RJ10447 (A0906-004) June 2, 2009
Computer Science

IBM Research Report

Unifying Broadcast Encryption and Traitor Tracing for Content Protection

Hongxia Jin, Jeff Lotspiech
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099
USA

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Unifying broadcast encryption and traitor tracing for content protection*

Hongxia Jin, Jeffrey Lotspiech
Content protection Group
Computer Science Division
IBM Almaden Research Center
San Jose, CA, 95120
jin@us.ibm.com, lots@almaden.ibm.com

Abstract

In this paper we study the design of efficient trace-revoke schemes for content protection. In state-of-art, broadcast encryption and traitor tracing are viewed as two orthogonal problems. Good traceability and efficient revocation seem to demand different types of design. When combined into trace-revoke schemes, existing schemes only offer efficiency on one aspect but weak on the other. Moreover, there are two major styles of pirate attacks, namely the clone device attack and anonymous re-broadcasting attack. In current state-of-art,defending against these two attacks are viewed as two different problems that demand different trace-revoke schemes. In current state-of-practice, a content protection system has to deploy two trace-revoke schemes in order to provide complete protections against both attacks. As a result, the system incurs the complexity of having to manage two schemes, even worse the overall strength of the system is the weakest link in either scheme.

In this paper we present a unified trace-revoke system that can offer superior efficiency on both traceability and revocation capability as well as simultaneously defend against two attacks in a unified way. Our unified system offers everything that the original two schemes combined can provide, but our system is much simpler and more efficient. The design of our unified framework carries both scientific and real world practical significance. We reduce the tracing time from tens of years to hours. The much improved simplicity and efficiency of our unified system caused it to be adopted by the new version of AACS [1], Advanced Access Content System, the industry content protection standard for the new Blu-ray ™high-definition-video optical discs. Scientifically our design shows it is possible to design an efficient broadcast encryption scheme and traitor tracing scheme in a unified way. We also showed the equivalence of the two major types of attacks which are currently viewed as different attacks. This opens brand new directions for future research on broadcast encryption and traitor tracing.

*This same paper entitled "A unified broadcast encryption and traitor tracing system for clone attack and anonymous attack" was accepted to appear in Annual Computer Security Application Conference 2007, but had to withdraw from proceeding due to on-going commercialization. A patent was filed early 2007.

1 Introduction

This paper is concerned with the protection of copyrighted digital content. Piracy has become a more and more serious concern for the movie and music industries. The receiving devices like DVD players are interchangeably called receivers, decoders or devices in this paper. A broadcast encryption system [2] has been shown to be very useful in content protection. It allows the broadcaster to distribute content to authorized devices and exclude(revoke) compromised devices.

Since the content in these models is usually large, a broadcast encryption scheme adopts hybrid encryption. More concretely, each receiver is assigned a set of unique secret keys (called device keys); another key (called the media key) is randomly chosen to indirectly encrypt the content. The enabling building block for revocation is a structure called a Media Key Block (MKB) that gets put in the header and distributed together with the content, for example, on the movie disc. An MKB is essentially the media key encrypted by non-revoked receivers' device keys again and again. A non-revoked device can use one of its valid device keys to decrypt the MKB and obtain the valid media key to decrypt the content. If a receiving device is revoked in this MKB, it will get garbage after decrypting the MKB and therefore cannot access the content.

Different pirate attacks can happen in the above system. In a *clone attack*, one or more devices are compromised and

the extracted device keys are used to construct a clone device (or a software program) that can decrypt the encrypted content. Once a clone is found, it is possible to feed testings to the clone and deduce which (traitor) keys are inside the clone box based on the outcome of the testings. Broadcast encryption and traitor tracing schemes [3] for clone attack can be combined into trace-and-revoke schemes [5, 6, 7].

Anonymous attack is also referred to be "re-broadcasting attack" in the literature [8, 9]. In this type of attack, attackers redistribute the media key or the decrypted content to stay anonymous and avoid being identified. Thus different versions of the content and content encrypting keys need to be used. A series of recovered pirated copies of the content/keys allows the detection of the traitors who have involved in constructing and distributing the copies. A trace-and-revoke scheme for anonymous attacks has been shown in [12].

The revocation efficiency of a broadcast encryption system is mainly measured by the size of the header, namely the MKB, since that is the communication overhead involved in revocation. In order to have a smaller MKB, intuitively it is better that any device key is shared by multiple devices so that one encryption in the MKB can enable multiple devices. On the other hand, the efficiency of a traitor tracing scheme is measured by how many tests (or recovered pirated copies) it takes in order to identify traitors. To enable faster tracing, intuitively one wants the devices to share as few keys with other devices as possible.

As one can see, the goal to achieve efficient tracing and efficient revocation seem to be conflicting. Furthermore defending against the above two attacks demands different types of tracing. In fact the state-of-art trace-revoke scheme [6] for clone attack achieves good revocation but not efficient tracing on clone devices, while the trace-revoke scheme [12] for anonymous attack achieves good traceability but not as efficient on revocation.

Unfortunately in order to provide complete protection for both attacks, a content protection system has to deploy both schemes. For example, the Advanced Access Content System (AACS)[1], the new industry content protection standard for Blu-ray high-definition-video opticial discs, contains the state-of-the-art broadcast encryption [6] and tracing traitors technology [12]. As one can imagine, the weakness on traceability and revocation in either scheme will defeat the system.

In this paper we will present a single trace-revoke scheme that unifies broadcast encryption and traitor tracing for both attacks. It provides superior efficiency for both traceability and revocation. A content protection system will now only need to deploy one scheme, a simpler and more efficient scheme. For that reason AACS has adopted our unified system in its "final" specification.

Although AACS inspired this work, we are more interested in the general problem. We think that any broadcast-distribution content protection scheme can be helped by the concepts in this paper. Scientifically the design of our unified system sheds new insights and should open new research directions on designing trace-revoke schemes that are efficient for both tracing and revocation, as well as defending two attacks in a unified way.

In rest of the paper, in Section 2 we will overview the state-of-art and practice broadcast encryption and traitor tracing technology and their drawbacks. We will summarize the main contributions of this paper in Section 3 and point out its many advantages over the current system. We will present our unified system in Section 4. In Section 5 we will show how our unified system can be used for traitor tracing for both clone attack and anonymous attack in the same way and greatly improves traceabilities for both attacks. We will present two tracing schemes, one dynamic and the other semi-static, and analyze their traceabilities. Our new system reduces the tracing time from tens of years to hours. In Appendix we will show security proof and some experimental results.

2 Current state-of-art and practice

Our work is highly inspired by real world applications like AACS. While there exist much work on public key based broadcast encryption [7], the very small storage available to store the device keys for content protection makes any public key based scheme impractical to use. The current state-of-art symmetric key based broadcast encryption scheme is the subset difference based "NNL scheme" [6], after the scheme's authors. The current state-of-art and practice of a trace-and-revoke scheme for re-broadcasting attack is the "JL scheme" shown in [12], after the scheme's authors. Both schemes are deployed in AACS.

2.1 NNL scheme: A revoke-trace scheme for clone attack

Let $\mathcal{D}$ be the set of devices and $\mathcal{K}$ be the set of device keys. Every device $d \in \mathcal{D}$ owns a subset of keys, denoted by $\mathcal{K}_d$. Similarly, associated with every key $k \in \mathcal{K}$ is a set of users $\mathcal{D}_k = \{d \in \mathcal{D} : k \in \mathcal{K}_d\}$. The NNL scheme [6] organizes the devices as the leaves of a tree and its subset difference based device key assignment provides the most concise MKB.

The goal of NNL tracing algorithm for a clone attack is either identify a traitor by detecting its compromised device keys, or create a MKB that the clone device cannot decrypt (i.e., the clone decoder is disabled). In a black box tracing algorithm, the only means to diagnosis traitors is to submit tests, also referred to as *forensic MKBs*, to the clone and observe its response. The structure of a forensic MKB is quite

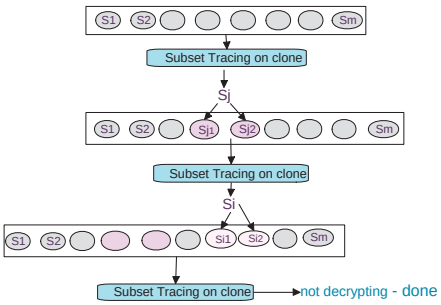

Figure 1. Dynamic Tracing algorithm

simple: we *disable* certain keys by encrypting a random bit string instead of the media key. The remaining keys are said to be *enabled*. Based on the keys inside the clone, the clone may or may not play the content, giving tracing agency information on which keys are inside the clone. The tracing in subset cover based schemes relies on two things:

1. *Bifurcation property*: for every key $k \in \mathcal{K}$ such that $|\mathcal{D}_k| > 1$, there exists keys k_1 and k_2 such that $\mathcal{D}_{k_1} \cup \mathcal{D}_{k_2} = \mathcal{D}_k$ and $D_{k_1} \cap D_{k_2} = \emptyset$. With this, we can replace k with k_1 and k_2 and still cover the same set of devices.

2. *subset tracing procedure*: given a set of keys F, finds at least one key in F owned by the clone device.

The tracing algorithm maintains a covering of all legitimate devices $\mathcal{F}$. The algorithm proceeds by repeatedly identifying a compromised key $k \in \mathcal{F}$, removing it, and adding to $\mathcal{F}$ k_1 and k_2 satisfying the bifurcation property. If $|\mathcal{D}_k| = 1$ then the single device in $\mathcal{D}_k$ is a traitor. This process is reiterated (see Figure 1) until the clone box is unable to play the MKB associated with $\mathcal{F}$.

The efficiency of this type of tracing is mainly measured by how many forensic MKBs are totally needed to complete tracing. That number for NNL tracing is $O(T^3 logT)$ where T is the number of colluding traitors involved in clone attack.

The above polynomial result seems to be satisfactory on paper. But it is not a practical solution. The measures taken by the clone might slow down the testing process. For example, each test might take 1 minute to finish. For a clone device comprising 100 key sets (T = 100), the polynomial results require over 100 million individual tests against the clone, which converts to 15 years' tracing time. Since time-to-respond to attacks is crucial, the significant tracing time may translate to expensive high-performance parallelized hardware for testing.

2.2 JL scheme: A trace-revoke scheme for anonymous attack

As mentioned earlier, the pirates may choose to redistribute the per-content encrypting (media key) or the decrypted plain content. In this type of anonymous attack, traitor tracing schemes need to distribute different versions of the content or encrypting keys to different devices. The current state-of-art and practice traitor tracing scheme for anonymous attack is the JL scheme shown in [12].

In the JL scheme, in order to economically prepare different versions of content for different users, the content owner chooses various points in the content and creates variations at those points. Each variation is not only differently watermarked, but more importantly also differently encrypted. Each device receives the same augmented content but can only decrypt one variation at each augmented segment. In other words, devices play back the content through different paths. This effectively creates different content versions. To avoid a big number of variations at any chosen point, JL scheme uses two levels of assignment as shown in [10]. An "inner code" is used to assign variations within each content, eg, a movie; an "outer code" is used to assign movie versions over a sequence of movies. What is relevant to the discussion in rest of the paper is that each movie comes with different version and each version is differently encrypted. Every device has only one key to decrypt one version for each movie. The keys link to movie versions are called tracing keys (dubbed as sequence keys in AACS).

Each device is assigned a set of tracing keys from a large matrix. The columns correspond to the movies in the sequence; the rows correspond to different versions for each movie. For example, the matrix might be 255 by 256. In a sequence of 255 movies, each movie has 256 movie versions. Each device is assigned exactly one key from each column, 255 in totals. Each key is one of the 256 versions.

When recovering a sequence of pirated movies, the license agency expects to detect traitors by linking the recovered versions to the devices who were originally assigned those versions. The JL scheme employs an efficient coalition detection algorithm [11] that only needs to recover approximately $O(T)$ number of pirated movies in order to detect traitors in a coalition of size T. The algorithm tries to find the smallest coalition that can explain the all recovered movies/keys and can incriminate traitors if the probability of them being framed by larger coalition is negligibly small. We will use this same coalition detection algorithm as a basic step in our new unified semi-static tracing scheme in Section 5.3.

JL scheme also allows revocation of a set of compromised tracing keys and supports multi-time tracing when new attacks arise. Similar to an MKB (media key block)

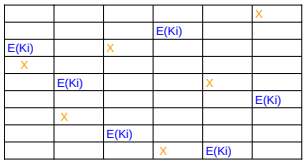

Figure 2. Sample Tracing Key Block

that has been used to revoke device keys in a broadcast encryption scheme, one can use a TKB (tracing key block) to revoke tracing keys. The only difference is that the JL scheme has more than one correct K, (called variant data in AACS), one for each version of the content. Figure 2 shows an example of an TKB with sample encrypted variant data K_i in the cell. The cells marked "X" is revoked, some garbage data is encrypted in those cells. Each compliant device uses its valid tracing key indirectly to decrypt the TKB and obtains a valid variant data from some column. The revoking device will get garbage after processing all the columns in the TKB.

However, as shown in [12], since one TKB can output multiple valid variant data, if the attackers combine the revoked keys with the keys that have not been detected, there are multiple paths to obtain the same valid variant data. As a result, recovering one variant data does not gain as much as information as before. Indeed the q variations now has to spread over c columns. Each column only effectively gets q/c variations. It is clear that traceability degrades when the effective q decreases. When the number of columns c becomes big enough, the traceability degrades so low that it becomes untraceable. The scheme is overwhelmed and broken in that case. As shown in [12], that puts a limit on the revocation capability of the scheme. Indeed it has a finite revocation capability and traceability degrades with revocations.

2.3 Using both NNL and JL schemes in one system

In order to defend against both the clone attack and the anonymous attack, currently a complete content protection system has to utilize two schemes. For example, AACS [1] deployed both NNL and JL schemes for these two attacks. Each device stores a set of device keys and a set of sequence (tracing) keys. When playing back movie $\#i$ from the movie sequence, a device will process MKB first to obtain the valid media key (K_m). It then uses its sequence key K_{s_i} for movie $\#i$, together with K_m to derive a media key variant K_{ms_i}. This key enables the device to decrypt TKB and obtain a valid variant data D_v, which ultimately allows the device to playback the content through its corresponding playback path. As one can see, these two schemes work together to provide content protection.

Putting together two schemes into one system requires the cost of managing two schemes. Moreover, as discussed earlier, NNL scheme provides efficient revocation but relatively weak tracing on clone attack; JL scheme offers good traceability but finite revocation limit based on TKB as well as degraded traceability with revocations. Unfortunately the overall strength of the entire system is the weakest link on either scheme on both traceability and revocation.

3 Main contribution of this paper

In this paper we shall present a system that seamlessly combines the features provided by the two schemes into one unified scheme. More importantly it exploits the best of what can be provided by the two schemes and improves revocation capability and traceabilities on both attacks. Overall, as summarized in the following table it provides significant improvements over the current system.

1. Current content protection system requires two set of keys stored into receiving devices, and manage multiple types of key blocks. Our new system makes use of only one set of keys and one unified type of key block.
2. Current system needs different solutions for two different attacks. Our unified system can defend against both attacks in exactly same way.
3. The traceability for clone attack is improved from $O(T^3 \log T)$ to $O(T)$.
4. The traceability for anonymous attack is greatly improved from the current system; revocation capability for anonymous attack is lifted from a finite limit in the current system to be unlimited in our new system.
5. Our unified system's revocation capability is unlimited, precise and concise

Measurements	Current system	New Unified System
Keys need to store at devices	Device keys; tracing keys	Device keys
Key blocks need to manage	MKB, TKB,Forensic MKB	Unified MKB
Tracing for clone attack	Forensic MKB	Unified MKB
Tracing for anonymous attack	Tracing key/TKB	unified MKB
Traceability for clone attack	$O(T^3 logT)$	$O(T)$
Traceability for anonymous attack	40 (based on sample parameters)	10,000 (same parameters)
Revocation for anonymous attack	40	Unlimited

In summary, the contributions of our paper is two-fold. Practically, we reduce the tracing time from tens of years to hours. The much simplified design and improved efficiency has caused AACS to adopt it in its new specification. Scientifically we show it is possible to look at broadcast encryption and traitor tracing in a unified way and design one

scheme that can offer superior efficiency on both revocation and traceability. We also show the equivalence of two types of attacks, which were originally viewed differently, and we show how to defend against them in exactly the same way. Our work opens new research directions in these areas.

4 A unified system for broadcast encryption and traitor tracing for both clone attack and anonymous attack

While our new system uses only one set of keys (device keys) from a broadcast encryption scheme, for example, from the tree-based NNL scheme, it employs additional media keys in a unified media key block instead of a single media key as is used in current broadcast encryption schemes. Those additional media keys replace the media key variants typically obtained in a traitor tracing system like the JL scheme. The content is prepared with multiple versions in a manner similar to that in the JL scheme, namely, with multiple variations of some chosen segments in the content. Processing this new unified media key block can directly obtain different valid media key variants for different devices, ultimately enabling devices to play back the content through different variations in the content. This is in contrast to the system in which devices have to process both the MKB and the TKB to obtain a media key variant data in order to playback the content. With the multiple media key variants and unified MKB in place, traitor tracing for clone attack and re-broadcasting attack become identical.

4.1 Preliminaries

Media Key Variant (K_{mv}): Any of several valid media keys that can be obtained by processing the new media key block.

Unified Media Key Block (MKBu): A structure comprising different media key variants encrypted by different compliant device keys. Compliant devices obtain different valid media key variants after processing the MKBu.

Title Key (K_t): The key actually used to encrypt and decrypt the segments in the content. Each variation of a segment is encrypted by a different title key.

Variant Title Key Table: A table that allows a device with a media key variant to calculate a list of title keys for different segments to playback the content. This table also comes together with the content. Rows of the table are indexed by K_{mv}. Columns of the table comprise the segments for the content.

Segment1	Segment2	...	...	Segment m
$E(Kt1)_{Km1}$, 1	$E(Kt7)_{Km1}$,7	...	...	$E(Kt20)_{Km1}$,20
$E(Kt1)_{Km2}$,1	$E(Kt4)_{Km2}$, 4	...	...	$E(Kt27)_{Km2}$, 27
...	...	...	...	...
$E(Kt1)_{Kmn}$,1	$E(Kt7)_{Kmn}$, 7	...	...	$E(Kt29)_{Kmn}$, 29

In the sample table shown above, segment 1 is a common segment and does not have any variations. Therefore any device using a valid media key variant K_{m_v} can decrypt an entry in first column and obtain the same valid title key (marked as K_{t_1} in the table). Segment 2 is a segment that has multiple variations. Devices with a media key variant K_{m_1} will get K_{t_7} for this segment; devices with a media key variant K_{m_2} will get K_{t_4} for this segment. Segment m shown in this example also has multiple variations.

4.2 Receiver playback process

The media player first uses the device key to read and process the unified media key block on the media and obtain a media key variant (K_{mv}). This indexes into a row in the variant key table and allows the decryption of one title key for each segment from the table. The player also locates variant numbers corresponding to the variations from the variant key table. Then the media player uses the decrypted title keys to decrypt and play back segment(s) or variation(s) of the segments of the encrypted content.

5 Traitor tracing in unified system

In NNL scheme, a production MKB revokes *known* compromised devices while a forensic MKB tries to detect *unknown* compromised devices. A forensic MKB only serves the purpose of forensics and does not perform revocations on compromised devices. However, a Tracing Key Block in JL scheme for anonymous attack is both for production and for forensics. It revokes known compromised devices and also collects new forensic information for remaining unknown compromised devices. In our new system, all these three types of key blocks are unified into one unified MKB, which is both production and forensic.

Furthermore, an old forensic MKB in NNL can only get one of the two testing results from the clone after feeding it a forensic MKB, namely, the clone box can either decrypt/playback content or not. In our new system, because the content is prepared and encoded with q playback paths, every time a clone device processes a unified MKB it will give the tracing agency one of the q results (the playback path). The effect of this is exactly same as the tracing agency recovers a pirated version of the content or content decrypting key in an anonymous attack. Therefore, traitor tracing for clone device attack and anonymous attack can be treated identically in our new unified system.

We know in order to create an MKB one first finds a set of nodes (subsets) that can cover all non-revoked devices. A unified MKB is multiple media key variants encrypted by the keys associated with the subsets in the cover. We continue to call the set cover the "frontier". However, a new

Algorithm 1 Dynamic Tracing

1: Initialize frontier F
2: **loop**
3: Construct a unified MKB for the current frontier F
4: Distribute unified MKB with next content release or Feed MKB into the clone device
5: Recover pirated content or collect clone box output
6: Identify variation q_i and its corresponding subset S_i used in the clone or recovered pirated content.
7: Update frontier F

question arises. One has to decide how to assign those media key variants among the non-revoked devices in the frontier. The answer to this question affects how efficiently one can revoke and trace. Different types of tracing strategies may mean different ways to create the unified MKBs.

5.1 Create unified MKB for dynamic tracing

The nature of the clone device tracing using forensic MKB is dynamic. A forensic MKB intentionally enables some keys in the frontier and disables other keys in the frontier. A series of forensic MKBs are fed into the clone and the black box subset tracing procedure identifies which key is compromised in current frontier. As shown in Figure 1, this process repeats until the clone cannot playback the forensic MKB or a traitor is found on the leaf.

As shown in Algorithm 1, the new unified system can also perform dynamic traitor tracing for clone device and re-broadcasting attack similar to the NNL tracing approach shown in Figure 1. It follows the same three main steps. The first step is to construct a unified MKB and distribute it with the content or feed it into the clone. Remember a unified MKB is both operational and forensic. The second step is to recover a pirated content/key version or get a response from the clone. The third step is to update the frontier based on the response and go to loop the first step again. The difference is that a forensic testing in NNL scheme gets only two results, namely play or not-play the content; in unified system tracing each unified MKB will give the tracing agency one of the q results.

Of course unified MKBs are created slightly differently as the original forensic MKBs. Now we have to carefully assign the different media keys to different subsets in the current frontier before constructing unified MKBs. As one can imagine, the best strategy to assign the media key versions to the subsets in current frontier partly depends on the attack strategy. For example, they may choose to use one traitor's keys until he is revoked and keep other identities in reserve; or they choose to use more traitors' keys to delay any one traitor's identity being detected. Since we donot know the attack strategy and the attackers equally donot know our tracing strategy, literature often assumes a random strategy. For example, NNL tracing assumed attackers randomly choose to use one key they have on the current frontier. If the key is enabled it plays; if the key is disabled it does not play. We show two different algorithms in Algorithm 2 and Algorithm 3 for two different attack strategies.

Algorithm 2 Dynamic Tracing with "single identity until revoked" strategy

1: Divide tree into q subsets to form a frontier
2: **loop**
3: Assign a separate media key variant to each of the q newly added subsets in the frontier
4: Assign one separate media key variant to all the remaining subsets in the frontier if any
5: Generate a unified media key block based on the media key assignment and distribute with content release or feed into clone
6: Recover variation q_i and identify subset S_i used in the clone or recovered pirated content.
7: **if** S_i is a leaf **then**
8: S_i is identified as a traitor and revoked in future unified MKBs; also remove S_i from F
9: **else**
10: Subdivide S_i into q subdivided subsets
11: Remove S_i from F, but add the q newly subdivided subsets into F

In Algorithm 2 for "single identity until revoked" strategy, it starts with a frontier with q subsets, each assigned one of the q media keys. We will create a unified MKB based on this assignment and distribute it with the content or feed into the clone. Once a version is responded, the identified suspect subset will be split into q subsets. Those newly spawn child subsets are added into the frontier replacing their parent subset. In the new frontier, we assign almost all of the q versions to those newly added subsets, and assign one version to all the remaining subsets. A new unified MKB is constructed based on this new assignment on the new frontier. The process repeats with the new unified MKB.

In Algorithm 3 for random attack strategy, we will group those non-revoking subsets in the frontier. An inactive group contains all currently believed innocent subsets. The frontier initially only contains the inactive group that consists of the subset associated with the root node. The subsets that correspond to a version responded from the attacker will form an active group. So an active group contains one or more subsets that the traitors belong to. With the attackers responding and suspect subsets being identified, more and more active groups are formed and added into the frontier. The q versions will spread evenly among the groups

Algorithm 3 Dynamic tracing to defend against attacks in random strategy

1: The inactive group I = {root}, The initial frontier F contains a single group I
2: **loop**
3: Distribute q versions evenly amongst each group in the frontier, each group gets $q/|F|$ versions
4: For each group G in F, distribute all versions assigned to the group evenly amongst each subset in the group, each subset gets: $x = q/(|F||G|)$
5: If $x >= 1$, for each subset S in group G, sub-divide S into x smaller subsets, and distribute x versions to x smaller subsets
6: If $x < 1$, randomly distribute $q/|F|$ versions among all subsets in G, some subsets get the same version.
7: Construct a unified MKB enabling all subsets using the different versions of the media keys assigned to each subset and distribute the MKB with content or feed into clone
8: With the recovered version v, find which subsets {Si, Sj..} were given version v
9: **if** A subset S_i is a leaf **then**
10: S_i is identified as a traitor and revoked in future MKBs
11: **else**
12: Create a new group G' = {Si, Sj...}
13: If $G' \subset children(I)$, add G' into F, update I = I-G',
14: If $G' \subset children(G)$(an existing active group in F), add G' to F, add G-G' to I, remove G from F

in the frontier. If there are multiple subsets in a group, the versions assigned to the group will be evenly spread among the multiple subsets in that group. After the versions are assigned to those subsets in the frontier, a unified MKB is created by encrypting different media keys with the subset keys that were assigned each media key version. These are shown in line 3-7 of the Algorithm 3.

When a unified MKB constructed from q media keys assigned to subsets in the frontier is distributed with the content or feed into the clone, the outcome of the clone or the recovered pirated content allows the tracing agency to immediately identify which of the q versions it is, thus identify which subsets it corresponds to, i.e., the suspect subsets (line 8 in Algorithm 3).

After the suspect subsets are identified, line 9-14 in Algorithm 3 update the frontier for next iteration. If the identified subset is a leaf, a traitor is identified. If the identified subsets are not at the leaf level, those suspect subsets form an active group and is added into the frontier to further split in next iteration. If those subsets are the children of the inactive group, when the newly formed active group is added into the frontier, they are also removed from the inactive group. If those subsets are the children of an active group, when the newly formed active group is added into the frontier, their parent group will be removed from the frontier. Their sibling subsets will be merged into the inactive group. With the new frontier, the algorithm will loop again from line 3.

5.2 Traceability analysis on dynamic tracing

In the "single identity until revoked" strategy, it only takes $O(1)$ probes to identify which subset to subdivide, because q media keys spreading among q subsets can immediately identify the suspect subset. Furthermore, it takes $\log_q N$ iterations in order to detect a traitor at the leaf. Therefore totally it takes $t \log_q N$ tests. This traceability is a significant improvement over the original $O(t^3 * logt * logN)$ tests needed in NNL tracing scheme shown in Section 2.1.

Before we analyze the performance of our tracing algorithm based on random strategy, we have an important observation. Due to the grouping used when assigning the versions to the subsets, the frontier always contains $r+1$ groups where r is the current coalition size that we deduce at that stage. Keep in mind we assume we do not know the actual coalition size t. We deduce the coalition size while tracing traitors. We start with $r = 0$ and the frontier contains only the inactive group. At each testing, if the attackers responds with a version that was assigned to subsets in the inactive group, they reveal to us there exists a new traitor. In our algorithm, a new active group is formed and added to the frontier. If the attacker responds with a version assigned to an existing active group, the newly formed group replaces its parent group in the frontier. The number of groups in the frontier does not change. Our knowledge of traitor coalition size remains same. Indeed the active groups contain the traitorous subsets. So at any step, the frontier contains $r + 1$ groups including r active traitorous group, and one inactive group. Since we assign q versions evenly among $r + 1$ group and each group is subdivided into $\frac{q}{r+1}$ child subsets, so after each testing the tracing process goes down $\log_2 \frac{q}{r+1}$ levels on one of the traitorous group. As a result the number of tests we need in order to detect t traitors is bounded by the following:

$$\frac{t * \log_2 N}{\log_2 \frac{q}{t+1}} = \frac{t * \log_2 N}{\log_2 q - \log_2(t+1)} \quad (1)$$

The superlinear traceability on random strategy is much improved from the $O(t^3 * logt * logN)$ in NNL tracing based on the same strategy. Our traceability improvement converts to the tracing time reduction from tens of years to hours.

5.3 Create unified MKBs for semi-static tracing

The above dynamic tracing approach means one has to wait for the tracing feedback information to incorporate into the new unified MKB. This is fine in clone tracing at a testing lab and might be also fine for other applications. But when used for content protection applications, keep in mind unified MKBs are production MKBs that we distribute together with the content on the discs.

Highly motivated by real applications like AACS, we are interested in designing schemes that fit into reality. A simple reality is that content owners order MKBs much in advance of the appearance of the discs in retail stores, it is impossible to wait for the tracing information to incorporate into the new MKB. So a traitor tracing scheme must be more static in order to fit into this reality.

As mentioned earlier, an efficient revocation scheme and efficient tracing scheme demand different design. Faster tracing requires any two devices be maximally far apart, i.e., sharing minimal number of keys. On contrast, efficient revocation (i.e., small MKB) requires any key to be shared by many devices. Thus, while it is possible to distinguish two neighbor devices in a reasonable size TKB in JL tracing scheme, the tree-based MKB in NNL scheme has to contain over billion leaf nodes to distinguish individual leaves in the tree.

As a result, the nature of the matrix-based JL scheme is mainly static, meaning that further TKBs can be produced ahead of time, independent of the forensic results from previous ones. This is especially true if assuming random attack strategy. However, note that in the "use a single player's keys until it is revoked" strategy, it will take at least T iterations, each iteration requiring modifications to the TKBs based on previous results, to completely stop the attack.

On contrast, a tree-based system can never be static. Its MKB to distinguish individual leaves is too huge to put on a disc. So its tracing has to be phases, detecting the traitorous subtrees first then producing new MKBs focusing on leaf nodes in the problematic subtrees. However, the tree-based approach did not have to be completely dynamic: multiple MKBs at the same level in the tree could be produced ahead of time, and each could provide forensic information.

It was the combination of the realization that an JL scheme could not be completely static, and a tree-based system did not need to be completely dynamic, that led us to a practical unified approach based on semi-static tracing. Basically the tree will be divided into more subtrees than that in the dynamic tracing case shown in Section 5.1, diving deeper down in the tree than before. Suppose the content is prepared with q versions and there are totally q media key variants (e.g., $q = 1024$). In dynamic tracing the tree can be divided into at most q subtrees. In this semi-static tracing, the tree can be divided into $32K$ subtrees, in other words, the frontier will dive into level 15 of the tree from the root to begin the tracing process.

Algorithm 4 Semi-static Tracing

1: Divide tree into S subtrees to form a frontier, $|S| > q$
2: **loop**
3: Encode q media key variants among S subtrees
4: For each encoding, creates a corresponding unified MKB by repeatedly encrypting each media key variant with its assigned subtree keys
5: Distribute the batch of unified MKBs to licensees or Feed into the clone device
6: Recovering a series of pirated contents or collect a series of testing results from the clone box
7: Using the variations recovered from above to identify a guilty subtree S_i or a set of guilty subtrees S' used in the piracy — call traitor coalition detection algorithm in [11].
8: **if** S_i or a subtree S_i in S' is a leaf **then**
9: S_i is identified as a traitor and revoked in future unified MKBs; also remove S_i from S
10: **else**
11: Add S_i or S' into S
12: Find the parent subtrees in S which contains S_i or any subtree in S'
13: Subtract S_i or all subtrees in S' from their parents

The semi-static tracing algorithm is shown in Algorithm 4. In line 1, the algorithm initializes a partition (frontier) S. As explained in previous paragraph, this partition occurs at a lower level of the tree than that in the dynamic tracing approach described in 5.1. As a result, the number of subsets contained in the frontier is more than q. At line 3 and 4, the encodings of q media key variants among S subtrees are statically assigned to create a batch of unified MKBs. The purpose of the code is similar to the outer code in JL scheme. The codeword is different movie by movie. At line 5 this batch of unified MKBs are used over a sequence of movies, one MKB per movie. Again, they can be distributed together with content or serve as forensic MKBs to feed into a clone device. In line 6 the license agency collects tracing information over q sequence of movies. In line 7 the license agency will use the collected information to identify guilty subsets. This is just as JL scheme does and will use the exact same traitor detection algorithm in JL scheme as shown in [11]. As a result, some subtrees are identified to be the suspects. At line 8 and 9 if some of the identified subtrees are the tree leaves, traitors are identified. If none of the identified subtrees are leaves of the tree, through line 11 to 13 the suspect subtrees will be subdivided into subtrees further down in the tree. The suspect subtrees are removed and

replaced with the newly subdivided subtrees. These new subtrees form a new frontier for next iteration. It goes to next iteration to line 3, similar to the dynamic NNL tracing scheme. Again, the algorithm statically encodes q media key variants among the new S subtrees to create the next batch of unified MKBs. Of course for those subtrees that were never suspects, the algorithm could assign one same media key variant to those subtrees in next batch. This is equivalent to merge them into higher level subtrees. Overall in this algorithm, processing within a batch is static and processing between batches is dynamic.

5.4 Traceability analysis on semi-static scheme

The process within a batch in the semi-static scheme is exactly same as that in current JL scheme. Indeed as shown in line 8, the semi-static algorithm employs the same efficient coalition detection algorithm shown in [11] which achieves almost linear traceability, i.e., it takes $O(T)$ MKBs to detect T traitors.

Of course, the coalition may employ different strategies in using the keys they compromised. In each iteration, they may choose to use all traitors' keys, or only one traitor's keys, or some traitors' keys. For example, a tree is 30 levels in total. Suppose the tree is divided into 32K subtrees and dived 15 levels down in each iteration, it only takes 2 iterations to detect active traitors whose keys are actually used in piracy. If they always use all traitors' keys in each iteration, it only takes 2 iterations to detect all traitors. In each iteration, it takes $O(T)$ MKBs to detect all guilty subtrees (or leaves) when the coalition size is T.

In the case they choose to only use one traitor's keys until it is revoked, it takes 2 iterations to detect one traitor but in each iteration it only takes constant number of unified MKBs to discover the guilty subtree to split. In other words it takes constant number of MKBs to detect one traitor, thus $O(T)$ MKBs to detect all T traitors. Again the traceability of the semi-static tracing is $O(T)$. This is in comparison with the current NNL tracing for clone attack which takes $O(T^3 logT)$ forensic MKBs in order detect traitors of coalition size T.

On the other hand, as shown in Section 2.2, the traceability of the JL scheme degrades with revocation and ultimately the system can be overwhelmed. This is due to the fact that the traitors in the coalition can mix-match the revoked keys with their non-revoked keys to get the same correct variant data thus not giving tracing agency any forensic information on which key has been used to get the correct variant data. On contrast, for the semi-static tracing in the unified approach, as long as the attackers do not have every possible version (i.e., they do not have all q "symbols"), each test or pirated movie yields forensic information. This is a great improvement over the JL scheme. Using a practical number $q = 1024$, tracing for re-broadcasting attack in JL scheme although provides linear traceability but has a very finite up limit (for example, about 40) on traceability and revocation capability before the system is overwhelmed. Our new system improves traceability from about 40 to 10,000 with the same parameter. And furthermore in our new system there is no limit on revocation.

6 Experimental results

In previous sections we have formally analyzed and derived traceabilities for our scheme. We have also performed simulations on the dynamic tracing Algorithm 3 on a ThinkPad T42. We emphasize our simulations using sample practical parameters. For example, we chose 4 million user and 1024 variations of keys/contents, very practical numbers for a real application like AACS. We run 50 times for each coalition size setting and get the average number of tests needed to detect all traitors in the coalition.

Coalition Size	T=10	T=50	T=100	T=200
NNL scheme	60000	15 million	100 million	1 billion
Our scheme	35	250	600	1600

The results shown above are very much consistent with our traceability analysis shown in Section 5.2. The number of tests reduced from our approaches is enormous. This converts the tracing time from tens of years to just hours.

7 Conclusion

In this paper we have shown the first unified system which seamlessly combines broadcast encryption and traitor tracing into one system. It carries both practical and scientific significance. The new unified system simplifies the current system into one set of keys on the device which are used to process a single unified type of key block. It significantly improves the traceability on both types of pirate attacks. It improves the traceability on the clone device attack from $O(T^3)$ to $O(T)$. This converts to tracing time reduction from tens of years to hours. Our semi-static tracing algorithm has practical significance. The simplicity, efficiency and practicality caused it to be adopted as the core technology in the new version of AACS [1]. As future work, we are interested in combining other engineering methods to potentially improve traceability in AACS even more.

Scientifically, while existing work has viewed revocation and tracing as two orthogonal problems that demand different designs to provide efficiency for both revocation and tracing, our system is the first design that unifies broadcast encryption and traitor tracing and provides efficiency

for both. It defends against the two different attacks in exactly the same way. As future work we are also interested in formalizing and generalizing the design of our unified system.

References

[1] http://www.aacsla.com/specifications.

[2] A. Fiat and M. Naor, "Broadcast Encryption," *Crypto'93, Lecture Notes in computer science*, Vol. 773, pp480-491. Springer-Verlag, Berlin, Heidelberg, New York, 1993.

[3] B. Chor, A, Fiat and M. Naor, "Tracing traitors," *Crypto'94, Lecture Notes in computer science*, Vol. 839, pp480-491. Springer-Verlag, Berlin, Heidelberg, New York, 1994.

[4] B. Chor, A, Fiat, M. Naor and B. Pinkas, "Tracing traitors," *IEEE Transactions on Information Theory*, Vol 46(2000), 893-910.

[5] M. Naor and B. Pinkas, "Efficient Trace and Revoke Schemes", Financial Cryptography'2000, Lecture Notes in Computer Science, Vol. 1962, pp. 1-20.

[6] D. Naor, M. Naor and J. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers", *Crypto 2001, Lecture Notes in computer science*, Vol. 2139, pp 41-62, 2001.

[7] D. Boneh and B. Waters, "A collusion resistant broadcast, trace and revoke system", ACM Communication and Computer Security, 2006.

[8] A. Fiat and T. Tassa, "Dynamic traitor tracing," *Crypto'99, Lecture Notes in computer science*, Vol. 1666, pp354-371. Springer-Verlag, Berlin, Heidelberg, New York, 1999.

[9] R. Safani-Naini and Y. Wang, "Sequential Traitor tracing," *Crypto'2000, Lecture Notes in computer science*, Vol. 1880, pp. 316-332. Springer-Verlag, Berlin, Heidelberg, New York, 2000.

[10] H. Jin, J.Lotspiech and S.Nusser, "Traitor tracing for prerecorded and recordable media", *ACM DRM workshop*, Oct. 2004.

[11] H.Jin, J. Lotspiech, Nimrod Megiddo, "Efficient Traitor Tracing", IBM research report, RJ10390, Computer Science, 2007.

[12] H.Jin and J. Lotspiech, "Renewable Traitor Tracing: A Trace-Revoke-Trace System For Anonymous Attack", 12th European Symposium on Research in Computer Security, Lecture Notes in Computer Science 4734, pp.563-577, Springer, 2007.