Fault Tolerant and Distributed Broadcast Encryption

Paolo D'Arco[1] and Douglas R. Stinson[2]

[1] Dipartimento di Informatica ed Applicazioni
Università degli Studi di Salerno, 84081, Baronissi (SA), Italy
paodar@dia.unisa.it
[2] School of Computer Science
University of Waterloo, N2L 3G1, Waterloo Ontario, Canada
dstinson@cacr.math.uwaterloo.ca

Abstract. A broadcast encryption scheme enables a server to broadcast information in a secure way over an insecure channel to an arbitrary subset of priviliged recipients. In a set-up phase, the server gives pre-defined keys to every user of the system, using secure point-to-point channels. Later on, it broadcasts an encrypted message along a broadcast channel, in such a way that only users in a priviliged subset can decrypt it, by using the pre-defined keys received in set-up phase. Usually, the broadcast message contains a fresh session key, which can subsequently be used for secure broadcast transmission to the priviliged set of recipients. In this paper we deal with two aspects of secure broadcast transmission: *reliability* and *trust* in the broadcaster. The first is a well-studied issue in communication over unreliable channels: packets can get lost and some redundancy is required to provide reliable communication. The second aspect concerns with the assumption that the broadcaster, who receives information for broadcasting from several entities, must be trusted. This issue has not previously been addressed in the broadcast transmission setting. We provide a motivating scenario in which the assumption does not hold and, for both problems, we review and extend some existing broadcast encryption schemes, in order to gain fault tolerance and to remove the need for trust in the broadcaster.

1 A Motivating Scenario

A movie company is willing to distribute its re-make of *Gone with the Wind*, and it is looking for an agreement with a pay-per-view broadcaster. The broadcaster uses a broadcast encryption scheme to distribute a common key to the users who have subscribed to the service, and later it broadcasts an encrypted version of the movie that can be decrypted only by the users who had received the common key when running the broadcast encryption scheme. Therefore, only users who have paid can enjoy the movie.

However, since people have been waiting for a long time for this re-make of the famous movie, several broadcasters would like to sign a contract with the movie company. It is supposed to be a good deal! Aware of that, the company, in its contract proposal, asks for a high percentage of the money paid by the users to subscribe to the broadcast service. One of the broadcasters, close to bankruptcy, accepts the offer, even if he thinks the contract is not fair to him. Therefore he decides, in order to increase his income, to sell illegally, on the "black market", a copy of the movie that will later be re-distributed by other means.

Notice that the broadcaster is almost completely safe: he could always say that some user has stored and illegally re-distributed the movie. There is no way to establish who is guilty, even if some watermark has been embedded by the company into the digital representation of the movie[1].

The main problem of the above transmission model is that the broadcaster gets a copy *in clear* of the movie. So he must be completely trusted by the movie company.

The question we are going to study in this paper is the following: Is there any way by means of which the company can broadcast the movie without trusting the broadcaster? And, actually, a more general question is: is it possible to design a system by means of which an entity can delegate another one *to blindly broadcast* information in a secure way to a given set of users?

A trivial solution could be that the broadcaster just receives an encrypted copy of the movie, and the company runs a broadcast encryption scheme (BES, for short) with the users in order to give them the decryption key. But this means that the company must be involved directly in transmission issues and must interact with millions of users.

The idea we would like to investigate is the following: instead of a single server, there are n servers across the system. The movie company, in a set-up phase, gives to each server an encrypted partial version of the movie and some key material. Later on, the n servers run a protocol with the users, enabling a given subset of them to recover the decryption key. Finally, each server starts broadcasting the encrypted partial version of the movie he got from the company. The priviliged users, by means of the broadcasts and of the decryption key, recover and watch the original movie. Notice that, with this approach, the movie company only needs to communicate with a fixed and *small number* of servers, compared to the scenario in which the company directly manages the key distribution with the users. Moreover, a single server does not get any information about the movie.

A small example. To exemplify the idea we are thinking about, let us consider the following simple protocol. Let $\mathcal{U} = \{U_1, \ldots, U_m\}$ be a set of m

[1] The Digital Watermarking Technology in recent years has produced plenty of possible solutions for copyright related problems. We point out that the notion of a distributed broadcast encryption scheme, which we are going to introduce in the following, is not a substitute for such schemes and it does not address the same problems. The problem it solves is how to remove trust in a third party responsible for the transmission of some valuable information. Maybe, in some settings or in the design of a complex architecture, such a scheme can be used in conjunction with digital watermarking schemes, but we stress that the goals are different.

users and let $\mathcal{S} = \{S_1, \dots, S_n\}$ be a set of n servers. The value n is relatively small ($n < 10$, say) but m can be very large (e.g., $m > 1000000$). Every user has a secure *point-to-point channel* with every server. Servers and users have access to a common *broadcast channel*. Let Σ be a broadcast encryption scheme. A distributed solution for the movie problem works as follows:

Set-up Phase.

1. The movie company chooses a random key k.
2. Then it uses a (t, n) secret sharing scheme to compute n shares of k, say $y_1, \dots, y_n$.
3. At the same time, every server runs an *independent* set up phase of a BES scheme Σ with the users.

Broadcast Phase.

1. The movie company splits the movie into n parts say, $stream_1, \dots, stream_n$, and encrypts every part by computing $E_k(stream_i)$ for $i = 1, \dots, n$. Then, it sends to server S_i, the encrypted stream $E_k(stream_i)$, for $i = 1, \dots, n$.
2. Every server S_i, using Σ, broadcasts the value y_i.
3. From any subsets of t out of the n values y_i's, each priviliged user recovers the decryption key k.
4. Every server S_i broadcasts $E_k(stream_i)$.
5. Every user belonging to the priviliged set collects and re-arranges the n parts, decrypts them, and watches the movie.

The above protocol is trivial. The company encrypts the movie and shares, according to a threshold scheme, the secret key among the servers. Then, each server broadcasts to the priviliged users his share. Finally, any t-subset of shares enables the recovering of the key. With such a system any $t - 1$ servers have no information about the movie (*security*) and there is enough redundancy in order to recover the decryption key, because t out of the n broadcasts are sufficient to recover the key k.

The question is: can we do better? Can we design a scheme *from scratch* where each server broadcasts *less information* compared to the trivial protocol (in which basically each server runs a broadcast encryption scheme)? Can we design a distributed broadcast encryption scheme where, during the set-up phase, the movie company generates information for a sort of "global broadcast encryption scheme" that it distributes to the servers and which is used by the servers to send pre-defined keys to the users?

Remark. Notice that, if the company, instead of just splitting the movie into n parts, divides the movies in packets in such a way that *at least* two different servers broadcast the same packet, or it uses a threshold secret sharing scheme to share the encrypted movie among the servers, or it uses other techniques like the IDA (Information Dispersal Algorithm) then, even during the broadcast of the movie, if some server crashes, the transmission is still successful (i.e., we enhance the reliability of the scheme). Of course, the technique used in this phase (i.e., simple splitting of the movie, secret sharing, IDA, etc.), implies different performance, storage requirement for the servers, and communication complexity of the transmission.

Previous work. Broadcast encryption was first considered in [4] and, subsequently, formally defined and studied in [16]. Since then, it has become a major topic in Cryptography, due to the large number of possible applications, and it has evolved in several directions. These directions include multicast communications [44,9,10,33,34,13], where the priviliged subset of recipients *dynamically* changes by means of join and remove operations; traitor tracing [11,32,14, 8,41,3,19,17,35,37,21,22], where the emphasis is on catching dishonest users who set up illegal decrypting devices; and revoking schemes [2,30,24,29,20,25], which allow efficient and fast revocation of a small group of users. Moreover, several schemes presented in the literature achieve more than one of the above-mentioned aspects, e.g., broadcast plus traitor tracing capabilities [18,40], or revocation and tracing capabilities [30,29,20]. Broadcast encryption requirements have been studied in several papers, e.g., [5,6,27]. A survey on unconditionally secure broadcast encryption schemes can be found in [38]. Recently, some papers have considered a setting in which packets can get lost during transmission. In [31], error correction techniques have been employed. In [45], short "hint" messages are appended to the packets. The schemes given in [24], by carefully choosing the values of the parameters, can provide resistance to packet loss as well. In [36,26], the group key establishment problem over unreliable networks has been addressed, and a key recovery mechanism, quite similar in both papers, has been provided: each packet sent by the broadcaster enables the users to recover the current session key and a share of previous and subsequent ones.

2 An Informal Model

In the following, we are mainly interested in schemes for the distribution of the encryption-decryption keys to the users by means of the servers. More precisely, we consider the following model. Let $\mathcal{CP}$ be a content provider (e.g., a movie company), let $\mathcal{U} = \{U_1, \ldots, U_m\}$ be a set of users, and let $\mathcal{S} = \{S_1, \ldots, S_n\}$ be a set of servers. The content provider is connected to the servers by means of *secure point-to-point channels*, and each server is connected to the users by means of *secure point-to-point channels*. Servers and users have access to a common *broadcast channel*.

An (m, n, t, ω)-DBES scheme enables a content provider $\mathcal{CP}$ to transmit a message to a priviliged subset of users via the set of n servers in such a way that no coalition of $t-1$ servers and no coalition of at most ω unauthorized users can compute any information about the message. The protocol is divided into four steps:

- *Set up Phase - Step* 1*: Content Provider-Servers.* The content provider $\mathcal{CP}$ sends information to the n servers along secure point-to-point channels.
- *Set up Phase - Step* 2*: Servers-Users.* The n servers send private information to every user of the system along secure point-to-point channels.
- *Broadcast Phase - Step* 1. The $\mathcal{CP}$ encrypts the message k he wishes to transmit, and sends encrypted information to the servers along secure point-to-point channels.
- *Broadcast Phase - Step* 2. The n servers broadcast encrypted information along the broadcast channel in such a way that only a priviliged subset of users $P \subseteq \mathcal{U}$ can recover the original message k. Notice that $P \subseteq \mathcal{U}$ could be arbitrary, and not known before the broadcast phase.

The properties that an (m, n, t, ω)-DBES must satisfy are the following:

1. *Fault-tolerant Reconstruction.* ¿From the broadcast of any t out of the n servers, each of the users belonging to the priviliged subset P can reconstruct the key k.
2. *Security with respect to receivers.* No coalition of at most ω unprivileged users can determine any information about k.
3. *Security with respect to servers.* No $t-1$ servers can determine any information[2] about k.
4. *Efficiency.* The amount of broadcasted and secret information is as small as possible.

Property 1 implies that we have *fault tolerant* transmissions: priviliged users are still able to compute k even if some broadcasts get lost during the transmission, or some servers of the system crashes. As we will show, *fault tolerance* and *distribution* are independent targets. Some of the techniques we propose in the following apply to both BES and DBES Schemes.

3 Blacklisting Problem Constructions

An interesting approach to a special kind of broadcast encryption has been introduced in [24]. The problem considered therein is how to broadcast information in a secure way to *all but a few users* of the system. This special case of broadcast encryption is also referred to as *broadcast exclusion* or *blacklisting problem.*

Let us state the following combinatorial definition:

Definition 1. *[15] Let $\mathcal{K} = \{k_1, \ldots, k_n\}$ be a ground set, and let d, ω be positive integers. A family $\mathcal{A} = \{A_1, \ldots, A_m\}$ of subsets of $\mathcal{K}$ is an (ω, d)-cover-free family if, for all $A, A_1, \ldots, A_\omega \in \mathcal{A}$, such that $A \notin \{A_1, \ldots, A_\omega\}$, it holds that*

$$| A \setminus \cup_{i=1}^{\omega} A_i | \geq d.$$

[2] This is the strongest possible security condition. Indeed, the reconstruction property implies that *any* subset of t servers, from the private information received by $\mathcal{CP}$ during the set up phase, can compute the secret key k. This is unavoidable if we require that the broadcast of t servers is sufficient for the recipients to recover the secret key k.

The definition guarantees that the union of ω subsets does not completely cover any other subset. More precisely, at least d elements are not covered.

Given a cover-free family, a broadcast exclusion scheme can be easily set up as follows: let $\mathcal{K}$ be the set of keys, and let $m = |\mathcal{A}|$ be the number of users of the system.

1. Each user u_i gets the keys associated with A_i.
2. When the broadcaster wishes to send a message to all but users $u_{i_1}, \ldots, u_{i_\omega}$, he encrypts the message m with each key $k \in \mathcal{K} \setminus \cup_{i=1}^{\omega} A_i$, and broadcasts $E_k(m)$.

Because $d \geq 1$, every priviliged user recovers the message, while for $u_{i_1}, \ldots, u_{i_\omega}$ there is no way to them to get any information since their keys are not used (i.e., they are excluded) by the broadcaster. The size of the broadcast can be improved if, instead of repeating the encryption of m for each key in $\mathcal{K} \setminus \cup_{i=1}^{\omega} A_i$, the message m is encoded through an *erasure code* into $|\mathcal{K} \setminus \cup_{i=1}^{\omega} A_i|$ smaller but redundant pieces, where any d of them enable recovering m. Several constructions were given in [24], and the reader is referred to that paper for details. Actually, it turns out that a more general goal can be achieved by means of *ramp schemes* (see Appendix A). By means of an accurate choice of the parameters of the ramp scheme, it is possible to gain fault tolerant transmission as well. For example, if the message m is shared according to a $(0, e, r)$-RS, where $e < d \leq r$ and $r = |\mathcal{K} \setminus \cup_{i=1}^{\omega} A_i|$, then any e shares enable to recover m. In this case, $d - e$ is the fault tolerance factor provided by the scheme.

The construction we have seen before can be adapted to our distributed setting by simply using $|\mathcal{K}|$ servers. More precisely, during the set up phase, server S_i gets key k_i from $\mathcal{CP}$, and user u_j gets all keys $k_i \in A_j$ from the corresponding servers S_i. The two-step broadcast phase, is carried out as follows: $\mathcal{CP}$ shares the message m according to a $(0, e, r)$-RS ramp scheme, and sends a different share to each server whose key does not belong to $\cup_{s=1}^{\omega} A_{j_s}$. Then, every server S_i encrypts with key k_i and broadcasts the received share.

4 A Small Subset of Priviliged Users

The same approach applied in [24] for the broadcast exclusion problem can be applied in another setting: namely, when *both the size of the priviliged subset of users and the size of the possible forbidden subsets of users are small.* More precisely, we can use the same idea to set up a broadcast encryption scheme where users in P recover the message, but *any subset* F of size at most ω has no information about it. We now need a generalization of the idea of (ω, d)-cover-free family. More precisely, we state

Definition 2. *[43] Let $\mathcal{K} = \{k_1, \ldots, k_n\}$ be a ground set, and let r, ω, d be positive integers. A family $\mathcal{A} = \{A_1, \ldots, A_m\}$ of subsets of $\mathcal{K}$ is an (r, ω, d)-cover-free family if, for all subsets $A_{i_1}, \ldots, A_{i_r}$, and $A_{j_1}, \ldots, A_{j_\omega} \in \mathcal{A}$, such that*

$A_{j_\ell} \neq A_{j_k} \forall k, \ell$, *it holds that*

$$| \cap_{s=1}^{r} A_{i_s} \setminus \cup_{s=1}^{\omega} A_{j_s} | \geq d.$$

The definition establishes that the intersection of any r subsets still has at least d cover-free elements, for any choice of ω other subsets. Efficient constructions of (r, ω, d)-cover-free family are given by Stinson and Wei in [43].

A broadcast encryption scheme can be set up as follows: Given an (r, ω, d)-cover-free family on the set of keys $\mathcal{K}$, let $P = \{u_{i_1}, \ldots, u_{i_r}\}$ be the subset of priviliged users. Then,

1. Each user u_i gets the keys associated with A_i. Let $Y = \cap_{s=1}^{r} A_{i_s}$ be the set of common keys held by users in P.
2. When the broadcaster wishes to send a message to users in P, in such a way that any disjoint subset of size ω, say $F = \{u_{j_1}, \ldots, u_{j_\omega}\}$, does not recover the message, he shares the message m according to a certain $(y-d, e, y)$-RS, where $y = |Y|$, and encrypts share m_i with key $k_i \in Y$, and broadcasts $E_{k_i}(m_i)$.

It is not difficult to see that every user in P has all y keys in $Y = \cap_{s=1}^{r} A_{i_s}$. On the other hand, every F of size ω has at most $y - d$ of these keys, due to the property of the (r, ω, d)-cover-free family. The use of a $(y - d, e, y)$-RS guarantees that priviliged users recover the message, while a forbidden subset gains no information at all about the message. Moreover, the above construction has a fault tolerance factor equal to $y - e$. It can be distributed along the same line as the blacklisting construction. We skip the details.

The above construction can still be improved in terms of the length of the broadcast, by applying the same technique given in subsection 3.1 of [38] in the context of key predistribution schemes. Basically, the idea is that, instead of using *all* the y keys to generate the broadcast, a smaller subset of z keys *is derived* from the y keys and used with a $(0, e, z)$-RS to encrypt the shares for the priviliged users. The construction is secure because *only* the priviliged users can derive the actual keys used in the broadcast. The fault tolerance factor in this case is given by $z - e$. Formally, we state the following:

Definition 3. *A function $f : GF(q)^n \to GF(q)^z$ is said to be balanced if, for each $y \in GF(q)^z$*

$$|\{x \in GF(q)^n | f(x) = y\}| = q^{n-z}.$$

In other words, each value $f(x) \in GF(q)^z$ has the same number of pre-images x.

Definition 4. *A function $f : GF(q)^n \to GF(q)^z$ is said to be k-resilient if, for every subset $\{j_1, \ldots, j_k\} \subseteq \{1, \ldots, n\}$ and every $(a_1, \ldots, a_k) \in GF(q)^k$, the function $f(x_1, \ldots, x_n)|_{x_{j_1}=a_1, \ldots, x_{j_k}=a_k}$ is balanced.*

Roughly speaking, if f is k-resilient, the knowledge of any k input values *does not* give any information about the output (i.e., the outputs are still uniformly distributed).

Hence, in the above protocol, assuming that a public $(y-d)$-resilient function $f : GF(q)^y \to GF(q)^z$ is available, and that $Y = GF(q)$, the broadcaster can use the keys in Y to derive the actual keys for encrypting the broadcast. Every user in P can compute such keys as well; while, the $(y-d)$-resilience property ensures that any coalition F does not get any information about them.

5 Distribution of Some BES Constructions

Many BES constructions can be rewritten to fit our model. The first ones we are going to consider are the one-level and multi-level schemes described in [16].

The idea underlying the following schemes is of first constructing a scheme that works for excluding a single user and then, using multi-layered hashing techniques, to break up coalitions of excluded users into single exclusions from sub-groups.

5.1 Basic Construction

The first broadcast encryption scheme given in [16], which will be used with $\omega = 1$ in the multi-layered constructions, enables the broadcaster to send a message to any $P \in \mathcal{P} \subseteq 2^{\mathcal{U}}$ in such a way that no $F \in \mathcal{F} = \{F \in \mathcal{U} : |F| \leq \omega\}$ such that $F \cap P = \emptyset$ gains any information about it. The scheme can be described as follows:

Basic Fiat-Naor BES.

- *Set up Phase.* For every subset $F \in \mathcal{F}$, $\mathcal{CP}$ chooses a random value $s_F \in GF(q)$ and gives s_F to every member of $\mathcal{U} \setminus F$. The key associated with a privileged set P is defined to be

$$\kappa_P = \sum_{F \in \mathcal{F} : F \cap P = \emptyset} s_F.$$

- *Broadcast Phase.* $\mathcal{CP}$ broadcasts the value

$$b_P = \kappa_P + m_P \bmod q.$$

It is not difficult to see that each user in P recovers the message, while any coalition F, disjoint from P, cannot, since the coalition does not have the value s_F.

5.2 One-level and Multi-level Schemes

In order to describe the multi-level constructions, let us recall the following definition:

Definition 5. *An (n, m, ω)-perfect hash family is a set $\mathcal{H}$ of functions*

$$f : \{1, \ldots, n\} \to \{1, \ldots, m\}$$

such that, for every subset $X \subseteq \{1, \ldots, n\}$ of size ω, there exists a function $f \in \mathcal{H}$ whose restriction to X is one-to-one.

An (n, m, ω)-perfect hash family is usually denoted by $\mathrm{PHF}(N, n, m, \omega)$, where $|\mathcal{H}| = N$. Using several one-resilient BES schemes and a $\mathrm{PHF}(N, n, m, \omega)$, it is possible to set up a ω-resilient BES scheme as follows [16]: For $1 \leq i \leq N$ and $1 \leq j \leq m$, let $R(i, j)$ be a $(n, 1)$-BES scheme, and let $\mathrm{PHF}(N, n, m, \omega)$ be a family of perfect hash functions.

- *Set-up Phase.* $\mathcal{CP}$ sends to every user $i \in \{1, \ldots, n\}$ the keys associated with him in the scheme $R(i, f_j(i))$, for $j = 1, \ldots, N$.
- *Broadcast Phase.* $\mathcal{CP}$, in order to send a message m, uses an (N, N) threshold scheme: he chooses $N-1$ random elements, $m_1, \ldots, m_{N-1}$, and computes
$$m_N = \bigoplus_{i=1}^{N-1} m_i \oplus m$$
where we assume that $m, m_1, \ldots, m_N$ are elements of a finite field and $\oplus$ denotes the sum operator.
- Then, he broadcasts, for $j = 1, \ldots, N$, the values m_j to the users belonging to $P \subseteq \{1, \ldots, n\}$ by means of the schemes $R(j, f_j(u))$, for every $u \in P$.

Every user in P can recover all the m_j's, and then can compute the message by a simple addition in a finite field. On the other hand, the properties of the hash family guarantee that, for any subset $X = \{i_1, \ldots i_\omega\}$ of users, one of the functions $f_j \in \mathcal{H}$ is one-to-one on X. Hence, the users in X cannot break any of the schemes $R(j, f_j(i_1)), \ldots, R(j, f_j(i_\omega))$ since they are one-resilient and can be broken only if at least two dishonest users are associated with the same scheme, i.e., $f_j(i_k) = f_j(i_\ell)$ for $k \neq \ell$. As a consequence, even if some user in P receives m_j, by means of one of the schemes $R(j, f_j(i_1)), \ldots, R(j, f_j(i_\omega))$, message m_j cannot be recovered by X. Therefore, m cannot be computed by X.

The above scheme can be easily distributed by associating some of the functions in $\mathcal{H}$ with each server. More precisely, an (m, n, n, ω)-DBES can be set up as follows:

- *Set-up Phase - Step* 1: $\mathcal{CP}$-Servers. $\mathcal{CP}$ generates a PHF(N, n, m, ω) and sends the descriptions of N/r of them to each server. Server S_i, for $i = 1, \ldots, r$, gets the functions $f_{(i-1)\frac{N}{r}+1}, \ldots f_{(i-1)\frac{N}{r}+\frac{N}{r}}$.
- *Set-up Phase - Step* 2: Servers-Users. Each server, for $\ell = 1, \ldots, N/r$ and for $j = 1, \ldots, m$, constructs $\frac{N}{r} \times m$ one-resilient schemes $R(\ell, j)$. Then, he sends to every user u, the values associated with the scheme $R(\ell, f_\ell(u))$, for every $\ell = 1, \ldots, N/r$.
- *Broadcast Phase - Step* 1: $\mathcal{CP}$ chooses $N-1$ random elements, $m_1, \ldots, m_{N-1}$, and computes

  $$m_N = \bigoplus_{i=1}^{N-1} m_i \oplus m$$

  Then he sends, for $i = 1, \ldots, r$, the values $m_{(i-1)\frac{N}{r}+1}, \ldots, m_{(i-1)\frac{N}{r}+\frac{N}{r}}$ to server S_i.
- *Broadcast Phase - Step* 2: Each server broadcasts, for $\ell = 1, \ldots, N/r$, the values m_ℓ he holds to the users belonging to $P \subseteq \{1, \ldots, n\}$ by means of the schemes $R(\ell, f_\ell(u))$, for every $u \in P$.

Along the same line, even the multi-level scheme can be distributed among a set of r servers. Each server still receives the description of some functions belonging to the perfect hash family and generates sets of smaller one-level schemes as proposed in [16].

Both the above constructions, the centralized one given in [16] and the distributed one here described, can be enhanced by adding fault tolerance. To this aim, we need the concept of a *generalized* perfect hash function family to set up a fault tolerant (m, n, t, ω)-DBES.

Definition 6. *[34] An α-$PHF(N, n, m, \omega)$ perfect hash family is a $PHF(N, n, m, \omega)$ of functions $\mathcal{H}$ such that, for every subset $X \subseteq \{1, \ldots, n\}$ of size ω, there exists at least α functions $f \in \mathcal{H}$ whose restriction to X is one-to-one.*

Notice that, if $\alpha = N - t + 1$, in any α-generalized perfect hash family, for every ω-subset $X \subseteq \{1, \ldots, n\}$ and *any* t-subset $Y \subset \mathcal{H}$, there exist at *least one* function in Y whose restriction to X is one-to-one. This property is exactly what we need in order to set up a fault tolerant BES (resp. an (m, n, t, ω)-DBES). Indeed, if an α-PHF is used, during the broadcast phase - step 1, instead of splitting m by means of an (N, N)-SSS, m is "split" according to a (t, N)-SSS.

It is easy to see that an α-$PHF(\alpha N, n, m, \omega)$ can be constructed by taking α copies of a $PHF(N, n, m, \omega)$. However, more efficient constructions are possible. For example, by applying the probabilistic method [1], we can prove the following existential result about α-generalized perfect hash functions.

Theorem 1. *Let $\alpha \leq \frac{Nq}{2} + 1$. Then, there exists an α-$PHF(N, n, m, \omega)$ if*

$$N > 8\frac{\omega \log n - \log(\omega!)}{q}.$$

Proof. Assume that the α-$PHF(N, n, m, \omega)$ is represented by means of a matrix A, where each row consists of a function of the family. Let $C \subseteq \{1, \ldots, n\}$ be a subset of size ω of the n columns. It is not difficult to check that, assuming that A is filled in *uniformly at random*, the probability that one of the rows of $A[C]$ has *all different values* is given by

$$q = \frac{m(m-1)(m-2)\cdots(m-\omega-1)}{m^{\omega}}.$$

Therefore, the probability that i rows of $A[C]$ have different values is equal to

$$\binom{N}{i} q^i (1-q)^{N-i}.$$

Let the random variable $X(C)$ be defined as follows:

$$X(C) = \begin{cases} 0, & \text{if more than } \alpha \text{ rows of } A[C] \text{ have all different values} \\ 1, & \text{otherwise.} \end{cases}$$

Moreover, let

$$Y_i = \begin{cases} 0, & \text{if the } i\text{-th row of } A[C] \text{ has all different values} \\ 1, & \text{otherwise,} \end{cases}$$

and let $Y = Y_1 + \ldots + Y_N$. It is not difficult to see that,

$$E[X(C)] = \sum_{i \leq \alpha - 1} \binom{N}{i} q^i (1-q)^{N-i} = P(Y \leq \alpha - 1).$$

A bound on the tail of the binomial probability distribution (see [23], p. 106) establishes that, for any $a \geq 0$,

$$P(Y \leq N(q-a)) \leq e^{\frac{-a^2 N}{2q}}.$$

If we want $N(q-a) = \alpha - 1$ and we set $\alpha - 1 = \frac{qN}{2}$, then $a = \frac{q}{2}$. Therefore, it holds that:

$$E[X(C)] = \sum_{i \leq \alpha - 1} \binom{N}{i} q^i (1-q)^{N-i} \leq e^{\frac{-qN}{8}}.$$

Denoting $X = \sum_{C:|C|=\omega} X(C)$, it holds that

$$E(X) = \binom{n}{\omega} E(X(C)).$$

Hence, we get

$$E(X) \leq \binom{n}{\omega} e^{\frac{-Nq}{8}} \leq \frac{n^\omega}{\omega!} e^{\frac{-Nq}{8}}.$$

The above expected value $E(X)$ is less than 1 if

$$\omega \log n - \log \omega! - \frac{Nq}{8} < 0,$$

which is satisfied if

$$N > 8\frac{[\omega \log n - \log(\omega!)]}{q}.$$

Therefore, the above condition guarantees the existence of an α-$PHF(N, n, m, \omega)$.

5.3 The KIO Construction

A general construction for BES schemes has been proposed in [38]. The idea is to use the Basic Fiat-Naor scheme in conjunction with an ideal secret sharing scheme (ISSS, for short). The goal in [38,39,42] was to obtain schemes where each user has to *store fewer pre-defined keys* and the broadcast messages are *shorter*, compared to other constructions. Using the KIO construction we obtain in our setting a Γ-DBES, where Γ is a specification of subsets of servers (i.e., access structure) that enable a subset of priviliged users to receive the secret message. In other words, each participant i in a priviliged set P uses the values received from a subset of servers belonging to Γ to recover the secret message. In this case the security of $\mathcal{CP}$ with respect to the servers depends on the size of the smallest authorized subset in Γ.

Let $\mathcal{B} = \{B_1, \ldots, B_\beta\}$ be a family of subsets of $\mathcal{U}$, and let ω be an integer. For each $1 \leq j \leq \beta$, suppose a Basic Fiat-Naor scheme $(\leq |B_j|, \leq \omega)$ is constructed with respect to user set B_j. The secret values associated with the j-th scheme will be denoted s_{jC}, where $C \subseteq B_j$ and $|C| \leq \omega$. The value s_{jC} is given to every user in $B_j \setminus C$. Moreover, suppose that $\Gamma \subseteq 2^{\mathcal{B}}$ and there exists a $\Gamma - ISSS$ defined on $\mathcal{B}$ with values in $GF(q)$. Let $\mathcal{F} \subseteq 2^{\mathcal{U}}$, and suppose that

$$\{B_j : i \in B_j\} \in \Gamma \quad \forall i \in \mathcal{U} \qquad \text{and } \{B_j : |F \cap B_j| \geq \omega + 1\} \notin \Gamma \quad \forall F \in \mathcal{F}. \qquad (1)$$

Then, we can construct a $(\leq n, \mathcal{F})$-BES as follows: let $P \subset \mathcal{U}$. $\mathcal{CP}$ can broadcast a message $m_P \in GF(q)$ to P using the following algorithm:

1. For each $B_j \in \mathcal{B}$, $\mathcal{CP}$ computes a share $y_j \in GF(q)$ corresponding to the secret m_P.
2. For each $B_j \in \mathcal{B}$, $\mathcal{CP}$ computes the key k_j corresponding to the set $P \cap B_j$ in the Basic Fiat-Naor scheme implemented on B_j:
$$k_j = \sum_{C \subseteq B_j : C \cap P = \emptyset, |C| \leq \omega} s_{jC}$$
3. For each $B_j \in \mathcal{B}$ $\mathcal{CP}$ computes $b_j = y_j + k_j$.
4. The broadcast is $b_P = (b_j : B_j \in \mathcal{B})$.

The basic idea of the KIO construction can be explained as follows: first, consider a user $i \in P$ and define $A_i = \{j : i \in B_j\}$. User i can compute k_j for every $j \in A_i$. Then, for each $j \in A_i$, i can compute $y_j = b_j - k_j$. Finally, since $A_i \in \Gamma$, i can compute the message m_P from the shares y_j where $j \in A_i$. On the other hand, let $F \in \mathcal{F}$ be such that $F \cap P = \emptyset$. Define

$$A_F = \{j : |F \cap B_j| \geq \omega + 1\}.$$

The coalition F can compute k_j, and hence y_j for every $j \in A_F$. However, they can obtain no information about the shares y_j, where $j \notin A_F$. Since $A_F \notin \Gamma$, F has no information about the value of m_P.

A straightforward application of the KIO construction to the distributed setting can be done as follows: $\mathcal{CP}$ gives the "responsability" of a subset $B_j \in \mathcal{B}$ to every server. Then, every server sets up a $(\leq |B_j|, \leq \omega)$ Basic Fiat-Naor scheme on its subset, and performs the set up phase of such a scheme with the users. Finally, during the broadcast phase, $\mathcal{CP}$ shares the message m_P he wishes to send to the subset P and sends a share to each server; the servers broadcast, according to the same logic of the non-distributed KIO, the values related to users in P.

The KIO construction can be improved by means of a specific set system. Using a suitable design it is possible to achieve *fault tolerance* for both the original KIO construction and the distributed one. To this aim, we introduce the following definition [39]:

Definition 7. *An (n, β, r, λ)-design is a pair $(X, \mathcal{B})$ such that the following properties are satisfied*

1. *$|X| = n$*
2. *$\mathcal{B}$ is a set of β subsets of X called blocks*
3. *every point occurs in exactly r blocks*
4. *every pair of point occurs in* at most *λ blocks.*

If a further property is satisfied we get the following:

Definition 8. *An (n, β, r, λ)-design is called an ω-threshold design provided that, for all $F \subseteq X$ such that $|F| \leq \omega$, we have that*

$$|\{B \in \mathcal{B} : |F \cap B| \geq 2\}| \leq r - 1.$$

Finally, it is possible to show that:

Lemma 1. *[39] An (n, β, r, λ)-design is an ω-threshold design if $r > \lambda\binom{\omega}{2}$.*

Using an ω-threshold design $(X, \mathcal{B})$, and in particular the collection of subsets $\mathcal{B}$, we can set up a fault tolerant BES (resp. (m, n, t, ω)-DBES) scheme. More precisely, at each subset $B_i \in \mathcal{B}$ is associated a $(\leq |B_i|, \leq 1)$ Basic Fiat-Naor scheme, and the secret k is split among the β subsets by means of a (t, β) threshold secret sharing scheme, where $t = \lambda\binom{\omega}{2} + 1 \leq r$.

Notice that t shares allow the secret to be reconstructed, while any coalition of at most ω participants does not possess enough shares (i.e., condition (1) of the KIO construction is satisfied due to Definition 8 and the choice of t). Moreover, if less than $r - t$ shares broadcasted by the servers are lost, then each priviliged participant gets at least t shares that he can decrypt (i.e., $r - t$ is the fault tolerance factor). Finally, it is interesting to point out that, by choosing in an appropriate way the values of the parameters $r \geq t > \lambda\binom{\omega}{2}$, it is possible to achieve a *tradeoff* between security and fault tolerance.

The above instance of the basic KIO construction was given in [39], and an explicit construction of a suitable ω-threshold design by means of universal hash families can be found in subsections 5.3 and 5.4 of that paper. Notice that ω-threshold designs were therein used in order to obtain BES schemes with better information rates compared to the KIO schemes based on other designs (i.e., Balanced Incomplete Block Design).

5.4 Key Distribution Patterns

Another DBES construction can be set up using Key Distribution Patterns. Key Distribution Patterns were introduced by Mitchell and Piper [28]. We briefly recall this concept by following the exposition given in [38].

Let $\mathcal{B} = \{B_1, \ldots, B_\beta\}$ be a set of subsets of $\mathcal{U}$. We say that $(\mathcal{U}, \mathcal{B})$ is a $(\mathcal{P}, \mathcal{F})$-Key Distribution Pattern ($(\mathcal{P}, \mathcal{F})$-KDP, for short) if

$$\{B_j : P \subseteq B_j \text{ and } F \cap B_j = \emptyset\} \neq \emptyset$$

for all $P \in \mathcal{P}$ and $F \in \mathcal{F}$ such that $P \cap F = \emptyset$.

Along the same lines of the Basic Fiat-Naor Scheme, a BES scheme for $(\mathcal{P}, \mathcal{F})$ can be set up as follows:

KDP-based BES.

- *Set up Phase.* For every subset $B_j \in \mathcal{B}$, $\mathcal{CP}$ chooses a random value $s_j \in GF(q)$ and gives s_j to every user in B_j. The key associated with a privileged set P is defined to be

$$\kappa_P = \sum_{j: P \in B_j} s_j.$$

- *Broadcast Phase.* $\mathcal{CP}$ broadcasts the value

$$b_P = \kappa_P + m_P \bmod q.$$

Notice that each user in P can compute m_P. On the other hand, if F is a coalition of users disjoint from P, then there is at least one block B_j such that $P \subseteq B_j$ and $F \cap B_j = \emptyset$. Therefore, F does not hold s_j and cannot compute κ_P.

The above scheme can be easily distributed to set up a $(m, n, t, \mathcal{F})$-DBES, if $\mathcal{CP}$, during the set up phase, gives, for $j = 1, \ldots, \beta$, to server S_j the value s_j, corresponding to the subset B_j, and each server sends this value to all the users in B_j. Then, during the broadcast phase, $\mathcal{CP}$ shares the message m_P among the servers S_j such that $P \subseteq B_j$. Each server, broadcasts $m_P^j + s_j$ where m_P^j is his share for m_P. If t represents the minimum number of supersets $B_j \in \mathcal{B}$ for the subset $P \in \mathcal{P}$, then any $t-1$ servers do not get any information about the message m_P. On the other hand, the number of servers that enable the recovering of the message is in general $\geq t$ and depends on P. Notice that, if for each $P \in \mathcal{P}$, the number of supersets B_j of P is exactly t, then we get an $(m, n, t, \mathcal{F})$-DBES (i.e., $t-1$ servers do not get any information, but t do/reconstruct the message). Fault tolerance depends on the choices of the parameters of the secret sharing scheme (or the ramp scheme) used by $\mathcal{CP}$ to share the message m. Resilient functions also can be used here in order to reduce the length of the broadcast message.

6 Conclusions

In this paper we have considered two aspects of secure broadcast transmission: *reliability* and *trust* in the broadcaster. We have extended some existing broadcast encryption schemes, in order to gain fault tolerance and to remove the need for trust in the broadcaster. These schemes permit a tradeoff between fault tolerance and trust, and can be applied in a variety of broadcast scenarios.

Acknowledgements

D. R. Stinson's research is supported by NSERC grants IRC # 216431-96 and RGPIN # 203114-02.

References

1. N. Alon and J. Spencer, **The Probabilistic Method**, John Wiley, (2nd Edition), 2000.
2. J. Anzai, N. Matsuzaki, and T. Matsumoto, *A Quick Group Key Distribution Scheme with Entity Revocation*, Advances in Cryptology - Asiacrypt '99, Lecture Notes in Computer Science, Vol. 1716, pp. 333-347.
3. O. Berkman, M. Parnas, and J. Sgall, *Efficient Dynamic Traitor Tracing*, Proc. of the 11-th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2000), pp. 586–595, 2000.
4. S. Berkovits, *How to Broadcast a Secret*, Advances in Cryptology - Eurocrypt '91, Lecture Notes in Computer Science, vol. **547**, pp. 536–541, 1991.
5. C. Blundo and A. Cresti, *Space Requirements for Broadcast Encryption*, Advances in Cryptology - Eurocrypt '94, Lecture Notes in Computer Science, vol. 950, pp. 287–298, 1995.
6. C. Blundo, Luiz A. Frota Mattos, and D. R. Stinson, *Generalized Beimel-Chor Schemes for Broadcast Encryption and Interactive Key Distribution*, Theoretical Computer Science, vol. 200, pp. 313–334, 1998.
7. G.R. Blakley and C. Meadows, *Security of Ramp Schemes*, Advances in Cryptology -Crypto '84, Lecture Notes in Computer Science, vol.196, pp. 242-268, 1984.
8. D. Boneh and M. Franklin, *An Efficient Public Key Traitor Scheme*, Advances in Cryptology - Crypto '99, Lecture Notes in Computer Science, vol. 1666, pp. 338–353, 1999.
9. R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas, *Issue in Multicast Security: A Taxonomy and Efficient Constructions*, Infocom '99, pp. 708–716, 1999.
10. R. Canetti, T. Malkin, and K. Nissim, *Efficient Communication-Storage Tradeoffs for Multicast Encryption*, Advances in Cryptology - Eurocrypt '99, Lecture Notes in Computer Science, vol. 1592, pp. 459–474, 1999.
11. B. Chor, A. Fiat, M. Naor and B. Pinkas, *Traitor Tracing*, IEEE Transactions on Information Theory, vol. 46, No. 3, pp. 893–910, May 2000.
12. T. M. Cover and J. A. Thomas, **Elements of Information Theory**, John Wiley & Sons, 1991.
13. G. Di Crescenzo and O. Kornievskaia, *Efficient Multicast Encryption Schemes*, Security in Communication Network (SCN02), Lecture Notes in Computer Science, 2002.
14. C. Dwork, J. Lotspiech, and M. Naor, *Digital Signets: Self-Enforcing Protection of Digital Information*, Proceedings of the 28-th Symposium on the Theory of Computation, pp. 489–498, 1996.
15. P. Erdos, P. Frankl, and Z. Furedi, *Families of finite subsets in which no set is covered by the union of r others*, Israel Journal of Mathematics, N. 51, pp. 75–89, 1985.
16. A. Fiat and M. Naor, *Broadcast Encryption*, Proceedings of Crypto '93, Lecture Notes in Computer Science, vol. 773, pp. 480-491, 1994.
17. A. Fiat and T. Tessa, *Dynamic Traitor Tracing*, Journal of Cryptology, Vol. 14, pp. 211–223, 2001.
18. E. Gafni, J. Staddon, and Y. L. Yin, *Efficient Methods for Integrating Traceability and Broadcast Encryption*, Advances in Cryptology - Crypto '99, Lecture Notes in Computer Science, vol. 1666, p. 372–387, 1999.

19. J. Garay, J. Staddon, and A. Wool, *Long-Lived Broadcast Encryption*, Advances in Cryptology - Crypto 2000, Lecture Notes in Computer Science, vol. 1880, pp. 333–352, 2000.
20. D. Halevy and A. Shamir, *The LSD Broadcast Encryption Scheme*, Advances in Cryptology - Crypto '02, Lecture Notes in Computer Science, vol. 2442, pp. 47-60, 2002.
21. A. Kiayias and M. Yung, *Traitor Tracing with Constant Transmission Rate*, Advances in Cryptology - Eurocrypt '02, Lecture Notes in Computer Science, vol. 2332, pp. 450-465, 2002.
22. A. Kiayias and M. Yung, *Self Protecting Pirates and Black-Box Traitor Tracing*, Advances in Cryptology - Crypto '01, Lecture Notes in Computer Science, vol.2139 , pp. 63-79, 2001.
23. D. E. Knuth, **The Art of Computer Programming**, Addison Wesley, (3rd Edition), 1997.
24. R. Kumar, S. Rajagopalan, and A. Sahai, *Coding Constructions for Blacklisting Problems without Computational Assumptions*, Advances in Cryptology - Crypto '99, Lecture Notes in Computer Science, Vol. 1666, pp. 609–623, 1999.
25. H. Kurnio, R. Safani-Naini, and H. Wang, *A Group Key Distribution Scheme with Decentralised User Join*, Security in Communication Network (SCN02), Lecture Notes in Computer Science, 2002.
26. H. Kurnio, R. Safani-Naini, and H. Wang, *A Secure Re-keying Scheme with Key Recovery Property*, ACISP 2002, Lecture Notes in Computer Science, Vol. 2384, pp. 40–55, 2002.
27. M. Luby and J. Staddon, *Combinatorial Bounds for Broadcast Encryption*, Advances in Cryptology - Eurocrypt '98, Lecture Notes in Computer Science, vol. 1403, pp. 512–526, 1998.
28. C. J. Mitchell and F. C. Piper, *Key Storage in Secure Networks*, Discrete Applied Mathematics, vol. 21, pp. 215–228, 1988.
29. D. Naor, M. Naor, and J. Lotspiech, *Revocation and Tracing Schemes for Stateless Receivers* Advances in Cryptology - Crypto '01, Lecture Notes in Computer Science, vol. 2139, pp. 41–62, 2001.
30. M. Naor and B. Pinkas, *Efficient Trace and Revoke Schemes*, Financial Cryptography 2000, Lecture Notes in Computer Science, vol. 1962, pp. 1–21, 2000.
31. A. Perrig, D. Song, and J. D. Tygar, *ELK, a new Protocol for Efficient Large-Group Key Distribution*, in IEEE Symposium on Security and Privacy (2000).
32. B. Pfitzmann, *Trials of Traced Traitors*, Information Hiding, Lecture Notes in Computer Science, vol. 1174, pp. 49-64, 1996.
33. R. Poovendran and J. S. Baras, *An Information Theoretic Analysis of Rooted-Tree Based Secure Multicast Key Distribution Schemes*, Advances in Cryptology, Crypto '99, vol. 1666, pp. 624-638, 1999.
34. R. Safavi-Naini and H. Wang, *New Constructions for Multicast Re-Keying Schemes Using Perfect Hash Families*, 7th ACM Conference on Computer and Communication Security, ACM Press, pp. 228–234, 2000.
35. R. Safavi-Naini and Y. Wang, *Sequential Traitor Tracing*, Lecture Notes in Computer Science, vol. 1880, p. 316–332, 2000.
36. J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin and D. Dean, *Self-Healing Key Distribution with Revocation*, IEEE Symposium on Security and Privacy, May 12-15, 2002, Berkeley, California.
37. J. N. Staddon, D.R. Stinson and R. Wei, *Combinatorial properties of frameproof and traceability codes*, IEEE Transactions on Information Theory vol. 47, pp. 1042-1049, 2001.

38. D. R. Stinson, *On Some Methods for Unconditionally Secure Key Distribution and Broadcast Encryption*, Designs, Codes and Cryptography, vol. 12, pp. 215–243, 1997.
39. D. R. Stinson and T. van Trung, *Some New Results on Key Distribution Patterns and Broadcast Encryption*, Designs, Codes and Cryptography, vol. 15, pp. 261–279, 1998.
40. D. R. Stinson and R. Wei, *Key preassigned traceability schemes for broadcast encryption*, Proceedings of SAC'98, Lecture Notes in Computer Science, vol. 1556, pp. 144-156, 1999.
41. D. R. Stinson and R. Wei, *Combinatorial properties and constructions of traceability schemes and frameproof codes*, SIAM Journal on Discrete Mathematics, vol. 11, pp. 41–53, 1998.
42. D. R. Stinson and R. Wei, *An Application of Ramp Schemes to Broadcast Encryption*, Information Processing Letters, Vol. 69, pp. 131–135, 1999.
43. D. R. Stinson and R. Wei, *Generalized Cover-Free Families*, preprint.
44. D. M. Wallner, E. J. Harder, and R. C. Agee, *Key Management for Multicast: Issues and Architectures*, Internet Draft (draft-wallner-key-arch-01.txt), ftp://ftp.ieft.org/internet-drafts/draft-wallner-key-arch-01.txt.
45. C. Wong, and S. Lam, *Keystone: A Group Key Management Service*, in International Conference on Telecommunications, ICT 2000.

A Ramp Secret Sharing Schemes

The idea of a ramp secret sharing scheme has been introduced in [7]. More precisely, a ramp secret sharing scheme ((t_1, t_2, n)-RS, for short) is a protocol by means of which a dealer distributes a secret s among a set of n participants $\mathcal{P}$ in such a way that subsets of $\mathcal{P}$ of size greater than or equal to t_2 can reconstruct the value of s; no subset of $\mathcal{P}$ of size less than or equal to t_1 can determine anything about the value of the secret; and a subset of size $t_1 < t < t_2$ can recover *some* information about the secret [7]. Using the entropy function [12], the three properties of a (linear) (t_1, t_2, n)-RS can be stated as follows. Assuming that P denotes both a subset of participants and the set of shares these participants receive from the dealer to share a secret $s \in S$, and denoting the corresponding random variables in bold, it holds that

– *Any subset of participants of size less than or equal to t_1 has no information on the secret value:* Formally, for each subset $P \in \mathcal{P}$ of size $|P| \leq t_1$, $H(\mathbf{S}|\mathbf{P}) = H(\mathbf{S})$.
– *Any subset of participants of size $t_1 < |P| < t_2$ has some information on the secret value:* Formally, for each subset $P \in \mathcal{P}$ of size $t_1 < |P| < t_2$, $H(\mathbf{S}|\mathbf{P}) = \frac{|P|-t_1}{t_2-t_1} H(\mathbf{S})$.
– *Any subset of participants of size greater than t_2 can compute the whole secret:* Formally, for each subset $P \in \mathcal{P}$ of size $|P| \geq t_2$, $H(\mathbf{S}|\mathbf{P}) = 0$.