ATTACKS AND FORENSIC ANALYSIS FOR MULTIMEDIA CONTENT PROTECTION

Hongxia Jin and Jeffery Lotspiech

IBM Almaden Research Center
San Jose, CA, 95120

ABSTRACT

Piracy is one of the biggest concerns in entertainment industry. Digital copies are perfect copies. An anti-piracy defense is to perform forensic analysis and identify who participates in the piracy and disable him/her from doing it again in the future. In this paper, we classify potential pirate attacks in various broadcast type content distribution systems. We shall also present corresponding forensic analysis scheme for those attacks.

1. INTRODUCTION

This paper concerns pirate attacks for copy-protected media content and potential forensic analysis to combat piracy in an one-to-many distribution system, such as broadcast and prerecorded physical media. When a pirate copy is found, forensic analysis can allow one to find out the source that the pirate copy comes from. Literatures sometimes refer to "traitor tracing". There are different The problem in a content protection system is to distribute content to a large group but only a limited subset (e.g. paid members or compliant devices) is able to decrypt the content. A copyright protection mechanism defines an algorithm which assigns keys to devices and encrypts the content. In fact, broadcast encryption was first formally studied by Fiat and Naor [1], and it has been used for copyright protection. In this system, each decoder box is assigned a unique set of decryption keys (called device keys). A key management algorithm is defined to assign keys to devices and encrypt the content that can guarantee that only compliant devices can decrypt the content, without requiring authentication of the device. Broadcast encryption is currently being used for content protection of recordable and prerecorded media (CPRM/CPPM) and is implemented in consumer electronics devices ranging from highly portable audio players that use Secure Digital Cards to top of the line DVD-Audio players supporting multiple channels, higher sampling rates and improved frequency response.

The media, such as CD, DVD or a flash memory card, typically contains in its header the encryption of the key K (called media key) which encrypts the content following the header. The media key is encrypted again and again using all the chosen device keys and forms a Media Key Block (MKB) which is sent alongside the content when the content is distributed. The device keys used to encrypt the media key are chosen in a way as to cover all compliant boxes. It allows all compliant devices, each using their set of device keys, to calculate the same key K. But the non-compliant devices cannot calculate the correct key using their compromised keys. Thus the Media Key Block (MKB) enables system renewability. If a device is found to be non-compliant, a set of his device keys is compromised, an updated MKB can be released that causes a device with the compromised set of device keys to be unable to calculate the correct key K. This effectively excludes the compromised device from access the future content. The compromised device keys are "revoked" by the updated MKB. The best broadcast encryption system that uses for MKB is referred to as the subset-difference approach, or simply the NNL tree [2], named after the inventors. The algorithm is based on the principle of covering all non-revoked users by disjoint subsets from a predefined collection. It is tree-based and the subsets are derived from a virtual tree structure imposed on all devices in the system. The algorithm allows very efficient revocation of any combination of device key sets.

Another type of multimedia content distribution may not involve physical media. It is distributed through communication channel. The cost shifts from manufacturing physical media to network bandwidth. For example, pay-TV systems and selling copyrighted music content through the Web. It is the "pay-per-view" type of system. A consumer device (e.g. a set top box as a movie rental box) receives digital movies from some inexpensive source of data, usually some broadcast source or network download. The content is bulk-encrypted and a content protection system needs to make sure that only paid customers can access the content.

It is not hard to see that the success of many of the business models hinges on the ability to securely distribute the content to paying customers. Piracy can cost millions of dollars to the content owners. In this paper, we shall examine some of the pirate attacks and how a forensic analysis scheme can help detect the piracy.

1.1. Attacks

What are the security problems with the broadcast encryption systems described above? A group of colluders (legitimate users), perhaps after successfully reverse-engineering, can construct a clone pirate decoder that can decrypt the content. They can sell the clone decoder for profit. It would be nice to be able to trace back to the individual decoders who participate in the construction of the pirate decoder once a pirate decoder is found. Those compromised devices, such as the infamous "DeCSS" application used for

0-7803-9332-5/05/$20.00 ©2005 IEEE

copying protected DVD Video disks, can be revoked and excluded from newly released content once their existence becomes known. The goal of a forensic analysis scheme is to distribute the decryption keys to the devices so as to allow the identification of at least one key used in the pirate box or clone.

Another possible attack, which is more popular in a business scenario like the movie rental box, is what we call anonymous attack. by redistributing the per-movie decryption keys. This is one of the most urgent concerns of the movie studios that are investigating this technology. Yet another attack is that an attacker redigitizes the analogue output from a compliant device and redistributes the content in unprotected form. In this case, the only forensic evidence available is the unprotected copy of the content. These are Napster-style attacks with movies instead of music.

The rest of the paper is organized as follows. Sections 2, 3 and 4 shall show different forensic analysis methods for the three types of attacks we listed above. We will show the traceability and simulation results in Section 5 and we conclude in Section 6.

2. AGAINST DECSS-TYPE ATTACK

A DeCSS type attack can end up with a clone decoder with built-in device keys. A forensic analysis scheme is designed to distribute the device keys in a way to allow the tracing and detection of at least one key that is used in a pirate box or clone. A deterministic tracing incriminates traitors and only traitors. A tracing scheme is probabilistic if it can sometimes have a small chance to incriminate an innocent user. The tracing schemes in [3, 4] are probabilistic.

Black box tracing is usually used against this type of attack when a clone is found. It assumes that only the outcome of the decoding box can be examined. It provides the black box with encrypted messages (e.g., MKB) and observes its output (the decrypted message, e.g., the media key) trying to figure out who leaked the device keys. A pirate decoder is of interest only if it can correctly decode with probability bigger than some threshold. In the NNL scheme [2], a subset tracing scheme can seamlessly be integrated into the revocation scheme. A subset tracing algorithm devises a sequence of queries (MKBs) through iteration based on constructing useful sets of revoked devices. These special MKBs will finally allow the detection of the colluding decoder's identity. The algorithm maintains a partition $S_{i_1}, S_{i_2}, ..., S_{i_m}$. At each iteration, the clone decoder is fed with a MKB. After observing the outcome of the clone, if it can decrypt the MKB with high probability, one of the subsets is partitioned into two roughly equal subsets, and the goal is to partition a subset only if it contains a traitor. After the split of the subset, a new MKB is constructed and the tracing continues with the new partitioning.

3. AGAINST REDIGITALIZATION ATTACK

If the only forensic evidence available is the unprotected copy of the content, the only way to trace the traitor who leaks the content is to use different versions of the movie content for different users. Unfortunately, it is oftentimes impossible to send a different version of the content to each user, because in this case the bandwidth usage is extremely poor. What is possible, however, is for some small variations to be broadcasted. For example, we could have ten 5-seconds scenes in a movie that have differently marked variations (the underlying scene remains the same). This method would increase the broadcast bandwidth by roughly 5% only. In reality a 5% - 10% of extra space on the disc or communication bandwidth is acceptable.

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 a copy of the content with one variation for each segment. Each copy 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 available to them. It would be nice to be able to do forensic analysis and trace back the colluders (traitors) who have constructed a pirated copy once such pirated copy is found.

Apparently, a tracing based on the pirated content itself requires good watermarking scheme. Watermarking in general can be used to embed information to either identify the author or identify the individual copy. When watermark is used to identify the author, it embeds authorship mark, such as copy right notice. The information embedded is usually same for every copy. On contrast, when used to identify an individual copy, it embeds information hopefully different for every copy. This has also been called fingerprint in literature. Since the first fingerprinting scheme against collusion was presented by Boneh and Shaw [5], there have been considerable work done in this area [6, 7, 8].

For content tracing, it normally requires the assumption that good watermarking techniques with the following properties available: it is possible to insert one of the q types of watermarks into the content so that the attackers cannot create a version with the watermark removed or with a watermark it did not receive. In other words, the model that we assume the hackers use to generate a pirate copy is as follows: given two variants v_1 and v_2 of a segment, the pirate can only use either v_1 or v_2, nothing else.

In reality, this might be an overly simplified assumption. it is not impossible that the watermark is manipulated so that it is unreadable or erased. We choose this setting mainly because of the reasons shown in Section 4.

Fiat and Tessa [9] proposed dynamic tracing, the assignment of the variations of each segment to a user is dynamically decided for the new content distribution based on the observed feedback from a pirate copy. This is similar to the sequence of MKBs fed into the black-box pirate decoder. But when it is done for content, it requires real-time processing, thus large computational overhead, making it impractical.

Our scheme attempts to satisfy realistic parameter restrictions. It needs less extra bandwidth and it can detect more traitors. It can also accomodate large number of players. It has two steps:

1. Assign a variation for each segment to users.

2. Based on the observed re-broadcasted keys or contents, trace back the traitors.

For the first step, unlike [9], our scheme does not need feedback from a previous segment to decide the assignment of the current segment. Our first step is different from the one in [3], which uses a random assignment of the variations for each subscriber. On the contrary, we systematically allocate the variations based on an error-correcting code.

Assume that each segment has q variations 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 $0 \leq i \leq n-1$. In order to yield a practical scheme, our idea is to concatenate codes [10]. Variations in each segment are assigned following an inner code, which are then encoded using an outer code. We call the nested code the *super code*. This super code avoids the bandwidth problem by having a small number of variations at any single point. When possible, both inner and outer codes can be MDS codes, for example, Reed-Solomon (RS) codes [10]. 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 $k_2 = 3$, thus $d_2 = 253$. The number of codewords in the outer code is $256^3 = 16777216$, which means that this example can accommodate more than 16 million subscribers. For the super code which is the concatenation of the inner and outer codes, $q = 16$, $n = n_1 \cdot n_2 = 15 \cdot 255 = 3825$, $k = k_1 \cdot k_2 = 6$, $d = d_1 \cdot d_2 = 14 \cdot 253 = 3542$.

The extra bandwidth needed in this example is about 10% of the normal need. So, both q, the extra bandwidth needed, and q^k, the number of users our scheme can accommodate, fit very well in a practical setting.

Note that MDS codes are usually short. A RS code on q variations can at most have its code length $q-1$ or extended to q. In reality, sometimes q has to be small and the code length may not meet the requirement. In this case, another non-MDS code, like BCH code may be used. The choices of the parameters depend on the parameter restrictions in real world. For example, the extra video required is $(q-1)*n*time\_per\_segment$. Based on the extra bandwidth available and the multimedia format restriction, we can decide q.

For the second step, 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.

4. AGAINST ANONYMOUS KEY ATTACK

In an anonymous attack, the hackers first succeed in the piracy attempt and find out some of the device keys for the decoder or the decryption keys for some variations at each segment for the content. It is probably more likely to happen than the redigitalization attack. They then set up an active server and serve individuals for profit on a per-movie base. For example, they can sell the decryption keys for each segment for a movie. Or they can request MKB from the client and process the MKB using their device keys and give the media key back to the individual client. This is usually an attack for profit. We can trace who are the attackers behind the server. If the server is asking client's MKB and giving back media key, we can carefully construct a sequence of MKBs to find out which device keys have been used in the processing of the MKB. The tracing scheme is same as that used for the DeCSS attack. But if the server is selling decryption keys for each segment, the tracing scheme will be same as that used for the redigitalization attack. However, now that the forensic evidence is the segment decryption key not the decrypted content. watermark is irrelevant. Therefore, the tracing scheme does not require the watermark robustness assumption. Furthermore, given two valid keys, it is unfeasible to create a third valid key. For this reason, we have chosen to assume when given two variants, the hackers participating in the piracy have to choose either one of them, nothing else.

5. TRACEABILITY AND SIMULATION RESULTS

The focus of this paper is on practical issues on forensic analysis, the formal mathematical analysis of our tracing schemes is referred to [11],[12].

There are at least two types of strategies that a coalition can use. Since they have multiple movie content versions or segment decryption keys available to them, they can choose movie-by -movie or mix-and-match. In movie-by-movie strategy, the coalition will choose to use the movie version that a single player has. It is always easier for attackers, e.g., in the redigitalization attack. For this strategy, the inner code is irrelevant for traceability. The outer code determines the traceability completely. Alternatively, in mix-and-match strategy, the coalition will construct a pirate copy by mixing and matching segments from different movie versions distributed to different players. Its tracing requires both inner code and outer code. This strategy may be better for attackers, maybe worse than the movie-by-movie strategy, depending on the size of the coalition.

Intuitively, when the coalition size is small, mix-and-match may not be good for attackers, because it may reveal more information about the coalition to the tracer than the movie-by -movie strategy. For example, suppose there are A, B,C,D four people in the coalition. In movie-by-movie strategy, they choose to publish one of the movie versions they have, e.g., publish A's version, then the tracer knows A is a potential candidate for a traitor. If B does not share a variation with A, the tracer has no way to know that B is part of the coalition. If the hackers mix and match the variations for different segments from all the four people, it is possible that the pirate movie reveals the potential existence of all four people in the coalition, which helps the tracer.

On the other hand, the bigger the coalition is, the more variations for each segment will be known to the coalition and the more likely the pirate copy constructed by mix-and-match can hide colluders' identity. When the coalition is big enough, they can know all the q variations for any segment, then they can freely construct any pirate movie.

The potential candidates for traitors are so big that the tracing scheme is basically broken at that time. The number of people needed to be in the coalition in order to know all q variations is expected to be $q * \log q$, based on the bin-ball theorem. The bin-ball theorem says, if there are q bins, after $q \log q$ random ball throwing, with high probability every bin will have at least one ball.

We have performed simulations on the supercode used in the content and anonymous tracing. In our simulation, we have chosen the concatenated construction with parameters $q = 16$, $n = 3825$, $k = 6$ and $d = 3542$. We randomly select users from the user pool to become traitors. To construct a pirated copy, the traitors randomly choose a variation from all the variations available to them for each segment. We assume the traitors do not know the codebook that is used to assign the codeword to each user. Under this assumption, we believe the best bet for traitors to frame an innocent user is for them to randomly choose the variation for each segment. Our simulation ranks and outputs every user's probability to be a traitor. Our simulation results show that it can output about a dozen of real traitors correctly long before we loop through 255 rounds (movies).

When using movie by movie strategy, the traceability is entirely dependent on the outer code. The outer code is a MDS code, q=256, n=255, d=253, the code is about 11-traceable. This is consistent with our simulation results.

Note that, the traceability of our traitor tracing scheme depends on the efficiency of the underlying watermarking scheme. For example, if a robust watermarking can be achieved on a 2 seconds of clip instead of 5 seconds, then for the same extra bandwidth, we could choose larger q and larger n. Thus the scheme can trace more traitors.

6. CONCLUSION

In this paper we have shown three major types of piracy attacks in a typical one-to-many content distribution system, namely the DeCSS type attack, the redigitalization attack and the anonymous attack. We have also presented potential corresponding forensic analysis scheme to detect the piracy. Our focus is on the practicality of the scheme. For redigitalization and anonymous attacks, we have presented a systematic way to assign the variations using classic error correcting codes. It fits into the real world parameter restrictions by requiring small extra bandwidth, accommodating large populations, being able to trace big coalitions.

As one of the future work, we like to formalize the analysis of mix-match attack. Another major future work is to address an important real life problem in tracing: one never knows how many attackers actually involve in the piracy. We think there are other improvements or adjustments that the authority can do to improve the real traceability of the scheme.

Our scheme can be also applied in other business scenarios, for example, for physical media in rental business models and business to business applications. We realize we encounter many practical issues when deploying our scheme, for example, it impacts on video authoring process and transcoding. As future work, we would like to continue focusing on those practical issues and improve the usability of the scheme.

7. REFERENCES

[1] A. Fiat and M. Naor, "Broadcast encryption," in *Crypto 1993, Lecture Notes in computer science*, 1993, vol. 773, pp. 480–491.

[2] D. Naor, M.Naor, and J.Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Crypto 2001, Lecture Notes in computer science*, 2001, vol. 2139, pp. 41–62.

[3] B. Chor, A. Fiat, and M. Naor, "Tracing traitors," in *Crypto 1994, Lecture Notes in computer science*, 1994, vol. 839, pp. 480–491.

[4] B.Chor, M. Naor A. Fiat, and B Pinkas, "Tracing traitors," in *IEEE Transactions on Information Theory*, 2000, vol. 46, pp. 893–910.

[5] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," in *IEEE Transactions on Information Theory*, 1998, vol. 44, No.5, pp. 1897–1905.

[6] R. Blakely A. Barg and G. Kabatiansky, "Digital fingerprinting codes: Problem statements, constructions, identification of traitors," in *IEEE Transactions on Information Theory*, 2003, vol. 49, pp. 852–865.

[7] G.Tardos, "Optimal probabilistic fingerprint codes," in *35th ACM Symposium on Theory of Computing*, 2003, pp. 116–125.

[8] M. Wu, W.Trappe, Z.J.Wang, and K.J.R.Liu, "Collusion-resistant fingerprinting for multimedia," in *IEEE Signal Processing Magazine*, 2004, vol. 21, pp. 15–27.

[9] A. Fiat and T. Tassa, "Dynamic traitor tracing," in *Crypto 1999, Lecture Notes in computer science*, 1999, vol. 1666, pp. 354–371.

[10] S. Lin and D.J.Costello, "Error control coding: Fundamentals and applications," 1983.

[11] H.Jin, J.Lotspiech, and M.Blaum, "Efficient traitor tracing," in *International Symposium on Communication Theory & Applications, 2003*, pp. 200–204.

[12] D.R. Stinson J. N. Staddon and R. Wei, "Combinatorial properties of frameproof and traceability codes," in *IEEE Transactions on Information Theory*, 2001, vol. 47, pp. 1042–1049.