Design Principles for HFEv- based Multivariate Signature Schemes

Albrecht Petzoldt[1], Ming-Shing Chen[2,3], Bo-Yin Yang[2], Chengdong Tao[4], Jintai Ding[5]

[1] Technische Universität Darmstadt, Germany
[2] Academia Sinica, Taiwan
[3] National University, Taiwan
[4] South China University of Technology, China
[5] ChongQing University, China and University of Cincinatti, Ohio, USA
`correponding author: Jintai Ding <jintai.ding@gmail.com>`

Abstract. The Hidden Field Equations (HFE) Cryptosystem as proposed by Patarin is one of the best known and most studied multivariate schemes. While the security of the basic scheme appeared to be very weak, the HFEv- variant seems to be a good candidate for digital signature schemes on the basis of multivariate polynomials. However, the currently existing scheme of this type, the QUARTZ signature scheme, is hardly used in practice because of its poor efficiency. In this paper we analyze recent results from Ding and Yang about the degree of regularity of HFEv- systems and derive from them design principles for signature schemes of the HFEv- type. Based on these results we propose the new HFEv- based signature scheme Gui, which is more than 100 times faster than QUARTZ and therefore highly comparable with classical signature schemes such as RSA and ECDSA.

Keywords: Multivariate Cryptography, Digital Signatures, HFEv-, Design Principles, Security, Performance

1 Introduction

Cryptographic techniques are an essential tool to guarantee the security of communication in modern society. Today, the security of nearly all of the cryptographic schemes used in practice is based on number theoretic problems such as factoring large integers and solving discrete logarithms. The best known schemes in this area are RSA [29], DSA [20] and ECC. However, schemes like these will become insecure as soon as large enough quantum computers arrive. The reason for this is Shor's algorithm [30], which solves number theoretic problems like integer factorization and discrete logarithms in polynomial time on a quantum computer. Therefore, one needs alternatives to those classical public key schemes, based on hard mathematical problems not affected by quantum computer attacks.

Besides lattice, code and hash based cryptosystems, multivariate cryptography is one of the main candidates for this [1]. Multivariate schemes are in general very fast and require only modest computational resources, which makes them attractive for the use on low cost devices like smart cards and RFID chips [6,7]. Additionally, at least in the area of digital signatures, there exists a large number of practical multivariate schemes [11,21].
In 2001, Patarin and Courtois proposed a multivariate signature scheme called QUARTZ [25], which is based on the concept of HFEv-. While QUARTZ produces very short signatures (128 bit), the signature generation process is very slow (at the time about 11 seconds per signature [7]). The main reason for this is the use of a high degree HFE polynomial (for QUARTZ this degree is given by $D = 129$), which makes the inversion of the central map very costly.
At the time of the design of the QUARTZ scheme, very little was known about the complexity of algebraic attacks against the HFE family of systems, in particular, the HFEv- schemes. Therefore, the authors of QUARTZ could not base their parameter choice on theoretical foundations. Recently, there has been a fundamental breakthrough in terms of understanding the behavior of algebraic attacks on the HFE family of systems [10,12], which gives an upper bound on the degree of regularity of Gröbner basis attacks against those schemes.
In this paper, we review and analyze the results of Ding and Yang and derive from these results design criteria for HFEv- based signature schemes. In particular we show that we can, by increasing the numbers a of Minus equations and v of Vinegar variables, achieve adequate security even for low degree HFE polynomials and that the upper bound on the degree of regularity given by Ding and Yang is reasonably tight. Based on our analysis, we propose the new HFEv-based signature scheme Gui [6], which uses HFE polynomials of very low degree, namely $D \in \{5, 9, 17\}$. This enables us to speed up the signature generation process by a factor of more than 100 compared to QUARTZ, without weakening the security of the scheme. By doing so, we create a highly practical multivariate signature scheme, whose performance is comparable to that of classical signature schemes such as RSA and ECDSA.
The rest of this paper is organized as follows. In Section 2 we give an introduction into the area of multivariate cryptography and in particular Big-Field signature schemes. Section 3 introduces the HFEv- signature scheme and the changes made to this scheme by Patarin and Courtois when defining QUARTZ. Furthermore, in this section, we discuss the performance and the security of HFEv- based signature schemes. In Section 4 we analyze the results of Ding and Yang on the behaviour of direct attacks on HFEv- schemes by performing a large number of experiments and present the design criteria we derive from that. Based on these principles, we propose in Section 5 our new multivariate signature scheme Gui. Section 6 gives details on the implementation of the scheme and compares the efficiency of Gui with that of some standard signature schemes. Finally, Section 7 concludes the paper.

[6] We call our new scheme Gui, referring to earthenware pottery dating back to the 4000-year-old Longshan culture [32].

2 Multivariate Cryptography

The basic objects of multivariate cryptography are systems of multivariate quadratic polynomials (see equation (1)).

$$\begin{aligned}
p^{(1)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(1)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(1)} \cdot x_i + p_0^{(1)} \\
p^{(2)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(2)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(2)} \cdot x_i + p_0^{(2)} \\
&\vdots \\
p^{(m)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(m)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(m)} \cdot x_i + p_0^{(m)} \qquad (1)
\end{aligned}$$

The security of multivariate schemes is based on the

MQ Problem: Given m multivariate quadratic polynomials $p^{(1)}(\mathbf{x}),\ldots,p^{(m)}(\mathbf{x})$ in n variables $x_1,\ldots,x_n$ as shown in equation (1), find a vector $\bar{\mathbf{x}} = (\bar{x}_1,\ldots,\bar{x}_n)$ such that $p^{(1)}(\bar{\mathbf{x}}) = \ldots = p^{(m)}(\bar{\mathbf{x}}) = 0$.

The MQ problem (for $m \approx n$) is proven to be NP-hard even for quadratic polynomials over the field GF(2) [16].

To build a public key cryptosystem based on the MQ problem, one starts with an easily invertible quadratic map $\mathcal{F} : \mathbb{F}^n \to \mathbb{F}^m$ (central map). To hide the structure of $\mathcal{F}$ in the public key, one composes it with two invertible affine (or linear) maps $\mathcal{S} : \mathbb{F}^m \to \mathbb{F}^m$ and $\mathcal{T} : \mathbb{F}^n \to \mathbb{F}^n$. The *public key* is therefore given by $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$. The *private key* consists of $\mathcal{S}$, $\mathcal{F}$ and $\mathcal{T}$ and therefore allows to invert the public key.

Note: Due to the above construction, the security of multivariate schemes is not only based on the MQ-Problem but also on the EIP-Problem ("Extended Isomorphism of Polynomials") of finding the composition of $\mathcal{P}$.

In this paper we concentrate on multivariate signature schemes of the BigField family. For this type of multivariate schemes, the map $\mathcal{F}$ is a specially chosen easily invertible map over a degree n extension field $\mathbb{E}$ of $\mathbb{F}$. One uses an isomorphism $\Phi : \mathbb{F}^n \to \mathbb{E}$ to transform $\mathcal{F}$ into a quadratic map

$$\bar{\mathcal{F}} = \Phi^{-1} \circ \mathcal{F} \circ \Phi \qquad (2)$$

from $\mathbb{F}^n$ to itself. The public key of the scheme is therefore given by

$$\mathcal{P} = \mathcal{S} \circ \bar{\mathcal{F}} \circ \mathcal{T} = \mathcal{S} \circ \Phi^{-1} \circ \mathcal{F} \circ \Phi \circ \mathcal{T} : \mathbb{F}^n \to \mathbb{F}^n. \qquad (3)$$

The standard signature generation and verification process of a multivariate BigField scheme works as shown in Figure 1.

Signature Generation

$$\begin{tikzcd}
 & X \in \mathbb{E} \arrow[r, "\mathcal{F}^{-1}"] & Y \in \mathbb{E} \arrow[d, "\Phi^{-1}"] & \\
\mathbf{h} \in \mathbb{F}^n \arrow[r, "\mathcal{S}^{-1}"] & \mathbf{x} \in \mathbb{F}^n \arrow[u, "\Phi"'] \arrow[r, "\bar{\mathcal{F}}^{-1}"] & \mathbf{y} \in \mathbb{F}^n \arrow[r, "\mathcal{T}^{-1}"] & \mathbf{z} \in \mathbb{F}^n \arrow[lll, bend left, "\mathcal{P}"]
\end{tikzcd}$$

Signature Verification

Fig. 1. General workflow of multivariate BigField signature schemes

Signature generation: To generate a signature for a message $\mathbf{h} \in \mathbb{F}^n$, one computes recursively $\mathbf{x} = \mathcal{S}^{-1}(\mathbf{h}) \in \mathbb{F}^n$, $X = \Phi(\mathbf{x}) \in \mathbb{E}$, $Y = \mathcal{F}^{-1}(X) \in \mathbb{E}$, $\mathbf{y} = \Phi^{-1}(Y) \in \mathbb{F}^n$ and $\mathbf{z} = \mathcal{T}^{-1}(\mathbf{y})$. The signature of the message $\mathbf{h}$ is $\mathbf{z} \in \mathbb{F}^n$.

Verification: To check the authenticity of a signature $\mathbf{z} \in \mathbb{F}^n$, one simply computes $\mathbf{h}' = \mathcal{P}(\mathbf{z}) \in \mathbb{F}^n$. If $\mathbf{h}' = \mathbf{h}$ holds, the signature is accepted, otherwise rejected.

A good overview on existing multivariate schemes can be found in [9].

Two widely used variations of multivariate BigField signature schemes are the Minus variation and the use of additional (Vinegar) variables.

Minus variation: The idea of this variation is to remove a small number of equations from the public key. The Minus-Variation was first used in schemes like SFLASH [26] to prevent Patarins Linearization Equations attack [27] against the Matsumoto-Imai cryptosystem [24].

Vinegar variation: In this variation one parametrizes the central map $\mathcal{F}$ by adding (a small set of) additional (Vinegar) variables. In the context of multivariate BigField signature schemes, the Vinegar variation can be used to increase the security of the scheme against direct and rank attacks.

3 The HFEv- Signature Scheme

In this section we introduce the HFEv- signature scheme, which is the basis of both QUARTZ and our new signature scheme Gui (see Section 5).
Let $\mathbb{F} = \mathbb{F}_q$ be a finite field with q elements and $\mathbb{E}$ be a degree n extension field of $\mathbb{F}$. Furthermore, we choose integers D, a and v. Let Φ be the canonical isomorphism between $\mathbb{F}^n$ and $\mathbb{E}$, i.e.

$$\Phi(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i \cdot X^{i-1}. \tag{4}$$

The central map $\mathcal{F}$ of the HFEv- scheme is a map from $\mathbb{E} \times \mathbb{F}^v$ to $\mathbb{E}$ of the form

$$\begin{aligned}\mathcal{F}(X) = &\sum_{0\le i\le j}^{q^i+q^j\le D} \alpha_{ij} \cdot X^{q^i+q^j} \\ &+ \sum_{i=0}^{q^i\le D} \beta_i(v_1, \ldots, v_v) \cdot X^{q^i} \\ &+ \gamma(v_1, \ldots, v_v),\end{aligned} \tag{5}$$

with $\alpha_{ij} \in \mathbb{E}$, $\beta_i : \mathbb{F}^v \to \mathbb{E}$ being linear and $\gamma : \mathbb{F}^v \to \mathbb{E}$ being a quadratic function.
Due to the special form of $\mathcal{F}$, the map $\bar{\mathcal{F}} = \Phi^{-1} \circ \mathcal{F} \circ \Phi$ is a quadratic polynomial map from $\mathbb{F}^{n+v}$ to $\mathbb{F}^n$. To hide the structure of $\bar{\mathcal{F}}$ in the public key, one combines it with two affine (or linear) maps $\mathcal{S} : \mathbb{F}^n \to \mathbb{F}^{n-a}$ and $\mathcal{T} : \mathbb{F}^{n+v} \to \mathbb{F}^{n+v}$ of maximal rank.

The *public key* of the scheme is the composed map $\mathcal{P} = \mathcal{S} \circ \bar{\mathcal{F}} \circ \mathcal{T} : \mathbb{F}^{n+v} \to \mathbb{F}^{n-a}$, the *private key* consists of $\mathcal{S}$, $\mathcal{F}$ and $\mathcal{T}$.

Signature generation: To generate a signature for a message $\mathbf{h} \in \mathbb{F}^{n-a}$, the signer performs the following three steps.

1. Compute a preimage $\mathbf{x} \in \mathbb{F}^n$ of $\mathbf{h}$ under the affine map $\mathcal{S}$.
2. Lift $\mathbf{x}$ to the extension field $\mathbb{E}$ (using the isomorphism Φ). Denote the result by X.
   Choose random values for the vinegar variables $v_1, \ldots, v_v \in \mathbb{F}$ and compute $\mathcal{F}_V = \mathcal{F}(v_1, \ldots, v_v)$.
   Solve the univariate polynomial equation $\mathcal{F}_V(Y) = X$ by Berlekamp's algorithm and compute $\mathbf{y}' = \Phi^{-1}(Y) \in \mathbb{F}^n$.
   Set $\mathbf{y} = (\mathbf{y}'||v_1||\ldots||v_v)$.
3. Compute the signature $\mathbf{z} \in \mathbb{F}^{n+v}$ by $\mathbf{z} = \mathcal{T}^{-1}(\mathbf{y})$.

Signature verification: To check the authenticity of a signature $\mathbf{z} \in \mathbb{F}^{n+v}$, one simply computes $\mathbf{h}' = \mathcal{P}(\mathbf{z}) \in \mathbb{F}^{n-a}$. If $\mathbf{h}' = \mathbf{h}$ holds, the signature is accepted, otherwise rejected.

3.1 QUARTZ

In 2001, Patarin and Courtois proposed the multivariate signature scheme QUARTZ [25], which is based on the concept of HFEv-. Indeed, the public and private maps of QUARTZ are HFEv- maps with the parameters

$$(\mathbb{F}, n, D, a, v) = (\mathrm{GF}(2), 103, 129, 3, 4).$$

Due to this choice, the public key $\mathcal{P}$ of QUARTZ is a quadratic map from $\mathbb{F}^{107}$ to $\mathbb{F}^{100}$. The public key size of QUARTZ is 71 kB, the private key size 3 kB.

The input length of QUARTZ is only $n - a = 100$ bit. Therefore, it is possible for an attacker to use a birthday attack to find two different messages m_1 and m_2 which map to the same input value $\mathbf{h} \in \mathbb{F}^{100}$ and therefore to the same signature.
To prevent this kind of attack, Patarin and Courtois developed a special procedure for the signature generation process of QUARTZ. Roughly spoken, one computes four HFEv- signatures (for the messages $\mathbf{h}$, $\mathcal{H}(\mathbf{h}||0x00)$, $\mathcal{H}(\mathbf{h}||0x01)$ and $\mathcal{H}(\mathbf{h}||0x02)$) and combines them to a single 128 bit signature of the message $\mathbf{h}$. Analogously, during the signature verification process, one has to use the public key $\mathcal{P}$ four times.

3.2 Performance

The most costly step during the signature generation process of HFEv- based signature schemes such as QUARTZ is the inversion of the univariate polynomial equation $\mathcal{F}_V$ over the extension field $\mathbb{E}$. This step is usually performed by Berlekamp's algorithm, whose complexity can be estimated by [28]

$$\mathcal{O}(D^3 + n \cdot D^2). \tag{6}$$

As can be seen from equation (6), the complexity of inverting $\mathcal{F}_V$ and therefore of the signature generation process of HFEv- based schemes is mainly determined by the degree D of the HFE polynomial. Due to the high degree of the HFE polynomial used in QUARTZ, the inversion of $\mathcal{F}_V$ is very costly. Furthermore, we have to perform this step four times during the signature generation of QUARTZ.
Additionally, the design of QUARTZ requires the central equation $\mathcal{F}_V(Y) = X$ to have a unique root. Since, after choosing random values for Minus equations and Vinegar variables, $\mathcal{F}_V$ can be seen as a random function, this happens with probability about $\frac{1}{e}$. Altogether, we therefore have to run Berlekamp's algorithm about $4 \cdot e$ times during the signature generation process of QUARTZ. Thus, the QUARTZ signature scheme is rather slow and it takes about 11 seconds to generate a signature [7].

3.3 Security of HFEv- based schemes

The most important attacks against signature schemes of the HFEv- type are

- the MinRank attack and
- direct algebraic attacks.

The MinRank attack on HFE In this paragraph we describe the attack of Kipnis and Shamir [22] against the HFE cryptosystem. For the simplicity of our description we restrict ourselves to homogeneous maps $\mathcal{F}$ and $\mathcal{P}$.
The key idea of the attack is to lift the maps $\mathcal{S}$, $\mathcal{T}$ and $\mathcal{P}$ to functions $\mathcal{S}^\star$, $\mathcal{T}^\star$ and $\mathcal{P}^\star$ over the extension field $\mathbb{E}$. Since $\mathcal{S}$ and $\mathcal{T}$ are linear maps, $\mathcal{S}^\star$ and $\mathcal{T}^\star$ have the form

$$\mathcal{S}^\star(X) = \sum_{i=1}^{n-1} s_i \cdot X^{q^i} \quad \text{and} \quad \mathcal{T}^\star(X) = \sum_{i=1}^{n-1} t_i \cdot X^{q^i}, \tag{7}$$

with coefficients s_i and $t_i \in \mathbb{E}$. The function $\mathcal{P}^\star$ can be expressed as

$$\mathcal{P}^\star(X) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} p^\star_{ij} X^{q^i+q^j} = \underline{X} \cdot P^\star \cdot \underline{X}^T, \tag{8}$$

where $P^\star = [p^\star_{ij}]$ and $\underline{X} = (X^{q^0}, X^{q^1}, \dots, X^{q^{n-1}})$. Due to the relation $\mathcal{P}^\star(X) = \mathcal{S}^\star \circ \mathcal{F} \circ \mathcal{T}^\star(X)$ we get $\mathcal{S}^{\star\ -1} \circ \mathcal{P}^\star(X) = \mathcal{F} \circ \mathcal{T}^\star(X)$ and

$$\tilde{P} = \sum_{k=0}^{n-1} s_k \cdot G^{\star k} = W \cdot F \cdot W^T \tag{9}$$

with $g^{\star\ k}_{ij} = (p^\star{}_{i-k \bmod n, j-k \bmod n})^{q^k}$, $w_{ij} = s^{q^i}_{j-i \bmod n}$ and F being the $n \times n$ matrix representing the central map $\mathcal{F}$. Note that, due to the special structure of $\mathcal{F}$, the only non zero entries in the matrix F are located in the upper left $r \times r$ submatrix ($r = \lfloor \log_q D - 1 \rfloor + 1$).
Therefore, the rank of the matrix $W \cdot F \cdot W^T$ is less or equal to r, which means that we can determine the coefficients s_k of equation (9) by solving an instance of the MinRank problem.
In the setting of HFEv-, the rank of this matrix is, for odd characteristic, bounded from above by [12]

$$\text{Rank}(\widetilde{P}) \leq r + a + v. \tag{10}$$

Under the assumption that the vinegar maps β_i look like random functions, we find that this bound is tight.
For fields of even characteristic we eventually have to decrease this rank by 1, since over those fields, the matrix $\widetilde{P}$ is always of even rank. The complexity of the MinRank attack against HFEv- based schemes is therefore given roughly by

$$\text{Complexity}_{\text{MinRank}} = \mathcal{O}(q^{n\cdot(r+v+a-1)} \cdot (n-a)^3). \tag{11}$$

In the paper [19] the authors showed that, due to the symmetry of the solutions of the equations for the MinRank problem in the Kipnis-Shamir attack and the fact that we work over a large extension field, the complexity of the Kipnis-Shamir attack is actually exponential in terms of the number of variables

in the HFE system using known MinRank methods, and not polynomial as was originally stated. Though the theoretical argument underlying this observation does not apply directly to the generic MinRank problem, it demonstrates that the Kipnis-Shamir attack, for which one needs to solve a non-generic MinRank problem, has a much higher complexity than originally estimated. We therefore conclude, that the complexity of a MinRank attack against an HFEv- based signature scheme is, in practice, higher than the above estimation.
There is one other formulation of the MinRank problem. According to [14], solving a MinRank problem with $n \times n$ matrices to a rank of r' involves computing a Gröbner basis with degree of regularity $r'(n - r') + 1$, where the rank is given by $r' = r + v + a - 1$. When we raise the rank r' (by increasing $a + v$), this means that the attack complexity of the MinRank attack is much higher than that of a direct attack.

Direct attacks For the HFE family of schemes, the direct attack, namely the attack by directly solving the public equation $\mathcal{P}(\mathbf{x}) = \mathbf{h}$ by an algorithm like XL or a Gröbner basis method such as F_4 [13] is a major concern due to which happened to HFE challenge 1. At the time of the design of QUARTZ, very little was known theoretically about the complexity of algebraic attacks against the HFE family of systems, in particular, the HFEv- schemes. The authors of QUARTZ did not actually give an explanation for their selection of the parameters and therefore the parameter selection of their scheme was not supported by theoretical results. We need to point out that, as has been shown by experiments [23], the public systems of HFEv- based schemes can be solved easier than random systems.
Recently, there has been a fundamental breakthrough in terms of understanding how algebraic attacks on the HFE family of systems work [10,12]. In particular, we now have a solid insight what happens in the case of HFEv-. An upper bound for the degree of regularity of a Gröbner Basis attack against HFEv- systems is given by [12]

$$\mathrm{d}_{\mathrm{reg}} \leq \begin{cases} \frac{(q-1)\cdot(r-1+a+v)}{2} + 2 & q \text{ even and } r+a \text{ odd} \\ \frac{(q-1)\cdot(r+a+v)}{2} + 2 & \text{otherwise} \end{cases}, \tag{12}$$

where r is given by $r = \lfloor \log_q(D-1) \rfloor + 1$.

Note: In [8] Courtois et al. estimated the complexity of a direct attack on QUARTZ by 2^{74} operations. However, they underestimated the degree of regularity of solving an HFEv- system drastically.

4 Design principles for HFEv- based signature schemes

The theoretical breakthrough mentioned in the previous subsection indicates that it might be possible to substantially improve the original design of QUARTZ without reducing the security of the scheme, if we adapt the number of Minus

equations and Vinegar variables in an appropriate way. By reducing the degree of the central HFEv- polynomial we can speed up the operations of Berlekamp's algorithm and therefore the signature generation process of the HFEv- scheme. In this section, we analyze by experiments the behavior of direct attacks against HFEv- schemes and the tightness of the upper bound given by equation (12). From our results we derive design principles for the construction of HFEv- based signature schemes, which we later apply to our new signature scheme Gui presented in the next section.
In particular, we answer in this section the following questions.

1. Equation (12) shows a tradeoff between the degree D of the HFE polynomial and the sum $a + v$ of minus equations and vinegar variables. This would enable us to use low degree HFE polynomials in the construction of HFEv- based signature schemes and therefore to improve their performance drastically. Can we verify this by experiments?
2. Is the ratio between a and v important for the security of the scheme?
3. Is the upper bound on the degree of regularity given by equation (12) reasonably tight?
4. Does it help to guess some variables before applying a Gröbner basis algorithm to the system $\mathcal{P}$ (Hybrid Approach)?

To answer these questions, we performed a large number of experiments with the F_4 algorithm integrated in MAGMA. As we found, adding the field equations $\{x_i^2 - x_i\}$ to the system makes a huge difference regarding the degree of regularity and the running time of the attack.

4.1 Can we use HFE polynomials of low degree D?

To improve the efficiency of the signature generation process we are interested in decreasing the degree of the HFE polynomial in use as far as possible without weakening the security of the scheme. Doing so will reduce the complexity of Berlekamp's algorithm (see equation (6)) and therefore improve the performance of the scheme significantly. So, the first question we have to answer in this context is the following.

How should we choose the degree D of the HFE polynomial in order to obtain secure and efficient HFEv- based schemes?

- $D = 2, 3$: Such small values of D would lead to matrices F of rank 2. We therefore do not think that these schemes can be secure.
- $D = 5$: Although the plain HFE scheme with an HFE polynomial of degree 5 ($r = 3$) is highly insecure, we believe that the modified HFEv- scheme provides adequate security.
- $D = 9, 17$: Other promising values for the degree of the HFE polynomial in use are $D = 9$ and $D = 17$, which lead to values of r of 4 and 5 respectively.

In the first row of experiments we analyzed the behavior of direct attacks against HFEv- systems over GF(2) with different values of D. For this, we fixed the number of equations in the system. For different values of D, a and v we created HFEv- systems and fixed $a + v$ variables randomly to get determined systems. After adding the field equations $\{x_i^2 - x_i\}$ we solved the systems using MAGMA's implementation of the F_4 algorithm. For each parameter set we performed 10 experiments.
Table 1 shows the results of our experiments with determined HFEv- systems of 20 and 25 equations respectively. The degree of regularity of a random system of this size is 5 and 6 respectively. The table shows, for different values of D, the minimal values of a and v needed to reach this degree. Although, because of memory restrictions, we could not perform our experiments for larger values of n, we expect that similar results hold for arbitrary numbers of equations.

		20 equations				25 equations			
D	r	minimal a,v	d_{reg}	time (s)	memory (MB)	minimal a,v	d_{reg}	time (s)	memory(MB)
129	8	$a = v = 0$	5	2.74	109.7	$a = v = 1$	6	276.2	7,621
65	7	$a = 0,\ v = 1$	5	2.73	110.2	$a = v = 2$	6	276.0	7,681
33	6	$a = v = 1$	5	2.75	109.7	$a = 2,\ v = 3$	6	273.4	7,762
17	5	$a = 1,\ v = 2$	5	2.72	109.7	$a = v = 3$	6	275.7	7,751
9	4	$a = v = 2$	5	2.73	109.9	$a = 3,\ v = 4$	6	276.4	7,693
5	3	$a = 2,\ v = 3$	5	2.73	109.6	$a = v = 4$	6	272.8	7,680
random system			5	2.85	110.8		6	286.3	7,683

Table 1. Experiments with F_4 on determined HFEv- systems with 20 and 25 equations

From the above experiments we obtain the following important observation

> Let d be the degree of regularity of a direct attack against an HFEv- system with parameters D_1, n, a_1, v_1 and let $D_2 < D_1$.
> By choosing large enough values for a_2 and v_2, we can obtain an HFEv- scheme with parameters D_2, n, a_2, v_2, such that the degree of regularity of a direct attack against this system is d, too.

From this observation we derive our first design principle for the construction of HFEv- based signature schemes.

Design Principle 1:
For the construction of HFEv- based signature schemes we use for efficiency reasons HFE polynomials of small degree D, namely $D \in \{5, 9, 17\}$. We then increase the numbers of Minus equations a and Vinegar variables v to obtain a secure scheme.

4.2 Is the ratio between a and v important for the security of the scheme?

To answer this question, we performed experiments of the following type. For a fixed degree D of the HFE polynomial, a fixed number of equations and a fixed value s we created HFEv- systems with $a \in \{0, \ldots, s\}$ and $v = s - a$. After fixing $v + a$ variables to get a determined system and adding the field equations $\{x_i^2 - x_i\}$ we solved the systems by the F_4 algorithm integrated in MAGMA. For each parameter set we performed 10 experiments. The results are shown in Table 2 and 3.

D=5, a+v=5					D=9, a+v=4					D=17, a+v=3				
a	v	d_{reg}	time (s)	memory (MB)	a	v	d_{reg}	time (s)	memory (MB)	a	v	d_{reg}	time (s)	memory (MB)
0	5	5	2.76	109.7	0	4	5	2.77	109.7	0	3	5	2.75	110.7
1	4	5	2.77	109.7	1	3	5	2.78	110.8	1	2	5	2.77	109.7
2	3	5	2.76	110.7	2	2	5	2.76	110.7	2	1	5	2.74	110.8
3	2	5	2.77	110.8	3	1	5	2.75	110.8	3	0	5	2.73	109.7
4	1	5	2.75	109.8	4	0	5	2.79	108.7	—				
5	0	**4**	**1.01**	**32.6**	—					—				

Table 2. Experiments with F_4 on determined HFEv- systems with 20 equations

D=5, a+v=8					D=9, a+v=7					D=17, a+v=6				
a	v	d_{reg}	time (s)	memory (MB)	a	v	d_{reg}	time (s)	memory (MB)	a	v	d_{reg}	time (s)	memory (MB)
0	8	6	246.6	7,582	0	7	6	248.9	7,582	0	6	6	247.0	7,581
1	7	6	246.2	7,579	1	6	6	247.4	7,582	1	5	6	247.6	7,581
2	6	6	246.6	7,580	2	5	6	248.0	7,580	2	4	6	247.6	7,581
3	5	6	248.1	7,581	3	4	6	246.4	7,593	3	3	6	248.3	7,579
4	4	6	247.1	7,581	4	3	6	248.3	7,578	4	2	6	246.5	7,580
5	3	6	248.3	7,582	5	2	6	248.5	7,579	5	1	6	248.8	7,580
6	2	6	248.3	7,554	6	1	6	247.3	7,581	6	0	6	247.9	7,581
7	1	**5**	**99.3**	**1,317**	7	0	**5**	**99.5**	**1,380**	—				
8	0	**5**	**88.3**	**1,509**	—					—				

Table 3. Experiments with F_4 on determined HFEv- systems with 25 equations

As the tables show, in particular for HFEv- schemes with low degree D, the number v of vinegar variables should not be too small. Especially, $v = 0$ (i.e. HFE-) seems to be a bad choice.
On the other hand, very high values of v do not increase the security of the scheme and increase the public key size of the scheme drastically. To achieve a good security and a moderate public key size, we therefore formulate our second design principle for HFEv- based signature schemes as follows.

Design Principle 2:
In the design of HFEv- based signature schemes we choose the number of Minus equations a and the number of Vinegar variables v to be as equal as possible, i.e. $v - a \leq 1$.

4.3 Is the upper bound on d_{reg} given by equation (12) reasonably tight?

In this section we check by experiments if the upper bound on the degree of regularity given by equation (12) is tight. Due to memory restrictions, we can show the tightness of equation (12) only for some small values of D, a and v. However, for all values of D used in our scheme Gui ($D \in \{5, 9, 17\}$) we could find parameter sets for which the bound (12) is tight (see Table 4).

D	a	v	upper bound for d_{reg} (12)	d_{reg} (experimental)	
5	0	0	3	3	for $n \geq 10$
	1	1	4	4	for $n \geq 23$
9	0	1	4	4	for $n \geq 23$
	1	1	4	4	for $n \geq 21$
17	0	0	4	4	for $n \geq 15$
	0	1	4	4	for $n \geq 12$

Table 4. Parameter sets, for which the upper bound (12) is tight

For most of the other parameter sets, we missed the upper bound on the degree of regularity given by equation (12) only by 1. We believe that, by increasing the number of equations in the systems, it would be possible to reach the upper bound for arbitrary values of (D, a, v). However, due to memory restrictions, we could not perform experiments with more than 38 equations.

Furthermore, as shown in Table 4.3, we could, for all of the proposed values of D, reach a degree of regularity of at least 7. These results are the basis of our parameter choice for Gui (see Section 5).

D	a	v	d_{reg} (experimental)		upper bound for d_{reg} (12)
5	6	6	7	for $n \geq 38$	9
9	5	5	7	for $n \geq 37$	8
17	4	4	7	for $n \geq 37$	8

Table 5. Parameter sets which lead to $d_{reg} \geq 7$

4.4 Does it help to guess some variables before applying a Gröbner basis algorithm?

In the case of multivariate signature schemes such as HFEv- the public key $\mathcal{P}$ is an underdetermined system of quadratic equations. In our case this system consists of $n-a$ quadratic equations in $n+v$ variables. For the experiments presented in the previous subsections we fixed $a+v$ of the variables of the system to create a determined system before applying the F_4 algorithm.
However, for some multivariate systems, it is a good strategy to guess some additional variables before applying the Gröbner basis algorithm (Hybrid Approach [5]). The goal of this strategy is to create overdetermined systems which hopefully will be significantly easier to solve. When guessing k variables one has, to find a solution of the original system, to solve q^k instances of the simplified system, where q is the cardinality of the underlying field. To check whether this Hybrid approach helps to solve the public systems of the HFEv- scheme faster, we performed a number of experiments. For the three parameter sets $(D, a, v) \in \{(5, 6, 6), (9, 5, 5), (17, 4, 4)\}$ and varying numbers of n and k we created HFEv- systems and solved them with the F_4 algorithm integrated in MAGMA. Table 6 shows, for $k \in \{0, \ldots, 5\}$, the minimal value of n needed to reach a degree of regularity of 7.

# k of guessed	minimal value of n to reach $d_{reg} \geq 7$		
variables	D=5,a=v=6	D=9,a=v=5	D=17,a=v=4
0	38	37	37
1	39	38	38
2	40	40	39
3	42	41	41
4	43	43	42
5	44	44	44

Table 6. Experiments on HFEv- systems with the Hybrid Approach

As the table shows, we could, for each of the above parameter sets and each value $k \in \{0, \ldots, 5\}$, create a HFEv- system offering a good level of security, simply by increasing the number of equations in the system. In fact, the degree of regularity of a direct attack against such a system of $n-a$ quadratic equations in $n-a-k$ variables will be at least 7.
We therefore assume that, for large enough n, all the multivariate systems which have to be solved in the course of a direct/hybrid attack against our schemes, will have a degree of regularity of at least 7. This is the basis for our parameter selection presented in the next section.

5 The New Multivariate Signature Scheme Gui

Based on our experiments presented in the previous section we propose three different versions of our HFEv- based signature scheme Gui over the field GF(2):

- Gui-96 with $(n, D, a, v) = (96, 5, 6, 6)$ with 90 equations in 102 variables,
- Gui-95 with $(n, D, a, v) = (95, 9, 5, 5)$ with 90 equations in 100 variables and
- Gui-94 with $(n, D, a, v) = (94, 17, 4, 4)$ with 90 equations in 98 variables.

The complexity of direct attacks against these schemes can be estimated as follows.
According to our experiments (see Table 6), the degree of regularity of the F_4 algorithm (even with the Hybrid Approach) against these schemes will be at least 7.
For the complexity of a direct attack against one of our schemes (with guessing k variables) we have

$$\text{Complexity}_{F_4/F_5} \geq 3 \cdot \tau \cdot T^2, \tag{13}$$

where T is the number T of top-level monomials in the solving step of the F_4 algorithm and τ is the number of non zero elements in each equation. We get

$$\begin{aligned}
\text{Compl}_{F_4/F_5} \geq 3 \cdot \tau \cdot T^2 &= 2^k \cdot 3 \cdot \binom{n-a-k}{2} \cdot \binom{n-a-k}{d_{reg}}^2 \\
&= 3 \cdot \binom{n-a}{2} \cdot \binom{n-a}{d_{reg}}^2 \\
&\quad \underbrace{\cdot 2^k \cdot \frac{(n-a-k)\cdot(n-a-k-1)}{(n-a)\cdot(n-a-1)} \cdot \left(\frac{(n-a-k)\cdot \ldots \cdot (n-a-k-d_{reg}+1)}{(n-a)\cdot \ldots \cdot (n-a-d_{reg}+1)}\right)^2}_{\geq 1} \\
&\geq 3 \cdot \binom{n-a}{2} \cdot \binom{n-a}{d_{reg}}^2 \geq 3 \cdot \binom{90}{7} \cdot \binom{90}{2} = 2^{80.7}.
\end{aligned} \tag{14}$$

Note that this number is very optimistic since we assume that the degree of regularity will not rise above 7.

Additionally, for better comparison to standard signature schemes, we propose a fourth version of Gui, Gui-127, with the parameters $(n, D, a, v) = (127, 9, 4, 6)$, providing a security level of 120 bits.

5.1 Signature Generation

The central component of the signature generation process of Gui is inverting the HFEv- core map.
To compute a pre-image of a $(n-a)$ bit digest $\mathbf{h}$, one first has to choose random values for the Minus equations and the Vinegar variables. In our concrete implementation, these values are the last $a+v$ bits of SHA-256($\mathbf{h}$). After that, one computes recursively $\mathbf{x} = \mathcal{S}^{-1}(\mathbf{h})$, $X = \Phi(x)$, $Y = \mathcal{F}_V^{-1}(X)$,

$\mathbf{y} = (\Phi^{-1}(Y)||v_1||\ldots||v_v)$ and $\mathbf{z} = \mathcal{T}^{-1}(\mathbf{y})$ (see Figure 2).

For the parameters of Gui, the length of the digest $\mathbf{h}$ is only $n - a = 90$ bits. To prevent birthday attacks, we therefore have to perform the above process several times (for different values of $\mathbf{h}$). We denote this repetition factor by k and set $k = 3$ for Gui-96 and Gui-95. For Gui-94 and Gui-127 the value k is chosen to be 4.

The signature generation process of Gui works as shown in Algorithm 1 and Figure 3.
We initialize the $n - a$ vector S_0 to be $\mathbf{0}$ and compute the SHA-256 hash value $\mathbf{h}$ of the message. Let D_1 be the bitstring consisting of the first $(n - a)$ bits of $\mathbf{h}$. We compute the pre-image of D_1 under the HFEv- core (see above) and split the result into an $(n - a)$ bit string S_1 and an $a + v$ bit string X_1.
We set D_2 to be the string consisting of the first $(n - a)$ bits of SHA-256($\mathbf{h}$) and compute the HFEv- pre-image of $D_2 \oplus S_1$. Again, the result is split into the two parts S_2 ($n - a$ bits) and X_2 ($a + v$ bits). This process is repeated, until we have values S_i, X_i for $i = 1, \ldots, k$.

The final signature of the message is given by $\sigma = (S_k||X_k||\ldots||X_1)$. The resulting signature sizes for our schemes can be found in Table 7.

A detailed description, how the inversion of the central HFEv- map is performed in our implementation, can be found in Section 6.2. Due to some flaws in the SHA-1 algorithm, we replace the SHA-1 hash function used in the original QUARTZ design by SHA-256.

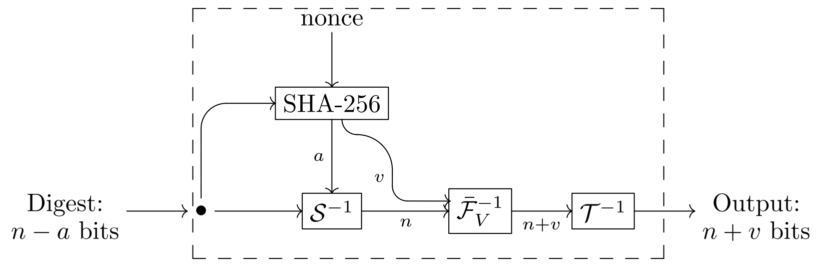

Fig. 2. Core operations of HFEv-

Algorithm 1 Signature Generation Process of Gui

```
Input: Gui private key $(\mathcal{S}, \mathcal{F}, \mathcal{T})$ message $\mathbf{d}$, repetition factor $k$
Output: signature $\sigma \in \mathrm{GF}(2)^{(n-a)+k(a+v)}$
 1: $\mathbf{h} \leftarrow$ SHA-256($\mathbf{d}$)
 2: $S_0 \leftarrow \mathbf{0} \in \mathrm{GF}(2)^{n-a}$
 3: for $i = 1$ to $k$ do
 4:     $D_i \leftarrow$ first $n-a$ bits of $\mathbf{h}$
 5:     $(S_i, X_i) \leftarrow \mathrm{HFEv-}^{-1}(D_i \oplus S_{i-1})$
 6:     $\mathbf{h} \leftarrow$ SHA-256($\mathbf{h}$)
 7: end for
 8: $\sigma \leftarrow (S_k || X_k || \ldots || X_1)$
 9: return $\sigma$
```

Algorithm 2 Signature Verification Process of Gui

```
Input: Gui public key $\mathcal{P}$, message $\mathbf{d}$, repetition factor $k$, signature $\sigma \in$
    $\mathrm{GF}(2)^{(n-a)+k(a+v)}$
Output: TRUE or FALSE
 1: $\mathbf{h} \leftarrow$ SHA-256($\mathbf{d}$)
 2: $(S_k, X_k, \ldots, X_1) \leftarrow \sigma$
 3: for $i = 1$ to $k$ do
 4:     $D_i \leftarrow$ first $n-a$ bits of $\mathbf{h}$
 5:     $\mathbf{h} \leftarrow$ SHA-256($\mathbf{h}$)
 6: end for
 7: for $i = k-1$ to 0 do
 8:     $S_i \leftarrow \mathcal{P}(S_{i+1} || X_{i+1}) \oplus D_{i+1}$
 9: end for
10: if $S_0 = \mathbf{0}$ then
11:     return TRUE
12: else
13:     return FALSE
14: end if
```

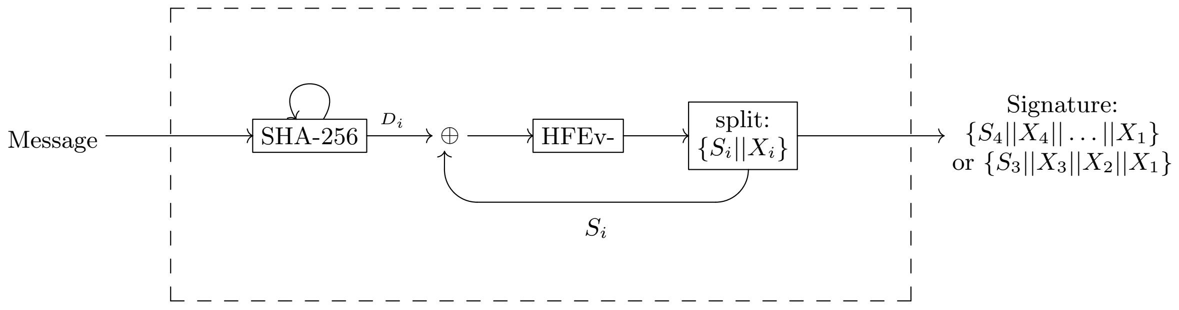
Message
SHA-256
D_i
$\oplus$
HFEv-
split: $\{S_i || X_i\}$
S_i
Signature: $\{S_4 || X_4 || \ldots || X_1\}$ or $\{S_3 || X_3 || X_2 || X_1\}$

Fig. 3. Signature Generation Process of Gui

5.2 Signature Verification

To check the authenticity of a signature $\sigma \in \mathrm{GF}(2)^{(n-a)+k(a+v)}$ we parse σ into $S_k, X_k, \ldots, X_1$ and compute $D_1, \ldots, D_k$ as shown in Section 5.1. For $i = k-1$ to 0 we compute recursively $S_i = \mathcal{P}(S_{i+1}||X_{i+1}) \oplus D_{i+1}$. The signature is accepted, if and only if $S_0 = \mathbf{0}$ holds.

By the above construction of the signature generation and verification process we prevent birthday attacks as follows. We consider an adversary A who wants to find two messages m_1 and m_2 which lead to the same signature σ.
For the plain HFEv- signature scheme it would be enough to find two messages m_1 and m_2 such that SHA-256$(m_1)_i$=SHA-256$(m_2)_i$ for the first $n-a$ bits. If $(n-a) \leq 160$, the adversary can find m_1 and m_2 by a birthday attack.
In the context of our scheme Gui, the adversary now has to find messages m_1 and m_2 which lead to the same values of $D_1, \ldots, D_k$. For our values of the repetition factor k, this corresponds to finding a collision for a hash function of length 270, 360 and 492 bit (Gui-95/96, Gui-94 and Gui-127 respectively). This is, in general, assumed to be infeasible.

scheme	core map HFEv-(n,D,a,v)	public key size (byte)	private key size (byte)	repetition factor k	signature size (bit)
Gui-96	(96,5,6,6)	63036	3175	3	126
Gui-95	(95,9,5,5)	60600	3053	3	120
Gui-94	(94,17,4,4)	58212	2943	4	122
Gui-127	(127,9,4,6)	142576	5350	4	163
QUARTZ	(103,129,3,4)	75515	3774	4	128

Table 7. Key and signature sizes of Gui-94,Gui-95,Gui-96, and Gui-127

6 Implementation and Comparison

In this section we present the details of our implementation of the Gui signature scheme and compare the performance of our scheme with that of the original QUARTZ and other standard signature schemes.

6.1 Arithmetics over Finite Fields

The first step in our implementation of the Gui signature scheme is to provide efficient arithmetics over the large binary fields in use. To speed up these computations, we use a set of new processor instructions for carry-less multiplication: `PCLMULQDQ` [31].

The instruction set PCLMULQDQ allows the efficient multiplication of two 64-bit polynomials over GF(2) resulting in an 128-bit polynomial. The PCLMULQDQ instructions are available on some new processors of Intel and AMD. Performance data of PCLMULQDQ can be found in Table 8. In the case of Gui, the extension

processor type		latency cycles	throughput cycles/multiplication
Intel	Sandy Bridge	14	8
	Ivy Bridge	14	8
	Hashwell	7	2
AMD	Bulldozer	12	7
	Piledriver	12	7
	Steamroller	11	7

Table 8. Performance of PCLMULQDQ on different platforms (source: [18], [15])

field $\mathbb{E}$ has less than 2^{128} elements. We represent an element of the field $\mathbb{E}$ as a polynomial over GF(2) which can be divided into two 64-bit polynomials.

A multiplication over the large field $\mathbb{E}$ is divided into two phases, namely a *multiplication* and a *reduction* phase.

In the *multiplication phase*, the multiplication of two 128-bit polynomials can be performed by 4 calls of PCLMULQDQ. With the help of the Karatsuba algorithm, we can avoid one call of PCLMULQDQ and therefore its long latency (see Table 8). To square an element of $\mathbb{E}$, we need only two calls of PCLMULQDQ since we are operating over a field of characteristic 2.

The *reduction phase* of the field multiplication heavily depends on the field representation. For the original QUARTZ scheme over the field $GF(2^{103})$ the authors used $GF(2^{103}) := GF(2)[x]/(x^{103} + x^9 + 1)$ [25]. For Gui, we choose the field representations

- $GF(2^{94}) := GF(2)[x]/(x^{94} + x^{21} + 1)$,
- $GF(2^{95}) := GF(2)[x]/(x^{95} + x^{11} + 1)$,
- $GF(2^{96}) := GF(2)[x]/(x^{96} + x^{10} + x^9 + x^6 + 1)$ and
- $GF(2^{127}) := GF(2)[x]/(x^{127} + x + 1)$ respectively.

The baseline for the reduction phase is two calls of PCLMULQDQ since, after the multiplication phase, the degree of the polynomial will be greater than 2×64. The irreducible polynomials above are chosen to contain only few terms of low degree. With few terms in the irreducible polynomials, we may replace the use of PCLMULQDQ by a few logic shifts and XOR instructions.

In the $GF(2^{127})$ case, for example, the reduction can be performed by only two 128-bit shifts for the x^{128} part and one conditional `XOR` for the x^{127} term, avoiding at least two calls of `PCLMULQDQ` while reducing the high 128 bit register.

Another technique is to represent elements as 128-bit polynomials while avoiding full reduction. This allows us to perform the reduction of degree 128-191 and 192-255 terms using only two calls of `PCLMULQDQ` without data dependency. In the GF(2^{96}) case, for example, we can perform the reduction phase by multiplying the degree 128-191 terms by $x^{128} = x^{42} + x^{41} + x^{38} + x^{32}$ and the degree 192-255 terms with $x^{192} = x^{20} + x^{18} + x^{12} + 1$. All the polynomials in use have degree ≤ 64, and we can perform the reduction by two calls of `PCLMULQDQ`.
The proposed implementation provides time-constant multiplication for preventing side channel leakage, since, regardless of the input, the same operations are performed. The same strategy is also applied to the calculation of multiplicative inverses. For example, for the sake of time-constant arithmetics, the inverse of an element $x \in$ GF(2^{127}) is calculated by raising x to $x^{2^{127}-2}$ instead of the faster extended Euclidean algorithm.

6.2 Inverting the HFEv- core

In this section we describe how we can perform the inversion of the central HFEv- equation $\mathcal{F}_V(Y) = X$ efficiently. During the signature generation process of Gui we have to perform this step several times to avoid birthday attacks (see Section 5.1). Therefore it is extremely important to perform this step efficiently.

To invert the central HFEv- equation, we have to perform Berlekamp's algorithm to find the roots of the polynomial $\mathcal{F}_V(Y) - X$. Since the design of QUARTZ and Gui requires $\mathcal{F}_V(Y) - X$ to have a unique solution, we only have to perform the first step of Berlekamp's algorithm, i.e. the computation of

$$\gcd(\mathcal{F}_V(Y) - X, Y^{2^n} - Y). \tag{15}$$

We have

$$\begin{aligned} &\gcd(\mathcal{F}_V(Y) - X, Y^{2^n} - Y) \\ &= \gcd(\mathcal{F}_V(Y) - X, \prod_{i \in \mathbb{F}_{2^n}, i \neq 0} (Y - i)) = \prod_{i : \mathcal{F}_V(i) = X} (Y - i). \end{aligned}$$

Therefore the main process in creating a signature consists in computing $\gcd(\mathcal{F}_V(Y) - X, Y^{2^n} - Y)$. The number of roots of $\mathcal{F}_V(Y) - X$ (as well as the only solution when that happens) can obviously be read off from the result.

Probability of a unique root Every time we choose the values of Minus equations and Vinegar variables, we basically pick a random central equation $\mathcal{F}_V(Y) - X = 0$. The probability of this equation having a unique solution is about $1/e$. Therefore, in order to invert the HFEv- central equation, we have to perform the gcd computation about e times.
The repeated computation of the gcd (see equation 15) is probably the most detectable

side channel leakage of our scheme. However, there are no known side channel attacks on big field schemes or HFEv- which use the information that one particular equation in the big field has no, respectively two or more solutions.

How do we optimize the computation of the gcd? The main computation consumption in this step comes from the division of the extreme high power polynomial $Y^{2^n} - Y \mod \mathcal{F}_V(Y)$. A naive long division is unacceptable for this purpose due to its slow reduction phase. Instead of this, we choose to recursively raise the lower degree polynomial Y^{2^m} to the power of 2.

$$\begin{aligned}&(Y^{2^m} \mod \mathcal{F}_V(Y))^2 \mod \mathcal{F}_V(Y)\\ =&\,(\sum_{i<2^m} b_i Y^i)^2 \mod \mathcal{F}_V(Y) = (\sum_{i<2^m} b_i^2 Y^{2i}) \mod \mathcal{F}_V(Y)\end{aligned}$$

By multiplying Y to the naive relation $Y^D = \sum_{0\le i\le j, 2^i+2^j<D} a_{ij} Y^{2^i+2^j}$, we can prepare a table for $Y^{2i} \mod \mathcal{F}_V(Y)$ first. The rest of computation of the raising process is to square all the coefficients b_i in $Y^{2^m} \mod \mathcal{F}_V(Y)$ and multiply them to the Y^{2i}s in the table.
Although the starting relation $\mathcal{F}_V(Y) = Y^D + \sum_{0\le i\le j, 2^i+2^j<D} a_{ij} Y^{2^i+2^j}$ is a sparse polynomial, the polynomials become dense quickly in the course of the raising process. However, the number of terms in the polynomials is restricted by D because of mod $\mathcal{F}_V(Y)$. We expect the number of terms to be in average D during the computation.
We implemented Berlekamp's algorithm in such a way that it takes the same number of iterations in the main GCD loop and the same number of operations in the big field for each run at very low cost. Therefore it runs, independently from the input, at constant time.
The number of field multiplications needed to compute the Y^{2i} table is $O(2\cdot D^2)$. To raise Y^{2^m} to Y^{2^n} we need $O((n-m)\cdot D)$ squarings and $O((n-m)\cdot D^2)$ multiplications. We can further reduce the number of computations needed for raising Y^{2^m} by using a higher degree Y^i table. For example, if we raise Y^{2^m} to $Y^{2^{4m}}$ in one step, we need only $O((n-m)\cdot D)$ squarings and $O(\frac{(n-m)}{2}\cdot D^2)$ multiplications. However, the computational effort for preparing the Y^i table increases. Table 9 shows the time needed to compute $\gcd(Y^{2^n} - Y, \mathcal{F}(Y))$ on three different CPUs.

6.3 Experiments and Comparison

Table 10 shows key and signature sizes as well as the running times of signature generation and verification of Gui and compares these data with those of some standard signature schemes. The data are benchmarked according to specifications given by the eBACS project [3].

scheme	security level (bit)	public key size (kB)	private key size (kB)	time needed for inverting $\mathcal{F}$ (kilo-cycles)
HFEv-(96,5,6,6)	80	61.6	3.1	72/76/55
HFEv- (95,9,5,5)	80	59.2	3.0	159 / 135 /79
HFEv- (94,17,4,4)	80	56.8	2.9	533 / 453/274
HFEv- (127,9,4,6)	120	139.2	5.2	170 / 156/128
HFEv- (103,129,3,4)	80	71.9	3.1	25,793 / 20,784/12,630

[1] AMD Opteron 6212, 2.5 GHz (Bulldozer)/ Intel Xeon CPU E5-2620, 2.0 GHz (Sandy Bridge) / Intel Xeon E3-1245 v3, 3.4 GHz (Hashwell)

Table 9. Key sizes of HFEv- schemes and running time of gcd$(X^{2^n} - X, \mathcal{F}(\mathcal{X}))$

scheme	security level(bits)	public key size (Bytes)	private key size (Bytes)	signature size(bits)	signing time (k-cycles) [1]	verification time (k-cycles) [1]
Gui-96 (96,5,6,6)	80	63,036	3,175	126	603/569/238	97/70/62
Gui-95 (95,9,5,5)	80	60,600	3,053	120	1,417/1,441/602	91/60/58
Gui-94 (94,17,4,4)	80	58,212	2,943	124	5,800/5,480/2,495	118/74/71
Gui-127 (127,9,4,6)	120	142,576	5,350	163	2,368/2,183/1,080	220/121/122
QUARTZ (103,129,3,4)	80	73,626	3,174	128	302,882/315,716/128,736	145/84/86
RSA-1024	80	128	128	128	2,080/1,058/1,073	74/32/33
RSA-2048	112	256	256	256	8,834/5,347/4,625	138/76/61
ECDSA P160	80	40	60	320	1,283/558/588	1,448/635/652
ECDSA P192	96	48	72	384	1,513/773/697	1,715/867/779
ECDSA P256	128	64	96	512	830/388/342	2,111/920/816

[1] AMD Opteron 6212, 2.5 GHz (Bulldozer) / Intel Xeon CPU E5-2620, 2.0 GHz (Sandy Bridge) / Intel Xeon E3-1245 v3, 3.4 GHz (Hashwell)

Table 10. Comparison between Gui and standard signature schemes

We should note that the timings for Gui given by Table 10 are for C programs with a few intrinsic function calls of `PCLMULQDQ`. The PKCs benchmarked in the eBACs project also do not represent optimal implementations of RSA and ECC. We present these numbers in an effort to compare apples to apples by using only reference implementations.

6.4 Platforms without `PCLMULQDQ`

We also optimized the arithmetics over large finite fields by SIMD table-lookup instructions for platforms without `PCLMULQDQ`. SIMD table-lookup instructions are common in contemporary CPUs, e.g., `PSHUFB` (Packed Shuffle Byte) on x86 and `VTBL` (Vector Table Lookup) on ARM platforms. For this we used an ARM Cortex-A9 processor with NEON instruction set, which is currently the most common version in smart phones. We use `PSHUFB` and `VTBL` as general table-lookup instructions with 4-bit index (although `VTBL` is capable of 5-bit indices), not necessarily restricted to the x86 platform. Since the length of indices in these instructions is only 4 or 5 bit, PSHUFB and VTBL were so far only applied to implementations over small fields, e.g., GF(16) and GF(256) [7]. For applying PSHUFB and VTBL to large finite fields, we have to represent the large field as an extension of the small field. In our case, we use for the implementation the following representation of GF(2^{96})

- GF(16) := $GF(2)[y]/(y^4 + y + 1)$,
- GF(2^{96}) := $GF(16)[x]/(x^{24} + y^3x^3 + x + y)$.

The multiplication in GF(16) is performed with PSHUFB and the multiplication in GF(2^{96}) corresponds to a polynomial multiplication over GF(16). Furthermore, we use Karatsuba's technique for the computation of coefficients in different registers. To prevent the scheme from side channel leakage, we implement the multiplication in GF(16) with logarithm/exponential tables instead of multiplication tables, except for the multiplication with fixed values in the reduction phase of the polynomial multiplication. With logarithm tables, the multiplication in GF(16) is performed by an addition in the exponents of a multiplicative generator and therefore consists of two table lookups, addition and reduction. Although there is only one table lookup in a normal implementation of multiplication tables, an intentional cache miss would result in a time difference since the tables are loaded with the values of input operands.
Performance data of these implementations as well as a reference implementation of school book multiplication for 64-bit variables, which is applicable to all platforms without these SIMD instructions, can be found in Table 11. Note that, in the sixth row of Table 11, we use the field GF(2^{128}) := $GF(16)[x]/(x^{32}+x^3+x+y)$ instead of GF(2^{127}), since the shape of this field contains some restrictions in the extension from GF(16).

	implementation	multiplication	square	inversion
GF(2^{96})	64-bit variables,school book	624/3392 [1]	624/3384	68,752/357,728
	128-bit register, PSHUFB/VTBL	138/731	87/424	11,242/ 48,825
	128-bit register, PCLMULQDQ	12/-	8/-	2,489/-
GF(2^{127})	64-bit variables,school book	743/4,009	735/3,997	105,235/546,881
	128-bit register, PSHUFB/VTBL [2]	318/813	187/ 531	28,565/77,703
	128-bit register, PCLMULQDQ	15/-	9/-	3,257/-

[1] Intel Xeon E3-1245 v3, 3.4 GHz (Hashwell)/Xilinx Zynq 7020, 667 MHz (ARM Cortex-A9)
[2] GF(2^{128})

Table 11. Average number of cycles for the arithmetics in GF(2^{96}) and GF(2^{127}) for various implementations.

Our ARM implementation takes on average 1,14 ms to invert the central map $\mathcal{F}_V(Y) = X$. Performance data for the implementation of Gui-96 on ARM platforms can be found in Table 12. As one can see, we are able to generate about 300 Gui signatures per second on a standard smart phone.

scheme	Key Generation	Signature Generation	Signature Verification
Gui-96(96,5,6,6)	99,555	3,291	102

Table 12. Performance data for Gui on ARM platforms (timings in 10^{-6} s)

6.5 Grover's Algorithm and Potential Extension to Larger Fields

By Grover's algorithm [17] it might be possible to cut down the complexity of a brute-force search in an n-bit space to $O(2^{n/2})$. We believe that this is no major threat to HFEv- and in particular to Gui because of the large number of quantum bits (qubits) needed in this case: While we need only 1024 qubits to solve Discrete Logarithms on a 256-bit prime modulus elliptic curve and 6000 qubits to factorize 3000-bit RSA numbers using Shor's Algorithm, the number of qubits and quantum gates needed to attack Gui by Grover's algorithm is in the order of a million (n^3), since it implies the evaluation of n quadratic polynomials in n variables. Therefore, quantum algorithms can be used much more easy for the cryptanalysis of schemes such as RSA and ECC than for that of multivariate schemes such as Gui and we do not consider Grover's algorithm to be a major problem for our scheme. However, even if we have to take Grover's algorithm into account, there is an easy way to prevent this kind of attack, namely by choosing the parameter n about twice as large while keeping all other parameters constant. In the implementation, this means an extra layer of the Karatsuba algorithm in the multiplication phase and therefore a factor of 3 slowdown. Furthermore, this increases the public key size by a factor of 8.

7 Conclusion and Future Work

In this paper, we analyzed the behavior of direct attacks against HFEv- based signature schemes. Experiments show that, even for low degree HFE polynomials in use, we can obtain adequate security levels by increasing the numbers a and v of Minus equations and Vinegar variables. Furthermore we find that the upper bound on the degree of regularity proposed by Ding and Yang in [12] is relatively tight. From our results we derive design principles for the construction of HFEv- based signature schemes, which lead to both secure and efficient schemes. We apply these principles to the construction of our new HFEv- based signature scheme Gui, which is more than 100 times faster than the original QUARTZ scheme. Furthermore we show that the performance of our scheme is highly comparable to that of standard signature schemes, including signatures on elliptic curves.
As future work we want to analyze the influence of the numbers a of Minus Equations and v of Vinegar variables on the security of HFEv- schemes further. Furthermore we plan to create for every common existing platform an optimal implementation of HFEv- (Gui) and compare it with some of the best optimized code for ECC and RSA, such as Ed25519 [2]. Another approach would be to verify such optimal Gui code for formal correctness. In short, we believe that there is still much work to be done on the HFEv- digital signature schemes.

Acknowledgements

We thank the anonymous reviewers of Asiacrypt for their comments which helped to improve the paper. Especially we want to thank the shepherd of our paper for his valuable advice. Due to this we included

- Further remarks on the complexity of the Kipnis-Shamir attack on HFE and its variants (Section 3.3).
- Additional experiments on the effect of the parameters a and v on the security of our scheme and the Hybrid approach (Section 4.2 and 4.4).
- Remarks on side channel leakage and countermeasures (Section 6.1 and 6.2).
- Implementation details of Gui on ARM platforms (Section 6.4).
- Remarks on how Grover's algorithm might affect our parameter choice (Section 6.5).

We would like to thank for partial support from the Charles Phelps Taft Research Center, the Center for Advanced Security Research Darmstadt (CASED), ECSPRIDE, Academia Sinica, the CAS/SAFEA International Partnership Program for Creative Research Teams, Taiwan's Ministry of Science and Technology, National Taiwan University and Intel Corporation under grands NIST 60NAN15D059, NSFC 61472054, MOST 103-2911-I-002-001, NTU-ICRP-104R7501 and NTU-ICRP-104R7501-1.

References

1. D.J. Bernstein, J. Buchmann, E. Dahmen (eds.): Post Quantum Cryptography. Springer, 2009.
2. D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, B.-Y. Yang: High-speed high-security signatures. Journal of Cryptographic Engineering 2:2, pp. 77-89 (2012).
3. D.J. Bernstein, T. Lange (eds.): eBACS: ECRYPT Benchmarking of Cryptographic Systems. http://bench.cr.yp.to, accessed 14 May 2014.
4. L. Bettale, J.C. Faugère, L. Perret: Cryptanalysis of HFE, multi-HFE and variants for odd and even characteristic. Des. Codes Cryptography 69:1 pp. 1–52 (2013).
5. L. Bettale, J.C. Faugère, L. Perret: Hybrid approach for solving multivariate systems over finite fields. Journal of Mathematical Cryptology 3, pp. 177 - 197 (2009).
6. A. Bogdanov, T. Eisenbarth, A. Rupp, C. Wolf: Time-area optimized public-key engines: MQ-cryptosystems as replacement for elliptic curves? CHES 2008, LNCS vol. 5154, pp. 45-61. Springer, 2008.
7. A.I.T. Chen, M.-S. Chen, T.-R. Chen, C.-M. Cheng, J. Ding, E. L.-H. Kuo, F. Y.-S. Lee, B.-Y. Yang: SSE implementation of multivariate PKCs on modern x86 CPUs. CHES 2009, LNCS vol. 5747, pp. 33 - 48. Springer, 2009.
8. N.T. Courtois, M. Daum, P. Felke: On the Security of HFE, HFEv- and QUARTZ. PKC 2003, LNCS vol. 2567, pp. 337 - 350. Springer 2003.
9. J. Ding, J. E. Gower, D. S. Schmidt: Multivariate Public Key Cryptosystems. Springer, 2006.
10. J. Ding, T. Kleinjung: Degree of regularity for HFE-. IACR eprint 2011/570.
11. J. Ding, D. S. Schmidt: Rainbow, a new multivariate polynomial signature scheme. ACNS 2005, LNCS vol. 3531, pp. 164-175. Springer 2005.

12. J. Ding, B.Y. Yang: Degree of Regularity for HFEv and HFEv-. PQCrypto 2013, LNCS vol. 7932, pp. 52-66. Springer, 2013.
13. J.C. Faugère: A new efficient algorithm for computing Gröbner bases (F4). Journal of Pure and Applied Algebra 139, pp. 61-88 (1999).
14. J.C. Faugère, M. Safey el Din, P.J. Spaenlehauer: On the Complexity of the Generalized MinRank Problem. Journal of Symbolic Computation, pp. 30-58, **55:0**(2013)
15. A. Fog: Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs. http://www.agner.org/optimize/ , 7th Dec 2014.
16. M. R. Garey and D. S. Johnson: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company 1979.
17. L.K. Grover: A Fast Quantum Mechanical Algorithm for Database Search. Proceedings of STOC, pp. 212 – 219. ACM, 1996.
18. Intel Corporation: Hashwell Cryptographic Performance. http://www.intel.com/content/dam/www/public/us/en/documents/white-papers/haswell-cryptographic-performance-paper.pdf
19. X. Jiang, J. Ding, L. Hu: Kipnis-Shamir attack on HFE Revisited. Inscrypt, LNCS vol. 4990, pp. 399–411. Springer, 2008.
20. D. Kravitz: Digital Signature Algorithm. US patent 5231668 (July 1991).
21. A. Kipnis, L. Patarin, L. Goubin: Unbalanced Oil and Vinegar Schemes. EUROCRYPT 1999, LNCS vol. 1592, pp. 206–222. Springer 1999.
22. A. Kipnis, A. Shamir: Cryptanalysis of the HFE Public Key Cryptosystem. CRYPTO 99, LNCS vol. 1666, pp. 19 - 30. Springer 1999.
23. M. S. E. Mohamed, J. Ding, J. Buchmann: Towards Algebraic Cryptanalysis of HFE Challenge 2, ISA 2011, Communications in Computer and Information Science vol. 200, pp. 123-131. Springer 2011.
24. T. Matsumoto, H. Imai: Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. EUROCRYPT 1988. LNCS vol. 330, pp. 419-453. Springer 1988.
25. J. Patarin, N. Courtois, L. Goubin: QUARTZ, 128-Bit Long Digital Signatures. CTRSA 2001, LNCS vol. 2020, pp. 282-297. Springer, 2001.
26. J. Patarin, N. Courtois, L. Goubin: Flash, a fast multivariate signature algorithm. CTRSA 2001, LNCS vol. 2020, pp. 298 - 307. Springer, 2001.
27. J. Patarin: Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt 88. CRYPTO 95. LNCS vol. 963, pp. 248 - 261. Springer 1995.
28. C. Richards: Algorithms for Factoring Square-Free Polynomials over Finite Fields. Master Thesis, Simon Fraser University (Canada), 2009.
29. R. L. Rivest, A. Shamir, L. Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Commun. ACM 21 (2), pp. 120-126 (1978).
30. P. Shor: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput. 26 (5), pp. 1484 - 1509 (1997).
31. J. Taverne, A. Faz-Hernàndez, D. F. Aranha, F. Rodrquez-Henrïquez, D. Hankerson, J. López: Software Implementation of Binary Elliptic Curves: Impact of the Carry-Less Multiplier on Scalar Multiplication. CHES 2011, LNCS vol. 6917, pp. 108-123. Springer 2011.
32. http://en.wikipedia.org/wiki/
File: CMOC_Treasures_of_Ancient_China_exhibit_white_pottery_gui_1.jpg.