Improving the Boneh-Franklin Traitor Tracing Scheme*

Pascal Junod[1], Alexandre Karlov[1,2], and Arjen K. Lenstra[2,3]

[1] Nagravision SA, CH-1033 Cheseaux-sur-Lausanne, Switzerland
[2] EPFL IC LACAL, Station 14, CH-1015 Lausanne, Switzerland
[3] Alcatel-Lucent Bell Laboratories, USA

Abstract. Traitor tracing schemes are cryptographically secure broadcast methods that allow identification of conspirators: if a pirate key is generated by k traitors out of a static set of ℓ legitimate users, then all traitors can be identified given the pirate key. In this paper we address three practicality and security issues of the Boneh-Franklin traitor-tracing scheme. In the first place, without changing the original scheme, we modify its tracing procedure in the non-black-box model such that it allows identification of k traitors in time $\tilde{O}(k^2)$, as opposed to the original tracing complexity $\tilde{O}(\ell)$. This new tracing procedure works independently of the nature of the Reed-Solomon code used to watermark private keys. As a consequence, in applications with billions of users it takes just a few minutes on a common desktop computer to identify large collusions. Secondly, we exhibit the lack of practical value of list-decoding algorithms to identify more than k traitors. Finally, we show that $2k$ traitors can derive the keys of *all* legitimate users and we propose a fix to this security issue.
Key words: Boneh-Franklin traitor tracing, Reed-Solomon codes, Berlekamp-Massey algorithm, Guruswami-Sudan algorithm

1 Introduction

Consider the following scenario: a center broadcasts data to ℓ receivers where only authorized users (typically, those who have paid a fee) should have access to the data. A way to realize this, widely deployed in commercial Pay-TV systems, is to encrypt the data using a symmetric key and to securely transmit to each authorized receiver this key which will be stored in a tamper-proof piece of hardware, like a smart card.

Unfortunately, tamper-resistant hardware is very difficult and costly to design, since it is vulnerable to a wide variety of attacks (see [1,27] as two good starting points). As a result, a malicious user (hereafter called a *traitor*) can attempt to retrieve the decryption key from his receiver and, if successful, distribute it (sell or give away) to unauthorized users (the *pirates*). Depending on the nature of the encryption schemes in use, we can even imagine situations where a dishonest user will try to *mix* several legitimate keys in order to build a new one and embed it in a pirate receiver.

The problem of identifying which receivers were compromised or which secret keys have leaked is called *traitor tracing*. Usually, two modes of traitor tracing are considered: in the *black-box* model, the tracing algorithm sends crafty ciphertexts to the pirate receiver and aims at determining which keys it uses while observing its behavior; in the *non-black-box model*, we assume that the keys can be extracted from the pirate receiver and are known to the tracing algorithm. The black-box model is widely considered by the cryptographic community as being a standard security model for evaluating traitor-tracing schemes security. However, based on our practical experience, we know that it is reasonable to assume that a tracing authority has at least the same technological and financial resources to reverse-engineer a pirate receiver as a traitor had, to perform the same operation on a legitimate receiver.

* Version of September 11, 2008

1.1 Related Work

Fiat and Naor introduced the concept of *broadcast encryption* in [17]. In their model[4], there exists a set of ℓ authorized users and the broadcasting center can *dynamically* specify a privileged subset of authorized users that can decrypt selected ciphertexts (like high-value content, for instance). Later, Chor, Fiat, and Naor [12] introduced the concept of traitor-tracing to overcome decryption key piracy in broadcast encryption schemes. Their scheme (which was improved by Naor and Pinkas in [13, 33]) is *k-collusion resistant* (or k-resilient) in the sense that at least one traitor can be identified with high probability given a pirate key generated by up to k traitors. Naor, Naor and Lotspiech presented more efficient broadcast encryption schemes [32] with tracing capabilities; it was however demonstrated by Kiayias and Pehlivanoglu [21] that the iterative nature of the tracing procedure allows a pirate to significantly leverage the compromise of a few keys. Although broadcast encryption and traitor-tracing are orthogonal problems in nature, and thus frequently studied separately, they are in practice indivisible: some *trace-and-revoke* schemes have been proposed accordingly [15, 16], culminating in [9]. The latter scheme, though resistant to any collusion size, is geared towards small-scale systems and impractical for the systems of tens of millions of users that we are dealing with and that inspired this paper; this is mainly due to the $O(\sqrt{\ell})$ complexity of [9] in terms of key storage and bandwidth requirements. Additionally, the tracing costs are $O(\ell^2)$, which also severely limits its applicability.

Kurosawa and Desmedt [24] proposed a *public-key* traitor tracing scheme, which was later broken by Stinson and Wei [40]. Boneh and Shaw [8] discussed collusion-resistant schemes for fingerprinting digital data based on error-correcting codes. Boneh and Franklin [5] proposed a new public-key traitor-tracing scheme also based on error-correcting codes, more precisely on Reed-Solomon codes. Actually, the traitor-tracing problem can be interpreted as an application of watermarking to secret keys that are distributed among users. The Boneh-Franklin non-black-box traitor tracing scheme is k-collusion resistant and deterministic in the sense that all of the traitors are identified with probability 1 if at most k of them collude to derive new pirate keys. The fastest claimed running time of the non-black-box tracing algorithm is $O(\ell \log \ell \log\log \ell)$ while the best known black-box tracing method has an exponential complexity $O(\binom{\ell}{k} k^2)$. Kurosawa and Yoshida [25] have generalized the Kurosawa-Desmedt and Boneh-Franklin schemes. The technique used by Boneh and Franklin to watermark private keys has since been re-used by Kiayias and Yung [23] to design an *asymmetric*[5] public-key traitor tracing scheme; other examples of Reed-Solomon codes use include schemes designed by Dodis *et al.* [15, 16]. Recently, Boneh *et al.* [7] have presented a *fully-collusion resistant* traitor tracing scheme which has private keys of constant size and ciphertexts of size $O(\sqrt{\ell})$. Finally, the low efficiency of tracing procedures in traitor tracing schemes has been addressed by Silverberg *et al.* in [37, 38]. The authors present several schemes based on algebraic codes which enable traitors to be traced in time polynomial in $k^2 \log \ell$. Recently, Billet and Phan [2] and Boneh and Naor [6] have independently proposed traitor-tracing schemes with constant size ciphertexts and having a black-box tracing complexity of $O(t^2 \ell \log \ell)$ and $O(t^4 \log \ell)$, respectively.

[4] Note that in this paper, we will only consider *stateless* receivers, i.e., receivers for which it is not possible to guarantee synchronism with the broadcast center and which are resettable. Broadcast encryption schemes for stateful receivers have been proposed in [41, 44].

[5] *Asymmetric* traitor tracing is a variant introduced by Pfitzmann [35] where the broadcasting center is not necessarily trusted, thus the tracing procedure must produce *undeniable* evidence of the implication of the traitor subscribers.

1.2 Our Contributions

While we agree that improving the exponential complexity of black-box tracing as cited above would be a very worthwhile cause to pursue, we choose to focus in this paper, in the light of the negative results obtained by Kiayias and Yung [22], on some security and efficiency issues that we encountered in practical applications of the Boneh-Franklin traitor-tracing scheme [5] in the *non-black-box model*. Although Boneh-Franklin traitor-tracing is one of the most elegant and efficient public-key traitor tracing schemes, it suffers from certain issues that limit its practical applicability in large-scale systems. We point out what the problems are and how they can be addressed. As usual, ℓ denotes the number of legitimate users and k the collusion threshold.

Complexity of Non-Black-Box Tracing One of the issues is the complexity of the non-black-box traitor tracing procedure which depends on ℓ. This is a major drawback when applied to systems of many millions of users, since tracing would require large computational power, or could even be infeasible in practice. We dissect the way Reed-Solomon codes are used to watermark private keys, and we show that, contrary to what is suggested in [5], it is possible to trace in time[6] $\tilde{O}(k^2)$, i.e., with a complexity independent of ℓ, using the Berlekamp-Massey algorithm instead of the Berlekamp-Welch algorithm. Although both algorithms require the same complexity to fully recover a noisy Reed-Solomon codeword, the complexity of the Berlekamp-Massey algorithm can be reduced if used for tracing only. The resulting new tracing procedure does not require any modification of the original Boneh-Franklin scheme. In practice, it takes us just a few minutes on a common desktop PC to trace large coalitions in systems having hundreds of millions of users. Our result improves the results obtained by Silverberg *et al.* [37, 38]. Our finding also applies to schemes using the same watermarking technique, such as the ones described in [15, 16, 23]. Another immediate benefit we identify is the possibility to use Reed-Solomon codes optimized specifically to allow faster decryption. In practice, for large systems and coalitions of medium size, we speed up the decryption by almost an order of magnitude.

Above-Threshold Tracing Secondly, we raise an issue concerning the above-threshold security of the Boneh-Franklin scheme and its variants. We show that the list-decoding techniques, such as the Guruswami-Sudan algorithm, as advocated by Boneh-Franklin to trace more than k traitors, detect only a few additional traitors, and this at a high cost.

Beyond-Threshold Tracing Finally, we show that if an adversary is able to recover $2k$ secret keys, then she is able to compute *any* other secret key, including the uncompromised ones. Thus, in this case the security of the system completely collapses. This somewhat embarrassing property is primarily due to the fact that the linear tracing code is public. We show how this issue can be addressed at the cost of keeping more than a single secret value in the receivers.

This paper is organized as follows. In §2 we review the Boneh-Franklin scheme [5]. Then, in §3, we speed up both its codeword generation and tracing procedures. In §4 we discuss the above-threshold tracing based on the Guruswami-Sudan list-decoding algorithm, while in §5 we study the security of the Boneh-Franklin scheme when the number of recovered secret keys is at least twice the allowed threshold.

[6] Here, the $\tilde{O}(n)$ notation hides the terms which are poly-logarithmic in n.

2 Boneh-Franklin Scheme

This section describes the Boneh-Franklin traitor tracing scheme [5] by first defining its encryption and decryption procedures, then by explaining the codeword generation mechanism and finally by describing the underlying non-black-box tracing mechanism. We adopt the notation used in [5] denoting by ℓ the number of users in the system and by k the maximal coalition size. Hence, the described scheme is supposed to be secure against any collusion of at most k users.

2.1 Encryption/Decryption

Let $\mathbb{G}_q$ denote a group of prime order q in which the Decision Diffie-Hellman problem [4] is hard. Typically, $\mathbb{G}_q$ is a subgroup of order q of $\mathbb{Z}_p^*$, where p is prime and $q|p-1$; alternatively, $\mathbb{G}_q$ can be a group of points of an elliptic curve over a finite field.

The key generation process proceeds as follows. Let g be a generator of $\mathbb{G}_q$. For $1 \leq j \leq 2k$, let $r_j \in_R \mathbb{Z}/q\mathbb{Z}$ and compute $h_j = g^{r_j}$. The *public key* is defined as $\langle y, h_1, \ldots, h_{2k}\rangle \in \mathbb{G}_q^{2k+1}$ where $y = \prod_{j=1}^{2k} h_j^{\alpha_j} \in \mathbb{G}_q$ for random $\alpha_j \in_R \mathbb{Z}/q\mathbb{Z}$. Here, we say that the vector $\boldsymbol{\alpha} = \langle \alpha_1, \ldots, \alpha_{2k}\rangle$ is a *representation* of y with respect to the base $\langle h_1, \ldots, h_{2k}\rangle$. Note that if $\boldsymbol{\rho}^{(1)}, \ldots, \boldsymbol{\rho}^{(n)}$ are n representations of the same element of $\mathbb{G}_q$ with respect to the same base, then so is any convex combination $\sum_{i=1}^{n} \eta_i \boldsymbol{\rho}^{(i)}$ of the representations, where $\eta_i \in \mathbb{Z}/q\mathbb{Z}$ are scalars such that $\sum_{i=1}^{n} \eta_i = 1$.

Let $\Gamma = \{\boldsymbol{\gamma}^{(1)}, \ldots, \boldsymbol{\gamma}^{(\ell)}\}$ be a *linear space tracing code*, i.e., a collection of ℓ codewords $\boldsymbol{\gamma}^{(i)}$, for $1 \leq i \leq \ell$, where each $\boldsymbol{\gamma}^{(i)} = \langle \gamma_j^{(1)}, \ldots, \gamma_j^{(2k)}\rangle$ is a $2k$-dimensional vector over $\mathbb{Z}/q\mathbb{Z}$. The set Γ is fixed in advance and not secret, and can thus be considered as being a public parameter of the Boneh-Franklin traitor tracing scheme. We detail in §2.2 the codeword generation process from [5]. In §3.2 we propose a slightly different way to define Γ that has interesting practical consequences.

A private key is an element $\theta_i \in \mathbb{Z}/q\mathbb{Z}$ such that $\theta_i \cdot \boldsymbol{\gamma}^{(i)}$ is a representation of y with respect to the base $\langle h_1, \ldots, h_{2k}\rangle$. Thus, the i-th private key θ_i can be derived from the i-th codeword $\boldsymbol{\gamma}^{(i)}$ as

$$\theta_i = \frac{\sum_{j=1}^{2k} r_j \alpha_j}{\sum_{j=1}^{2k} r_j \gamma_j^{(i)}}, \tag{1}$$

where, obviously, the calculation takes place in $\mathbb{Z}/q\mathbb{Z}$. To encrypt a message $m \in \mathbb{G}_q$, one picks a random $a \in_R \mathbb{Z}/q\mathbb{Z}$ and calculates the ciphertext as $\langle m \cdot y^a, h_1^a, \ldots, h_{2k}^a\rangle$. Given a ciphertext $\langle s, p_1, \ldots, p_{2k}\rangle$, and the i-th secret key θ_i, the message m can be recovered as:

$$m = \frac{s}{\left(\prod_{j=1}^{2k} p_j^{\gamma_j^{(i)}}\right)^{\theta_i}}. \tag{2}$$

The correctness follows in a straightforward way from the fact that $\theta_i \cdot \boldsymbol{\gamma}^{(i)}$ is a representation of y with respect to the base $\langle h_1, \ldots, h_{2k}\rangle$. It follows that it is possible to decrypt a ciphertext given *any* representation $\langle \delta_1, \ldots, \delta_{2k}\rangle$ of y with respect to the base $\langle h_1, \ldots, h_{2k}\rangle$, since $\prod_{j=1}^{2k} (h_j^a)^{\delta_j} = y^a$; in other words, to decrypt it suffices to have a representation of y with respect to the base $\langle h_1, \ldots, h_{2k}\rangle$. Interestingly, Boneh and Franklin show in [5, Lemma 1] that if it is infeasible to compute discrete logarithms in $\mathbb{G}_q$, then convex combinations of $n < 2k$ given representations $\boldsymbol{\rho}^{(1)}, \ldots, \boldsymbol{\rho}^{(n)}$ of y are the *only* representations of y that can efficiently be constructed.

2.2 Codewords Generation

We describe the codewords $\boldsymbol{\gamma}^{(i)}$ generation process from [5] which is based on the use of Reed-Solomon codes [36]. Given the $(\ell - 2k) \times \ell$ matrix

$$\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 3 & \dots & \ell \\ 1^2 & 2^2 & 3^2 & \dots & \ell^2 \\ \vdots & \vdots & \vdots & & \vdots \\ 1^{\ell-2k-1} & 2^{\ell-2k-1} & 3^{\ell-2k-1} & \dots & \ell^{\ell-2k-1} \end{pmatrix} \bmod q \tag{3}$$

over $\mathbb{Z}/q\mathbb{Z}$, let $\boldsymbol{b}_1, \dots, \boldsymbol{b}_{2k}$ be a basis of the nullspace of $\boldsymbol{A}$. Boneh and Franklin define Γ as the rows of the $\ell \times 2k$ matrix

$$\boldsymbol{B} = \begin{pmatrix} | & | & | & & | \\ \boldsymbol{b}_1 & \boldsymbol{b}_2 & \boldsymbol{b}_3 & \dots & \boldsymbol{b}_{2k} \\ | & | & | & & | \end{pmatrix}, \tag{4}$$

also over $\mathbb{Z}/q\mathbb{Z}$. Thus, Γ contains ℓ codewords each of length $2k$. By observing that any vector in the span of the rows of $\boldsymbol{A}$ corresponds to a polynomial of degree at most $\ell - 2k - 1$ evaluated at the points $1, \dots, \ell$, one can construct a basis of the nullspace of $\boldsymbol{A}$ using Lagrange interpolation. Using this construction the i-th codeword becomes $\langle u_i, iu_i, i^2u_i, \dots, i^{2k-1}u_i \rangle$ where $u_i^{-1} = \prod_{j \neq i}(i-j)$ and all computations are in $\mathbb{Z}/q\mathbb{Z}$. Naive computation of the ℓ codewords requires $\Omega(\ell^2)$ operations in $\mathbb{Z}/q\mathbb{Z}$. This can easily be turned into $O(\ell)$ operations using the following recursive formula:

$$u_1^{-1} = \prod_{j=1}^{\ell-1}(-j) \text{ and } u_{i+1}^{-1} = \frac{u_i(i-1)}{i-\ell} \text{ for } 1 \leq i \leq \ell - 1. \tag{5}$$

2.3 Tracing Procedure

We briefly recall the non-black-box tracing procedure [5]. Let $\boldsymbol{d} \in (\mathbb{Z}/q\mathbb{Z})^{2k}$ be a vector formed by taking a linear combination of at most k vectors in Γ. In practice $\boldsymbol{d}$ will be a convex combination, but we do not need that here. Since the vectors in Γ form the rows of the matrix $\boldsymbol{B}$, we know there exists a vector $\boldsymbol{w} \in (\mathbb{Z}/q\mathbb{Z})^\ell$ (having Hamming weight at most k) such that $\boldsymbol{wB} = \boldsymbol{d}$. The tracing procedure then works as follows. First, we determine a vector[7] $\boldsymbol{v} \in (\mathbb{Z}/q\mathbb{Z})^\ell$ such that $\boldsymbol{vB} = \boldsymbol{d}$. Since $(\boldsymbol{v} - \boldsymbol{w})\boldsymbol{B} = \boldsymbol{0}$, we know that $\boldsymbol{v} - \boldsymbol{w}$ lies in the linear span of the rows of $\boldsymbol{A}$ (recall that the rows of $\boldsymbol{A}$ span the vector space orthogonal to the one spanned by the columns of $\boldsymbol{B}$). In other words, there exists a unique polynomial f of degree at most $\ell - 2k - 1$ over $\mathbb{Z}/q\mathbb{Z}$ such that $\boldsymbol{v} - \boldsymbol{w} = \langle f(1), \dots, f(\ell) \rangle$. Taking into account that $\boldsymbol{w}$ has Hamming weight of at most k, we know that $\langle f(1), \dots, f(\ell) \rangle$ equals $\boldsymbol{v}$ in all but at most k components. Hence, it is possible to use Berlekamp-Welch algorithm [42] to find f from $\boldsymbol{v}$, after which f gives us $\boldsymbol{v} - \boldsymbol{w}$, from which we recover $\boldsymbol{w}$. The Berlekamp-Welch algorithm, published in a patent [42] granted in 1986, runs in $O(\ell^2)$. Asymptotically faster variants exist (see [3]), the fastest known being the one described by Pan [34] which runs in $O(\ell \log \ell \log\log \ell)$.

As mentioned in §1.1, the best known black-box tracing procedure for the Boneh-Franklin scheme is not efficient since it has a $O(\binom{\ell}{k} k^2)$ complexity. We refer the reader to [5] for its description since it is out of the scope of this paper. Furthermore, we note that the black-box tracing

[7] Note that several such vectors exist.

procedure is vulnerable to the attacks described by Kiayias and Yung [22] which demonstrate that the Boneh-Franklin scheme is essentially incapable of black-box tracing super-logarithmic self-protecting traitor collusions unless the ciphertext size is linear in the number of users. Those two facts considerably limit the application of black-box tracing with the Boneh-Franklin scheme.

3 Revisiting the Tracing Mechanism

We recall several notions from coding theory. A *linear code* $\mathcal{C}$ over the vector space $(\mathbb{Z}/q\mathbb{Z})^\ell$ is a *subspace* of $(\mathbb{Z}/q\mathbb{Z})^\ell$. For our purposes we may assume that $\mathcal{C}$ has dimension $2k$ with $0 \leq 2k \leq \ell$. It follows that $\mathcal{C}$ contains q^{2k} *codewords*. The *minimal distance* d *of* $\mathcal{C}$ is the minimum Hamming weight of its non-zero codewords. A code $\mathcal{C}$ is called *maximum-distance separable* (*MDS*) if its minimal distance reaches the Singleton bound, i.e., if $d = \ell - 2k + 1$. A $2k \times \ell$ matrix $\boldsymbol{G}$ over $\mathbb{Z}/q\mathbb{Z}$ is called a *generator matrix* or *encoding matrix for* $\mathcal{C}$ if its rows form a linearly independent basis for $\mathcal{C}$. Thus, $\mathcal{C} = \{\boldsymbol{x} \in (\mathbb{Z}/q\mathbb{Z})^\ell : \boldsymbol{x} = \boldsymbol{z}\boldsymbol{G} \text{ where } \boldsymbol{z} \in (\mathbb{Z}/q\mathbb{Z})^{2k}\}$ and $\mathcal{C}$ is the code *associated* to G. The *dual code* $\mathcal{C}^\perp$ of a linear code $\mathcal{C}$ is the linear code $\mathcal{C}^\perp = \left\{\boldsymbol{x} \in (\mathbb{Z}/q\mathbb{Z})^\ell : \boldsymbol{x}\boldsymbol{c}^T = \boldsymbol{0} \text{ for all } \boldsymbol{c} \in \mathcal{C}\right\}$. A *reduced parity-check matrix* for the code $\mathcal{C}$ is an $(\ell - 2k) \times \ell$ matrix $\boldsymbol{H}$ over $\mathbb{Z}/q\mathbb{Z}$ such that $\mathcal{C} = \left\{\boldsymbol{x} \in (\mathbb{Z}/q\mathbb{Z})^\ell : \boldsymbol{x}\boldsymbol{H}^T = \boldsymbol{0}\right\}$. Receiving a noisy version $\tilde{\boldsymbol{x}}$ of a codeword $\boldsymbol{x}$, the vector $\boldsymbol{s} = \tilde{\boldsymbol{x}}\boldsymbol{H}^T$ is called the *syndrome*. Writing $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{e}$, where $\boldsymbol{e}$ is called the *error pattern*, we see that the syndrome $\boldsymbol{s} = (\boldsymbol{x} + \boldsymbol{e})\boldsymbol{H}^T = \boldsymbol{0} + \boldsymbol{e}\boldsymbol{H}^T = \boldsymbol{e}\boldsymbol{H}^T$ depends only on the error pattern, and not on the codeword itself. Finally, the following lemma makes clear the link between a reduced parity-check matrix of a linear code and its dual code.

Lemma 1. *$\boldsymbol{H}$ is a parity-check matrix for the linear code $\mathcal{C}$ if and only if $\mathcal{C}$ spans the subspace orthogonal to the row space of $\boldsymbol{H}$.*

Therefore, a reduced parity-check matrix for $\mathcal{C}$ is an encoding matrix for the dual code $\mathcal{C}^\perp$ and conversely.

3.1 Generalized Reed-Solomon Codes

Given vectors $\boldsymbol{\pi} = (\pi_i)_{i=1}^\ell, \boldsymbol{c} = (c_i)_{i=1}^\ell \in (\mathbb{Z}/q\mathbb{Z})^\ell$, a *Generalized Reed-Solomon code* $\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{c})$ is defined as follows:

$$\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{c}) = \left\{(c_i f(\pi_i))_{i=1}^\ell : f(x) \in (\mathbb{Z}/q\mathbb{Z})[x] \text{ and } \deg(f) < 2k\right\}. \tag{6}$$

Thus, a codeword in $\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{c})$ is a vector consisting of a polynomial of degree less than $2k$ over $\mathbb{Z}/q\mathbb{Z}$ evaluated at the ℓ points $\pi_1, \ldots, \pi_\ell$ scaled by $c_1, \ldots, c_\ell$. It is well-known that GRS codes are MDS codes, i.e., $d = \ell - 2k + 1$. When $\boldsymbol{c} = (1, 1, \ldots, 1)$, we speak of Reed-Solomon codes. The following theorem states that the dual of a GRS code is a GRS code (see [20, page 66] for a proof).

Theorem 1. *The dual of a* $\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{c})$ *code is*

$$\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{c})^\perp = \mathrm{GRS}_{\ell,\ell-2k}(\boldsymbol{\pi}, \boldsymbol{d}) \tag{7}$$

where $\boldsymbol{d} = (d_1, \ldots, d_\ell)$ *with* $d_i^{-1} = c_i \prod_{j \neq i}(\pi_i - \pi_j)$.

The above allows us to rephrase the Boneh-Franklin codeword generation mechanism described in §2.2 as follows: the matrix $\boldsymbol{A}$ defined in (3) is the generator matrix of a $\mathrm{GRS}_{\ell,\ell-2k}(\boldsymbol{\pi}, \boldsymbol{c})$ code

over $\mathbb{Z}/q\mathbb{Z}$ with $\boldsymbol{\pi} = (1, \ldots, \ell)$ and $\boldsymbol{c} = (1, 1, \ldots, 1)$ (this fact was already recognized by [23], for instance), while the matrix $\boldsymbol{B}$ defined in (4) is a (transposed) reduced parity-check matrix for the same code. Conversely, in the light of Lemma 1 and Theorem 1, the matrix $\boldsymbol{B}^T$ can be seen as a generator matrix of the dual $\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{d})$ of $\mathrm{GRS}_{\ell,\ell-2k}(\boldsymbol{\pi}, \boldsymbol{c})$, where $\boldsymbol{d}$ is as in Theorem 1. Thus, Γ consists of vectors forming a basis of the $2k$-dimensional vector space which contains the syndromes of $\mathrm{GRS}_{\ell,\ell-2k}(\boldsymbol{\pi}, \boldsymbol{c})$.

3.2 More Efficient Codewords

The above more general framework allows us to define the code Γ in such a way that both the codeword generation and decryption become faster without affecting the security of the Boneh-Franklin scheme.

Using Theorem 1, we observe that in order to compute the codewords we can avoid Lagrange interpolation and recursive formula (5): let $\boldsymbol{B}$ be the generator matrix of a $\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{d})$ code with $\boldsymbol{\pi} = (1, 2, \ldots, \ell)$ and $\boldsymbol{d} = (1, 1, \ldots, 1)$, then the i-th codeword can simply be defined as $\boldsymbol{\gamma}^{(i)} = \langle 1, i, i^2, \ldots, i^{2k-1} \rangle$, for $i = 1, 2, \ldots, \ell$. This in turn allows us to rewrite the decryption operation (2) as

$$m = \frac{s}{\left(\prod_{j=1}^{2k} p_j^{i^{j-1}}\right)^{\theta_i}} = \frac{s}{\left(\left(\left(\left(p_{2k}^i p_{2k-1}\right)^i \ldots\right)^i p_2\right)^i p_1\right)^{\theta_i}}. \tag{8}$$

Compared to (2), this replaces $2k$ of the $2k+1$ $\log_2 q$-bit modular exponentiation exponents by $\log_2 \ell$-bit ones. With $\ell \approx 2^{20}$ and assuming 80-bit security with 160-bit q, this results in a speedup by a factor of 7, which is much more effective than using multi-exponentiations (cf. [30, page 617]) as suggested in [5]. In practice, the efficiency of our decryption is comparable to [29]. Furthermore, provided each receiver knows its identity number i, it can directly compute codeword $\boldsymbol{\gamma}^{(i)}$ without requiring knowledge of the Lagrange coefficients attached to the receiver with identity $i-1$.

We note that the semantic security of the Boneh-Franklin scheme is not impacted by the nature of the code, while its tracing capabilities only depend on the minimal distance of the code. In our case, we use Generalized Reed-Solomon codes with the same minimal distance as the one used by Boneh and Franklin.

3.3 An Efficient Tracing Procedure

In this section, we present in two steps a new and efficient non-black-box tracing procedure for the Boneh-Franklin scheme. We stress that this new tracing procedure can be used for any type of Reed-Solomon and generalized Reed-Solomon codes, being the original code described in [5], the faster code discussed in §3.2 or the variant we will discuss in §5. First, we reduce the complexity from $O(\ell \log \ell \log\log \ell)$ to $O(\ell)$, using a technique based on the Berlekamp-Massey algorithm [28] and Chien search [11]. Then, we improve it to expected complexity $\tilde{O}(k^2)$ by replacing Chien search by the Cantor-Zassenhaus factorization algorithm [10]

As outlined in §2.3, the Boneh-Franklin tracing procedure based on Berlekamp-Welch algorithm consists in finding a noisy codeword which results in the syndrome discovered in the pirate receiver, and in decoding this codeword. More precisely, let $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ denote a codeword belonging to $\mathrm{GRS}_{\ell,\ell-2k}(\boldsymbol{\pi}, \boldsymbol{c})$ with $\boldsymbol{c} = (1, 1, \ldots, 1)$ and its noisy version, respectively. We can interpret both $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ as polynomials $f(x)$ and $\tilde{f}(x)$ in $(\mathbb{Z}/q\mathbb{Z})[x]$. If no error is introduced in the codeword,

then $d_i f(\pi_i) = \tilde{f}_i$ for $1 \leq i \leq \ell$, where $\tilde{f}(x) = \sum_{i=1}^{\ell} \tilde{f}_i x^{i-1}$. Let $g(x) \in (\mathbb{Z}/q\mathbb{Z})[x]$ be a polynomial (hereafter called an error-locator polynomial) of degree at most k with $g(\pi_i) = 0$ for those π_i's for which $d_i f(\pi_i) \neq \tilde{f}_i$. This leads to the following system of ℓ linear equations in ℓ unknowns: $d_i f(\pi_i) g(\pi) = g(\pi)\tilde{f}_i$. Solving the system, one obtains the polynomial $g(x)$, from which the error locations can be derived. Along with $g(x)$, one also gets $d_i f(x) g(x)$ and thus $f(x)$. Straightforward implementation using Gaussian reduction would lead to $O(\ell^3)$ complexity. Faster approaches would be to use the Berlekamp-Welch algorithm in $O(\ell^2)$ or others of complexity $\tilde{O}(\ell)$ (see [3, 34, 42]).

The key observation to derive a faster tracing algorithm is to note that computing a (noisy) codeword from the syndrome retrieved from a pirate receiver and then decoding this codeword, as done above, is *not* necessary: actually, the pirate syndrome itself suffices to trace the legitimate syndromes used to derive it. Indeed, as pointed out by Massey [28], the *Berlekamp-Massey algorithm* allows reconstruction of the error-locator polynomial from the syndrome only. This key property permits us to stop the decoding process earlier for the purpose of tracing and thus reduce the complexity, since we are interested in the error-locator polynomial only and we do not need the amplitudes of the errors.

We now clarify the link between the error-locator polynomial and the syndrome, following [43, page 214]. Let $\tilde{f}(x) = f(x) + e(x)$, where $\tilde{f}(x)$, $f(x)$ and $e(x)$ are the received codeword, the original codeword, and the error polynomial, respectively. Let $s(x) = s_0 + s_1 x + \cdots + s_{2k-1} x^{2k-1}$ denote the syndrome vector interpreted as a polynomial. Let $g(x)$ denote an error-locator polynomial whose zeroes are the *inverses* of the error locations $\sigma_j = \pi_i$ with $1 \leq j \leq k$ and with $i \in \mathcal{I}$ for a cardinality k subset $\mathcal{I}$ of $\{1, 2, \ldots, \ell\}$:

$$g(x) = \prod_{j=1}^{k} (1 - \sigma_j x) = g_0 + g_1 x + \cdots + g_k x^k. \tag{9}$$

Let $t_1, t_2, \ldots, t_k$ be the indices of the non-zero coefficients of $e(x)$. Because $g(\sigma_m^{-1}) = 0$ for all error locations σ_m with $1 \leq m \leq k$, it follows that $e_{t_m} \sigma_m^j g(\sigma_m^{-1}) = 0$, and thus

$$e_{t_m} (g_k \sigma_m^{-k+j} + g_{k-1} \sigma_m^{-k+j+1} + \cdots + g_1 \sigma_m^{j-1} + g_0 \sigma_m^j) = 0 \tag{10}$$

for any j. Summing (10) over $m = 1, 2, \ldots k$ gives an expression from which Newton's identities can be constructed:

$$\begin{aligned} & \textstyle\sum_{m=1}^{k} e_{t_m} (g_k \sigma_m^{-k+j} + g_{k-1} \sigma_m^{-k+j+1} + \cdots + g_1 \sigma_m^{j-1} + g_0 \sigma_m^j) \\ = \; & g_k \textstyle\sum_{m=1}^{k} e_{t_m} \sigma_m^{j-k} + g_{k-1} \sum_{m=1}^{k} e_{t_m} \sigma_m^{j-k+1} + \cdots + g_0 \sum_{m=1}^{k} e_{t_m} \sigma_m^j \\ = \; & g_k s_{j-k} + g_{k-1} s_{j-k+1} + \cdots + g_1 s_{j-1} + g_0 s_j = 0. \end{aligned}$$

The last equality comes from the fact that the following system of equations can be written using the parity-check matrix:

$$\begin{aligned} s_0 &= e_{t_1} + e_{t_2} + \cdots + e_{t_k} \\ s_1 &= e_{t_1} \sigma_1 + e_{t_2} \sigma_2 + \cdots + e_{t_k} \sigma_k \\ s_2 &= e_{t_1} \sigma_1^2 + e_{t_2} \sigma_2^2 + \cdots + e_{t_k} \sigma_k^2 \\ &\cdots \\ s_{2k-1} &= e_{t_1} \sigma_1^{2k-1} + e_{t_2} \sigma_2^{2k-1} + \cdots + e_{t_k} \sigma_k^{2k-1}. \end{aligned}$$

From (9) it follows that $g_0 = 1$, which leads to the order k linear recurrence relation

$$g_k s_{j-k} + \cdots + g_1 s_{j-1} = -s_j. \tag{11}$$

Given $2k$ consecutive terms of an order k linear recurrence, the Berlekamp-Massey algorithm computes the coefficients of the recurrence in time $O(k^2)$. Because the s_i for $i = 0, 1, \ldots, 2k-1$ are known, the g_i can thus be computed directly in time $O(k^2)$.

After the error-locator polynomial $g(x)$ has been computed, the remaining task consists in finding its roots, which are the inverses of identities of the traitors. Traditionally, Reed-Solomon decoders rely on the Chien search algorithm [11] which searches over the possible roots. In our case, this results in a complexity of $O(\ell)$. The roots can, however, be located faster by *factorizing* $g(1/x)$ using the Cantor-Zassenhaus algorithm [10] within expected time $O(k^2 \log k \log\log k(\log q + \log k)) = \tilde{O}(k^2)$. This algorithm works recursively on the squarefree polynomial $g(x)$ whose irreducible factors[8] are all of degree 1. It is based on the fact[9] that $g(x) = \gcd(g(x), r(x)) \cdot \gcd(g(x), r(x)^{(p^d-1)/2} + 1) \cdot \gcd(g(x), r(x)^{(p^d+1)/2} - 1)$ for any polynomial $r(x) \in (\mathbb{Z}/q\mathbb{Z})[x]$.

Then, the obtained roots directly reveal the identities of the traitors. The overall complexity of our tracing procedures is $\tilde{O}(k^2)$ which is independent of ℓ. The latter is not the case for the schemes based on algebraic codes described by Silverberg *et al.* in [37, 38].

Our method makes it possible to trace large coalitions in Boneh-Franklin systems with a virtually *unlimited* number of users, and this without requiring any modification of the encryption scheme. Our implementation, based on the `GMP` [31] and `LiDIA` [26] software libraries and working over the group of points of an elliptic curve over a finite field of cardinality approximately 2^{160}, allows tracing of a coalition of $k = 1024$ traitors in a system of $\ell = 200'000'000$ users in less than *two minutes* on a common desktop PC. These parameter values cannot realistically be handled using the Berlekamp-Welch algorithm as described in [5].

4 Above-Threshold Tracing

In [5] Boneh and Franklin emphasize an interesting property of their scheme, namely the possibility to trace a collusion of more than k traitors using list-decoding techniques like the Guruswami-Sudan algorithm [18,19]. This would correspond to finding more than k errors in a codeword. In such cases, the Berlekamp-Welch algorithm fails to find the polynomial $f(x)$. The Berlekamp-Massey approach fails as well, since it outputs a polynomial of degree k that does not have k roots over $\mathbb{Z}/q\mathbb{Z}$. The algorithm of Guruswami and Sudan allows, under certain circumstances, to find a candidate for the polynomial $f(x)$. In this section we investigate under which circumstances tracing is possible and how it will influence system parameters. We finally show that the Guruswami-Sudan algorithm can detect only a few additional traitors, and this at high cost.

4.1 Guruswami-Sudan Algorithm for Reed-Solomon Codes

This algorithm attempts to find the message polynomial $f(x)$ given a received codeword when more than k errors occurred. It can be thought of as a generalization of the Berlekamp-Welch algorithm. Let ℓ and k be as above. Given ℓ pairs $(\pi_i, c_i) \in (\mathbb{Z}/q\mathbb{Z})^2$ for $1 \le i \le \ell$, message length $\ell - 2k$, and an error parameter $k' \le \ell - 1 - \sqrt{\ell(\ell - 2k - 1)}$, the Guruswami-Sudan algorithm finds all

[8] g(x) in fact fulfills these conditions if $g(x)$ has at most k roots.
[9] The interested reader will find more details about the Cantor-Zassenhaus algorithm in [14, page 128].

univariate polynomials f of degree at most $\ell - 2k - 1$ such that $f(\pi_i) = c_i$ for at least $\ell - k'$ values of i. Thus, the algorithm allows correction of at most k' errors. It consists of two steps. In the first step a parameter r is selected and a system of $O(\ell r^2)$ linear equations is solved to find a non-zero bivariate polynomial $Q(x, y)$ of a certain weighted degree[10] such that $Q(\pi_i, c_i) = 0$ for $1 \leq i \leq \ell$. The parameter r, which is the multiplicity of the singularity of $Q(x, y)$, is chosen in such a way that as many errors as possible can be handled while keeping the system of equations tractable. In the second step, factors $(y - f(x))$ of $Q(x, y)$ are determined such that $\deg(f(x)) \leq \ell - 2k - 1$. For a complete description of the method see [18,19]. Below we are interested in its practical feasibility (in particular of the first step) in the context of the traitor tracing problem.

4.2 List Decoding and Traitor Tracing

In this section we have a closer look at the various parameters of of the Guruswami-Sudan algorithm. We will see that this leads to the unavoidable conclusion that it is of little practical significance for our type of applications.

Since the traditional algorithms (such as Berlekamp-Welch) can trace up to k traitors, the only case of interest is $k' > k$. Let $\delta = k' - k$ be the number of additional traitors we wish to be able to trace, and let $\phi = \ell - 2k - 1$. Because at most $\ell - 1 - \sqrt{\ell\phi}$ traitors can be traced, only k's need to be considered for which

$$k + \delta \leq \ell - 1 - \sqrt{\ell\phi} \tag{12}$$

for a $\delta \geq 1$.

With $\omega = r(\ell - k - \delta) - 1$, in the first step of the Guruswami-Sudan algorithm a system needs to be solved over $\mathbb{Z}/q\mathbb{Z}$ involving $\ell r(r + 1)/2$ constraints and

$$\left(\omega + 1 - \frac{\phi}{2}\left\lfloor\frac{\omega}{\phi}\right\rfloor\right)\left(\left\lfloor\frac{\omega}{\phi}\right\rfloor + 1\right)$$

unknowns [18,19]. It follows that

$$\left(\omega + 1 - \frac{\phi}{2}\left\lfloor\frac{\omega}{\phi}\right\rfloor\right)\left(\left\lfloor\frac{\omega}{\phi}\right\rfloor + 1\right) \geq \frac{\ell r(r + 1)}{2}. \tag{13}$$

Furthermore, since in practice q will have at least 160 bits, it is reasonable to limit the number of constraints to 10000 if we want to be able to store the matrix in 2GB of memory. This leads to

$$\frac{\ell(r + 1)r}{2} < 10000. \tag{14}$$

Note that this immediately limits the practical applicability of the Guruswami-Sudan algorithm to tracing in systems of at most a few thousand users. This is in sharp contrast with our syndrome-only tracing which allows millions of users.

Define the *minimal coalition size* as the smallest k such that (12), (13), and (14) are satisfied. For any ℓ and δ, this k follows from a simple search, as illustrated in Fig. 1 for several (small) numbers of users. For example, in a system with $\ell = 512$ users the minimal initial coalition size is 69 in order to be able to trace a single additional key if 70 pirates collude. In many applications, this results in an overkill, because the ciphertext and private key, which are dependent on the coalition

[10] $\deg_x(Q(x, y))m + \deg_y(Q(x, y))n$ is called the (m, n)-weighted degree of $Q(x, y)$.

size, become too large. As illustration, let us consider the following case: for $\ell = 1024$ and $k = 500$, we get $k + \delta = 855$, which may seem fairly good. However, the required bandwidth to transmit the ciphertext is equal to 1001 group elements. This is only 2.24% less than a trivial scheme involving an individual encryption based on El-Gamal which additionally would bring natural revocation capabilities. Besides that, a system of size $\ell = 1024$ is not far from the limit capacity of the original Berlekamp-Welch algorithm. Hence this method is not applicable for systems with large number of users, constrained bandwidth and key-space storage capability.

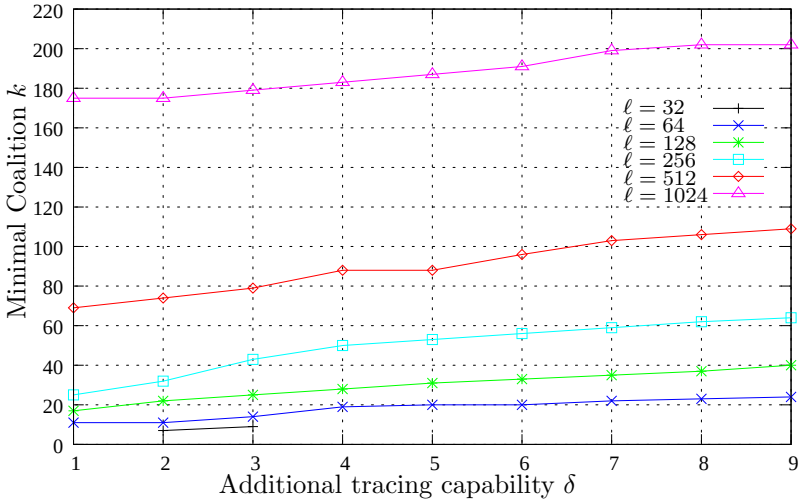

Fig. 1. Minimal coalition with respect to a given above-threshold tracing capacity

5 Beyond-Threshold Security

In practical scenarios, there are three distinct cases for the number of compromised keys in a coalition, namely: at most k, between $k + 1$ and $2k - 1$, and $2k$ keys or more. The first case corresponds to the situation for which the Boneh-Franklin scheme has been designed and security guarantees have been derived, while the second case corresponds to the above-threshold tracing scenario described in §4. In this section we discuss the third case.

Suppose that an adversary has managed to get $2k$ private elements θ_{i_s}, for $1 \leq s \leq 2k$ and assume, as before, that the vectors in Γ are public. Because Eq. (1) over $\mathbb{Z}/q\mathbb{Z}$ can be rewritten as

$$\theta_{i_s}^{-1} = \frac{\sum_{j=1}^{2k} r_j \gamma_j^{(i_s)}}{\sum_{j=1}^{2k} r_j \alpha_j} = \sum_{j=1}^{2k} \omega_j \gamma_j^{(i_s)} \tag{15}$$

with $\omega_j = r_j / \sum_{j=1}^{2k} r_j \alpha_j$, knowledge of the $2k$ private keys θ_{i_s} leads to a system of $2k$ linear equations in the $2k$ unknowns ω_j, for $1 \leq j \leq 2k$. After determining the ω_j's using for instance Gaussian reduction, the adversary can compute *any* other private key θ_i in the system:

$$\theta_i = \left(\sum_{j=1}^{2k} \omega_j \gamma_j^{(i)} \right)^{-1}.$$

Not only will the adversary be able to create any number of untraceable combinations of keys, but he will also be able to distribute newly derived keys so that *innocent* users (whose keys were *a priori* never compromised) may be accused of treachery. We note that this observation applies not only to the Boneh-Franklin scheme, but to many tracing schemes that are based on a publicly-known linear code such as the generalizations described by Kurosawa and Yoshida [25].

An obvious way to repair this annoying property of the Boneh-Franklin scheme would require keeping the tracing code matrix secret, while making sure that the vectors $\boldsymbol{\gamma}^{(i)} = \langle \gamma_1^{(i)}, \ldots, \gamma_{2k}^{(i)} \rangle$ are statistically decorrelated. In that case acquiring $2k$ representations should give an adversary no information about other representations. This idea was already used by Kiayias and Yung in [23] for the different goal of obtaining an asymmetric traitor-tracing scheme. A way to achieve this would be to choose the i-th codeword $\gamma^{(i)}$ as $\boldsymbol{\gamma}^{(i)} = \langle 1, \zeta_i, \ldots, \zeta_i^{2k-1} \rangle$ where $\zeta_i \in_R \mathbb{Z}/q\mathbb{Z}$ with $1 \leq i \leq \ell$ is drawn independently and uniformly[11] at random for each $\gamma^{(i)}$. Here, a $\mathrm{GRS}_{\ell,2k}(\boldsymbol{\pi}, \boldsymbol{d})$ code is used, with $\boldsymbol{\pi} = (\zeta_1, \zeta_2, \ldots, \zeta_\ell)$ and $\boldsymbol{d} = (1, 1, \ldots, 1)$. The ith receiver has to protect the entire representation $\langle \theta_i \gamma_1^{(i)}, \ldots, \theta_i \gamma_{2k}^{(i)} \rangle$, and thus, to store at least θ_i and ζ_i in tamper-proof memory. Furthermore, the fast codeword generation method from §3.2 can no longer be used.

By applying the above codeword distribution method, an adversary who acquires $2k$ or more keys will be unable to derive any information about the tracing codewords that are used in the representations. She will only be capable of creating combinations of the representations. If there are fewer than $k+1$ keys in a combination, we are back to a standard tracing scenario. Otherwise, combinations of $k+1$ or more keys will be detected, but not traceable, since our tracing algorithm will be unable to factorize the error-locator polynomial nor the original approaches will reveal the traitors.

6 Conclusion

In this paper, we have presented new insights as well as several improvements to the Boneh-Franklin traitor tracing scheme [5]. First of all, we revisited the private key watermarking scheme based on Reed-Solomon codes; based on this, we describe a new non-black-box tracing algorithm whose complexity only depends on the square of the maximal coalition size k and is independent of the total number ℓ of users. Our new tracing algorithm does not require any change in the encryption scheme and can be used with any generalized Reed-Solomon codes.

This allows us to implement the scheme in a system with a virtually unlimited number of users; in other words, the maximal coalition size is only constrained by the channel bandwidth and the computational capacity of the receivers. This new tracing algorithm can also be applied with any other scheme relying on (generalized) Reed-Solomon codes to watermark the distributed private keys.

Additionally, we discussed the application of the Guruswami-Sudan list-decoding algorithm, whose use was proposed in [5], and showed that, in practice, it brings only a marginal improvement in tracing capabilities, and this at high cost.

Finally, we studied the above-threshold security of the Boneh-Franklin scheme, i.e., the malicious capabilities of an adversary having access to many more than k keys. We showed that, given a coalition size of k, an adversary who has recovered $2k$ private keys or more can derive *any* other private key, provided the code Γ is publicly known, as advocated in [5]. To the best of our knowledge,

[11] Note that for practical values of ℓ, a collision between two codewords has a negligible probability to occur.

this 'feature' has not been reported in the literature. To deal with this problem, we suggest to keep the tracing code matrix secret and to distribute *statistically independent* codewords to the receivers.

Even though the Boneh-Franklin scheme can encrypt only small messages (basically, one group element), and even though using it in a hybrid fashion by encrypting a symmetric session key is prone to a trivial untraceable strategy[12], we believe based on our results that, in order to fight illegitimate clones of tamper-proof modules, the Boneh-Franklin scheme is now really worth to be considered in scenarios where trivial untraceable strategies are unavoidable[13] by design. Of course, this statement is based on the assumption that it is possible to revoke a traced clone by some other mechanism.

Acknowledgments

We would like to thank Olivier Billet, Olivier Brique, Nicolas Fischer, Jim Fuller, Michael Hill, Corinne Le Buhan Jordan, André Nicoulin, Karl Osen and Martijn Stam for interesting discussions and comments about this paper.

References

1. R. Anderson. *Security engineering – a guide to building dependable distributed systems*. Wiley, 2001.
2. O. Billet and D. Phan. Efficient traitor tracing from collusion secure codes. In R. Safavi-Naini, editor, *Information Theoretic Security, Third International Conference, ICITS 2008, Calgary, Canada, August 10-13, 2008. Proceedings*, volume 5155 of *Lecture Notes in Computer Science*, pages 171–182. Springer-Verlag, 2008.
3. D. Bini and V. Pan. *Polynomial and matrix computations: fundamental algorithms*, volume 1 of *Progress in Theoretical Computer Science Series*. Birkhauser Verlag, 1994.
4. D. Boneh. The decision Diffie-Hellman problem. In J. Buhler, editor, *Algorithmic Number Theory, Third International Symposium (ANTS III), Portland, Oregon, USA, June 21 - 25, 1998. Proceedings*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer-Verlag, 1998.
5. D. Boneh and M. Franklin. An efficient public key traitor tracing scheme. In M. Wiener, editor, *Advances in Cryptology – Crypto'99: 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999. Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 338–353. Springer-Verlag, 1999.
6. D. Boneh and M. Naor. Traitor tracing with constant size ciphertext. Manuscript available on `http://crypto.stanford.edu/~dabo/papers/const-tt.pdf`, 2008.
7. D. Boneh, A. Sahai, and B. Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In S. Vaudenay, editor, *Advances in Cryptology – Eurocrypt 2006: 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006. Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 573–592. Springer-Verlag, 2006.
8. D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, 1998.
9. D. Boneh and B. Waters. A fully collusion resistant broadcast, trace and revoke system. In A. Juels, R. Wright, and S. De Capitani de Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communication Security, CCS 2006, Alexandria, USA, October 30 - November 3, 2006*, pages 211–220. ACM Press, 2006.
10. D. Cantor and H. Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36(154):587–592, April 1981.
11. R. Chien. Cyclic decoding procedures for Bose-Chaudhuri-Hocquenghem codes. *IEEE Transactions on Information Theory*, 10(4):357–363, October 1964.
12. B. Chor, A. Fiat, and M. Naor. Tracing traitors. In Y. Desmedt, editor, *Advances in Cryptology – Crypto'94: 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994. Proceedings*, volume 839 of *Lecture Notes in Computer Science*, pages 257–270. Springer-Verlag, 1994.

[12] This strategy is simply to share the session key.
[13] Like in Pay-TV systems using the DVB-CSA [39] standard encryption, for instance.

13. B. Chor, A. Fiat, M. Naor, and B. Pinkas. Tracing traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, 2000.
14. H. Cohen. *A course in computational algebraic number theory*. Springer-Verlag, 2000.
15. Y. Dodis, N. Fazio, A. Kiayias, and M. Yung. Scalable public-key tracing and revoking. In S. Rajsbaum, editor, *PODC 2003, Proceedings of the Twenty-Second ACM Symposium on Principles of Distributed Computing, July 13-16, 2003, Boston, USA*, pages 190–199. ACM Press, 2003.
16. Y. Dodis, N. Fazio, A. Kiayias, and M. Yung. Scalable public-key tracing and revoking. *Distributed Computing*, 17(4):323–347, 2005.
17. A. Fiat and M. Naor. Broadcast encryption. In D. Stinson, editor, *Advances in Cryptology – CRYPTO'93: 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993. Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer-Verlag, 1994.
18. V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. In *39th Annual Symposium on Foundations of Computer Science (FOCS'98), November 8-11, 1998, Palo Alto, California, USA*, pages 28–39. IEEE Computer Society, 1998.
19. V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999.
20. J. Hall. Notes on coding theory – Generalized Reed-Solomon codes. Available on `http://www.mth.msu.edu/~jhall/classes/codenotes/GRS.pdf`, 2003.
21. A. Kiayias and S. Pehlivanoglu. Pirate evolution: how to make most of your traitor keys. In A. Menezes, editor, *Advances in Cryptology – CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2007. Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, pages 448–465. Springer-Verlag, 2007.
22. A. Kiayias and M. Yung. Self protecting pirates and black-box traitor tracing. In J. Kilian, editor, *Advances in Cryptology – CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001. Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 63–79. Springer-Verlag, 2001.
23. A. Kiayias and M. Yung. Breaking and repairing asymmetric public-key traitor tracing. In J. Feigenbaum, editor, *Security and Privacy in Digital Rights Management, ACM CCS-9 Workshop, DRM 2002, Washington DC, USA, November 18, 2002. Revised Papers*, volume 3557 of *Lecture Notes in Computer Science*, pages 32–50. Springer-Verlag, 2003.
24. K. Kurosawa and Y. Desmedt. Optimum traitor tracing and asymmetric schemes with arbiter. In K. Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98: International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May/June 1998. Proceedings*, volume 1403 of *Lecture Notes in Computer Science*, pages 145–157. Springer-Verlag, 1998.
25. K. Kurosawa and T. Yoshida. Linear code implies public-key traitor tracing. In D. Naccache and P. Paillier, editors, *Public-Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC'02, Paris, France, February 12-14, 2002, Proceedings*, volume 2274 of *Lecture Notes in Computer Science*, pages 172–187. Springer-Verlag, 2002.
26. LiDIA A C++ Library for Computational Number Theory. Software available on `http://www.cdc.informatik.tu-darmstadt.de/TI/LiDIA/`.
27. S. Mangard, E. Oswald, and T. Popp. *Power analysis – revealing the secrets of smart cards*. Springer, 2007.
28. J. Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15(1):122–127, 1969.
29. J. McGregor, Y. Yin, and R. Lee. A traitor tracing scheme based on RSA for fast decryption. In J. Ioannidis, A. Keromytis, and M. Yung, editors, *Applied Cryptography and Network Security, Third International Conference, ACNS 2005, New-York, USA, June 7–10, 2005. Proceedings*, volume 3531 of *Lecture Notes in Computer Science*, pages 56–74. Springer-Verlag, 2005.
30. A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of applied cryptography*. The CRC Press series on discrete mathematics and its applications. CRC-Press, 1997.
31. GNU Multiple Precision Arithmetic Library. Software available on `http://gmplib.org`.
32. D. Naor, M. Naor, and J. Lotspiech. Revocation and tracing schemes for stateless receivers. In J. Kilian, editor, *Advances in Cryptology – CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001. Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer-Verlag, 2001.
33. M. Naor and B. Pinkas. Threshold traitor tracing. In H. Krawczyk, editor, *Advances in Cryptology – CRYPTO'98: 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1998. Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 502–517. Springer-Verlag, 1998.

34. V. Pan. Faster solution of the key equation for decoding BCH error-correcting codes. In F. Leighton and P. Shor, editors, *Proceedings, 29th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 168–175. ACM Press, 1997.
35. B. Pfitzmann. Trials of traced traitors. In R. Anderson, editor, *Information Hiding, First International Workshop, Cambridge, UK, May 30 - June 1, 1996. Proceedings*, volume 1174 of *Lecture Notes in Computer Science*, pages 49–64. Springer-Verlag, 1996.
36. I. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, 8(2):300–304, 1960.
37. A. Silverberg, J. Staddon, and J. Walker. Efficient traitor tracing algorithms using list decoding. In C. Boyd, editor, *Advances in Cryptology – Asiacrypt 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001. Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pages 175–192. Springer-Verlag, 2001.
38. A. Silverberg, J. Staddon, and J. Walker. Applications of list decoding to traitor tracing. *IEEE Transactions on Information Theory*, 49(5):1312–1318, 2003.
39. Digital Video Broadcasting (DVB) Conditional Access Standards. Available on `http://www.dvb.org/technology/standards/index.xml#conditional`.
40. D. Stinson and R. Wei. Key preassigned traceability schemes for broadcast encryption. In S. Tavares and H. Meijers, editors, *Selected Areas in Cryptography: 5th Annual International Workshop, SAC'98, Kingston, Ontario, Canada, August 1998. Proceedings*, volume 1556 of *Lecture Notes in Computer Science*, pages 144–156. Springer-Verlag, 1999.
41. D. Wallner, E. Harder, and R. Agee. Key management for multicast: issues and architectures. RFC 2627, 1999. Available on `http://www.ietf.org`.
42. L. Welch and E. Berlekamp. Error correction for algebraic block codes. US Patent 4'633'470, 1986.
43. S. Wicker. *Error control systems for digital communications and storage*. Prentice Hall, 1995.
44. C. Wong, M. Gouda, and S. Lam. Secure group communications using key graphs. In *Proceedings of the ACM SIGCOMM'98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, August 31 - September 4, 1998, Vancouver, British Columbia, Canada*, pages 68–79. ACM Press, 1998.