Fourteenth International Workshop on Algebraic and Combinatorial Coding Theory
September 7–13, 2014, Svetlogorsk (Kaliningrad region), Russia *pp. 324–329*

Broadcast Encryption Scheme Based on Binary Cubes

Alexey V. Urivskiy alexey.urivskiy@mail.ru, urivskiy@infotecs.ru
JSC "InfoTeCS", Moscow, Russia

Abstract. A problem of Broadcast Encryption is concerned. We propose a symmetric secret key BE-scheme, called CuBES, which is based on binary cubes as a combinatorial underlying structure. We estimated the main parameters of the scheme and compared it other known ones.

1 Introduction

A Broadcast Encryption Scheme (BES) [1] enables a center to deliver encrypted data to a particular subset of privileged users listening to a broadcast channel. Accordingly, any subset of nonprivileged users – the *revoked users* — are not able to decrypt. Typical applications for BESs are pay-TV systems, tactical radio, positioning systems, digital rights management solutions, etc. The most interesting variant of the BE-problem deals with stateless receivers not capable of recording the past history of transmissions and change its state, including the keys.

In symmetric BESs, which we solely discuss further, the data is typically encrypted on a short-term session secret key K, and K is in turn encrypted on long-term secret *key encryption keys* (KEKs) known to the users. Along with the encrypted data and session key the center distributes a so called ID-header which unambiguously identifies which users are in which subset.

The encrypted K and the ID-header forms transmission overhead. One of the main problem of designing a good BES is to balance the number of KEKs belonging to a user (the *user key block*) and the transmission overhead given the number of the revoked users. It should be noted that BESs using elliptic curves (pairing-based) proposed recently [2] might be more efficient in terms of user key block and transmission overhead. However, their security relies heavily on computational assumptions, and the main operations are resource consuming.

A lot of BESs were published in a last couple of decades. In this paper, we propose a BES based on combinatorial construction — binary cubes. This BES offers information-theoretic security if KEKs are generated randomly and independently. For a given size of the network and the number of the revoked users we estimated the upper bound on the transmission overhead. We called this scheme *CuBES* for binary-*Cu*bes-based *B*roadcast *E*ncryption *S*cheme [3].

2 New BES

2.1 Incidence matrix

Any BES deals with key distribution. In a symmetric BES, a user receives a subset of keys from a larger set. Let there be at most N users in the network, and there be K keys held by the center. The users are labeled by index j, $j = 1, \ldots, N$, and the keys are labeled by index i, $i = 1, \ldots, K$.

Form a binary $k \times n$ *incidence matrix* $\mathbf{A} = [a_{ij}]$: If user j possesses i-th key then $a_{ij} = 1$, otherwise $a_{ij} = 0$.

If some users should be revoked, the columns of $\mathbf{A}$ corresponding to them must be discarded as well as all rows incident to these columns. Next the center should find a set of rows incident to the remaining columns, and use the corresponding keys to broadcast the message. There may be several sets of keys (rows) that cover the privileged users. Any set will do, though a set of the smallest size, which we will call a *coverage*, minimizes transmission overhead.

2.2 n-dimensional binary cubes

Consider binary n-vectors $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. There are exactly 2^n such vectors. A union of all these vectors is called *n-dimensional binary cube.*

Let every vector of n-dimensional binary cube, except the all-0 vector, be put as a row of an incidence matrix.

Lemma 1. *The incidence matrix induced by n-dimensional binary cube gives a BES for a network of n users with the smallest possible coverage of size* 1.

Proof. For any r let users $i_1, i_2, \ldots, i_r$ be revoked. The row, where 0 are at positions $i_1, \ldots, i_r$ and ones are at all the other positions, gives the coverage. $\square$

2.3 Logical hierarchy

Binary cubes define the best possible BES from the transmission overhead point of view. However, the user key block includes $2^N - 1$ keys which is huge even for small N. To limit the dimensions of the cubes we introduce a logical hierarchy over users. This hierarchy is organized as a tree as follows.

At the lowest level make groups of n_1 users, and establish for every group a BES based on n_1-dimensional binary cubes. There will be N/n_1 groups.

Next, we consider every group as a new group-user. We join every n_2 of these new users in a higher level group, and establish for each such group a BES based on n_2-dimensional binary cubes. An so forth. So we obtain a tree of ℓ levels, where

$$N = n_1 \cdot n_2 \cdot \ldots \cdot n_\ell. \tag{1}$$

The user key block will comprise $\sum_{i=1}^{\ell} 2^{n_i - 1}$ keys. In fact, a user needs a bit smaller number of keys. At two consecutive levels, say i and $i+1$, there will be two keys, which both unique to the same group of users: to all users at level i, and to one group-user at level $i+1$. For the BES to work, it suffices to have only one of those keys. So we left the key at level $i+1$, and discard the one from level i.

At the top of our tree we add a key which will belong to every user. This key is useful to broadcast a message if there are no revoked users. Summarizing, for the network of N users we have a BES such that every user holds

$$V = \sum_{i=1}^{\ell}(2^{n_i-1} - 1) + 1 \tag{2}$$

KEKs. The storage at the center is all keys from all binary cubes. There are

$$1 + \sum_{i=1}^{\ell} \frac{N}{\prod_{j=1}^{i} n_j} \left(2^{n_i-1} - 1\right) \tag{3}$$

keys in total. However, in real-world applications the storage at the center is of much less importance than the user key block.

2.4 ID-header and session key recovery

Due to lack of space we do not describe two pieces of information important to a BES. First, it is how the ID-header is constructed. Second, the information, which shows the order in which KEKs in the coverage are used to encrypt the session key.

For symmetric BESs it is typical that some additional information is used to help the privileged user to infer what particular KEKs were used. This depends on a particular ID-header and KEKs ordering. Also sometimes to decrypt the session key trial decryptions are required. We skip these both topics either.

3 Transmission Overhead

3.1 Finding a coverage

How can the center find a coverage for a given configuration of r revoked users? If there are no users revoked, then use the special key defined for the whole network. When at least one user is revoked, the process starts from the top level of the tree. At this level there are n_ℓ groups. First the center finds those groups from which none of the users are revoked. This set of groups is given one key from the top level. For the other groups go one level down.

Every group from which at least one user is revoked is split into $n_{\ell-1}$ smaller groups. From these $n_{\ell-1}$ groups find the ones from which none of the users is revoked, and add one key (for each ℓ-level group), corresponding to that groups, to the coverage. For the other groups go one level down. And so forth.

At the final stage we consider groups of n_1 users from which one or more users were revoked. The revoked users do not get any keys, and for all the others one key (once again for each group) is added to the coverage.

It is readily seen that in the coverage there is exactly one key belonging to a privileged user. When finding a coverage the center checks at most all

$$W = \sum_{i=1}^{\ell} \frac{N}{\prod_{j=1}^{i} n_j} \tag{4}$$

binary cubes. So the time- and operational complexities are $O(W)$ units.

3.2 Size of coverage

Let us estimate how many KEKs will be in a coverage. We give a worst case analysis meaning we look at worst-case configurations of the revoked users giving the largest coverage. Next without a proof we give an upper bound the size of a coverage as function of the number r of revoked users.

Suppose there are ℓ levels in the tree. When one user is revoked, the coverage contains ℓ keys. When a second revoked user is in the same highest level branch of the tree, then at most $\ell-2$ KEKs are added to the coverage. When the second revoked user is from another branch, then exactly $\ell - 1$ KEKs are added.

Suppose now that n_ℓ users are revoked and it is the worst-case configuration affecting all the highest level branches. What will be the worst case if one more user is revoked? Evidently this is when this revoked user is in the branches of $\ell-1, \ell-2, \dots, 1$ levels that are different from branches to which n_ℓ previously revoked users belong to. Thus, the worst case happens when r revoked users belong to as many as possible different branches of the hierarchical tree.

Lemma 2. *Let a decomposition of N be given by (1). Then for a given number of revoked users r the upper bound L on size of a coverage is a piece-wise linear function in r with the following $\ell + 2$ breaking points.*

The start point is $r_0 = 0, \quad L_0 = 1.$

The first point is $r_1 = n_\ell - 1, \quad L_1 = (\ell-1)n_\ell - (\ell-2).$

The second point is $r_2 = (n_{\ell-1} - 1)n_\ell, \quad L_2 = \big((\ell-2)n_{\ell-1} - (\ell-3)\big)n_\ell.$

The i-th point is $r_i = (n_{\ell-i+1} - 1)n_{\ell-i+2} \cdot \ldots \cdot n_\ell,$

$L_i = \big((\ell-i)n_{\ell-i+1} - (\ell-i-1)\big)n_{\ell-i+2} \cdot \ldots \cdot n_\ell.$

The $(\ell-1)$-th point is $r_{\ell-1} = (n_2 - 1)n_3 n_4 \ldots n_\ell, \quad L_{\ell-1} = n_2 n_3 \ldots n_\ell.$

The ℓ-th point is $r_\ell = (n_1 - 1)n_2 n_3 \ldots n_\ell, \quad L_\ell = n_2 n_3 \ldots n_\ell.$

And the final $(\ell+1)$-th point is $r_{\ell+1} = n_1 n_2 \ldots n_\ell, \quad L_{\ell+1} = 0.$

Proof (sketch). The proof can be obtained from the configurations of the worst case, and the fact the coverage for a BES on a binary cube is always of size 1. □

4 Example and Comparison to Known Schemes

Numerical example. Let $N = 2^{20}$, $\ell = 9$, $n_1 = n_2 = 8$, $n_3 = \ldots = n_9 = 4$.

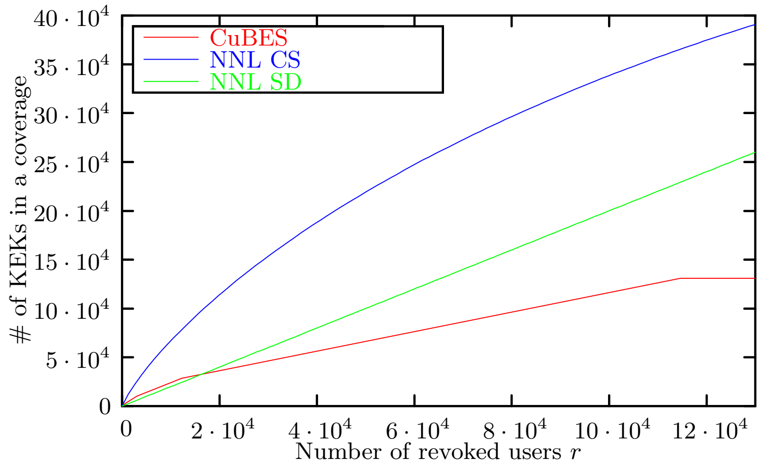
The breaking points in the upper bound on the coverage in CuBES, written as $\{i : (r_i, L_i)\}$, are $\{0 : (0, 1)\}$, $\{1 : (3, 25)\}$, $\{2 : (12, 88)\}$, $\{3 : (48, 304)\}$, $\{4 : (192, 1024)\}$, $\{5 : (768, 3328)\}$, $\{6 : (3072, 10240)\}$, $\{7 : (12288, 28672)\}$, $\{8 : (114688, 131072)\}$, $\{9 : (917504, 131072)\}$, $\{10 : (1048576, 0)\}$.

According to (2) the user key block includes $V = 304$ KEKs. According to (3) the storage at the center is 2^{24} KEKs, or 16 keys per user. According to (4) to find a coverage the center checks at most all $W = 2^{17}$ binary cubes.

Comparison. We compare the CuBES to other BESs described in [4], called NNL Complete Subtree and NNL Subset Difference. The LSD scheme [5] is a modification of NNL Subset Difference with the number of KEKs increased 2 times, and smaller user key block.

We continue with the values from the numerical example. The size of the user key block in NNL CS is 21 KEKs, and in NNL SD is 211 KEKs.

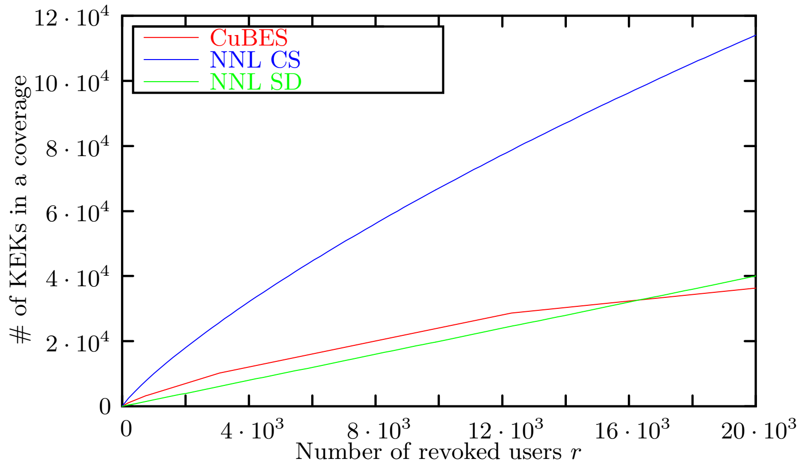
For these three BESs we depicted the upper bound on the size of a coverage (the number of KEKs) as a function of the value r. There are two figures. The first one shows the overall dependence of the number of KEKs relative to r. The second one shows the region of small r.



In CuBES when $r \sim O(N)$, the number of KEKs is constant and is N/n_1. For NNL CS this number is $r \log \frac{N}{r}$. For NNL SD this number is upperbounded by $2r$, though this bound is too rough, and the real growth is slower.

Asano's BES. The basic part of the BES due to Asano [6] is very close to CuBES. Asano also suggests using binary cubes (or 'power sets'). It should be noted that in [6] user key block is computed from user's personal master key. However, we do not consider security and complexity issues of it.

An important difference is that in Asano's scheme all cubes are of the same dimension. So CuBES is a generalization of the combinatorial part of Asano's BES. Meanwhile, as we see from Lemma 2 the limit on the size of the coverage depends only on n_1: given the size of the user key block, we should maximize n_1. Also, comparing to [6] we improved the upper bound on the size of the coverage.



References

[1] A. Fiat, M. Naor, "Broadcast Encryption", CRYPTO 1993, *LNCS* 773, pp. 480–491, 1993.

[2] D. Boneh, C. Gentry, B. Waters, "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys", CRYPTO 2005, *LNCS* 3621, pp. 258–275, 2005.

[3] A. Urivskiy, A. Chmora, "Method of managing a key of user for broadcast encryption," patent US 7,774,598, US PTO, 2010.

[4] D. Naor, M. Naor, J. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers", CRYPTO 2001, *LNCS* 2139, pp. 41–62, 2001.

[5] D. Halevy, A. Shamir, "The LSD Broadcast Encryption Scheme", CRYPTO 2002, *LNCS* 2442, pp. 47–60, 2002.

[6] T. Asano, "A Revocation Scheme with Minimal Storage at Receivers", ASIACRYPT 2002, *LNCS* 2501, pp. 433-450, 2002.