TRAITOR TRACING FOR SUBSCRIPTION-BASED SYSTEMS

Hongxia Jin and Jeffory Lotspiech
IBM Almaden Research Center
San Jose, CA, USA

Mario Blaum
Hitachi Global Storage Technologies
San Jose, CA, USA

Keywords: Traitor tracing, error correcting code, traceability code.

Abstract: In this paper we study the traitor tracing problem, which originates in attempting to combat piracy of copyrighted materials. When a pirated copy of the material is observed, a traitor tracing scheme should allow to identify at least one of the real subscribers (traitors) who participate in the construction of a pirated copy. In this paper, we focus on the pay-per-view type of subscription-based scenarios, in which materials are divided into multiple segments and each segment has multiple variations. We present a systematic way to assign the variations for each segment and for each subscriber using an error-correcting code. We give sufficient conditions for a code to be able to trace at least a traitor when faced with a coalition of m traitors. We also prove that these sufficient conditions are also necessary when the code is an MDS code.

1 INTRODUCTION

This paper is concerned with the protection of copyrighted materials. A number of business models has emerged whose success hinges on the ability to securely distribute digital content only to paying customers. Examples of these business models include pay-TV systems and selling copyrighted music content through the Web. This paper focuses on subscription-based business models. Each subscriber is allowed to decrypt the broadcast encrypted content that he or she pays for. The security threat in this scenario is that a group of subscribers resell the contents by redistributing the decryption keys or the decoded contents over the Internet. This is a Napster-style "anonymous" attack, and is the most urgent security concern for the content providers. The real subscriber who participates in this piracy will be called indistinctly either a *traitor* or a *colluder*. A traitor tracing scheme allows to identify at least one of the traitors. In this case the only way to trace traitors is to use different versions of the content for different subscribers (users). This allows the re-broadcasted decryption keys or contents to be linked to the group of users who were given that version.

The traitor tracing problem was first defined by Chor, Fiat and Naor for a broadcast encryption system (Fiat and Naor, 1993). This system allows encrypted contents to be distributed to a privileged group of receivers (decoder boxes). Each decoder box is assigned a unique set of decryption keys that allows it to decrypt the encrypted content. The security threat, which is different from the one we are dealing in this paper, is that a group of colluders constructs a clone pirate decoder that can decrypt the broadcast content. The traitor tracing scheme proposed in (Chor et al., 1994; B.Chor et al., 2000) randomly assigns the decryption keys to users before the content is broadcast. The main goal of their scheme in this context is to make the probability of exposing an innocent user negligible under as many real traitors in the coalition as possible.

The threat model that this paper is concerned with is what we have called the "anonymous attack". Attackers can construct a pirate copy of the content (content attack) and try to resell the pirate copy over the Internet. Or the attackers reverse-engineer the devices and extract the decryption keys (key attack). They can then set up a server and sell decryption keys on demand, or build a circumvention device and put the decryption keys into the device. There are two well-known models for how a pirated copy (be it the content or the key) can be generated:

1. Given two variants v_1 and v_2 of a segment, the pirate can only use v_1 or v_2, not any other valid variant.

Jin H., Lotspiech J. and Blaum M. (2006).
TRAITOR TRACING FOR SUBSCRIPTION-BASED SYSTEMS.
In *Proceedings of the International Conference on Security and Cryptography*, pages 223-228
DOI: 10.5220/0002097202230228

2. Given two variants v_1 and v_2 of a movie segment $(v_1 \neq v_2)$, the pirate can generate any valid variant out of v_1 and v_2

The second model, of course, assumes the attackers have more power. We believe it fits documents, texts better than media. Therefore in this paper, same as other traitor tracing schemes shown in literatures, we will be assuming that the attackers are restricted to the first model. When using watermark to create variations, it is the watermark robustness assumption. This is not an unreasonable assumption many real application: first of all, building a different variation is a media-format-specific problem. Watermarking is only one of the possible ways to create variations. In a practical watermarking scheme, when given some variants of a movie segment, it would be infeasible for the colluders to come up with another valid variant because they do not have the essential information to generate such a variant. If colluders manipulate the watermarks, for example, by averaging two watermarks v_1 and v_2, we may detect both v_1 and v_2 in best case; In worst case, however, if the watermark cannot be recognized or simply removed, we may not gain information and have to skip this particular variant. This still fits into our first model. Of course, there are ways as (I. Cox and Shamoon, 1997) to make it very difficult for colluders to remove the marks. We have consulted industrial experts on the minimal durations of the watermark in order to achieve the watermark robustness assumption. However, in a DVD format using Blue Laser, the variation can be simply a different playlist. In this case, it may have nothing to do with watermark; thus it is not restricted by the watermark robustness requirement.

Also, for the key attack, the traitors will need to redistribute at least one set of keys for each segment. For cryptographic keys, it is impossible to generate a valid third key from combining two valid other keys. This is equivalent to the first model.

In general, a deterministic traitor tracing scheme incriminates traitors and only traitors. A probabilistic scheme can sometimes have a small chance to incriminate an innocent user. A tracing scheme is static if it pre-determines the assignment of the decryption keys for the decoder or the watermarked variations of the content before the content is broadcast. The traitor tracing schemes in (Chor et al., 1994; B.Chor et al., 2000) are static and probabilistic. Fiat and Tassa introduced a dynamic traitor tracing scheme (Fiat and Tassa, 1999) to combat the same piracy under the same business scenario considered in this paper. In their scheme, each user gets one of the *q* variations for each segment. However, the assignment of the variation of each segment to a user is dynamically decided based on the observed feedback from the previous segment. The scheme can detect up to *m* traitors. Its main drawback is that doing this is impractical in the scenario described above.

A practical traitor tracing scheme needs no more than 10% of the extra bandwidth for the variations, to accomodate billion users and to detect traitors under large coalitions. We have designed a traitor tracing scheme that attempts to meet all the practical requirements. The existing traitor tracing schemes either need more bandwidth than the content provider can economically afford, or the number of players their schemes can accommodate is too few to be practical, or the number of colluding traitors under which their schemes can handle is too few.

Our scheme is static. Like all tracing schemes in this category, it consists of two basic steps:

1. Assign a variation/key for each segment to devices.
2. Based on the observed re-broadcast keys or contents, trace back the traitors.

In rest of the paper, we will present our two main contributions of the paper. In Section 2, we focus on the first step, in other words, the key assignment for the tracing. we will give intuitions why designing a practical tracing scheme is inherently difficult and show how we overcome the difficulties and attempt to meet the practical requirements. Then in Section 3, we will give more in-depth analysis on traceability codes, which is another contribution of this paper. We present a sufficient and necessary condition for a MDS code to be a traceability code.

2 OUR SCHEMA

As shown above, the only way to trace traitors is to use different versions of the content for different users. This allows the re-broadcasted decryption keys or contents to be linked to the group of users who were given that version.

The first problem to be solved is the overhead that a tracing traitors scheme might require. This is a problem encountered in the first step of a traitor tracing scheme. If it was possible to send a different version of the content to each user, the problem would be solved. Unfortunately, in this case the bandwidth usage is extremely poor. While it is perfectly reasonable in a theoretical context to talk about schemes that used large number of variations and increased the space required by 200% or 300%, no movie studio would have accepted this. Although the new generation of DVDs has substantially more capacity, the studios want to use that capacity to provide a high definition picture and to offer increased features on the disc, not to provide better forensics. Most studios were willing, however, to accept some overhead for forensics, if it could be kept below 10%. As a nominal figure, we began to design assuming we had roughly 8 additional minutes (480 seconds) of video strictly for forensic purposes.

The cryptographic literature implied that we should use our 480 seconds to produce the most variations possible. For example, at one particular point in the movie, we could have produced 960 variations of a 1/2 second duration. This clearly would not work. The attackers could simply omit that 1/2 second in the unauthorized copy without significantly diminishing the value of that copy. Instead, we settled on a model where there were on the order of 15 carefully-picked points of variation in the movie, each of duration 2 seconds, and each having 16 variations. Even then, you could argue that the attackers can avoid these 30 or so seconds of the movie. As a result, when we mapped our scheme to the actual disc format, we made sure that the duration of the variations was not pre-determined. Of course, longer durations require more overhead. This is a tradeoff studios can make. We should also mention that whether or not a given movie uses tracing traitors technology is always the studio's choice. In the absence of attacks, they would never use it, and the discs would have zero overhead for this purpose.

In our model, assume that each content (for example, a movie) is divided into multiple segments, among which n segments are chosen to have differently marked variations. Each of these n segments has q possible variations. Each user receives the same bulk-encrypted content with all the small variations at chosen points in the content. However, each user plays back the content through a different path, which effectively create a different content version copy. Each copy of the content contains one variation for each segment. Each version can be denoted as an n-tuple $(x_0, x_1, \ldots, x_{n-1})$, where $0 \leq x_i \leq q-1$ for each $0 \leq i \leq n-1$. A coalition could try to create a pirated copy based on all the variations broadcasted to them. For example, suppose that there are m colluders. Colluder j receives a content copy $\underline{t}_j = (t_{j,0}, t_{j,1}, \ldots, t_{j,n-1})$. The m colluders can build a pirated copy $(y_0, y_1, \ldots, y_{n-1})$ where the ith segment comes from a colluder $\underline{t}_k$, in other words, $y_i = t_{k,i}$ where $1 \leq k \leq m$ and $0 \leq i \leq n-1$. It would be nice to be able to trace back the colluders (traitors) who have constructed a pirated copy once such pirated copy is found. Unfortunately, the variations $(y_0, y_1, \ldots, y_{n-1})$ associated with the pirated copy could happen to belong to an innocent subscriber. A weak traitor tracing scheme wants to prevent a group of colluders from thus "framing" an innocent user. A strong traitor tracing scheme allows at least one of the colluders to be identified. In this paper we deal only with strong traitor tracing schemes, so, when we say "traitor tracing scheme," we mean strong traitor tracing scheme.

2.1 Basic Key Assignment

For the first step, assume that each segment has q variations (symbols) and that there are n segments. We represent the assignment of segments for each user using a *codeword* $(x_0, x_1, \ldots, x_{n-1})$, where $0 \leq x_i \leq q-1$ for each $0 \leq i \leq n-1$. The codewords assignment can be random or systematic using error-correcting code. We have chosen to use the latter.

A practical scheme needs to have small extra disc space overhead, accommodate a large number of devices in the system, and be able to trace devices under as large a coalition as possible. It is not hard to see these requirements are inherently conflicting. We will use error-correcting code to show some intuitions on the conflicts between these parameters. q is the number of variations, in other words, different symbols that can be used in each coordinate of a codeword. Take a look at a code $[n, k, d]$, where n is the length of the codewords, k is the source symbol size, or the number of coordinates that can uniquely identify a codeword.and d is the Hamming distance of the code, namely, the mininum number of coordinates by which any two codewords differ. Mathematically these parameters are connected to each other. The number of codewords is q^k, and Hamming distance has the property that $d <= n - k + 1$. q is also related to n, for example, for a "maximal difference separable" (MDS) code, $n <= q$. We know the number of variations q decides the extra bandwidth needed for distributing the content. Without variations, $q = 1$. The extra bandwidth needed for the content is $(q-1) * length\_of\_each\_variation * n$. The Hamming distance d decides its traceability. To defend against a collusion attack, intuitively we would like the variant assignment to be as far apart as possible. In other words, the larger the Hamming distance is, the better traceability of the scheme. On the other hand, to accommodate a large number of devices, e.g. billions, intuitively either q or k or both have to be relatively big. Unfortunately a big q means big bandwidth overhead and a big k means smaller Hamming distance and thus weaker traceability. It is inherently difficult to defend against collusions.

In order to yield a practical scheme to meet all the requirements, we concatenate codes. The number of variations in each segment are assigned following a *code*, namely the inner code, which are then encoded using another *code*, namely the outer code. We call the nested code the *super code*. The inner code effectively create multiple movie version for any movie and the outer code assign different movie versions to the user over a sequence of movies—hence the term "sequence keys". This super code avoids the overhead problem by having a small number of variations at any single point. For example, both inner and outer codes can be Reed-Solomon (RS) codes. In a RS code, if

q is the alphabet size, $n \leq q-1$ is the length of the code. If k is its source symbol size, then its Hamming distance is $d = n-k+1$ and the number of codewords is q^k. For example, for our inner code, we can choose $q_1 = 16$, $n_1 = 15$ and $k_1 = 2$, thus $d_1 = 14$. For the outer code, we can choose $q_2 = 256$, $n_2 = 255$ and $k_2 = 4$, thus $d_2 = 252$. The number of codewords in the outer code is 256^4, which means that this example can accommodate more than 4 billion devices. For the super code which is the concatenation of the inner and outer codes, $q = 16$, $n = n_1 \cdot n_2 = 15 \cdot 255 = 3825$, $k = k_1 \cdot k_2 = 6$, $d = d_1 \cdot d_2 = 14 \cdot 252 = 3528$. Suppose each segment is a 2-second clip, the extra video needed in this example is 450 seconds, within the 10% contraint being placed on us by the studios. So, both q, the extra bandwidth needed, and q^k, the number of devices our scheme can accommodate, fit. The actual choices of these parameters used in the scheme depend on the requirements and are also constrained by the inherent mathematical relationship between the parameters q, n, k, d. In fact, there does not exist a single MDS code that can satisfy all the practical requirements. For a MDS code, $n <= q$, $d = n-k+1$. An MDS code is, in general, too short.

For the second step, same as other tracing schemes (Chor et al., 1994)(B.Chor et al., 2000)(Safani-Naini and Wang, 2003)(Trung and Martirosyan, 2004), we use a straightforward approach. For each user we compute the number of segment variations that his or her copy matches with the observed pirated copy. The scheme incriminates the user who has the largest matching with the pirated copy.

3 MATHEMATICAL ANALYSIS

Codes that can provide traceability have been extensively studied in recent years. Weak form of such codes are frameproof codes introduced by Boneh and Shaw (Boneh and Shaw, 1998). Strong form of codes called Identifiable Parent Property (IPP) and traceability codes (TA). Combintatorial properties of IPP codes and TA codes have been studied in (J. N. Staddon and Wei, 2001)(A. Barg and Kabatiansky, 2003).

In this section, we will give more in-depth analysis on the combinaorial properties of the traceability code for deterministic tracing. For reasons of space, the results are given without proof. Definitions 3.1 and 3.2 and Lemma 3.1 are taken from (J. N. Staddon and Wei, 2001).

Definition 3.1 Let $\mathcal{C}$ be a code of length n over an alphabet Q of size q, and C' a subset of $\mathcal{C}$ of size m, i.e., $C' \subseteq \mathcal{C}$ and $|C'| = m$. Moreover, let $C' = \{\underline{t}_1, \underline{t}_2, \ldots, \underline{t}_m\}$, where $\underline{t}_i = (t_{i,0}, t_{i,1}, \ldots, t_{i,n-1})$ for $1 \leq i \leq m$. We say that a word $\underline{v} \in Q^n$, say, $\underline{v} = (v_0, v_1, \ldots, v_{n-1})$, is a descendant of $\mathcal{C}'$ if, for each $0 \leq j \leq n-1$, there is an i, $1 \leq i \leq m$, such that $v_j = t_{i,j}$. The set of descendants of C' is denoted as $\mathrm{desc}(C')$.

In a traitor tracing scheme in a static setting where the assignment of variations is predetermined before the content is broadcast, the rule used to determine a traitor is to incriminate the user who has the maximum number of segments matching with the pirated copy, i.e., the user whose codeword is at the smallest Hamming distance from the pirated copy. We call this rule the *maximum selection rule*.

Definition 3.2 A code $\mathcal{C}$ is called an *m-traceability code* if, given any $C' \subseteq \mathcal{C}$ with $|C'| \leq m$ and $\underline{t} \in \mathrm{desc}(C')$, if there is a codeword $\underline{c} \in \mathcal{C}$ such that $d_H(\underline{c}, \underline{t}) \leq d_H(\underline{c}', \underline{t})$ for any $\underline{c}' \in \mathcal{C}$, $\underline{c}' \neq \underline{c}$, then $\underline{c} \in C'$.

Given an m-traceability code, Definition 3.2 guarantees a deterministic tracing algorithm such that, when inputed a pirated copy by a coalition C' with $|C'| \leq m$, it always outputs an element of C' using the maximum selection rule.

Lemma 3.1 Assume that a code $\mathcal{C}$ with length n and distance d is used to assign the variations for each segment to each user and that there are m traitors. If code $\mathcal{C}$ satisfies

$$d \quad > \quad (1 - 1/m^2)n, \tag{1}$$

then $\mathcal{C}$ is an m-traceability-code.

Is condition (1) also a necessary condition to yield a deterministic tracing algorithm? As we will show below, the answer is no. In order to come up with a tighter condition, we will first introduce a new concept, namely, group distance, which is more relevant than Hamming distance in the traitor tracing context dealing with coalitions.

An $[n, k, d]$ linear code $\mathcal{C}$ over a finite field $GF(q)$ is a linear subspace of $(GF(q))^n$ with dimension k and minimum distance d.

Definition 3.3 Assume that we have a code $\mathcal{C}$ over a field $GF(q)$. Let $\underline{u}$ be any codeword in $\mathcal{C}$, and C_m a set of m codewords in $\mathcal{C}$ not containing $\underline{u}$. We say that $D_m(\underline{u}, C_m)$ is the number of coordinates in $\underline{u}$ that are different to the corresponding coordinates of *each* codeword in C_m. We define the minimum group distance $D_m(\mathcal{C})$ to be the minimum of the $D_m(\underline{u}, C_m)$s for each possible codeword $\underline{u}$ and m-set C_m in $\mathcal{C}$ not containing $\underline{u}$. When there is no ambiguity, we simply denote $D_m(\mathcal{C})$ by D_m.

The following condition on a code is sufficient for a deterministic tracing scheme.

Lemma 3.2 Consider a code $\mathcal{C}$ of length n. If

$$D_m \quad > \quad (1 - 1/m)n, \tag{2}$$

then $\mathcal{C}$ is an m-traceability code.

We want to establish a relationship between D_m and the minimum distance d of code $\mathcal{C}$. Moreover, we will show that an MDS code gives the best possible D_m for any m. The next lemma connects the sufficient conditions (1) and (2) given by lemmas 3.1 and 3.2, respectively.

Lemma 3.3 Let $\mathcal{C}$ be a code of length n and minimum distance d. If $d > (1 - 1/m^2)n$, then $D_m > (1 - 1/m)n$.

The opposite direction in Lemma 3.3 does not hold in general, but it does for MDS codes. We state this result next, as well as further connections between the group distance and the Hamming distance.

Lemma 3.4 Let $\mathcal{U}$ be a vector space of length n and dimension k over a field F. Then, given any $k-1$ coordinates $0 \leq i_1 < i_2 < \cdots < i_{k-1} \leq n-1$, there is a non-zero vector $\underline{u} = (u_0, u_1, \ldots, u_{n-1})$ in $\mathcal{U}$ such that $u_{i_l} = 0$ for $1 \leq l \leq k-1$.

Lemma 3.5 For any $[n, k, d]$ linear code $\mathcal{C}$ over a field $GF(q)$,

$$\max\{0\ ,\ d-(m-1)(n-d)\} \leq D_m \leq$$
$$\max\{0\ ,\ d-(m-1)(k-1)\}.$$

Lemma 3.6 For any $[n, k, d]$ linear MDS code $\mathcal{C}$ over a field $GF(q)$,

$$D_m = \max\{0\ ,\ d-(m-1)(k-1)\} =$$
$$\max\{0\ ,\ n-m(k-1)\}.$$

The next lemma, together with Lemma 3.3, shows that conditions (1) and (2) are equivalent when the lower bound in Lemma 3.5 is met with equality, i.e., $D_m = n - m(n-d)$, and hence for MDS codes.

Lemma 3.7 Let $\mathcal{C}$ be an $[n, k, d]$ linear code such that $D_m = n - m(n-d)$. If $D_m > (1 - 1/m)n$, then $d > (1 - 1/m^2)n$.

Corollary 3.1 Let $\mathcal{C}$ be an $[n, k, d]$ linear MDS code. If $D_m > (1 - 1/m)n$, then $d > (1 - 1/m^2)n$.

As shown in Lemmas 3.1 and 3.2, both $D_m > (1 - 1/m)n$ and $d > (1 - 1/m^2)n$ are sufficient conditions for a deterministic tracing scheme. From Lemmas 3.3 and 3.7, we know that these two conditions are equivalent for codes satisfying with equality the lower bound in Lemma 3.5 (in particular, MDS codes). However, neither of these conditions is necessary for an m-traceability code. In (J. N. Staddon and Wei, 2001), it is listed as an open problem whether or not having an $[n, k, d]$ linear $m-$traceability code implies $d > (1 - 1/m^2)n$. The following trivial example shows that the answer is no. In effect, assume that we have a $[4, 1, 4]$ repetition code over the alphabet $\{0, 1, 2, 3\}$. It is clear that this code is an m-traceability code for any $1 \leq m \leq 3$. It is also clear that $d = D_m = 4$ for $1 \leq m \leq 3$. Notice that both conditions $D_m > (1 - 1/m)n$ and $d > (1 - 1/m^2)n$ are satisfied. Now, consider the [12,1,4] code that coincides with the previous one in the first 4 coordinates and is 0 in the last 8. It is clear that the new code is also an m-traceability code for $1 \leq m \leq 3$, but conditions (1) and (2) are not satisfied.

We have shown that if $d > (1 - 1/m^2)n$ then $D_m > (1 - 1/m)n$ (Lemma 3.3). We have also shown that if the code satisfies with equality the lower bound in Lemma 3.5 (like MDS codes) and if $D_m > (1 - 1/m)n$, then $d > (1 - 1/m^2)n$ (Lemma 3.7). What happens if the code does not satisfy with equality the lower bound in Lemma 3.5? The same example as above shows that $D_m > (1 - 1/m)n$ does not necessarily imply that $d > (1 - 1/m^2)n$. In effect, consider again the $[4, 1, 4]$ repetition code, and take the $[5, 1, 4]$ code that consists of appending a 0 to each of the four vectors of the $[4, 1, 4]$ code. Then, for $m = 3$, $4 = D_3 > (1-1/3)5$, but $4 = d < (1-1/3^2)5$. This example shows that in general, condition (2) is stronger than condition (1) to check if a code $\mathcal{C}$ is an m-traceability code.

Both conditions (1) and (2) imply that the code $\mathcal{C}$ is an m-traceability code, but the examples above show that they are not necessary. However, they are necessary when $\mathcal{C}$ is an MDS code. We close this section by stating this important fact.

Theorem 3.1 Let $\mathcal{C}$ be a linear $[n, k, d]$ MDS code over a finite field $GF(q)$ such that $n \leq q + 1$. Then, for $m \geq 2$, $\mathcal{C}$ is an m-traceability code if and only if $D_m > (1 - 1/m)n$.

The condition $n \leq q + 1$ seems somehow limiting and arbitrary. However, this is the so called MDS conjecture. There are no known non-trivial MDS codes violating this condition. Moreover, MDS codes used in practice, like RS and extended RS codes, satisfy it. So, in actual applications, the condition $n \leq q + 1$ is realistic.

Corollary 3.2 Let $\mathcal{C}$ be a linear $[n, k, d]$ MDS code over a finite field $GF(q)$ such that $n \leq q + 1$. Then, for $m \geq 2$, $\mathcal{C}$ is an m-traceability code if and only if $d > (1 - 1/m^2)n$.

3.1 Is an MDS Code Always the Best?

Based on the definition of group distance D_m, we know that the larger D_m is, the more traitors m under which we can trace an actual traitor deterministically. We have shown that an MDS code gives the largest value of D_m for any m. Does this mean that an MDS code always gives the best deterministic tracing scheme? In some realistic situations, given the size of the field, it might not. To be more precise, let us first

try to apply an actual MDS code, a Reed-Solomon (RS) code, to a real application scenario.

Assume that each segment has q variations. The parameter q determines how much extra bandwidth is needed to broadcast the multiple variations of the segments. A practical solution requires a small extra percentage, like 5% of the normal bandwidth needs. This implies that q cannot be too large. Since the number of codewords (thus, the number of subscribers) that the scheme can accommodate in the application is q^k, a small q requires a not too small k.

Assume that we choose $k=2$. Let $\mathcal{C}$ be an $[n,2,d]$ MDS code with $n=q$ (like an extended RS code), thus $d=n-(k-1)=q-1$. From Corollary 3.2, we know that the condition $d>(1-1/m^2)n$ gives a deterministic tracing scheme for up to m traitors. This in turn means $q-1>(1-1/m^2)q$ for $\mathcal{C}$, so $q>m^2$. Therefore, the maximum number of traitors a deterministic tracing scheme can handle is $\sqrt{q}$. However, if q is a small number (like 16), the total number of subscribers that can be accomodated is q^2, which will also be a small number. A practical tracing scheme may require the capability to deterministically trace more traitors than $\sqrt{q}$ or to accomodate more subscribers than q^2. So, there are no MDS codes that can be directly applied here and still meet these requirements. For that reason, the scheme in (Safani-Naini and Wang, 2003) may not be practical. Are we doomed not to have a better tracing scheme simply because the best codes are MDS? In fact, our question is very similar to one of the open problems listed in (J. N. Staddon and Wei, 2001) asking whether or not we can construct an $m-$traceability code with $q<m^2$ and $a>q$, where a is the number of codewords in the code (for an $[n,k,d]$ linear code, notice that $a=q^k$). This question has been affirmatively answered in a recent paper (Trung and Martirosyan, 2004). However, the codes presented in (Trung and Martirosyan, 2004) either have too few codewords or the size q of the alphabet is too large for practical applications.

4 CONCLUSION

In this paper, we study the problem of tracing the legitimate users (traitors) who instrument their devices and illegally redistributing the pirated copies of the contents or decryption keys on the Internet.

In our scheme, we systematically assign the variations for each segment to the movie and keys to the devices. we use the two-level codes to overcome the overhead problem to prepare the content and enable tracing at the receipt end. Consider our concatenated construction with parameters $q=16$, $n=3825$ and $k=6$. The choice of the parameters allows small extra bandwidth but large number of codewords (16 million), much larger than q, meeting practical requirements.

We also provide formal analysis on traceability code. we introduced a new concept, called "group distance". We believe it is inherently more relevant to measure the traceability of the codes. Using group distance we showed a sufficient condition 2 for traceability code. We also showed this condition is necessary for MDS codes. As future work we would like to find an efficient traceability code that satisfies condition 2 but not 1. Note when traitors become bigger, the tracing has to be probabilistic. Our current traceability analysis is based on a bruteforce step 2 and deterministic tracing. As future work, we want to develop a more efficient step 2.

REFERENCES

A. Barg, R. B. and Kabatiansky, G. (2003). Digital fingerprinting codes: Problem statements, constructions, identification of traitors. In *IEEE Transactions on Information Theory*, volume 49, pages 852–865.

B.Chor, A. Fiat, M. N., and Pinkas, B. (2000). Tracing traitors. In *IEEE Transactions on Information Theory*, volume 46, pages 893–910.

Boneh, D. and Shaw, J. (1998). Collusion-secure fingerprinting for digital data. In *IEEE Trans. on Information Theory*, volume 44, No.5, pages 1897–1905.

Chor, B., Fiat, A., and Naor, M. (1994). Tracing traitors. In *Crypto 1994, Lecture Notes in computer science*, volume 839, pages 480–491.

Fiat, A. and Naor, M. (1993). Broadcast encryption. In *Crypto 1993, Lecture Notes in computer science*, volume 773, pages 480–491.

Fiat, A. and Tassa, T. (1999). Dynamic traitor tracing. In *Crypto 1999, Lecture Notes in computer science*, volume 1666, pages 354–371.

I. Cox, J. Killian, T. L. and Shamoon, T. (1997). Secure spread spectrum watermarking for multimedia,. In *IEEE Transactions on Imaging Processing*, volume 6(12), pages 1673–1687.

J. N. Staddon, D. S. and Wei, R. (2001). Combinatorial properties of frameproof and traceability codes. In *IEEE Transactions on Information Theory*, volume 47, pages 1042–1049.

Safani-Naini, R. and Wang, Y. (2003). Sequential traitor tracing. In *IEEE Transactions on Information Theory*, volume 49, No.5, pages 1319–1326.

Trung, T. V. and Martirosyan, S. (2004). On a class of traceability codes. In *Design, Codes and Cryptography*, volume 31(2), pages 125–132.