attack on the Rainbow instance with m equations and n variables has at least the complexity of the direct attack, i.e.

$$\bar{n}(m) = \min\{n|\ C_{\text{RBS}}(q, m, n) \geq C_{\text{direct}}(q, m, n)\} \tag{15}$$

This number $\bar{n}$ is optimal in the following sense: (1) for less than $\bar{n}$ variables the security of the Rainbow scheme is not as high as it would be possible for a Rainbow scheme with m equations and (2) more than $\bar{n}$ variables will not provide higher security, but increase the size of the public (and the private) key.

The optimal ratio between m and n depends on the algorithm one uses for solving the systems of quadratic equations.

To get $\bar{n}$ for XL-Wiedemann, we computed the complexity of the direct attack on Rainbow schemes with m equations ($10 \leq m \leq 50$). After that, we computed the complexity of the RBS attack on a Rainbow scheme with m equations and $n = m + 1$ variables. Then we increased n until this complexity was at least the same as that of the direct attack against a scheme with m equations. We found

$$\bar{n}_{\text{XL}-\text{Wiedemann}} = 2 \cdot (m - 1). \tag{16}$$

To translate the complexity estimations of formula (14), which are given in $GF(2^8)$-field multiplications, into MIPS years, we use a data-point computed by J. Ding et al. in [DY08]. There the authors solve a system of 37 quadratic equations in 22 variables over $GF(2^8)$ in about $1.06 \cdot 10^6$ seconds on a single 2.2 GHz Opteron machine by XL-Wiedemann. This corresponds to approximately 329.7 MIPS years [4]. Since the complexity of the system is about $2^{46.7}$ m, we get

$$1 \text{ MIPS year} = 3.49 \cdot 10^{11} \text{ m} \tag{17}$$

4.2 MAGMA's Implementation of the F4 Algorithm

We carried out a number of experiments with MAGMA, which contains an implementation of Faugere's F4 algorithm for computing Gröbner bases. We used MAGMA version 2.13-10 and solved the quadratic systems (given as sequences of polynomials in reversed graded lexicographical order) by the function `GroebnerBasis`. The experiments were carried out on a single core Opteron 2.3 GHz CPU, which achieves about 10200 MIPS. So a MIPS year corresponds to 3094 seconds or roughly 52 minutes of computation.

Running time of direct attacks. For a direct attack on a Rainbow scheme one has to solve an underdetermined system of m quadratic equations in $n > m$ variables. Since the Gröbner bases MAGMA creates for underdetermined systems are not convenient for our purposes (because of the high complexity of the

[4] The given processor achieves about 9800 MIPS (SiSoft Sandra).

Table 2. Solving determined systems with F4 with guessing

# equations	11	12	13	14	15	16
no guessing	6.4 m 342 MB	0.8 h 1236 MB	6.6 h 7426 MB	47.2 h 35182 MB	- ooM	-
guessing 1 variable	29 m 11 MB	2.8 h 23 MB	23 h 76 MB	134 h 285 MB	48 d 997 MB	257 d 3953 MB
guessing 2 variables	264 m 8.6 MB	30 h 10.7 MB	170 h 14.5 MB	1214 h 42 MB	230 d 118 MB	1259 d 335 MB
guessing 3 variables	5880 m 8.3 MB	715 h 9.0 MB	3830 h 11.2 MB	23597 h 14.8 MB	4449 d 24.8 MB	18443 d 51.7 MB
guessing 4 variables	93807 m 7.9 MB	8126 h 8.6 MB	43465 h 10.6 MB	22652 h 11.8 MB	67129 d 12.9 MB	382986 d 18.0 MB

corresponding variety), we had to guess at at least $n - m$ variables before applying the algorithm. Table 2 shows the results of our experiments when solving determined systems and guessing at $a = 0, 1, 2, 3$ or 4 variables. When doing so, we had to multiply the running time of MAGMA by a factor 256^a.

So, in our examples, we get the best results without guessing. But, as our extrapolation shows, for $m \geq 22$ equations it will be better to guess at one variable, and for $m \geq 29$ to guess at two variables before applying F4 (see figure 3 in the appendix).

The time MAGMA needs for solving a determined system with m equations can then be estimated by the formula

$$\begin{aligned} \mathrm{RT}_{\mathrm{F4}}(2^8, \mathrm{m}) &= 2^{2.74 \cdot m - 19.4} \text{ sec} \quad (22 \leq m \leq 28) \\ \mathrm{RT}_{\mathrm{F4}}(2^8, m) &= 2^{2.55 \cdot m - 13.9} \text{ sec} \quad (29 \leq m \leq 50) \end{aligned} \tag{18}$$

Running time of the RBS attack. In this paragraph we try to find out the optimal number of variables $\bar{n}(m)$ we should use in our Rainbow scheme (as defined by formula (15)). To get this number, we carried out some experiments where the number m of equations is a small multiple of the number n of variables. Table 3 shows the results.

Systems like these are created during the first part of the RBS attack, where an underdetermined system of m equations in n variables leads to an overdetermined system with $m + n - 1$ equations in n variables. So, a system of n variables and $a \cdot n$ equations is created from a Rainbow scheme with n variables and $(a - 1) \cdot n + 1$ equations. The other way round, a Rainbow scheme with m equations and $n = a \cdot (m - 1)$ variables leads via the first part of the RBS attack to an overdetermined system of $\frac{a+1}{a} \cdot n$ equations in n variables. For example, a Rainbow scheme with $m = 16$ equations in $n = \frac{5}{3} \cdot (m - 1) = 25$ variables leads to an overdetermined system of $\frac{8}{5} \cdot n = 40$ equations in $n = 25$ variables. Figure 4 shows the data-points from the table in terms of the number of equations of the original scheme.

Table 3. Solving overdetermined systems with F4

$m = \frac{3}{2} \cdot n$	# equations	21	24	27	30
	# variables	14	16	18	20
		36 s 30 MB	804 s 214 MB	7293 s 765 MB	120831 s 2890 MB
$m = \frac{8}{5} \cdot n$	# equations	16	24	32	
	# variables	10	15	20	
		0.15 s 0.7 MB	52.5 s 37 MB	18263 s 2081 MB	
$m = \frac{5}{3} \cdot n$	# equations	20	25	30	35
	# variables	12	15	18	21
		0.8 s 1.2 MB	42,7 s 36 MB	985 s 231 MB	40298 s 3291 MB

As can be seen from the graph, for a Rainbow scheme with m and $n = \frac{5}{3} \cdot (m-1)$ variables the running time of the RBS attack is nearly the same as that of a direct attack on the same scheme (when solving the quadratic systems with MAGMA). So, for MAGMA, the number $\bar{n}$ of variables we should use in our Rainbow scheme is given by

$$\bar{n}_{\mathrm{F4}}(m) = \lceil \tfrac{5}{3} \cdot (m-1) \rceil \tag{19}$$

4.3 Singular's Implementation of Buchbergers Algorithm

For better comparison, we also carried out some experiments with Singular [GP09], which contains an efficient implementation of Buchberger's algorithm. We used Singular version 3-1-0 and the command `std` to find Groebner bases for ideals of polynomials in reversed graded lexicographical order. The experiments with Singular were carried out in the same environment as those with MAGMA, namely on a single core Opteron 2.3 GHz CPU, which achieves about 10200 MIPS.

Running time of direct attacks. When attacking a Rainbow scheme directly, one has to solve an underdetermined system of m quadratic equations in $n > m$ variables. Since Buchberger's algorithm does not lead to a "nice" Gröbner basis for such a system, one has to guess at at least $n - m$ variables. We carried out experiments on determined systems by guessing at $a = 0, 1, 2, 3$ or 4 variables before applying Buchberger's algorithm. Again we had to run the algorithm 256^a times. Table 4 shows the results of these experiments.

So, in our experiments, we get the best results without guessing. But, as it seems, for $m \geq 23$ it should be better to guess at one variable and then to find Gröbner bases for all the 256 overdetermined systems with m equations and $m-1$ variables. And for $m \geq 30$, it might be even better to guess at 2 variables before applying Buchbergers algorithm (see figure 5 in the appendix). Note, that these results are very similar to those with MAGMA.

Table 4. Solving determined systems with Buchberger's algorithm with guessing

# equations	11	12	13	14	15	16
no guessing	12.3 m 137 MB	1.7 h 264 MB	15 h 1167 MB	146 h 4268 MB	-	-
guessing 1 variable	96.5 m 8.4 MB	11.9 h 29 MB	62.4 h 99 MB	548 h 354 MB	137 d 1280 MB	957.2 d 4631 MB
guessing 2 variables	1442 m 1.7 MB	142 h 4.8 MB	975 h 14 MB	4174 h 40 MB	1041 d 145 MB	8134 d 467 MB
guessing 3 variables	19712 m 0.7 MB	1590 h 1.2 MB	12175 h 3.3 MB	63847 h 8.0 MB	10892 d 23 MB	95313 d 69 MB
guessing 4 variables	456167 m 0.6 MB	45661 h 0.8 MB	267564 h 1.2 MB	1593056 h 2.3 MB	382769 d 5.4 MB	2250127 d 14 MB

Hence, we can estimate the time needed for solving a determined system with m equations with Buchberger's algorithm by the formula

$$\begin{aligned} \mathrm{RT}_{\mathrm{Buchberger}}(2^8, m) &= 2^{2.76 \cdot m - 17.9} \text{ s} \quad (23 \leq m \leq 29) \\ \mathrm{RT}_{\mathrm{Buchberger}}(2^8, m) &= 2^{2.55 \cdot m - 11.7} \text{ s} \quad (30 \leq m \leq 50) \end{aligned} \tag{20}$$

As a comparison of formulas (18) and (20) shows, for large m MAGMA solves the systems about 4.5 times faster.

Running time of the RBS attack. In this paragraph we try to find out the optimal number of variables $\bar{n}$ we should use in our Rainbow scheme (as defined by formula (15)). To get this number, we carried out the same experiments as presented for MAGMA in the previous subsection with Singular, too (see Table 5).

Table 5. Solving overdetermined systems with Buchberger's algorithm

$m = \frac{3}{2} \cdot n$	# equations	21	24	27	30
	# variables	14	16	18	20
		219 s 35 MB	1871 s 178 MB	16500 s 960 MB	179473 s 4953 MB
$m = \frac{8}{5} \cdot n$	# equations	16	24	32	
	# variables	10	15	20	
		0.7 s 2.3 MB	250 s 50 MB	109468 s 2130 MB	
$m = \frac{5}{3} \cdot n$	# equations	20	25	30	35
	# variables	12	15	18	21
		4.1 s 5.9 MB	113.5 s 37 MB	4723 s 335 MB	172863 s 2520 MB

Again we looked at the Rainbow schemes from which these systems were created during the first part of the RBS attack (see figure 6 in the appendix).

As the figure shows, the running time of the RBS attack against a Rainbow scheme with m equations in $n = \frac{5}{3} \cdot (m-1)$ variables is nearly the same as the running time of a direct attack against a scheme with m equations. The results are very similar to those with MAGMA, and so we get

$$\bar{n}_{\text{Buchberger}}(m) = \lceil \tfrac{5}{3} \cdot (m-1) \rceil \tag{21}$$

4.4 Storage

As you can see from tables 2 and 4, the storage both MAGMA and Singular need for solving determined systems is generally large (for example, we can't solve a determined system of 15 equations on our server with 128 GB main memory using MAGMA's F_4 algorithm). However, when guessing at some of the variables, the storage needed for solving the systems reduces by a large factor. To this, we want to remark the following:

When fixing a variables before applying F_4 or Buchbergers algorithm, one has to run the algorithm q^a times to find a solution (where q is the number of elements in the underlying field). So the time needed to solve the original system is $q^a \cdot t_a$, where t_a is the time one needs to solve the overdetermined system. But, the storage needed to solve the system is not influenced by the repeated execution of the algorithm. If one overdetermined system does not have a solution, one discards it and tries again with a different choice of the fixed variables. That is why fixing some variables reduces the storage needed to solve the systems by a large amount.

Since we found out in the previous subsections, that guessing 2 or 3 variables is also a good strategy to reduce the time you need to solve the system, we do not think that storage is a bigger problem when attacking Rainbow. That's why we don't take storage into account when proposing the parameters.

One interesting thing concerning the storage is, that the values for Singular (see tables 4 and 5) are not much better (and in some cases even worse) as those for MAGMA, especially for large numbers of equations. This is against our expectations, according to which Buchbergers algorithm should require less memory than F_4.

5 Parameter Choice

In this Section we give actual parameters which guarantee the security of the Rainbow signature scheme for now and the near future under the model assumptions presented in Section 3. The parameters are chosen for an underlying field of 256 elements.

5.1 Parameters for Rainbow Optimized for Small Public Key Size

The following table shows the number of equations m and variables n we propose to use for Rainbow in the year y. The numbers are chosen to be the smallest ones, such that a Rainbow scheme with m equations in n variables fulfills the necessary security levels for each of the three algorithms we looked at.

In each case there are various possibilities for the actual layout of the Rainbow layers. As long as one takes into account the limitations given in Table 1, the details of the layout will not affect the security of the scheme. For each pair of m and n we give one example scheme.

To get the numbers in the table, we first computed for each year y the minimal number m_0 of equations such that the complexity of a direct attack against the scheme is as least as high as the security level for the year y , i.e.

$$m_0(y) = \min\{m|\ C_{\text{direct}}(q, m) \geq \text{ Security level}(y)\} \tag{22}$$

For this step we used our results with MAGMA (as presented in Section 4.2), because they lead to the most conservative parameter choices.
Such we get (with formulas (5) and (18))

$$\begin{aligned} m_0(y) &= \lceil 0.523 \cdot y - 1025.3 \rceil \ (y \leq 2014) \\ m_0(y) &= \lceil 0.562 \cdot y - 1103.7 \rceil \ (2015 \leq y \leq 2050) \end{aligned} \tag{23}$$

We define $n_0(y)$ to be the minimal number of variables such that the complexity of the RBS attack against a Rainbow Scheme with m_0 equations and n_0 variables lies above our security margin, i. e.

$$n_0(y) = \min\{n|\ C_{\text{RBS}}(q, m_0(y), n) \geq \text{ Security level}(y)\}, \tag{24}$$

where $m_0(y)$ is defined by formula (22).

One can easily see that $\bar{n}(m_0)$ (as defined by formula (15)) fulfills this condition. Thus we have $n_0(y) \leq \bar{n}(m_0)$.

To find $n_0(y)$ for XL-Wiedemann, we use a similar technique as we used in Section 4.1 to find $\bar{n}$.

For each year y ($2010 \leq y \leq 2050$) we first computed the neccessary number $m_0(y)$ of equations and the complexity of an RBS attack against a scheme with m_0 equations in $n = m_0 + 1$ variables. Then we increased n until the complexity of the RBS attack reached our Security level. Note that the numbers $n_0(y)$ we get by doing so fulfill the conditions

$$\begin{aligned} n_0(y) &\geq \bar{n}_{\text{MAGMA}}(m_0) \ \text{ and} \\ n_0(y) &\geq \bar{n}_{\text{Singular}}(m_0) \end{aligned} \tag{25}$$

(see formulas (19) and (21)).

Therefore we have

$$C_{\text{RBS}_{\text{MAGMA}}}(q, m_0, n_0) \geq C_{\text{direct}_{\text{MAGMA}}}(q, m_0, n_0) \geq \text{Security level}(y) \text{ and} \tag{26}$$

$$C_{\text{RBS}_{\text{Singular}}}(q, m_0, n_0) \geq C_{\text{direct}_{\text{Singular}}}(q, m_0, n_0) \geq \text{Security level}(y) \tag{27}$$

So, the proposed schemes will be secure for MAGMA and Singular, too.

Table 6. Proposed parameters for the Rainbow Signature Scheme

					example scheme	
Year	(m, n)	public key size (kB)	hash size (bit)	signature size (bit)	(v_1, o_1, o_2)	private key size (kB)
2010	(26,43)	25.7	208	344	(17,13,13)	19.1
2020	(32,53)	47.5	256	424	(21,16,16)	34.3
2030	(38,65)	84.0	304	520	(27,19,19)	60.5
2040	(43,74)	122.6	344	592	(31,21,22)	87.7
2050	(49,85)	183.3	392	680	(36,24,25)	130.2

Proposition 2: By following the above strategy we get not only the minimal m but also the minimal n required for the security of Rainbow in the year y. Hence, the schemes proposed in the table below also minimize the size of the public key.

Proof: The proof of Proposition 2 can be found in the appendix of this paper.

The complete table of the proposed parameters can be found in the appendix of this paper.

5.2 Rainbow Schemes for Limited Hash Size

In this subsection we look at the question what security level we can achieve with restricted hash sizes. We want to examine until what year it is safe to use Rainbow along with a hash function of a given size and look for the optimal Rainbow parameters in this scenario.

A given size s of the hash value determines the number of equations in our system by $m = \frac{s}{8}$. So the maximal security we can obtain is upper bounded by the complexity of solving an underdetermined system of m quadratic equations. In the following, we use the results we got from our experiments with MAGMA, because they lead to the most conservative parameter choices.

To compute the year until which it is secure to use Rainbow along with a hash function of a given size s ($s \equiv 0 \bmod 8$ bit) we have to invert formula (23) and come up with

$$\begin{aligned} y &= \lfloor 0.239 \cdot s + 1960.1 \rfloor \ (s \leq 224) \\ y &= \lfloor 0.223 \cdot s + 1963.9 \rfloor (232 \leq s \leq 400) \end{aligned} \tag{28}$$

Table 7 shows for several hash sizes the year until which it is safe to use Rainbow along with a hash function of the given size as well as one example scheme.

As the table shows, using Rainbow along with a hash function which provides a hash length of 160 bit (like SHA-1), is no longer secure.

Table 7. Rainbow Schemes for limited hash size

hash size (bit)	m	secure until (formula (27))	proposed scheme (m,n)	public key size (kB)	(v_1, o_1, o_2)	private key size (kB)
208	26	2010	(26,43)	25.7	(17,13,13)	19.1
224	28	2013	(28,46)	31.6	(18,14,14)	23.1
240	30	2017	(30,51)	41.3	(21,15,15)	30.5
256	32	2020	(32,53)	47.3	(21,16,16)	34.4
288	36	2027	(36,60)	68.1	(24,18,18)	48.7
320	40	2035	(40,69)	99.4	(29,20,20)	71.6
352	44	2042	(44,78)	139.0	(32,22,22)	94.4
384	48	2049	(48,85)	179.6	(37,24,24)	128.8

6 Conclusion

Although nobody can say, which cryptanalytic developments and developments in computing devices will take place in the next years, we hope that this paper will help people to choose appropriate parameters for the Rainbow signature scheme. The proposed parameter sets should give the reader an impression, what public key sizes are needed to achieve given levels of security.

Acknowledgements

The first author wants to thank Jintai Ding, Bo-Yin Yang and Erik Dahmen for many helpful comments.

References

[BB08] Bernstein, D.J., Buchmann, J., Dahmen, E. (eds.): Post Quantum Cryptography. Springer, Heidelberg (2009)

[BG06] Billet, O., Gilbert, H.: Cryptanalysis of Rainbow. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 336–347. Springer, Heidelberg (2006)

[CC08] Chen, A.I.-T., Chen, C.-H.O., Chen, M.-S., Cheng, C.M., Yang, B.-Y.: Practical-Sized Instances for Multivariate PKCs: Rainbow, TTS and ℓIC-Derivatives. In: Buchmann, J., Ding, J. (eds.) PQCrypto 2008. LNCS, vol. 5299, pp. 95–108. Springer, Heidelberg (2008)

[CS94] Coppersmith, D., Stern, J., Vaudenay, S.: Attacks on the Birational Signature Scheme. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 435–443. Springer, Heidelberg (1994)

[DS05] Ding, J., Schmidt, D.: Rainbow, a new multivariate polynomial signature scheme. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 164–175. Springer, Heidelberg (2005)

[Di04] Ding, J.: A new variant of the Matsumoto-Imai cryptosystem through perturbation. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 305–318. Springer, Heidelberg (2004)

[DY08] Ding, J., Yang, B.-Y., Chen, C.-H.O., Chen, M.-S., Cheng, C.M.: New Differential-Algebraic Attacks and Reparametrization of Rainbow. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 242–257. Springer, Heidelberg (2008)

[DW07] Ding, J., Wolf, C., Yang, B.-Y.: ℓ-invertible Cycles for Multivariate Quadratic Public Key Cryptography. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 266–281. Springer, Heidelberg (2007)

[DY07] Ding, J., Yang, B.-Y., Cheng, C.-M., Chen, O., Dubois, V.: Breaking the symmetry: A way to resist the new Differential attacks, `http://www.eprint.iacr.org/2007/366.pdf`

[Fa99] Faugere, J.C.: A new efficient algorithm for computing Groebner bases (F4). Journal of Pure and Applied Algebra 139, 61–88 (1999)

[Fa02] Faugere, J.C.: A new efficient algorithm for computing Groebner bases without reduction to zero (F5). In: International Symposium on Symbolic and Algebraic Computation — ISSAC 2002, pp. 75–83. ACM Press, New York (2002)

[FP08] Faugere, J.-C., Perret, L.: On the security of UOV. In: Proceedings of the First International Conference on Symbolic Computation and Cryptology, Beijing (2008)

[GC00] Goubin, L., Courtois, N.T.: Cryptanalysis of the TTM cryptosystem. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 44–57. Springer, Heidelberg (2000)

[GP09] Greuel, G.-M., Pfister, G., Schönemann, H.: Singular 3.1.0 — A computer algebra system for polynomial computations (2009), `http://www.singular.uni-kl.de`

[KP99] Kipnis, A., Patarin, L., Goubin, L.: Unbalanced Oil and Vinegar Schemes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 206–222. Springer, Heidelberg (1999)

[KS98] Kipnis, A., Shamir, A.: Cryptanalysis of the Oil and Vinegar Signature scheme. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 257–266. Springer, Heidelberg (1998)

[LV00] Lenstra, A.K., Verheul, E.R.: Selecting Cryptographic Key Sizes. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 446–465. Springer, Heidelberg (2000), `www.keylength.com`

[MI88] Matsumoto, T., Imai, H.: Public Quadratic Polynomial-Tuples for efficient Signature-Verification and Message-Encryption. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 419–453. Springer, Heidelberg (1988)

[Pa96] Patarin, J.: Hidden Field equations (HFE) and Isomorphisms of Polynomials (IP). In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 33–48. Springer, Heidelberg (1996)

[Pa97] Patarin, J.: The oil and vinegar signature scheme. Presented at the Dagstuhl Workshop on Cryptography (September 1997)

[PG98] Patarin, J., Goubin, L., Courtois, N.: $C^{\star}_{+}$ and HM: Variations about two schemes of H. Matsumoto and T. Imai. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 35–50. Springer, Heidelberg (1998)

[PC01] Patarin, J., Courtois, N., Goubin, L.: Flash, a fast multivariate signature algorithm. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 298–307. Springer, Heidelberg (2001)

[YC05] Yang, B.-Y., Chen, J.-M.: Building secure tame like multivariate public-key cryptosystems: The new TTS. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 518–531. Springer, Heidelberg (2005)

[YC07] Yang, B.-Y., Chen, J.-M.: All in the XL family: Theory and practice. In: Park, C.-s., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 67–86. Springer, Heidelberg (2005)

[Ya07] Yang, B.-Y., Chen, C.-H.O., Bernstein, D.J., Chen, J.-M.: Analysis of QUAD. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 290–308. Springer, Heidelberg (2007)

Appendix

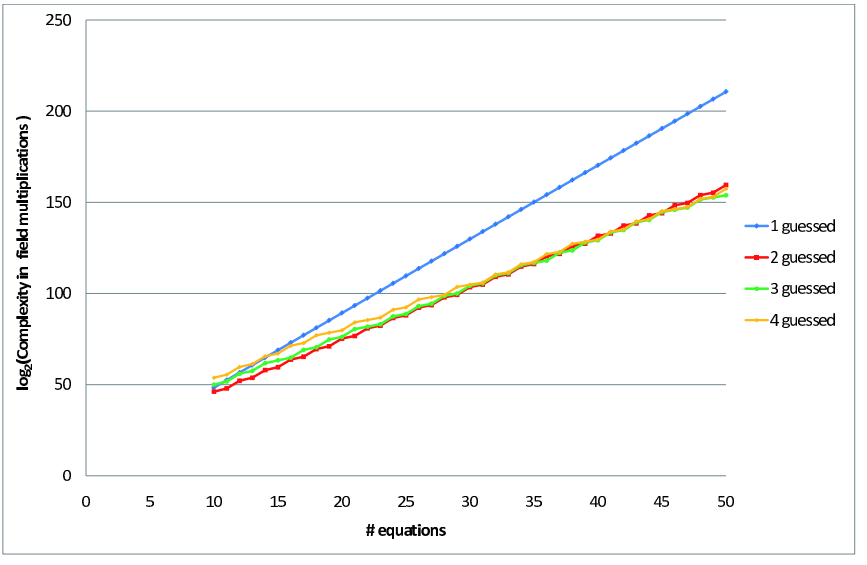

Fig. 1. Solving determined systemes with XL-Wiedemann with guessing additional variables

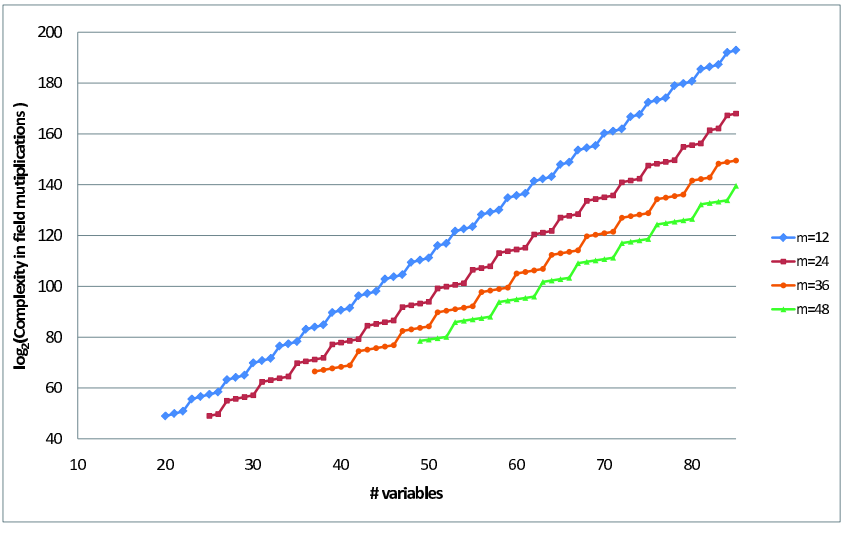

Fig. 2. Complexity of the RBS attack with XL-Wiedemann for different numbers of equations

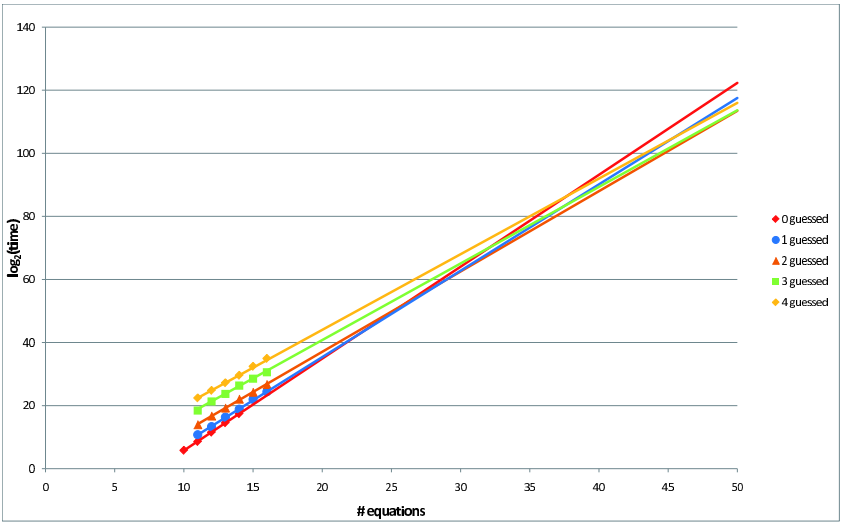

Fig. 3. Solving determined systems with F4 with guessing (datapoints and extrapolation)

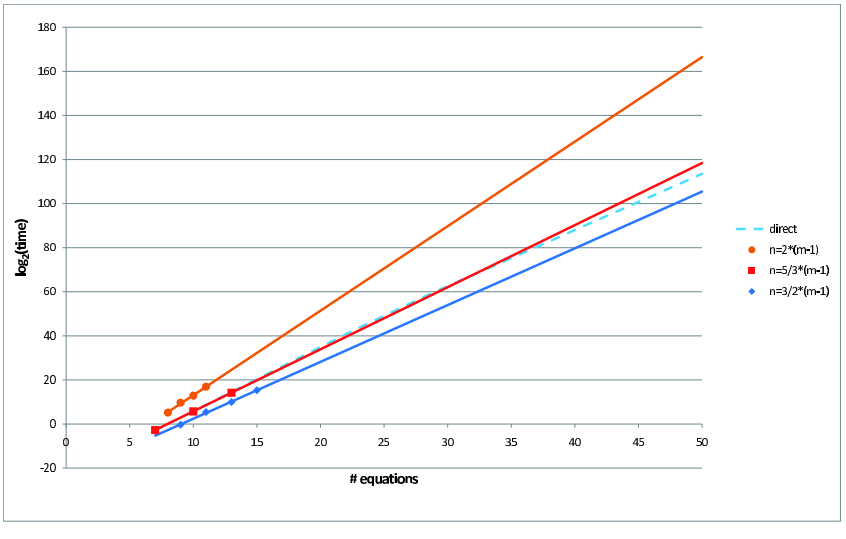

Fig. 4. Running time of the RBS attack with F4 for different ratios of m and n The dotted line marks the running time of a direct attack against a scheme with m equations

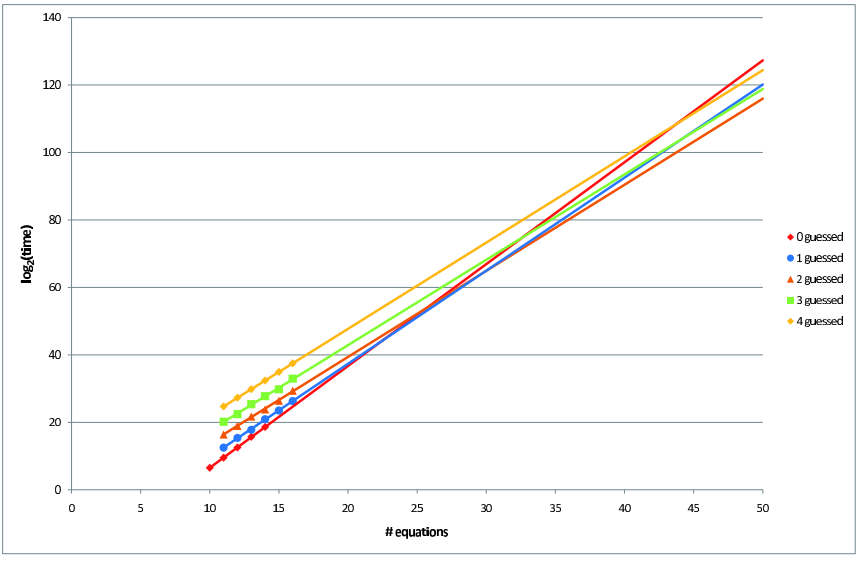

Fig. 5. Running time of the direct attack with Buchberger's algorithm (datapoints and extrapolation)

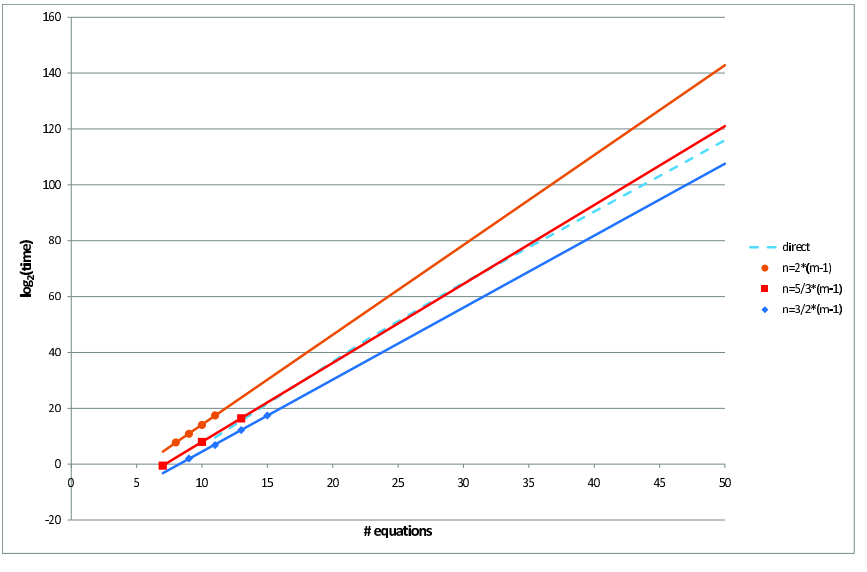

Fig. 6. Running time of the RBS attack with Buchberger's algorithm for different ratios of m and n The dotted line marks the running time of a direct attack against a scheme with m equations

Proof of Proposition 1

Proposition 1: A Rainbow instance over $GF(2^a)$ with parameters $v_1, o_1, \ldots, o_u$ and $n \geq m \geq 10$, for which the items

1. $v_1 \geq \frac{\ell}{a} - 1$
2. $o_u \geq \frac{\ell}{a}$
3. $n - 2 \cdot o_u \geq \frac{\ell}{a} + 1$

hold, has a security level of ℓ bits against the MinRank, the HighRank and the UOV attack.

Proof:

$$C_{\mathrm{MR}}(q,m,n,v_1)=[q^{v_1+1}\cdot m\cdot(n^2/2-m^2/6)]\,\mathrm{m} \overset{1.}{\geq} [2^{a\cdot\ell/a}\cdot m\cdot(n^2/2-m^2/6)]\,\mathrm{m} > 2^{\ell}\,\mathrm{m}$$

$$C_{\mathrm{HR}}(q,n,o_u) = [q^{o_u} n^3/6]\ \mathrm{m} \overset{2.}{\geq} [2^{a\cdot\ell/a}\cdot n^3/6]\ \mathrm{m} > 2^{\ell}\ \mathrm{m}$$

$$C_{\mathrm{UOV}}(q,n,o_u) = [q^{n-2o_u-1}\cdot o_u^4]\ \mathrm{m} \overset{3.}{\geq} [2^{a\cdot\ell/a}\cdot o_u^4]\ \mathrm{m} > 2^{\ell}\ \mathrm{m} \qquad \square$$

Proof of Proposition 2

Proposition 2: By following the above strategy we get not only the minimal m but also the minimal n required for the security of Rainbow in the year y. Hence, the schemes proposed in the table below also minimize the size of the public key.

Proof: It is clear, that m_0 is the minimal number of equations we need for the security of Rainbow in the year y. So, it remains to show that we get with n_0 the minimal required number of variables, too. To show this, we assume that we had another Rainbow instance with m' equations and $n' < n_0$ variables which also fulfills our security level.
Since m_0 was defined to be the minimal number of equations such that the complexity of the direct attack lies above our security level, we have $m' \geq m_0$. In the following, we distinguish between two cases:

Case 1: $m' = m_0$: Since n_0 was defined by

$$n_0 = \min\{n | C_{\mathrm{RBS}}(q, m_0, n) \geq \text{Security level}\}$$

we have $n' \geq n_0$ which contradicts our assumption.

Case 2: $m' > m_0$: Here we assume that we had a Rainbow instance with $m' > m_0$ equations and $n' < n_0$ variables which fulfills our security level. Such a Rainbow Scheme leads via the first part of the RBS attack to an overdetermined system of $m'+n'-1$ equations in n' variables, which has a complexity to solve of

about $C_{MQ(q,m'+n'-1,n')}$, which lies (as we assume) above the security margin. Thus we have

$$\text{Security margin} \leq C_{MQ(q,m'+n'-1,n')} \overset{m'>m_0}{\leq} C_{MQ(q,m_0+n'-1,n')}$$

So we have found a Rainbow instance with m_0 equations and $n' < n_0$ variables which fulfills our security level. This is a contradiction to Case 1.

As a consequence, the strategy described in Section 5 minimizes both the number m of equations and the number n of variables.
Since the public key size of Rainbow is given as (see formula (1))

$$\text{size(public key)} = m \cdot \frac{(n+1)\cdot(n+2)}{2} \text{ byte},$$

the strategy also minimizes the public key size. □

Table 8. Proposed parameters (for $K = GF(2^8)$, $t = 18$, $b = 10$ and $r = 18$), optimized for small public key size

	Rainbow example scheme				Sym-	RSA Key Size	Elliptic Curve	Elliptic Curve	Infeasible
		public key		private key	metric	and SDL	Key Size	Key Size	number of
Year	$(m,\ n)$	size (kB)	(v_1, o_1, o_2)	size (kB)	Key Size	Field Size	$c = 0$	$c = 18$	MIPS years
1982					56	417 288	105	85	$5.00 \cdot 10^5$
2010	(26,43)	25.7	(17,13,13)	19.1	78	1369 1056	146	160	$1.45 \cdot 10^{12}$
2011	(27,45)	29.2	(18,13,14)	21.7	79	1416 1088	148	163	$2.47 \cdot 10^{12}$
2012	(27,45)	29.2	(18,13,14)	21.7	80	1464 1120	149	165	$4.19 \cdot 10^{12}$
2013	(28,46)	31.6	(18,14,14)	23.1	80	1513 1184	151	168	$7.14 \cdot 10^{12}$
2014	(29,47)	34.1	(18,14,15)	24.8	81	1562 1216	152	172	$1.21 \cdot 10^{13}$
2015	(29,47)	34.1	(18,14,15)	24.8	82	1613 1248	154	173	$2.07 \cdot 10^{13}$
2016	(30,49)	38.3	(19,15,15)	27.7	83	1664 1312	155	177	$3.51 \cdot 10^{13}$
2017	(30,51)	41.3	(21,15,15)	30.5	83	1717 1344	157	180	$5.98 \cdot 10^{13}$
2018	(31,52)	44.4	(21,15,16)	32.4	84	1771 1376	158	181	$1.02 \cdot 10^{14}$
2019	(31,52)	44.4	(21,15,16)	32.4	85	1825 1440	160	185	$1.73 \cdot 10^{14}$
2020	(32,53)	47.3	(21,16,16)	34.4	86	1881 1472	161	188	$2.94 \cdot 10^{14}$
2021	(33,54)	50.8	(21,16,17)	36.5	86	1937 1536	163	190	$5.01 \cdot 10^{14}$
2022	(33,55)	52.7	(22,16,17)	38.1	87	1995 1568	164	193	$8.52 \cdot 10^{14}$
2023	(34,57)	58.2	(23,17,17)	42.0	88	2054 1632	166	197	$1.45 \cdot 10^{15}$
2024	(34,58)	60.2	(24,17,17)	43.8	89	2113 1696	167	198	$2.47 \cdot 10^{15}$
2025	(35,59)	64.1	(24,17,18)	46.3	89	2174 1728	169	202	$4.20 \cdot 10^{15}$
2026	(35,59)	64.1	(24,17,18)	46.3	90	2236 1792	170	205	$7.14 \cdot 10^{15}$
2027	(36,60)	68.1	(24,18,18)	48.7	91	2299 1856	172	207	$1.21 \cdot 10^{16}$
2028	(37,61)	72.3	(24,18,19)	51.4	92	2362 1888	173	210	$2.07 \cdot 10^{16}$
2029	(37,63)	77.0	(26,18,19)	55.6	93	2427 1952	175	213	$3.52 \cdot 10^{16}$
2030	(38,65)	84.0	(27,19,19)	60.5	93	2493 2016	176	215	$5.98 \cdot 10^{16}$
2031	(38,65)	84.0	(27,19,19)	60.5	94	2560 2080	178	219	$1.02 \cdot 10^{17}$
2032	(39,66)	88.8	(27,19,20)	63.6	95	2629 2144	179	222	$1.73 \cdot 10^{17}$
2033	(39,66)	88.8	(27,19,20)	63.6	96	2698 2208	181	224	$2.95 \cdot 10^{17}$
2034	(40,68)	96.7	(28,20,20)	69.1	96	2768 2272	182	227	$5.01 \cdot 10^{17}$
2035	(40,69)	99.4	(29,20,20)	71.6	97	2839 2336	184	229	$8.53 \cdot 10^{17}$
2036	(41,72)	110.7	(31,20,21)	80.3	98	2912 2400	185	232	$1.45 \cdot 10^{18}$
2037	(42,73)	116.6	(31,21,21)	83.8	99	2986 2464	187	236	$2.47 \cdot 10^{18}$
2038	(42,73)	116.6	(31,21,21)	83.8	99	3061 2528	188	239	$4.20 \cdot 10^{18}$
2039	(43,74)	122.6	(31,21,22)	87.7	100	3137 2592	190	241	$7.14 \cdot 10^{18}$
2040	(43,74)	122.6	(31,21,22)	87.7	101	3214 2656	191	244	$1.22 \cdot 10^{19}$
2041	(44,76)	132.1	(32,22,22)	94.4	102	3292 2720	193	246	$2.07 \cdot 10^{19}$
2042	(44,78)	139.0	(32,22,22)	94.4	103	3371 2784	194	248	$3.52 \cdot 10^{19}$
2043	(45,79)	145.8	(34,22,23)	104.8	103	3451 2880	196	251	$5.99 \cdot 10^{19}$
2044	(46,80)	152.8	(34,23,23)	109.1	104	3533 2944	197	255	$1.02 \cdot 10^{20}$
2045	(46,80)	152.8	(34,23,23)	109.1	105	3616 3008	199	258	$1.73 \cdot 10^{20}$
2046	(47,81)	159.9	(34,23,24)	113.6	106	3700 3072	200	260	$2.95 \cdot 10^{20}$
2047	(47,82)	163.8	(35,23,24)	117.1	106	3785 3168	102	262	$5.02 \cdot 10^{20}$
2048	(48,84)	175.4	(36,24,24)	125.2	107	3871 3232	203	265	$8.53 \cdot 10^{20}$
2049	(48,85)	179.6	(37,24,24)	128.8	108	3958 3296	105	269	$1.45 \cdot 10^{21}$
2050	(49,85)	183.3	(36,24,25)	130.2	109	4047 3392	206	272	$2.47 \cdot 10^{21}$