QUARTZ, 128-bit long digital signatures*

http://www.minrank.org/quartz/

Jacques Patarin, Nicolas Courtois and Louis Goubin

Bull CP8
68 route de Versailles – BP45
78431 Louveciennes Cedex
France
J.Patarin@frlv.bull.fr,courtois@minrank.org, Louis.Goubin@bull.net

Abstract. For some applications of digital signatures the traditional schemes as RSA, DSA or Elliptic Curve schemes, give signature size that are not short enough (with security 2^{80}, the minimal length of these signatures is always ≥ 320 bits, and even ≥ 1024 bits for RSA). In this paper we present a first well defined algorithm and signature scheme, with concrete parameter choice, that gives $128 - bit$ signatures while the best known attack to forge a signature is in 2^{80}. It is based on the basic HFE scheme proposed on Eurocrypt 1996 along with several modifications, such that each of them gives a scheme that is (quite clearly) strictly more secure. The basic HFE has been attacked recently by Shamir and Kipnis (cf [3]) and independently by Courtois (cf this RSA conference) and both these authors give subexponential algorithms that will be impractical for our parameter choices. Moreover our scheme is a modification of HFE for which there is no known attack other that inversion methods close to exhaustive search in practice. Similarly there is no method known, even in theory to distinguish the public key from a random quadratic multivariate function.
QUARTZ is so far the only candidate for a practical signature scheme with length of 128-bits.
QUARTZ has been accepted as a submission to NESSIE (New European Schemes for Signatures, Integrity, and Encryption), a project within the Information Societies Technology (IST) Programme of the European Commission.

1 Introduction

In the present document, we describe the QUARTZ public key signature scheme.

QUARTZ is a HFEV$^-$ algorithm (see [4, 5]) with a special choice of the parameters. QUARTZ belongs to the family of "multivariate" public key schemes, *i.e.* each signature and each hash of the messages to sign are represented by some elements of a small finite field K.

* Part of this work is an output of project "Turbo-signatures", supported by the french Ministry of Research.

QUARTZ is designed to generate very very short signatures: only 128 bits ! Moreover, in QUARTZ, all the state of the art ideas to enforce the security of such an algorithm have been used: QUARTZ is built on a "Basic HFE" scheme secure by itself at present (no practical attack are known for our parameter choice) and, on this underlying scheme, we have introduced some "perturbation operations" such as removing some equations on the originally public key, and introducing some extra variables (these variables are sometime called "vinegar variables"). The resulting schemes look quite complex at first sight, but it can be seen as the resulting actions of many ideas in the same direction: to have a very short signature with maximal security (*i.e.* the "hidden" polynomial F of small degree d is hidden as well as possible).

As a result, the parameters of QUARTZ have been chosen in order to satisfy an extreme property that no other public key scheme has reached so far: very short signatures. QUARTZ has been specially designed for very specific applications because we thought that for all the classical applications of signature schemes, the classical algorithms (RSA, Fiat-Shamir, Elliptic Curves, DSA, etc) are very nice, but they all generate signatures of 320 bits or more (1024 for RSA) with a security in 2^{80}, so it creates a real practical need for algorithms such as QUARTZ.

QUARTZ was designed to have a security level of 2^{80} with the present state of the art in Cryptanalysis.

2 QUARTZ: the basic ideas

(This paragraph is here to help the understanding of QUARTZ. QUARTZ will then be described in details in the next paragraphs.)

Let $K = \mathbf{F}_q = \mathrm{GF}(q)$ be a small finite field (in QUARTZ we will choose $K = \mathbf{F}_2$). Let d and n be two integers (in QUARTZ we will have $d = 129$ and $n = 103$).

Let α_{ij}, $1 \leq i \leq n$, $1 \leq j \leq n$, be some elements of $\mathbf{F}^{q^n}$ such that:

$$\forall i, j,\ 1 \leq i \leq n,\ 1 \leq j \leq n, q^i + q^j > d \Rightarrow \alpha_{ij} = 0.$$

Let β_i, $1 \leq i \leq n$, be some elements of $\mathbf{F}_{q^n}$ such that

$$\forall i,\ 1 \leq i \leq n, q^i > d \Rightarrow \beta_i = 0.$$

Let γ be an element of $\mathbf{F}_{q^n}$.

Now let F be the following function:

$$F : \begin{cases} \mathbf{F}_{q^n} \to \mathbf{F}_{q^n} \\ X \mapsto \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \alpha_{ij} X^{q^i+q^j} + \sum_{i=0}^{n-1} \beta_i X^{q^i} + \gamma \end{cases}$$

This function F can be seen in two different ways:

1. It can be seen as a polynomial function with only one variable $x \in \mathbf{F}_{q^n}$, of degree d.
2. Or, if we write this function F as a function from $\mathbf{F}_{q^n}$ to $\mathbf{F}_{q^n}$ (*i.e.* if we consider $\mathbf{F}_{q^n}$ as a vector space over $\mathbf{F}_q$), it can be seen as a multivariate function of n variables $(x_1, \ldots, x_n) \in K^n$ to n variables $(y_1, \ldots, y_n) \in K^n$ of total degree 2.

   Note: Here the total degree is only 2 because all the functions $X \mapsto X^{q^i}$ are linear functions over $\mathbf{F}_{q^n}$, *i.e.* they can be written as functions from K^n to K^n of total degree one.

From the univariate representation (1) it is possible when d is not too large to invert F (*i.e.* to compute all the roots of $F(X) = Y$ when Y is a given element of $\mathbf{F}_{q^n}$). (Some root finding algorithms exist, such as the Berlekamp algorithm for example, for these univariate algorithms. Their complexity is polynomial in d, so d cannot be too large if we want those algorithms to be efficients.)

From the multivariate representation (2) we will be able to "hide" this function F by introducing two secret bijective affine transformations s and t from K^n to K^n, and we will compute $G' = t \circ F \circ s$, and keep F secret.

This function G' is a quadratic function from K^n to K^n.

Now, two other ideas will be introduced.

Remark: These two other ideas, that we denote by "$-$" and "V", are introduced in order to enforce the security of the scheme, as we will explain in section 8. However, the scheme might be secure even if we did not add these two ideas.

First, we will not publish all the n quadratic equations that define G', but only $n - r$ of these equations ($r = 3$ in the QUARTZ algorithm).

Secondly, we will "mix" the n variables $x_1, \ldots, x_n$ with v "extra variables" ($v = 4$ in the QUARTZ algorithm). These v "extra variables" will be introduced in the β_i and γ parameters. (We will describe in detail in section 4 how this will be done.)

Finally, we obtain a trapdoor one-way function G from 107 bits to 100 bits. Without any secret it is possibe to compute $y = G(x)$ when x is given, and with a secret it is possible to compute all the values of x such that $G(x) = y$ when y is given (x:107 bits, y: 100 bits).

Remark: QUARTZ is a special case of a more general scheme called HFEV$^-$. This scheme is described in [4] and [5]. However, there are many possible parameters in HFEV$^-$, so that we think it is interesting to give an example of the possible choices of these parameters to obtain 128 bit public key digital signatures with 2^{80} security (with the best known attacks).

3 The birthday paradox: how can a digital signature be as short as 128 bits with 2^{80} security

In all signature schemes in which checking the validity of the signature S of a message M consists in verifying an equation $f(S) = g(M)$, where f and g are two public functions, it is always possible, from the birthday paradox, to find a signature S and a message M such that S will be a valid signature of M, after approximately $\sqrt{2^n}$ computations (and storages), where n is the number of bits of the signature and the number of output bits of f and g. (Just store $\sqrt{2^n}$ values $g(M)$, compute $\sqrt{2^n}$ values $f(S)$ and look for a collision).

However, with QUARTZ, we will avoid this "birthday" attack because checking the validity of the signature S of a message M consists in verifying an equation $f(S, M) = 0$, where f is a public function.

Remark If G denotes the trapdoor one-way function from 107 bits to 100 bits that we will use, four computations of this function G will be needed to check whether $f(S, M) = 0$ in the QUARTZ algorithm, as we will see below. A more general theory about how small a digital signature can be, can be found in the extended version of [4], available from the authors (or from our Web page http://www.smartcard.bull.com/sct/uk/partners/bull/index.html).

However, with a signature of only 128 bits, there is still something to be careful with: no more than 2^{64} messages must be signed with the same public key. If more than 2^{64} messages are signed with the same public key, there is a large probability that two different messages will have the same signature and this may create troubles for some applications. However, this is not a very restrictive fact for practical applications since here, only the people who know the secret key can create or avoid this 2^{64} birthday fact. Somebody who does not know the secret key cannot use this fact to create an attack on the signature scheme with 2^{64} complexity.

This explains why in QUARTZ, the best known attacks are in 2^{80}, despite the fact that the length of the signature is only 128 bits.

4 Notations and Parameters of the Algorithm

In all the present document, $||$ will denote the "concatenation" operation. More precisely, if $\lambda = (\lambda_0, \ldots, \lambda_m)$ and $\mu = (\mu_0, \ldots, \mu_n)$ are two strings of bits, then $\lambda||\mu$ denotes the string of bits defined by:

$$\lambda||\mu = (\lambda_0, \ldots, \lambda_m, \mu_0, \ldots, \mu_n).$$

For a given string $\lambda = (\lambda_0, \ldots, \lambda_m)$ of bits and two integers r, s, such that $0 \leq r \leq s \leq m$, we denote by $[\lambda]_{r \to s}$ the string of bits defined by:

$$[\lambda]_{r \to s} = (\lambda_r, \lambda_{r+1}, \ldots, \lambda_{s-1}, \lambda_s).$$

The QUARTZ algorithm uses the field $\mathcal{L} = \mathbf{F}_{2^{103}}$. More precisely, we chose $\mathcal{L} = \mathbf{F}_2[X]/(X^{103} + X^9 + 1)$. We will denote by φ the bijection between $\{0,1\}^{103}$ and $\mathcal{L}$ defined by:

$$\forall \omega = (\omega_0, \ldots, \omega_{102}) \in \{0,1\}^{103},$$
$$\varphi(\omega) = \omega_{102} X^{102} + \ldots + \omega_1 X + \omega_0 \pmod{X^{103} + X^9 + 1}.$$

4.1 Secret parameters

1. An affine secret bijection s from $\{0,1\}^{107}$ to $\{0,1\}^{107}$. Equivalently, this parameter can be described by the 107×107 square matrix and the 107×1 column matrix over $\mathbf{F}_2$ of the transformation s with respect to the canonical basis of $\{0,1\}^{107}$.
2. An affine secret bijection t from $\{0,1\}^{103}$ to $\{0,1\}^{103}$. Equivalently, this parameter can be described by the 103×103 square matrix and the 103×1 column matrix over $\mathbf{F}_2$ of the transformation s with respect to the canonical basis of $\{0,1\}^{103}$.
3. A family of secret functions $(F_V)_{V \in \{0,1\}^4}$ from $\mathcal{L}$ to $\mathcal{L}$, defined by:

$$F_V(Z) = \sum_{\substack{0 \le i < j < 103 \\ 2^i + 2^j \le 129}} \alpha_{i,j} \cdot Z^{2^i + 2^j} + \sum_{\substack{0 \le i < 103 \\ 2^i \le 129}} \beta_i(V) \cdot Z^{2^i} + \gamma(V).$$

   In this formula, each $\alpha_{i,j}$ belongs to $\mathcal{L}$ and each β_i $(0 \le i < 103)$ is an affine transformation from $\{0,1\}^7$ to $\mathcal{L}$, *i.e.* a transformation satisfying

$$\forall V = (V_0, V_1, V_2, V_3) \in \{0,1\}^4,\ \beta_i(V) = \sum_{k=0}^{3} V_k \cdot \xi_{i,k}$$

   with each $\xi_{i,k}$ being an element of $\mathcal{L}$. Finally, γ is a quadratic transformation from $\{0,1\}^7$ to $\mathcal{L}$, *i.e.* a transformation satisfying

$$\forall V = (V_0, V_1, V_2, V_3) \in \{0,1\}^4,\ \gamma(V) = \sum_{k=0}^{3} \sum_{\ell=0}^{3} V_k V_\ell \cdot \eta_{k,\ell}$$

   with each $\eta_{k,\ell}$ being an element of $\mathcal{L}$.
4. A 80-bit secret string denoted by Δ.

4.2 Public parameters

The public key consists in the function G from $\{0,1\}^{107}$ to $\{0,1\}^{100}$ defined by:

$$G(X) = \Big[t\Big(\varphi^{-1}\big(F_{[s(X)]_{103 \to 106}}(\varphi([s(X)]_{0 \to 102})) \big) \Big) \Big]_{0 \to 99}.$$

By construction of the algorithm, G is a quadratic transformation over $\mathbf{F}_2$, *i.e.* $(Y_0, \ldots, Y_{99}) = G(X_0, \ldots, X_{106})$ can be written, equivalently:

$$\begin{cases} Y_0 = P_0(X_0, \ldots, X_{106}) \\ \quad \vdots \\ Y_{99} = P_{99}(X_0, \ldots, X_{106}) \end{cases}$$

with each P_i being a quadratic polynomial of the form

$$P_i(X_0, \ldots, X_{106}) = \sum_{0 \leq j < k < 107} \zeta_{i,j,k} X_j X_k + \sum_{0 \leq j < 107} \nu_{i,j} X_j + \rho_i,$$

all the elements $\zeta_{i,j,k}$, $\nu_{i,j}$ and ρ being in $\mathbf{F}_2$.

5 Signing a message

In the present section, we describe the signature of a message M by the QUARTZ algorithm.

5.1 The signing algorithm

The message M is given by a string of bits. Its signature S is obtained by applying successively the following operations (see figure 1):

1. Let M_1, M_2 and M_3 be the three 160-bit strings defined by:

$$M_1 = \text{SHA-1}(M),$$

$$M_2 = \text{SHA-1}(M_1),$$

$$M_3 = \text{SHA-1}(M_2).$$

2. Let H_1, H_2, H_3 and H_4 be the four 100-bit strings defined by:

$$H_1 = [M_1]_{0 \to 99},$$

$$H_2 = [M_1]_{100 \to 159} || [M_2]_{0 \to 39},$$

$$H_3 = [M_2]_{40 \to 139},$$

$$H_4 = [M_2]_{140 \to 159} || [M_3]_{0 \to 79}.$$

3. Let $\tilde{S}$ be a 100-bit string. $\tilde{S}$ is initialized to $00 \ldots 0$.
4. For $i = 1$ to 4, do
   (a) Let Y be the 100-bit string defined by:

$$Y = H_i \oplus \tilde{S}.$$

   (b) Let W be the 160-bit string defined by:

$$W = \text{SHA-1}(Y || \Delta).$$

   (c) Let R be the 3-bit string defined by:

$$R = [W]_{0 \to 2}.$$

   (d) Let V be the 4-bit string defined by:

$$V = [W]_{3 \to 6}.$$

(e) Let B be the element of $\mathcal{L}$ defined by:

$$B = \varphi\Big(t^{-1}(Y||R)\Big).$$

(f) Consider the following univariate polynomial equation in Z (over $\mathcal{L}$):

$$F_V(Z) = B.$$

- If this equation has a unique solution in $\mathcal{L}$, then let A be this solution.
- Else replace W by SHA-1(W) and go back to (c).

(g) Let X be the 107-bit string defined by:

$$X = s^{-1}\Big(\varphi^{-1}(A)||V\Big).$$

(h) Define the new value of the 100-bit string $\tilde{S}$ by:

$$\tilde{S} = [X]_{0\to 99}\ ;$$

(i) Let X_i be the 7-bit string defined by:

$$X_i = [X]_{100\to 106}.$$

5. The signature S is the 128-bit string given by:

$$S = \tilde{S}||X_4||X_3||X_2||X_1.$$

5.2 Solving the equation $F_V(Z) = B$

To sign a message, we need to solve an equation of the form $F_V(Z) = B$, with B belonging to $\mathcal{L}$ and Z being the unknown, also in $\mathcal{L}$. More precisely, if we refer to step 4.f in section 5.1, we must:

1. Decide whether there is a unique solution or not;
2. In the case of a unique solution, find it.

The following method can be used: we compute the polynomial

$$\Psi(Z) = \gcd\Big(F_V(Z) - B, Z^{2^{103}} - Z\Big).$$

The equation $F_V(Z) = B$ has a number of solutions (in $\mathcal{L}$) equal to the degree of Ψ over $\mathcal{L}$. As a consequence, if Ψ is *not* of degree one, then the number of solutions is *not* one. On the contrary, if Ψ is of degree one, it is of the form $\Psi(Z) = \kappa \cdot (Z - A)$ (with $\kappa \in \mathcal{L}$) and A is the unique solution of the equation $F_V(Z) = B$.

To compute the gcd above, we can first recursively compute $Z^{2^i} \mod (F_V(Z) - B)$ for $i = 0, 1, \ldots, 103$ and then compute $\Theta(Z) = Z^{2^{103}} - Z \mod (F_V(Z) - B)$. Finally $\Psi(Z)$ is easily obtained by

$$\Psi(Z) = \gcd\Big(F_V(Z) - B, \Theta(Z)\Big).$$

Thanks to this method, the degrees of the polynomials involved in the computation never exceed $2 \times 129 = 258$.

Note that more refined methods have also been developed to compute $\Psi(Z)$ (see [2]).

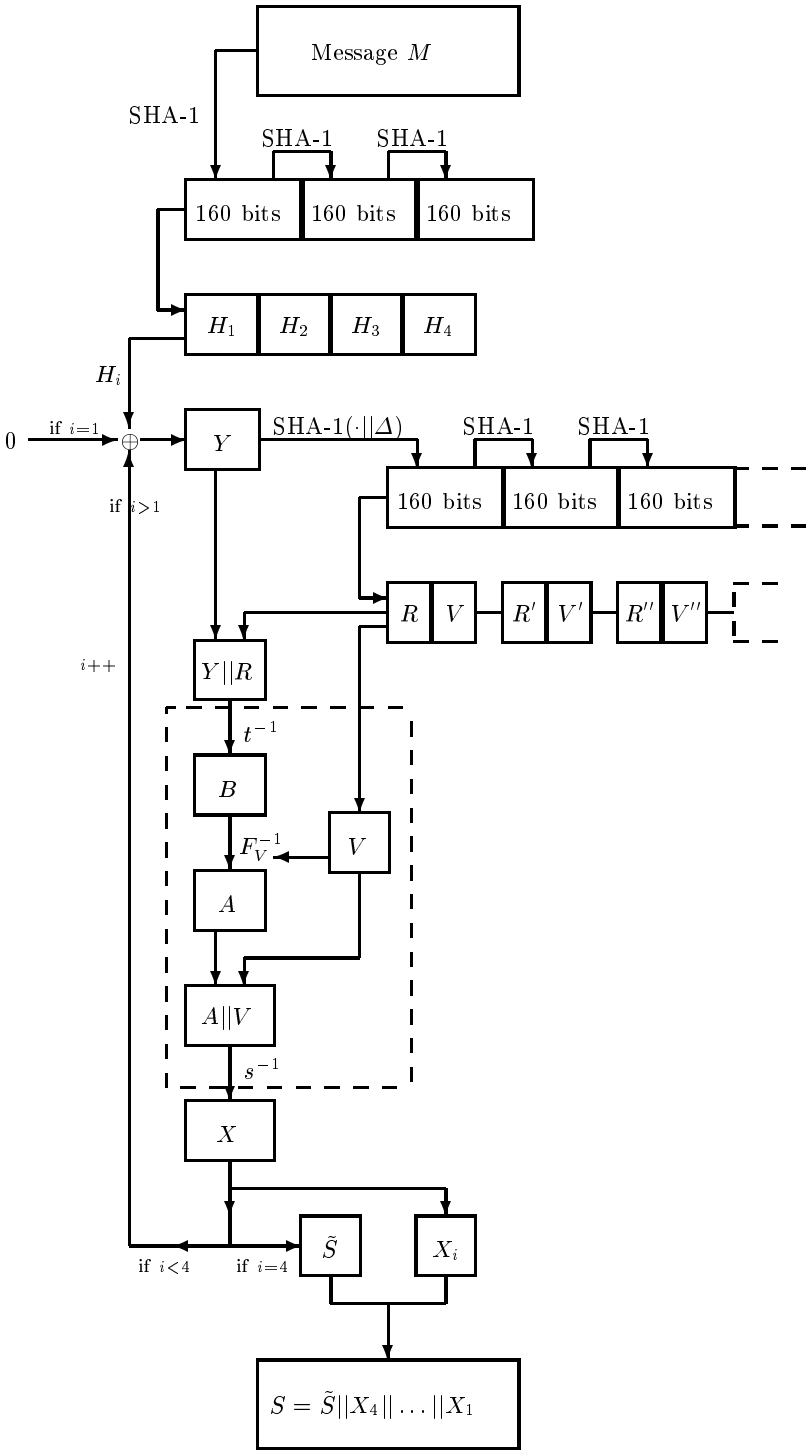

Fig. 1. Signature generation with QUARTZ (beginning with $i = 1$)

5.3 Existence of the signature

The success of the signing algorithm relies on the following fact: for at least one of the successive values of the pair (R, V), there exist a unique solution (in Z) for the equation $F_V(Z) = B$.

It can be proven that, for a randomly chosen B, the probability of having a unique solution in Z is approximately $\frac{1}{e}$. If we suppose that the successive values (R, V) take all the possible values in $\{0,1\}^7$, the probability of never having a unique solution is approximately given by:

$$\left(1 - \frac{1}{e}\right)^{128} \simeq 2^{-85}.$$

Since the signing algorithm has to solve this equation four times, the probability that the algorithm fails is:

$$\mathcal{P} \simeq 1 - \left(1 - \left(1 - \frac{1}{e}\right)^{128}\right)^4 \simeq 2^{-83}.$$

This probability is thus completely negligible.

6 Verifying a signature

Given a message M (*i.e.* a string of bits) and a signature S (a 128-bit string), the following algorithm is used to decide whether S is a valid signature of M or not:

1. Let M_1, M_2 and M_3 be the three 160-bit strings defined by:

$$M_1 = \text{SHA-1}(M),$$

$$M_2 = \text{SHA-1}(M_1),$$

$$M_3 = \text{SHA-1}(M_2).$$

2. Let H_1, H_2, H_3 and H_4 be the four 100-bit strings defined by:

$$H_1 = [M_1]_{0 \to 99},$$

$$H_2 = [M_1]_{100 \to 159} || [M_2]_{0 \to 39},$$

$$H_3 = [M_2]_{40 \to 139},$$

$$H_4 = [M_2]_{140 \to 159} || [M_3]_{0 \to 79}.$$

3. Let $\tilde{S}$ be the 100-bit string defined by:

$$\tilde{S} = [S]_{0 \to 99}.$$

4. Let X_4, X_3, X_2, X_1 be the four 7-bit string defined by:

$$X_4 = [S]_{100 \to 106},$$

$$X_3 = [S]_{107 \to 113},$$

$$X_2 = [S]_{114 \to 120},$$

$$X_1 = [S]_{121 \to 127}.$$

5. Let U be a 100-bit string. U is initialized to $\tilde{S}$.
6. For $i = 4$ down to 1, do
   (a) Let Y be the 100-bit string defined by:

$$Y = G(U||X_i).$$

   (b) Define the new value of the 100-bit string U by:

$$U = Y \oplus H_i.$$

7. – If U is equal to the 100-bit string $00\ldots0$, accept the signature.
   – Else reject the signature.

7 Computation of the G function

The verification algorithm of QUARTZ requires the fast evaluation of the function G, which can be viewed as a set of 100 public quadratic polynomials of the form

$$P_i(x_0, \ldots, x_{106}) = \sum_{0 \le j < k < 107} \zeta_{i,j,k} x_j x_k + \sum_{0 \le j < 107} \nu_{i,j} x_j + \rho_i \qquad (0 \le i \le 99)$$

(see section 3.2).

To perform this computation, three methods can be used:

First method:
We can proceed directly, *i.e.* by successively compute the multiplications and the additions involved in P_i.

Second method:
Each of the P_i can be rewritten as follows:

$$P_i(x_0, \ldots, x_{106}) = x_0 \ell_{i,0}(x_0, \ldots, x_{106}) + x_1 \ell_{i,1}(x_1, \ldots, x_{106}) + \\ + \ldots + x_{106} \ell_{i,106}(x_{106}) + \rho_i,$$

with the $\ell_{i,0}, \ldots, \ell_{i,106}$ $(0 \le i \le 99)$ being 107×100 linear forms that can be explicited. As a result, since each x_j equals 0 or 1, we just have to compute modulo 2 additions of x_j variables.

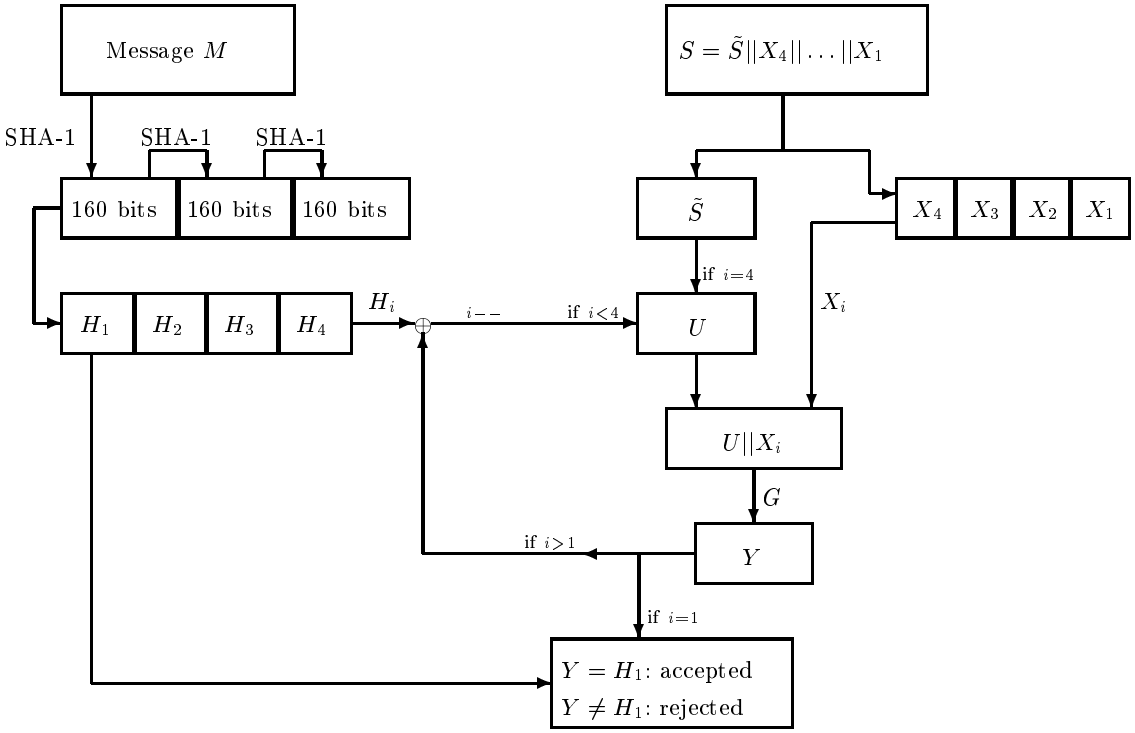

Fig. 2. Signature verification with QUARTZ (beginning with $i = 4$)

Third method:
Another possible technique consists in writing

$$G(x_0, \ldots, x_{106}) = \sum_{0 \le j < k < 107} x_j x_k \cdot Z_{j,k} \oplus \sum_{0 \le j < 107} x_j \cdot N_j \oplus R$$

with

$$Z_{j,k} = (\zeta_{0,j,k}, \zeta_{1,j,k}, \ldots, \zeta_{99,j,k}) \,,$$

$$N_j = (\nu_{0,j}, \nu_{1,j}, \ldots, \nu_{99,j})$$

and

$$R = (\rho_0, \rho_1, \ldots, \rho_{99}) \,.$$

The computation can then be performed as follows:

1. Let Y be a variable in $\{0,1\}^{100}$. Let Y be initialized to $R = (\rho_0, \rho_1, \ldots, \rho_{99})$.
2. For each monomial $x_j x_k$ $(0 \le j < k < 107)$: if $x_j = x_k = 1$ then replace Y by $Y \oplus Z_{j,k}$.
3. For each monomial x_j $(0 \le j < 107)$: if $x_j = 1$ then replace Y by $Y \oplus N_j$.

If, for instance, we use a 32-bit architecture, this leads to a speed-up of the algorithm: each vector $Z_{j,k}$ or N_j or R can be stored in four 32-bit registers. By using the 32-bit XOR operation, the $\oplus$ operations can be performed 32 bits by 32 bits. This means that we compute 32 public equations simultaneously.

8 Security of the QUARTZ algorithm

Traditionally, the security of public key algorithms relies on a problem which is both simple to describe and has the reputation to be difficult to solve (such as the factorization problem, or the discrete logarithm problem). On the opposite, traditionally, the security of secret key algorithms and of hash functions relies (not on such a problem but) on specific arguments about the construction (such as the soundness of the Feistel construction for example) and on the fact that the known cryptanalytic tools are far to break the scheme.

There are some exceptions. For example the public key scheme based on error correcting codes (such as the McEliece scheme, or the Niederreiter scheme) or the NTRU scheme do not have a security that provably relies on a well defined problem, and some hash functions have been designed on the discrete logarithm problem.

The security of the QUARTZ algorithm is also not proved to be equivalent to a well defined problem. However we have a reasonable confidence in its security due to some arguments that we will present in the sections below, and these arguments are not only subjective arguments.

Remark: As an example, let $\mathcal{F}$ be the composition the five AES finalists, with five independent keys of 128 bits. Almost everybody in the cryptographic community thinks that this $\mathcal{F}$ function will be a very secure function for the next 20 years, despite the fact that it security is not provably relied on a clearly, famous, and simple to describe problem.

Our (reasonable) confidence in the security of QUARTZ comes from the following five different kinds of arguments, that we will explain in more details below:

1. All the known attacks are far from being efficient.
2. There is a kind of "double layered" security in the design of the scheme: algebraic and combinatorial.
3. MQ looks really difficult in average (not only in worst case).
4. When the degree d (of the hidden polynomial F) increases, the trapdoor progressively disappears so that all the attacks must become more and more intractable.
5. The secret key is rather long (but it can be generated from a small seed of 80 bits for example), even for computing very short signatures.

8.1 All the known attacks are far from being efficient

Three kinds of attacks have been studied so far on schemes like the basic HFE or HFEV$^-$ (QUARTZ is a HFEV$^-$ scheme with a special choice for the parameters).

Some attacks are designed to recover the secret key (or an equivalent information) In this family of attack, we have the exhaustive search of the key (of course intractable) and the (much more clever) Shamir-Kipnis on the basic HFE scheme (cf [3]). However this Shamir-Kipnis attack would not be efficient on the QUARTZ algorithm (much more than 2^{80} computations are required) even if we removed the $-$ and V perturbations. Moreover, the Shamir-Kipnis seems to work only for the basic HFE scheme (*i.e.* without the perturbations $-$ and V) and in QUARTZ we have some $-$ and V. So in fact, at present for a scheme like QUARTZ we do not see how the Shamir-Kipnis attack may work at all.

Some attacks are designed to compute a signature S from a message M directly from the equations of the public key, as if there was no trapdoor (*i.e.* by solving a general system of quadratic equations) The MQ (= Multivariate Quadratic) problem of solving a general set of multivariate quadratic equations is a NP-Hard problem. Some (non polynomial but sometimes better than exhaustive search) algorithms have been designed for this problem, such as some Gröbner bases algorithms, or the XL and FXL algorithms (see [1]) but for our choices of the QUARTZ parameters, all these algorithms need more than 2^{80} computations.

Some attacks are designed to compute a signature S from a message M by detecting some difference on the public key compared to a system of general quadratic equations Many analysis have been made in these lines of attacks. Some "affine multiple attacks" have been design, and many variations around these attacks ("higher degree attacks" etc). At present, with the parameters of the QUARTZ algorithm all these attacks need more the 2^{80} computations.

8.2 There is a kind of "double layered" security in the design of the scheme: algebraic and combinatorial

The security of the basic HFE scheme (*i.e.* a HFE scheme with no perturbations such as $-$ and V) can be considered as a kind of "Algebraic" problem, since from the Shamir-Kipnis attack we know that it can be linked to a MinRank problem on very large algebraic fields. (The general MinRank problem is NP-Hard, but for the basic HFE it may not be NP-Hard, but it is still not polynomial when d is not fixed and $d = \mathcal{O}(n)$ for example). However this basic HFE scheme is Hidden in the QUARTZ algorithm with the perturbations $-$ and V. To remove these perturbations seems to be a very difficult combinatorial problem. So to break the QUARTZ scheme, it is expected that a cryptanalyst will have to solve a double problem: Combinatorial and Algebraic, and these problems do not appear separately but in a deeply mixed way to him on the public key.

8.3 MQ looks really difficult in average (not only in worst case)

In the past, some public key schemes apparently (not provably) based on some NP-Hard problems, such as the Knapsack problem were broken. However the MQ problem (*i.e.* solving a general set of multivariate quadratic equations) seems to be a much more difficult problem to solve than the Knapsack Problem: on the Knapsack Problem an algorithm such as LLL is very often efficient, while on the opposite ? on the MQ problem all the known algorithms are not significantly better than exhaustive search when the number m of equations is about the same as the number n of variables and is larger than, say, about 12.

It is also interesting to notice that almost all the "Knapsack Schemes" were broken due to a new algorithm on the general Knapsack problem (LLL) and not due to the fact that the security of these schemes was not properly proved to be equivalent to the Knapsack problem. Something similar seems to appear with the schemes based on error correcting codes, such as the McEliece Scheme, or the Niederreiter scheme: so far all the attacks on these schemes try to solve the general (and NP-Hard) problem of decoding a word of small weight in a general linear code, and not to try to use the fact that it is not proved that the security of these schemes is equivalent to solving this problem. If, for these schemes as for QUARTZ the practical cryptanalysis becomes in practice the problem of solving the general problem, then for QUARTZ the MQ problem looks really very difficult.

8.4 When the degree d (of the hidden polynomial F) increases, the trapdoor progressively disappears so that all the attacks must become more and more intractable

The degree d of the QUARTZ algorithm is fixed to 129. However if d was not fixed, and d could be as large as $2h$ ($h = 103$ in the QUARTZ algorithm), then all the possible systems of quadratic equations would appear in the public key, so the problem of solving it would be exactly as hard as the general MQ problem (on this number of variables). Of course, we have fixed d to 129 in order to be able to compute a signature in a reasonable time on a computer, but this result shows that when d increases, the trapdoor progressively disappears, so that all the attacks must become more and more intractable. So d is really an important "security parameter". Our choice of $d = 129$ has been made to be far from the current state of the art on the cryptanalysis with small d while still having a reasonable time on a computer to compute a signature.

8.5 The secret key is rather long (but it can be generated from a small seed of 80 bits for example), even for computing very short signatures

Many secrets are used in QUARTZ: the secret affine permutations s and s, the secret function F, the secret vinegar variables V, and the secret removed equations. To specify all the secret we need a rather long secret key. However, it is also possible to compute this secret key from a short seed by using any pseudorandom bit generator. In general the time to generate the secrets from the small seed will not increase a lot the time to generate a signature. Moreover it has to be done only once if we can store the secret key in a safe way on the computer. So for practical applications it is always possible to generate the secret key from a seed of, say, 80 bits, but this secret key for a cryptanalyst of QUARTZ will always be similar to a much larger secret key.

So QUARTZ has a property that already existed in schemes like DSS (where the lengths of p and q are different): the length of the secret key is not directly linked to the length of the signature. (This property does not exist in RSA, where the length of the secret key is never larger than the length of the signature. It explain why a QUARTZ or DSS signature can be much smaller than a RSA signature).

The fact that a cryptanalyst of QUARTZ has to face such a large secret key, may also be an argument to say that in practice the time to find a QUARTZ secret key may be intractable in practice, even if a new sub-exponential algorithm is found and used. (So far many cryptanalysis, such as the "affine multiple attacks", have to solve huge systems of linear equations by Gaussian reductions, and often the number of variables in these systems increases very fast with the length of the secret, so these attacks become impractical due to space and time limitations). However this argument is not very convincing and is maybe not as strong as the other arguments presented above.

9 Summary of the characteristics of QUARTZ

- Length of the signature: 128 bits.
- Length of the public key: 71 Kbytes.
- Length of the secret key: the secret key (3 Kbytes) is generated from a small seed of at least 128 bits.
- Time to sign a message[1]: 30 seconds on average.
- Time to verify a signature[2]: less than 0.12 ms (3×0.006 ms for the three SHA-1, plus 0.1 ms for the quadratic equations).
- Best known attack: more than 2^{80} computations.

References

1. N. Courtois, A. Shamir, J. Patarin, A. Klimov, *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations*, in Advances in Cryptology, Proceedings of EUROCRYPT'2000, LNCS n° 1807, Springer, 2000, pp. 392-407.
2. E. Kaltofen, V. Shoup, *Fast polynomial factorization over high algebraic extensions of finite fields*, in Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation, 1997.
3. A. Kipnis, A. Shamir, *Cryptanalysis of the HFE public key cryptosystem*, in Advances in Cryptology, Proceedings of Crypto'99, LNCS n° 1666, Springer, 1999, pp. 19-30.
4. J. Patarin, *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of asymmetric algorithms*, in Advances in Cryptology, Proceedings of EUROCRYPT'96, LNCS n° 1070, Springer Verlag, 1996, pp. 33-48.
5. A. Kipnis, J. Patarin and L. Goubin, *Unbalanced Oil and Vinegar Signature Schemes*, in Advances in Cryptology, Proceedings of EUROCRYPT'99, LNCS n° 1592, Springer, 1999, pp. 206-222.
6. The HFE cryptosystem web page: http://www.hfe.minrank.org

[1] On a Pentium III 500 MHz. This part can be improved: the given software was not optimized.

[2] This part can be improved: the given software was not optimized.