Improved Efficiency for Revocation Schemes via Newton Interpolation

Noam Kogan* and Tamir Tassa†

December 6, 2006

Abstract

We present a novel way to implement the secret sharing based family of revocation schemes of Naor and Pinkas [23]. The basic scheme of [23] uses Shamir's polynomial secret sharing to revoke up to r users, where r is the degree of the secret sharing polynomial, and it is information theoretically secure against coalitions of up to r collaborators. The non-revoked users use Lagrange interpolation in order to compute the new key. Our basic scheme uses a novel modification of Shamir's polynomial secret sharing: The secret equals the leading coefficient of the polynomial (as opposed to the free coefficient as in the original scheme) and the polynomial is reconstructed by Newton interpolation (rather than Lagrange interpolation). Comparing our scheme to one variant of the Naor-Pinkas scheme, we offer revocation messages that are shorter by a factor of almost 2, while the computation cost at the user end is smaller by a constant factor of approximately $\frac{13}{2}$. Comparing to a second variant of the Naor-Pinkas scheme, our scheme offers a reduction of $O(r)$ in the computation cost at the user end, without affecting any of the other performance parameters. We then extend our basic scheme to perform multi-round revocation for stateless and stateful receivers, along the lines offered by Naor and Pinkas [23] and Kogan et al. [19]. We show that using Newton rather than Lagrange interpolants enables a significantly more efficient transmission of the new revocation message and shorter response time for each round. Pay TV systems that implement broadcast encryption techniques can benefit significantly from the improved efficiency offered by our revocation schemes.

Keywords: User revocation, broadcast encryption, secret sharing, Newton interpolation.

*Dept. of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel. Supported in part by a grant from Giesecke and Devrient, GmbH.

†Department of Mathematics and Computer Science, The Open University, Ramat Aviv, Tel Aviv, Israel. Email: tamir_tassa@yahoo.com.

1 Introduction

One of the main security problems in networks that distribute digital content is piracy. Piracy is exercised by authorized users who copy the content that they received and redistribute it to illegal users. Such activity incurs great losses to the producers and distributors of the digital content. This problem affects all forms of distribution of digital content such as software, music CDs, DVDs, satellite and cable TV.

Satellite and cable TV systems are often based on a uni-directional broadcast distribution channel. Access to the content is regulated by encryption through conditional access systems. The content is encrypted and the relevant encryption keys are given to paying customers only. Those customers are given a set-top box (STB) that is coupled with a smartcard (SC). The SC, which is a tamper-resistant device, holds the key material and is responsible for the decryption of all entitlement messages (the encrypted messages that contain the necessary decryption keys). The STB, on the other hand, performs the actual decryption of the encrypted content.

One of the characteristics of digital Pay TV systems is the large choice of broadcast content. Users may choose between different packages of channels or special Pay-Per-View events. This introduces the problem of *broadcast encryption*: how may the content provider communicate securely with a dynamically changing subset of the users, over the insecure broadcast channel, while minimizing the key management overhead. The transmission overhead required by the key management is an important parameter. One reason for that is the large size of systems (millions of users). In addition, because STBs may be disconnected for a while, the key material must be transmitted repeatedly and in a high rate in order to allow a reasonable acquisition time upon reconnection.

In this paper we are dealing with the closely related subject of *revocation schemes*. Revocation schemes are concerned with permanent revocation of users from all access rights. In such settings, the set of permanently revoked users is small and slowly growing. Permanent revocation may be required in order to disconnect users who leaked their decryption keys to pirates in order to construct illegal decoders. Revoking such users would also render the pirate decoders useless. In fact, implementing revocation schemes might deter users from engaging in such illegal activities in the first place. Another example where permanent revocation may be a preferred mode of action is the discontinuation of service for users who stopped their subscription but failed to return their SC. In both examples, the number of users that need to be permanently revoked is expected to be significantly smaller than the number of users in the system, and it is expected to grow slowly.

Such revocation may be obtained by implementing broadcast encryption schemes. However, by focusing on permanent revocation only, where the number of users to revoke, r, is typically significantly smaller than the number of users in the system, n, one may obtain more efficient solutions. In such solutions, all meaningful cost measures are independent of n and depend only on r.

In practice, TV operators are less concerned about the average typically-honest user; instead, they are more concerned with malicious users who exercise piracy that might translate into painful financial losses. Protection against honest (and even curious) users may be achieved by non-cryptographic and cheap means. For example, in most conditional access systems nowadays, all users get the keys that enable the decryption of all content; it is the SC that decides which of those keys should be given to the STB according to the user's access rights. Such protection is most satisfactory against the honest-but-curious users, and it offers a practical alternative for broadcast encryption schemes. Protection against malicious users, on the other hand, justifies the costs that are incurred by the implementation of cryptographic means in the form of revocation schemes.

1.1 The setting

Let $\mathcal{U} = \{u_1, \ldots, u_n\}$ be the set of all users. A *Group Controller* GC is responsible for controlling the decryption capabilities of those users, by providing the necessary key material to all legitimate members of that group. The GC initializes each user's SC with secret data that enable the recovery of future cryptographic messages. Typically, conditional access systems make use of a periodic key that encrypts the entitlement messages through which users receive the short term keys that encrypt the content.

At some point the GC learns that some users need to be revoked. Consequently, the GC wishes to permanently revoke the decryption capabilities of these users by changing the periodic key. To do so, it randomly generates a new periodic key; then, in order to communicate this new key to all non-revoked users, it broadcasts a specially tailored message (*the revocation message* henceforth) that enables all non-revoked users to recover that key, while disabling the revoked users from doing so. An essential requirement from such a solution is resiliency: even if all revoked users collude and pool together their secret data, they should not be able to recover the new key.

The important factors that determine the efficiency of a given revocation scheme are:

◇ The communication overhead, i.e., the length of the revocation message sent by the GC.

◇ Storage overhead at the user end, i.e., the amount of secret data that needs to be stored in the EEPROM of the user's SC.

◇ The computational overhead, especially that which is imposed on the users, whenever the key is renewed.

Revocation can be trivially achieved as follows: whenever revocation is required, the GC generates a new periodic key and transmits it via unicast messages to each of the $(n - r)$ non-revoked users, encrypted with the personal key of the addressee. This solution, however, has two drawbacks. First, it is highly inefficient since its communication overhead is $O(n)$, even if only one user is to be revoked. As a consequence of this significant toll on bandwidth, and since usually only a small portion of bandwidth is allocated for control messages, such update cycles span along a considerable amount of time. Since it is necessary to repeat these update cycles many times before switching to the new key in order to ensure that all users received their personal message, the response time of this solution may be unacceptable. Hence, revocation techniques are called upon in order to achieve the same goal with a substantially smaller cost in bandwidth and faster response time.

The revocation solution should allow many revocation rounds since pirate revocation is an on-going process and, consequently, the subset of users to revoke, $\mathcal{R} = \{v_1, \ldots, v_r\} \subset \mathcal{U}$, is monotonically increasing. More specifically, since we are concerned with revoking *pirate* users, as opposed to revoking users who did not purchase some service or are late in paying their fees, revocation should be *permanent*. Consequently, as the GC performs multi-round revocation, the set of revoked users grows: the subset of revoked users in one revocation round includes all revoked users in the previous round as well as the newly detected pirate users. When discussing revocation and transition between revocation rounds one needs to consider two scenarios: stateless receivers and stateful receivers. In the first scenario, the personal information on the SCs is not updated.

In the second scenario there exists a low rate unicast channel through which the SCs may be updated with new personal information.

1.2 Related work

The subject of broadcast encryption was introduced in [3, 14]. Fiat and Naor [14] presented broadcast encryption schemes in which each user has a fixed reusable set of keys. They devised schemes that are secure against coalitions of a bounded size k. More specifically, given a resilience parameter k, they devised broadcast encryption schemes such that any subset of k (or less) revoked users could not compute the new key. The performance of their schemes is strongly dependent on k. This is disadvantageous for pay-TV environments where pirates can easily get hold of many SCs in order to beat the resilience threshold.

Broadcast encryption schemes usually fall into one of three categories, according to the mathematical tools on which they are based: combinatorial, secret sharing, and tree-based constructions.

Combinatorial schemes were proposed in [1, 15, 20, 21]. Luby and Staddon [21] studied the inherent trade-off between the transmission length and the storage overhead in the SC. More specifically, for a given bound on the number of keys that every SC has to store, they obtain an essentially tight lower bound on the length of the transmission that is required in case of a key change.

Kumar et al. [20] proposed constructions that enable one time revocation of up to t users that are secure against coalitions of all revoked users. Their constructions are based on cover-free sets with revocation messages of length $O(t \log n)$, in one method, and $O(t^2)$ in another.

Garay et al. [15] offer revocation schemes based on carefully planned random methods. Their schemes issue keys to users from a pool of keys that serves as a cover-free family with a resilience parameter k. Upon revocation, the GC discards the keys of compromised SCs and uses a threshold protocol to broadcast the revocation message. They consider the case when the group of revoked users is slowly expanding, in which case a multi-round pirate revocation is necessary, and obtain an upper bound on the total number of required SC state-modifications per round.

Anzai et al. [2] and, independently, Naor and Pinkas [23] devised revocation schemes that are based on the secret sharing techniques of Shamir [27]. The basic scheme of Naor and Pinkas can revoke up to t users, where t is the degree of the secret sharing polynomial. Their scheme is information theoretically secure against the coalition of all revoked users. Apart from the basic one-time revocation scheme, they considered the problem of multi-round revocation. One suggested solution was to use a sequence of revocation schemes, of growing revocation capabilities. Another solution was to lift Shamir's secret sharing scheme to the exponent and rely upon the Decisional Diffie-Hellman problem, an idea first offered by [13]. We elaborate on the Naor-Pinkas scheme, NP, later on as it serves as our reference point. Kogan et al. [19] proposed an efficient implementation of the stateless version of the NP scheme. They observed that the workload at the user end may be split between the SC and the STB in a way that off loads all the non-secret computations (the cost of which is $O(t)$) to the STB, and leaves for the SC only $O(1)$ computations. Such a split is essential in order to enable secret sharing of large sizes as it circumvents the inherent SC limitations in terms of RAM, CPU and communication rate.

Tzeng and Tzeng [29] proposed a public-key traitor tracing scheme with revocation capability using dynamic shares. Their techniques are similar to the ones presented in [23, Section 2.2] and are semantically secure assuming DDH (the decisional Diffie-Hellman assumption).

Dodis and Fazio [10] constructed the first trace-and-revoke scheme that is secure against chosen ciphertext attacks, assuming DDH. Their scheme is also the first adaptively secure scheme, as it allows the adversary to corrupt players at any point during execution. Kim, Hwang and Lee [18] improved that scheme by reducing the length of the enabling block by approximately one half and reducing the computational overhead of the user at the expense of increasing the computational overhead of the server (this is an improvement since the computing power of the user is usually weaker than that of the server).

Tree-based revocation schemes were proposed in [5, 6, 17, 22, 25, 30, 31] for both the stateless and stateful receiver scenarios. Those schemes are considered to boast the best combination of properties. In all of those schemes, the number of keys that need to be stored in the SC is poly-logarithmic in n.

Naor et al. [22] designed the so-called *Subset Difference* (SD) scheme that enables the GC to revoke any subset of users, $\mathcal{R} \subset \mathcal{U}$. In order to do that, they place the users in the leaves of a complete binary tree and then use the tree representation in order to define a base family of subsets of $\mathcal{U}$ with the property that for any $\mathcal{R} \subset \mathcal{U}$, its complement $\mathcal{U} \setminus \mathcal{R}$ may be covered by a disjoint union of $O(r)$ base subsets, where $r = |\mathcal{R}|$. The content is made accessible to all non-revoked users by sending an encrypted message containing the new key to each of the base subsets in the cover of $\mathcal{U} \setminus \mathcal{R}$. In order to decrypt at least one of those messages, each user has to store $\frac{1}{2} \cdot \log^2 n$ keys and perform $O(\log n)$ computations.

Halevi and Shamir [17] introduced the *Layered Subset Difference* (LSD) technique which offers an improvement of the Subset Difference scheme of Naor et al. They observed that the base family of subsets that "spans" all of $2^{\mathcal{U}}$ may be reduced, at the expense of a factor of 2 in the number of base subsets that are required in order to cover a given subset. As a result, their scheme requires that each user holds only $\log^{\frac{3}{2}} n$ keys, while raising the revocation message length by a factor of two.

Goodrich et al. [16] studied the problem of broadcasting confidential information to a group of n users while providing the ability to revoke an arbitrary subset of those (possibly colluding) users. They present Stratified Subset Difference methods that require each user to store only $O(\log n)$ keys. Their methods are suitable only for the case of stateful receivers.

The SD and LSD methods of [22] and [17] were extended to the public key setting, for stateless receivers, in [9]. Another recent work that deals with trace and revoke schemes in the public key setting is [11]; Dodis et al. presented there a public key revocation scheme that applies to the stateful receiver case while obviating the need for unicast channels. Theirs is a hybrid approach between the stateless and stateful scenarios: receivers do not maintain state between various system operations, but the system does change phases or periods across which the receiver has to maintain state.

1.3 Definitions and notations

Throughout this paper $\mathbb{F}$ is a fixed finite field of a large size. Typically $\mathbb{F} = \mathbb{F}_p$ is the Galois field of size p, where p is a large prime. We let ℓ denote the number of bits in the binary representation of p, $\ell = \lfloor \log p \rfloor + 1$. Each user is assigned an *identity* that is a field element. To avoid cumbersome notations, we do not distinguish between the user u and his identity. Namely, $\mathcal{U}$ is simply a subset of $\mathbb{F}$. For practical purposes, those identities are taken to be usually small (typically, $\lceil \log n \rceil$ bits suffice). We let s denote the number of bits in the binary representation of user identities. A practical choice for s would be $s = 32$. As for ℓ, since it serves as a security parameter, it should be chosen in the vicinity of 100 in order to provide reasonable security (e.g., $\ell = 128$ if the shared secret is an AES key).

We estimate the efficiency of our proposed scheme and compare it to other schemes in terms of temporal and spatial costs. Temporal costs are measured in terms of arithmetic operations: $[A]$ denotes hereinafter addition or subtraction, $[M]$ denotes multiplication and $[D]$ stands for division. The cost of addition in $\mathbb{F}$ is $O(\ell)$, while that of multiplication or division is $O(\ell^2)$. However, division is significantly more time consuming than multiplication. The ratio between the cost of $[D]$ and that of $[M]$ depends on the field $\mathbb{F}$ and on the implementation; but for p larger than 100 bits it may be estimated that

$$10 \leq \theta := \frac{\mathrm{Cost}[D]}{\mathrm{Cost}[M]} \leq 31 \ , \tag{1}$$

see [7, §3.2]. Spatial costs are measured in terms of ℓ and s – sizes of elements in $\mathbb{F}$ (*long*) and $\mathcal{U}$ (*short*), respectively. The cost parameters are:

- $\mathtt{C}_{\mathrm{GC}}$ – Computation at the GC upon activating a revocation.
- $\mathtt{C}_{\mathrm{user}}$ – Computation that legitimate users need to carry out in order to recover the shared secret.
- $\mathtt{S}$ – Amount of personal data that every user needs to store.
- $\mathtt{L}$ – The revocation message length, namely, the amount of data that needs to be broadcast by the GC so that all users in $\mathcal{U} \setminus \mathcal{R}$ could recover the shared secret while revoked users cannot.

1.4 Our contribution

We present a novel and more efficient way to implement the secret sharing based family of revocation schemes of Naor and Pinkas [23].

The basic scheme of [23] uses Shamir's polynomial secret sharing to remove up to t users, where t is the degree of the secret sharing polynomial, and it is information theoretically secure against coalitions of up to t collaborators. The GC selects a future secret key and shares it among all of the current users using Shamir's secret sharing scheme. When it is necessary to revoke t of the users, the GC transmits a revocation message that contains the identities and shares of those revoked users. Each of the non-revoked users uses the published shares of the revoked users, together with his own share, in order to compute the new key (which is the free coefficient of the polynomial) by means of Lagrange interpolation. For efficiency, the GC pre-computes a set of t values and adds them to the revocation message. This reduces the computational cost at the user end from quadratic to roughly $3t \times [M] + t \times [D]$.

Our basic scheme (Section 3) uses also Shamir's polynomial secret sharing, but it is based on Newton interpolation. The common secret in our suggested scheme is the coefficient of x^t, where t is the degree of the polynomial. In order to revoke t users, the GC expresses the polynomial in its Newton form with respect to the set of t revoked users and transmits a revocation message that includes the t identities of the revoked users and the t lower coefficients in the Newton form. This is shorter by a factor of almost 2 with respect to the NP revocation message. Each non-revoked user can combine his share with the information conveyed by the revocation message and perform Newton interpolation to compute the new key. This requires each user to perform a computation of approximate cost $2t \times [M] + 1 \times [D]$. This offers a significant reduction of computational cost at the user end without affecting any of the other performance parameters.

In Section 4 we throw in the time dimension. Since revocation is an on-going process, the GC must update the blacklist of pirate users that need to be revoked whenever new pirate users are detected. Since such users need to be revoked permanently, the blacklist – the subset of revoked users – is slowly growing from one revocation round to the next one. Here, we consider the computational effort and the revocation message length that a transition between two successive rounds entails.

In the case of stateful receivers, each revocation round is independent of the previous rounds and, hence, multi-round revocation may be viewed as a sequence of one-time revocations. In the stateless receiver scenario, on the other hand, we may take advantage of the fact that the set of revoked users is slowly growing in order to bring up another advantage of Newton over Lagrange interpolation. As in Kogan et al. [19], we use m polynomials of linearly increasing degrees with a dilation factor d. Using Lagrange interpolation, the GC cannot compute (and hence cannot send) the t values that enable the quadratic-to-linear reduction of computational cost at the user end before the identities of *all* revoked users are known. In Newton interpolation, on the other hand, it is possible to send a preliminary revocation message that contains the identities of the t users that are already revoked and the t lower coefficients that they "induce" in the Newton form of the polynomial to be used next. When the GC learns the identities of the (up to d) new users to revoke, it may send a short complementary revocation message, containing only the identities of these users and the corresponding Newton coefficients. This translates into two advantages: (a) The complementary revocation message with Newton interpolants is much shorter than the revocation message with Lagrange interpolants. (b) The users may perform a substantial part of the computation already upon the receipt of the preliminary revocation message; this leaves them a much shorter computation to be carried out upon the reception of the complementary revocation message in order to recover the new secret. Therefore, using Newton interpolants enables a shorter response time (the time that elapses from the disclosure of the identities of the new users to be revoked until they are actually revoked) in each round.

In Section 5 we consider a different solution for the multi-round revocation problem that is based on the Decisional Diffie-Hellman assumption.

Pay TV systems that implement broadcast encryption techniques can benefit significantly from the improved efficiency offered by our revocation schemes. In addition, our schemes may also be used for fast re-keying and/or revocation in multicast environments and tactical wireless networks, where savings in communication and computational overhead can also be of great significance.

2 Preliminaries

2.1 A brief overview of univariate interpolation

The problem of interpolating a univariate function $f : \mathbb{F} \to \mathbb{F}$ by a polynomial from a set of sample points is a basic problem in approximation theory and numerical analysis. It is basically a linear algebra problem and, like many other problems in linear algebra, the choice of a basis is of paramount importance.

Assume that the values of $f(x)$ are given in $\{x_i\}_{0 \le i \le n}$. We look for a polynomial $P(x) = P_n(x)$ in the subspace $\mathbb{F}_n[x] = \{P(x) \in \mathbb{F}[x] : \deg P \le n\}$ such that $P(x_i) = f(x_i)$, $0 \le i \le n$. In order to solve this problem, we first set a basis for $\mathbb{F}_n[x]$, $\{B_k(x)\}_{0 \le k \le n}$. Then, we represent $P(x)$ with respect to that basis, $P(x) = \sum_{k=0}^{n} a_k B_k(x)$, and write down the set of linear equations that the coefficients satisfy,

$$\sum_{k=0}^{n} a_k B_k(x_i) = f(x_i) \quad , \quad 0 \le i \le n \ . \tag{2}$$

When $B_k(x) = x^k$ (the standard basis), $(B_k(x_i))_{0\le i,k\le n}$ is the Vandermonde matrix. This is a full matrix that needs to be solved. (In addition, when $\mathbb{F} = \mathbb{R}$, that matrix tends to be ill-conditioned in the sense that it is very sensible to rounding errors.) Hence, this turns out to be a bad choice. A better choice is the Lagrange polynomials,

$$B_k(x) = L_k(x) = \prod_{j\neq k} \left(\frac{x - x_j}{x_k - x_j} \right) \quad , \quad 0 \le k \le n \ . \tag{3}$$

Here, the coefficient matrix is the identity – the best that one could hope for, and, consequently, $a_k = f(x_k)$, $0 \le k \le n$. However, this form of the interpolant suffers from two computational disadvantages:

- *Inefficiency.* The evaluation of the polynomial at a point requires a great number of arithmetic operations.
- *Unscalability.* Assume that we have already computed the value of the interpolant at some point $\hat{x}$, $P(\hat{x}) = P_n(\hat{x})$, and then we are given the function value at a new sample point x_{n+1}. Then, in order to compute $P_{n+1}(\hat{x})$ – the value of the higher order interpolant on $\{x_i\}_{0\le i\le n+1}$, we can make no use of the value $P_n(\hat{x})$ and an entire new computation is required.

These problems are solved with the Newton polynomials,

$$B_k(x) = N_k(x) = \prod_{j=0}^{k-1} (x - x_j) \quad , \quad 0 \le k \le n \ . \tag{4}$$

This choice of a basis results in a triangular system which is easily solvable without Gaussian elimination. Here, $a_k = f[x_0, \ldots, x_k]$ – the kth order Newton divided difference of f on $\{x_i\}_{0\le i\le n}$. Those divided differences are defined recursively by $f[x_0] = f(x_0)$ and $f[x_0, \ldots, x_k] = (f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}])/(x_k - x_0)$ for $k \ge 1$. Using this form, the evaluation of $P(x)$ is efficient. Moreover, the unscalability of the Lagrange interpolants is solved since $P_{n+1}(x) = P_n(x) + f[x_0, \ldots, x_{n+1}] \cdot N_{n+1}(x)$.

In the following sections we show how these computational advantages of Newton interpolation translate into more efficient revocation schemes.

2.2 Threshold secret sharing

The basic problem of secret sharing is that in which one aims at sharing a secret among the participants of a set of size n so that any k of them can deduce the value of the secret from their shares, while any subset of less than k participants can learn nothing about the value of the secret. The scheme that was proposed in the seminal work of Shamir [27] uses polynomial interpolation in order to achieve that goal. To that end, let $\mathcal{U}$ be the set of n participants and $\mathcal{S}$ be the domain of secrets. Let $\mathbb{F}$ be a finite field whose size allows to embed $\mathcal{S}$ in $\mathbb{F}$ and $\mathcal{U}$ in $\mathbb{F}^*$. Then the dealer selects a random polynomial of degree $k-1$ over $\mathbb{F}$, $P(x)=\sum_{i=0}^{k-1} a_i x^i$, where a_0 is the secret, and then every participant $x_j \in \mathcal{U} \subseteq \mathbb{F}^*$ gets the share $P(x_j)$, $1 \le j \le n$. Clearly, every k participants may recover all of the coefficients of $P(x)$ and, consequently, the secret $S=a_0$, while any smaller number of participants can not learn anything about the secret. (Secret sharing schemes that satisfy the latter property are called *perfect*, e.g. [28, Definition 13.2].)

We introduce herein a modification of this scheme, where the secret is the leading coefficient a_{k-1}. Namely, the dealer selects a random polynomial of degree $k-1$ over $\mathbb{F}$, $P(x)=\sum_{i=0}^{k-1} a_i x^i$, where a_{k-1} is the secret, and, as before, every participant $x_j \in \mathcal{U} \subseteq \mathbb{F}^*$ gets the share $P(x_j)$, $1 \le j \le n$.

Theorem 2.1 *The modified Shamir's secret sharing scheme where the secret equals the leading coefficient of the polynomial is a perfect scheme.*

Proof. Let $\mathcal{V}=\{x_j\}_{1\le j\le k-1} \subset \mathcal{U}$ be an arbitrary maximal unauthorized subset. We aim at showing that the information that the members of $\mathcal{V}$ hold does not reveal any information about the secret a_{k-1}. To show that, we need to prove that the $k-1$ linear equalities in the unknowns $(a_0,\ldots,a_{k-1})$ that are implied by the shares possessed by the members of $\mathcal{V}$ do not pose any restriction on the value of a_{k-1}. This is equivalent to the statement that the vector $(0,\ldots,0,1) \in \mathbb{F}^k$ is not spanned by the rows of the matrix

$$\begin{pmatrix} 1 & x_1 & \cdots & x_1^{k-2} & x_1^{k-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{k-1} & \cdots & x_{k-1}^{k-2} & x_{k-1}^{k-1} \end{pmatrix} .$$

This is indeed the case since

$$\det\begin{pmatrix} 1 & x_1 & \cdots & x_1^{k-2} & x_1^{k-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{k-1} & \cdots & x_{k-1}^{k-2} & x_{k-1}^{k-1} \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} = \det\begin{pmatrix} 1 & x_1 & \cdots & x_1^{k-2} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{k-1} & \cdots & x_{k-1}^{k-2} \end{pmatrix} = \prod_{1\le i<j\le k-1} (x_j - x_i) \neq 0 .$$

This completes the proof. $\Box$

We note in passing that a_0 and a_{k-1} are the only coefficients that could be safely used as the secrets. The intermediate coefficients a_i, $0<i<k-1$, are a risky choice since a careless embedding of $\mathcal{U}$ in $\mathbb{F}^*$ might mar this essential property of perfect security. For example, assume that $k=3$, $P(x)=a_0+a_1x+a_2x^2$, and that a_1 is selected as the secret. Then, if there exist two participants x_i and x_j such that $x_i=-x_j$, they may recover a_1 since

$$\frac{P(x_i)-P(x_j)}{2x_i} = \frac{(a_0+a_1x_i+a_2x_i^2)-(a_0+a_1x_j+a_2x_j^2)}{2x_i} = a_1 .$$

3 The basic one-time scheme

Here we concentrate on the case where $\mathcal{R}=\{v_1,\ldots,v_r\} \subset \mathcal{U}$ is fixed.

3.1 The Procedure

3.1.1 Initialization

Given a secret S, the GC generates a random polynomial of degree r over $\mathbb{F}$,

$$P(x) = \sum_{k=0}^{r} b_k x^k \quad , \quad b_k \in \mathbb{F} \ , \tag{5}$$

where $b_r = S$ is the secret. This polynomial is referred to as *the secret generating polynomial.* The GC proceeds to issue a personal share $P(u) \in \mathbb{F}$ for each user $u \in \mathcal{U}$.

3.1.2 Revocation

The GC expresses the polynomial $P(x)$ as the Newton interpolation polynomial with respect to $\mathcal{R}$ as the base set of points; namely,

$$P(x) = \sum_{k=0}^{r} b_k x^k = \sum_{k=0}^{r} a_k \prod_{j=1}^{k} (x - v_j) \ . \tag{6}$$

In order to allow all non-revoked users in $\mathcal{U} \setminus \mathcal{R}$ to compute the secret $a_r = b_r$, the GC broadcasts a revocation message that consists of the identities of the r revoked users and the first r coefficients in the Newton representation of $P(x)$ with respect to $\mathcal{R}$,

$$\{v_j : 1 \leq j \leq r\} \ ; \ \{a_k : 0 \leq k \leq r-1\} \ . \tag{7}$$

The formulas for computing these coefficients are

$$a_k = \frac{P(v_{k+1}) - \sum_{i=0}^{k-1} a_i \prod_{j=1}^{i} (v_{k+1} - v_j)}{\prod_{j=1}^{k} (v_{k+1} - v_j)} \quad , \quad 0 \leq k \leq r-1 \ . \tag{8}$$

3.1.3 Acquisition at the user end

The non-revoked users $u \in \mathcal{U} \setminus \mathcal{R}$ may compute a_r using their personal share $P(u)$ and the data in the revocation message (7),

$$a_r = \frac{P(u) - \sum_{i=0}^{r-1} a_i \prod_{j=1}^{i} (u - v_j)}{\prod_{j=1}^{r} (u - v_j)} \ . \tag{9}$$

Equation (9) agrees with (8) with $k = r$ and $v_{r+1} = u$. A revoked user could not compute this coefficient because the denominator vanishes when $u = v_j$ for some $1 \leq j \leq r$. Moreover, even if all revoked users collude, they can not learn anything about the value of a_r, as implied by Theorem 2.1.

3.2 Cost analysis

⋄ **Computation at the GC.** The GC needs to compute the coefficients a_k, $0 \le k \le r-1$. As stated in Proposition 3.1, the cost of this computation is,

$$\mathtt{C}_{\mathrm{GC}} = r(r-1) \times [A] + (r-1)(r-2) \times [M] + (r-1) \times [D] \ . \tag{10}$$

⋄ **Computation at the user end.** The cost of this operation, as stated in Proposition 3.2, is

$$\mathtt{C}_{\mathrm{user}} = 2r \times [A] + 2(r-1) \times [M] + [D] \ . \tag{11}$$

⋄ **Storage at user end.** The personal identity and share, i.e., $\mathtt{S} = s + \ell$.

⋄ **Revocation message length.** The length of the revocation message (7) is r short field elements and r full-length field elements, i.e., $\mathtt{L} = rs + r\ell$.

Proposition 3.1 *The cost of computing the coefficients a_k, $0 \le k \le r-1$, is*

$$C[a_0, \ldots, a_{r-1}] = r(r-1) \times [A] + (r-1)(r-2) \times [M] + (r-1) \times [D] \ .$$

Proof. We begin with the first two terms in the recursion (8). The first coefficient $a_0 = P(v_1)$ requires no computation, $C[a_0] = 0$. The cost of evaluating $a_1 = (P(v_2) - a_0)/(v_2 - v_1)$ is $C[a_1] = 2[A] + [D]$. In general, in order to compute a_k based on the previous coefficients, we have to do the following operations:

1. Compute and store all products $\prod_{j=1}^{i}(v_{k+1} - v_j)$, $1 \le i \le k$; this requires k additions and $k-1$ multiplications.

2. Evaluating the sum in the numerator in (8); this requires $k-1$ additions and $k-1$ multiplications.

3. One final addition to complete the computation of the numerator plus one final division.

Altogether, we conclude that

$$C[a_k] = 2k \times [A] + 2(k-1) \times [M] + [D] \ . \tag{12}$$

Adding up those costs, we get

$$C[a_0, \ldots, a_{r-1}] = \sum_{k=1}^{r-1} C[a_k] = r(r-1) \times [A] + (r-1)(r-2) \times [M] + (r-1) \times [D] \ .$$

□

Setting $k = r$ in (12) we arrive at the following.

Proposition 3.2 *A user $u \in \mathcal{U} \setminus \mathcal{R}$ can compute a_r in cost $2r \times [A] + 2(r-1) \times [M] + [D]$.*

3.3 Comparison to the NP scheme

3.3.1 Review of the basic NP scheme

In the NP scheme, the secret is $P(0) = b_0$ and not b_r as it is in our scheme. The NP scheme makes use of the Lagrange interpolation formula: if $w_j \in \mathbb{F}$, $0 \le j \le r$, are $r+1$ field elements and $P(x)$ is a polynomial in $\mathbb{F}_r[x]$, (5), then

$$P(0) = \sum_{j=0}^{r} c_j P(w_j) \quad \text{where} \quad c_j = \prod_{0 \le i \le r, i \ne j} \frac{w_i}{w_i - w_j} \ . \tag{13}$$

Each user $u \in \mathcal{U} \setminus \mathcal{R}$ needs to use this identity with $w_0 = u$ and $w_j = v_j$ for $1 \le j \le r$ in order to recover the secret $P(0)$. Of course, revoked users cannot compute $P(0)$ because they lack one share.

As most of those calculations are common to all users, it is suggested in [23] that the GC precomputes the following set of common values:

$$c'_j = \prod_{1 \le i \le r, i \ne j} \frac{v_i}{v_i - v_j} \quad , \quad 1 \le j \le r \ , \tag{14}$$

and then the revocation message is

$$\{v_j : 1 \le j \le r\} \ ; \ \{P(v_j) : 1 \le j \le r\} \ ; \ \{c'_j : 1 \le j \le r\} \ . \tag{15}$$

Upon receiving the revocation message, user u proceeds to compute the coefficients c_j in (13) as follows:

$$c_0 = \prod_{1 \le i \le r} \frac{v_i}{v_i - u} \quad \text{and} \quad c_j = c'_j \cdot \frac{u}{u - v_j} \quad , \quad 1 \le j \le r \ . \tag{16}$$

Then, u may complete the computation of the secret $P(0)$ according to (13). Of course, every user will compute a different set of coefficients c_j but will arrive at the end at the same value of $P(0)$.

3.3.2 Cost analysis.

$\diamond$ **Computation at the GC.** For a more efficient computation of c'_j, we rewrite (14) in the following manner:

$$c'_j = \frac{z}{v_j \cdot \prod_{1 \le i \le r, i \ne j}(v_i - v_j)} \quad \text{where} \quad z = \prod_{1 \le i \le r} v_i \quad , \quad 1 \le j \le r \ . \tag{17}$$

Hence, the GC needs to start with $r(r-1)$ subtractions (to compute $v_i - v_j$ for all $1 \le i \ne j \le r$), followed by $r-1$ multiplications in order to compute z. Then, for each c'_j the GC needs to perform $r-1$ multiplications, in order to compute the denominator in (17), followed by one division. In total, we conclude that

$$\mathsf{C}_{\text{GC}} = r(r-1) \times [A] + (r^2 - 1) \times [M] + r \times [D] \tag{18}$$

Comparing this cost to the equivalent cost in our method, (10), we see that (10) is slightly more efficient but the improvement is insignificant.

$\diamond$ **Computation at the user end.** A user $u \in \mathcal{U}$ needs to compute $P(0) = c_0 P(u) + \sum_{j=1}^{r} c_j P(v_j)$ where c_j, $0 \le j \le r$, are given in (16). To evaluate c_0, the user performs r subtractions (for $u - v_j$), $2(r-1)$ multiplications (in order to evaluate the numerator and denominator) and, finally, 1 division. To complete the evaluation of $\sum_{j=1}^{r} c_j P(v_j)$, using (16), we observe that u may be extracted from all addends in order to save multiplications. Since the expressions $u - v_j$ were already computed for c_0, we are left with r divisions ($\frac{c'_j}{u-v_j}$, $1 \le j \le r$), r multiplications of those values by $P(v_j)$, followed by $r-1$ additions and 1 multiplication (the external multiplication by u). To wrap up the computation we perform one more multiplication (for $c_0 P(u)$) and one more addition (to add $c_0 P(u)$ to $\sum_{j=1}^{r} c_j P(v_j)$). Altogether, we get

$$\mathsf{C}_{\text{user}} = 2r \times [A] + 3r \times [M] + (r+1) \times [D] \ . \tag{19}$$

Comparing this to (11), we note that the number of multiplications is reduced by a third while the number of divisions is dramatically reduced by a factor of r. Recalling (1), and ignoring the cost of additions, the overall improvement factor is

$$\frac{\mathsf{C}_{\text{user}}[\mathsf{NP}]}{\mathsf{C}_{\text{user}}[\mathsf{KT}]} = \frac{3r + (r+1)\theta}{2(r-1) + \theta} \ . \tag{20}$$

As $\theta \in [10, 31]$, this factor demonstrates a substantial improvement. Indeed, as the fraction in (20) is increasing with respect to θ for all $r \ge 2$, it may be bounded from below by its value for $\theta = 10$, which is approximately $\frac{13}{2}$ for large values of r. This improvement stems from the multiplicative form of Newton interpolation polynomials, as opposed to Lagrange interpolants that depend heavily on divisions.

$\diamond$ **Storage at the user end.** The personal identity and share, i.e., $\mathsf{S} = s + \ell$ (same as in our scheme).

$\diamond$ **Revocation message length.** The length of the revocation message, (15), is $\mathsf{L} = rs + 2r\ell$. Comparing this to the revocation message in our scheme, (7), we see that our scheme offers an improvement by a factor of $\frac{s+2\ell}{s+\ell} \approx 2$.

3.3.3 Variants of the NP scheme

What we described in Section 3.3.1 is only one of several variants of the basic NP scheme. We discuss here three additional possible variants of the basic NP scheme that offer some advantages over the variant that was described in Section 3.3.1. The first two of those variants were proposed in [23], while the third one is a novel one.

Shorter revocation messages. The NP scheme does not mandate the inclusion of $\{c'_j\}_{1 \le j \le r}$ in the revocation message. These parameters could be omitted from the revocation message at the expense of increasing the computational cost at the user end. If NP were to be implemented that way, its revocation message length would coincide with ours, but then the computational cost at the user end in such a scheme would be the sum of (18) and (19),

$$\mathsf{C}_{\text{user}} = (r^2 + r) \times [A] + (r^2 + 3r - 1) \times [M] + (2r + 1) \times [D] \ . \tag{21}$$

That would increase the improvement factor in (20) to

$$\frac{\mathsf{C}_{\text{user}}[\mathsf{NP}]}{\mathsf{C}_{\text{user}}[\mathsf{KT}]} = \frac{(r^2 + 3r - 1) + (2r+1)\theta}{2(r-1) + \theta} \ . \tag{22}$$

It may be easily verified that the fraction in (22) is increasing with respect to θ for all $r \geq 2$. Hence, as $\theta \in [10, 31]$, it may be bounded from below by its value at $\theta = 10$, which is $\frac{r^2+23r+9}{2r+8} = O(r)$.

In the remainder of this paper we keep comparing our schemes to the first variant of the NP scheme that was described in Section 3.3.1, keeping in mind the possible tradeoff between the revocation message length and the computational cost at the user end, as offered by the above described second variant. We proceed to discuss now two additional variants.

Working over smaller fields. Naor and Pinkas [23] proposed a way to reduce the computational load. They offered to use c small secrets of size $\frac{\ell}{c}$ bits and c corresponding secret generating polynomials over a smaller field of size $\frac{\ell}{c}$ bits. After the c small secrets are reconstructed they are concatenated to obtain the new key of size ℓ bits. The advantage of this method is that the field operations (multiplication and more importantly division) become cheaper. Since each of the smaller c schemes is information theoretically secure against coalitions of size t, so is the entire scheme, and, consequently, the constructed ℓ-bit secret may serve as a key of a strong symmetric algorithm. This method does not require any more SC memory. We note that the same can be performed with Newton interpolation.

Using the Montgomery trick. One of the anonymous referees suggested the following variant of the NP scheme that has the same revocation message length as our scheme, $\mathsf{L} = rs + r\ell$, with computational cost at the user end that is significantly smaller than (21), but still greater than the one in our scheme, (11). In that variant, the GC broadcasts the following revocation message,

$$\{v_j : 1 \leq j \leq r\} \ ; \ \{Q_j := P(v_j) \cdot c'_j : 1 \leq j \leq r\} \ , \tag{23}$$

where c'_j are as in (14) (this requires the GC to perform r additional multiplications). Next, each user recovers the secret via the identity

$$P(0) = -u \cdot \sum_{j=1}^{r} \left(Q_j \cdot (v_j - u)^{-1} \right) + c_0 \cdot P(u) \quad \text{where} \quad c_0 = \prod_{j=1}^{r} (v_j - u)^{-1} \cdot \prod_{j=1}^{r} v_j \ . \tag{24}$$

The Montgomery trick of simultaneous inversions enables the computation of r inversions at the cost of 1 inversion and $3r - 3$ multiplications (see [7, §3.4]). Using this trick in order to compute the inverse of $v_j - u$, $1 \leq j \leq r$, we arrive at a final cost at the user end of

$$\mathsf{C}_{\text{user}} = 2r \times [A] + (5r - 1) \times [M] + [D] \ . \tag{25}$$

This is still inferior with respect to the cost of the simple and straightforward computation that is carried out by the users in our scheme, (11). Moreover, this version of the NP scheme does not enjoy the other advantages of our scheme that we discuss next.

4 Scalability – the multi-round scheme

Here we discuss scalability issues. As explained in the introduction, the set of revoked users $\mathcal{R}$ is slowly expanding. Therefore, the secret generating polynomial $P(x)$, (5), must be replaced once $\mathcal{R}$ is updated. Hence, it is necessary to perform multi-round revocation in an efficient and scalable way. To set the grounds for this discussion, we introduce the following notations:

• τ_i is the time of the ith revocation request (namely, the ith time in which the GC identifies new users that need to be revoked).

• $\mathcal{R}_i = \{v_1, \ldots, v_{r_i}\}$ is the accumulated set of all revoked users that were identified until time τ_i (including the ones that were identified at τ_i). Clearly, $\mathcal{R}_i \subset \mathcal{R}_{i+1}$ for all $i \geq 1$ (we set $\mathcal{R}_0 = \emptyset$ and $r_0 = 0$).

• $\hat{\tau}_i > \tau_i$ is the time in which the revocation of the newly revoked users in $\mathcal{R}_i \setminus \mathcal{R}_{i-1}$ becomes effective. It is desired to keep the response time $\hat{\tau}_i - \tau_i$ small.

• $[\hat{\tau}_i, \hat{\tau}_{i+1})$ is the ith revocation round. During this time, the set of revoked users is $\mathcal{R}_i$.

• P_i is the secret generating polynomial during the ith revocation round. The common secret during that round is the leading coefficient of P_i.

There are two scenarios to consider: stateless receivers and stateful receivers. The solution in both cases has many common points. Hence, we begin by describing the solution in the stateless scenario, Section 4.1, and in Section 4.2 we comment on how that solution should be adapted for the stateful scenario.

4.1 Stateless receivers

4.1.1 The solution

In this scenario, each SC must be loaded upon issuing with secret shares that will enable its operation for its expected lifetime. To that end, let d be a reasonable upper bound for the number of new revoked users in each revocation round (namely, d should be set so that $r_i - r_{i-1} \leq d$ in most all revocation rounds). Then, as in Kogan et al. [19],the GC generates a sequence of random polynomials of linearly increasing degrees,

$$\Pi = \{P_i(x)\}_{i=1}^{m} \quad , \quad \deg P_i = t_i = i \cdot d \quad , \quad 1 \leq i \leq m \; , \tag{26}$$

where the number of polynomials, m, is a bound on the expected revocation rounds in the lifetime of a typical SC. For example, if the lifetime expectancy of a SC is ten years and the expected duration of a revocation round is at least one month, m should be set to 120.

Assume that we are during the ith revocation round: the secret generating polynomial is P_i, all users in $\mathcal{U} \setminus \mathcal{R}_i$ had recovered the common secret which is the leading coefficient of P_i, and that common secret is already effective. We proceed to describe the transition to the next revocation round. We assume that $r_{i+1} \leq t_{i+1} = t_i + d$, i.e., the new total number of revoked users does not exceed the degree of the next polynomial in the sequence. It is expected that this will usually be the case since $r_i \leq t_i$ and d was selected so that in most rounds $r_{i+1} - r_i \leq d$. The event $r_{i+1} > t_{i+1}$ should not happen frequently since it means that $r_{i+1} - r_i > d + (t_i - r_i)$, namely, the number of newly revoked users exceeds the dilation factor d plus the "credit" $t_i - r_i$ that was accumulated thus far. However, should this event occur, there are two ways to handle it: either to delay the revocation of $r_{i+1} - t_{i+1}$ users to the next round; or to skip to the first polynomial in the sequence P_j, $j > i+1$, for which $r_{i+1} \leq t_j$. For the sake of simplicity of notations, we assume hereinafter that the first strategy is adopted. This way, none of the polynomials in the sequence is skipped and P_i serves as the secret generating polynomial in the ith revocation round.

Let $\rho_{i+1} = t_{i+1} - r_{i+1}$ be the gap between the actual number of revoked users and the degree of P_{i+1}. Then the GC creates ρ_{i+1} phantom users, $\{v_j\}_{r_{i+1}+1 \leq j \leq t_{i+1}} \subset \mathbb{F} \setminus \mathcal{U}$ and augments $\mathcal{R}_{i+1}$ with those fake revoked users. Hence, we get a set of genuine and fake revoked users,

$\hat{\mathcal{R}}_{i+1} = \{v_1, \ldots, v_{t_{i+1}}\}$. With this, the revocation procedure in the new round will be just as described in Section 3, with $\mathcal{R} = \hat{\mathcal{R}}_{i+1}$ and $r = t_{i+1}$.

4.1.2 Efficient implementation

To minimize the response time $\hat{\tau}_{i+1} - \tau_{i+1}$, the key idea is to exploit the Newton form of the interpolants in order to compute and transmit as much information as possible prior to time τ_{i+1}, when the identities of the newly revoked users are revealed. The main observation here is that the GC can split the computation and transmission of the revocation message into two parts: a preliminary part that depends only on the identities of the currently revoked users, and a complementary part that depends on the newly revoked users. The preliminary part, which is typically longer, may be computed already at $\hat{\tau}_i$ (the activation time of the ith revocation round) and be transmitted during idle time in a very low rate since it is not urgent. The complementary part, that depends on the newly revoked users, is much shorter and requires much less computation. An efficient implementation of these ideas consists of the following steps:

1. Preliminary computation and transmission: The GC computes the first r_i coefficients in the Newton representation of P_{i+1} with respect to $\mathcal{R}_i$, (8). Then, starting at $\hat{\tau}_i$, it starts broadcasting the preliminary revocation message, which consists of (see (7)) the identities of the currently revoked users and the above mentioned coefficients, $\{a_k\}_{0 \le k \le r_i - 1}$. Even though the identities of the currently revoked users are already known to all, it is necessary to keep broadcasting them for the sake of new decoders, or decoders that recover from some failure.

2. Complementary computation and transmission: At time τ_{i+1}, when $\mathcal{R}_{i+1} \setminus \mathcal{R}_i = \{v_j\}_{r_i+1 \le j \le r_{i+1}}$ become known, the GC creates the ρ_{i+1} phantom users $\{v_j\}_{r_{i+1}+1 \le j \le t_{i+1}} \subset \mathbb{F} \setminus \mathcal{U}$ and computes the remaining coefficients $\{a_k\}_{r_i \le k \le t_{i+1}-1}$ of P_{i+1} according to (8). Then , the GC broadcasts to all users the complementary revocation message consisting of the identities of the newly revoked users $\{v_j\}_{r_i+1 \le j \le t_{i+1}}$ and the remaining coefficients $\{a_k\}_{r_i \le k \le t_{i+1}-1}$. This transmission, being short and urgent, may take place over a high-rate channel in order to allow fast revocation.

3. Acquisition at the user end: Upon receiving the preliminary revocation message, each user stores the identities of currently revoked users and the corresponding coefficients. In addition, the user may perform at that stage part of the computations in (9), store the results and then complete the computation of the new key when the complementary revocation message arrives. More specifically, a non-revoked user does not need to keep the identities and coefficients that are received through the preliminary revocation message. Instead, he may calculate and keep the following two (non secret) values only:

$$q_1 = \sum_{\ell=0}^{r_i-1} a_\ell \prod_{j=1}^{\ell} (u - v_j) \quad ; \quad q_2 = \prod_{j=1}^{r_i} (u - v_j) \ . \tag{27}$$

Upon receiving the complementary revocation message, each non-revoked user may complete the computation of $a_{t_{i+1}}$ according to (9), rewritten with q_1, q_2:

$$a_{t_{i+1}} = \frac{P(u) - q_1 - \sum_{\ell=r_i}^{t_{i+1}-1} a_\ell \cdot q_2 \cdot \prod_{j=r_i+1}^{\ell} (u - v_j)}{q_2 \cdot \prod_{j=r_i+1}^{t_{i+1}} (u - v_j)} \ . \tag{28}$$

We note that this procedure may be even further improved. Since the degrees of the preloaded polynomials reflect the worst scenario in which in every round d newly revoked users are introduced, $t_i = \deg P_i = i \cdot d$, the difference $t_{i+1} - r_i$ may become quite large in time. Hence, if that difference is larger than d, a good practice would be to add to the preliminary part of the message $(t_{i+1} - r_i - d)$ phantom revoked users and the corresponding coefficients $\{a_k\}_{r_i \le k \le t_{i+1}-d-1}$. This still leaves room for d new revoked users. By doing so, the length of the complementary revocation message is kept a small constant.

4.2 Stateful receivers

In this scenario, there is no need to generate in advance secret generating polynomials for the expected lifetime of the receivers. Instead, the GC generates the next random polynomial P_{i+1} only after time $\hat{\tau}_i$, when the ith round had started. The degree of that polynomial is always set to $t_{i+1} = d$, where d is, as before, an expected upper bound on $r_{i+1} - r_i$. Then, the GC computes $P_{i+1}(u)$ for every user $u \in \mathcal{U} \setminus \mathcal{R}_i$ and transmits that value to the corresponding user in an encrypted message over the low rate unicast channel. As the already revoked users are excluded from this transmission, the new revocation round is independent of the previous ones and, therefore, each round looks like the basic one-time scheme. Therefore, at time τ_{i+1}, when the $r_{i+1} - r_i$ newly revoked users become known, the GC creates $\rho_{i+1} = d - (r_{i+1} - r_i)$ phantom users in order to have a set of exactly d revoked users and then broadcasts the revocation message, as in (7), with $r = d$.

The obvious tradeoff between the stateless and the stateful setting is that the first requires large storage on the SC's EEPROM while the second requires a unicast channel. But the stateful scenario has an additional advantage over the stateless one. In the latter scenario, if a user is revoked in some round, the GC must keep revoking him in all subsequent rounds since that user has shares of the secret generating polynomials of those rounds as well. Consequently, the degrees of the secret generating polynomials are monotonically increasing. Moreover, as those polynomials are determined at initiation, their degrees must relate to the worst predicted scenario. In the stateful scenario, on the other hand, the GC can simply exclude all currently revoked users from the transmission of shares of the new polynomial. Hence, the degree of those polynomials may always be set to d, the estimated upper bound on the number of newly revoked users.

4.3 Cost analysis

We analyze here the cost of the solution in the two settings and compare it to an efficient multi-round implementation of the NP scheme. t_i denotes here the degree of the secret generating polynomial in the ith round. In the stateless scenario, $t_i = i \cdot d$ while in the stateful scenario, $t_i = d$.

$\diamond$ **Computation at the GC.** The preliminary part requires the computation of $\{a_k\}_{0 \le k \le t_i - 1}$. In view of Proposition 3.1, and ignoring the negligible cost of additions and subtractions, we are facing a work load of roughly $t_i^2 \times [M] + t_i \times [D]$. This may be somewhat improved by keeping the products $\{\prod_{j=1}^{i}(v_{k+1} - v_j) : 1 \le i \le k, 0 \le k \le t_i - 1\}$ from the previous round. At time τ_{i+1} the GC has to complete the computation of only d additional coefficients, $\{a_k\}_{t_i \le k \le t_{i+1}-1}$. The cost of this task is approximately $(2dt_i + d^2) \times [M] + d \times [D]$, see (12). We note that in the NP scheme a preliminary computation coincides with the computation in the previous round. Therefore, an efficient implementation of a multi-round NP scheme would keep the values c'_j, (14), from the previous round; the complementary computation can be carried out efficiently in an approximate cost of $(2dt_i + d^2) \times [M] + t_{i+1} \times [D]$.

$\diamond$ **Computation at the user end.** By (11), the computational cost at the user end in the ith round is $\mathtt{C}_{\text{user}} = 2t_i \times [A] + 2(t_i - 1) \times [M] + [D]$. Upon receiving the preliminary revocation message, the user may compute and store q_1 and q_2, see (27). The cost of this preliminary computation is roughly $2t_i \times [M]$. When the complementary part is received, each non-revoked user can perform an additional computation, see (28), the cost of which is approximately $2d \times [M] + [D]$ in order to arrive at the required coefficient $a_{t_{i+1}}$.

$\diamond$ **Storage at the user end.** In the stateless receiver case, each SC holds one identity and m shares, i.e., $\mathtt{S} = s + m\ell$. In the stateful receiver setting, this is reduced to $\mathtt{S} = s + \ell$ as the SC needs to store only the share for the next round. The difference is quite significant. Given that the current life time expectancy of a SC is 10 years and assuming that the average duration of a revocation round is a month, m should be set to 120. Taking $\ell = 128$ and $s = 32$ we get that in the stateless scenario $\mathtt{S} = s + m\ell = 15392$ bits, or 1924 bytes of EEPROM, while in the stateful setting $\mathtt{S} = 160$, i.e., only 20 bytes of EEPROM.

$\diamond$ **Revocation message length.** The revocation message length for the $(i+1)$th round is $\mathtt{L} = t_{i+1} \cdot (s + \ell)$. As we saw earlier, using Newton interpolants enables us to split the revocation message into two parts. The preliminary part, whose length is $t_i \cdot (s + \ell)$, grows linearly with i in the stateless scenario. It can be computed already at time $\hat{\tau}_i$ and be transmitted during idle time throughout the entire ith round. The complementary part, on the other hand, is of short constant length $\mathtt{CL} = d \cdot (s + \ell)$. At time τ_{i+1}, when the identities of the newly revoked users are revealed, the GC computes and broadcasts it in high-rate in order to allow fast transition time.

This offers a significant advantage over the NP scheme. There, the only information that the GC may compute and broadcast prior to τ_{i+1} is the identities and shares of the r_i users that are currently revoked with respect to the next polynomial P_{i+1}. However, since NP uses Lagrange interpolation, it can compute and send the values $\{c'_j\}_{1 \le j \le t_{i+1}}$, (14), only at time τ_{i+1}, together with the identities and shares of the $t_{i+1} - r_i$ newly revoked users. That leaves us with $\mathtt{CL} = d \cdot (s + \ell) + t_{i+1}\ell$ (assuming that we adopt our suggested practice to send the identities and shares of the $t_{i+1} - r_i - d$ phantom users prior to time τ_{i+1}). We see that our scheme offers a saving of $t_{i+1}\ell$ in the high-rate part of the revocation message. Since usually $d \ll t_{i+1}$ in the stateless scenario, this is a significant saving.

To illustrate this difference, assume that $s = 32$, $\ell = 128$, $d = 10$, $r_i = 1000$ and $t_{i+1} = 2000$ (these last two numbers reflect a reasonable assumption that the bound on the number of revoked users in each round is twice the average number). Suppose that just before The Super-bowl, pirate decoders were captured and identified and the GC wishes to revoke them as fast as possible, before the big event starts. The length of the complementary revocation message in our scheme, would be $d \cdot (s + \ell) = 1600$ bits. In the improved NP scheme it would be $d \cdot (s + \ell) + t_{i+1}\ell = 257600$ bits. Consequently, to achieve a given response time, $\Delta\tau = \hat{\tau}_{i+1} - \tau_{i+1}$, the bandwidth that should be allocated to those messages in our scheme is about 160 times smaller than in the NP scheme.

5 A multi-round scheme based on the Decisional Diffie-Hellman assumption

As we saw in the previous section, one way of making the one time revocation scheme reusable is to use new secret generating polynomials. Another option that was explored in [23] is to borrow an idea due to Feldman [13] who suggested to lift Shamir's secret sharing to the exponents. After reviewing the implementation of this idea in the NP scheme, we describe its implementation with Newton interpolation and show that the revocation message in the latter case is shorter. Then, we prove that such a scheme is secure in the sense that the problem that needs to be solved by the revoked users in order to gain information on the common key is at least as hard as the Decisional Diffie-Hellman problem.

Let p be a large prime and $\mathbb{F}_p$ be the Galois field of size p. Let q be a large prime factor of $p-1$ and $\mathcal{G}$ be a subgroup of $\mathbb{F}_p^*$ of size q. Specifically, if a is a generator of $\mathbb{F}_p^*$, we let $\mathcal{G} = \langle g \rangle$ where $g = a^{(p-1)/q}$. We also let $\mathbb{F}_q$ denote the Galois field of size q. The identities of all users will be taken from this smaller field, namely, $\mathcal{U} \subset \mathbb{F}_q$. Finally, we let ℓ_p and ℓ_q denote the number of bits in p and q respectively.

Assume that the (accumulated) number of users to be revoked is expected to be always bounded by t. Then the GC selects a secret generating polynomial $P(x) \in \mathbb{F}_q[x]$ of degree t and assigns each user $u \in \mathcal{U}$ the share $P(u)$. Let $v_1, \ldots, v_t$ be t users that need to be revoked in the next round (some of them may be phantom users). In the DH-based NP scheme, the GC selects a random number $h \in \mathbb{F}_q$ and then broadcasts the revocation message

$$\{v_j : 1 \le j \le t\} \ ; \ g^h \ ; \ \{g^{hP(v_j)} : 1 \le j \le t\} \ ; \ \{c'_j : 1 \le j \le t\} \tag{29}$$

where c'_j are as in (14) (hereinafter, in every reference to a formula of Section 3, set $r = t$). The secret for the next round, in the DH-based NP scheme, is $g^{hP(0)}$. A non-revoked user u may compute that secret using Lagrange interpolation, (13),

$$g^{hP(0)} = (g^h)^{c_0 P(u)} \cdot \prod_{j=1}^{t} \left(g^{hP(v_j)}\right)^{c_j} \tag{30}$$

where c_j, $0 \le j \le t$, are given in (16). In the corresponding version of our scheme, the revocation message will consist of

$$\{v_j : 1 \le j \le t\} \ ; \ g^h \ ; \ \{g^{ha_k} : 0 \le k \le t-1\} \ , \tag{31}$$

and the secret is g^{ha_t}, where a_k, $0 \le k \le t$, are the coefficients of the Newton form of $P(x)$ with respect to $\{v_j : 1 \le j \le t\}$, (8). A non revoked user u may compute that secret, using (9), in the following manner

$$g^{ha_t} = \frac{(g^h)^{P(u)/\alpha_0}}{\prod_{i=0}^{t-1} (g^{ha_i})^{1/\alpha_i}} \quad \text{where} \quad \alpha_i = \prod_{j=i+1}^{t} (u - v_j) \ . \tag{32}$$

Comparing (31) to (29) we see that we offer shorter revocation messages, $\mathtt{L} = ts + (t+1)\ell_p$ instead of $\mathtt{L} = ts + (t+1)\ell_p + t\ell_q$. Moreover, the overhead of $t\ell_q$ in (29) compared to (31) consists of information that could not be part of the preliminary low rate revocation message, and it must be included in the complementary high rate revocation message, see Section 4.3.

5.1 The security of the DH-based multi-round scheme.

The Decisional Diffie-Hellman (DDH) assumption plays a significant role in the construction of cryptographic primitives, e.g., the Diffie-Hellman key exchange protocol [8] or El-Gamal public key encryption and signature schemes [12]. Basically, the assumption states that if $\mathcal{G} = \langle g \rangle$ and $|\mathcal{G}| = q$ is large, no efficient algorithm can distinguish between the two distributions $\langle g^{\alpha}, g^{\beta}, g^{\alpha\beta} \rangle$ and $\langle g^{\alpha}, g^{\beta}, g^{\gamma} \rangle$ where α, β and γ are chosen randomly in $\mathbb{F}_q$, see [4, 24]. Naor and Pinkas showed that the DH-based version of their scheme is secure against up to t revoked users in the sense that, even if those t users were not revoked in polynomially many revocation rounds in the past, any attempt on their behalf to reveal information on the shared secret at the current round, $g^{hP(0)}$, involves the solution of a problem that is at least as hard as DDH, [23, Theorem 2]. We proceed to prove that the DH-based version of our scheme is also secure in that sense.

Theorem 5.1 *The DH-based version of our scheme is secure against coalitions of up to t revoked users. Namely, even if the t revoked users know the secret key from m previous rounds, $\{g^{h_i a_t}\}_{1 \le i \le m}$, where $m = O((\log q)^c)$ for some constant c, they cannot distinguish between the key in the current round, g^{ha_t}, and a random value, assuming DDH.*

Proof. We recall that the scheme uses a fixed secret generating polynomial $P \in \mathbb{F}_q[x]$ of degree t. Letting a_t denote the coefficient of x^t in that polynomial, the secret in each round is g^{ha_t} for some randomly chosen value h. Let us assume, by contradiction, that the coalition of revoked users has an efficient algorithm $\mathcal{A}$ that is capable of distinguishing between g^{ha_t} and a random value when it is given the following additional inputs:

1. Identities and shares of all coalition members, i.e., v_j and $P(v_j)$, $1 \le j \le t$.
2. $m = O((\log q)^c)$ inputs of the form
$$\{w_j^i : 1 \le j \le t\} \ ; \ g^{h_i} \ ; \ \{g^{h_i a_k^i} : 0 \le k \le t-1\} \ ; \ g^{h_i a_t} \ . \tag{33}$$
Here, h_i are random and independent values, $\{w_j^i\}_{1 \le j \le t}$ is the set of revoked users in round number i, $1 \le i \le m$, and a_k^i are the Newton coefficients of $P(x)$ with respect to $\{w_j^i : 1 \le j \le t\}$ (note that the highest coefficient a_t is always the same). The first three components in (33) constitute the revocation message in round number i, see (31). The last component is the secret in that round, and it is known to the coalition because in all m previous rounds at least one member of the coalition was not revoked and was able to compute it by (32).
3. g^h and $\{g^{ha_k} : 0 \le k \le t-1\}$, where a_k here are the Newton coefficients of $P(x)$ with respect to the identities of the coalition v_j, $1 \le j \le t$.

Next, we use $\mathcal{A}$ in order to design an algorithm $\mathcal{B}$ to solve the DDH problem. Assume that the input to $\mathcal{B}$ is the triplet $\langle g^{\alpha}, g^{\beta}, g^{\gamma} \rangle$ and $\mathcal{B}$ needs to decide whether $\gamma = \alpha\beta$ or not. To that end, $\mathcal{B}$ randomly selects $2t$ values from $\mathbb{F}_q$: v_j, $1 \le j \le t$, and a_k, $0 \le k \le t-1$, and defines

$$P(x) = \sum_{k=0}^{t} a_k \prod_{j=1}^{k} (x - v_j), \tag{34}$$

where $a_t = \alpha$ is a coefficient that is unknown to $\mathcal{B}$. It then calls algorithm $\mathcal{A}$ with the following inputs:

1. v_j and $P(v_j)$, $1 \le j \le t$. Note that even though the coefficient $a_t = \alpha$ of $P(x)$ is not known to $\mathcal{B}$, the values of $P(v_j)$, $1 \le j \le t$, depend only on a_k, $0 \le k \le t-1$.
2. Set $m = O((\log q)^c)$ and then, for each $1 \le i \le m$, generate an input of the form (33), where the values w_j^i may be arbitrary and h_i are random and independent. This requires the computation of a_k^i, $0 \le k \le t-1$, namely, the Newton coefficients of $P(x)$, (34), with respect to the base set $\{w_j^i : 1 \le j \le t\}$. These values may be found as follows: First, using (34), it is possible to determine the value of $\{P(w_j^i)\}_{1 \le j \le t}$ up to the unknown coefficient $a_t = \alpha$; namely, we may express those polynomial values in the form $P(w_j^i) = \mu_j^i + \alpha \nu_j^i$, where μ_j^i and ν_j^i are known. Next, using equations (8) and (9), we may express the Newton coefficients $\{a_k^i\}_{0 \le k \le t}$ as an affine function of the unknown coefficient $a_t = \alpha$. Finally, even though those coefficients depend on $a_t = \alpha$, we may still compute the values $\{g^{h_i a_k^i}\}_{0 \le k \le t}$ in (33) since their dependence on α is only through g^α, which is given as an input.
3. g^h and $\{g^{ha_k} : 0 \le k \le t-1\}$, where a_k here are as in (34) and $h = \beta$.
4. g^γ as the random value that needs to be distinguished from g^{ha_t}.

Algorithm $\mathcal{A}$ then returns `TRUE` if it decides that the last input, g^γ, equals g^{ha_t} and `FALSE` otherwise. Since $g^{ha_t} = g^{\alpha\beta}$, $\mathcal{B}$ returns the same value as $\mathcal{A}$ returns. Hence, the above described algorithm $\mathcal{B}$ solves the DDH problem in polynomial time, in contradiction to the DDH assumption. $\square$

5.2 A concluding remark.

The advantage of raising the secret sharing to the exponents is clear – the secret sharing polynomial is made reusable and, consequently, each SC has to store only one share. However, there are two disadvantages to keep in mind. First, the degree of the polynomial must be set to a sufficiently large value that takes into account the worst scenario of (accumulated) pirate activity. This means that usually we shall be working with a polynomial of degree that is unnecessarily much larger than the actual number of revoked users. This translates into longer revocation messages and more computational work by the GC and by the users. Moreover, due to sub-exponential attacks on the discrete logarithm problem, e.g., the index calculus method, the bit-length of p must be much higher than ℓ in order to furnish comparable security. This means that instead of working in a field of size, say, 128 bits, the computations in the Diffie-Hellman-based schemes should take place in a field of much larger size (e.g., 1024 bits). This, again, has a toll on both bandwidth and computation time. Hence, this variant of the revocation scheme is more suitable for the stateless scenario. In the stateful receiver scenario, the original scheme seems more attractive.

6 Applications to other environments

6.1 Reed-Solomon codes

Reed-Solomon codes [26] are ubiquitous. As they are based on polynomial interpolation, they too can benefit from replacing Lagrangian by Newtonian interpolation. A fundamental idea in Reed-Solomon codes is the grouping together of runs of consecutive bits to form a *symbol* and then consider the symbols as the basic data unit. Letting t denote the length of a symbol in bits, each symbol may be viewed as an element of the Galois field $\mathbb{F}_q$ where $q = 2^t$. Then, by breaking the message into *blocks* of k consecutive symbols, each block $(a_0, \ldots, a_{k-1}) \in \mathbb{F}_q^k$ is identified with the polynomial $P(x) = \sum_{i=0}^{k-1} a_i x^i$. By choosing k to be less than q, the idea is to transmit all of the q polynomial values in the field $\{P(x) : x \in \mathbb{F}\}$. The difference $q - k$ denotes the redundancy of such a transmission, as any k of the transmitted q values is sufficient for the recovery of $P(x)$ at the receiver. This redundancy is used to correct errors in the transmission. By combinatorial reasoning and some linear algebra it may be shown that the Reed-Solomon code may correct up to $(q - k)/2$ errors. An erasure is an error with a known location. In settings where error locations are known (e.g., by side information in a demodulator signal-to-noise ratios, or by CRC error detection), symbols with errors may be erased and ignored; such errors are called erasures. The Reed-Solomon code may cope with up to $q - k$ erasures. In general, if the transmission introduces e errors and s erasures, the Reed-Solomon code may correct them provided that $2e + s \leq q - k$. A typical choice of parameters is $q = 256$ (namely, each symbol is a byte, $t = 8$) and $k = 233$. Working with symbols as the basic data unit, rather than with bits, makes these codes better suited to applications where errors occur in bursts (since, for example, a burst of 16 wrong bits may induce no more than 3 wrong symbols).

In settings where there are only erasures, the data is recovered by means of interpolation. Here, using Newton rather than Lagrange interpolation enjoys all of the benefits that were discussed before. Namely, the receiver may start decoding the message the moment it receives the first symbol (while, when using Lagrange interpolation, it has to wait until k correct symbols are received) and the computational cost is smaller. Hence, Newton interpolation is a better choice for implementing Reed-Solomon codes.

6.2 Multicast Scenarios

Here we consider multicast scenarios, e.g., multimedia broadcasting over the internet, where the GC controls the decryption capabilities of the members and transmits the streams of data, or working groups, where one of the members may act as the GC by controlling the decryption capabilities of the group members and deciding which members may join the group or should leave it. In such scenarios, as opposed to broadcast scenarios, when a member leaves the group the network stops routing packets to him, while, conversely, when a new member joins the group the network starts routing packets to him. Such a solution for controlling the access to data may not suffice in some settings, for instance, when the transmitted data is of confidential nature. In such cases it may be desirable to perform re-keying when new members join the group in order to prevent them from accessing old content, as well as when members are revoked in order to prevent them from accessing any further communications.

As mentioned in Section 1.2, re-keying and revocation schemes for multicast communication networks have been very popular in recent years. Quite a few of the proposed schemes are based on the LKH (Logical Key Hierarchy) algorithm and its many variants [30, 31, 5, 25]. The basic LKH scheme was suggested in the context of secure multicast, that builds secure communications on top of the Internet's multicast layer. In LKH schemes every user keeps a personal key that is composed of $\log n$ keys and the GC keeps a (binary) tree of keys. When a single user is revoked, the GC must change all the keys in the tree that are in the path from the revoked user's leaf to the root. All the non-revoked users must change their keys in the intersection between the path from their leaf to the root and the path from the revoked user's leaf to the root. Therefore, a message revoking a single user consists of $O(\log n)$ keys.

A major drawback of the LKH family of schemes is that users must update their state whenever the group key is changed (either when users leave or join the group). This means that whenever a receiver is turned on after a dormant period, the GC must send it a synchronization message to bring it up to date. Those messages have a dear toll in terms of bandwidth. Stateless revocation schemes are free of this drawback. Therefore, our Newton interpolation based revocation scheme in its stateless variant enjoys a low channel communication overhead, low computational overhead and flexible storage requirements at the user end. These advantages make it a good choice for such settings.

6.3 Tactical Wireless Networks

Wireless ad-hoc networks are commonly used for communication in tactical situations such as anti-terrorist operations, rescue missions, and battlefields, where usually there is no network infrastructure. In such networks, access to the (broadcast) channel is typically regulated by a mixture of TDMA, FDMA and CDMA protocols. In some systems the communication is only half-duplex; namely, in any given time each radio device may either transmit or listen to the channel. Modern tactical wireless networks provide at least half duplex voice communication between the group members, as well as data communications.

Since the messages transmitted in such systems are usually sensitive, a group key, that is shared by all members of the group, is used to encrypt all transmissions. This simple solution may pose a severe problem in case one of the radio devices is captured by the enemy. Therefore, the capability to revoke nodes is of paramount importance. Revocation protocols for wireless tactical networks should take into account the special nature of the communication. Due to the high mobility of the nodes there are many packet losses and communication disruptions. In addition, the communication channels of tactical wireless networks have a relatively low, and dynamically changing bandwidth (compared to wired networks, and even to static wireless networks). The available bandwidth for key management and revocation purposes is strictly limited. Therefore, it is highly preferable to implement a revocation scheme that is based solely on broadcast communications. That way, each time a revocation message is broadcast, several radio devices can simultaneously receive and update their keys, and each of them may also serve as a repeater for those transmissions.

Typical wireless tactical networks are of much smaller sizes than multimedia distribution systems. Hence, secret-sharing based revocation schemes, that are usually limited in the stateless scenario regarding the maximal number of users that can be revoked before the personal information of all users must be updated, are well-suited for systems of smaller sizes. Therefore, our Newton based revocation scheme in its stateless variant is a good candidate for such networks.

References

[1] M. Abdalla and Y. Shavitt and A. Wool, Key management for restricted multicast using broadcast encryption, *IEEE/ACM Transactions on Networking*, Vol.8, 4, pp. 443-454, 2000.

[2] J. Anzai, N. Matsuzaki and T. Matsumoto, A quick group key distribution scheme with entity revocation, *Advances in Cryptology - ASIACRYPT '99*, LNCS 1716, Springer Verlag 1999, pp. 333-347.

[3] S. Berkovits, How to broadcast a secret, *Advances in Cryptology - EUROCRYPT '91*, LNCS 547, Springer Verlag 1991, pp. 535-541.

[4] D. Boneh, The decision Diffie-Hellman problem, *The Third Algorithmic Number Theory Symposium*, LNCS 1423, Springer Verlag 1998, pp. 48-63.

[5] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor and B. Pinkas, Multicast security: a taxonomy and efficient constructions, *IEEE INFOCOM '99*, 1999, pp. 708-716.

[6] R. Canetti and T. Malkin and K. Nissim, Efficient communication-storage tradeoffs for multicast encryption, *Advances in Cryptology - EUROCRYPT '99*, LNCS 1592, Springer Verlag 1999, pp. 459-474.

[7] H. Cohen, A. Miyaji and T. Ono, Efficient elliptic curve exponentiation using mixed coordinates, *Advances in Cryptology - ASIACRYPT '98*, LNCS 1514, Springer Verlag 1998, pp. 51-65.

[8] W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory*, Vol. 22, 1976, pp. 644-654.

[9] Y. Dodis and N. Fazio, Public key broadcast encryption for stateless receivers, *Digital Rights Management Workshop*, LNCS 2696, 2002, pp. 61-80.

[10] Y. Dodis and N. Fazio, Public key trace and revoke scheme secure against adaptive chosen ciphertext attack, *Public Key Cryptography 2003*, LNCS 2567, Springer Verlag 2002, pp. 100–115.

[11] Y. Dodis, N. Fazio, A. Kiayias and M. Yung, Scalable public-key tracing and revoking, *Principles of Distributed Computing - PODC '03*, 2003, pp. 190-199.

[12] T. El-Gamal, A public key cryptosystem and a signature scheme based on discrete logarithms, *Advances in Cryptology - CRYPTO '84*, LNCS 196, Springer Verlag 1985, pp. 10-18.

[13] P. Feldman, A practical scheme for non-interactive verifiable secret sharing, *The 28th IEEE Symposium on Foundations of Computer Science*, 1987, pp. 427-437.

[14] A. Fiat and M. Naor, Broadcast encryption, *Advances in Cryptology - CRYPTO '93*, LNCS 773, Springer Verlag 1994, pp. 480-491.

[15] J. A. Garay and J. Staddon and A. Wool, Long-lived broadcast encryption, *Advances in Cryptology - CRYPTO 2000*, LNCS 1880, Springer Verlag 2000, pp. 333-352.

[16] M. T. Goodrich, J. Z. Sun and R. Tamassia, Efficient tree-based revocation in groups of low-state devices, *Advances in Cryptology - CRYPTO 2004*, LNCS 3152, Springer Verlag 2004, pp. 511-527.

[17] D. Halevy and A. Shamir, The LSD broadcast encryption scheme, *Advances in Cryptology - CRYPTO 2002*, LNCS 2442, Springer Verlag 2002, pp. 47-60.

[18] C. H. Kim, Y. H. Hwang and P. J. Lee, An efficient public-key trace and revoke scheme secure against adaptive chosen ciphertext attack, *ASIACRYPT 2003*, LNCS 2894, Springer Verlag 2003, pp. 359-373.

[19] N. Kogan, Y. Shavitt and A. Wool. A practical revocation scheme for broadcast encryption using smartcards, *The 24th IEEE Symposium on Security and Privacy*, 2003, pp. 225-235.

[20] R. Kumar, S. Rajagopalan and A. Sahai, Coding constructions for blacklisting problems without computational assumptions, *Advances in Cryptology - CRYPTO '99*, LNCS 1666, Springer Verlag 1999, pp. 609-623.

[21] M. Luby and J. Staddon, Combinatorial bounds for broadcast encryption, *Advances in Cryptology - EUROCRYPT '98*, LNCS 1403, Springer Verlag 1998, pp. 512-526.

[22] D. Naor, M. Naor and J. B. Lotspiech, Revocation and tracing schemes for stateless receivers, *Advances in Cryptology - CRYPTO 2001*, LNCS 2139, Springer Verlag 2001, pp. 41-62.

[23] M. Naor and B. Pinkas, Efficient trace and revoke schemes, *Financial Cryptography 2000*, LNCS 1962, Springer Verlag 2000, pp. 1-20.

[24] M. Naor and O. Reingold, Number-theoretic constructions of efficient pseudo-random functions, *The 38th IEEE symposium on Foundations of Computer Science*, 1997, pp. 458-467.

[25] B. Pinkas, Efficient state updates for key management, *Digital Rights Management Workshop 2001*, LNCS 2320, 2001, pp. 40-56.

[26] I. S. Reed and G. Solomon, Polynomial codes over certain finite fields, *Journal of the Society of Industrial and Applied Mathematics 1960*, Vol. 8, pp. 300-304.

[27] A. Shamir, How to share a secret, *Communications of the ACM 1979*, Vol. 22, pp. 612-613.

[28] D. Stinson, *Cryptography, Theory and Practice*, Third Edition, CRC Press, 2005.

[29] W. G. Tzeng and Z. J. Tzeng, A public-key traitor tracing scheme with revocation using dynamic shares, *Public Key Cryptography 2001*, LNCS 1992, Springer Verlag 2001, pp. 207-224.

[30] D. M. Wallner, E. J. Harder and R. C. Agee, Key management for multicast: issues and architectures, *Request for Comments 2627*, June 1999.

[31] C. K. Wong, M. Gouda and S. S. Lam, Secure group communications using key graphs, *IEEE/ACM Transactions on Networking 2000*, Vol. 8, 1, Feb 2000, pp. 16-30.