PAPER Special Section on Information and Communication System Security

A Trade-off Traitor Tracing Scheme*

Go OHTAKE[†,††a)], ***Student Member***, **Kazuto OGAWA**[†], ***Member***, **Goichiro HANAOKA**[†††], ***Nonmember***, ***and*** **Hideki IMAI**[†††], ***Fellow***

SUMMARY There has been a wide-ranging discussion on the issue of content copyright protection in digital content distribution systems. Fiat and Tassa proposed the framework of *dynamic traitor tracing*. Their framework requires dynamic computation transactions according to the real-time responses of the pirate, and it presumes real-time observation of content redistribution. Therefore, it cannot be simply utilized in an application where such an assumption is not valid. In this paper, we propose a new scheme that provides the advantages of dynamic traitor tracing schemes and also overcomes their problems.

key words: *dynamic traitor tracing, watermarking, digital rights management, content distribution*

1. Introduction

There are a lot of approaches to protect content copyrights in content distribution services. Watermarking is one of the most important primitives of these approaches and a lot of methods employ it. The framework of *dynamic traitor tracing* proposed by Fiat and Tassa is one of these methods. It assigns each user to a certain subset in order to trace illegal redistributors (traitors) dynamically in real time according to the illegally redistributed content. Dynamic subset assignment enables tracers to obtain the most useful information to trace traitors according to the traitors' strategy. However, it needs a real-time feedback channel, and thus a delayed attack, which redistributes content with some delay, is effective against it. As the above implies, dynamic traitor tracing is not a practical protection scheme. Our goal is to develop a new traitor tracing scheme that has the advantages of dynamic traitor tracing and fewer of its shortcomings.

1.1 Background

In recent years, the bandwidth available for Internet access has become wider, personal computers have become widespread, and high-density storage media such as DVDs and memory sticks have become inexpensive. As a result, audio and video content can now be easily distributed in digital form, but they can also be copied and redistributed illegally. For example, episodes of serial TV dramas can be illegally uploaded on Internet websites after they were broadcast. The users who did the uploading would face no penalty, since no tracing countermeasures are undertaken. Copyright holders such as broadcasting companies are becoming increasingly concerned about such violations of copyright. Copyright protection has become a major issue in content distribution services.

1.2 Related Works

Traitor Tracing. Traitor tracing is one of the major schemes for protecting copyrighted works. In a system, a content provider encrypts content and distributes it, and each user decrypts it with his/her decryption key, which is distributed prior to the service. Each user's decryption key is unique, so if the user illegally redistributes the decryption key, it is possible to identify the decryption key's owner [4], [5], [8], [13], [15]. However, these traitor tracing schemes cannot protect the decrypted content from illegal copying.

Watermarking. One sort of countermeasure is watermarking. A simple watermarking scheme works as follows [3], [21]. The content provider produces different contents and distributes them to users. These contents are generated from a single original, but their embedded information is different. Effectively then, each user gets content that is different from any other user's content. Unfortunately, as part of a broadcasting service, this scheme requires a network capacity in proportion to the number of users and thus is not practical. Moreover, the scheme is ineffective against a collusion attack, whereby authorized users collude to make other content whose embedded information is not the same as any of the content they receive.

***C*-secure Code.** There are approaches for creating code [6], [7], [20], called c-secure code. Using c-secure code as embedded information, one can specify at least one of up to c colluding attackers. Unfortunately, its code length is very long even if c is relatively small.

Dynamic Traitor Tracing. Fiat and Tassa assume that con-

Manuscript received August 1, 2008.
Manuscript revised December 29, 2008.

[†]The authors are with Science & Technical Research Laboratories, Japan Broadcasting Corporation, Tokyo, 157–8510 Japan.
[††]The author is with the Graduate School of Information Security, Institute of Information Security, Yokohama-shi, 221–0835 Japan.
[†††]The authors are with National Institute of Advanced Industrial Science and Technology, Tokyo, 101–0021 Japan.

*A preliminary version of this paper was presented at 8th International Conference on Cryptology in India (INDOCRYPT'07) [17].

a) E-mail: ohtake.g-fw@nhk.or.jp
DOI: 10.1587/transinf.E92.D.859

Table 1 Comparison of three schemes: in terms of delayed attack security (DA-security), number of dynamic computations to trace all p traitors (# DC), and number of variants (# Var). †: It is static computation and not dynamic one. ‡: This number is proportion to maximum network costs.

Scheme	[10], [11]	[1], [2]	[18], [19]	Ours
DA-security	–	–	√	√
# DC	$p(\log_2 n+1)$	$p(\log_3 n+1)$	$1^{\dagger}$	$max(p(\log_4(n/p)+4)-3, p(\log_4 n+1))$
# Var‡	$2p+1$	$3p+1$	$max(1+\sqrt{2n}, 2p^2+2p-3)$	$3p+1$

tent is redistributed in real time and that it is possible to get the redistributed content in real time [10], [11]. Berkman, Parnas, and Sgall improved Fiat and Tassa's scheme [1], [2]. In this case, the system must dynamically assign codes in real time soon after getting the redistributed content, and this enables one to identify traitors at a lower network cost compared with a static scheme such as c-secure code. Such schemes are called *dynamic traitor tracing* (DTT).

However, DTT has some shortcomings. One is that it must get redistributed content in real time. This means a real-time feedback channel is required. The other shortcoming is that a real-time dynamic watermark assignment is required, which implies that the CPU cost of the watermark assignment server is extremely high. Moreover, there is an effective attack whereby content is redistributed with some delay. That is, since the content provider gets redistributed content with some delay, it is hard for it to assign watermarks dynamically.

Sequential Traitor Tracing. Safavi-Naini and Wang proposed another approach to solve these problems, called *sequential traitor tracing* (STT) [18], [19]. In this scheme, even if there are no traitors, it is necessary to distribute multiple contents, and the number of contents is in proportion to the number of traitors whom the content provider assumes to be colluding and to redistribute content. Hence, the scheme's network cost is high. However, a real-time dynamic watermark assignment is not required, which implies that the CPU cost of the watermark assignment server is low.

The above discussion illustrates that while DTT and STT are effective ways of tracing traitors, they do not meet all of the requirements for copyright protection.

1.3 Our Contribution

Our goal is to develop a new scheme that has the advantages of DTT but fewer of its shortcomings; we call it *trade-off traitor tracing*[†].

In our scheme, several segments are stored and the watermarks embedded into them are detected. Then, the subsequent pattern of watermarks embedded into several consecutive segments is determined after analyzing the detected watermarks pattern and with one dynamic computation. The determination must be made in a way that the information obtained from the watermarks detected in these consecutive segments works most effectively to identify traitors. This method is robust against delayed attacks [18], [19].

(1) Comparison with DTT

Consider a likely scenario in which attackers try to redistribute a serial drama episode the day after it was broadcast. The conventional scheme would not work in this case, but ours would. Say that an episode is broadcast every day. The attackers perform delayed attacks and illegally redistribute the j^{th} episode the next day. The tracers, who would like to identify the attackers, determine the watermark patterns in the $j+1^{th}$ episode broadcast once they notice that this episode has been illegally redistributed. In the case of DTT, only one new watermark pattern is determined for the next episode. On the other hand, in our scheme, one episode is divided into multiple segments (two segments in the following construction to make it easier to assess the performance of our scheme), and a distinct watermark pattern is assigned to each segment. This means one episode is considered to be *one segment* in DTT, but *multiple (two) segments* in our scheme. When the tracers find the next illegal redistribution of the $j+1^{th}$ episode, they can decrease the set of users that includes the attackers to $1/2$ in DTT and to $1/4$ in our scheme. Hence, our scheme is more robust than DTT against delayed attacks.

Moreover, the computational cost (the number of dynamic watermark assignments needed to trace all traitors) of our proposal is less than that of DTT. Table 1 shows a comparison with the schemes of [1], [2], [10], [11], when the number of segments t is two ($t=2$) and the number of watermark variants to identify one traitor α is three ($\alpha=3$). The comparison shows that our proposal can decrease the number of dynamic computations to about $79\%(\simeq \log_4 3 = (p\log_4 n)/(p\log_3 n))$, where p is the number of traitors and n is the total number of users ($p \ll n$).

Totally, our scheme has all of the advantages and fewer of the shortcomings of DTT, and it is more practical than DTT.

(2) Comparison with STT

Our scheme's network cost is lower than STT's [18], [19]. More precisely, even if there is only one traitor, STT always requires the tracer to distribute multiple contents, and the number of such contents q_{STT} increases with the number of traitors p' that the tracer assumes to be colluding. It remarks that $p' \geq p$, and roughly speaking, $q_{STT} = max(1+\sqrt{2n}, 2p^2+2p-3)$. On the other hand, in our

[†]Our scheme has intermediate performance between STT and DTT in terms of network and CPU costs, and that is why we call it "trade-off" traitor tracing.

scheme, if the number of contents to identify one traitor is α and if there is only one traitor, only $\alpha + 1$ contents are distributed. If two or more traitors exist, the number of distributed contents changes gradually and the maximum number is $q_{Ours} = \alpha \cdot p + 1$. For $\alpha = 3$, $q_{Ours} = 3p + 1$, and $max(1 + \sqrt{2n}, 2p^2 + 2p - 3)$ is more than $3p + 1$ in almost all cases. Hence, STT's network cost is almost always higher than ours.

STT does not employ any *dynamic* computations; it employs only one *static* computation. That is, once the distributed variants to each user are determined, the variants are not changed until all p traitors are identified. Thus, STT has a small CPU cost for dynamic computations, and it needs only one static computation to identify all p traitors. However, its large network cost offsets this saving. Table 1 also summarizes a comparison with the schemes of [18], [19].

2. Model: Trade-off Traitor Tracing

In some content distribution systems, any user can become a traitor who illegally redistributes his/her received content. In addition, any countermeasure to trace the traitor is currently not undertaken as described in Sect. 1.1, so the traitor can freely redistribute the content. When some kinds of countermeasures will be undertaken, several traitors may collude and try to break the countermeasures, and such collusion is easy, since the traitors are connected through networks. Moreover, the traitors redistribute content during any time span, since they have storage to store the content. Trade-off traitor tracing scheme is one of countermeasures against such traitor's illegal redistribution.

Our model is similar to that of DTT [10], [11] (we describe the model of DTT in Appendix A.1 and its construction and shortcomings in Appendixes A.2 and A.3 briefly), but real-time content feedback is unnecessary. That is, the next watermark pattern is determined with *adaptive and dynamic* computations, depending on the watermark information extracted from illegally redistributed content, and we assume that the following traitors exist:

- Traitors redistribute content before a broadcaster (tracer) determines a pattern of watermarks embedded into the next content.

Actually, illegal redistribution of content often occurs soon after the content is broadcast, and there seems to be enough time to determine the next watermark pattern. For example, as mentioned in Sect. 1.1, an episode of a serial drama is illegally uploaded on Internet websites just after it was broadcast and the next episode will be broadcast tomorrow or one week later. One day or one week remains to determine the next pattern and it is enough time. Thus, the assumption holds in almost all cases.

Content providers distribute content, and then traitors redistribute it illegally. The providers can see the redistributed content through a feedback channel. One piece of content consists of multiple segments, and multiple variants are generated for each segment. Distinct information is embedded in each variant. In addition, the set of users is divided into subsets, and each variant is distributed to each subset. In this process, several current segments are stored and the watermark information is detected. Then, as shown in Fig. 1, the detected watermark pattern is used to determine the next pattern, which would be embedded in the subsequent segments. That is, users are dynamically assigned to *distinct* subsets for *each* segment at a particular timing. On the other hand, DTT assigns users to the *same* subsets for *all* segments.

With this modification, the computational cost can be decreased and delayed redistribution attacks can be thwarted. In the following sections, we describe the details of our model by using the notation in Table 2.

2.1 Content Structure

Figure 2 shows the content structure of our model. This structure is the same as in DTT. One piece of content is divided into several consecutive segments. In the case of video content, for example, one piece of video content is divided into minute-long segments. We assume that a watermarking scheme exists such that no detection error occurs for any attack (we call it an "ideal" watermarking scheme). Multiple

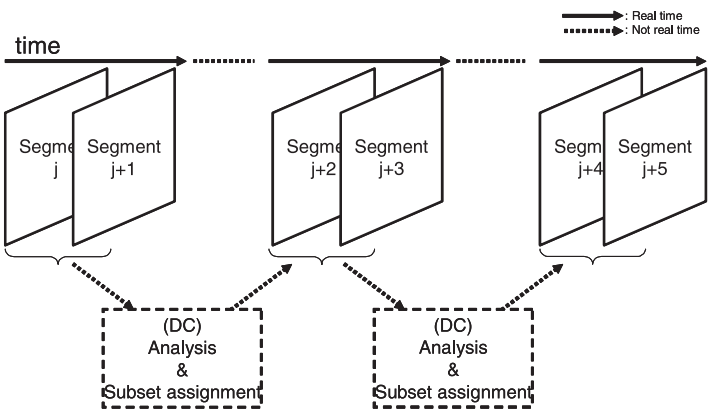

Fig. 1 Trade-off traitor tracing.

Table 2 Notation.

m	:	the number of total segments
t	:	the number of segments which are used at each assignment (more precisely, the number of illegal redistributed segments which are used for one TRC (see the following part))
U	:	the set of all users, $\|U\|=n$
n	:	the number of all users
T	:	a set of users (traitors) who collude and redistribute content illegally, $T \subset U$ and $\|T\|=p$
p	:	the number of traitors
Σ	:	the set of alphabets that are used for watermarking, $\Sigma=\{\sigma_1, \cdots, \sigma_r\}$
r	:	the number of different variants
α	:	the number of variants used to trace one traitor
v_k^j	:	the variant of the j^{th} segment into which σ_k is embedded, $1 \le j \le m$, $1 \le k \le r$
S_k^j	:	the set of the users who received the variant v_k^j
h	:	side information to trace traitors. It includes the attributes of each subset of authorized users. The subset is generated in the tracing process.

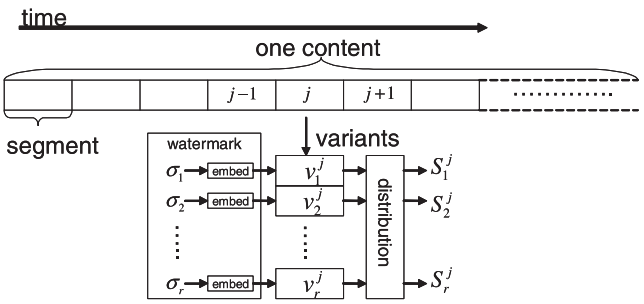

Fig. 2 Content structure.

variants of one segment are generated with this watermarking scheme and the information embedded in each variant is different from the others. The following conditions are required for these variants:

- Fundamentally, all variants carry the same information to the extent that humans cannot easily distinguish between them.
- Given any set of variants of the j^{th} segment ($1 \le j \le m$), $v_1^j, \cdots, v_\lambda^j$, it is impossible to generate another variant that can not be traced back to one of the original variants v_i^j ($1 \le i \le \lambda$).

Clearly, assuming that there exists a watermarking scheme which meets the above requirements, it would be possible to identify at least one variant with illegally redistributed variants and prove that there is one traitor among the set of users who received the same variant with the identified variant.

2.2 Algorithms

Trade-off traitor tracing consists of two algorithms, WMK and TRC. WMK is the algorithm to embed watermarks. TRC is the algorithm to trace traitors who redistribute content illegally.

WMK: This is an algorithm which takes as inputs U, t consecutive segments (from the j^{th} to $j+t-1^{th}$ segments), and h. It generates multiple variants v_k^i ($1 \le k \le r$) for all i segments ($j \le i \le j+t-1$). For all i and k ($j \le i \le j+t-1$, $1 \le k \le r$), it determines the sets of users S_k^i and the variant v_k^i distributed to S_k^i, updates the side information h, and returns v_k^i, S_k^i and h.
TRC: This is an algorithm which takes as inputs U, the detected watermark information from t consecutive segments (from j^{th} to $j+t-1^{th}$ segments), S_k^i ($j \le i \le j+t-1$, $1 \le k \le r$), and h, and returns the updated h.

These two algorithms are used as follows. When the content is distributed, WMK generates multiple variants v_k^i ($1 \le k \le r$) for each segment i ($j \le i \le j+t-1$). If the content is subsequently found to have been illegally redistributed, the variants detected in the content, the user set information S_k^i ($j \le i \le j+t-1$, $1 \le k \le r$), and information h, which shows the relationship between the sets of users and the distributed variants, are inputted to TRC. TRC analyzes these data and reduces the number of suspicious users. It outputs h, which includes information about new subsets from which to collect the most meaningful information. After that, WMK takes as inputs these new subsets and generates new variants v_k^i and sets S_k^i ($1 \le k \le r$) for the next t consecutive segments ($j+t \le i \le j+2t-1$). This process is repeated until all traitors are identified.

3. Our Construction of Trade-off Traitor Tracing

Here, we show the construction of the trade-off traitor tracing scheme. Although the scheme is based on [10], [11], it is significantly different from these other schemes in regard to their strategy to identify traitors. That is, trade-off traitor tracing collects t consecutive segments and analyzes them simultaneously. Different user subsets are created for each segment, and this is the trick to get the most meaningful and most effective information.

On the other hand, direct use of the scheme described in [10], [11] is not effective, since the traitors can adaptively choose which segments to redistribute and thus the content provider (tracer) collects less information. That is, once a set of users is divided into subsets, the subsets are not changed until an illegal redistribution is found. One subset assignment is used for all segments, regardless of the number of segments. The traitors can thus follow a strategy in which they redistribute only the segments distributed to one subset.

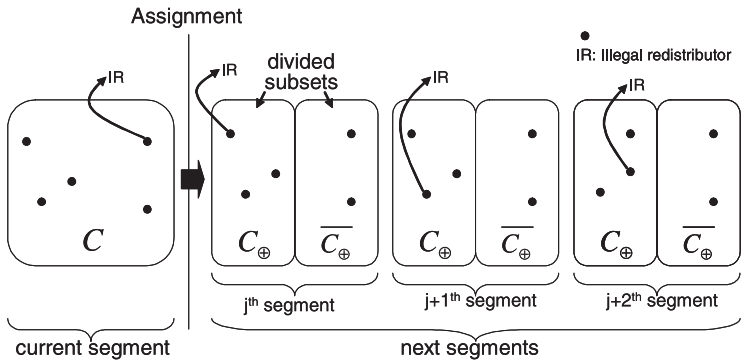

Fig. 3 Direct use of dynamic traitor tracing.

This strategy is illustrated in Fig. 3. C denotes the set of users to which an illegal redistributed variant is distributed first, and $C_\oplus$ and $\overline{C_\oplus}$ denote subsets of C which are created by the tracer for the next distribution and to which different variants are distributed, e.g. two distinct variants v_1 and v_2 are distributed to $C_\oplus$ and $\overline{C_\oplus}$, respectively. At the time of the first content distribution, all of the traitors are in C, but at the time of the next distribution, traitors exist in both subsets $C_\oplus$ and $\overline{C_\oplus}$. Suppose that the traitors only in $C_\oplus$ redistribute the segments in that case. The traitors can select such segments, since they have storage and can analyze the segments. If the duration for storing content and analysis is less than the time that users can wait for, the illegal redistribution would become a viable service, and this attack would become effective since the tracer can only obtain information that the traitor is in one subset $C_\oplus$. That is, the traitors can choose segments such that the information that the tracer can collect from t consecutive redistributed segments is the same information obtained from one of the t segments.

3.1 Basic Strategy

The strategy to collect the most meaningful information to identify traitors from the variants of two segments is as follows. To simplify our explanation in the following, we set t to two ($t = 2$) and the number of watermark variants to identify one traitor α to three ($\alpha = 3$), even though the larger these numbers are, the more effective our scheme becomes. Let C be the set of users who receive an illegal redistributed variant. Regarding the two segments that the content provider will distribute next, the provider makes four subsets of C, $C_\oplus$, $\overline{C_\oplus}$, $C_\otimes$ and $\overline{C_\otimes}$, where $C_\oplus \cup \overline{C_\oplus} = C_\otimes \cup \overline{C_\otimes} = C$, $|C_\oplus| = |\overline{C_\oplus}| = |C_\otimes| = |\overline{C_\otimes}| = \frac{1}{2}|C|$, and $|C_\oplus \cap C_\otimes| = |C_\oplus \cap \overline{C_\otimes}| = |\overline{C_\oplus} \cap C_\otimes| = |\overline{C_\oplus} \cap \overline{C_\otimes}| = \frac{1}{4}|C|$.

For the first segment, one variant is distributed to $C_\oplus$ and another variant to $\overline{C_\oplus}$. For the next segment, one is distributed to $C_\otimes$ and another to $\overline{C_\otimes}$. For example, two distinct variants of the first segment v_1^1 and v_2^1 are distributed to $C_\oplus$ and $\overline{C_\oplus}$, respectively, and two distinct variants of the second segment v_1^2 and v_2^2 are distributed to $C_\otimes$ and $\overline{C_\otimes}$, respectively. If, for example, the variants v_1^1 and v_1^2 assigned to $C_\oplus$ and $C_\otimes$ are found to be illegally redistributed, the following situations can be imagined:

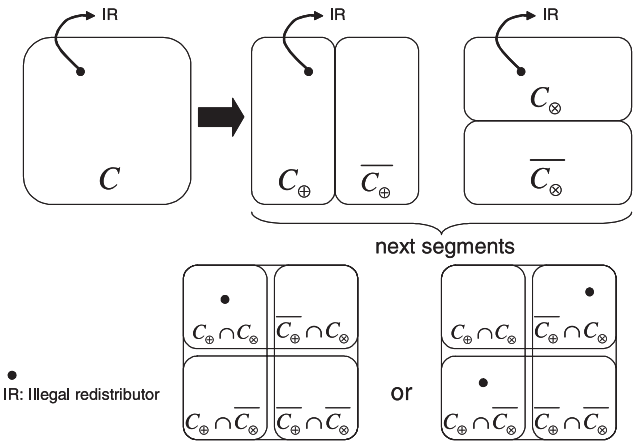

Fig. 4 Basic strategy of our proposal.

(i) At least one traitor is in $C_\oplus \cap C_\otimes$.
(ii) At least one of them is in $C_\oplus \cap \overline{C_\otimes}$ and at least one of them is in $\overline{C_\oplus} \cap C_\otimes$.

In particular, when only one traitor exists in C, (i) is true, and hence, one dynamic computation enables tracers to decrease the number of suspicious users to 1/4. Figure 4 illustrates the strategy. In contrast, the conventional traitor tracing scheme [10], [11] requires two dynamic computations to decrease the number to 1/4.

Realistically (and especially at the beginning of tracing), it is natural to suppose that multiple traitors exist in C. A decision such that (i) or (ii) is true may make it so that traitors can not be identified in the subsequent tracing process. Hence, a decision made from one piece of collected information is not likely to be effective, and we need another strategy to identify the true traitors.

The correct strategy is one in which the tracers set a high probability on (i) and they utilize a scheme (scheme-i) to identify traitors in $C_\oplus \cap C_\otimes$. Simultaneously, for the case that (ii) is true, the tracers utilize another scheme (scheme-i i) to identify traitors in $(C_\oplus \cap \overline{C_\otimes}) \cup (\overline{C_\oplus} \cap C_\otimes)$. Concretely, scheme-i and scheme-ii are as follows:

scheme-i. To investigate $C_\oplus \cap C_\otimes$ in more detail when the next two segments are distributed, $C_\oplus \cap C_\otimes$ is divided into

four subsets $C'_{\oplus}$, $\overline{C'_{\oplus}}$, $C'_{\otimes}$ and $\overline{C'_{\otimes}}$, where $C'_{\oplus} \cup \overline{C'_{\oplus}} = C'_{\otimes} \cup \overline{C'_{\otimes}} = C_{\oplus} \cap C_{\otimes}$, $|C'_{\oplus}| = |\overline{C'_{\oplus}}| = |C'_{\otimes}| = |\overline{C'_{\otimes}}| = \frac{1}{2}|C_{\oplus} \cap C_{\otimes}|$, and $|C'_{\oplus} \cap C'_{\otimes}| = |C'_{\oplus} \cap \overline{C'_{\otimes}}| = |\overline{C'_{\oplus}} \cap C'_{\otimes}| = |\overline{C'_{\oplus}} \cap \overline{C'_{\otimes}}| = \frac{1}{4}|C_{\oplus} \cap C_{\otimes}|$.

scheme-ii. To investigate $(C_{\oplus} \cap \overline{C_{\otimes}}) \cup (\overline{C_{\oplus}} \cap C_{\otimes})$, $(C_{\oplus} \cap \overline{C_{\otimes}}) \cup (\overline{C_{\oplus}} \cap C_{\otimes})$ is treated as one subset L' when the next two segments are distributed.

For example, suppose that three different variants v_1^3, v_2^3 and v_3^3 of the first segment of the next content are distributed to the subsets $C'_{\oplus}$, $\overline{C'_{\oplus}}$ and L', respectively, and three different variants v_1^4, v_2^4 and v_3^4 of the second segment of the next content are distributed to the subsets $C'_{\otimes}$, $\overline{C'_{\otimes}}$ and L', respectively. That is, $C_{\oplus} \cap C_{\otimes}$ is intensively investigated, since there is a strong chance that at least one traitor is in $C_{\oplus} \cap C_{\otimes}$. On the other hand, $(C_{\oplus} \cap \overline{C_{\otimes}}) \cup (\overline{C_{\oplus}} \cap C_{\otimes})$ is not intensively investigated, since there is not much of a chance of finding traitors in it. Moreover, $(\overline{C_{\oplus}} \cap \overline{C_{\otimes}})$ is checked off the list of suspects, since there is no chance of finding traitors in it, and it is added to the innocent subgroup I.

If (ii) is true, the scheme-ii works effectively and the tracers can get more information about the traitors than in the scheme-i. Concretely, if v_3^3 or v_3^4 is redistributed, the tracer can know at least two traitors exist. This information cannot be obtained by using only scheme-i. As a result, traitors cannot help but perform in the way in which (i) is true.

3.2 State Transitions of User Subsets

Our scheme has four states, State0 to State3, and seven transitions, Case1 to Case7. Figure 5 shows the state transition diagram. In the figure, the subsets for two segments are expressed in one object. State$i$-j denotes a State$i$ with an index j, where j is only used to make a distinction from another State$i$. In addition, we use the following variables:

ω: The fewest number of traitors who inevitably exist. $1 \le \omega \le p$.
I: The set of users in which traitors have not been found.
C_l $(1 \le l \le \omega)$: This is a set of users, in which a traitor is known to exist lth.
$C_{\oplus,l}$, $\overline{C_{\oplus,l}}$, $C_{\otimes,l}$, $\overline{C_{\otimes,l}}$ $(1 \le l \le \omega)$: These are sets of users such that $C_{\oplus,l} \cup \overline{C_{\oplus,l}} = C_{\otimes,l} \cup \overline{C_{\otimes,l}} (:= C_l)$ and at least one traitor exists in C_l.
$C'_{\oplus,l}$, $\overline{C'_{\oplus,l}}$, $C'_{\otimes,l}$, $\overline{C'_{\otimes,l}}$, L'_l $(1 \le l \le \omega)$: These are sets of users such that $C'_{\oplus,l} \cup \overline{C'_{\oplus,l}} = C'_{\otimes,l} \cup \overline{C'_{\otimes,l}} (:= C'_l)$ and C'_l includes at least one traitor or L'_l includes at least two traitors.
C''_l, L''_l, R''_l $(1 \le l \le \omega)$: These are sets of users such that two sets among C''_l, L''_l, and R''_l include at least one traitor.
$\emptyset$: This is an empty set of users.

Roughly speaking, the subsets $C_l, C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\otimes,l}, \overline{C_{\otimes,l}}, C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}}$ are sets in which traitors seem likely to exist. The subset L' is the set in which traitors seem not so likely to exist, but the probability is not 0. Tracers set the same probability in subsets C''_l, L''_l, R''_l.

We define State0 to State3 as follows:
State0: The state in which there is only one subset I or C_l.
State1: The state in which there are four subsets $C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\otimes,l}, \overline{C_{\otimes,l}}$.
State2: The state in which there are seven subsets $I, C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}}, L'_l$.
State3: The state in which there are four subsets I, C''_l, L''_l, R''_l.

Diagram. We define the transitions, Case1 to Case7, shown in the diagram. In the following, the pair (A, B) denotes two subsets A and B that received the same variants with the illegally redistributed variants, where A received the variant distributed with the first segment (the first segment of two consecutive segments) and B received the variant distributed with the second segment (the second segment of two consecutive segments).

- State0 is the state in which the set is I or C_l. When illegal redistribution is detected in State0, the state changes into State1. This is Case1 transition.
- When illegal redistribution is detected in State1, the state changes into State2. This is Case2 transition.
- If illegal redistribution is detected in State2 and the detected subsets' pair is one of $\{(C'_{\oplus,l}, C'_{\otimes,l}), (C'_{\oplus,l}, \overline{C'_{\otimes,l}}), (\overline{C'_{\oplus,l}}, C'_{\otimes,l}), (\overline{C'_{\oplus,l}}, \overline{C'_{\otimes,l}})\}$, the state changes into another State2. This is Case3 transition. This state is different from the previous State2, in that the number of $|I|$ increases and the numbers of $|C'_{\oplus,l}|$, $|\overline{C'_{\oplus,l}}|$, $|C'_{\otimes,l}|$, $|\overline{C'_{\otimes,l}}|$ and $|L'_l|$ decrease to 1/4.
- If illegal redistribution is detected in State2 and the detected subsets' pair is (L'_l, L'_l), the state changes into State3. This is Case4 transition.
- If illegal redistribution is detected in State2 and one of the detected subsets is $\{C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}}\}$ and the other is L'_l, the state changes into one State0 and two State1s (State1-1 and State1-2). This is Case5 transition. In this case, there are at least two traitors, and hence, ω is updated to $\omega = \omega + 1$. At least one of the traitors is in a subset ($C'_{\oplus,l}$ in Fig. 5) among $\{C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}}\}$, and at least one of them is in L'_l. $C'_{\oplus,l}$ can then be treated as C_l in State0, and L'_l can be treated as C_ω in State0, where ω is the maximum number of l. Each of these states changes into State1.
- If illegal redistribution is detected in State3 and the detected subsets' pair is one of $\{(C''_l, C''_l), (L''_l, L''_l), (R''_l, R''_l)\}$, the state changes into one State0 and two State1s (State1-1 and State1-2). This is Case6 transition. In this case, there are at least two traitors, and hence, ω is updated to $\omega = \omega + 1$. At least one of the traitors is included in a subset (L''_l in Fig. 5) among $\{C''_l, L''_l, R''_l\}$, and at least one of them is included in a combined subset ($C''_l \cup R''_l$ in Fig. 5), which is generated

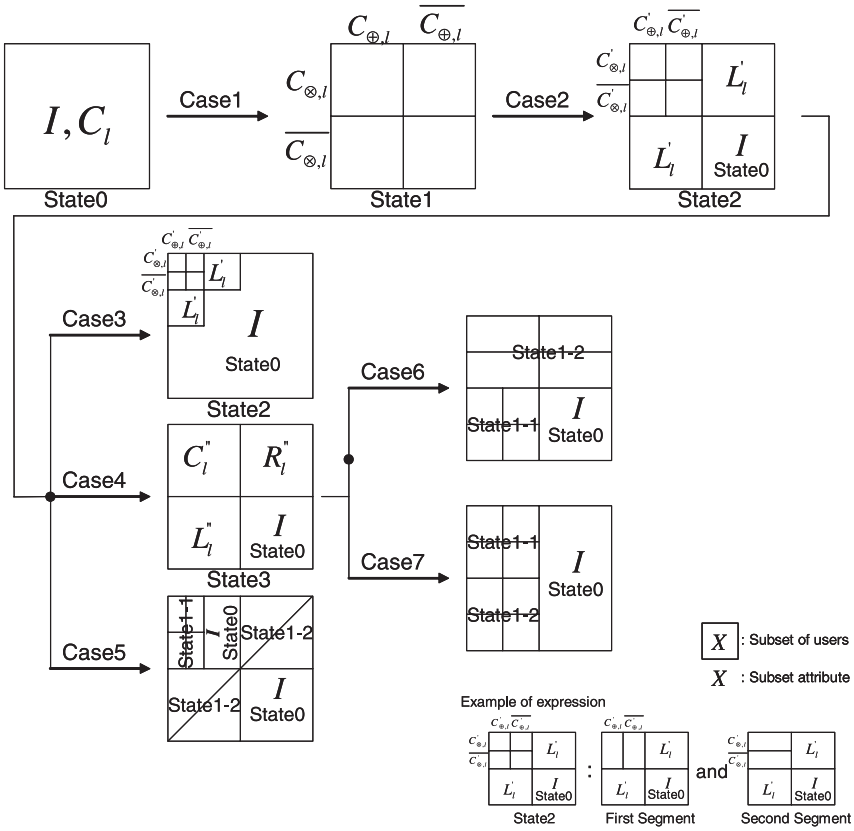

Fig. 5 State transition diagram.

by excluding the detected subset (L_l'' in Fig. 5) from the subset $\{C_l'' \cup L_l'' \cup R_l''\}$. The detected subset ($L_l''$ in Fig. 5) can be treated as C_ω in State0, where ω is the maximum number of l, the other subset ($C_l'' \cup R_l''$ in Fig. 5) can be treated as C_l in State0. Each of these states changes into State1.

- If illegal redistribution is detected in State3 and the detected subsets' pair is one of $\{(C_l'', L_l''), (C_l'', R_l''), (L_l'', C_l''), (L_l'', R_l''), (R_l'', C_l''), (R_l'', L_l'')\}$, the state changes into one State0 and two State1s (State1-1 and State1-2). This is Case7 transition. In this case, there are at least two traitors, and hence, ω is updated to $\omega = \omega + 1$. The traitors are in two different subsets (C_l'' and L_l'' in Fig. 5) among three possible subset pairs, (C_l'' and L_l'', C_l'' and R_l'', or, L_l'' and R_l''). One subset of the two different subsets (L_l'' in Fig. 5) can be treated as C_ω in State0, where ω is the maximum number of l, and the other one (C_l'' in Fig. 5)) can be treated as C_l in State0. Each of these states changes into State1.

As described above, the number of subsets for any segment of any state is less than or equal to three. For example, the number of subsets for the first segment of State1 is two ($C_{\oplus,l}$ and $\overline{C_{\oplus,l}}$), the number of subsets for the second segment of State2 is three ($C'_{\otimes,l}$, $\overline{C'_{\otimes,l}}$ and L'_l), and the number of subsets for the first and second segments of State3 is three (C_l'', L_l'' and R_l''). That is, at least three variants are necessary for tracing one pirate, since one unique variant is distributed to one subset for every segment. Generically, when the number of pirates is p and the number of variants to trace one traitor is α, $\alpha \cdot p$ variants are necessary to trace all p traitors (particularly if p traitors are colluding). On the other hand, when the number of variants r is limited, p is limited to $p \leq \lfloor r/\alpha \rfloor$. In the above case, $\alpha = 3$ and p is limited to $p \leq \lfloor r/3 \rfloor$.

3.3 Full Description of Our Trade-off Traitor Tracing

Our scheme uses the following basic function Sub, and its basic strategy is to decrease the number of users in a set, which includes traitors, to $1/4$.

Sub: It takes as inputs a set of users X, and returns four subsets of X: $X_\oplus$, $\overline{X_\oplus}$, $X_\otimes$, and $\overline{X_\otimes}$, where $X_\oplus \cup \overline{X_\oplus} = X_\otimes \cup \overline{X_\otimes} = X$, $|X_\oplus| \simeq |\overline{X_\oplus}| \simeq |X_\otimes| \simeq |\overline{X_\otimes}| \simeq \frac{1}{2}|X|$, $|X_\oplus \cap X_\otimes| \simeq |X_\oplus \cap \overline{X_\otimes}| \simeq |\overline{X_\oplus} \cap X_\otimes| \simeq |\overline{X_\oplus} \cap \overline{X_\otimes}| \simeq \frac{1}{4}|X|$, and $X_\oplus \cap \overline{X_\oplus} = X_\otimes \cap \overline{X_\otimes} = \emptyset$.

In addition, the function Div2 is used, which is used in DTT [10], [11].

Div2: It takes as inputs a set of users X, and returns two subsets L and R, where $L \cup R = X$, $|L| \simeq |R| \simeq \frac{1}{2}|X|$, and $L \cap R = \emptyset$.

Our scheme also uses the side information h to identify traitors.

h: It includes the number of existing traitors ω (only the number is known at this time) and the attribute information about each subset of users, such as distributed variants and embedded watermarks. Moreover, the attribute information, such as I, $C_{\oplus,l}$, $\overline{C_{\oplus,l}}$, $C_{\otimes,l}$, $\overline{C_{\otimes,l}}$, $C'_{\oplus,l}$, $\overline{C'_{\oplus,l}}$, $C'_{\otimes,l}$, $\overline{C'_{\otimes,l}}$, L'_l, C''_l, L''_l, R''_l $(1 \le l \le p)$, is included. Below, we will refer to the set that holds the attributes X as X for short.

In the following, P denotes the current subset pattern of all users, and it is expressed as sets of subsets. For example, $P = \{I, C_{\oplus,l}, \overline{C_{\oplus,l}}\}$ means that the set of all users is divided into three sets, I, $C_{\oplus,l}$, and $\overline{C_{\oplus,l}}$ and that each set of I, $C_{\oplus,l}$ and $\overline{C_{\oplus,l}}$ is the set to which one of different variants of a segment is distributed. $P_\oplus$ denotes the P that includes only the following subsets, I, $C_{\oplus,l}$, $\overline{C_{\oplus,l}}$, $C'_{\oplus,l}$, $\overline{C'_{\oplus,l}}$, L'_l, C''_l, L''_l, R''_l $(1 \le l \le p)$. Similarly, $P_\otimes$ denotes the P that includes only the following subsets, I, $C_{\otimes,l}$, $\overline{C_{\otimes,l}}$, $C'_{\otimes,l}$, $\overline{C'_{\otimes,l}}$, L'_l, C''_l, L''_l, R''_l $(1 \le l \le p)$. If ω and P are given, the watermark information that is used to generate each variant of a segment is assigned to each subset of P, and the variants that are to be distributed to each user are determined. Hence, we shall focus on how a content provider (tracer) updates P. In our scheme, $P_\oplus$ and $P_\otimes$ show the relationship between the variants of two consecutive segments and subsets of users. h includes ω, $P_\oplus$, $P_\otimes$ and the information about the relationship between the subset patterns and watermarks embedded into the variants.

Our Construction. Trade-off traitor tracing is performed with the algorithms WMK and TRC. WMK generates multiple variants of each segment and updates h. TRC has seven cases. In each case, new subsets patterns for the next distribution are determined, the current subsets patterns are erased from $P_\oplus$ and $P_\otimes$, and the new subset patterns are added to $P_\oplus$ and $P_\otimes$. Basically, the idea is that the new subsets taken from the most suspicious subset are smaller than the current most suspicious subset, and new attributes are added to the other subsets without eliminating the possibility of them including traitors. In Case2, for example, $C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}} \leftarrow \mathsf{Sub}(C_{\oplus,l} \cap C_{\otimes,l})$ and $I \leftarrow I \cup (\overline{C_{\oplus,l}} \cap \overline{C_{\otimes,l}})$ are processes for determining the new subsets, $L'_l \leftarrow (C_{\oplus,l} \cap \overline{C_{\otimes,l}}) \cup (\overline{C_{\oplus,l}} \cap C_{\otimes,l})$ are processes for adding new attributes, and $P_\oplus \leftarrow (P_\oplus \setminus \{C_{\oplus,l}, \overline{C_{\oplus,l}}\}) \cup \{C'_{\oplus,l}, \overline{C'_{\oplus,l}}, L'_l\}$ and $P_\otimes \leftarrow (P_\otimes \setminus \{C_{\otimes,l}, \overline{C_{\otimes,l}}\}) \cup \{C'_{\otimes,l}, \overline{C'_{\otimes,l}}, L'_l\}$ are processes for erasing the current subsets ($\{C_{\oplus,l}, \overline{C_{\oplus,l}}\}$ and $\{C_{\otimes,l}, \overline{C_{\otimes,l}}\}$) and adding new subsets ($\{C'_{\oplus,l}, \overline{C'_{\oplus,l}}, L'_l\}$ and $\{C'_{\otimes,l}, \overline{C'_{\otimes,l}}, L'_l\}$).

$I, \omega, P_\oplus, P_\otimes$ are initialized to $U, 0, \emptyset, \emptyset$, and once a certain subset has only one user and illegally redistributed content has been distributed to it, the user corresponding to that subset is determined to be a traitor, and the content provider suspends subsequent content distributions to him or her.

WMK: It takes as inputs U, two consecutive segments (j^{th} and $j+1^{th}$ segments), and h. It divides U into multiple subsets and chooses different watermarks for each subset. It generates multiple variants v^i_k $(i = j, j+1,\ 1 \le k \le r)$ and updates h depending on the relationship between the subsets and the watermarks. It returns v^i_k, S^i_k $(i = j, j+1,\ 1 \le k \le r)$, and the updated h.

TRC: It takes as inputs $U, \sigma_{k^{(j)}}, \sigma_{k^{(j+1)}}, S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}$ $(1 \le k^{(j)}, k^{(j+1)} \le r)$ and h, and performs the following.

Case1. If $(S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}) = (I, I)$, it performs

$$\omega \leftarrow \omega + 1$$
$$C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}, C_{\otimes,\omega}, \overline{C_{\otimes,\omega}} \leftarrow \mathsf{Sub}(I)$$

and also performs

$$P_\oplus \leftarrow (P_\oplus \setminus \{I\}) \cup \{C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}\}$$
$$P_\otimes \leftarrow (P_\otimes \setminus \{I\}) \cup \{C_{\otimes,\omega}, \overline{C_{\otimes,\omega}}\}\ .$$

Case2. If $(S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}) = (C_{\oplus,l}, C_{\otimes,l})$ for a given l $(1 \le l \le \omega)$, it performs

$$C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}} \leftarrow \mathsf{Sub}(C_{\oplus,l} \cap C_{\otimes,l})\ .$$

It also performs

$$I \leftarrow I \cup (\overline{C_{\oplus,l}} \cap \overline{C_{\otimes,l}})$$
$$L'_l \leftarrow (C_{\oplus,l} \cap \overline{C_{\otimes,l}}) \cup (\overline{C_{\oplus,l}} \cap C_{\otimes,l})$$

and

$$P_\oplus \leftarrow (P_\oplus \setminus \{C_{\oplus,l}, \overline{C_{\oplus,l}}\}) \cup \{C'_{\oplus,l}, \overline{C'_{\oplus,l}}, L'_l\}$$
$$P_\otimes \leftarrow (P_\otimes \setminus \{C_{\otimes,l}, \overline{C_{\otimes,l}}\}) \cup \{C'_{\otimes,l}, \overline{C'_{\otimes,l}}, L'_l\}\ .$$

A similar computation is performed for the other subsets pairs such that $(S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}) = (C_{\oplus,l}, \overline{C_{\otimes,l}})$, $(\overline{C_{\oplus,l}}, C_{\otimes,l})$ or $(\overline{C_{\oplus,l}}, \overline{C_{\otimes,l}})$.

Case3. If $(S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}) = (C'_{\oplus,l}, C'_{\otimes,l})$ for a given l $(1 \le l \le \omega)$, it performs

$$I \leftarrow (I \cup (\overline{C'_{\oplus,l}} \cap \overline{C'_{\otimes,l}})) \cup L'_l$$
$$L'_l \leftarrow (C'_{\oplus,l} \cap \overline{C'_{\otimes,l}}) \cup (\overline{C'_{\oplus,l}} \cap C'_{\otimes,l})$$
$$C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}} \leftarrow \mathsf{Sub}(C'_{\oplus,l} \cap C'_{\otimes,l})\ .$$

A similar computation is performed for the other subsets pairs such that $(S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}) = (C_{\oplus,l}, \overline{C_{\otimes,l}})$, $(\overline{C_{\oplus,l}}, C_{\otimes,l})$ or $(\overline{C'_{\oplus,l}}, \overline{C'_{\otimes,l}})$.

Case4. If $(S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}) = (L'_l, L'_l)$ for a given l $(1 \le l \le \omega)$, it performs

$$C''_l \leftarrow C'_{\oplus,l} \cup \overline{C'_{\oplus,l}} (= C'_{\otimes,l} \cup \overline{C'_{\otimes,l}})$$
$$L''_l, R''_l \leftarrow \mathsf{Div2}(L'_l)$$

and

$$P_\oplus \leftarrow (P_\oplus \setminus \{C'_{\oplus,l}, \overline{C'_{\oplus,l}}, L'_l\}) \cup \{C''_l, L''_l, R''_l\}$$
$$P_\otimes \leftarrow (P_\otimes \setminus \{C'_{\otimes,l}, \overline{C'_{\otimes,l}}, L'_l\}) \cup \{C''_l, L''_l, R''_l\}\ .$$

Case5. If $(S^j_{k^{(j)}}, S^{j+1}_{k^{(j+1)}}) = (C'_{\oplus,l}, L'_l)$ for a given l $(1 \le l \le \omega)$,

it performs

$$\omega \leftarrow \omega+1$$
$$C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}, C_{\otimes,\omega}, \overline{C_{\otimes,\omega}} \leftarrow \mathsf{Sub}(L_l^{'})$$
$$C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\otimes,l}, \overline{C_{\otimes,l}} \leftarrow \mathsf{Sub}(C_{\oplus,l}^{'})$$
$$I \leftarrow I \cup \overline{C_{\oplus,l}^{'}}$$

and

$$P_{\oplus} \leftarrow (P_{\oplus} \setminus \{C_{\oplus,l}^{'}, \overline{C_{\oplus,l}^{'}}, L_l^{'}\}) \cup \{C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}\}$$
$$P_{\otimes} \leftarrow (P_{\otimes} \setminus \{C_{\otimes,l}^{'}, \overline{C_{\otimes,l}^{'}}, L_l^{'}\}) \cup \{C_{\otimes,l}, \overline{C_{\otimes,l}}, C_{\otimes,\omega}, \overline{C_{\otimes,\omega}}\}\ .$$

A similar computation is performed for the other subsets pairs, $(S_{k^{(j)}}^{j}, S_{k^{(j+1)}}^{j+1}) = (\overline{C_{\oplus,l}^{'}}, L_l^{'})$, $(L_l^{'}, C_{\otimes,l}^{'})$ or $(L_l^{'}, \overline{C_{\otimes,l}^{'}})$.

Case6. If $(S_{k^{(j)}}^{j}, S_{k^{(j+1)}}^{j+1}) = (L_l^{''}, L_l^{''})$ for a given l $(1 \le l \le \omega)$, it performs

$$\omega \leftarrow \omega+1$$
$$C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}, C_{\otimes,\omega}, \overline{C_{\otimes,\omega}} \leftarrow \mathsf{Sub}(L_l^{''})$$
$$C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\otimes,l}, \overline{C_{\otimes,l}} \leftarrow \mathsf{Sub}(C_l^{''} \cup R_l^{''})$$

and

$$P_{\oplus} \leftarrow (P_{\oplus} \setminus \{C_l^{''}, L_l^{''}, R_l^{''}\}) \cup \{C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}\}$$
$$P_{\otimes} \leftarrow (P_{\otimes} \setminus \{C_l^{''}, L_l^{''}, R_l^{''}\}) \cup \{C_{\otimes,l}, \overline{C_{\otimes,l}}, C_{\otimes,\omega}, \overline{C_{\otimes,\omega}}\}\ .$$

A similar computation is performed for the other subsets pairs, $(S_{k^{(j)}}^{j}, S_{k^{(j+1)}}^{j+1}) = (C_l^{''}, C_l^{''})$ or $(R_l^{''}, R_l^{''})$.

Case7. If $(S_{k^{(j)}}^{j}, S_{k^{(j+1)}}^{j+1}) = (L_l^{''}, C_l^{''})$ for a given l $(1 \le l \le \omega)$, it performs

$$\omega \leftarrow \omega+1$$
$$C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}, C_{\otimes,\omega}, \overline{C_{\otimes,\omega}} \leftarrow \mathsf{Sub}(L_l^{''})$$
$$C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\otimes,l}, \overline{C_{\otimes,l}} \leftarrow \mathsf{Sub}(C_l^{''})$$
$$I \leftarrow I \cup R_l^{''}$$

and

$$P_{\oplus} \leftarrow (P_{\oplus} \setminus \{C_l^{''}, L_l^{''}, R_l^{''}\}) \cup \{C_{\oplus,l}, \overline{C_{\oplus,l}}, C_{\oplus,\omega}, \overline{C_{\oplus,\omega}}\}$$
$$P_{\otimes} \leftarrow (P_{\otimes} \setminus \{C_l^{''}, L_l^{''}, R_l^{''}\}) \cup \{C_{\otimes,l}, \overline{C_{\otimes,l}}, C_{\otimes,\omega}, \overline{C_{\otimes,\omega}}\}\ .$$

A similar computation is performed for the other subsets pairs, $(S_{k^{(j)}}^{j}, S_{k^{(j+1)}}^{j+1}) = (C_l^{''}, R_l^{''})$ or $(L_l^{''}, R_l^{''})$.

The existence of multiple traitors is confirmed in Case4, Case5, Case6 and Case7 transitions, whereas the existence of more than one traitor can not be confirmed in Case1, Case2 and Case3 transitions.

Remark 1: Although this scheme looks complex, it is based on simple strategy we described in Sect. 3.1. The strategy is to decrease the number of suspicious users to 1/4 with the basic function Sub, and when the traitors are not included in the subset that the tracer suspected, the tracer can detect this fact and can continue the tracing in other subsets. For example, let us consider a situation in which the traitors are not included in the set that the tracer suspects. When the tracer finds variants $S_{k^{(j)}}^{j} = C_{\oplus,l}$ and $S_{k^{(j+1)}}^{j+1} = C_{\otimes,l}$, it tries to decrease the number of suspicious users to 1/4 and sets $\{C_{\oplus,l} \cap C_{\otimes,l}\}$ as the main target and divides the main target to four subsets, $\{C_{\oplus,l}^{'}, \overline{C_{\oplus,l}^{'}}, C_{\otimes,l}^{'}$ and $\overline{C_{\otimes,l}^{'}}\}$. If one traitor is in $\{C_{\oplus,l} \cap C_{\otimes,l}\}$, this process works effectively. However, it is also possible that multiple traitors exist and that one of them belongs to $\{C_{\oplus,l} \cap \overline{C_{\otimes,l}}\}$ and one of them belongs to $\{\overline{C_{\oplus,l}} \cap C_{\otimes,l}\}$, and that they collude to make illegal content for redistribution. In this case, new attributes are given to the sets, $\{C_{\oplus,l} \cap \overline{C_{\otimes,l}}\}$ and $\{\overline{C_{\oplus,l}} \cap C_{\otimes,l}\}$, and even if the traitors are not included in the main target set $\{C_{\oplus,l} \cap C_{\otimes,l}\}$, the tracer can learn this and can take measures without having to worry about being misled. If traitors employ such a collusion strategy, the tracer can learn that there are at least two traitors. That is, the tracer can identify two traitors at once. The most adequate strategy for traitors is one in which the tracer gets as little information as possible, for example, information revealing the existence of only one traitor. Nonetheless, employing such a strategy results in the tracer's main target being correct.

4. Evaluation

A similar discussion to the one in [10], [11] on security proves that the following attack is the strongest. Traitors select the variants to be redistributed such that if $p < 60$, the state moves as S1 $\rightarrow$ (S2 $\rightarrow$ S4 $\rightarrow$ S6 or S7) $\rightarrow$ (S2 $\rightarrow$ S4 $\rightarrow$ S6 or S7) $\rightarrow \cdots$, and otherwise S1 $\rightarrow$ S2 $\rightarrow$ S2 $\rightarrow$ S2 $\rightarrow \cdots$, where S1, $\cdots$, S7 denote the states after Case1, $\cdots$, Case7 transitions and A $\rightarrow$ B denotes the transition from state A to state B. We thus shall address the traceability of the tracing scheme and evaluate it with regard to the number of dynamic computations in comparison with DTT [1], [2], [10], [11]. In addition, we shall evaluate our scheme's effectiveness against delayed attacks. Moreover, we shall evaluate it with regard to the network costs in comparison with STT [18], [19].

4.1 Security Analysis

We discuss two kinds of security: "traceability" and "delayed attack resilience".

4.1.1 Traceability

We show that our tracing algorithm can trace at most p $(1 \le p < \lfloor r/3 \rfloor)$ traitors perfectly, where $\lfloor x \rfloor$ denotes a function which outputs a maximum integer less than or equal to x. Formally, we prove the following theorem.

Theorem 1: If the number of traitors p is less than $\lfloor r/3 \rfloor$, the tracing algorithm can trace all p traitors.

For the proof of this theorem, we utilize the following two claims under the condition of $p < \lfloor r/3 \rfloor$. These claims and their proofs use the notation Π_l, where $\Pi_l \in \{C_l, C_{\oplus,l}, \overline{C_{\oplus,l}},$

$C_{\otimes,l}, \overline{C_{\otimes,l}}, C'_{\oplus,l}, \overline{C'_{\oplus,l}}, C'_{\otimes,l}, \overline{C'_{\otimes,l}}, L'_l, C''_l, L''_l, R''_l\}$, and its index is $l\,(1 \le l \le p)$. In the following proofs, $v^j_{\Pi_l}$ denotes the variant of the j^{th} segment distributed to a subset of users Π_l.

Claim 1: When there are p traitors and they belong to p distinct subsets, which have p distinct indices, $(\Pi_1, \cdots, \Pi_p)$, the tracing algorithm can trace all p traitors.

Claim 2: When multiple traitors belong to one subset Π_l and the traitors in Π_l select a variant at every segment for illegal redistribution such that the traitor whose received variant is used at the j^{th} segment is different from the traitor whose received variant is used at the $j+1^{th}$ segment, the tracing algorithm can divide the traitors into two subsets Π_l and $\Pi_{l'}$ $(l \ne l')$.

Proof of Claim 1. Suppose that p $(1 \le p < \lfloor r/3 \rfloor)$ traitors $(u_1, \cdots, u_p)$ belong to p distinct subsets of users $\Pi_1, \cdots, \Pi_p$. Traitors have no strategy except that they select one of p distinct variants that have been distributed to $\Pi_1, \cdots, \Pi_p$ and redistribute it. When the one they select is $v^j_{\Pi_l}$, which has been distributed to $u_l \in \Pi_l$ in S0 $(\Pi_l = C_l)$, the tracer performs Case 1 transition and the state of Π_l moves to S1. When the variants that only u_l receives are repeatedly redistributed, the state moves as follows: $\rightarrow$ S2 $\rightarrow \cdots \rightarrow$ S2 (Case 3 transition) until the number of users in Π_l becomes one, since the other traitors cannot receive the same variants. Even if the variant $v^{j'}_{\Pi_{l'}}$ $(l' \ne l)$ is redistributed on the way, the side information related to Π_l is held and that of $\Pi_{l'}$ is updated. The state of $\Pi_{l'}$ moves as follows: S2 $\rightarrow \cdots$ $\rightarrow$ S2 until the number of users in $\Pi_{l'}$ becomes one. These processes are repeated until the number of users included in every p subset becomes one, and finally, all p traitors can be traced. □

Proof of Claim 2. Suppose that there are multiple traitors in a subset C_l and that the variants distributed to two of these traitors (u_1 and u_2) are redistributed.

We consider the following cases: S1 $\rightarrow$ S2 $\rightarrow$ S2 $\rightarrow$ S2 $\rightarrow \cdots$, S1 $\rightarrow$ (S2 $\rightarrow$ S4 $\rightarrow$ S6 or S7) $\rightarrow \cdots$, and other cases. We show that the two traitors can be traced in all cases.

Case of S1 $\rightarrow$ S2 $\rightarrow$ S2 $\rightarrow$ S2 $\rightarrow \cdots$. Upon discovery of illegally distributed content, the tracer performs Case 1 transition and the state of C_l moves to S1. Then, if both traitors are included in the same subsets at the times of continuous two segments distribution, e.g. $u_1, u_2 \in C_{\oplus,l} \cap C_{\otimes,l}$, the tracer performs Case 2 transition and the state moves to S2. Similarly, if u_1 and u_2 are in the same subsets in S2, e.g. $u_1, u_2 \in C'_{\oplus,l} \cap C'_{\otimes,l}$, the tracer performs Case 3 transition and the state moves to another S2. Naturally, the number of users included in subset I increases, but the number of users included in $C'_{\oplus,l}$, $\overline{C'_{\oplus,l}}$, $C'_{\otimes,l}$, $\overline{C'_{\otimes,l}}$ and L'_l decreases. Consequently, these two traitors are assigned to two distinct subsets at some time, e.g. $u_1 \in C'_{\oplus,l} \cap C'_{\otimes,l}$ and $u_2 \in \overline{C'_{\oplus,l}} \cap C'_{\otimes,l}$.

Case of S1 $\rightarrow$ (S2 $\rightarrow$ S4 $\rightarrow$ S6 or S7) $\rightarrow \cdots$. If u_1 and u_2 receive different variants such that $v^{2j}_{C_{\oplus,l}}$ and $v^{2j+1}_{C_{\otimes,l}}$ are distributed to u_1 and that $v^{2j}_{\overline{C_{\oplus,l}}}$ and $v^{2j+1}_{\overline{C_{\otimes,l}}}$ are distributed to u_2, it means that u_1 and u_2 are in two distinct subsets, $u_1 \in C_{\oplus,l} \cap C_{\otimes,l}$ and $u_2 \in \overline{C_{\oplus,l}} \cap \overline{C_{\otimes,l}}$. u_1 and u_2 can then use both variants for redistribution, e.g. $v^j_{C_{\oplus,l}}$ and $v^{j+1}_{\overline{C_{\otimes,l}}}$. The tracer then performs Case 2 transition and the state moves to S2. In the S2, u_1 and u_2 belong to L'_l. u_1 and u_2 receive variants $v^{2j+2}_{L'_l}, v^{2j+3}_{L'_l}$, and redistributes them. The tracer performs Case 4 transition and the state moves to S4. u_1 then receives $v^{2j+4}_{L''_l}$ and $v^{2j+5}_{L''_l}$, and u_2 receives $v^{2j+4}_{R''_l}$ and $v^{2j+5}_{R''_l}$. u_1 and u_2 then redistribute $v^{2j+4}_{L''_l}$ and $v^{2j+5}_{R''_l}$, or $v^{2j+4}_{R''_l}$ and $v^{2j+5}_{L''_l}$. Consequently, the tracer performs Case 7 transition, the state moves to S7, and u_1 and u_2 are assigned to two distinct subsets Π_l and Π_ω. If u_1 and u_2 select a variant pair ($v^{2j+4}_{L''_l}$ and $v^{2j+5}_{L''_l}$), or ($v^{2j+4}_{R''_l}$ and $v^{2j+5}_{R''_l}$), the tracer performs Case 6 transition, the state moves to S6, and u_1 and u_2 are assigned to two distinct subsets Π_l and Π_ω.

Other cases. We have to consider other cases such that the same variant of a certain segment is distributed to both u_1 and u_2, and that two distinct variants of the next segment are distributed to them, e.g. $v^{2j}_{C_{\oplus,l}}$ and $v^{2j+1}_{C_{\otimes,l}}$ are distributed to u_1 and $v^{2j}_{C_{\oplus,l}}$ and $v^{2j+1}_{\overline{C_{\otimes,l}}}$ are distributed to u_2, i.e., $u_1 \in C_{\oplus,l} \cap C_{\otimes,l}$ and $u_2 \in C_{\oplus,l} \cap \overline{C_{\otimes,l}}$. u_1 and u_2 redistribute $v^{2j}_{C_{\oplus,l}}$ and $v^{2j+1}_{C_{\otimes,l}}$, or $v^{2j}_{C_{\oplus,l}}$ and $v^{2j+1}_{\overline{C_{\otimes,l}}}$. The tracer then performs Case 2 transition. The result is $u_1 \in C'_{\oplus,l} \cup C'_{\otimes,l}$ and $u_2 \in L'_l$, and receive different variants at the time of the next consequent two segments distribution, u_1 receives $v^{2j+2}_{C'_{\oplus,l}}$ and $v^{2j+3}_{C'_{\otimes,l}}$, and u_2 receives $v^{2j+2}_{L'_l}$ and $v^{2j+3}_{L'_l}$. Accordingly, the tracer performs Case 5 transition, and it can place u_1 and u_2 into two distinct subsets Π_l and Π_ω.

It remarks that, when α variants are necessary to trace one traitor, it is almost impossible to trace traitors if $p \ge \lfloor r/\alpha \rfloor$, since the number of variants is insufficient. On the other hand, in the case of $p < \lfloor r/\alpha \rfloor$, the number of variants is sufficient to trace all traitors. In the above scheme, $\alpha = 3$ and $p < \lfloor r/3 \rfloor$. Hence, the above always holds true under the condition that the consecutively redistributed variants are the ones that distinct two traitors received, who are in two distinct subsets Π_l and Π'_l whose indices are the same, e.g. $\Pi_l = C'_{\oplus,l}$ and $\Pi'_l = \overline{C'_{\oplus,l}}$, and the two traitors can be assigned to two distinct subsets Π_l and Π_ω $(l \ne \omega)$. As a result, when there are multiple traitors in a subset Π_l, they can be placed into two distinct subsets Π_l and Π_ω $(l \ne \omega)$, each of which includes at least one traitor. □

Proof of Theorem 1 (Sketch). Suppose that there are p $(1 \le p \le \lfloor r/3 \rfloor)$ traitors and that multiple traitors belong to one subset Π_l. The tracing algorithm can lead to the situa-

tion such that the traitors in Π_l are assigned to two distinct subsets Π_l and $\Pi_{l'}$ from claim 2. By repeating this process, p traitors can be divided into p distinct subsets $\Pi_1, \cdots, \Pi_p$. Moreover, p traitors, who belong to p distinct subsets, can be traced perfectly from claim 1. Hence, the tracing algorithm can trace all p traitors. □

Discussion: It is important to consider the gap between ideal watermark and non-ideal (real) one since there is actually no ideal watermarking scheme.

In our scheme, a tracer must perform the following processes:

(1) The tracer gets illegally redistributed content through a feedback channel and detects the watermark embedded into the content.
(2) The tracer assigns users into subgroups for the next subsequent segments according to the result of the process (1).
(3) The tracer embeds watermarks into the next subsequent segments of the next content.

When some detection errors occur in the process (1), the new user assignment does not work effectively. That is, the subgroup, which includes a true traitor and should be investigated intensively, is not investigated intensively. For example, suppose that a traitor is in a subset $C_\oplus$ and that a variants of a certain segment $v_\oplus$, into which information $i_\oplus$ is embedded, and $v_{\overline{\oplus}}$, into which information $i_{\overline{\oplus}}$ is embedded, were distributed to $C_\oplus$ and $\overline{C_\oplus}$, respectively. The traitor redistributes a variant $v_\oplus$ and the tracer obtains it. However, watermark detection error occurs and the detected information is $i_{\overline{\oplus}}$. Consequently, the tracer will investigate intensively $\overline{C_\oplus}$ which does not include the traitor and an innocent user would be identified as a traitor. However, some practically available watermark embedding and detecting systems exist [12], [14], [16] and there has been no error report on them at the moment. Therefore, these real watermarking systems can be regarded as close to the ideal one and can be used practically.

Moreover, when considering traitors' behavior (attack), algorithms of the watermarking scheme should be closed. If the entity who embeds watermarks can be the same with the one who detects watermarks, the algorithms can be highly concealed. Consequently, it is possible to protect embedded information from modifying and to protect the information from detection error caused by the attacks. Such a management makes real watermarking schemes closer to the ideal one. However, even if such a management method is employed, there remains low possibility to trace innocent users. We hope that a watermarking scheme that does not have this problem will be developed in the future.

We also have to consider the time span between certain illegal redistribution and the next content distribution that the tracer requires. The tracer has to perform the above (1), (2) and (3) precesses in the span. Actually, the watermarking systems [12], [14], [16] have a real-time property. That is, it takes only the same amount of time with the length of content to perform the process (1) or (3). Furthermore, the required time for process (2) is extremely less than those of the processes (1) and (3). Consequently, the required total time for all processes (1), (2) and (3) is only about twice as long as length of the content. When considering actual illegal redistribution described before, there is enough time for the tracer to perform above all processes since content is redistributed soon after it was broadcast and since a watermark assignment is performed once a day or once a week before the next version of the same program will be broadcast. As a result, our scheme can be practically used.

4.1.2 Delayed Attack Resilience

Our scheme is robust against delayed attacks. The subset assignment for each segment is determined before the next consecutive segments are distributed, and the assignment is recorded in the side information in order to trace traitors. In the worst case, the traitors wait for the distribution to be completed and then start redistribution. However, since distinct subsets are assigned to users for *each* segment, not one piece but multiple pieces of the information to be used for tracing can be obtained and can be used for the next dynamic computation. This is in contrast to DTT, which is completely insecure against a delayed attack.

In addition, both DTT and our scheme require the feedbacks from traitors. However, DTT requires real-time feedback channels, whereas our scheme does not necessarily require them. Therefore, our scheme is more robust against delayed attacks than DTT.

4.2 Costs for Dynamic Computations

We show that our scheme uses fewer dynamic computations than [10], [11].

Number of Dynamic Computations of Our Scheme.
Regarding the number of dynamic computations to trace all p traitors, the following claim is true and we prove it here.

Claim 3: The largest number of dynamic computations of our scheme to trace all p traitors is $p \times (\log_4 n - \log_4 p) + 4p - 3$ if $p < 60$, otherwise $p \times (\log_4 n + 1)$.

Proof of Claim 3. We assume that the attack method is the best for the traitors such that the traitors can redistribute content as long as possible. When traitors find the best attack method, they repeat it, and the same state appears repeatedly. The states S2, S5, S6 and S7 become its candidates. We then calculate the largest number of dynamic computation among the cases in which the states S2, S5, S6 or S7 are appeared repeatedly. Consequently, we derive the conditions for traitors to select one candidate. The number is the maximum number of dynamic computations of our scheme. In the following, we calculate the number of dynamic computations and compare them.

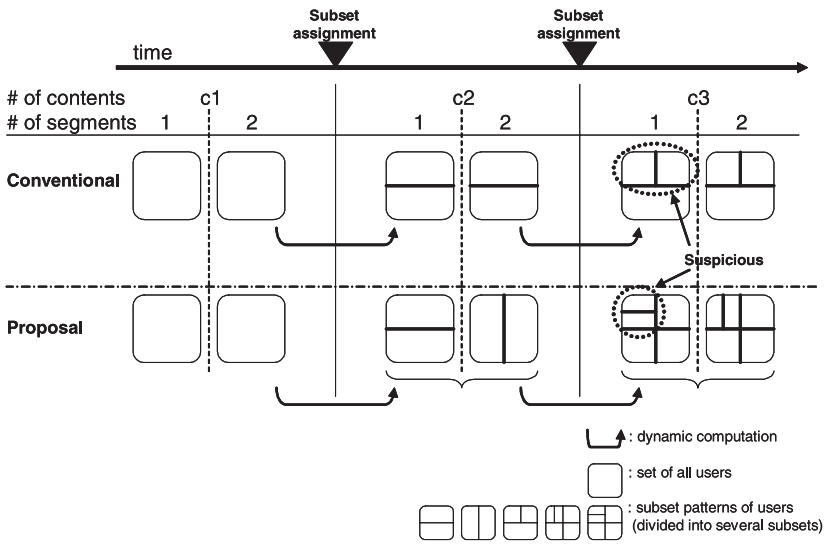

Fig. 6 Model comparison.

Case of S2. One of the traitors redistributes content repeatedly until he/she is identified. The traitors, then, change the members and repeat the same things. Thus, the number of dynamic computations to identify all p traitors is $N_{2,all} := p(\log_4 n+1)$.

Cases of S5, S6 and S7. These three states are similar, since they all have two State1s and one State0. We thus can show only the number of dynamic computations of S6 as a representative.

Before the state becomes S6, three dynamic computations are performed and the existence of at least two traitors is detected. The traitors can then select one attack method out of two. One is that they select State0 for their redistribution; the other is that they select State1-1 or State1-2.

When the traitors select State0, the tracer confirm the existence of new two traitors with four subsequent dynamic computations. When the traitors repeat the same attack, $3+(p-2)/2\times 4=2p-1$ dynamic computations put p traitors into p distinct subsets, and all subsets move to State2. The average number of users in one subset is n/p, when p traitors are put into distinct subsets. This average number means that the number of users in each subset is the same and that the number of dynamic computations necessary to identify all the traitors is the largest. That is, this is the most optimal attack. The number of dynamic computations is then $p \times (\log_4(n/p) + 1)$. Thus the total number of dynamic computations is $N_{6,0,all} := 2p - 1 + p \times (\log_4(n/p) + 1) = p \times (\log_4 n - \log_4 p) + 3p - 1$.

When the traitors select State1-1 or State1-2, the tracer can confirm the existence of one more traitor with less than or equal to three dynamic computations, and $3 + 3 \times (p - 2) = 3p - 3$ dynamic computations are necessary to put p traitors into p distinct subsets. Then, all the subsets move to State2 and $p\times(\log_4(n/p)+1)$ dynamic computations identify p traitors. The total number of dynamic computations is $N_{6,1,all} := 3p - 3 + p \times (\log_4(n/p) + 1) = p \times (\log_4 n - \log_4 p) + 4p - 3$.

Comparing $N_{6,0,all}$ with $N_{6,1,all}$, we see that $N_{6,1,all}$ is larger than $N_{6,0,all}$ if $p \geq 2$. The number of traitors in this case is more than two, and hence, $N_{6,all}$, the largest number of dynamic computations in the case of S6, is $N_{6,1,all}$.

Similarly, in the case of S7, $N_{7,all} = N_{6,all} = p \times (\log_4 n - \log_4 p) + 4p - 3$, and in the case of S5, $N_{5,all} = p \times (\log_4 n - \log_4 p) + 2p - 2$.

Comparison. We compare $N_{2,all}$, $N_{5,all}$, $N_{6,all}$ and $N_{7,all}$. Comparing $N_{5,all}$ with $N_{6,all}$ ($= N_{7,all}$), we see that $N_{6,all}$ is larger than $N_{5,all}$. Comparing $N_{2,all}$ with $N_{6,all}$, we see that

$$\begin{aligned} &N_{6,all} - N_{2,all} \\ &= (p \times (\log_4 n - \log_4 p) + 4p - 3) - (p \times (\log_4 n + 1)) \\ &= 3p - 3 - p \log_4 p \,. \end{aligned}$$

This value changes with p. In fact, we can get the border value $p = 60$ from a computer calculation, and hence, when $p < 60$, $3p - 3 - p \log_4 p > 0$ and when $p \geq 60$, $3p - 3 - p \log_4 p < 0$. That is, when $p < 60$, then $N_{6,all} > N_{2,all}$; otherwise $N_{6,all} < N_{2,all}$. Consequently, the largest number of dynamic computations $max(N_{2,all}, N_{5,all}, N_{6,all}, N_{7,all}) = max(N_{2,all}, N_{6,all})$ is $max(p \times (\log_4 n - \log_4 p) + 4p - 3, p \times (\log_4 n + 1))$, and it is $p \times (\log_4 n - \log_4 p) + 4p - 3$ if $p < 60$; otherwise $p \times (\log_4 n + 1)$. □

Comparing Our Scheme with the Schemes of [10], [11]. Our scheme can decrease the number of suspicious users to 1/4 with one dynamic computation, whereas the conventional scheme [10], [11] can decrease the number only to 1/2 (see Fig. 6). Thus, our scheme can identify all traitors by using only half the number of dynamic computations. However, while our scheme uses up to three variants to identify one traitor, compared with the conventional scheme's two variants, our scheme does *not always* use three variants.

To evaluate these schemes, a conventional scheme us-

ing three variants should be considered, and such an improvement is easy to achieve. It can decrease the number of suspicious users to $1/3$ with one dynamic computation in a way that *always* uses three variants. Such an improvement is described in [1], [2]. Table 1 compares the proposed scheme with the conventional schemes.

Generally, we can assume $0 < p \ll n$, and then $p \log_2 n \simeq p(\log_2 n + 1)$, $p \log_3 n \simeq p(\log_3 n + 1)$, and $p \log_4 n \simeq p(\log_4 n + 1) \simeq p(\log_4(n/p) + 4) - 3$. Hence, our scheme can decrease the number of dynamic computations to about $50\%(= \log_4 2 = (p \log_4 n)/(p \log_2 n))$ of the conventional scheme's. Compared with the improved conventional scheme, it can decrease it to about $79\%(\simeq \log_4 3 = (p \log_4 n)/(p \log_3 n))$. This proves that if the content provider generates enough variants, our scheme is more effective than the conventional scheme.

Comparing Our Scheme with the Schemes of [18], [19]. The conventional scheme [18], [19] requires only one subset assignment of users, and the assignment is static, not dynamic.

Regarding the network cost, the conventional scheme always requires a network capacity in proportion to the maximum number of variants of one segment. That is, it always distributes all variants of a segment, and this requires a large capacity. On the other hand, in our scheme, the required network capacity changes gradually. That is, the number of variants that have to be simultaneously distributed changes from two to $3p + 1$. Our scheme does not require a network capacity in proportion to the maximum number of variants of one segment, but rather one in proportion to the number of variants distributed at a time.

Let us compare the costs of the two schemes. In the worst case of our scheme, the number of variants is $3p + 1$. On the other hand, the conventional scheme has $max(1 + \sqrt{2n}, 2p^2 + 2p - 3)$ variants. Comparing $3p + 1$ with $2p^2 + 2p - 3$, we see that $2p^2 + 2p - 3 > 3p + 1$ holds true for $p \geq 2$. In STT, p is the number of traitors whom the tracer assumes to be colluding and to redistribute content. Hence, $p = 1$ is not likely. Thus, $2p^2 + 2p - 3 > 3p + 1$ always holds true. That is, in the case of $max(1 + \sqrt{2n}, 2p^2 + 2p - 3) = 1 + \sqrt{2n}$, $1 + \sqrt{2n} \geq 2p^2 + 2p - 3 > 3p + 1$ holds true, and in the case $max(1 + \sqrt{2n}, 2p^2 + 2p - 3) = 2p^2 + 2p - 3$, $2p^2 + 2p - 3 > 3p + 1$ holds true. The result is that $max(1 + \sqrt{2n}, 2p^2 + 2p - 3) > 3p + 1$ always holds true, so even in the worst case, our scheme has fewer variants than the conventional scheme and it is less costly.

Table 1 summarizes the above discussion.

Regarding the number of variants (network costs) and the number of dynamic computations, our scheme and DTT and STT have a trade-off relationship; that is why we call our scheme *trade-off traitor tracing*.

5. Conclusion

We proposed the *trade-off traitor tracing* scheme. This scheme requires fewer dynamic computations than the conventional scheme does, and it does not need to make a dynamic computation in real time, since the computation is performed after several segments have been stored. Moreover, our scheme is more resilient against delayed attacks than the conventional scheme.

To reduce network costs, our scheme needs a lot of edge routers to control the distribution of watermark variants, but it is difficult to add the new function to all edge routers on the Internet. However, an overlay network, such as a P2P network, enables a user's terminal to play a the role of a "pseudo-router". In this case, an application software must be installed on each terminal, but that could be done more easily than on each edge router. Therefore, our scheme is practical.

DTT and our scheme require the feedbacks from traitors. However, there is no method for retrieving all pirate copies efficiently. Actually, many broadcast contents are illegally uploaded on the Internet websites, so right holders must look over uploaded contents carefully and check whether they are illegal or not. Therefore, it takes a long time to retrieve a pirate copy. However, real-time feedbacks from traitors are realized on some specific websites. DTT requires real-time feedbacks from traitors, so delayed attacks is effective. On the other hand, our scheme does not necessarily require real-time feedbacks from traitors. Therefore, our scheme is more robust against delayed attacks than DTT.

Finally, in this paper, we discussed only the case in which the maximum number of watermark variants is $3p+1$ and the number of segments is two ($t = 2$). In fact, we can enlarge the number of segments and the number of variants, but in doing so, it becomes very difficult to evaluate the performance. In future, we will evaluate the performance of an enhanced scheme and develop an optimal construction for any parameter setting.

Acknowledgments

We would like to thank the anonymous reviewers for their useful comments.

References

[1] O. Berkman, M. Parnas, and J. Sgall, "Efficient dynamic traitor tracing," Proc. ACM-SODA'00, pp.586–595, 2000.
[2] O. Berkman, M. Parnas, and J. Sgall, "Efficient dynamic traitor tracing," SIAM J. Comput., vol.30, no.6, pp.1802–1828, 2001, full version of [1].
[3] G.R. Blakley, C. Meadows, and G.B. Purdy, "Fingerprinting long forgiving messages," Proc. Crypto'85, pp.180–189, 1985.
[4] D. Boneh and M. Franklin, "An efficient public key traitor tracing scheme," Proc. Crypto'99, pp.338–353, 1999.
[5] D. Boneh and M. Franklin, "An efficient public key traitor tracing scheme," http://crypto.stanford.edu/~dabo/pubs.html, full version of [4].
[6] D. Boneh and J. Shaw, "Collusion secure fingerprinting for digital data," Proc. Crypto'95, pp.452–465, 1995.
[7] D. Boneh and J. Shaw, "Collusion secure fingerprinting for digital

data," IEEE Trans. Inf. Theory, vol.44, no.5, pp.1897–1905, 1998, full version of [6].
[8] B. Chor, A. Fiat, and M. Naor, "Tracing traitors," Proc. Crypto'94, pp.252–270, 1994.
[9] B. Chor, A. Fiat, M. Naor, and B. Pinkas, "Tracing traitors," IEEE Trans. Inf. Theory, vol.46, no.3, pp.893–910, 2000, full version of [8].
[10] A. Fiat and T. Tassa, "Dynamic traitor tracing," Proc. Crypto'99, pp.354–371, 1999.
[11] A. Fiat and T. Tassa, "Dynamic traitor tracing," J. Cryptol., vol.14, no.3, pp.211–223, 2001, full version of [10].
[12] Hitachi, "Contents distribution system with real-time watermarking," http://www.hitachi.co.jp/New/cnews/month/2006/10/1013.html
[13] A. Kiayias and M. Yung, "Traitor tracing with constant transmission rate," Proc. Euroctypt'02, pp.450–465, 2002.
[14] KBS, "Broadcast technology," http://www.kbs.co.kr/aboutkbs/annual_report/download_0607/eng0607_m.pdf
[15] K. Kurosawa and Y. Desmedt, "Optimum traitor tracing and asymmetric schemes," Proc. Eurocrypt'98, pp.145–157, 1998.
[16] Mitsubishi, "Real-time watermarking scheme robust against image encodings," http://www.mitsubishielectric.co.jp/news-data/2003/pdf/0213-a.pdf
[17] K. Ogawa, G. Ohtake, G. Hanaoka, and H. Imai, "Trade-off traitor tracing," Proc. Indocrypt'07, pp.331–340, 2007.
[18] R. Safavi-Naini and Y. Wang, "Sequential traitor tracing," Proc. Crypto'00, pp.316–332, 2000.
[19] R. Safavi-Naini and Y. Wang, "Sequential traitor tracing," IEEE Trans. Inf. Theory, vol.49, no.5, pp.1319–1326, 2003, full version of [18].
[20] G. Tardos, "Optimal probabilistic fingerprint codes," Proc. STOC'03, pp.116–125, 2003.
[21] N. Wagner, "Fingerprinting," Proc. IEEE Symposium on S&P'83, pp.18–22, 1983.

Appendix A: Brief Review of Dynamic Traitor Tracing [10], [11]

A.1 Model of Dynamic Traitor Tracing

In the model of DTT, the next watermark pattern is determined with *adaptive and dynamic* computations in real time, depending on the watermark information detected from illegally redistributed content. Content providers distribute content and then traitors illegally redistribute it. The providers have real-time feedback channels and can see the content currently being redistributed. One piece of content consists of multiple segments and it is possible to generate multiple variants of each segment. Distinct information is embedded in each variant. In addition, users are assigned to multiple subsets, and each subset receives a unique variant. The subsets are dynamically determined after information embedded in an illegally redistributed segment is analyzed, and the new subset is used in the next distribution. Figure A· 1 shows the model.

A.2 Fiat and Tassa's Construction

We show their construction of DTT. The basic strategy is to decrease the number of users in a set, which includes traitors, to $1/2$ by using a function Div2.

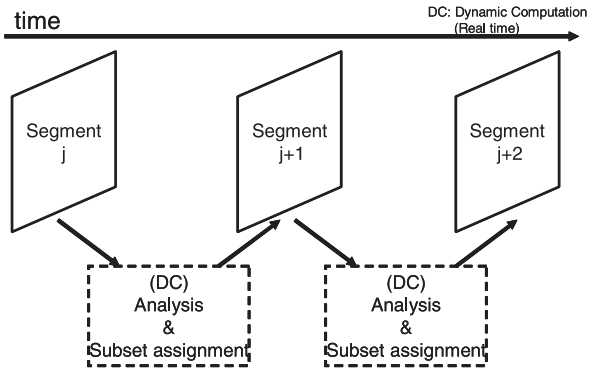

Fig. A· 1 Dynamic traitor tracing.

Div2: It takes as inputs a set of users X, and returns two subsets L and R, where $L \cup R = X$, $|L| \simeq |R| \simeq \frac{1}{2}|X|$, and $L \cap R = \emptyset$.

Construction: The set of users U is partitioned into $2\omega+1$ subsets $U = \cup_{S \in P} S$, where $P = \{L_1, R_1, \cdots, L_\omega, R_\omega, I\}$, and each of those sets receives a distinct variant. I is the subset of users that is not known to include a traitor. These I, ω, and P are initialized to $U, 0, \{I\}$. Then, a distinct variant for every nonempty set of users $S \in P$ is transmitted, and the traitor transmits a variant v.

Case1. If v is associated with I, tracers (content providers) increment ω by one, split I into two subsets L_ω and R_ω, add those sets to P, and set $I = \emptyset$. Namely,

$$\omega \leftarrow \omega+1$$
$$L_\omega, R_\omega \leftarrow \mathsf{Div2}(I)$$
$$P \leftarrow P \cup \{L_\omega, R_\omega\}$$
$$I \leftarrow \emptyset\ .$$

Case2. If v is associated with one of the sets L_l, $1 \le l \le \omega$, then tracers add R_l to I and split L_l into new subsets L_l and R_l. Namely,

$$I \leftarrow I \cup R_l$$
$$L_l, R_l \leftarrow \mathsf{Div2}(L_l)\ .$$

If L_l is a singleton set, content providers suspend distribution to the user included in L_l, add R_l to I, and remove R_l from P. Namely,

$$I \leftarrow I \cup R_l$$
$$P \leftarrow P \setminus R_l\ .$$

If v is associated with R_l, tracers do as above while switching the roles of R_l and L_l.

A.3 Major Shortcoming of Dynamic Traitor Tracing: Delayed Attack

The major shortcoming of DTT is that the regrouping of users and assignment of watermarks to users in each interval depend on the rebroadcast content, also called feedback

from the channel. This means that if there is no feedback from the channel no regrouping will occur and so the system is vulnerable to a delayed rebroadcast attack. In this attack, the attackers do not immediately rebroadcast, but record the content and rebroadcast it with some delay. The broadcaster has no alternative but keep the watermark assignment unchanged.

For example, assume that the distributed content is a motion picture and that it is divided to segments consisting of 500 frames each, and assume that a different watermark is assigned to each segment. To prevent an attack where traitors redistribute the content with a 1,000 frame delay, the provider has to assign watermarks every 1,000 frames, even if it has the ability to assign watermarks every 500 frames.

Appendix B: Example of Trade-off Traitor Tracing

Suppose we want to provide protection for up 16 ($n = 16$) users against up to 2 colluders ($p = 2$), and the user identities are $u_1, \cdots, u_{16}$ and the colluders are u_3 and u_{14}. In addition, assume that the colluders u_3 and u_{14} select variants for illegal redistribution such that the variant for the first segment of each content is the variant distributed to u_3 and the variant for the second segment is the variant distributed to u_{14}.

Table A· 1 shows the relationship between the tracing process and illegally distributed content (variants). For simplicity, we use only $j \in \{1, 2\}$ as the segment number for each content. In addition, we use i as the variant index instead of v_i^j. That is, we only list the indices of the variants.

Initially, when there is no illegal redistribution, all users are in I and the same variants are distributed to all users. Once the tracer finds an illegal redistribution, it assigns users to new subsets for the two segments of the next content according to Case1 transition. That is, for the first segment, $C_{\oplus,1} = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ and $\overline{C_{\oplus,1}} = \{u_9, u_{10}, u_{11}, u_{12}, u_{13}, u_{14}, u_{15}, u_{16}\}$, and thus u_3 and u_{14} are in $C_{\oplus,1}$ and $\overline{C_{\oplus,1}}$, respectively. Similarly, for the second segment, $C_{\otimes,1} = \{u_1, u_2, u_5, u_6, u_9, u_{10}, u_{13}, u_{14}\}$ and $\overline{C_{\otimes,1}} = \{u_3, u_4, u_7, u_8, u_{11}, u_{12}, u_{15}, u_{16}\}$, and thus u_3 and u_{14} are in $\overline{C_{\otimes,1}}$ and $C_{\otimes,1}$, respectively. Accordingly, u_3 receives variant 1 for the first segment and variant 2 for the second, and u_{14} receives variant 2 for the first segment and variant 1 for the second. The colluders select variant 1 that was distributed to u_3 in the first segment and the variant 1 that was distributed to u_{14} in the second, and they illegally redistribute the first content (variant 1 for the first segment and variant 1 for the second) illegally.

When the tracer gets the redistributed first content, it analyzes the content and learns that the variant is 1 for both segments. That is, the tracer learns that at least one traitor is in $C_{\oplus,1}$ and at least one traitor is in $C_{\otimes,1}$. The tracer then assigns users to new subsets for the next two segments according to Case2 transition. That is, for the first segment, $C'_{\oplus,1} = \{u_1, u_2\}$, $\overline{C'_{\oplus,1}} = \{u_5, u_6\}$, $L'_1 = \{u_3, u_4, u_7, u_8, u_9, u_{10}, u_{13}, u_{14}\}$, and $I = \{u_{11}, u_{12}, u_{15}, u_{16}\}$, and thus u_3 and u_{14} are in L'_1. Similarly, for the first segment, $C'_{\otimes,1} = \{u_1, u_5\}$, $\overline{C'_{\otimes,1}} = \{u_2, u_6\}$, $L'_1 = \{u_3, u_4, u_7, u_8, u_9, u_{10}, u_{13}, u_{14}\}$, and $I = \{u_{11}, u_{12}, u_{15}, u_{16}\}$, and thus u_3 and u_{14} are in L'_1. Hence, both u_3 and u_{14} receive variant 3 for the first segment and variant 3 for the second segment of the second content. Accordingly, the colluders can only select variant 3 for the first segment and variant 3 for the second when they redistribute the second content (variant 3 for the first segment and variant 3 for the second).

When the tracer gets the redistributed second content, it learns that the variants are 3 and 3 for the first and second segments. From this analysis, the tracer learns that at least one traitor is in L'_1. However, the conclusion that there is only one traitor and he or she is in L'_1 is contradictory to the analysis of the first content distribution. Thus, the tracer knows that at least two traitors exist. It then assigns users to new subsets for the next two segments according to Case4 transition. That is, for the first and second segments, $C''_1 = \{u_1, u_2, u_5, u_6\}$, $R''_1 = \{u_3, u_4, u_7, u_8\}$, $L''_1 = \{u_9, u_{10}, u_{13}, u_{14}\}$ and $I = \{u_{11}, u_{12}, u_{15}, u_{16}\}$, and thus u_3 is in R''_1 and u_{14} is in L''_1. u_3 receives variant 2 for both segments and u_{14} receives variant 3 for both segments. The colluders select variant 2 that was distributed to u_3 in the first segment and variant 3 that was distributed to u_{14} in the second, and they redistribute the third content (variant 2 for the first segment and variant 3 for the second).

When the tracer gets the redistributed third content, it learns that the variants are 2 and 3 for the first and second segments, respectively. From this analysis, it learns that at least one traitor is in R''_1 and at least one traitor is in L''_1. It then assign users to new subsets for the next two segments according to Case7 transition. That is, for the first segment, $C'_{\oplus,1} = \{u_3, u_4\}$, $\overline{C'_{\oplus,1}} = \{u_7, u_8\}$, $C'_{\oplus,2} = \{u_9, u_{10}\}$, $\overline{C'_{\oplus,2}} = \{u_{13}, u_{14}\}$, and $I = \{u_1, u_2, u_5, u_6, u_{11}, u_{12}, u_{15}, u_{16}\}$, and thus u_3 is in $C'_{\oplus,1}$ and u_{14} is in $\overline{C'_{\oplus,2}}$. Similarly, for the second segment, $C'_{\otimes,1} = \{u_3, u_7\}$, $\overline{C'_{\otimes,1}} = \{u_4, u_8\}$, $C'_{\otimes,2} = \{u_9, u_{13}\}$, $\overline{C'_{\otimes,2}} = \{u_{10}, u_{14}\}$, and $I = \{u_1, u_2, u_5, u_6, u_{11}, u_{12}, u_{15}, u_{16}\}$, and thus u_3 is in $C'_{\otimes,1}$ and u_{14} is in $\overline{C'_{\otimes,2}}$. Accordingly, u_3 receives variant 1 for both segments, and u_{14} receives variant 5 for both segments. The colluders select variant 1 that was distributed to u_3 in the first segment and variant 5 that was distributed to u_{14} in the second, and they redistribute the fourth content (variant 1 for the first segment and variant 5 for the second).

When the tracer gets the redistributed fourth content, it learns that the variants are 1 and 5 for the first and second segments, respectively. From this analysis, it learns that at least one traitor is in $C'_{\oplus,1}$ and at least one traitor is in $\overline{C'_{\otimes,2}}$. It then assigns users to new subsets for the next two segments according to Case3 transition. That is, for the first segment, $C'_{\oplus,1} = \{u_3\}$, $\overline{C'_{\oplus,1}} = \{u_4\}$, $C'_{\oplus,2} = \{u_{10}\}$, $\overline{C'_{\oplus,2}} = \{u_{14}\}$, and $I = \{u_1, u_2, u_5, u_6, u_7, u_8, u_9, u_{11}, u_{12}, u_{13}, u_{15}, u_{16}\}$, and thus u_3 is in $C'_{\oplus,1}$ and u_{14} is in $\overline{C'_{\oplus,2}}$. Similarly, for the second

Table A·1 Example in which u_3 and u_{14} are colluders: transition of distributed variants, illegally redistributed variants, and subset assignment. DV denotes distributed variants, SA denotes assigned subsets and IR denotes illegally redistributed variants. †: this number is proportional to the maximum network cost.

#Content	0				1				2				3			
	SA		DA		SA		DA		SA		DA		SA		DA	
#Segment	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
User																
u_1	I	I	0	0	$C_{\oplus,1}$	$C_{\otimes,1}$	1	1	$C'_{\oplus,1}$	$\overline{C'_{\otimes,1}}$	1	1	C''_1	C''_1	1	1
u_2	I	I	0	0	$C_{\oplus,1}$	$\overline{C_{\otimes,1}}$	1	1	$C'_{\oplus,1}$	$C'_{\otimes,1}$	1	2	C''_1	C''_1	1	1
u_3	I	I	0	0	$C_{\oplus,1}$	$\overline{C_{\otimes,1}}$	1	2	L'	L'	3	3	R''	R''	2	2
u_4	I	I	0	0	$C_{\oplus,1}$	$\overline{C_{\otimes,1}}$	1	2	$\overline{L'}$	L'	3	3	R''	R''	2	2
u_5	I	I	0	0	$C_{\oplus,1}$	$C_{\otimes,1}$	1	1	$\overline{C'_{\oplus,1}}$	$\overline{C'_{\otimes,1}}$	2	1	C''_1	C''_1	1	1
u_6	I	I	0	0	$C_{\oplus,1}$	$C_{\otimes,1}$	1	1	$C'_{\oplus,1}$	$C'_{\otimes,1}$	2	2	C''_1	C''_1	1	1
u_7	I	I	0	0	$C_{\oplus,1}$	$\overline{C_{\otimes,1}}$	1	2	L'	L'	3	3	R''	R''	2	2
u_8	I	I	0	0	$\overline{C_{\oplus,1}}$	$\overline{C_{\otimes,1}}$	1	2	L'	L'	3	3	R''	R''	2	2
u_9	I	I	0	0	$\overline{C_{\oplus,1}}$	$C_{\otimes,1}$	2	1	L'	L'	3	3	L''	L''	3	3
u_{10}	I	I	0	0	$\overline{C_{\oplus,1}}$	$C_{\otimes,1}$	2	1	L'	L'	3	3	L''	L''	3	3
u_{11}	I	I	0	0	$\overline{C_{\oplus,1}}$	$\overline{C_{\otimes,1}}$	2	2	I	I	0	0	I	I	0	0
u_{12}	I	I	0	0	$\overline{C_{\oplus,1}}$	$\overline{C_{\otimes,1}}$	2	2	I	I	0	0	I	I	0	0
u_{13}	I	I	0	0	$\overline{C_{\oplus,1}}$	$C_{\otimes,1}$	2	1	L'	L'	3	3	L''	L''	3	3
u_{14}	I	I	0	0	$\overline{C_{\oplus,1}}$	$C_{\otimes,1}$	2	1	L'	L'	3	3	L''	L''	3	3
u_{15}	I	I	0	0	$\overline{C_{\oplus,1}}$	$\overline{C_{\otimes,1}}$	2	2	I	I	0	0	I	I	0	0
u_{16}	I	I	0	0	$\overline{C_{\oplus,1}}$	$\overline{C_{\otimes,1}}$	2	2	I	I	0	0	I	I	0	0
			↓	↓			↓	↓			↓	↓			↓	↓
IR			0	0			1	1			3	3			2	3
# Var†			1				2				4				4	

#Content	4				5				trace result
	SA		DA		SA		DA		
#Segment	1	2	1	2	1	2	1	2	
User									
u_1	I	I	0	0	I	I	0	0	
u_2	I	I	0	0	I	I	0	0	
u_3	$C_{\oplus,1}$	$C_{\otimes,1}$	1	1	$C'_{\oplus,1}$	$C'_{\otimes,1}$	1	1	
u_4	$C_{\oplus,1}$	$\overline{C_{\otimes,1}}$	1	2	$\overline{C'_{\oplus,1}}$	$\overline{C'_{\otimes,1}}$	2	2	
u_5	I	I	0	0	I	I	0	0	
u_6	I	I	0	0	I	I	0	0	
u_7	$\overline{C_{\oplus,1}}$	$C_{\otimes,1}$	2	1	I	I	0	0	
u_8	$\overline{C_{\oplus,1}}$	$\overline{C_{\otimes,1}}$	2	2	I	I	0	0	
u_9	$C_{\oplus,2}$	$C_{\otimes,2}$	4	4	I	I	0	0	
u_{10}	$C_{\oplus,2}$	$\overline{C_{\otimes,2}}$	4	5	$C'_{\oplus,2}$	$C'_{\otimes,2}$	4	4	
u_{11}	I	I	0	0	I	I	0	0	
u_{12}	I	I	0	0	I	I	0	0	
u_{13}	$\overline{C_{\oplus,2}}$	$C_{\otimes,2}$	5	4	$\overline{I}$	$\overline{I}$	0	0	
u_{14}	$\overline{C_{\oplus,2}}$	$\overline{C_{\otimes,2}}$	5	5	$\overline{C'_{\oplus,2}}$	$\overline{C'_{\otimes,2}}$	5	5	
u_{15}	I	I	0	0	I	I	0	0	
u_{16}	I	I	0	0	I	I	0	0	
			↓	↓			↓	↓	
IR			1	5			1	5	u_3, u_{14}
# Var†			5				5		

segment, $C'_{\otimes,1} = \{u_3\}$, $\overline{C'_{\otimes,1}} = \{u_4\}$, $C'_{\otimes,2} = \{u_{10}\}$, $\overline{C'_{\otimes,2}} = \{u_{14}\}$, and $I = \{u_1, u_2, u_5, u_6, u_7, u_8, u_9, u_{11}, u_{12}, u_{13}, u_{15}, u_{16}\}$, and thus u_3 is in $C'_{\otimes,1}$ and u_{14} is in $\overline{C'_{\otimes,2}}$. Accordingly, u_3 receives variant 1 for both segments, and u_{14} receives variant 5 for both segments. The colluders select variant 1 that was distributed to u_3 in the first segment and variant 5 that was distributed to u_{14} in the second, and they redistribute the fifth content (variant 1 for the first segment and variant 5 for the second).

When the tracer gets the redistributed fifth content, it learns that the variants are 1 and 5 for the first and second segments, respectively. From this analysis, it learns that at least one traitor is in $C'_{\oplus,1}$ and at least one traitor is in $\overline{C'_{\otimes,2}}$. Consequently, the tracer can identify the traitors as being u_3 and u_{14}, since each subset of $C'_{\oplus,1}$ and $\overline{C'_{\otimes,2}}$ has only one user (u_3 or u_{14}).

In this case, each assignment of users is performed before each content is distributed, and hence, the total number

of dynamic computations is five. The number is less than the maximum number $p(\log_4(n/p)+4)-3 = 8$ shown in Table 1. This attack method is not the best one for the colluders.

The network cost is proportional to the number of necessary variants, and we can evaluate it by using the number of variants. In this case, the maximum number of variants is five, and it is less than the number $3p + 1$ shown in Table 1. The network cost is low, since the total number of users is small and the attack method is not the best one for the colluders.

Go Ohtake received the B.E. and M.E. degrees from Tokyo Institute of Technology in 1999 and 2001, respectively. He is a research engineer of NHK Science and Technical Research Laboratories, and he is also a student of Institute of Information Security. His research interests include copyright protection technology, content distribution system, and digital signature.

Kazuto Ogawa received the B.E. and Ph.D. degrees from the University of Tokyo, Tokyo, Japan in 1987 and 2008, respectively. He joined NHK (Japan Broadcasting Corporation) in 1987. He has mainly engaged in the research and development on video image processing systems and digital content rights management systems. He is currently a senior research engineer of NHK Science and Technical Research Laboratories.

Goichiro Hanaoka received his bachelors degree in Electronic engineering from the University of Tokyo in 1997, and received his masters and Ph.D. degrees in information and communication engineering from the University of Tokyo in 1999 and 2002, respectively. From 2002 to 2005 he was a Research Fellow of Japan Society for the Promotion of Science (JSPS). Since 2005 he has been with the National Institute of Advanced Industrial Science and Technology, Japan. He received the Wilkes Award from the British Computer Society in 2007.

Hideki Imai was born in Shimane, Japan on May 31, 1943. He received the B.E., M.E., and Ph.D. degrees in electrical engineering from the University of Tokyo in 1966, 1968, and 1971, respectively. From 1971 to 1992 he was on the faculty of Yokohama National University. From 1992 to 2006 he was a Professor in the Institute of Industrial Science, the University of Tokyo. In 2006 he was appointed as an Emeritus Professor of the University of Tokyo and a Professor of Chuo University. Concurrently he serves as the Director of Research Center for Information Security, National Institute of Advanced Industrial Science and Technology. His current research interests include information theory, coding theory, cryptography, and information security. From IEICE Dr. Imai received Best Book Awards in 1976 and 1991, Best Paper Awards in 1992, 2003 and 2004, Yonezawa Memorial Paper Award in 1992, Achievement Award in 1995, Inose Award in 2003, and Distinguished Achievement and Contributions Award in 2004. He also received Golden Jubilee Paper Award from the IEEE Information Theory Society in 1998, Wilkes Award from the British Computer Society in 2007, and Official Commendations from the Minster of Internal Affairs and Communications in June 2002 and from the Minister of Economy, Trade and Industry in October 2002. He was awarded Honor Doctor Degree by Soonchunhyang University, Korea in 1999 and Docteur Honoris Causa by the University of Toulon Var, France in 2002. He is also the recipient of the Ericsson Telecommunications Award 2005. Dr. Imai is a member of the Science Council of Japan. He was elected a Fellow of IEEE, IEICE, and IACR in 1992, 2001, and 2007, respectively. He has chaired many committees of scientific societies and organized a number of international conferences. He served as the President of the Society of Information Theory and its Applications in 1997, of the IEICE Engineering Sciences Society in 1998, and of the IEEE Information Theory Society in 2004. He is currently the Chair of CRYPTREC (Cryptography Techniques Research and Evaluation Committee of Japan).