Le Van Ly

Polly two: a new algebraic polynomial-based Public-Key Scheme

Abstract Since Fellows and Koblitz introduced the generic combinatorially algebraic public-key cryptosystem Polly Cracker in 1993, the question whether there exist secure and efficient instances of Polly Cracker remains unsettled. Namely, many of these schemes succumb to the Linear-Algebra Attacks by D. Naccache et al. and Lenstra. In this paper I present a public-key cryptoscheme Polly Two that is efficient and in some way a modified Polly-Cracker instance, but comprises techniques defending the linear-algebra attacks.

Keywords Combinatorially algebraic public-key cryptoschemes · Polly Cracker · EnRoot · Linear-Algebra attacks · Sparse polynomials · System of algebraic equations · Elimination ideals

Introduction

In 1993 Fellows [6] and Koblitz [11] presented a general public-key cryptoscheme called Polly Cracker (section 5.3) that is of general relevance, because an instance of it can be constructed for any NP-Search Problem so that the security with regard to the secret key is equivalent to the hardness of solving the corresponding NP-instance. Koblitz [11] illustrates this by the NP-Problems: Graph 3-Coloring and Graph Perfect Code.

Ignoring the combinatorial background, Polly Cracker works as follows. The public key is an ideal $\mathfrak{a}$ in the polynomial ring $P = \mathbb{F}[x_1, \ldots, x_n]$ over a finite field $\mathbb{F}$ in $n$ indeterminates. For instance, the ideal is given by generators: $\mathfrak{a} = (f_1, \ldots, f_s)$. The secret key is a point $\xi \in \mathbb{F}^n$ that is a zero of $\mathfrak{a}$, i.e., $f(\xi) = 0$ for every $f \in \mathfrak{a}$. To
encrypt a message \( m \in \mathbb{F} \), we choose an element \( h \) from the ideal \( \mathfrak{a} \), for instance, by selecting polynomials \( h_1, \ldots, h_s \in P \) and putting \( h = \sum_{i=1}^{s} h_i f_i \). The ciphertext is \( c = h + m \). Decryption is simply the evaluation of \( c \) at \( \xi \), because \( h \) vanishes at \( \xi \). Security with respect to the secret key resides on the NP-hardness of computing a solution of a system of algebraic equations.

In this abstract version, the semantic security of Polly Cracker holds without doubt, as it is equivalent to the NP-hard Ideal-Membership Problem (IMP) [10]. However, bridging the gap between theory and practice while maintaining the hardness of the ideal-membership problem turned out to be more challenging as it may look like at the first glance. To see this, we briefly describe the method of the Linear-Algebra Attacks, which successfully break some straightforward instances of Polly-Cracker.

The scheme Polly Cracker works over the polynomial ring \( P \). Representing polynomials by their coefficients and monomials, efficiency considerations restrict us to two cases: dense polynomials of small degree and sparse polynomials of high degree.

D. Naccache et al. [3] describe a Linear-Algebra Attack in the dense case. To break a dense Polly-Cracker instance they put up a system of linear equations by comparing coefficients of the equation \( c = \sum_{i=1}^{s} h_i f_i + m \) so that the coefficients of \( h_i \) and \( m \) are the unknowns. The reason why this attack works is that, if we represent polynomials in dense way, we use them as if they were vectors and our encryption function was just a mapping between vector spaces. It is clear that in such a case linear algebra methods easily invert the encryption function in polynomial time in the dimension of the vector space, which is in fact our complexity for representing the polynomials.

If we work with sparse polynomials of high degree over a large finite field, we represent these polynomials by their coefficients and monomials in the support. Certainly the linear attack in its original version does not apply at once, but in some way the polynomials \( h_i \)’s “shine through” the ciphertext. H. W. Lenstra exploited this fact in the so-called Intelligent Linear-Algebra Attack ([11], p. 114). More precisely, as the encryptor has to ensure that the representation of the ciphertext is still short, he is restricted to a small set of polynomials for the \( h_i \)’s. For instance, to use sparse \( h_i \)’s for encryption is one way to provide sparsity of the ciphertext. But, as the field is large and the degrees are high, the probability that some monomials coincide or cancel one another in the ciphertext is quite small. Hence, an attacker is able to guess the monomials of the \( h_i \)’s, to put up his linear attack in these monomials, and in the majority of cases to reveal the message in polynomial time in the ciphertext length.

This attack breaks, for instance, the proposal EnRoot of Grant et al. [8], which is a sparse instance of Polly Cracker. More precisely, the scheme works with three indeterminates and a field of size about \( 2^{31} - 1 \). Their strategy for selecting the public key polynomials \( f_i \) and the encrypting polynomials \( h_i \) is random choice. Hence, this system is vulnerable to Lenstra’s attack. If even stronger conditions on the support of the public key hold then more sophisticated linear attacks by Bao et al. [1] and Endsuleit et al. [5] apply, as well.

While it is easy to find polynomials so that these sophisticated linear attacks do not work ([12], p.36), it is much harder to get instances that also resist Lenstra’s attack. One idea is as follows: construct the \( f_i \)’s in such a way that is easy