A "Differential" Attack on Polly Cracker

Dennis Hofheinz and Rainer Steinwandt

Abstract— **We describe an attack on the public key cryptosystem Polly Cracker for which it is not necessary to know a superset of the monomials used during encryption. In particular, the attack can be used to reveal "hidden" monomials and thereby increases the applicability of known linear algebra attacks on this system. The approach is demonstrated with Koblitz's "graph perfect code instance" of Polly Cracker.**

Keywords— **public key cryptography; cryptanalysis; multivariate polynomials**

I. Introduction

In [1], [2] M. Fellows and N. Koblitz describe the public key cryptosystem Polly Cracker. In this system the public key consists of multivariate polynomials having a (secret) common zero, and the public polynomials can be selected in such a way that revealing the private key is equivalent to finding a solution to an instance of an NP-complete problem.

The cryptosystems ENROOT [3] and ENROOT II [4] can be seen as special instances of Polly Cracker where sparse polynomials are used to dodge efficiency problems in the general scheme. Unfortunately, both ENROOT and ENROOT II have been cryptanalyzed successfully [5], [6]—the attacks exploit the fact that the monomials involved in the encryption can often be read off from the ciphertext. To avoid this kind of attack, it is tempting to "hide" some monomials as described by H. W. Lenstra (see [2, Ch. 5, §6]). Subsequently, we demonstrate that the latter approach does not necessarily guarantee acceptable cryptographic security. Namely, we describe a method for revealing "hidden" monomials, so that sometimes a linear algebra attack becomes feasible again. To illustrate the attack we use Koblitz's "graph perfect code instance" of Polly Cracker [2, Ch. 5, §7]; our experimental results give strong evidence that this cryptosystem is insecure.

II. The public key system Polly Cracker

Let $\mathbb{F}_{p^s}[x] := \mathbb{F}_{p^s}[x_1, \ldots, x_n]$ be the ring of multivariate polynomials over some finite field $\mathbb{F}_{p^s}$. Then Polly Cracker can be described as follows (for more details see [1], [2]):

- First, Alice selects $q_1, \ldots, q_r \in \mathbb{F}_{p^s}[x]$ with a common zero $\sigma \in \mathbb{F}_{p^s}^n$, i. e., $q_1(\sigma) = \ldots = q_r(\sigma) = 0$. The polynomials $q_1, \ldots, q_r$ are made public, while their common zero σ forms Alice's secret key.
- To encrypt a plaintext message $\alpha \in \mathbb{F}_{p^s}$, Bob first selects some element from the ideal generated by $q_1, \ldots, q_r$ in $\mathbb{F}_{p^s}[x]$, i. e., he selects polynomials $h_1, \ldots, h_r \in \mathbb{F}_{p^s}[x]$ and computes $\tilde{c} := \sum_{i=1}^{r} h_i \cdot q_i$. Then he computes the ciphertext as $c := \alpha + \tilde{c} \in \mathbb{F}_{p^s}[x]$.
- For decrypting the ciphertext c, Alice simply evaluates it at her secret key σ:

$$c(\sigma) = (\alpha + \tilde{c})(\sigma) = \alpha + \sum_{i=1}^{r} h_i(\sigma) \cdot q_i(\sigma) = \alpha$$

Of course, the actual choice of the parameters is crucial for the security of the resulting system. Fellows and Koblitz suggest to construct the polynomials $q_1, \ldots, q_r \in \mathbb{F}_2[x]$ in such a way that finding a common zero σ of them is equivalent to solving an instance of an NP-complete problem. However, they do not specify a concrete key generation procedure; in the sequel we assume that finding a common zero of the public polynomials is not feasible for the attacker. Moreover, we leave aside the question of side-channel attacks against Polly Cracker (cf. [7]). Instead, we focus on the security of the encryption procedure executed by Bob.

III. Linear algebra attacks

As remarked in [7] already, for revealing the plaintext $\alpha \in \mathbb{F}_{p^s}$, it is sufficient for an attacker to recover the constant terms $h_1(0), \ldots, h_r(0)$ used by Bob, because we have

$$\alpha = c(0) - \sum_{i=1}^{r} h_i(0) \cdot q_i(0).$$

This simple observation is quite important: assume that each q_i contains a "characteristic monomial", i. e., a monomial[1] $m_i := x_1^{\nu_1} \cdot \ldots \cdot x_n^{\nu_n}$ with q_i being the only public polynomial where m_i occurs with non-zero coefficient. If Bob does not choose the h_i carefully, then such characteristic monomials can enable an attacker to read off $h_i(0)$ from the ciphertext c (cf. [5], [7]). To avoid such an attack one may think of choosing the public polynomials in such a way, that several of them consist of the same monomials, i. e., only the values of the non-zero coefficients of the monomials vary. However, this approach still does not rule out an "intelligent linear algebra attack" like the following one (cf. [2, Ch. 5, §6]):

assume that for each monomial m_h occurring in one of Bob's polynomials h_i there is a monomial m_c in the ciphertext c and a monomial m_q in some (public) q_j such that $m_c = m_h \cdot m_q$. Then an attacker can easily determine a (comparatively small) superset $\mathcal{M}$ of the set of monomials occuring in the polynomials h_i. If such a set is known, decrypting the ciphertext reduces to solving a system of linear equations over $\mathbb{F}_{p^s}$: for $1 \leq i \leq r$, $m \in \mathcal{M}$ let A_{im} be indeterminates. Then comparing coefficients in the

The authors are with the Institut für Algorithmen und Kognitive Systeme (IAKS), Professor Dr. Th. Beth, Arbeitsgruppen Computeralgebra & Systemsicherheit, Universität Karlsruhe, Germany.
E-mail: {hofheinz,steinwan}@ira.uka.de.

[1] We adopt the convention that a monomial is a monic term, i. e., each polynomial can be written as a sum of terms.

equation

$$c = A_0 + \sum_{i=1}^{r}\Big(\sum_{m\in\mathcal{M}} A_{im}\cdot m\Big)\cdot q_i$$

(with A_0 an indeterminate for the unknown plaintext) yields a linear system of equations in the indeterminates A_{im} and A_0. By construction this system of equations is solvable, and each solution yields the correct plaintext $A_0 = \alpha$.

For the practicability of such a linear algebra attack it is crucial for the adversary to know a (small) set $\mathcal{M}$ with

$$\mathcal{M} \supseteq \bigcup_{1\le i\le r} \mathrm{M}(h_i)$$

(where $\mathrm{M}(h_i)$ is the set of monomials occuring in the polynomial h_i with non-zero coefficient). So how can Bob keep the attacker from learning the monomials used during the encryption? In [2, Ch. 5, §6] the following idea is given:
"…, Bob must artfully build at least one monomial d' into at least one h_j such that d' times *any* term in q_j is canceled in the entire sum (so that it doesn't occur in C [(the set of monomials occuring in the ciphertext c)]). Also, the monomials d' with that property should not be too few and/or too easy to guess, …"

The attack in the next section aims at revealing monomials "hidden" during the encryption. We demonstrate it through an example, and thereafter apply it to the encryption technique used in Koblitz's "graph perfect code instance" of Polly Cracker (see [2, Ch. 5, §7]).

IV. Exploiting the structure of the ciphertext

Our main tool is a function Δ which to each polynomial in $\mathbb{F}_{p^s}[x]$ associates a set of ("difference") terms in the ring $\mathbb{F}_{p^s}[x, x^{-1}] := \mathbb{F}_{p^s}[x_1, x_1^{-1}, \ldots, x_n, x_n^{-1}]$ of Laurent polynomials. To define Δ we make use of the following lexicographic order $\succeq$ on $\mathbb{N}_0^n$: $\mu \succeq \eta$ iff $\mu = \eta$ or the left-most non-zero entry of $\mu - \eta := (\mu_1 - \eta_1, \ldots, \mu_n - \eta_n)$ is positive. Actually, we could use an arbitrary monomial ordering on $\mathbb{N}_0^n$ (cf., e. g., [8, Ch. 2, §2]), but here we stick to $\succeq$. Now Δ is defined as follows:

$$\begin{array}{rccc} \Delta: & \mathbb{F}_{p^s}[x] & \longrightarrow & 2^{\mathbb{F}_{p^s}[x,x^{-1}]} \\ & \sum\limits_{\nu\in\mathbb{N}_0^n} \gamma_\nu\cdot x^\nu & \longmapsto & \Big\{\frac{\gamma_\mu}{\gamma_\eta}\cdot x^{\mu-\eta}\,\big|\,\mu\succ\eta,\ \gamma_\mu\cdot\gamma_\eta\neq 0\Big\} \end{array}$$

The following properties of Δ can be verified easily:

Remark 1: Let $a, b \in \mathbb{F}_{p^s}[x]$ be polynomials that have no monomial in common, i. e., $\mathrm{M}(a)\cap\mathrm{M}(b) = \emptyset$. Moreover, let $\gamma_\nu\cdot x^\nu \in \mathbb{F}_{p^s}[x]$ be a non-zero term and Δ as above. Then
(i) $\Delta(a) = \emptyset$ iff a is a term or $a = 0$
(ii) $|\Delta(a)| \le \frac{|\mathrm{M}(a)|^2 - |\mathrm{M}(a)|}{2}$
(iii) $\Delta(a) = \Delta(\gamma_\nu x^\nu\cdot a)$
(iv) $\Delta(a+b) \supseteq \Delta(a)\cup\Delta(b)$

Proof:
(i) "⇒": Assume that a contains at least two non-zero terms $\gamma_\mu\cdot x^\mu, \gamma_\eta\cdot x^\eta$. Say $\mu\succ\eta$, then $(\gamma_\mu/\gamma_\eta)\cdot x^{\mu-\eta}\in\Delta(a)$. "⇐": trivial.
(ii) As $\succeq$ induces a total ordering on the terms occuring in a, we have

$$|\Delta(a)| \le \sum_{i=1}^{|\mathrm{M}(a)|-1} i,$$

and the latter sum evaluates to $(|\mathrm{M}(a)|^2 - |\mathrm{M}(a)|)/2$ as required.
(iii) Immediate from the definition of Δ, as $\preceq$ is a monomial ordering.
(iv) Immediate from the assumption $\mathrm{M}(a)\cap\mathrm{M}(b) = \emptyset$. □

Now let $c = \alpha + \sum_{i=1}^r h_i\cdot q_i$ be some ciphertext computed by Bob. Then from Remark 1 (iii)–(iv) we conclude that with some luck there is an $i\in\{1,\ldots,r\}$ satisfying

$$\Delta(q_i)\cap\Delta(c)\neq\emptyset.$$

Assume that for some $i\in\{1,\ldots,r\}$ there exist terms $\gamma_{\mu_i}x^{\mu_i}$, $\gamma_{\nu_i}x^{\nu_i}$ in q_i such that
- there is a

$$\delta_i := \gamma_{\mu_i}x^{\mu_i}/\gamma_{\nu_i}x^{\nu_i} \in \Delta(q_i)\setminus\left(\bigcup_{j\neq i}\Delta(q_j)\right), \text{ and} \qquad (1)$$

- there is a term $\gamma_{\eta_i}x^{\eta_i}$ in h_i such that $x^{\eta_i}\cdot x^{\mu_i}$ and $x^{\eta_i}\cdot x^{\nu_i}$ do not occur among the monomials of $c - \gamma_{\eta_i}x^{\eta_i}\cdot q_j$.

The first condition means that the public polynomial q_i possesses a "characteristic term difference". The idea of the second condition is to exclude "collisions" in the coefficients of the monomials $x^{\eta_i}\cdot x^{\mu_i}$ and $x^{\eta_i}\cdot x^{\nu_i}$ during the computation of the ciphertext $c = \alpha + \sum_{i=1}^r h_i\cdot q_i$. In particular, the second condition guarantees that

$$\delta_i = \frac{\gamma_{\eta_i}x^{\eta_i}\cdot\gamma_{\mu_i}x^{\mu_i}}{\gamma_{\eta_i}x^{\eta_i}\cdot\gamma_{\nu_i}x^{\nu_i}} \in \Delta(c). \qquad (2)$$

Note that for this conclusion it is not relevant whether the term x^{η_i} itself occurs somewhere in the ciphertext c.

Now, if for some $1\le i\le r$ an attacker can find terms t_1, t_2 in the ciphertext with $x^{\mu_i}\mid t_1$ and the quotient t_1/t_2 being equal to a characteristic difference (1) of q_i, then he can make the assumption that both of the above conditions hold, and with (2) he can identify a potential term t_h of h_i as

$$t_h := \frac{t_1}{\gamma_{\mu_i}x^{\mu_i}} = \frac{t_2}{\gamma_{\nu_i}x^{\nu_i}}.$$

Although the attacker cannot be sure about the correctness of his guess—i. e., whether t_h is indeed a term of h_i—by replacing c with $c' := c - t_h\cdot q_i$, he obtains another valid encryption of the plaintext represented by c, and if the number of terms in c' is smaller than in c, then this can be taken as evidence of the correctness of the guess. In particular, if $c'\in\mathbb{F}_{p^s}$, then the encrypted plaintext has been recovered successfully.

The number of terms in a "simplified ciphertext" $c - t_h\cdot q_i$ can also serve as guidance if there are several possible terms t_1, t_2 in c for producing a "characteristic" δ_i. In this case it seems sensible to choose the term t_h which yields the

largest reduction in the number of terms when subtracting $t_h \cdot h_i$ from the ciphertext. After having simplified the ciphertext in the above manner, the attacker can try to iterate the procedure or check whether to c' an intelligent linear algebra attack can be applied.

The following toy example illustrates the technique just described:

Example 1: Alice publishes the following polynomials from $\mathbb{F}_{53}[x_1, x_2]$:

$$\begin{aligned} q_1 &:= 25x_1^{30} + 10x_1x_2 + 11 \\ q_2 &:= -31x_2^{14} + 5x_1x_2 + 32 \end{aligned}$$

(with (secret) common zero $(x_1, x_2) = (5, 7)$). Next, Bob chooses polynomials

$$\begin{aligned} h_1 &:= 17x_1^5x_2^{21} - 13x_1^6x_2^8 \\ h_2 &:= 18x_1^5x_2^7 + 12x_1^4x_2^6 \end{aligned}$$

for encrypting the plaintext message $18 \in \mathbb{F}_{53}$, and obtains the following ciphertext $c = 18 + h_1q_1 + h_2q_2$:

$$18 + x_1^{35}x_2^{21} - 7x_1^{36}x_2^8 + 11x_1^6x_2^{22} - x_1^4x_2^{20} - 24x_1^7x_2^9 + 13x_1^4x_2^6$$

Note that the monomial $x_1^5x_2^7$ from h_2 is not contained in the set $\{m_c/m_q : m_c \in \mathrm{M}(c), m_q \in \mathrm{M}(q_1) \cup \mathrm{M}(q_2)\}$. Consequently, the intelligent linear algebra attack mentioned in Section III does not apply. However, the term $4x_2^{14} = -31x_2^{14}/32 \in \Delta(q_2) \setminus \Delta(q_1)$ is also contained in $\Delta(c)$:

$$4x_2^{14} = -x_1^4x_2^{20}/(13x_1^4x_2^6)$$

Subtracting $(13x_1^4x_2^6/32) \cdot q_2$ from c yields the (simplified) ciphertext

$$c' = 18 + x_1^{35}x_2^{21} - 7x_1^{36}x_2^8 + 11x_1^6x_2^{22} - 24x_1^7x_2^9 - 7x_1^5x_2^7$$

to which the intelligent linear algebra attack from Section III can be applied successfully.

To get an idea of the practical value of the above approach, in the next section we apply a variant of it to the encryption procedure of Koblitz's graph perfect code instance of Polly Cracker (see [2, Ch. 5, §7]). To our knowledge, the latter is the only suggested encryption procedure for Polly Cracker that has not been broken yet.

V. Koblitz's graph perfect code instance

To derive a private key-public key pair, Alice chooses a 3-regular (undirected) graph $G = (V, E)$ such that there is a subset $V' \subseteq V$ of the vertices that forms a perfect code. In other words, each vertex $v \in V$ of G is in the neighbourhood $\mathrm{N}(v')$ of one and only one $v' \in V'$ (here $\mathrm{N}(v')$ is the set consisting of v' and all vertices joined to v' by an edge). From this graph the public polynomials can be derived easily as described in [2, Ch. 5, §3]. For the sequel it is sufficient to know that Alice's public polynomials q_i are contained in $\mathbb{F}_2[x] = \mathbb{F}_2[x_1, \ldots, x_n]$ where n is equal to the number of vertices of G. Koblitz suggests a value of $n \approx 500$ and to represent the ciphertext c as a polynomial of degree $d \approx 2\log_2 n \approx 18$. For a detailed description of the encryption procedure we refer to [2, Ch. 5, §7]; here it is sufficient to mention that in addition to d and n also a third parameter $d_0 \approx d/3$ is used for controlling details of the encryption procedure.

All the same, for the security of the resulting cryptosystem it is crucial that the attacker cannot find the perfect code V' from G. As described in [7], on the one hand choosing the graphs G at random does not ensure acceptable security, and on the other hand no specification of how to construct secure keys has been given. Lacking such a specification, in our experiments we used random graphs, but ignored the possible feasibility of a direct attack on the secret key V'. Instead, we restricted ourselves to attacking individual plaintexts through the approach from Section IV.

Experimentally, Koblitz's suggested parameter values turn out to be rather cumbersome: already for $n = 128$ ($d = 14$, $d_0 = 5$) for encoding a single plaintext bit $\alpha \in \mathbb{F}_2$ the ciphertext consists of some 60,000 terms. If each term is represented by 7 ($= \log_2 128$) bytes, this means that a single bit of plaintext translates into about 410 KBytes of ciphertext! For $n = 160$ ($d = 15$, $d_0 = 5$) already about 250,000 terms are used for encoding a single plaintext bit, and with $n = 200$ ($d = 15$, $d_0 = 5$) the encryption procedure yields a ciphertext with more than $500,000$ terms. With the suggested value $n = 500$ already performing one complete encryption of a plaintext bit is a tremendous effort. With the huge ciphertexts required here, on a "normal" PC practical experiments with our attack do not really make sense: $|\Delta(c)|$ is in the magnitude of $|\mathrm{M}(c)|^2$ (cf. Remark 1 (ii)), i. e., the computation of $\Delta(c)$ becomes rather cumbersome then. We do not consider this as a serious weakness of our approach, because for $n = 500$ the message expansion occurring in the considered cryptosystem makes it already rather hard to implement encryption at all.

In the sequel we restrict to cases with $100 \leq n \leq 200$ where a ciphertext (encrypting one plaintext bit) consists of "only" several thousand terms.—Of course, this message expansion is also far from being acceptable for a realistic public key cryptosystem, but at least such ciphertexts can be dealt with easily on a "normal" PC. For the experiments we used a 1,333 Mhz Linux PC with a GNU C compiler. For a given ciphertext $c \in \mathbb{F}_2[x]$ the attack performs the following steps:

1. Pick one of the (public) polynomials q_i of the form $1 - \sum_{u \in \mathrm{N}[v]} x_u$ (cf. [2, Ch. 5, §3]) randomly.
2. For each term t_c in c and each term t_q in q_i with $t_q | t_c$ check if subtracting $\mathrm{R}(q_i \cdot \frac{t_c}{t_q})$ reduces c in size (i.e., in the number of terms; see [2, Ch. 5, §7] for a definition of the reduction operator R).
3. If so, iterate the attack with the "simplified" ciphertext from step 2 until we end up with a constant c.
4. Otherwise proceed with step 2, and eventually skip back to step 1.

Note that for reasons of efficiency $\Delta(c)$ is not explicitly computed. Rather we are "guessing" potential terms t_c/t_q in h_i in a quite canonical way. (Interestingly it turns out

that the above algorithm does *not* reconstruct the polynomials h_i used during encryption completely. On the contrary, attacking a ciphertext which results from an encryption of the plaintext bit $\alpha = 0$ yields a large combination of public key-polynomials as a "simplified" ciphertext which expands to the zero polynomial in most cases, as experimental results suggest.) We did not combine our approach with a linear algebra based attack either; but the experimental results demonstrate that already the above simple procedure is extraordinary successful when the encrypted plaintext bit is 0:

- For $n = 100$ ($d = 13$, $d_0 = 4$), in $1,000$ encryptions of the plaintext $\alpha = 0$ we could 951 times reveal α successfully.
- For $n = 128$ ($d = 14$, $d_0 = 5$) in 100 encryptions of $\alpha = 0$ we could 94 times reveal α successfully.
- For $n = 160$ ($d = 15$, $d_0 = 5$) in 30 encryptions of $\alpha = 0$ we could 28 times reveal α successfully.
- For $n = 200$ ($d = 15$, $d_0 = 5$) from a ciphertext with $575,182$ terms the plaintext $\alpha = 0$ could be recovered successfully in ≈ 8.75 hours.

We did not check whether these results can be improved through "post-processing" with an intelligent linear algebra attack. Anyway, according to these experiments a failure of our attack implies with high probability that the encrypted plaintext bit is 1. In other words, we have quite a good chance to recover the plaintext bit successfully without needing the secret key.

VI. Conclusion

A new attack on Polly Cracker for dealing with "hidden" monomials has been described; in particular the attack can be used to enhance the feasibility of a linear algebra based attack. The experimental results with Koblitz's graph perfect code instance give strong evidence that this cryptosystem does not offer acceptable cryptographic security. It remains unclear whether practical and secure instances of Polly Cracker can be constructed.

References

[1] Michael Fellows and Neal Koblitz, "Combinatorial cryptosystems galore!," in *Finite Fields: Theory, Applications and Algorithms, Proceedings of Second International Conference on Finite Fields*, Gary L. Mullen and Jau-Shyong Shiue, Eds. 1994, number 168 in Contemporary Mathematics, pp. 51–61, American Mathematical Society.

[2] Neal Koblitz, *Algebraic Aspects of Cryptography*, vol. 3 of *Algorithms and Computations in Mathematics*, Springer-Verlag, 1998.

[3] David Grant, Kate Krastev, Daniel Lieman, and Igor Shparlinski, "A public key cryptosystem based on sparse polynomials," in *International Conference on Coding Theory, Cryptography and Related Areas, Proceedings of ICCC 1998*, Johannes Buchmann, Tom Høholdt, Henning Stichtenoth, and Horacio Tapia-Recillas, Eds., 2000, pp. 114–121, Online available at `http://www.comp.mq.edu.au/~igor/GKLS.ps`.

[4] William D. Banks, Daniel Lieman, Igor E. Shparlinski, and Van Thuong To, "Cryptographic applications of sparse polynomials over finite rings," in *Information Security and Cryptology, Proceedings of ICISC 2000*, Dongho Won, Ed. 2001, number 2015 in Lecture Notes in Computer Science, pp. 206–220, Springer-Verlag.

[5] Feng Bao, Robert H. Deng, Claus Schnorr, Rainer Steinwandt, and Hongjun Wu, "Cryptanalysis of two sparse polynomial based public key cryptosystems," in *Public Key Cryptography, Proceedings of PKC 2001*, Kwangjo Kim, Ed. 2001, number 1992 in Lecture Notes in Computer Science, pp. 153–164, Springer-Verlag, Online available at `http://icsd.i2r.a-star.edu.sg/publications/BaoFeng_19920153.pdf`.

[6] Feng Bao, Thomas Beth, Robert H. Deng, Claus Schnorr, Rainer Steinwandt, and Hongjun Wu, "Cryptanalysis of SPIFI II and ENROOT II," in *Security through Analysis and Verification*, number 294 in Dagstuhl Seminar Report, p. 7. Schloss Dagstuhl, 2000, Abstract, online available at `ftp://ftp.dagstuhl.de/pub/Reports/00/00501.ps`.

[7] Rainer Steinwandt, Willi Geiselmann, and Regine Endsuleit, "Attacking a polynomial-based cryptosystem: Polly Cracker," *International Journal of Information Security*, vol. 1, no. 3, pp. 143–148, Nov. 2002, Online available at `http://iaks-www.ira.uka.de/home/endsuleit/publications/polly.ps`.

[8] David Cox, John Little, and Donal O'Shea, *Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, Undergraduate Texts in Mathematics. Springer-Verlag, 1992.