Tracing Traitors

Benny Chor[1] and Amos Fiat[2] and Moni Naor[3]

[1] Dept. of Computer Science, Technion, Haifa 32000, Israel.
[2] Dept. of Computer Science, School of Mathematics, Tel Aviv University, Tel Aviv, Israel, and Algorithmic Research Ltd.
[3] Dept. of Computer Science and Applied Math, Weizmann Institute, Rehovot, Israel.

Abstract. We give cryptographic schemes that help trace the source of leaks when sensitive or proprietary data is made available to a large set of parties. This is particularly important for broadcast and database access systems, where the data should be accessible only to authorized users. Such schemes are very relevant in the context of pay television, and easily combine with and complement the Broadcast Encryption schemes of [6].

1 Introduction

If only one person is told about some secret, and this next appears on the evening news, then the guilty party is evident. A more complex situation arises if the set of people that have access to the secret is large. The problem of determining guilt or innocence is (mathematically) insurmountable if all people get the exact same data and one of them behaves treacherously and reveals the secret.

Any data that is to be available to some while it should not be available to others can obviously be protected by encryption. The *data supplier* may give authorized parties cryptographic keys allowing them to decrypt the data. This does not solve the problem above because it does not prevent one of those authorized to view the message (say, Alice) from transferring the *cleartext* message to some unauthorized party (say, Bob). Once this is done then there is no (cryptographic) means to trace the source of the leak. We call all such unauthorized access to data *piracy*. The *traitor* or *traitors* is the (set of) authorized user(s) who allow other, non-authorized parties, to obtain the data. These non-authorized parties are called *pirate users*.

In many interesting cases it is somewhat ineffective piracy if the relevant *cleartext* messages must be transmitted by the "traitor" to the "enemy". Typical cases where this is so include

- Pay-per-view or subscription television broadcasts. It is simply too expensive and risky to start a pirate broadcast station.
- CD ROM distribution of data where a surcharge is charged for different parts of the data. The cleartext data can only be distributed on a similar storage device.
- Online databases, freely accessible (say on the internet) where a charge may be levied for access to all or certain records.

Y.G. Desmedt (Ed.): Advances in Cryptology - CRYPTO '94, LNCS 839, pp. 257-270, 1994.
© Springer-Verlag Berlin Heidelberg 1994

In all these cases, transmitting the cleartext from a traitor, Alice, to an pirate-user, Bob, is either irrelevant or rather expensive. As piracy in all these cases is a criminal commercial enterprise the risk/benefit ratio becomes very unattractive. These three examples can be considered generic examples covering a wide range of data services offered.

Our contribution in this paper may be viewed in the following manner: Consider a ciphertext that may be decrypted by a large set of parties, but each and every party is assigned a different *personal key* used for decrypting the ciphertext. (We use the term personal key rather than private key to avoid confusion with public key terminology). Should the personal key be discovered (by taking apart a television pirate decoder or by counter-espionage), the traitor will be identified.

We note that in fact, our schemes have the very desirable property that the identity of the traitor can be established by considering the pirate decryption process as a black box. It suffices to capture one pirate decoder and it's behavior will identify the traitor, there is no need to "break it open" or read any data stored inside. We use the term pirate decoder to represent the pirate decryption process, this may or may not be a physical box, this may simply be be some code on a computer.

Clearly, a possible solution is to encrypt the data separately under different personal keys. This means that the total length of the ciphertext is at least n times the length of the cleartext, where n is the number of authorized parties. This is clearly impossible in any broadcast environment. This is also very problematic in the context of CD ROM distributed databases because this means that every CD ROM must be different. An encrypted online database, freely accessible as above, must store an individually encrypted copy of the database for each and every authorized user.

The underlying security assumption of our schemes is either information theoretic security (where the length of the personal keys grows with the length of the messages to be transmitted) or it may be based on the security of any symmetric scheme of your choice. In both cases, security depends on a scheme parameter k, the largest group of colliding traitors.

In practice today it is often considered sufficient to prevent piracy by supplying the authorized parties with so-called secure hardware solutions that are designed to prevent interference and access to enclosed cryptographic keys (smart-cards and their like). Our schemes do not require any such assumption, they obtain their claimed security without any secure hardware requirements. Should such devices be used to store the keys they will undoubtedly make the attack more expensive, but this is not a requirement.

Fighting piracy in general has the following components:

1. Identify that piracy is going on and prevent the transmittal of information to pirate users, while harming no legitimate users.
2. Take legal measures against the source of such piracy, supply legal evidence of the pirate identity.

Any solution to fighting piracy must be considered in light of the following performance parameters:

(a) What are the memory and computation requirements per authorized user?
(b) What are the memory and computation requirements for the data supplier?
(c) What is the data redundancy overhead? This is measured in multiples of the the cryptographic security parameter and refers to the communications overhead (in broadcast or online systems) or the additional "wasted" storage in CD ROM type systems.

Consider a pirate user who has already obtained all keys required to read a CD ROM in it's entirety. Clearly, there is little one can do technically to prevent her from continuing to use the CD ROM. The situation is somewhat different if the system requires some action on behalf of the data supplier, *e.g.*, television broadcast or online database.

The broadcast encryption scheme of Fiat and Naor [6] deals with disabling active pirate users very efficiently. These schemes allow one to broadcast messages to any dynamic subset of the user set, this is specifically suitable for pay-per-view TV applications but also implies the piracy protection above. These schemes require a single short transmission to disable all pirate decoders if they were manufactured via a collaborative effort of no more than k traitors.

The number of traitors above, k, is a parameter of the broadcast encryption schemes. While this may not be evident at first, the same scheme could be used by any online database supplier to kill off illegitimate access simply by telling users who log on what users are currently blacklisted.

The goal of this paper is to deal *traitor tracing* (item 2 above), *i.e.*, to identify the source of the problem and to deal with it via legal or extra-legal means. Our solution, called traitor tracing, is valid for all examples cited above, broadcast, online, and CD ROM type systems.

We devise *k-resilient traceability* schemes with the following properties:

1. Either the cleartext information itself is continuously transmitted to the enemy by a traitor, or
2. Any captured pirate decoder will correctly identify a traitor and will protect the innocent even if up to k traitors combine and collude.

It would make sense to have both broadcast encryption and traitor tracing schemes available, at different security levels. The costs of such schemes are measured in the memory requirements at the user end and in the total transmission length required. In practice one would want a broadcast encryption scheme with a different security level (measured in the numbers of traitors required to disable the scheme). Fortunately, both types of scheme, at arbitrary security levels, can be trivially combined simply by XOR'ing the results.

We deal with schemes of the following general form: The data supplier generates a base set R of r random keys and assigns subsets of these keys to users, m keys per user (these parameters will be specified later). These m keys jointly form the user personal key. Note that different personal keys may have a nonempty

intersection. We denote the personal key for user u by $P(u)$, this is a set of keys over the base set R.

A traitor tracing message consists of many pairs of (*enabling block*, *cipher block*). The cipher block is the symmetric encryption of the actual data (say a few seconds of a video clip), under some secret random key S. Alternately, it could simply be the XOR of the message with S and we would get an information theoretic secure version of the scheme. The enabling block allows authorized users to obtain S. The enabling block consists of encrypted values under some or all of the r keys at the data supplier. Every user will be able to compute S by decrypting the values for which he has keys and then computing the actual key from these values. The computation on the user end, for all schemes we present, is simply the exclusive or of all values the user has been able to decrypt.

Figures 1 and 2 describe the general nature of our traitor tracing schemes.

Traitors may conspire and give an unauthorized user (or users) a subset of their keys so that the unauthorized user will also be able to compute the real message key from the values he has been able to decrypt. The goal of the system designer is to assign keys to the users such that when a pirate decoder is captured and the keys it possesses are examined, it should be possible to detect at least one traitor, subject to the limitation that the number of traitors of is at most k. (We cannot hope to detect all traitors as one traitor may simply provide his personal key and others may provide nothing).

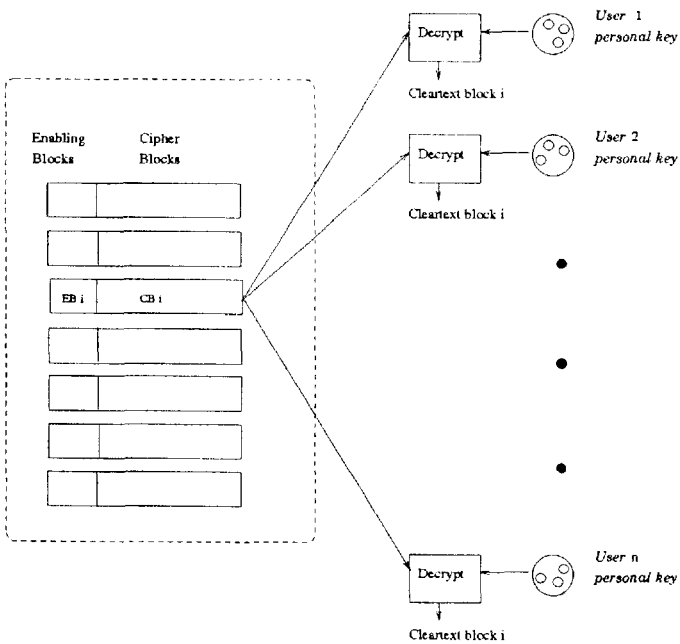

Fig. 1.

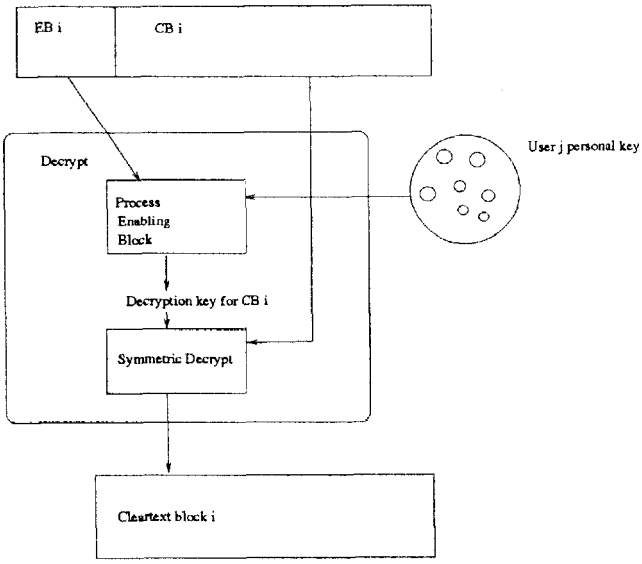

Fig. 2.

We remark that in many cases it is preferable to predetermine a fixed number of users n, and to assign them personal keys, even if the actual number of users is smaller. Later users who join the system by purchasing a subscription to a television station, online database, or CD ROM access privilege are assigned personal keys from those preinstalled. This is especially important in the case of data distributed on CD ROM.

1.1 An Example

Using the 1-level secret scheme described hereinafter, allocating 5% of a compressed MPEG II digital video channel to the traitor tracing scheme allows us to change keys every minute or so (a new enabling block every minute).

The traitor tracing scheme is resilient to $k = 32$ traitors, with probability $1 - 2^{-10}$, and can accommodate up to 1,000,000,000 authorized users. The total number of keys stored by the data supplier (the television broadcaster) is 2^{19}, the personal key of every user consists of 2^{13} keys. These parameters are overly pessimistic because they are derived from the general theorem concerning the scheme using the Chernoff bound.

In practice, there is no real need to change keys every minute, even changing keys once every hour will make any pirate broadcaster give up in despair.

2 Definitions

For messages generated by a data supplier for a set of n users, we define three elements that jointly constitute a *traceability scheme*:

- A *user initialization scheme*, used by the data supplier to add new users. The data supplier supplies user u_i with her personal key, in our case this consists of a set $P(u_i)$ containing decryption keys.
- A *decryption scheme*, used by every user to decrypt messages generated by the data supplier. In our schemes, the messages are decrypted block by block where every block decryption consists of a preliminary decryption of encrypted keys in the enabling block, combining the results to obtain a common key, followed by a decryption of the cipher block.
- A *traitor tracing algorithm*, used upon confiscation of a pirate decoder, to determine the identity of a traitor. We assume below that the contents of a pirate decoder can be viewed by the traitor tracing algorithm.

We distinguish between circumstances where the decryption schemes used by all users are in the public domain, whereas the decryption keys themselves are kept secret, called *open schemes*, versus the case where the actual decryption scheme as well as the keys are kept secret, called *secret schemes*.

The goal of an adversary is to construct a pirate decoder that allows decryption and prevents the guilty from being identified. In particular, one way to ensure that the guilty are safe is to try to incriminate someone else. Clearly, the adversaries task is no harder with an open scheme compared to a secret scheme. On the other hand, secret schemes pose additional security requirements at the data supplier cite and the correctness of the traitor identity may be based on probabilistic arguments, which may be somewhat less convincing in a court of law.

We present efficient schemes of both types, and our constructions give better results for secret schemes. It is clearly advantageous to use secret schemes in practice, and any real implementation will do so.

To simplify the definitions below we will assume that it is impossible to guess a secret key. The probability of guessing a secret key is exponentially small in the length of the key, and thus we will ignore this question in the definitions below. An alternative would be to talk about probability differences rather than absolute probabilities, this is done in [6] and analogous definitions could be used here.

Definition 1. An n user open traceability scheme is called *k-resilient* if for every coalition of at most k traitors the following holds: Suppose the coalition uses the information its members got in the initialization phase to construct a pirate decoder. If this decoder is capable of applying the decryption scheme, then the traitor tracing algorithm will correctly identify one of the coalition members.

Definition 2. An n user secret traceability scheme is called *(p, k)-resilient* if for all but at most p of the $\binom{n}{k}$ coalition of k traitors the following holds: Suppose the coalition uses the information its members got in the initialization phase to construct a pirate decoder. If this decoder is capable of applying the decryption scheme, then the traitor tracing algorithm will identify one of the coalition members with probability at least $1 - p$.

3 Construction of Traceability Schemes

In this section we describe three constructions of k-resilient traceability schemes. All these schemes are based on the use of hash functions, combined with *any* private key cryptosystem. (For more information on hash functions and their applications, see [7, 3, 9, 5].) The basic use of hash functions is to assign decryption keys to authorized users in a manner which prevents any coalition of traitors from combining keys taken from the personal keys of its members into a set of keys that allows decryption yet is "close" to the personal key of any innocent user.

The first scheme is the simplest one. It is an open scheme, based on "one level" hash functions. Each hash function maps the n users into a set of $2k^2$ decryption keys. The keys themselves are kept secret, but the mapping (which user is mapped to what key) is publicly known. This is a simple scheme, but its performance can be improved upon: Every user personal key consists of $O(k^2 \log n)$ decryption keys, and the enabling block consists of $O(k^4 \log n)$ encrypted keys.

The second scheme is an open "two level" scheme. Here, a set of first level hash functions map the n users into a set of size k. each function thereby induces a partition of the n users to k subsets. Each of these subsets is mapped separately by "second level" hash functions into $\log^2 k$ decryption keys. This scheme requires $O(k^2 \log^2 k \log n)$ keys per user, and an enabling block of $O(k^3 \log^4 k \log n)$ encrypted keys.

The third scheme is a "one level" secret scheme. Here, we assume that the hash functions, as well as the decryption keys, are kept secret. There is a positive probability p $(0 < p < 1)$ that the adversary will be able to produce pirate decoders which prevent the identification of any traitor.

However, even if the keys known to the k collaborators enable the construction of such "wrongly incriminating" pirate decoders, choosing such set is highly improbable. Even if this unlikely event occurs, the adversary will not know that this is the case.

Being a secret scheme implies that the adversary does not know what keys corresponds to any specific user. The personal key consists of $O(k \log(n/p))$ decryption keys, and has $O(k^2 \log(n/p))$ encrypted keys per enabling block.

All schemes are constructed by choosing hash values at random, and using probabilistic arguments to assert that the desired properties hold with overwhelming probability. Therefore, these schemes are not constructive. However, the properties of the simplest scheme can be verified.

3.1 A Simple Scheme

Let k be an upper bound on the number of traitors. Every enabling block consists of r encryptions, and m denotes the number of keys comprising a user personal key.

We first deal with the case $k = 1$. The data supplier generates $r = 2 \log n$ keys $s_1^0, s_1^1, s_2^0, s_2^1, \ldots, s_{\log n}^0, s_{\log n}^1$. The personal key for user i is the set of $m = \log n$ keys $s_1^{b_1}, s_2^{b_2}, \ldots, s_{\log n}^{b_{\log n}}$, where b_i is the ith bit in the ID of u.

To encrypt a secret s, the data supplier splits it into $\log n$ secrets $s_1, s_2, \ldots, s_{\log n}$, *i.e.*, the data supplier chooses random $s_1, s_2, \ldots, s_{\log n}$ such that q is the bitwise XOR of the s_i's (its j-th bit equals $s^{(j)} = XOR_{i=1}^{\ell} s_i^{(j)}$). The value s_i is encrypted using keys s_i^0 and s_i^1 and both encryptions are added to the enabling block. Every user u can reconstruct all the s_i's and hence can decrypt s. On the other hand, any pirate decoder must contain a key for every $1 \leq i \leq \log n$ (otherwise s_i would remain unknown and consequently s could not be obtained).

Thus, given that at most one traitor is involved, the keys stored in the pirate decoder uniquely identify the traitor.

When dealing with larger coalition, the idea is to generalize the above scheme. Instead of one bit per index we will have larger domains (and have a key for every element in the domain). We will also split s into more than $\log n$ parts and have appropriately more indices or hash functions. The major difficulty we encounter is in the procedure for detecting traitors. Since, unlike the case $k = 1$, keys may be mixed from several members of the coalition, we must make sure that the two users are not only different on some indices, but are different in almost all indices. A detailed description of the scheme is given below.

Initialization: A set of ℓ "first level" hash functions $h_1, h_2, \ldots, h_\ell$ is chosen at random by the data supplier. Each hash function h_i maps $\{1, \ldots, n\}$ into an independent set of $2k^2$ random keys, $S_i = \{s_{i,1}, s_{i,2}, \ldots, s_{i,2k^2}\}$. The personal key for user u is the set $P(u) = \ell$ keys $\{h_1(u), h_2(u), \ldots, h_\ell(u)\}$.

Distributing a Key: For each i ($i = 1, 2, \ldots, \ell$) the data supplier encrypts a key s_i under each of the $2k^2$ keys in S_i. The final key s is the bitwise XOR of the s_i's (its j-th bit equals $s^{(j)} = XOR_{i=1}^{\ell} s_i^{(j)}$). Each authorized user has one key from S_i, so he can decrypt every s_i, and thus compute s.

Parameters: The memory required per user is $m = \ell$ keys. An enabling block to encode a secret value s consists of $= 2k^2\ell$ key encryptions.

Fraud: The k traitors can get together and combine their personal keys. They may choose one key from every set S_i ($i = 1, 2, \ldots, \ell$). These ℓ keys are put together in a pirate decoder. This set of keys F, enables the purchaser of such a decoder to decrypt every s_i, and thus compute s.

Detection of Traitors: Upon confiscation of a pirate decoder, the set of keys in it, F, is exposed. The set F must contain at least ℓ keys (at least one key per set S_i). Denote the key with minimum index in S_i by $f_i \in S_i$. For each i, the users in $h_i^{-1}(f_i)$ are identified and marked. The user with largest number of marks is exposed.

Goal: We want to show that for all coalitions of size k, the probability of exposing a user who is not a traitor is negligible.

Clearly, at least one of the traitors contributes at least ℓ/k of the keys to the pirate decoder (we ignore duplicate keys from the same S_i). We want to show that the probability (over all choices of hash functions) that a good user is marked ℓ/k times is negligible. Consider a specific user, say 1, and a specific coalition T of k traitors (which does not include 1). As hash functions are chosen at random, the value $a_i = f_i(1)$ is uniformly distributed in S_i. The coalition gets at most k keys in S_i. The probability that a_i is among these keys is at most $1/2k$.

Let X_i be a zero-one random variable, where $X_i = 1$ if $a_i \in h_i(T)$. The mean value of $\sum_{i=1}^{\ell} X_i$ is $\ell/2k$. We use the following version of Chernoff bound (see [2], Theorem A.12) to bound the probability that $\sum_{i=1}^{\ell} X_i \geq \ell/k$. Let $X_1, \ldots, X_\ell$ be mutually independent random variables, with

$$Pr(X_j = 1) = p$$
$$Pr(X_j = 0) = 1 - p$$

Then, for all $\beta \geq 1$

$$Pr\left(\frac{1}{\ell}\sum_{j=1}^{\ell} X_j \geq \beta p\right) < \left(\frac{e^{\beta-1}}{\beta^{\beta}}\right)^{p\ell}.$$

In our case, substituting $p = 1/2k$ and $\beta = 2$, we have

$$Pr\left(\frac{1}{\ell}\sum_{i=1}^{\ell} X_i \geq \frac{1}{k}\right) < \left(\frac{e}{4}\right)^{\ell/2k} < 2^{-\ell/4k}.$$

In order to overcome all $\binom{n}{k}$ coalitions and all n choices of users, we choose ℓ satisfying

$$n \cdot \binom{n}{k} \cdot 2^{-\ell/4k} < 1,$$

namely $\ell > 4k^2 \log n$. With this parameter, there is a choice of ℓ hash functions such that for every coalition and every authorized user not in the coalition, the user is not incriminated by the tracing algorithm. We summarize these results in the next theorem:

Theorem 3. *There is an open k-resilient traceability scheme, where a user personal key consists of $m = 4k^2 \log n$ decryption keys, and an enabling block consists of $r = 8k^4 \log n$ key encryptions.*

Explicit Constructions: The discussion above shows the existence of open k resilient traceability schemes, and does provide us with a randomized method for constructing the scheme that works with high probability. It does not, however, suggests an explicit construction. Note however that a given construction can be verified quite efficiently. The idea is to examine all the pairs of elements u, v and check the number of function h_i such that $h_i(v) = h_j(u)$. If this number is smaller than ℓ/k^2 than we can conclude that no coalition T of at most k elements "covers" more than a $1/k$ fraction of the keys of u and hence cannot incriminate u.

By considering pairwise differences, we can phrase the construction problem as a problem in coding theory (see [8] for more information): construct a code with n codewords over an alphabet of size $2k^2$ of length ℓ such that the distance between every two codewords is at least $\ell - \ell/k^2$. The goal is construct such a code with as small ℓ as possible. There are no known explicit construction that match the probabilistic bound. For the best known construction see [1] and references therein. For small k the constructions there yields a scheme with $m \in O(k^6 \log n)$ and $r \in O(k^8 \log n)$.

3.2 An Open Two Level Scheme

The "two level" traceability scheme, described in this subsection, more complicated than the simple scheme, but it saves about a factor of k in the broadcast overhead.

Theorem 4. *There is an open k-resilient traceability scheme, where a user personal key consists of $m = 2k^2 \log^2 k \log n$ decryption keys, and an enabling block consists of $r = 4k^3 \log^4 k \log n$ key encryptions.*

Proof. We describe the system, step by step. As in the one-level scheme, the proof is existential. We do not know however how to verify efficiently that a given scheme is "good".

Initialization: A set of ℓ "first level" hash functions $h_1, h_2, \ldots, h_\ell$, each mapping $\{1, \ldots, n\}$ to $\{1, \ldots, k\}$, is chosen at random. For each i ($i = 1, 2, \ldots, \ell$) and each element a in $\{1, \ldots, k\}$, a set of d "second level" hash functions $g_{i,a,1}, \ldots, g_{i,a,d}$ is chosen at random. Each second level function $g_{i,a,j}$ maps the users in $h_i^{-1}(a) \subset \{1, \ldots, n\}$ into a set of $4 \log^2 k$ random independent keys (the ranges of different functions are independent).

Each user $u \in \{1, \ldots, n\}$ receives $\ell \cdot d$ keys

$$\begin{array}{ccc} g_{1,h_1(u),1}(u), & \cdots & , g_{1,h_1(u),d}(u) \\ \vdots & & \vdots \\ g_{\ell,h_\ell(u),1}(u) & , \ldots , & g_{\ell,h_\ell(u),d}(u) \end{array}$$

Distributing a Key: The data supplier chooses at random ℓ independent keys $s_1, \ldots, s_\ell$. The final key is $s = BITWISE - XOR_{i=1}^{\ell}(s_i)$.

For each i ($i = 1, 2, \ldots, \ell$), a ($a = 1, \ldots, k$), and j ($j = 1, \ldots, d$), let $s_{i,a,j}$ be an independent random key, satisfying

$$\begin{aligned} s_i &= BITWISE - XOR(s_{i,1,1}, \ldots, s_{i,1,d}) \\ &= BITWISE - XOR(s_{i,2,1}, \ldots, s_{i,2,d}) \\ &\vdots \\ &= BITWISE - XOR(s_{i,k,1}, \ldots, s_{i,k,d}) \end{aligned}$$

The key $s_{i,a,j}$, encrypted under each of the $4 \log^2 k$ keys in the range of the function $g_{i,a,j}$, is added to the enabling block.

User u possesses the d keys $g_{i,h_i(u),1}(u), \ldots, g_{i,h_i(u),d}(u)$ and so he is capable of decoding $s_{i,h_i(u),1}, \ldots, s_{i,h_i(u),d}$, allowing him to reconstruct s_i and then compute the final key s.

Parameters: The personal key consists of $m = \ell d$ keys. The total number of key encryptions in an enabling block encoding s is $4k\ell d \log^2 k$.

Fraud: The k traitors can get together and expose their own keys in order to construct a pirate decoder. By the bit sensitivity of XOR, the box must be able to decrypt every s_i ($i = 1, 2, \ldots, \ell$). To do this, the decoder must be

able to decrypt a complete row $s_{i,a,1}, \ldots, s_{i,a,d}$ for some a, $1 \leq a \leq k$. So, for each i $(i = 1, 2, \ldots, \ell)$ the traitors choose $a = h_i(u)$ for some $u \in T$, and d keys $g_{i,a,1}(u_1), \ldots, g_{i,a,d}(u_d)$, where $u_1, \ldots, u_d \in T$ and $h_i(u_1) = h_i(u_2) = \ldots = h_i(u_d) = a$. For every i, these d keys are placed in the pirate decoder.

Detection of Traitors: Upon confiscation of a pirate decoder, the set of keys in it, F, is exposed. As argued above, the decoder must contain a block of d keys of the form $k_{i,a,1} = g_{i,a,1}(u_1), \ldots, k_{i,a,d} = g_{i,a,d}(u_d)$ corresponding to each i $(i = 1, 2, \ldots, \ell)$. (If more than one row is in the decoder, only the one with minimum a is used by the detection algorithm.) For each $j = 1, \ldots, d$, the detective identifies the users in $g_{i,a,j}^{-1}(k_{i,a,j})$. Each of these users is called *marked*. All users who are marked at least $d/\log k$ times, are suspects for s_i. The user who is a suspect for the largest number of s_i's is identified as a traitor.

Goal: We want to show that there is a choice of hash functions such that for all coalitions, a good user is never identified as a traitor.

Consider a specific user, say 1, and a specific coalition T of k traitors (which does not include 1). We first bound the probability that user 1 will be a suspect for s_i. The first level hash function h_i partitions the users to k subsets $\{h_i^{-1}(1), \ldots, h^{-1}(k)\}$. The expected maximum number of traitors in these k subsets is $\log k / \log\log k$. The probability that user 1 is hashed to a subset together more than $\log k$ traitors is at most $1/16k$ [2]. Denote $h_i(1) = a$. Consider the conditional probability space where $T \cap h_i^{-1}(a)$ contains indeed at most $\log k$ traitors. In this conditional space, the d keys $k_{i,a,1}, \ldots, k_{i,a,d}$ in the pirate decoder come from the personal keys of $T \cap h_i^{-1}(a)$. As this set contains fewer than $\log k$ members, there must be at least one member in $T \cap h_i^{-1}(a)$ who is marked at least $d/\log k$ times. Therefore at least one member of T is a suspect for s_i.

Returning to our innocent user 1, the detective marks user 1 with respect to $g_{i,a,j}$ if there is some $u \in T \cap h_i^{-1}(a)$ such that $g_{i,a,j}(1) = g_{i,a,j}(u)$. The range of $g_{i,a,j}$ contains $4\log^2 k$ keys. At most $\log k$ of these are in $g_{i,a,j}\left(T \cap h_i^{-1}(a)\right)$. So the probability that user 1 is marked with respect to $g_{i,a,j}$ is at most $1/(4\log k)$. The expected number of times user 1 will be marked, with respect to the d functions $g_{i,a,1}, \ldots, g_{i,a,d}$, is $d/(4\log k)$. We use the Chernoff bound to estimate the probability that user 1 is a suspect for s_i.

Set $X_j = 1$ if user 1 is marked with respect to $g_{i,a,d}$, and $X_j = 0$ otherwise. Then $Pr(X_j = 1) \leq 1/(4\log k)$. By the version of the Chernoff bound mentioned above, with $p = 1/(4\log k)$ and $\beta = 4$,

$$Pr\left(\frac{1}{d}\sum_{j=1}^{d} X_j \geq \frac{1}{\log k}\right) < \left(\frac{e^3}{4^4}\right)^{d/4\log k} \leq 2^{-3d/4\log k}.$$

Setting $d = 2\log^2 k$, the conditional probability that user 1 is a suspect for s_i is at most $2^{-3\log k/2} < 1/16k$. The probability of the condition not happening is at most $1/16k$. So overall, the total (unconditional) probability that user 1 is the suspect for s_i is at most $1/8k$.

For $i = 1, \ldots, \ell$, let $Y_i = 1$ if 1 is the suspect for s_i, and $Y_i = 0$ otherwise. Then

$$Pr\left(\frac{1}{\ell}\sum_{i=1}^{\ell} X_i \geq \frac{1}{2k}\right) < \left(\frac{e^7}{8^8}\right)^{\ell/8k} < 2^{-\ell/k}.$$

So with probability at least $1 - 2^{-\ell/k}$, user 1 is a suspect for fewer than ℓ/k of the s_i.

For every s_i $(i = 1, \ldots, \ell)$, at least one member of T is a suspect for s_i. T contains k traitors, and so there must be one or more traitor who is a suspect for at least ℓ/k s_i's. Therefore the probability that user 1 is mistakenly identified as a traitor is smaller than $2^{-\ell/k}$. The probability that for one of the $\binom{n}{k}$ possible coalitions T of size k, some good user is mistakenly identified, is smaller than $n \cdot \binom{n}{k} \cdot 2^{-\ell/k}$. Setting $\ell = k^2 \log n$, this probability is smaller than 1. This means that there exists a choice of hash functions h_i and $g_{i,a,j}$ such that a good user is never mistakenly identified as a traitor. The resulting open k-traceability scheme has parameters $m = \ell d = 2k^2 \log^2 k \log n$ and $r = 2k\ell d \log^2 k = 4k^3 \log^4 k \log n$.

3.3 A Secret One Level Scheme

We simplify the construction and improve its costs by using a secret scheme. The proposed scheme is one level, and the hash values of users are kept secret. The major source of saving is that it suffices to map the n users into a set of $4k$ keys (rather than k^2 keys as in the simple one level scheme). A coalition of size k will contain the key of any specific user with constant probability. However, as the traitors do not know which key this is, any key they choose to insert into the pirate decoder will miss the key of the authorized user (with high probability).

Initialization: Each user u $(u \in \{1, 2, \ldots, n\})$ is assigned a random name n_u from a universe $\mathcal{U}$ of size exponential in n. These names are kept secret. A set of ℓ hash functions $h_1, h_2, \ldots, h_\ell$ are chosen independently at random. Each hash function h_i maps $\mathcal{U}$ into a set of $4k$ random keys $S_i = \{s_{i,1}, s_{i,2}, \ldots, s_{i,4k}\}$. The hash functions are kept secret as well. User u receives, upon initialization, ℓ keys $\{h_1(n_u), h_2(n_u), \ldots, h_\ell(n_u)\}$.

Distributing a Key: For each i $(i = 1, 2, \ldots, \ell)$ the data supplier encrypts a key s_i under each of the $4k$ keys in S_i. The final key is the bitwise XOR of the s_i's (the j-th bit is $s^{(j)} = XOR_{i=1}^{\ell} s_i^{(j)}$). Each authorized user has one key from S_i, so he can decrypt every s_i, and thus compute s.

Parameters: The memory required per user is $m = \ell$ keys. The total number of broadcasts, used in distributing the key s, is $r = 4k\ell$.

Fraud: The k traitors can get together and expose their own keys. Given these keys, they chose one key per set S_i $(i = 1, 2, \ldots, \ell)$. These ℓ keys are put together in a pirate decoder. This set of keys F, enables the purchaser of such decoder to decrypt every s_i, and thus compute s.

Detection of Traitors: Upon confiscation of a pirate decoder, the set of keys in it, F, is exposed. F contains ℓ keys, one per set S_i. Denote these keys by $f_i \in S_i$. For each i, the users in $h_i^{-1}(f_i)$ are identified and marked. The user with largest number of marks is exposed.

Goal: We want to show that for all (almost all) coalitions, the probability of exposing a user who is not a traitor is negligible.

Clearly, at least one of the traitors contributes at least ℓ/k of the keys to the pirate decoder. We want to show that the probability that a good user is marked ℓ/k times is negligible. Consider a specific user, say 1, and a specific coalition T of k traitors (which does not include 1). As the name assigned to user 1 is random and the hash functions are random, the value $a_i = h_i(n_1)$ is uniformly distributed in S_i, even given the k values hashed by h_i from the names of the coalition members. The probability that the value f_i, chosen by the coalition to the pirate decoder, equals a_i is therefore $q = 1/4k$. Let X_i be a zero-one random variable, where $X_i = 1$ if $a_i = f_i$. The mean value of $\sum_{i=1}^{\ell} X_i$ is $\ell/4k$. By the Chernoff bound

$$Pr\left(\frac{1}{\ell}\sum_{i=1}^{\ell} X_i \geq 4q\right) < \left(\frac{e^3}{4^4}\right)^{q\ell} < 2^{-3\ell/4k}.$$

In order to overcome all but p of the $\binom{n}{k}$ coalitions and all n choices of users, we choose ℓ satisfying $n \cdot 2^{-3\ell/4k} < p$. That is, $4k\log(n/p)/3 < \ell$, which gives

Theorem 5. *There is a (p,k)-resilient secret traceability scheme, where a user personal key consists of $m = 4k\log(n/p)/3$ decryption keys, and an enabling block consists of $16k^2\log(n/p)/3$ key encryptions.*

4 Lower Bounds

In this section we derive lower bounds for the case where incrimination has to be absolute, i.e. with no error probability. We assume that the keys the data supplier distributes to the users are unforgeable. This is not accurate, since there is always the small chance that the adversary guesses the keys of the user it wants to incriminate. However, we distinguish between the probability of guessing the keys (which is exponentially small in the length of the key) and the probability of incrimination for other reasons which we would like to be zero. Our view of the system is therefore as follows: let the set of keys used be $S = \{s_1, s_2, \ldots s_r\}$ and let each user i obtain a subset $U_i \subset S$ of size m.

Claim 1 *If no coalition of k users $i_1, i_2, \ldots i_k$ should be able to incriminate a user $i_0 \notin \{i_1, i_2, \ldots i_k\}$, then for all such $i_0, i_1, i_2, \ldots i_k$ we should have that*

$$U_{i_0} \not\subset \cup_{j=1}^{k} U_{i_j}$$

Proof. Suppose not, i.e. there exist $i_0, i_1, \ldots i_k$ such that $U_{i_0} \subset \cup_{j=1}^{k} U_{i_j}$, then the coalition of $i_1, i_2, \ldots i_k$ can reconstruct the keys of U_{i_0} and put them in the pirate decoder for sale. Anyone examining the contents of the box will have to deduce that i_0 is the traitor that generated it.

Luckily, the issue of set systems obeying the conditions of Claim 1 has been investigated by Erdös, Frankl and Füredi [4]. From Theorem 3.3 and Proposition 3.4 there we can deduce that r is $\Omega(\min\{n, k^2\frac{\log n}{\log\log n}\})$ and from Proposition 2.1 there we get that $m \geq k\frac{\log n}{\log r}$. Hence we have:

Theorem 6. *In any open k-resilient traceability scheme distributing every one of the n user m keys out of r we have that r is $\Omega(\min\{n, k^2\frac{\log n}{\log\log n}\})$ and $m \geq k\frac{\log n}{\log r}$.*

Note that the lower bounds on both r and m are roughly a factor of k smaller than the best construction we have for an open traceability system.

Acknowledgments

We are very grateful to Alain Catrevaux, Christophe Declerck, and Henri Joubaud for educating us on the issues of pay television and for motivating all of this work. We also thank to Amos Beimel for his comments on earlier drafts of this manuscript.

References

1. N. Alon, J. Bruck, J. Naor, M. Naor and R. Roth, *Construction of Asymptotically Good Low-Rate Error-Correcting Codes through Pseudo-Random Graphs*, IEEE Transactions on Information Theory, vol. 38 (1992), pp. 509-516.
2. N. Alon and J. Spencer, **The Probabilistic Method**, Wiley, 1992.
3. J. L. Carter and M. N. Wegman, *Universal Classes of Hash Functions*, Journal of Computer and System Sciences 18 (1979), pp. 143-154.
4. P. Erdös, P. Frankl, Z. Füredi, *Families of finite sets in which no set is covered by the union of r others*, Israel J. of math. **51**, 1985, pp. 79–89.
5. M.L. Fredman, J. Komlós and E. Szemerédi, *Storing a Sparse Table with $O(1)$ Worst Case Access Time*, Journal of the ACM, Vol 31, 1984, pp. 538–544.
6. A. Fiat and N. Naor, *Broadcast Encryption*, Proc. Advances in Cryptology - Crypto '93, 1994, pp. 480–491.
7. K. Mehlhorn, **Data Structures and Algorithms: Sorting and Searching**, Springer-Verlag, Berlin Heidelberg, 1984.
8. F. J. MacWilliams and N. J. A. Sloane, **The theory of error correcting codes**, North Holland, Amsterdam, 1977.
9. M. N. Wegman and J. L. Carter, *New Hash Functions and Their Use in Authentication and Set Equality*, Journal of Computer and System Sciences 22, pp. 265-279 (1981).