Chapter 7
TCHo: A Code-Based Cryptosystem

Alexandre Duc and Serge Vaudenay

Abstract TCHo is a public-key cryptosystem based on the hardness of finding a multiple polynomial with low weight and on the hardness of distinguishing between the output of an LFSR with noise and some random source. An early version was proposed in 2006 by Finiasz and Vaudenay with non-polynomial (though practical) decryption time. The latest version came in 2007 with more co-authors. It reached competitive (heuristic) polynomial complexity and IND-CPA security. Since then, a key-recovery chosen ciphertext attack was published by Herrmann and Leander in 2009. In this paper we review the state of the art on this cryptosystem, together with some latest improvements regarding implementation and selection of parameters. We provide also more formal results regarding correctness and we update the key generation algorithm.

7.1 Introduction

Public-key cryptography first appeared with the seminal paper of Diffie and Hellman in 1976 [21]. From this work, Rivest, Shamir, and Adleman presented the RSA cryptosystem in 1978 [60], which is still the mostly used one nowadays. Among the popular cryptosystems, there are the Rabin cryptosystem [56], which is very close to RSA, and the ElGamal family of cryptosystems [25].

Every public-key cryptosystem relies on problems that are believed to be computationally infeasible. As far as we know, all cryptosystems which are used in practice rely on two problems: the integer factorization problem [56, 60] and the discrete logarithm problem [25]. However, these two problems can easily be solved

A. Duc (✉) • S. Vaudenay
EPFL, Lausanne, Switzerland
e-mail: alexandre.duc@epfl.ch

in polynomial time on a *quantum computer* using Shor's algorithm [62] and its generalizations [32]. Hence, if one can build a quantum computer with sufficiently many qubits to solve these problems, the mostly used public-key cryptographic systems will be broken and will have to be replaced.

To be prepared for this, we need *crypto-diversity*. Then, if one cryptosystem is broken, another ideally well-studied cryptosystem will be available for use. In particular, some of these systems should be secure on quantum computers as well. Such cryptosystems are referred to *post-quantum cryptosystems*. Nowadays, this has become a hot topic and dozens of quantum-resistant schemes have been designed.

Several types of post-quantum cryptosystems have been proposed. Some are based on multivariate equations [22, 36, 45, 52–54], whereas some others are code-based [3, 29, 46, 50] or lattice-based [1, 2, 30, 34, 44, 55, 57, 58]. The former are, to the best of our knowledge, used only to design signature schemes. In the following, we focus on *code-based* and *lattice-based* cryptosystems.

7.1.1 Code-Based Cryptosystems

Code-based cryptosystems rely on error correcting codes and the addition of random noise during the encryption.

The most famous code-based cryptosystem is the McEliece cryptosystem [46]. It was introduced by McEliece in 1978 and is still unbroken. In this scheme, the private key is the generator matrix G of a random $[k,n]$-Goppa code able to correct up to t errors along with two matrices P and S, where P is a random $k \times k$ permutation and S a random $n \times n$ non-singular matrix. The public key is then a scrambled version $\hat{G}$ of G, defined as $\hat{G} := SGP$. A message is encrypted by first encoding it with the code associated to $\hat{G}$ and by adding a random noise with exactly t ones. The security of the system relies on the hardness of decoding a scrambled Goppa code. This problem is also known as the *McEliece problem*. Decryption of a ciphertext can be performed efficiently if one is in possession of P, S, and G, since Goppa codes are efficiently decodable.

Niderreiter described a dual variant of the McEliece cryptosystem [50]. Instead of representing the message as a codeword, the encryption is performed with the parity check matrix H of a Goppa code. The security of the two schemes have been shown to be equivalent [39]. A signature scheme [19] was derived from the Niderreiter cryptosystem by Courtois et al.

A code-based symmetric key cipher, LPN-C [29], was introduced by Gilbert et al. This cipher is based on another problem which is believed to be hard: the *Learning from Parity with Noise* problem (LPN). The LPN problem consists in finding an unknown k-bit vector $\mathbf{x}$ given access to an oracle that returns $(\mathbf{a}, \mathbf{a} \cdot \mathbf{x} + \nu)$, for some biased noise ν and some random vector $\mathbf{a}$.

The main drawback of these schemes is that the key length needed to obtain reasonable security is pretty large. For the McEliece cryptosystem, public keys of size 2^{16} achieve only 84.88-bit security [9]. To obtain 266.94-bit security, 2^{20}-bit keys are needed. Note that, asymptotically, the McEliece cryptosystem has better key sizes than RSA. For a security parameter λ, a McEliece key has size $\lambda^{2+o(1)}$ while RSA keys have size $\lambda^{3+o(1)}$ [8], but this holds only for impractical λ. Another drawback of these schemes is that the ciphertext length has to be bigger than the plaintext. This comes from the use of an error-correcting code and cannot be changed.

TCHo belongs also to the code-based cryptosystem category. However, its security rely on a somewhat different problem: the low weight polynomial multiple problem.

7.1.2 Lattice-Based Cryptosystems

The other category of post-quantum cryptosystems are lattice-based cryptosystems. One of the strengths of lattice-based cryptography is that its security is often based on the *worst-case hardness* of problems instead of average-case hardness.

One of the computationally hard problems on which lattice-based systems rely on is the *Shortest Vector Problem* (SVP). Briefly, this problem consists of finding the shortest non-zero vector in a lattice or an approximation of it within a polynomial factor. Polynomial algorithms like LLL [38] or its improvements [61] can only find subexponential approximations of it.

Lattice-based cryptography was introduced by Ajtai in 1996 [1]. Shortly after, Ajtai and Dwork designed the first lattice-based public-key cryptosystem based on the worst-case hardness of SVP [2]. This scheme was later improved in [30, 57]. However, they suffer from large key sizes and a large expansion factor, and are inefficient in practice. Indeed, for a lattice of dimension n, the keys in this scheme have size $\widetilde{O}\left(n^4\right)$ and the ciphertexts $\widetilde{O}\left(n^2\right)$ [59].[1]

Another class of lattice-based cryptosystems are based on the worst-case complexity of the *Learning With Errors* (LWE) problem [44,55,58]. The LWE problem is the following: given a dimension n, a modulus q and an error distribution χ over $\mathbb{F}_q$, the goal is to find a secret vector $s \in \mathbb{F}_q^n$ using independent LWE-samples:

$$(a, \langle a, s\rangle + \varepsilon) \in \mathbb{F}_q^n \times \mathbb{F}_q, \quad a \overset{U}{\leftarrow} \mathbb{F}_q^n, \quad \varepsilon \overset{\chi}{\leftarrow} \mathbb{F}_q.$$

Reference [44] introduces the *ring-LWE* problem, an algebraic variant of the LWE problem. According to the authors, it is the first truly practical lattice-based cryptosystem based on LWE.

The most famous and efficient lattice-based cryptosystem is NTRU [34] which is based on the work of Goldreich et al. [31]. Its security is based on the hardness

[1]A function is $\widetilde{O}(f(n))$ if it is $O\left(f(n) \cdot \log(f(n))^k\right)$ for some k.

of SVP and the Closest Vector Problem (CVP) in convolution modular lattices, a particular class of lattices. Unlike some schemes we named above, NTRU's security has not been shown equivalent to the hardness of SVP or CVP. Nevertheless, for a security parameter λ, the asymptotic cost for encryption and decryption is $O\left(\lambda^2\right)$ and the key sizes is $O(\lambda)$ which makes of NTRU one of the most efficient public-key cryptosystems.

7.1.3 TCHo

It is often the case in stream cipher cryptanalysis that we need to cancel the effect of a linear feedback shift register with noise by using a low weight multiple of its connection polynomial. This happens in fast correlation attacks [47]. For instance, in the cryptanalysis of Bluetooth E0 [40–43], we need to find such a multiple with low weight for a given polynomial coming from the E0 specifications. Actually, the lower the degree and the weight, the more efficient the attack. This happens to be hard in practice. However, the designer of E0 could have selected the polynomial as a factor of some secret low weight multiple. That is, a trapdoor could have been hidden to break the cipher. Refining this idea, in 2006, Finiasz and Vaudenay [26] came up with the notion of *trapdoor cipher* on which TCHo is based. Indeed, the name "TCHo" stands for "Trapdoor Cipher, Hardware Oriented".[2] This early version of TCHo was using a linear code based on another LFSR.

One drawback of this design was that decryption (using the trapdoor) was not polynomially bounded, although still feasible in practice. Then, Meier suggested using other codes. Finally, in 2007, Aumasson et al. presented the latest version [3] with polynomial complexity using heuristic algorithms. They proved semantic security based on some new complexity assumptions. They further proposed to apply the Fujisaki-Okamoto construction [27] to achieve IND-CCA security.

In 2009, Herrmann and Leander [33] have shown that we can mount a key recovery chosen ciphertext attack, which seemingly proves that the key recovery problem and the decryption problem are somewhat equivalent.

Since then, Duc [23] has shown how to generate better parameter vectors, and Bindschaedler [10] implemented it as a new cipher in TLS for a browser and an HTTP server.

In this paper, we survey known results about TCHo. Additionally, we provide more formal (i.e. non-heuristic) results regarding correctness, with an updated key generation algorithm.

[2]The word "tchô" happens to come from some French slang which originated from the famous Swiss cartoonist Zep who created a comics magazine for kids with this name in 1998.

7.1.4 Structure of This Paper

In Sect. 7.2 we describe our notation and give basic definitions used throughout the paper. In Sect. 7.3 we present the problems on which the security of TCHo relies on and we survey algorithms that solve them. The complexity of these algorithm will be needed to find secure parameters for TCHo. In Sect. 7.4 we present the TCHo cipher and prove that it is a cryptosystem with heuristic key generation. In Sect. 7.5 we discuss the security of TCHo, we prove that TCHo is IND-CPA secure and we show how to achieve IND-CCA security. In Sect. 7.6 we give some practical parameters for TCHo and we discuss some implementation results. We conclude in Sect. 7.7.

7.2 Notations and Definitions

We denote by "log" the logarithm in base two and by "ln" the natural logarithm. We write $x \overset{U}{\leftarrow} \mathcal{D}$ if an element x is drawn uniformly at random in a domain $\mathcal{D}$. We write $x \overset{\chi}{\leftarrow} \mathcal{D}$ if x is drawn from domain $\mathcal{D}$ using distribution χ. For TCHo, we consider only binary polynomials, i.e., polynomials with coefficients in $\mathbb{F}_2$. The *degree* of a polynomial $P \in \mathbb{F}_2[x]$ is denoted d_P. We use uppercase characters to represent polynomials and the same letter in lowercase to represent its coefficients. Hence, we write $P = p_0 + p_1X + p_2X^2 + \cdots + p_{d_P}X^{d_P}$. The number of nonzero coefficients of P is called the *weight* of the polynomial and is denoted w_P. In other words, $w_P = \sum_{i=0}^{d_P} p_i$, where p_i's are considered as elements in $\mathbb{Z}$. A polynomial P with $w_P \ll d_P$ is called a *sparse* polynomial or a *low weight* polynomial.

The *bias* γ of a random bit B is the difference between the probability of occurrence of a zero and the probability of occurrence of a one, i.e., $\gamma = \Pr[B = 0] - \Pr[B = 1]$. Hence, a source producing random bits with bias γ outputs a zero with probability $\frac{1}{2}(1+\gamma)$ and a one with probability $\frac{1}{2}(1-\gamma)$. We call a finite sequence of bits x a *bitstring*. We write its *length* $|x|$, which denotes its number of bits. As for polynomials, we call the weight of a bitstring its number of ones. The concatenation of two bitstrings x and y is written $x\|y$. The (possibly infinite) output of a *bitsource* S is called a *bitstream*. If we need to specify the input (e.g. the *seed*) r of a source S, we write $\mathsf{S}(r)$. The bitstring constructed from the first ℓ bits produced by S is denoted S^{ℓ}. We denote by S_γ a bitsource producing independent bits with bias γ. Given a bitstring x, we denote by $\mathsf{trunc}_\ell(x)$ the substring of x made by its first ℓ bits.

Given some initial parameters Π and a predicate P, we write

$$\Pr\left[P(v_1,\ldots,v_m;r_p) : \begin{array}{l} v_1 \leftarrow f_1(\Pi;r_1) \\ \vdots \\ v_m \leftarrow f_m(\Pi,v_1,\ldots,v_{m-1};r_m) \end{array}\right]$$

to denote

$$\Pr_{\substack{r_1,\dots,r_m\\ r_p}}\left[\bigvee_{v_1,\dots,v_m} P(v_1,\dots,v_m;r_p)\wedge v_1 = f_1(\Pi;r_1)\wedge\cdots\wedge v_m = f_m(\Pi,v_1,\dots,v_m;r_m)\right].$$

A *Linear Feedback Shift Register* (LFSR) can be described by its *feedback polynomial* $P=\sum_{i=0}^{d_P} p_i X^i$. It is then denoted $\mathcal{L}_P$. When given an initial state $s=(s_0,s_1,\dots,s_{d_P-1})$, an LFSR $\mathcal{L}_P$ produces a bitstream denoted $\mathrm{S}_{\mathcal{L}_P}(s)$. Recall that an LFSR with feedback polynomial P and initial state $s=(s_0,s_1,\dots,s_{d_P-1})$ produces the bitstream s_i with $s_{i+d_P}=\sum_{k=0}^{d_P-1} p_k s_{i+k}$ in $\mathbb{F}_2$.

Finally, we define two operations used in TCHo. The *bitwise sum* (in $\mathbb{F}_2$) of two bitstrings x and y of same length is written $x+y$. The *product* of a polynomial $K\in\mathbb{F}_2[X]$ of degree d, $K=\sum_{j=0}^{d} k_j X^j$ and a bitstring $\mathrm{S}^{d+N}=(s_0,\dots,s_{N+d-1})$ is denote $K\otimes \mathrm{S}^{d+N}$ and is defined as

$$K\otimes \mathrm{S}^{d+N}=(s'_0,\dots,s'_{N-1}),$$

with $s'_i := s_i k_0 + s_{i+1}k_1+\cdots+s_{i+d}k_d$. We can also associate the polynomial K with an $N\times(d+N)$ matrix M_K^N defined as

$$M_K^N := \begin{bmatrix} k_0 & k_1 & \dots & k_d & 0 & 0 & \dots & 0\\ 0 & k_0 & k_1 & \dots & k_d & 0 & \dots & 0\\ & & \ddots & & & \ddots & & \\ 0 & 0 & \dots & 0 & k_0 & k_1 & \dots & k_d \end{bmatrix}.$$

Then, we have

$$\left[s'_0 \,\dots\, s'_{N-1}\right]^T = M_K \left[s_0 \,\dots\, s_{N+d-1}\right]^T.$$

Note that $P\otimes \mathrm{S}^{\ell}_{\mathcal{L}_P}=\mathbf{0}$, i.e., when the feedback polynomial is used for the multiplication, we obtain the zero bitstring. This multiplication operator verifies also $(PQ)\otimes\mathrm{S}=P\otimes(Q\otimes\mathrm{S})$. Thus, if P divides K, $K\otimes\mathrm{S}^{\ell}_{\mathcal{L}_P}=\mathbf{0}$.

A function $f(\lambda)$ is *negligible* if for all $d\in\mathbb{R}$ we have $f(\lambda)=O\left(\lambda^{-d}\right)$.

Definition 7.1 (Cryptosystem). A cryptosystem over a given message space $\mathcal{M}$ and random coin space $\mathcal{R}$ consists of three *polynomial-time* algorithms:

- *A probabilistic key-generation algorithm* $\mathsf{Gen}(1^\lambda)$ taking as input some security parameter 1^λ in unary representation, and producing a secret key K_s and a public key K_p;
- *A probabilistic encryption algorithm* $\mathsf{Enc}(K_p,m;r)$ taking as input a public key K_p and a message $m\in\mathcal{M}$ with some random coins $r\in\mathcal{R}$, and producing a ciphertext y in the ciphertext space $\mathcal{C}$;
- *A deterministic decryption algorithm* $\mathsf{Dec}(K_s,c)$ taking as input a secret key K_s and a ciphertext $c\in\mathcal{C}$, and producing a message or an error.

The cryptosystem must satisfy the following correctness property:

$$\max_{m\in\mathcal{M}} \Pr\left[\mathsf{Dec}(K_s,\mathsf{Enc}(K_p,m;\rho)) \neq m : \quad (K_s,K_p) \leftarrow \mathsf{Gen}(1^\lambda;\rho_g)\right]$$

is negligible as λ increases.

We will also use the following security notions and acronyms. Adaptive Chosen Ciphertext Attack is denoted CCA, Chosen Plaintext Attack CPA, Indistinguishability IND, and One-wayness OW.

Definition 7.2 (OW-CCA-security). A cryptosystem is (t,ε)-OW-CCA-secure if no adversary $\mathcal{A}$, with access to a decryption oracle $O_{K_s,c}$ and with running time bounded by t, can recover the plaintext from a given ciphertext with a probability higher than ε. More formally, for all $\mathcal{A}$ bounded by t,

$$\Pr\left[\mathcal{A}^{O_{K_s,c}}(c;\rho) = m \ : \ \begin{array}{r} m \xleftarrow{U} \mathcal{M};\ r \xleftarrow{U} \mathcal{R} \\ (K_s,K_p) \leftarrow \mathsf{Gen}(1^\lambda) \\ c \leftarrow \mathsf{Enc}(K_p,m;r) \end{array}\right] \leq \varepsilon,$$

where $O_{K_s,c}(y) = \mathsf{Dec}(K_s,y)$ for $y \neq c$ and $O_{K_s,c}(y) = \perp$ otherwise. Asymptotically, a cryptosystem is OW-CCA-secure if for any polynomial $t(\lambda)$ there exists a negligible function $\varepsilon(\lambda)$ such that it is $(t(\lambda),\varepsilon(\lambda))$-OW-CCA-secure.

Definition 7.3 (IND-CPA-security). A cryptosystem is said (t,ε)-IND-CPA-secure or (t,ε)-*semantically secure* against chosen plaintext attacks if no adversary $\mathcal{A} = (\mathcal{A}_1,\mathcal{A}_2)$ with running time bounded by t can distinguish the encryption of two different plaintexts m_0 and m_1 with a probability higher than ε. More formally, for all $\mathcal{A}$ bounded by t,

$$\Pr\left[\mathcal{A}_2(K_p,c;\rho) = b \ : \ \begin{array}{l} (K_s,K_p) \leftarrow \mathsf{Gen}(1^\lambda) \\ m_0,m_1 \leftarrow \mathcal{A}_1(K_p;\rho) \\ r \xleftarrow{U} \mathcal{R};\ b \xleftarrow{U} \{0,1\} \\ c \leftarrow \mathsf{Enc}(K_p,m_b;r) \end{array}\right] \leq \frac{1}{2} + \varepsilon.$$

Asymptotically, a cryptosystem is IND-CPA-secure if for any polynomial $t(\lambda)$ there exists a negligible function $\varepsilon(\lambda)$ such that it is $(t(\lambda),\varepsilon(\lambda))$-IND-CPA-secure.

IND-CPA-security can also be represented in the real-or-random game model [5,6].

Definition 7.4 (Real-or-random IND-CPA game security). A cryptosystem is (t,ε)-IND-CPA-secure in the real-or-random game model if no adversary $\mathcal{A} = (\mathcal{A}_1,\mathcal{A}_2)$ with running time bounded by t can distinguish the encryption of a chosen plaintexts m_0 to a random one with a probability higher than ε. More formally, for all $\mathcal{A}$ bounded by t,

$$\Pr\left[\mathcal{A}_2(K_p,c;\rho)=b \;:\; \begin{array}{c}(K_s,K_p)\leftarrow \mathsf{Gen}(1^\lambda)\\ m_0\leftarrow\mathcal{A}_1(K_p;\rho);\; m_1\overset{U}{\leftarrow}\mathcal{M}\\ r\overset{U}{\leftarrow}\mathcal{R};\; b\overset{U}{\leftarrow}\{0,1\}\\ c\leftarrow\mathsf{Enc}(K_p,m_b;r)\end{array}\right]\leq\frac{1}{2}+\varepsilon.$$

Asymptotically, a cryptosystem is IND-CPA-secure in the real-or-random game model if for any polynomial $t(\lambda)$ there exists a negligible function $\varepsilon(\lambda)$ such that it is $(t(\lambda),\varepsilon(\lambda))$-IND-CPA-secure in the real-or-random game model.

A (t,ε)-IND-CPA-secure system in the real-or-random game model is $(t,2\varepsilon)$-IND-CPA-secure in the standard model [5]. Conversely, a (t,ε)-IND-CPA-secure system in the standard model is (t,ε)-IND-CPA-secure in the real-or-random game model. Asymptotically, both models are equivalent.

Definition 7.5 (IND-CCA-security). A cryptosystem is said (t,ε)-IND-CCA-secure or (t,ε)-*secure* against adaptive chosen plaintext attacks if no adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$, with access to a decryption oracle $O_{K_s,c}$ and with running time bounded by t can distinguish the encryption of two different plaintexts m_0 and m_1 with a probability higher than ε. More formally, for all $\mathcal{A}$ bounded by t,

$$\Pr\left[\mathcal{A}_2^{O_{K_s,c}}(K_p,c;\rho)=b \;:\; \begin{array}{c}(K_s,K_p)\leftarrow \mathsf{Gen}(1^\lambda)\\ m_0,m_1\leftarrow\mathcal{A}_1^{O_{K_s}}(K_p;\rho)\\ r\overset{U}{\leftarrow}\mathcal{R};\; b\overset{U}{\leftarrow}\{0,1\}\\ c\leftarrow\mathsf{Enc}(K_p,m_b;r)\end{array}\right]\leq\frac{1}{2}+\varepsilon,$$

where $O_{K_s,c}(y)=\mathsf{Dec}(K_s,y)$, and $O_{K_s,c}(y)=\mathsf{Dec}(K_s,y)$ for $y\neq c$ and $O_{K_s,c}(c)=\perp$. Asymptotically, a cryptosystem is IND-CCA-secure if for any polynomial $t(\lambda)$ there exists a negligible function $\varepsilon(\lambda)$ such that it is $(t(\lambda),\varepsilon(\lambda))$-IND-CCA-secure.

Definition 7.6. Given two sources S_0 and S_1, a *distinguisher* between them is an algorithm $\mathcal{D}$ that takes as input one sample x from either S_0 or S_1 and has to decide which source was used. Its *advantage* is

$$\mathrm{Adv}_{\mathcal{D}}(\mathrm{S}_0,\mathrm{S}_1)=\Pr[\mathcal{D}(x)=1:x\leftarrow\mathrm{S}_1]-\Pr[\mathcal{D}(x)=1:x\leftarrow\mathrm{S}_0].$$

We say that the two sources are (t,ε)-*computationally indistinguishable* if for any distinguisher $\mathcal{D}$ with running time bounded by t,

$$|\mathrm{Adv}_{\mathcal{D}}(\mathrm{S}_0,\mathrm{S}_1)|\leq\varepsilon.$$

Asymptotically, two sources are *computationally indistinguishable* if for any polynomial $t(\lambda)$ there exists a negligible function $\varepsilon(\lambda)$ such that, they are $(t(\lambda),\varepsilon(\lambda))$-computationally indistinguishable.

We will be using the following Chernoff bound:

Theorem 7.1 (Chernoff [14]). *For $X_1,\ldots,X_n$ independent identically distributed Bernoulli random variables with $E(X_1) \le \frac{1}{2}$,*

$$\Pr\left[\frac{1}{n}\sum_{i=1}^{n} X_i \ge \frac{1}{2}\right] \le \exp\left(-2n\left(E(X_1)-\frac{1}{2}\right)^2\right).$$

7.3 Computational Problems

TCHo 's security is based on computational problems which are believed to be hard. In this section, we survey the existing algorithms used to solve these problems and we focus on their best complexity to find suitable parameters for TCHo.

7.3.1 Low Weight Polynomial Multiple Problem

Unlike the integer factorization, efficient algorithms exist to factor polynomials over a finite field [7, 13]. However, finding a multiple of a given polynomial that has a bounded degree and a bounded weight can be hard. TCHo 's security relies on the *Low Weight Polynomial Multiple problem* (LWPM).

Problem 7.1 (LWPM). Let $w,d,d_P \in \mathbb{N}$ be three parameters such that $0 < d_P < d$ and $w < d$. Given an instance $P \in \mathbb{F}_2[x]$ of degree d_P, find a multiple K of P of degree at most d and weight at most w.

In TCHo, P will be the public key and one will be able to recover some information about the plaintext from a ciphertext using such a low weight polynomial K. Note that in TCHo, the private key will be one of the solutions to this problem. Hence, we are ensured that a solution exists. In fact, the public key P is generated as an irreducible factor of a random polynomial K of weight w_K and degree at most d with a nonzero constant factor. In this case, one can heuristically estimate the number of solutions with nonzero constant term as

$$\mathcal{N}_{\text{sol}} \approx 1 + 2^{-d_P}\binom{d}{w-1}. \tag{7.1}$$

Note that the additional solution comes from the way we generate our polynomial P. We present now algorithms used to find low weight polynomial multiples. In what follows, we review some existing algorithms to solve the LWPM problem in order to derive heuristically some hard domain parameters.

To support the hardness of the LWPM problem, Giesbrecht et al. showed that finding sparse multiple polynomials *with unbounded degree* in a finite field is at least as hard as computing orders in an extension of this field [28], a problem which is believed to be hard. Unfortunately, this result is not directly applicable to TCHo because we consider polynomially bounded degrees and we know that a solution exists within this bound.

7.3.1.1 Exhaustive Search

When d is close to d_P, we can use exhaustive search to find *all* low weight multiples of P. The exhaustive search is performed by simply checking the weight of all multiples of P. The complexity for finding all multiples is $\Theta\left(\mathrm{Poly}(d)2^{d-d_P}\right)$. However, this method is inefficient in TCHo, since $d - d_P$ is very large.

7.3.1.2 Birthday Paradox

Following Meier and Staffelbach [47], we build two lists L_1 and L_2 in which we store respectively polynomials with weight $\lfloor (w-1)/2 \rfloor$, degree smaller than d, and zero constant term and polynomials with weight $\lceil (w-1)/2 \rceil$, degree smaller than d, and constant term equal to one. Once we have these lists, we look for pairs that sum to 0 modulo P. This collision search can be done efficiently using a hash table. Note that when w is odd (as in TCHo), one can use only the list L_1 and search for pairs (in L_1) summing to 1 instead. The list size is $\binom{d}{\lceil (w-1)/2 \rceil}$. Hence, the memory use is

$$2\binom{d}{\lceil (w-1)/2 \rceil}(w-1)\log(d) \approx \Theta\left(d^{\lceil (w-1)/2 \rceil}\log(d)\right)$$

for small weights w. The time complexity is $\Theta\left(\binom{d}{(w-1)/2}\right)$. This strategy is clearly faster than exhaustive search but uses a lot of memory. In the case of TCHo, d is typically greater than 2^{15} and w greater than 68. Hence, the lists contains more than $\Omega\left(2^{388}\right)$ elements, which is way too much. An improvement of this method is proposed by Chose, Joux and Mitton to solve this problem using $\Theta\left(d^{\lceil (w-1)/4 \rceil}\log(d)\right)$ space instead [15]. An alternative solution was also proposed by Didier and Laigle-Chapuy using discrete logarithms [20]. Assuming that in practice the discrete logarithm with base element X has a negligible complexity over $\mathbb{F}_2[X]/\langle P \rangle$, they achieve a time complexity of $\Theta\left(d^{(w-1)/2-1}\right)$ for a memory cost equal to the original birthday paradox method.

7.3.1.3 Generalized Birthday Paradox

When there is a large number of solutions, one can use Wagner's generalized birthday paradox [64] to find more efficiently one solution. The idea is to make use of 2^k lists of polynomial of weight $(w-1)/2^k$ instead of two lists as in the birthday paradox algorithm. Collisions are then found in pairs of lists until one single list remains containing a wanted solution. This algorithm will not return all possible solutions but can find one of them. However, the lists need to have a size greater than $2^{d_P/(k+1)}$. Hence, we need

$$\binom{d}{(w-1)/2^k} \geq 2^{d_P/(k+1)},$$

for a $k > 1$. In this case, a solution can be found with time and memory complexity

$$\Theta\left(2^k 2^{d_P/(k+1)}\right).$$

7.3.1.4 Finding a Low Weight Multiple Using Lattices

El Ailmani and von zur Gathen [24] presented a lattice-based algorithm to solve the LWPM problem. The set of multiples of P with degree lower than d form a lattice L_d. More formally

$$L_d := \{Q \in \mathbb{Z}[X] : Q \in P\mathbb{Z}[X] + 2\mathbb{Z}[X], \deg(Q) < d\}.$$

This lattice has dimension d. Now, note that a low weight polynomial multiple of P is a short vector in L_d. Such a vector can be found by first finding a basis of L_d, then by reducing this basis using for instance the LLL algorithm [38]. The algorithm uses $O\left(d^6\right)$ time and $O\left(d \times d_P\right)$ memory if we use LLL. However, this method is strongly limited by the lattice dimension. When d is too large, the size of the short vector we find using LLL becomes greater than w. This comes from the fact that this vector is only an approximation of the shortest vector in the lattice. Hence, this technique is inefficient to attack TCHo.

7.3.1.5 Syndrome Decoding

Syndrome decoding can also solve this problem. First, we compute the matrix H, whose column i is defined by $X^i \bmod P$, for $1 \leq i \leq d-1$. Once this matrix is computed, we search for a low weight polynomial in the preimages of 1 of this matrix. Following Canteaut and Chabaud [11], one solution can be found in time

$$\Theta\left(\frac{1}{\mathcal{N}_{\text{sol}}}\left(\frac{d-1}{d_P}\right)^{w-1}\right).$$

By using (7.1), this approximates to

$$\Theta\left(\frac{\left(\frac{d-1}{d_P}\right)^{w-1}}{2^{-d_P}\binom{d}{w-1}+1}\right).$$

7.3.1.6 Hardness of the LWPM Problem

Out of this survey, we deduce the following assumption

Assumption 7.2. *Let w, d, P be an instance of the LWPM problem. Let λ be the security parameter. The LWPM problem is believed to require a super-polynomial complexity if*

$$(w-1)\log\frac{d}{d_p} \geq \lambda, \tag{7.2}$$

and

$$\binom{d}{w-1} < 2^{d_P}. \tag{7.3}$$

Indeed, these inequalities give no complexity better than $\Theta\left(2^{\lambda}\right)$ with the previous algorithms. Note that (7.3) implies that we shall expect no more solution than the one which was hidden.

7.3.2 The Noisy LFSR Decoding Problem

TCHo 's security relies also on the *noisy LFSR decoding problem.*

Problem 7.2 (Noisy LFSR Decoding). Let $\ell > 0$ be a length, let P be a polynomial of degree d_P, and let $0 \leq \gamma \leq 1$ be a bias. Recover X, the random seed of an LFSR, given $Y := \mathrm{S}^{\ell}_{L_P}(X) + \mathrm{S}^{\ell}_{\gamma}$, i.e., the bitwise addition between the output of this LFSR with feedback polynomial P and seed X, and some random noise with bias γ.

In TCHo, the plaintext will be hidden by such a Y. Hence, since the noise is strongly biased, if one can easily recover the seed of the LFSR, one can recover the plaintext. We survey now the techniques used to solve the noisy LFSR decoding problem.

7.3.2.1 Information Set Decoding

Information set decoding is performed as follows. We pick d_P random bits out of the ℓ output bits of Y and we solve the linear system induced by the columns of the generator matrix of the LFSR corresponding to these bits (e.g. by performing

Gaussian elimination). X can be recovered if there are no errors among the d_P selected bits. This happens with probability

$$\left(\frac{1}{2}+\frac{\gamma}{2}\right)^{d_P}.$$

Hence, we can recover X with complexity[3]

$$\Theta\left(\frac{1}{2}+\frac{\gamma}{2}\right)^{-d_P}. \tag{7.4}$$

This is $\Omega(2^{\lambda})$ for

$$\gamma \leq 2^{1-\lambda/d_P}-1.$$

7.3.2.2 Maximum Likelihood Decoding

Maximum likelihood (ML) decoding is a bruteforce technique that consists in computing $S^{\ell}_{\mathcal{L}_P}(X)$ for all possible random seeds X and keep the one with smallest distance to Y. This costs $\Theta\left(2^{d_P}\ell\right)$ and can be improved to $\Theta\left(2^{d_P}d_P\right)$ by using a fast Walsh transform [42]. More subtle ML algorithms exist that decode only a subcode of the full code. In this case, even though we do not recover completely the seed X, we may recover some bits of information which may threaten TCHo. It can be shown [26] that if we take $d_P \geq 2\lambda$, these algorithms yield less than 1 bit of information.

7.3.2.3 Iterative Decoding

Iterative decoding [12] consists in finding low weight multiples of P forming parity check equations. The low-weight parity check code associated to these equations can then be solved. When $d_P \geq 2\lambda$, decoding is not possible using this technique [26].

7.3.2.4 Hardness of the Noisy LFSR Decoding Problem

Out of this survey, we deduce the following assumption.

Assumption 7.3. *Let ℓ, P, γ be an instance of the noisy LFSR decoding problem. Let λ be the security parameter. The problem is believed to require a super-polynomial complexity if*

$$d_P \geq 2\lambda, \tag{7.5}$$

[3]Note that we can neglect the cost of the Gaussian elimination by using improved algorithms [11].

and

$$\gamma \leq 2^{1-\lambda/d_P} - 1. \tag{7.6}$$

Indeed, these inequalities give no complexity better than $\Theta\left(2^{\lambda}\right)$ with the previous algorithms.

7.3.3 The Noisy LFSR Distinguishing Problem

The previous problem has a decisional counterpart.

Problem 7.3 (Noisy LFSR Distinguishing). Let $\ell > 0$ be a length, let P be a polynomial of degree d_P, and let $0 \leq \gamma \leq 1$ be a bias. Given an ℓ-bit string Y, decide whether Y was generated by $Y = \mathrm{S}^{\ell}_{\mathcal{L}_P}(X) + \mathrm{S}^{\ell}_{\gamma}$ or by a uniformly distributed source $Y = \mathrm{S}^{\ell}_{0}$.

7.3.3.1 Linear Noise Cancellation

We can think of two strategies to distinguish $\mathrm{S}^{\ell}_{*} = \mathrm{S}^{\ell}_{\mathcal{L}_P} + \mathrm{S}^{\ell}_{\gamma}$ from $\mathrm{S}^{\ell}_{*} = \mathrm{S}^{\ell}_{0}$. The first one is to apply the solutions we presented to solve the noisy LFSR decoding problem, i.e., to recover the random seed used by $\mathcal{L}_P$. We know that if (7.5) and (7.6) hold, the problem is supposed hard.

Another solution is to multiply S^{ℓ}_{*} by P or any of its multiple Q. If S^{ℓ}_{*} is not S^{ℓ}_{0}, we have

$$Q \otimes \mathrm{S}^{\ell}_{*} \approx \mathrm{S}^{\ell - d_Q}_{\gamma^{w_Q}}.$$

The best advantage [4] one can get to distinguish N bits of bias γ^w from random ones is $\mathrm{Adv} \approx \gamma^w \sqrt{N/(2\pi)}$. From Sect. 7.3.1, we know that the cost to find a multiple of P with degree bounded by d and weight bounded by w is $\mathrm{Comp} = (d/d_P)^{w-1}$ if we use syndrome decoding. It works with probability bounded by $\mathrm{Success} = 2^{-d_P}\binom{d}{w-1}$.

To identify the range of parameters for which this method does not work, we want

$$\frac{1}{\mathrm{Adv}} + \frac{\mathrm{Comp}}{\mathrm{Success}} \leq 2^{\lambda}.$$

So, N, w, d_P are polynomially bounded. We have

$$\frac{1}{\mathrm{Adv}} + \frac{\mathrm{Comp}}{\mathrm{Success}} = \frac{\gamma^{-w}}{\sqrt{N/2\pi}} + \frac{(d/d_P)^{w-1}}{2^{-d_P}\binom{d}{w-1}} \geq \mathrm{Poly}(\lambda)\left(\frac{\gamma^{-w}}{\sqrt{N/2\pi}} + \left(\frac{we^{-1}}{d_P}\right)^{w} 2^{d_P}\right).$$

Let $f(w)$ be the term under parentheses. The function $w \mapsto \left(we^{-1}/d_P\right)^{w}$ decreases until $w = d_P$ and then increases. Let τ be the root of $\gamma e^{-1} = \tau 2^{-\tau}$. Since we will take $\gamma = 1 - o(1)$, we have $\tau = \tau_0 + o(1)$ where $\tau_0 :\approx 3.053$. Note that $\tau \geq 1$ since $\gamma \leq 1$.

We assume that $d_P \log(1/\gamma) \geq \lambda\tau$. Let $w_0 := d_P/\tau$. If $w \leq w_0$, since $w < d_P$ (because $\tau \geq 1$), we have

$$f(w) \geq \left(\frac{w\mathrm{e}^{-1}}{d_P}\right)^{w} 2^{d_P} \geq \left(\frac{w_0\mathrm{e}^{-1}}{d_P}\right)^{w_0} 2^{d_P} = \left(2\left(\frac{1}{\tau}\mathrm{e}^{-1}\right)^{1/\tau}\right)^{d_P} = \gamma^{-d_P/\tau} \geq 2^{\lambda}.$$

If $w \geq w_0$, we have

$$f(w) \geq \gamma^{-w} \geq \gamma^{-w_0} = \gamma^{-d_P/\tau} \geq 2^{\lambda}.$$

This leads us to the following assumption:

Assumption 7.4. *Let ℓ, P, γ be an instance of the noisy LFSR distinguishing problem. Let λ be the security parameter. Let τ be the root of $\gamma\mathrm{e}^{-1} = \tau 2^{-\tau}$. The problem is believed to require a super-polynomial complexity if*

$$\gamma \leq 2^{1-\lambda/d_P} - 1,$$

and

$$d_P \log\frac{1}{\gamma} \geq \lambda\tau.$$

Indeed, these inequalities give no complexity better than $\Theta\left(2^{\lambda}\right)$ for advantage $\Omega(2^{-\lambda})$ with the previous algorithms.

Note that if $d_P < 2\lambda$, we have $\gamma > \sqrt{2} - 1$ so $d_P \log(1/\gamma) \leq 1.28 \times d_P$. Furthermore, we have in this case $\tau \geq 5$ which implies that $\lambda\tau \geq 5\lambda$. We deduce from it that $1.28 \times d_P \geq 5\lambda$ which contradicts $d_P < 2\lambda$. Hence, the hypotheses imply $d_P \geq 2\lambda$ which was used in Assumption 7.3.

7.4 Presentation of the TCHo Cryptosystem

In this section, we describe the TCHo cryptosystem and give algorithms for key generation, encryption and decryption. We also prove that TCHo is a cryptosystem.

7.4.1 Parameters

TCHo 's secret key consists in a low weight polynomial K over $\mathbb{F}_2[X]$ of degree d_K bounded by d and of weight w_K. The public key is a polynomial P such that P divides K and whose degree is in a given interval $[d_{\min}, d_{\max}]$. The security of the scheme relies on noise added by an LFSR with the public key as feedback polynomial and some strongly biased random noise. The bias of the noise γ along with the plaintext length k and the ciphertext length $\ell > d$ are the remaining parameters. Hence, for a fixed system with security parameter λ, we can define a parameter vector

$$(k, d_{\min}, d_{\max}, d, w_K, \gamma, \ell) .$$

We require that $k, d_{\min}, d_{\max}, d, w_K, \ell$ are positive integers, polynomially bounded, $d_{\max} > d_{\min}$, w_K odd, $3 \leq w_K \leq k$, $d \geq d_{\max}$, $k + d \leq \ell$, and that γ is subject to the following requirement which is needed for correctness.

$$\gamma^{2w_K} \frac{\ell - d}{k} = \Omega(\lambda^{\alpha}) \tag{7.7}$$

for some constant $\alpha > 0$. Later, we shall add the requirements of Assumptions 7.2 and 7.4 for security.

There are two approaches for selecting the parameters: in practice, we select some for which we have good implementation performances and a fair understanding of the security. This will be covered in Sect. 7.6. In theory, we select a family of parameters based on λ so that algorithms are polynomially bounded and whose security relies on complexity assumptions. This will be addressed in Theorems 7.5 and 7.6.

7.4.2 Key Generation

First, a random polynomial K of degree bounded by d and odd weight w_K with constant term 1 is generated. Nonzero coefficients in K shall be selected at positions which are pairwise different modulo k. If $K(X)$ is not coprime with $X^k - 1$ (which would be exceptional), we try again. Then, an irreducible factor P of degree $d_P \in [d_{\min}, d_{\max}]$ is searched. This procedure is repeated with another K until an appropriate P is found:

Generate:

```
 1: repeat
 2:     pick a random subset $I$ of $\{1, \ldots, k-1\}$ of cardinality $w_K - 1$
 3:     for each $i \in I$, pick a random $j_i$ such that $j_i \bmod k = i$ and $0 < j_i < d$
 4:     take $K(X) = 1 + \sum_{i \in I} X^{j_i}$
 5:     if $K(X)$ is coprime with $X^k - 1$ then
 6:         factor $K$ as a product of irreducible polynomials over $\mathbb{F}_2$
 7:         pick an irreducible factor $P$ of degree $d_P \in [d_{\min}, d_{\max}]$
 8:     end if
 9: until $P$ found
10: return $K$ and $P$
```

Note that since K is sparse, it can be stored efficiently using only $\lceil w_K \log(d) \rceil$ bits.

The number of irreducible polynomials of degree d is equivalent to $2^d/d$. So, a random polynomial has an irreducible factor of degree d with probability $1/d$. From that, we deduce that a random polynomial has an irreducible factor of degree in $[d_{\min}, d_{\max}]$ with probability $O\left((d_{\max} - d_{\min})/d_{\max}\right)$. Hence, $O\left(d_{\max}/(d_{\max} - d_{\min})\right)$ factorization attempts are needed in average. Using the Cantor-Zassenhaus [13] factoring algorithm, every attempt costs $O\left(d^2 \log d \log\log d\right)$.

The total complexity of the key generation algorithm is, thus,

$$O\left(\frac{d_{\max}}{d_{\max} - d_{\min}} d^2 \log d \log\log d\right).$$

We make the heuristic assumption that the complexity is the same when the polynomial K is sparse instead of being fully random. This assumption will be discussed in Sect. 7.6. However, we have no formal proof so far that this algorithm is polynomially bounded. This is left open for future work.

7.4.3 Encryption

Let $C(m) : \{0,1\}^k \to \{0,1\}^\ell$ be the repetition code that, given an $m \in \{0,1\}^k$ returns the ℓ bit word

$$m\|m\|\ldots\|\tilde{m},$$

where $\tilde{m}$ is the bitstring m truncated such that $C(m)$ has length ℓ. Given a plaintext m of length k, the ciphertext y of length ℓ is

$$y := \mathsf{Enc}_{\text{TCHo}}(P, m; r_1\|r_2) = C(m) + \mathrm{S}^{\ell}_{\mathcal{L}_P}(r_1) + \mathrm{S}^{\ell}_{\gamma}(r_2),$$

where r_1 and r_2 are random seeds. Care has to be taken about the size of these seeds. The first seed, r_1, consists in a random initial state for the LFSR. Hence, it has to be a random bitstring in $\{0,1\}^{d_P}$. The second seed, r_2, is a random seed for a biased pseudo random bit source. To ensure a proper security, this seed needs to be at least λ-bit long, where λ is the security parameter.

The encryption cost is $O(\ell \times d_P)$ if the random bit generator has not a higher complexity. In the case of a dedicated hardware implementation, the encryption can be done in $O(\ell)$ time with $O(d_P)$ gates.

```
Encrypt(P, m; r_1, r_2):
1: compute $y = C(m) + \mathrm{S}^{\ell}_{\mathcal{L}_P}(r_1) + \mathrm{S}^{\ell}_{\gamma}(r_2)$
2: return $y$
```

7.4.4 Decryption

Let $y \in \{0,1\}^\ell$ be the ciphertext, i.e., $y = C(m) + \mathrm{S}^{\ell}_{\mathcal{L}_P}(r_1) + \mathrm{S}^{\ell}_{\gamma}(r_2)$. We decrypt y as follows.

First, we remove the contribution of the noise induced by the LFSR $\mathcal{L}_P$. This is done by computing $t := \mathsf{trunc}_{\ell-d}(K \otimes y)$. The resulting t is truncated to $\ell - d$ bits.[4] Since the multiplication operator is distributive and since K is a multiple of P, which is the feedback polynomial of $\mathcal{L}_P$, this operation completely removes the noise generated by the LFSR. However, this operation has also an effect on $C(m)$ and $\mathrm{S}_\gamma(r_2)$. We have $K \otimes S_\gamma^\ell \approx S_{\gamma^{w_K}}^{\ell-d}$ in the sense that every bit of $K \otimes S_\gamma^\ell$ has a bias of γ^{w_K} but they are not perfectly independent. Hence, if the weight of K is low, the noise remains strongly biased. We have also $K \otimes C(m) = C(m')$, where $m' = \psi_K(m)$, for some linear map ψ_K. Thus, $t \approx C(\psi_K(m)) + \mathrm{S}_{\gamma^{w_K}}^{\ell-d}$. Under some conditions, ψ_K is invertible.

Since the noise $\mathrm{S}_{\gamma^{w_K}}^{\ell-d}$ is strongly biased, we can recover $\psi_K(m)$ by performing majority logic decoding (MJD), i.e., by taking for each bit its majority value in the repetition code. For a proper choice of w_K, γ and ℓ, the probability of error will be negligible.

MJD decodes the correct $\psi_K(m)$ if for every $i = 0, \ldots, k-1$, there is a majority of j's such that $(K \otimes \mathrm{S}_\gamma^\ell)_{i+kj} = 0$.

Finally, we invert ψ_K to recover m. Recall that the operation $K \otimes C(m)$ can also be written as $M_K^{\ell-d} C(m)$, with

$$M_K^{\ell-d} := \begin{bmatrix} k_0 & k_1 & \ldots & k_d & 0 & 0 & \ldots & 0 \\ 0 & k_0 & k_1 & \ldots & k_d & 0 & \ldots & 0 \\ & & \ddots & & & \ddots & & \\ 0 & 0 & \ldots & 0 & k_0 & k_1 & \ldots & k_d \end{bmatrix}.$$

Since $C(m)$ is a repetition code, $\psi_K(m)$ can be written as $C_K m$, with

$$C_K = \begin{bmatrix} c_0 & c_1 & \ldots & c_{k-1} \\ c_{k-1} & c_0 & \ldots & c_{k-2} \\ \vdots & & \ddots & \vdots \\ c_1 & c_2 & \ldots & c_0 \end{bmatrix},$$

where

$$c_j = \sum_{\{i \in [0,d] :\, i \equiv j \bmod k\}} k_i.$$

The matrix C_K is invertible if and only if $c_0 + c_1 X + \cdots + c_{k-1} X^{k-1}$ is coprime with $X^k - 1$, which is equivalent to $K(X)$ being coprime with $X^k - 1$, which is a condition in our key generation algorithm. Hence, $m = C_K^{-1} m'$.

[4]Since K may have a degree less than d, $K \otimes y$ may have more than $\ell - d$ bits. To avoid side channels, we only use the first $\ell - d$ bits, as if K had degree d.

The decryption complexity is $O\left(w_K \times \ell + k^3\right)$ since the first operation takes $O\left(w_K \times \ell\right)$, the second $O\left(\ell - d\right)$ and the third $O\left(k^3\right)$ time.

```
Decrypt(K, y):
1: compute $t = \mathsf{trunc}_{\ell-d}(K \otimes y)$
2: for $i = 0$ to $k-1$ do
3:     set $\psi(m)_i = \text{majority}(t_{i+kj}\,;\, 0 \le i+kj < \ell - d)$
4: end for
5: compute $m = C_K^{-1}\psi(m)$
6: return $m$
```

7.4.5 TCHo Is a Cryptosystem

In this section, we show that TCHo is a cryptosystem as defined in Definition 7.1.

Lemma 7.1. *The probability that a correctly generated ciphertext is badly decrypted satisfies*

$$\Pr[\text{bad decoding}] \le k \times \exp\left(-\frac{1}{2}\left\lfloor\frac{\ell-d}{k}\right\rfloor \gamma^{2w_K}\right).$$

Proof. We note from the requirement on K that nonzero coefficients have indices which are pairwise different modulo k. Hence, for a fixed i, all bits $(K \otimes \mathrm{S}_\gamma^\ell)_{i+kj}$ are independent. So,

$$\Pr[\text{bad decoding}] \le \rho,$$

where

$$\rho := \sum_{i=0}^{k-1} \sum_{w=0}^{\frac{1}{2}\lfloor\frac{\ell-d-i}{k}\rfloor} \binom{\lfloor\frac{\ell-d-i}{k}\rfloor}{w} \left(\frac{1+\gamma^{w_K}}{2}\right)^{w} \left(\frac{1-\gamma^{w_K}}{2}\right)^{\lfloor\frac{\ell-d-i}{k}\rfloor - w}. \tag{7.8}$$

We conclude thanks to the Chernoff bound (Theorem 7.1). □

Theorem 7.5. *There are some parameter selections making TCHo a cryptosystem with heuristic key generation that verifies the inequalities in Assumptions 7.2–7.4.*

Proof. Let λ be the security parameter. We select parameters satisfying

$$w_K = a\lambda \qquad k = w_K + \Theta(\lambda) \qquad d = \Theta\left(\lambda^2 \times k\right)$$
$$d_{\min} = \Theta\left(\lambda^2\right) \qquad d_{\max} = d_{\min} + \Theta\left(\lambda^2\right) \qquad \ell = d + \Theta\left(\lambda^2 \times k\right) \qquad \gamma = \lambda^{-c/\lambda},$$

such that these are positive integers, w_K is odd, $a, c > 0$ and with the following condition:

$$0 < ac < 1.$$

With these parameters, key generation takes $O\left(\lambda^4 \times k^2 \times \log\lambda \times \log\log\lambda\right)$ (heuristically), encryption takes $O\left(\lambda^4 \times k\right)$, decryption $O\left(\lambda^3 \times k\right)$ and (7.7) is verified. Furthermore, these parameters satisfy the inequalities in Assumptions 7.2 and 7.4, which will be needed to show the security of our scheme.

Since the parameters satisfy (7.7), there exists a constant $f \geq 0$ such that

$$\Pr[\mathsf{bad\ decoding}] \leq k \times \exp(-f\lambda^{\alpha})$$

when λ is large enough. So, this probability is negligible. Hence, the cryptosystem satisfies also the correctness property. □

7.5 Security of TCHo

In this section, we show results on the security of TCHo. In particular, we show that TCHo is IND-CPA-secure and not OW-CCA-secure. We show also how to achieve IND-CCA security.

7.5.1 TCHo Is IND-CPA-Secure

Theorem 7.6 (Aumasson et al. [3]). *Let S_0 be an unbiased source of random bits. There exists a constant μ such that, for every TCHo parameters, if $\mathsf{S}^{\ell}_{\mathcal{L}P} + \mathsf{S}^{\ell}_{\gamma}$ cannot be distinguished from S^{ℓ}_0 in time t with an advantage larger than ε, then TCHo is $(t - \mu\ell, 2\varepsilon)$-IND-CPA-secure.*

Proof. Instead of proving the semantic security directly, we prove it in the real-or-random game model. Recall that in this model, the adversary submits first a chosen plaintext x following an algorithm $\mathcal{A}^{\mathrm{ror}}_1(K_p;\rho)$. Then, given a ciphertext z, the adversary has to decide using an algorithm $\mathcal{A}^{\mathrm{ror}}_2(K_p, z;\rho)$ whether z is a ciphertext corresponding to x or to another random plaintext.

We show that using this adversary $\mathcal{A}^{\mathrm{ror}} = (\mathcal{A}^{\mathrm{ror}}_1, \mathcal{A}^{\mathrm{ror}}_2)$, we can build a distinguisher between $\mathsf{S}^{\ell}_{\mathcal{L}P} + \mathsf{S}^{\ell}_{\gamma}$ and S^{ℓ}_0. Let S^{ℓ}_* be the unknown instance we have to distinguish. First we recover a plaintext $x = \mathcal{A}^{\mathrm{ror}}_1(P)$. Let $z = C(x) + \mathsf{S}^{\ell}_*$. If S^{ℓ}_* is random, then z is also a totally random bitstring. Note that this z corresponds also to a valid ciphertext, since every bitstring in $\{0,1\}^{\ell}$ is valid. On the other hand, if S^{ℓ}_* is $\mathsf{S}^{\ell}_{\mathcal{L}P} + \mathsf{S}^{\ell}_{\gamma}$, then z is a valid ciphertext of x. Hence, using $\mathcal{A}^{\mathrm{ror}}_2(P, z)$, we can decide, whether z is a ciphertext corresponding to x or not. The cost of this simulation is $\mu\ell$, for $\mu > 0$ large enough. Thus, since we know by assumption that we cannot distinguish $\mathsf{S}^{\ell}_{\mathcal{L}P} + \mathsf{S}^{\ell}_{\gamma}$ from S^{ℓ}_0 in time t with an advantage larger than ε, $\mathcal{A}^{\mathrm{ror}}$ has complexity at least $t - \mu\ell$. Hence, TCHo is $(t - \mu\ell, \varepsilon)-$IND-CPA secure in the real-or-random game model. This implies that TCHo is $(t - \mu\ell, 2\varepsilon)-$IND-CPA secure. □

Now, we just have to find suitable parameters such that the Noisy LFSR Distinguishing Problem is hard to obtain an IND-CPA-secure scheme.

7.5.2 TCHo Is Not OW-CCA Secure

We recall two negative results from [3] regarding the security of TCHo.

Lemma 7.2. *TCHo is malleable.*

Proof. Given a ciphertext y corresponding to a plaintext x, $y+C(x')$ is a valid ciphertext of $x+x'$ with correct distribution. $\square$

This result implies also that TCHo is not IND-CCA secure as we can just use the malleability of ciphertexts and call the decryption oracle on the modified ciphertext to play the IND-CCA game.

Lemma 7.3. *TCHo is* not OW-CCA *secure.*

Proof. Given a ciphertext y to invert, modify the first bit an submit it to the decryption oracle. With high probability, the obtained plaintext will correspond to the original one. $\square$

7.5.3 The Herrmann-Leander Attack

In PKC 2009, Herrmann and Leander presented a chosen ciphertext key recovery attack on TCHo [33]. They were able to recover the secret key in about $d^{3/2}$ queries to a decryption oracle. As shown in Lemma 7.3, TCHo is not OW-CCA secure. However, their attack is by far worse than the traditional OW-CCA message recovery attack since it reveals the private key. It is important to notice that their attack does not solve the LWPM problem but extracts this low weight polynomial by querying the decryption oracle in a clever way.

This proves that the key recovery problem is easy with a Dec_K oracle. This does not prove that decryption and key recovery are equivalent because we are using the Dec_K oracle with some inputs which do not follow the distribution of genuine ciphertexts. So, a decryption oracle able to decrypt ciphertexts may not fully emulate the Dec_K oracle for our purpose. We briefly present Herrmann and Leander's attack here.

The attack is an adaptive differential attack, i.e., pairs of ciphertexts with a well-chosen difference are submitted to the decryption oracle. Let the private key be $K=\sum_{j=0}^{d}k_jX^j$. Let also $N:=\ell-d$. For simplicity, we assume that $\lfloor\frac{N}{k}\rfloor$ is odd so that there is always a non-ambiguous majority among $\lfloor\frac{N}{k}\rfloor$ bits. The idea is to recover every bit one after the other starting from k_1 to k_{d_K}. (Note that k_0 is fixed to 1.) To recover the key bit k_i knowing $k_0,\ldots,k_{i-1}$, two ciphertexts y and y' are submitted to the decryption oracle such that the difference between them before the MJD step during the decryption process is $t \oplus t' = \Delta := (1 \oplus k_i, 0, \ldots, 0, 1, 0, \ldots, 0)$, where the 1 is at index i. Let the two obtained plaintexts be respectively m and m'. If m and m' differ, this means that the MJD algorithm made different decisions for the two ciphertexts. With a clever choice of the ciphertexts y and y' and using our knowledge of the previous bits of K, we can ensure that $t = (b, 0, r)$, with $b = \mathsf{trunc}_i(1, 0^{(2k-1)}, 1, 0^{(2k-1)}, \ldots)$ and r an uniformly distributed random bitstring of length $\ell - i - d - 1$. The b part ensures that the first sum in the majority decoding algorithm is as much balanced as possible.

Let

$$M' := \begin{bmatrix} k_0 & k_1 & \ldots & k_{i-1} \\ 0 & k_0 & \ldots & k_{i-2} \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & 0 & k_0 \end{bmatrix}.$$

To construct y, we take $y := (\hat{y}, 0^{(d+1)}, r)$, where $\hat{y}$ is the solution of $M'\hat{y} = b$ and r is a random bit string of size $N - i - 1$. For y', let δ be defined as

$$\delta_0 = \sum_{j=0}^{i-2} \delta'_j k_{j+1} \oplus 1,$$

$$\delta_j = \delta'_{j-1} \qquad \text{for } 1 \leq j \leq i,$$

$$\delta_j = 0 \qquad \text{for } i+1 \leq j < \ell,$$

where δ' is the solution of $M'\delta' = (0, \ldots, 0, 1)^t$. Then, $y' := y + \delta$. We refer the reader to [33] for a proof of correctness of this construction. These two ciphertexts produce the required t and t'.

To recover k_i, we distinguish two cases:

- When $i \equiv 0 \pmod{k}$, both the difference $1 \oplus k_i$ and 1 in Δ contribute to the same bit during MJD. Since $t_0 = 1$ and $t_i = 0$, we have $t'_0 = k_i$ and $t'_i = 1$. Hence, $m \neq m'$ implies that $k_i = 1$. Thus, the resulting plaintext will be different only when $\sum_{j=0}^{\lfloor N/k \rfloor} t_{kj} = \left\lfloor \frac{1}{2} \left\lfloor \frac{N}{k} \right\rfloor \right\rfloor$ and $k_i = 1$. However, in the case $k_i = 0$, the majority cannot change. By repeating this attack with a sufficient number of ciphertext pairs we recover k_i with negligible probability of error by making statistics.
- When $i \not\equiv 0 \pmod{k}$, the difference $1 \oplus k_i$ and the difference 1 does not contribute to the same bit during MJD. If $k_i = 0$, t and t' differ in their coordinates at index 0 and i and the majority at index 0 changes if $\sum_{j=0}^{\lfloor N/k \rfloor} t_{kj} = \left\lfloor \frac{1}{2} \left\lfloor \frac{N}{k} \right\rfloor \right\rfloor$ and $k_i = 1$. When $k_i = 1$, this majority cannot change. The one at index i may change depending on $\sum_{j=0}^{\lfloor (N-i)/k \rfloor} t_{kj+i}$. However, the difference $m \oplus m'$ will not be the same as when the majority changes at index 0, so it can be filtered out. Like in the previous case, we recover k_i by submitting sufficiently many ciphertext pairs.

We refer the reader to [33] for further details.

This attack implies that TCHo cannot be used in its original form. We show in the next section how TCHo can be transformed into an IND-CCA secure scheme.

7.5.4 Achieving IND-CCA Security

Following the Fujisaki-Okamoto construction [27], we can obtain an IND-CCA secure scheme using TCHo. For this, we need first to define Γ-uniformity.

Definition 7.7 (Γ-uniformity). Let Asym be an asymmetric encryption scheme, with key generation algorithm $\mathsf{Gen}(1^\lambda)$ and encryption algorithm $\mathsf{Enc}_{\mathsf{Asym}}(K_p, m; r)$ over the message space $\mathcal{M}$ and the random coins space $\mathcal{R}$. Asym is Γ-*uniform* if for any plaintext $m \in \mathcal{M}$, for any keys drawn by Gen, and for any $y \in \{0,1\}^*$, we have

$$\Pr\left[h \overset{U}{\leftarrow} \mathcal{R} : y = \mathsf{Enc}_{\mathsf{Asym}}(K_p, m; h)\right] \leq \Gamma,$$

i.e., the probability that a plaintext and a ciphertext match is bounded.

Now, we recall the Fujisaki-Okamoto paradigm: Given an OW-CPA and Γ-uniform asymmetric cipher Asym with public (resp. private) key K_p (resp. K_s), a one-time secure symmetric cipher Sym, and two random oracles G and H, the following construction is IND-CCA secure in the random oracle model:

Encryption: Given a message m:

```
1: Let $\sigma \overset{U}{\leftarrow} \{0,1\}^k$.
2: Let $\psi \leftarrow G(\sigma)$ be the symmetric key.
3: Encrypt the message using the symmetric cipher: $y = \mathsf{Enc}_{\mathsf{Sym}}(\psi, m)$.
4: Encapsulate the key with the asymmetric cipher: $\chi \leftarrow \mathsf{Enc}_{\mathsf{Asym}}(K_p, \sigma; H(\sigma \| m))$.
5: return $(\chi, y)$.
```

Decryption: Given a ciphertext (χ, y):

```
1: Decrypt the asymmetric part: $\hat{\sigma} = \mathsf{Dec}_{\mathsf{Asym}}(K_s, \chi)$.
2: Recover the symmetric key: $\hat{\psi} = G(\hat{\sigma})$.
3: Recover the message: $\hat{m} = \mathsf{Dec}_{\mathsf{Sym}}(\hat{\psi}, y)$.
4: if $\chi = \mathsf{Enc}_{\mathsf{Asym}}(K_p, \hat{\sigma}; H(\hat{\sigma} \| \hat{m}))$ then
5:     return $\hat{m}$.
6: else
7:     return $\perp$.
8: end if
```

Note that the check done at Step 4 during the decryption is necessary to obtain an IND-CCA secure scheme.

To use this construction with TCHo, we need to show that TCHo is Γ-uniform.

Lemma 7.4. *TCHo is* $((1+\gamma)/2)^\ell$*-uniform.*

Proof. Recall that the TCHo encryption of x is $y = C(x) + \mathsf{S}^{\ell}_{\mathcal{L}_P}(r_1) + \mathsf{S}^{\ell}_{\gamma}(r_2)$, for random coins r_1 and r_2. We need to bound the probability (taken over r_1 and r_2) that a given plaintext x and ciphertext y match. Since in TCHo we consider only positive bias, the most probable ciphertext corresponds to $y = C(x) + \mathsf{S}^{\ell}_{\mathcal{L}_P}$, i.e., when $\mathsf{S}^{\ell}_{\gamma}$ is the zero bitstring. This happens with probability $((1+\gamma)/2)^{\ell}$. When we take the average on the possible r_1, this probability can only decrease. Hence, TCHo is $((1+\gamma)/2)^{\ell}$-uniform. □

Since TCHo is $((1+\gamma)/2)^{\ell}$-uniform and since IND-CPA security implies OW-CPA security, we can use the Fujisaki-Okamoto paradigm to obtain a IND-CCA secure scheme. An example of a simple one-time secure symmetric cipher one could use is $\mathsf{Enc}_{\mathsf{Sym}}(\psi, m) = m + F(\psi)$ for a random oracle F. We obtain the following cryptosystem:

Encryption: Given a message m:

```
1: Let $\sigma \overset{U}{\leftarrow} \{0,1\}^k$.
2: Let $\psi \leftarrow G(\sigma)$ be the symmetric key.
3: Encrypt the message using the symmetric cipher: $y = m + F(\psi)$.
4: Encapsulate the key with the asymmetric cipher: $\chi \leftarrow \mathsf{Enc}_{\mathrm{TCHo}}(P, \sigma; H(\sigma \| m))$.
5: return $(\chi, y)$.
```

Decryption: Given a ciphertext (χ, y):

```
1: Decrypt the asymmetric part: $\hat{\sigma} = \mathsf{Dec}_{\mathrm{TCHo}}(K, \chi)$.
2: Recover the symmetric key: $\hat{\psi} = G(\hat{\sigma})$.
3: Recover the message: $\hat{m} = y + F(\hat{\psi})$.
4: if $\chi = \mathsf{Enc}_{\mathrm{TCHo}}(P, \hat{\sigma}; H(\hat{\sigma} \| \hat{m}))$ then
5:     return $\hat{m}$.
6: else
7:     return $\perp$.
8: end if
```

7.6 Implementation Results

In this section, we discuss various topics regarding the implementation of TCHo. First we describe a way to find good parameters and give some examples. Next we discuss our heuristic assumption used in the key generation algorithm. Finally, we comment on TCHo software and hardware implementation.

7.6.1 Parameter Selection

We show now how to find good parameters vectors that can be used for TCHo in practice. Recall from Sect. 7.4.4 that the probability that a message is incorrectly

Table 7.1 Example of parameter vectors for TCHo

ID	k	λ	$d_{\min}$	$d_{\max}$	d	w_K	γ	ℓ	$\rho_{\max}$
I	128	80	15,000	16,000	37,069	69	0.98862	55,000	2^{-20}
II	128	128	23,740	24,740	67,805	91	0.98853	100,233	2^{-20}
III	256	256	63,500	64,500	221,169	147	0.99141	326,100	2^{-20}
IV	384	384	145,000	146,000	455,356	237	0.99433	644,900	2^{-20}
V	512	512	155,000	156,000	845,405	213	0.99243	1,291,800	2^{-20}

decoded is bounded by ρ, defined by (7.8). We call this value the *unreliability* of the system. This value has an huge impact on the ciphertext length and the maximum unreliability accepted $\rho_{\max}$ has to be selected carefully depending on the application we consider. Recall that the parameters have to verify

$$(w_K - 1)\log\frac{d}{d_{\max}} \geq \lambda \qquad \text{and} \qquad \binom{d}{w_K - 1} < 2^{d_{\min}}$$

from Assumption 7.2 and

$$d_{\min}\log\frac{1}{\gamma} \geq \lambda\tau \qquad \text{and} \qquad \gamma \leq 2^{1-\lambda/d_{\min}} - 1,$$

where τ is the root of $\gamma e^{-1} = \tau 2^{-\tau}$, from Assumption 7.4. We need also to verify

$$\rho \leq \rho_{\max}.$$

Inequalities in Assumption 7.3 are consequences of the ones in Assumption 7.4 as already observed.

To find the best possible parameters, we used the following approach. We fix first the plaintext length k, the security parameter λ, and ρ_{max}, the maximum unreliability accepted, since all these three values will clearly depend on the application we consider and will drastically modify the ciphertext length. Then, we search for the best $d_{\min}$ that minimizes the ciphertext length. Indeed, d_{min} has a huge impact on the ciphertext length: a too small d_{min} implies that semantic security is harder to achieve, which leads to a smaller γ and finally to a larger ℓ too keep a tolerable unreliability. Similarly, a too large d_{min} implies a larger d or w_K, which leads also to a larger ciphertext length. Table 7.1 shows possible parameter vectors for $k = 128, 256, 384$ and 512 bits. We also set $\lambda = k$ for 4 of the 5 vectors, since TCHo will mostly be used to encrypt symmetric keys and, hence, should provide at least as much security as the key length.

7.6.2 *Heuristic Assumption for the Key Generation Algorithm*

In Sect. 7.4.2, we made the heuristic assumption that the probability for a random sparse polynomial to have an irreducible polynomial factor of degree in $[d_{\min}, d_{\max}]$

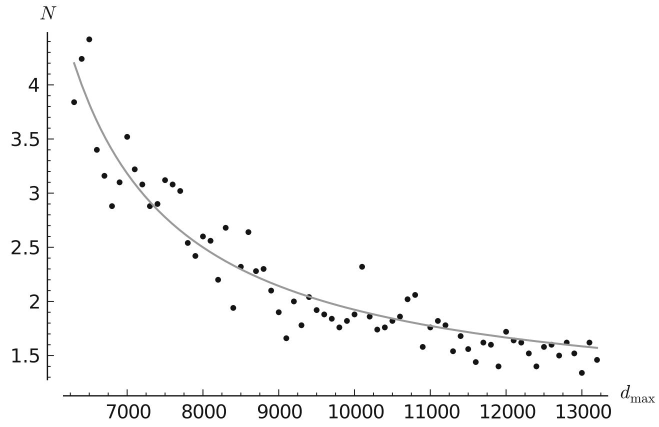

Fig. 7.1 Comparison between $d_{\max}/(d_{\max}-d_{\min})$ (*continuous line*) and the average number of iterations N for the key generation algorithm with parameters $k=128$, $w_K=69$, $d=32{,}069$ and $d_{\min}=4{,}800$ and using 40 samples (*points*)

is the same than when the polynomial is fully random, i.e., this probability is $O((d_{\max}-d_{\min})/d_{\max})$. To verify this heuristic assumption, we made some simulations in which we compared the average number of iterations N needed to generate the public key P with $d_{\max}/(d_{\max}-d_{\min})$. The result is depicted in Fig. 7.1. We can see that both distributions are quite similar and, hence, that our heuristic assumption seems reasonable.

7.6.3 Software Implementation

TCHo was implemented in software [10] as an extension of Network Security Services (NSS) [48] so that it can be used as a TLS/SSL cipher suite in browsers making use of NSS. One implementation issue concerning the LFSR was raised there. LFSRs are a very efficient component in hardware since a bit can be produced every clock cycle. However, in software, the complexity necessary to compute a bit is proportional to the weight of the feedback polynomial used to describe the LFSR. Hence, for TCHo, this complexity is proportional to w_P, the weight of the public key. Note that $w_P = d_P/2$ in average.

The trivial software implementation can be improved by considering blocks of outputs instead of a single bit. Typically, the size of the block b will be a small multiple of the machine word size. Then, using methods described in [16, 17], it is possible to obtain a cost of $O(w_P/b)$ operations per bit. However, the cost needed to compute the output of the LFSR is still large and dominates the cost of encryption.

This issue shows that TCHo is meant to be a hardware-oriented cipher and is, hence, less efficient in software.

Table 7.2 Performances of TCHo

ID	Key generation (s)	Encryption (ms)	Decryption (ms)	Secret key (bit)	Public key (bit)
I	882	54	11	1,048	16,000
II	7,033	158	42	1,461	24,740
III	46,994	253	120	2,610	64,500

Table 7.2 shows performances for the first three parameter vectors of TCHo presented in Table 7.1. These results were performed on a 2.66 GHz Core 2 Duo with a C++ implementation using the NTL library [63].

7.6.4 Hardware Implementation

We discuss in this section how to implement the encryption and the decryption in hardware.

For encryption, the implementation of the repetition code is straightforward. The LFSR can be efficiently implemented using integrated circuits. However, the length of the LFSRs we are dealing with is unusually big and we assume that this length does not alter the performances too much. Encryption requires also a biased source of noise. As mentioned in [3], this can be implemented using a precomputed binary tree where each leaf corresponds to a biased sequence of bits. Then, using a uniform random bitstream fed with physical entropy, one can decide which branch to take in the binary tree and output the biased bits.

Decryption is harder to implement. However, the product $K \otimes y$ consists of a sequence of dot products and can, hence, be implemented using some library able to do linear algebra computations for FPGA devices—for instance [65]. Similarly, such a library can be used to compute $C_K^{-1}\psi(m)$. Majority logic decoding is easy to implement but requires some additional memory—$k \times \log((\ell - d)/k))$—to store the number of occurrences of each bit.

Reference [3] estimates that a 128 bit key with $\lambda = 80$ can be encrypted with a circuit of about 10,000 gates. Hence, TCHo is well suited for RFID.

7.7 Conclusion

TCHo is still a young cryptosystem and has a large margin of progression. We indicate directions for further work.

Complexity. We are still missing a rigorous complexity analysis about the key generation algorithm, although we have a heuristic complexity matching well experimental results.

A Shorter Ciphertext Length. The use of a repetition code during the encryption process leads to a very simple decryption algorithm. However, the length of the ciphertexts is quite long. A possible solution would be to replace the repetition code with a code with a smaller overhead size.

A More Efficient IND-CCA Scheme. One drawback of the Fujisaki-Okamoto construction is that a TCHo encryption has to be performed during the decryption process. If we neglect the cost of the hash functions, this increases the cost of decryption to $O\left((w_K + d_P) \times \ell + k^3 + \rho\right)$, where ρ is the complexity of decryption of the symmetric scheme. This complexity can be reduced by using other hybrid constructions like, for instance, REACT [51] or GEM [18]. For this, we need the asymmetric scheme to be OW-PCA-secure, i.e., one-way against an adversary with an plaintext-checking oracle. Hence, we wonder whether TCHo is OW-PCA-secure.

Generalization. Another further work is to generalize the TCHo construction by replacing the LFSR with a random linear code. We wonder also if we can link TCHo to lattice-based cryptography.

In conclusion, TCHo is asymptotically an efficient encryption scheme based on a new hard problem, the low weight polynomial multiple problem. It is IND-CPA secure and can be used to obtain an IND-CCA scheme. However, it still suffers from two drawbacks: the key generation algorithm is expensive and the expansion factor is huge. In this paper, we reviewed the existing previous work on TCHo. We also provided new non-heuristic proofs of correctness and new parameters for different plaintext sizes.

References

1. Ajtai, M.: Generating Hard Instances of Lattice Problems (Extended Abstract). In: STOC, pp. 99–108 (1996)
2. Ajtai, M., Dwork, C.: A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence. In: STOC, pp. 284–293 (1997)
3. Aumasson, J.P., Finiasz, M., Meier, W., Vaudenay, S.: TCHo: A Hardware-Oriented Trapdoor Cipher. In: J. Pieprzyk, H. Ghodosi, E. Dawson (eds.) ACISP, *Lecture Notes in Computer Science*, vol. 4586, pp. 184–199. Springer (2007)
4. Baignères, T., Junod, P., Vaudenay, S.: How Far Can We Go Beyond Linear Cryptanalysis? In: Lee [37], pp. 432–450
5. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation (Full Version) (1997). Available at http://cseweb.ucsd.edu/users/mihir
6. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A Concrete Security Treatment of Symmetric Encryption (Extended Abstract). In: FOCS, pp. 394–403 (1997)
7. Berlekamp, E.: Factoring polynomials over large finite fields. Mathematics of Computation **24**(111), 713–735 (1970)
8. Bernstein, D.J.: Introduction to post-quantum cryptography. In: D.J. Bernstein, J. Buchmann, E. Dahmen (eds.) Post-Quantum Cryptography, pp. 1–14. Springer (2009)
9. Bernstein, D.J., Lange, T., Peters, C.: Attacking and Defending the McEliece Cryptosystem. In: J. Buchmann, J. Ding (eds.) PQCrypto, *Lecture Notes in Computer Science*, vol. 5299, pp. 31–46. Springer (2008)

10. Bindschaedler, V.: TCHo Software Implementation: Extending Firefox's Security Services Library. EPFL Bachelor Thesis (unpublished) (2010)
11. Canteaut, A., Chabaud, F.: A New Algorithm for Finding Minimum-Weight Words in a Linear Code: Application to McEliece's Cryptosystem and to Narrow-Sense BCH Codes of Length 511. IEEE Transactions on Information Theory **44**(1), 367–378 (1998)
12. Canteaut, A., Trabbia, M.: Improved Fast Correlation Attacks Using Parity-Check Equations of Weight 4 and 5. In: B. Preneel (ed.) EUROCRYPT, *Lecture Notes in Computer Science*, vol. 1807, pp. 573–588. Springer (2000)
13. Cantor, D., Zassenhaus, H.: A new algorithm for factoring polynomials over finite fields. Mathematics of Computation **36**(154), 587–592 (1981)
14. Chernoff, H.: A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. The Annals of Mathematical Statistics **23**(4), 493–507 (1952)
15. Chose, P., Joux, A., Mitton, M.: Fast Correlation Attacks: An Algorithmic Point of View. In: L.R. Knudsen (ed.) EUROCRYPT, *Lecture Notes in Computer Science*, vol. 2332, pp. 209–221. Springer (2002)
16. Chowdhury, S., Maitra, S.: Efficient Software Implementation of Linear Feedback Shift Registers. In: C.P. Rangan, C. Ding (eds.) INDOCRYPT, *Lecture Notes in Computer Science*, vol. 2247, pp. 297–307. Springer (2001)
17. Chowdhury, S., Maitra, S.: Efficient Software Implementation of LFSR and Boolean Function and Its Application in Nonlinear Combiner Model. In: J. Zhou, M. Yung, Y. Han (eds.) ACNS, *Lecture Notes in Computer Science*, vol. 2846, pp. 387–402. Springer (2003)
18. Coron, J.S., Handschuh, H., Joye, M., Paillier, P., Pointcheval, D., Tymen, C.: GEM: A Generic Chosen-Ciphertext Secure Encryption Method. In: B. Preneel (ed.) CT-RSA, *Lecture Notes in Computer Science*, vol. 2271, pp. 263–276. Springer (2002)
19. Courtois, N., Finiasz, M., Sendrier, N.: How to Achieve a McEliece-Based Digital Signature Scheme. In: C. Boyd (ed.) ASIACRYPT, *Lecture Notes in Computer Science*, vol. 2248, pp. 157–174. Springer (2001)
20. Didier, F., Laigle-Chapuy, Y.: Finding low-weight polynomial multiples using discrete logarithm. In: IEEE International Symposium on Information Theory, 2007 (ISIT 2007), pp. 1036–1040 (2007)
21. Diffie, W., Hellman, M.: New directions in cryptography. Information Theory, IEEE Transactions on **22**(6), 644–654 (1976)
22. Ding, J., Schmidt, D.: Rainbow, a New Multivariable Polynomial Signature Scheme. In: J. Ioannidis, A.D. Keromytis, M. Yung (eds.) ACNS, *Lecture Notes in Computer Science*, vol. 3531, pp. 164–175 (2005)
23. Duc, A.: TCHo: a Postquantum Public-Key Cryptography Toolkit. Unpublished Report (2010)
24. El Aimani, L., von zur Gathen, J.: Finding low weight polynomial multiples using lattices. Cryptology ePrint Archive, Report 2007/423 (2007). http://eprint.iacr.org
25. El Gamal, T.: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In: CRYPTO, pp. 10–18 (1984)
26. Finiasz, M., Vaudenay, S.: When Stream Cipher Analysis Meets Public-Key Cryptography. In: E. Biham, A.M. Youssef (eds.) Selected Areas in Cryptography, *Lecture Notes in Computer Science*, vol. 4356, pp. 266–284. Springer (2006)
27. Fujisaki, E., Okamoto, T.: Secure Integration of Asymmetric and Symmetric Encryption Schemes. In: M.J. Wiener (ed.) CRYPTO, *Lecture Notes in Computer Science*, vol. 1666, pp. 537–554. Springer (1999)
28. Giesbrecht, M., Roche, D.S., Tilak, H.: Computing Sparse Multiples of Polynomials. In: O. Cheong, K.Y. Chwa, K. Park (eds.) ISAAC (1), *Lecture Notes in Computer Science*, vol. 6506, pp. 266–278. Springer (2010)
29. Gilbert, H., Robshaw, M.J.B., Seurin, Y.: How to Encrypt with the LPN Problem. In: L. Aceto, I. Damgård, L.A. Goldberg, M.M. Halldórsson, A. Ingólfsdóttir, I. Walukiewicz (eds.) ICALP (2), *Lecture Notes in Computer Science*, vol. 5126, pp. 679–690. Springer (2008)
30. Goldreich, O., Goldwasser, S., Halevi, S.: Eliminating Decryption Errors in the Ajtai-Dwork Cryptosystem. In: Kaliski Jr. [35], pp. 105–111

31. Goldreich, O., Goldwasser, S., Halevi, S.: Public-Key Cryptosystems from Lattice Reduction Problems. In: Kaliski Jr. [35], pp. 112–131
32. Hallgren, S., Vollmer, U.: Quantum computing. In: D.J. Bernstein, J. Buchmann, E. Dahmen (eds.) Post-Quantum Cryptography, pp. 15–34. Springer (2009)
33. Herrmann, M., Leander, G.: A Practical Key Recovery Attack on Basic TCHo. In: S. Jarecki, G. Tsudik (eds.) Public Key Cryptography, *Lecture Notes in Computer Science*, vol. 5443, pp. 411–424. Springer (2009)
34. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A Ring-Based Public Key Cryptosystem. In: J. Buhler (ed.) ANTS, *Lecture Notes in Computer Science*, vol. 1423, pp. 267–288. Springer (1998)
35. Kaliski Jr., B.S. (ed.): Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17–21, 1997, Proceedings, *Lecture Notes in Computer Science*, vol. 1294. Springer (1997)
36. Kipnis, A., Patarin, J., Goubin, L.: Unbalanced Oil and Vinegar Signature Schemes. In: J. Stern (ed.) EUROCRYPT, *Lecture Notes in Computer Science*, vol. 1592, pp. 206–222. Springer (1999)
37. Lee, P.J. (ed.): Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5–9, 2004, Proceedings, *Lecture Notes in Computer Science*, vol. 3329. Springer (2004)
38. Lenstra, A., Lenstra, H., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen **261**(4), 515–534 (1982)
39. Li, Y.X., Deng, R.H., Wang, X.M.: On the equivalence of McEliece's and Niederreiter's public-key cryptosystems. IEEE Transactions on Information Theory **40**(1), 271 (1994)
40. Lu, Y., Meier, W., Vaudenay, S.: The Conditional Correlation Attack: A Practical Attack on Bluetooth Encryption. In: V. Shoup (ed.) CRYPTO, *Lecture Notes in Computer Science*, vol. 3621, pp. 97–117. Springer (2005)
41. Lu, Y., Vaudenay, S.: Cryptanalysis of Bluetooth Keystream Generator Two-Level E0. In: Lee [37], pp. 483–499
42. Lu, Y., Vaudenay, S.: Faster Correlation Attack on Bluetooth Keystream Generator E0. In: M.K. Franklin (ed.) CRYPTO, *Lecture Notes in Computer Science*, vol. 3152, pp. 407–425. Springer (2004)
43. Lu, Y., Vaudenay, S.: Cryptanalysis of an E0-like Combiner with Memory. Journal of Cryptology **21**(3), 430–457 (2008)
44. Lyubashevsky, V., Peikert, C., Regev, O.: On Ideal Lattices and Learning with Errors over Rings. In: H. Gilbert (ed.) EUROCRYPT, *Lecture Notes in Computer Science*, vol. 6110, pp. 1–23. Springer (2010)
45. Matsumoto, T., Imai, H.: Public Quadratic Polynominal-Tuples for Efficient Signature-Verification and Message-Encryption. In: C.G. Günther (ed.) EUROCRYPT, *Lecture Notes in Computer Science*, vol. 330, pp. 419–453. Springer (1988)
46. McEliece, R.: A public-key cryptosystem based on algebraic coding theory. DSN progress report **42**(44), 114–116 (1978)
47. Meier, W., Staffelbach, O.: Fast correlation attacks on certain stream ciphers. Journal of Cryptology **1**(3), 159–176 (1989)
48. Mozilla Corporation: Network Security Services (NSS) (2009). http://www.mozilla.org/projects/security/pki/nss/
49. Naccache, D. (ed.): Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001, San Francisco, CA, USA, April 8–12, 2001, Proceedings, *Lecture Notes in Computer Science*, vol. 2020. Springer (2001)
50. Niederreiter, H.: Knapsack-type cryptosystems and algebraic coding theory. Problems of Control and Information Theory **15**(2), 159–166 (1986)
51. Okamoto, T., Pointcheval, D.: REACT: Rapid Enhanced-Security Asymmetric Cryptosystem Transform. In: Naccache [49], pp. 159–175

52. Patarin, J.: Asymmetric Cryptography with a Hidden Monomial. In: N. Koblitz (ed.) CRYPTO, *Lecture Notes in Computer Science*, vol. 1109, pp. 45–60. Springer (1996)
53. Patarin, J., Courtois, N., Goubin, L.: FLASH, a Fast Multivariate Signature Algorithm. In: Naccache [49], pp. 298–307
54. Patarin, J., Courtois, N., Goubin, L.: QUARTZ, 128-Bit Long Digital Signatures. In: Naccache [49], pp. 282–297
55. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: M. Mitzenmacher (ed.) STOC, pp. 333–342. ACM (2009)
56. Rabin, M.: Digitalized signatures and public-key functions as intractable as factorization (1979)
57. Regev, O.: New lattice-based cryptographic constructions. J. ACM **51**(6), 899–942 (2004)
58. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: H.N. Gabow, R. Fagin (eds.) STOC, pp. 84–93. ACM (2005)
59. Regev, O.: Lattice-Based Cryptography. In: C. Dwork (ed.) CRYPTO, *Lecture Notes in Computer Science*, vol. 4117, pp. 131–141. Springer (2006)
60. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM **21**(2), 120–126 (1978)
61. Schnorr, C.P.: A Hierarchy of Polynomial Time Lattice Basis Reduction Algorithms. Theoretical Computer Science **53**, 201–224 (1987)
62. Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput. **26**(5), 1484–1509 (1997)
63. Shoup, V.: NTL: A Library for doing Number Theory. http://www.shoup.net/ntl/
64. Wagner, D.: A Generalized Birthday Problem. In: M. Yung (ed.) CRYPTO, *Lecture Notes in Computer Science*, vol. 2442, pp. 288–303. Springer (2002)
65. Zhuo, L., Prasanna, V.K.: High Performance Linear Algebra Operations on Reconfigurable Systems. In: SC, p. 2. IEEE Computer Society (2005)