Practical Cryptanalysis of k-ary C^*

Daniel Smith-Tone[1,2]

[1]Department of Mathematics, University of Louisville,
Louisville, Kentucky, USA
[2]National Institute of Standards and Technology,
Gaithersburg, Maryland, USA

daniel.smith@nist.gov

Abstract. Recently, an article by Felke appeared in Cryptography and Communications discussing the security of biquadratic C^* and a further generalization, k-ary C^*. The article derives lower bounds for the complexity of an algebraic attack, directly inverting the public key, under an assumption that the first-fall degree is a good approximation of the solving degree, an assumption that the paper notes requires “greater justification and clarification.”

In this work, we provide a practical attack breaking all k-ary C^* schemes. The attack is based on differential techniques and requires nothing but the ability to evaluate the public key and solve linear systems. In particular, the attack breaks the parameters provided in CryptoChallenge 11 by constructing and solving linear systems of moderate size in a few minutes.

Key words: Multivariate Cryptography, k-ary C^*, differential attack

1 Introduction

Massively multivariate public key cryptography was first introduced outside of Japan in the EuroCrypt ’88 paper by Matsumoto and Imai, see [1], that presented what has become known as the C^* cryptosystem. After Shor discovered polynomial-time factoring and discrete logarithm quantum algorithms, see [2], schemes based on different problems, and in particular on NP-hard problems such as that of solving multivariate nonlinear systems, became much more interesting to cryptographers. Now with the ongoing post-quantum standardization effort by the National Institute of Standards and Technology (NIST), see [3], such multivariate schemes are now being considered for practical widespread use.

In [4], Patarin broke the original C^* scheme with an attack based on linearization equations. At around this time, in the late ’90s, there was an explosion of research in multivariate cryptography. Numerous schemes were introduced and cryptanalyzed, see, for example, [5–11].

In 2005, Dobbertin et al. present a cryptographic challenge based on the idea of C^*. The scheme is called a biquadratic C^* and has a massive public key of quartic polynomials. Like C^*, biquadratic C^* is based off of a power function, but with an exponent of Hamming weight four in its q-ary expansion, where q is the size of the public finite field. Naturally, this construction can be generalized to a k-ary C^* in which the q-ary expansion of the exponent of the private power function has Hamming weight k.

This more general k-ary C^* is analyzed by Felke in [12], where he derives lower bounds for the first-fall degree of the public key under direct attacks via Gröbner bases. Although we should note that first-fall degree is dependent on both the polynomial system and the Gröbner basis algorithm, Felke's result implies a lower bound in the complexity of solving such a system with *any* Gröbner basis algorithm. As noted in [12], the complexity estimates of the direct attack on k-ary C^* derived therein depend on an assumption that the first-fall degree is equal to the solving degree which is not always the case. Even granting these complexities, it is interesting to note that the complexity of quantum algorithms such as quantum-FXL, see [13], which were ignored in [12], outperform these optimistic analyses.

In this work, we provide an efficient cryptanalysis of k-ary C^* and some modest generalizations. This attack is based on a property of the differential of a power function that the author derived over ten years ago, see [14]. The attack reduces the task of deriving a decryption key to that of solving systems of linear equations. In particular, for the CryptoChallenge 11, see [15], one evaluation of the public key, the calculation of the differential of two public equations and the solution of two linear systems of size 627 and 625, respectively, are sufficient to completely break the scheme. The complexity for an optimized implementation for these parameters is roughly 2^{38} operations over GF(16). We implemented the attack using crude and simple symbolic algebra techniques and, after a few minutes of sloppily gathering coefficients, solved the linear system and broke the proposed parameters in an instant. In the most general case, the complexity of the optimized attack is $\mathcal{O}\left(n^2\binom{n}{k}^2\right)$. Using the full formula for this estimate produces an upper bound of 2^{68} operations over $GF(16)$ even for the "secure" biquadratic scheme proposed in [12].

2 k-ary C^*

Let $\mathbb{F}_q$ be a finite field with q elements. Consider $\mathbb{K}$, a degree n extension of $\mathbb{F}_q$. Fix an $\mathbb{F}_q$-vector space isomorphism $\phi : \mathbb{F}_q^n \to \mathbb{K}$. Then for any univariate map $f : \mathbb{K} \to \mathbb{K}$ we can construct the vector-valued map $F : \mathbb{F}_q^n \to \mathbb{F}_q^n$ defined by $F = \phi^{-1} \circ f \circ \phi$. Since any multivariate function on a finite field is a polynomial, each coordinate of F is a polynomial in its n inputs.

To hide the structure of an efficiently invertible univariate map it is necessary to randomize the input and output bases of the representation of $\mathbb{K}$ as a commutative $\mathbb{F}_q$-algebra. Thus the public key P is related to the private map F by an isomorphism (T, U) where T and U are $\mathbb{F}_q$-affine maps of dimension n. Thus the entire construction is given by Figure 1

$$\begin{tikzcd}
& \mathbb{K} \arrow[r, "f"] & \mathbb{K} \arrow[d, "\phi^{-1}"] & \\
\mathbb{F}_q^n \arrow[r, "U"] \arrow[rrr, dashed, bend right, "P"] & \mathbb{F}_q^n \arrow[u, "\phi"] \arrow[r, dashed, "F"] & \mathbb{F}_q^n \arrow[r, "T"] & \mathbb{F}_q^n
\end{tikzcd}$$

Fig. 1. The structure of big field public key cryptosystems.

As defined in [15], a k-ary C^* map is an univariate function $f : \mathbb{K} \to \mathbb{K}$ of the form $f(x) = x^e$, where the q-ary expansion of e is binary having Hamming weight k and e is coprime with $|U(\mathbb{K})|$. Notice that

$$x^e = x^{q^{a_1}+\cdots+q^{a_k}} = \prod_{i=1}^{k} x^{q^{a_i}},$$

and since the Frobenius automorphisms are $\mathbb{F}_q$-linear, $F = \phi^{-1} \circ f \circ \phi$ is of $\mathbb{F}_q$-degree k.

3 Previous Cryptanalyses of C^* and Variants

In [4], Patarin breaks the original C^* scheme by deriving the so-called linearization equations. He noticed that given a C^* map of the form $f(x) = x^{q^\theta+1}$, we obtain the relation $uf(u)^{q^\theta} = u^{q^{2\theta}} f(u)$. That is, if we let $v = f(u)$, then we obtain a bilinear relation between u and v. Since u and v are related to the plaintext and ciphertext of the public key system via the maps U and T, respectively, we have a bilinear relation between plaintext and ciphertext. A simple analysis shows that even in the most fortuitous case, the adversary can reduce the dimension of the possible preimage space by a factor of three, thus rendering C^* too inefficient for practical use.

As a method of repairing the scheme, it was suggested in [16] to remove some of the public equations. The technique avoids the linearization equations attack since the bilinear relation between plaintext and ciphertext pairs for C^* is explicitly given by

$$(Ux)(T^{-1}(y))^{q^\theta} = (Ux)^{q^{2\theta}}(T^{-1}(y)).$$

This idea eventually evolved in to the SFLASH digital signature scheme of [9].

In [17], an attack that completely breaks SFLASH is presented. The attack uses the discrete differential of the public key. Given a function $F : \mathbb{F}_q^n \to \mathbb{F}_q^n$, the discrete differential is defined by

$$DF(a, x) = F(a + x) - F(a) - F(x) + F(0).$$

The attack proceeds by way of a symmetric relation satisfied by a C^* monomial map $f(x) = x^{q^\theta+1}$. Specifically,

$$Df(\sigma a, x) + Df(a, \sigma x) = (\sigma^{q^\theta} + \sigma)Df(a, x).$$

This property is inherited by the public key $P = \Pi \circ T \circ f \circ U$ in the following form:

$$D\left[\Pi \circ P\right](N_\sigma a, x) + D\left[\Pi \circ P\right](a, N_\sigma x) = \Pi \circ \Lambda_\sigma DP(a, x), \tag{1}$$

where $N_\sigma = U^{-1}M_\sigma U$ and M_σ is a left multiplication representation of $\sigma \in \mathbb{K}$.

For any validly formed N_σ, Equation 1 guarantees that the left-hand side is a linear combination of the differential coordinate forms without equations removed. Thus, Equation 1 provides a criterion for finding such an N_σ. Specifically, if we insist that a few coordinates of the left-hand side of Equation 1 are in the span of the known differential coordinate forms, then it is likely that N_σ is a multiplication. In this way, one can recover such a multiplication. Once found, $P \circ N_\sigma$ provides new linearly independent equations that can be added to the original public key to recreate a compatible C^* public key. At this point, Patarin's original linearization equations attack can be used to break the scheme.

4 A Different Cryptanalysis of C^*

The attack of [17] inspires a new idea for attacking the original C^* directly. The idea is to interpret the map recovered via the differential symmetry technique as a multiplication map under a different basis, one parameterized by U. Using this map one may recover a representation of $\mathbb{K}$ as an $\mathbb{F}_q$-algebra. Then one uses this information along with the public key to recover another representation of $\mathbb{K}$ as an $\mathbb{F}_q$-algebra, this time parameterized by T. Then one can view the public key as a power function between these two representations. Once the function is known, a single input-output pair can be used to construct an efficient inverse function.

4.1 Alternate Decryption Key Recovery

Suppose that we have a solution N_σ of Equation 1. Then necessarily,

$$\begin{aligned} P \circ N_\sigma &= T \circ F \circ M_\sigma \circ U \\ &= T \circ M_{f(\sigma)} \circ F \circ U \\ &= \left(T \circ M_{f(\sigma)} \circ T^{-1}\right) \circ T \circ F \circ U \\ &= Z_{f(\sigma)} \circ P. \end{aligned}$$

Thus P translates right composition of multiplications in the basis U^{-1} into left composition of multiplications in the basis T.

Given such a matrix N_σ, we compute $Z_{f(\sigma)}$, and by guessing f (since there are so few possibilities), we may recover the corresponding pair $(N_{f(\sigma)}, Z_{f(\sigma)})$.

Naturally, if we have guessed f, then we can raise $Z_{f(\sigma)}$ to the appropriate power to similarly recover Z_σ. Either way, with high probability σ is a generator of $\mathbb{K}^*$ and thus $f(\sigma)$ is also a generator. So we may form a basis for the two representations of $\mathbb{K}$ as $\mathbb{F}_q$-algebras by computing $\{I, N_\sigma, N_\sigma^2, \dots, N_\sigma^{n-1}\}$ and $\{I, Z_\sigma, Z_\sigma^2, \dots, Z_\sigma^{n-1}\}$.

Now, given a single input output pair $y_0 = P(x_0)$, we can decrypt any message $y = P(x)$ by first finding the appropriate multiplication Z_τ such that $Z_\tau y_0 = y$. Given the representation of Z_τ over its basis,

$$Z_\tau = \sum_{i=0}^{n-1} \lambda_i Z_\sigma^i,$$

we construct

$$N_\tau = \sum_{i=0}^{n-1} \lambda_i N_\sigma^i.$$

Then, by construction, we have that

$$\begin{aligned} y = Z_\tau y_0 &= Z_\tau \circ P(x_0) \\ &= T \circ M_\tau \circ T^{-1} \circ T \circ F \circ U(x_0) \\ &= T \circ M_\tau \circ F \circ U(x_0) \\ &= T \circ F \circ M_{f^{-1}(\tau)} \circ U(x_0) \\ &= T \circ F \circ U \circ U^{-1} \circ M_{f^{-1}(\tau)} \circ U(x_0) \\ &= P \circ N_{f^{-1}(\tau)}(x_0). \end{aligned}$$

Thus $P^{-1}(y) = x = N_{f^{-1}(\tau)}(x_0)$. To find $N_{f^{-1}(\tau)}$, we simply find $he = 1$ modulo $|\mathbb{K}^*|$, and compute $N_\tau^h = N_{f^{-1}(\tau)}$.

Thus, the key step in breaking C^* in this manner is a solution of Equation 1 in the case that Π is the identity map. We generically have no extraneous solutions as long as $3\theta \neq n$ as proven in [18].

This method provides a distinct cryptanalysis of C^* involving only solving linear systems. The technique is quite efficient, and provides a new signing key that is different from the original signing key and the one derived with the linearization equations attack.

These computational techniques are described in more detail in Algorithm 1. One should note that the random selection in step 6 is selecting from exactly an n-dimensional $\mathbb{F}_q$-vector space of solutions corresponding to the "multipication maps" of the form N_σ as proven in [18]. This step can be modified to assure that a nontrivial solution is obtained.

4.2 Full Key Decomposition

One may extend the attack further to recover a private key of the form (T', U')—recall that f was already guessed. We consider the decomposition in stages. First, we derive linear maps (T'', U'') such that $T''^{-1} \circ P \circ U''^{-1}$ is multiplicative. Once

Algorithm 1: Decrypt* C^*

```
Input  : public key $P$, ciphertext $y = P(x)$
Output: plaintext $x$ such that $P(x) = y$
$x_0 \xleftarrow{\$} \mathbb{F}_q^n$;
$y_0 \longleftarrow P(x_0)$;
$DP(a, x) \longleftarrow P(a + x) - P(a) - P(x) + P(0)$;
$N_\sigma \longleftarrow$Matrix($[[r_1, \dots, r_n], \dots, [r_{n^2-n+1}, \dots, r_{n^2}]]$);
$\Lambda_\sigma \longleftarrow$Matrix($[[s_1, \dots, s_n], \dots, [s_{n^2-n+1}, \dots, s_{n^2}]]$);
$v \xleftarrow{\$}$LinearSolve($DP(N_\sigma a, x) + DP(a, N_\sigma x) = \Lambda_\sigma DP(a, x)$);
$N_\sigma \longleftarrow$Eval($N_\sigma$,$[v[i] :\ i \in [1..n^2]]$);
$Z_{f(\sigma)} \longleftarrow$Matrix($[[r_1, \dots, r_n], \dots, [r_{n^2-n+1}, \dots, r_{n^2}]]$);
$w \longleftarrow$LinearSolve($Z_{f(\sigma)} \circ P = P \circ N_\sigma$);
$Z_{f(\sigma)} \longleftarrow$Eval($Z_{f(\sigma)}$,$w$);
for $e$ in $[1 + q^1, \dots, 1 + q^{n-1}]$ st $(e, q^n - 1) = 1$ do
    $h \longleftarrow$InverseMod($e$,$q^n - 1$);
    $Z_\sigma \longleftarrow Z_{f(\sigma)}^h$;
    $\lambda \longleftarrow$LinearSolve($\sum_{i=1}^n \lambda_i Z_\sigma^{i-1} y_0 = y$);
    $N_\tau \longleftarrow \sum_{i=1}^n \lambda_i N_\sigma^{i-1}$;
    $N_{f^{-1}(\tau)} \longleftarrow N_\tau^h$;
    $x_{cand} \longleftarrow N_{f^{-1}(\tau)} x_0$;
    if $y == P(x_{cand})$ then
        return $x_{cand}$
    end
end
```

obtained, a single input/output pair for this map is computed and used to anchor this multiplicative to f and ultimately to derive equivalent maps (T', U').

Having recovered the maps N_σ and $Z_{f(\sigma)}$, we consider the relations

$$N_\sigma = U^{-1} \circ M_\sigma \circ U \text{ and } Z_{f(\sigma)} = T M_{f(\sigma)} T^{-1}.$$

Clearly, the minimal polynomial $\min(N_\sigma) = \min(M_\sigma)$ which is the same as the minimal polynomial of σ or any of its conjugates. In particular, under the action of $\mathbb{K}^* \rtimes \mathrm{Gal}_{\mathbb{F}_q}(\mathbb{K}) \hookrightarrow GL_n(\mathbb{F}_q)$ by conjugation, the orbit of M_σ is

$$\{M_\tau : \phi(\sigma) = \tau \text{ for some } \phi \in \mathrm{Gal}_{\mathbb{F}_q}(\mathbb{K})\}.$$

Thus the stabilizer corresponds to the subgroup isomorphic to $\mathbb{K}^*$.

We directly solve the linear system

$$\widehat{U} N_\sigma = M_\tau \widehat{U},$$

in the unknown coefficients of $\widehat{U}$ for some τ a root of $\min(N_\sigma)$. Since the action of $\mathbb{K}^* \rtimes \mathrm{Gal}_{\mathbb{F}_q}(\mathbb{K})$ on the image of $\mathbb{K}$ in $GL_n(\mathbb{F}_q)$ is transitive and since the choice of τ in general fixes the automorphism, there are usually n degrees of freedom in $\widehat{U}$. We similarly solve the linear system

$$Z_{f(\sigma)} \widehat{T} = \widehat{T} M_{f(\tau)},$$

with the same τ as the first step, again with n degrees of freedom usually.

Next, we construct the augmented key $\widehat{P} = \widehat{T}^{-1} \circ P \circ \widehat{U}^{-1}$. Notice that

$$\begin{aligned}
\widehat{P} \circ M_\tau &= \widehat{T}^{-1} \circ T \circ F \circ U \circ \widehat{U} \circ M_\tau \\
&= \widehat{T}^{-1} \circ T \circ F \circ M_\sigma \circ U \circ \widehat{U} \\
&= \widehat{T}^{-1} \circ T \circ M_{f(\sigma)} \circ F \circ U \circ \widehat{U} \\
&= M_{f(\tau)} \circ \widehat{T}^{-1} \circ T \circ F \circ U \circ \widehat{U},
\end{aligned}$$

where σ is a conjugate of τ. Thus $\widehat{P}$ is an isomorphic copy of the public key that is multiplicative.

Finally, we fix and arbitrary input/output pair $y' = \widehat{P}(x')$. We can now directly compute a decomposition of the public key as $T'' = \widehat{T} M_{y'}$, $U'' = M_{x'}^{-1} \widehat{U}$, and of course f which was guessed before. Note that if $y = P(x)$, then $\widehat{T}^{-1} y$ can be viewed as the output of $\widehat{P}$ with input $\widehat{U}(x)$. So we may use the same trick from Subsection 4.1 to find a preimage of $\widehat{T}^{-1} y$ under $\widehat{P}$. Specifically, this involves dividing by y' (multiplying on the left by $M_{y'}^{-1}$), inverting F and multiplying by x' (that is, $M_{x'}$). At this point we have obtained $\widehat{U}(x)$, so inversion is completed by the application of $\widehat{U}^{-1}$. More explicitly, observe that

$$\begin{aligned}
&\left(\widehat{T} \circ M_{y'}\right) \circ \widehat{T}^{-1} \circ T \circ F \circ U \circ \widehat{U}^{-1} \circ \left(M_{x'}^{-1} \circ \widehat{U}\right) \\
&\qquad = \left(\widehat{T} \circ M_{y'}\right) \circ \widehat{T}^{-1} \circ T \circ F \circ M_{\overline{x'}^{-1}} \circ U \circ \left(\widehat{U}^{-1} \circ \widehat{U}\right) \\
&\qquad = \left(\widehat{T} \circ M_{y'}\right) \circ \widehat{T}^{-1} \circ T \circ M_{\overline{f(x')}^{-1}} \circ F \circ U \\
&\qquad = \widehat{T} \circ \left(M_{y'} \circ M_{f(x')^{-1}}\right) \circ \widehat{T}^{-1} \circ T \circ F \circ U \\
&\qquad = \left(\widehat{T} \circ \widehat{T}^{-1}\right) \circ T \circ F \circ U \\
&\qquad = T \circ F \circ U.
\end{aligned}$$

5 Cryptanalysis of k-ary C^*

We now prove for any k that k-ary C^* has a differential symmetry. Moreover, multiplication maps are the only maps inducing symmetry in this way, assuring that once the symmetric equations are solved that a multiplication map has been found. We then use this fact to construct an attack analogous to that of Section 4.

We first define the rth discrete differential.

Definition 1 *The rth discrete differential of a map $F : \mathbb{F}_q^n \to \mathbb{F}_q^n$ is defined as*

$$D^rF(x_1,\dots,x_r) = \begin{cases} F & \textit{if } r=0 \\ D^{r-1}F(x_1+x_2,x_3,\dots,x_r) & \\ \quad -D^{r-1}F(x_1,x_3,\dots,x_r) & \textit{otherwise.} \\ \quad -D^{r-1}F(x_2,x_3,\dots,x_r) & \\ \quad +D^{r-1}F(0,x_3,\dots,x_r) & \end{cases}$$

We note explicitly that since the discrete differential operator D is symmetric, when given a symmetric multivariate function $G(a,\dots,b)$, we have that $D_aG(x,a,\dots,b) = D_bG(a,\dots,b,x)$ and is symmetric; that is, the same function is obtained when taking the differential with respect to any variable. Thus all higher order differentials are the same regardless of the sequence of variables with respect to which the differentials are taken and the rth differential is well-defined.

Theorem 1 *Let $f : \mathbb{K} \to \mathbb{K}$ be the k-ary C^* map $f(x) = x^{q^{i_1}+\cdots+q^{i_k}}$. Then f satisfies the differential symmetry*

$$\sum_{j=1}^{k} D^{k-1}f(\sigma^{\delta_{j,1}}x_1,\dots,\sigma^{\delta j,k}x_k) = (\sum_{j=1}^{k}\sigma^{q^{i_j}})D^{k-1}f(x_1,\dots,x_k), \tag{2}$$

where $\delta_{r,s}$ is the Kronecker delta function.

Proof. By calculation, $D^{k-1}f(x_1,\dots,x_k)$ is $\mathbb{F}_q$-multilinear and so every monomial summand is of the form

$$x^\alpha = x_1^{q^{\alpha_1}}x_2^{q^{\alpha_2}}\cdots x_k^{q^{\alpha_k}},$$

for some α, a permutation of $(i_1,\dots,i_k)$. Each summand of the left hand side of Equation 2 contains exactly one term of the form $\sigma^{q^{\alpha_i}}x^\alpha$ and the contribution of each differential is distinct. Thus, the sum of the x^α terms of the left hand side of Equation 2 is $(\sum_{j=1}^k \sigma^{q^{i_j}})x^\alpha$ for every α. Summing over all possible α and factoring out $(\sum_{j=1}^k \sigma^{q^{i_j}})$, we obtain the result.

Thus, k-ary C^* monomial maps satisfy the same multiplicative symmetry that C^* monomial maps exhibit. The key here seems to be that these maps are multiplicative, and the multiplicative symmetry is the manifestation of that property in the differential. By an argument analogous to that in [18], it can be shown that if L induces a differential symmetry with a k-ary C^* map, then $\phi(Lx) = \sigma(\phi(x))$ for some $\sigma \in \mathbb{K}$. See [14] for details.

Now we may implement an attack of the exact same manner as that of Section 4. The main difference is that we must compute a higher order differential and guess an encryption exponent of a different form. For all of the details, see Algorithm 2.

```
Algorithm 2: Decrypt* k-ary C*
Input  : public key $P$, ciphertext $y = P(x)$
Output: plaintext $x$ such that $P(x) = y$
$x_0 \xleftarrow{\$} \mathbb{F}_q^n$;
$y_0 \longleftarrow P(x_0)$;
$D^{k-1}P(a,x) \longleftarrow$ Differential($P$,$k$-1);
$N_\sigma \longleftarrow$ Matrix($[[r_1,\ldots,r_n],\ldots,[r_{n^2-n+1},\ldots,r_{n^2}]]$);
$\Lambda_\sigma \longleftarrow$ Matrix($[[s_1,\ldots,s_n],\ldots,[s_{n^2-n+1},\ldots,s_{n^2}]]$);
$v \xleftarrow{\$}$ LinearSolve($\sum_{j=1}^{k} D^{k-1}P(N_\sigma^{\delta_{j,1}}x_1,\ldots,N_\sigma^{\delta_{j,k}}x_k) = \Lambda_\sigma D^{k-1}P(a,x)$);
$N_\sigma \longleftarrow$ Eval($N_\sigma$,$[v[i] :\ i \in [1..n^2]]$);
$Z_{f(\sigma)} \longleftarrow$ Matrix($[[r_1,\ldots,r_n],\ldots,[r_{n^2-n+1},\ldots,r_{n^2}]]$);
$w \longleftarrow$ LinearSolve($Z_{f(\sigma)} \circ P = P \circ N_\sigma$);
$Z_{f(\sigma)} \longleftarrow$ Eval($Z_{f(\sigma)}$,$w$);
for $e$ in $[1+q^1+\cdots+q^{k-1},\ldots,1+q^{n-k+1}+\cdots+q^{n-1}]$ st $(e,q^n-1)=1$ do
    $h \longleftarrow$ InverseMod($e$,$q^n-1$);
    $Z_\sigma \longleftarrow Z_{f(\sigma)}^h$;
    $\lambda \longleftarrow$ LinearSolve($\sum_{i=1}^{n} \lambda_i Z_\sigma^{i-1} y_0 = y$);
    $N_\tau \longleftarrow \sum_{i=1}^{n} \lambda_i N_\sigma^{i-1}$;
    $N_{f^{-1}(\tau)} \longleftarrow N_\tau^h$;
    $x_{cand} \longleftarrow N_{f^{-1}(\tau)} x_0$;
    if $y == P(x_{cand})$ then
        return $x_{cand}$
    end
end
```

6 Complexity

Even a direct symbolic approach to implementing the attack of Section 5 is sufficient to break the parameters of CryptoChallenge11 from [15]. Specifically, using symbolic algebra, we broke the biquadratic C^* with parameters $q = 16$, $n = 25$ and $e = 1+q+q^3+q^{12}$ with a simple and straightforward Magma implementation with symbolic algebra, in 593.25 seconds using 3.9GB of memory.

The implementation is not at all optimized, as it is not necessary to make a complex implementation to break the full-sized parameters. The implementation uses symbolic algebra over a polynomial ring over a polynomial ring over a polynomial ring over $\mathbb{F}$! We did, however, incorporate some of the trivial to implement optimization techniques we now present. An optimized implementation will make use of the fact that the symmetry relations derived to effect the attack are linear in the coefficients of the public key; thus, with some engineering, the entire attack can be reduced to a few operations on some matrices of moderate size. We describe this technique in more detail at the end of the section.

First, the linear system

$$\sum_{i=1}^{k} D^{k-1}P(N_\sigma^{\delta_{i,1}}x_1,\ldots,N_\sigma^{\delta_{i,k}}x_k)=\Lambda_\sigma D^{k-1}P(a,x), \tag{3}$$

where $\delta_{i,j}$ is the Kronecker delta, is massively redundant. The system is dramatically overdefined typically even when one coordinate of the left-hand side is used.

Each monomial $x_{1,i_1}\cdots x_{k,i_k}$ with the i_j pairwise distinct in each coordinate of $D^{k-1}P$ produces an equation. Thus the entire linear system in Equation 3 is $n\binom{n}{k}$ equations in the $2n^2$ unknown coordinates of N_σ and Λ_σ.

Since we are only interested in solving for N_σ, we can reduce this system dramatically by considering fewer coordinates of the left-hand side. The resulting system will use a corresponding number of rows of the matrix Λ_σ, so fewer variables are required as well. We may choose r coordinates to recover $r\binom{n}{k}$ equations in n^2+r*n unknowns. Clearly, the system is fully determined with 3 coordinates when $k=2$ and $n\geq 9$ or with even a single coordinate when $k>3$ and $n>10$, for example. In particular, the large values of k make the system more overdetermined when even a single coordinate on the left hand side is considered.

We can improve the complexity even further by not considering all of the coordinates of $D^{k-1}P$ on the right-hand side of Equation 3. As in the attack on SFLASH of [17], we may consider an analysis of the number of linear maps whose symmetric action on the first r coordinates of the differential map it into the span of the first s coordinates of $D^{k-1}P$.

Fix an arbitrary linear map M and consider the expression

$$\widetilde{M_i}=\sum_{j=1}^{k} D^{k-1}P_i(M^{\delta_{j,1}}x_1,\ldots,M^{\delta_{j,k}}x_k),\ \text{for}\ i\in\{1,\ldots,r\},$$

which can be viewed as an r-tuple of symmetric k-tensors. The span of all such symmetric k-tensors $\mathcal{S}$, under the heuristic that P_i is random, q and n are sufficiently large and $k>2$, has dimension rn^2, that is, r times the dimension of $\mathcal{M}_{n\times n}(\mathbb{F}_q)$. The first s coordinates of $D^{k-1}P$ generate an s-dimensional space V_s of k-tensors. We note explicitly that since each multiplication of the form N_σ produces k-tensors that are guaranteed to be in V_n that V_n, and therefore V_s is contained in $\mathcal{S}$.

Membership of each coordinate of $\widetilde{M}$ in V_s requires the satisfaction of n^2-s linear equations. Thus the membership of all coordinates of $\widetilde{M}$ in V_s requires the satisfaction of $r(n^2-s)$ linear equations. This analysis thus suggests that it is unlikely for all coordinates of $\widetilde{M}$ to be in V_s for random M as soon as $r>1$.

On the other hand, if M is already a multiplication map of the form N_σ then $\widetilde{M}$ is already guaranteed to be in V_n. Moreover, the condition that each of the first r coordinates of $\widetilde{M}$ is in V_s is satisfied explicitly under the appropriate change of basis by the preimage of $\mathrm{Span}(1,\alpha^{-1},\ldots,\alpha^{1-s})$ under the linear map

$x \mapsto x^{q^{a_1}} + x^{q^{a_2}} + \cdots + x^{q^{a_k}}$ if $r \leq s$. In particular, if $r = s$ we obtain an s-dimensional space of multiplications.

Considering the above analysis, we expect for $k > 2$ and n sufficiently large that choosing the first two coordinates of the left-hand side of Equation 3 to be in the span of the first two coordinates of the right-hand side provides enough relations to produce a 2-dimensional subspace consisting entirely of maps of the form of N_σ. Our experiments confirm that this approach works. Table 1 provides performance numbers for this attack using $r = s = 2$ for biquadratic C^* instances.

n	9	11	13	15	25
(s)	0.9	2.88	8.04	21.3	593.25
(MB)	22.6	46.71	85.99	287.63	3883.34

Table 1. The performance of a simple Magma implementation of the above attack against biquadratic C^* over GF(16) using $r = 2$ coordinates of the left-hand side and the span of $s = 2$ coordinates of the right-hand side of Equation 3. The last column is the performance in breaking CryptoChallenge11 from [15].

We note a couple of properties of this attack. Since the symmetric relations of Equation 3 are linear in the highest degree terms of the public key, there exists a massive binary matrix that produces the symmetric relations from the public coefficients. In the symbolic implementation above, almost all of the time was spent recovering these linear equations, with all of the overhead of the polynomial rings with hundreds of variables, before they were nearly instantly solved.

To make the attack more efficient, one can note that the differential symmetric equations are linear functions of the coefficients of the public key. Thus one may construct a linear function to derive the relations directly from the public key coefficients. We derive this function in the $k = 2$ case. The general case is similar and quite tedious to build.

Note that

$$\begin{aligned}
&DP_l(M\mathbf{a}, \mathbf{x}) + DP_l(\mathbf{a}, M\mathbf{x}) \\
&\qquad = \sum_{i<j} c_{ijl} \left[\sum_{k=1}^{n} m_{ik} a_k x_j + \sum_{k=1}^{n} m_{jk} a_i x_k + \sum_{k=1}^{n} m_{jk} a_k x_i + \sum_{k=1}^{n} m_{ik} a_j x_k \right] \\
&\qquad = \sum_{k=1}^{n} \sum_{j=2}^{n} \sum_{i<j} c_{ijl} m_{ik} a_k x_j + \sum_{k=1}^{n} \sum_{i=1}^{n-1} \sum_{i<j} c_{ijl} m_{jk} a_i x_k \\
&\qquad\quad + \sum_{k=1}^{n} \sum_{i=1}^{n-1} \sum_{i<j} c_{ijl} m_{jk} a_k x_i + \sum_{k=1}^{n} \sum_{j=2}^{n} \sum_{i<j} m_{ik} a_j x_k.
\end{aligned}$$

Collecting coefficients of $a_r x_s$ we obtain

$$[a_r x_s] = \sum_{i<s} c_{isl} m_{ir} + \sum_{r<i} c_{ril} m_{is} + \sum_{s<i} c_{sil} m_{ir} + \sum_{i<r} c_{irl} m_{is}.$$

We form a matrix $\mathcal{A}_l$ whose rows are indexed by (r, s) with $r < s$ and whose columns are indexed by (u, v) with $1 \leq u, v \leq n$.

$$\mathcal{A}_{l,(r,s),(u,v)} = \begin{cases} c_{usl} \text{ if } u < s \text{ and } v = r \\ c_{url} \text{ if } u < r \text{ and } v = s \\ c_{rul} \text{ if } r < u \text{ and } v = s \\ c_{sul} \text{ if } s < u \text{ and } v = r \\ 0 \text{ otherwise.} \end{cases}$$

From this expression, we may derive as many as n matrices of size $\binom{n}{2} \times n^2\binom{n}{2}$ which can be multiplied on the left by the vector of cross term coefficients of each public formula to produce row vectors of $\mathcal{A}_l$. Each row of $\mathcal{A}_l$ now represents the coefficients of m_{ij} occuring in the left-hand side of coordinate l of Equation 3. In a similar way we can construct additional matrices generating the right-hand side of the relations from the public coefficients and horizontally join the result to $\mathcal{A}_l$. Elements in the nullspace of this matrix then correspond to matrices M satisfying Equation 3.

Considering the more general case of k-ary C^*, for $k > 2$, we may limit the number of matrices above to 2 for each of the left and right-hand sides. Then deriving the symmetry relations requires linear algebra on matrices of size $n^2\binom{n}{k}$, and solving the system requires finding a kernel vector for a matrix of size $2n^2\binom{n}{k} \times (n^2 + 4)$. Note that a nontrivial kernel vector exists when the rank of this matrix is bounded by $n^2 + 3$, and in this case we can find a vector with high probability by only considering $\mathcal{O}\left(n^2\right)$ rows. Thus the complexity for the entire recovery of the multiplication map is

$$2n^2\binom{n}{k}^2 + \mathcal{O}\left(n^2\right)\left(n^2 + 4\right) = \mathcal{O}\left(n^2\binom{n}{k}^2\right),$$

ignoring sparse optimizations. For CryptoChallenge11, this quantity is upper bounded by 2^{38}, which is far superior to the symbolic implementation described and executed above. For the "secure" variant of biquadratic C^* recently proposed in [12], the formula above provides an upper bound of 2^{68}, far less than the claimed security bound of 80 bits.

7 Conclusion

Although C^* has been the foundation of one of the main approaches to multivariate public key cryptography in the last decades, it has also been a source of failure for many constructions based too directly on it, see, for example, [19]. The k-ary generalization of C^* falls into this category as well. While the differential relations are more cumbersome to derive in the k-tensor space than for the original C^*, the extent of the symmetry inherent to the central map makes it easy to derive the polynomially sized overdetermined linear system required to break the scheme.

Some of the major accomplishments of multivariate cryptography in the twenty-first century are derivations of proofs that certain modifications of schemes preclude certain classes of attacks. For C^* variants, one may provably prevent an attack recovering a differential symmetry on the public key by using nontrivial projections on both the input variables and the output polynomials, see [20, 18]. It is an interesting theoretical, if not entirely practical, question as to whether the same result can be derived in the k-ary case. Clearly, the attack presented here can be used to recover a full rank scheme from a minus modified one and break it similarly to SFLASH. It is an open question as to whether a projected k-ary C^*- scheme can be secure.

References

1. Matsumoto, T., Imai, H.: Public quadratic polynomial-tuples for efficient signature verification and message-encryption. Eurocrypt '88, Springer **330** (1988) 419–545
2. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Sci. Stat. Comp. **26, 1484** (1997)
3. Group, C.T.: Submission requirements and evaluation criteria for the post-quantum cryptography standardization process. NIST CSRC (2016) http://csrc.nist.gov/groups/ST/post-quantum-crypto/documents/call-for-proposals-final-dec-2016.pdf.
4. Patarin, J.: Cryptoanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt '88. In Coppersmith, D., ed.: CRYPTO. Volume 963 of Lecture Notes in Computer Science., Springer (1995) 248–261
5. Patarin, J.: The oil and vinegar algorithm for signatures. Presented at the Dagstuhl Workshop on Cryptography (1997)
6. Kipnis, A., Patarin, J., Goubin, L.: Unbalanced oil and vinegar signature schemes. EUROCRYPT 1999. LNCS **1592** (1999) 206–222
7. Patarin, J.: Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. In: EUROCRYPT. (1996) 33–48
8. Patarin, J., Goubin, L., Courtois, N.: C^*_{-+} and HM: Variations Around Two Schemes of T. Matsumoto and H. Imai. In Ohta, K., Pei, D., eds.: ASIACRYPT. Volume 1514 of Lecture Notes in Computer Science., Springer (1998) 35–49
9. Patarin, J., Courtois, N., Goubin, L.: Flash, a fast multivariate signature algorithm. CT-RSA 2001, LNCS **2020** (2001) 297–307
10. Kipnis, A., Shamir, A.: Cryptanalysis of the HFE public key cryptosystem by relinearization. Advances in Cryptology - CRYPTO 1999, Springer **1666** (1999) 788
11. Faugere, J.C.: Algebraic cryptanalysis of hidden field equations (HFE) using grobner bases. CRYPTO 2003, LNCS **2729** (2003) 44–60
12. Felke, P.: On the security of biquadratic C $_*$ public-key cryptosystems and its generalizations. Cryptography and Communications **11** (2019) 427–442
13. Bernstein, D.J., Yang, B.: Asymptotically faster quantum algorithms to solve multivariate quadratic equations. In Lange, T., Steinwandt, R., eds.: Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings. Volume 10786 of Lecture Notes in Computer Science., Springer (2018) 487–506

14. Smith-Tone, D.: Multivariate Cryptography. ProQuest (2010)
15. Dobbertin, H., Faugère, J., Felke, P.: Mystery twister crypto challenge 11 (2005) https://www-polsys.lip6.fr/ jcf/Papers/CC11_twister.pdf.
16. Patarin, J., Goubin, L., Courtois, N.: C^{*}_{-+} and HM: Variations around two schemes of T.Matsumoto and H.Imai. Asiacrypt 1998, Springer **1514** (1998) 35–49
17. Dubois, V., Fouque, P.A., Shamir, A., Stern, J.: Practical Cryptanalysis of SFLASH. In Menezes, A., ed.: CRYPTO. Volume 4622 of Lecture Notes in Computer Science., Springer (2007) 1–12
18. Smith-Tone, D.: On the differential security of multivariate public key cryptosystems. In Yang, B.Y., ed.: PQCrypto. Volume 7071 of Lecture Notes in Computer Science., Springer (2011) 130–142
19. Ding, J.: A new variant of the matsumoto-imai cryptosystem through perturbation. PKC 2004, LNCS **2947** (2004) 305–318
20. Smith-Tone, D.: Properties of the discrete differential with cryptographic applications. In Sendrier, N., ed.: PQCrypto. Volume 6061 of Lecture Notes in Computer Science., Springer (2010) 1–12