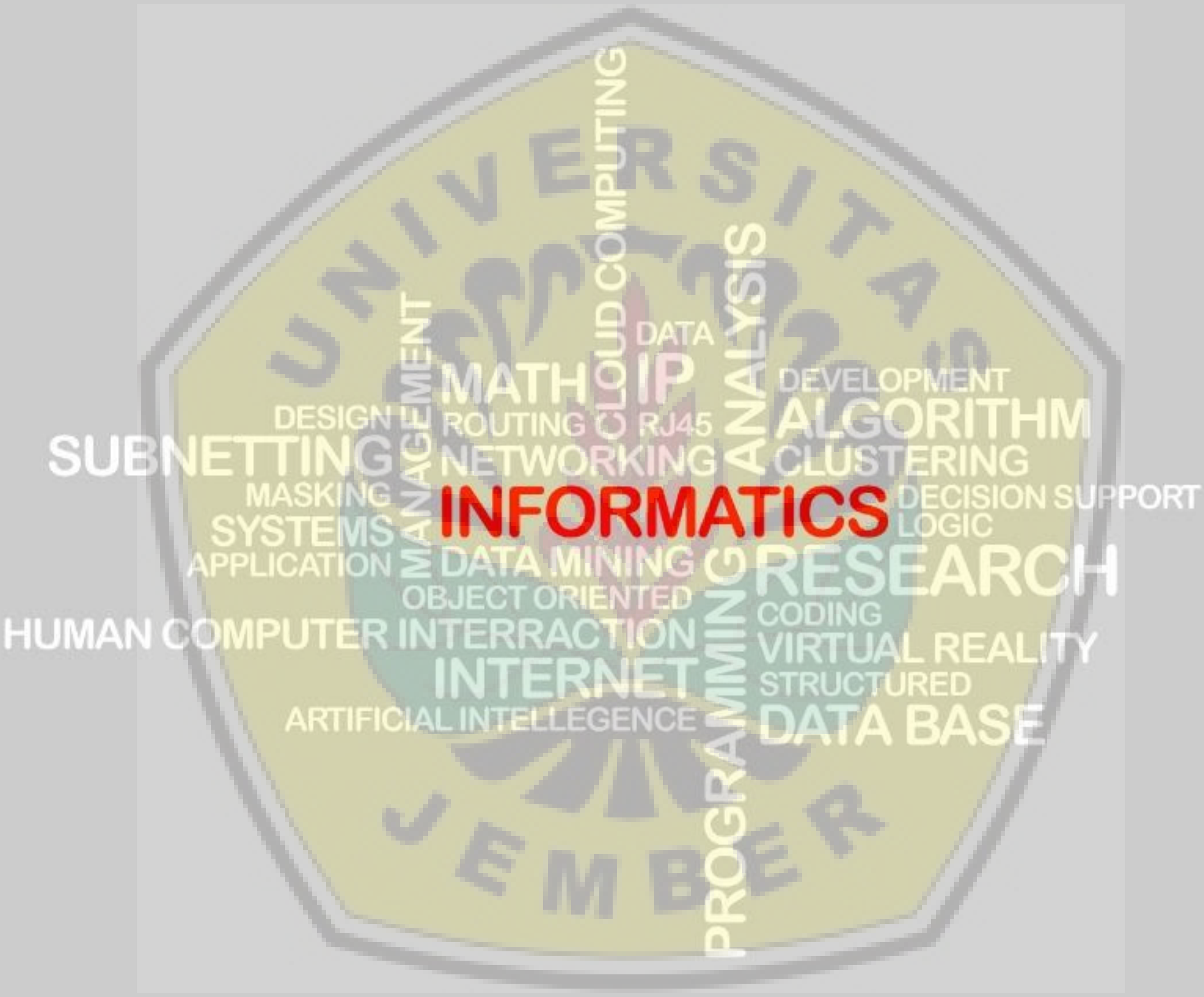
ISSN:2503-250X

INFORMATICS JOURNAL

Volume 3, Number 2, Agustus 2018



INFORMATION SYSTEM DEPARTMENT
UNIVERSITY OF JEMBER

ISSN : 2503-250X

INFORMAL : Informatics Journal

Fakultas Ilmu Komputer
Universitas Jember

Web:
https://jurnal.unej.ac.id/index.php/INFORMAL/issue/view/737
Vol. 3 No. 2, August 2018

ISSN : 2503-250X

INFORMAL : Informatics Journal

Volume 3 Number 2, August 2018

Editorial Team

Journal Contact
Program Studi Sistem Informasi Universitas Jember,
Jl. Kalimantan 37 Jember 68121, Indonesia.
Phone/Fax : +62.331.326911
Email: informal@unej.ac.id

ISSN : 2503-250X

INFORMAL : Informatics Journal

Volume 3 Number 2, August 2018

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Kuasa atas curahan kasih dan rahmatNya, sehingga edisi kedua INFORMAL (Informatics Journal), yakni volume 3 nomor 2 tahun 2018, dapat terbit. Kehadiran jurnal ini diharapkan dapat menambah referensi dalam kegiatan ilmiah. Informatics Journal merupakan salah satu media komunikasi bagi para peneliti di bidang Ilmu Komputer, khususnya pada disiplin Sistem Informasi, Teknologi Informasi, dan Informatika.

Tim Informatics Journal mengucapkan banyak terima kasih atas partisipasi para penulis pada edisi ini. Semoga kerja sama ini akan terus berlangsung pada kesempatan selanjutnya. Kami juga mengundang partisipasi para peneliti lain untuk dapat memberikan kontribusi pada terbitan INFORMAL berikutnya. INFORMAL dijadwalkan terbit setiap catur wulan, yakni pada bulan April, Agustus dan Desember.

Pada terbitan ini masih banyak kekurangan di sana sini. Oleh karena itu, kami mohon saran dan kritik yang membangun untuk lebih baiknya jurnal ini. Akhirnya, semoga INFORMAL dapat memberikan manfaat bagi para penulis dan pembaca sekalian.

Salam,

INFORMAL Editorial Team

ISSN : 2503-250X

INFORMAL : Informatics Journal

Volume 3 Number 2, August 2018

Table of Contents

Traitor Tracing Schemes: a Review

Antonius Cahya Prihandoko
University of Jember, Indonesia

antoniuscp.ilkom@unej.ac.id

ABSTRACT

This paper provides a review on traitor tracing schemes that are developed to counter piracy strategies. The review starts with a formal definition of the traitor tracing schemes. The paper is then outlined based on two main strategies which may taken by digital content pirates. Mostly the pirates have strategy to make use leaked decryption key or leaked decrypted content. For each piracy strategy, we presents traitor tracing schemes that can be used to counter the piracy. We also analysis strength and weakness of each group of the schemes. At the end of this paper, we propose to combine some schemes for better protection.

Keyword: Digital rights management, Traitor tracing schemes, Piracy strategies

1. Introduction (10 PT)

Traitor tracing is a copyright infringement detection system which works by discovering the cause of disclosed information rather than by direct copy protection. This system is studied in different contexts: broadcast encryption and data fingerprinting. In the perspective of broadcast encryption [1], the information is delivered firmly to a dynamic subset of legitimate users over an insecure network. Each user has a decoder with a single distinctive key to decrypt the protected broadcast. The broadcast message is assumed not being decrypted by, or revealed to, unauthorized users. However, a group of traitors may construct a pirate decoder to illegally decrypt the encrypted information. Traitor tracing schemes are intended to trace this pirate decoder. In the environment of data fingerprinting, traitors may use the copies of their content to develop a pirate duplication of the content. In this case, again, traitor tracing schemes are intended to trace one of the colluders. This mechanism may be adopted for general content distribution system. Content provider adds a unique key to each copy given out. When a copy is leaked to the public, the distributor can check the value on it and trace it back to the "leaker".

Generally, a traitor tracing scheme consists of three components: a user initialization scheme, an encryption-decryption scheme, and a traitor tracing algorithm [2].

- The *user initialization scheme* is utilized by the content provider to add new users and allocate a unique personal key for each of them.
- The *encryption scheme* is used by the data supplier to encrypt message. The *decryption scheme* is used by every user to decrypt the message.
- The *traitor tracing algorithm* is utilized upon confiscation of a pirate decoder or a pirate copy of content, to identify the traitor.

Traitor tracing schemes are expected to trace piracy source without harm innocent user. This expectation means that the scheme should be capable to identify the real culpable users correctly. Once the source of piracy has found, its access is disconnected from further content transmissions. The scheme should also be able for supplying legal evidence of pirate's identity and deterring potential traitors.

A traitor tracing scheme must be developed based on the piracy strategy it is itended to counter. Basically, there are two strategies that are likely utilized by traitors to make an illegal access to a protected content [3]. Firstly, traitors may make an effort to obtain the decryption keys to build a pirate decoder. Secondly, they may legitimately decrypt content and then illegally redistribute it for their own profits. As a consequence, traitor tracing schemes can be classified based on these strategies.

2. Schemes to Counter Leaked Decryption Keys Based Traitors

Piracy strategy that makes use of leaked decryption keys can be described as follows. In a content distribution system, some legal users may conspire to combine their secret keys to build a pirate decoder and then sell it to unauthorized users. If all legitimate users were assigned the same key, then this piracy scenario would be totally risk-free for the traitors. Even though the decryption keys are bound to users' identities, the traitors may still make an effort to construct an untraceable key. This strategy is relatively less expensive and, thus, more likely chosen by the pirates to enable illegal mass-access to copyrighted content.

Traitor tracing schemes to counter such a piracy strategy are intended to protect the distribution system against pirate decoders. When a pirate decoder is found, the schemes enable the authorities to trace the identities of the users who contribute to construct the decoder. The schemes can be symmetric or asymmetric. In symmetric schemes, content provider and users share the same keys to encrypt and decrypt content. In asymmetric schemes, encryption and decryption processes use different keys. The decryption key is initially split into two or more shares. Each authorized user has a share as a unique personal key. The combination of the personal key and the other shares enable user to decrypt content. The objective is to keep the decryption key secret and enable users to decrypt content using their traceable personal key.

2.1 Symmetric Schemes

The symmetric schemes use the same session key *s* for both encrypting and decrypting message. The initial formal model of traitor tracing scheme, presented by Chor [3], was implemented symmetrically. In the initial step, the content provider uses a meta-key α to allocate personal keys for each user. The meta-key α defines a mapping $P_\alpha : U \rightarrow \{0,1\}^h$ where *U* is the set of possible users and *h* is the number of bits in the personal key that each user gets. User $u_i \in U$ receives personal key $P_\alpha(u_i)$ which consists of a subset of decryption keys out of a larger set of keys.

To make the session key *s* stay confidential, it is divide into several shares $s_1, s_2, \dots, s_{\lceil log_2 N \rceil}$, such that $s = s_1 \oplus s_2 \oplus \dots \oplus s_{\lceil log_2 N \rceil}$, where N = |U|. The message is encrypted block by block. Each encrypted block comprise an *enabling block* (EB) and a *cipher block* (CB). EB contains encrypted session key shares, while CB loads ciphertext of the message. Each share s_j is encrypted using two keys $k_{j,0}$ and $k_{j,1}$ so that EB containing 2 $\lceil log_2 N \rceil$ sub blocks. In the decryption mechanism, a user decrypts $\lceil log_2 N \rceil$ sub blocks to get the shares. By decrypting a particular sub block of EB, a legitimate user is able to decrypt all s_j and hence obtain the session key *s*. The user then can use *s* to decrypt the cipher block.

Since each user is assigned a unique personal key, if a user reveals his personal key, he can be traced from the exposed key. A cospiracy of *t* dishonor users may disclose information of their keys to an adversary so that the adversary is able to build a pirate decoder. Upon confiscation the decoder, the traitor tracing algorithm can be utilized to identify a traitor. The algorithm is assumed not able to view the contents of such a decoder, but rather it can access the decoder as a black-box. The decoder is tested how it decrypts an input ciphertext.

2.2 Asymmetric Schemes

Asymmetric traitor tracing schemes have the same components as the symmetric ones, but use different keys to encrypt and decrypt the message *m*. Suppose *e* and *d* are the encryption and decryption keys, respectively. Typically, *d* is secret. An approach to protect the secret is by splitting *d* into two components: δ_{ID} and d_{ID} [4]. Personal string δ_{ID} is constructed from a unique identifier ID, while the secret value d_{ID} is determined in such a way to fulfill $d = R(\delta_{ID}, d_{ID})$. *R* denotes the combining function, such as XOR. Additionally, a public key cryptosystem, such as RSA, is used to implement the encryption and decryption processes.

2.2.1 Basic Scheme

Initially, content provider establish a pair of encryption and decryption keys *(e,d)*. A new subscriber has to provide a unique identifier *ID*. The system then computes $\delta_{ID} = f(ID)$, where *f* is a specific function defined by the content provider. The subscriber then receives the pair of δ_{ID} and d_{ID}. To disclose a protected content *c*, the user inputs δ_{ID} into the decoder which will decrypts *c* in two steps:

- compute $d = R(\delta_{ID}, d_{ID})$;
- decrypt *c* using *d*.

The mechanism to trace a traitor is simple. Suppose that an authorized user is suspected to illegally duplicate his decoder. When the pirate decoder is found, it can be examined whether it relates to the user *ID*. First of all, a string $\delta_{ID} = f(ID)$ is computed from the putative traitor's identifier. Together with a valid ciphertext *c*, δ_{ID} is entered to the pirated decoder. If this decoder decrypts *c* correctly, then the suspected user is identified as the traitor. The knowledge of the derivation function *f* is required to trace the traitor. If *f* is public, anyone can evaluate whether a given decoder relates to any identifier *ID*. Otherwise, the tracing capability is exclusively possible for authorized parties.

2.2.2 Sharing the Secret

The security of a traitor tracing scheme depends on how secure splitting the decryption key *d*. In general, for most public-key cryptosystems, the fundamental algebraic structure can be utilized to undertake splitting process with different properties.

Consider an RSA cryptosystem[5] of modulus *n=pq* where *p* and *q* are two large primes. A public key e is relative prime to $\phi(n)$ and relates to a private key *d*, such that $ed \approx 1\ mod\ \phi(n)$. $\phi(n)$ denotes the Euler function. For an RSA modulus $n\ =\ pq$, $\phi(n)\ =\ lcm(p\text{-}1,q\text{-}1)$. By construction, if a ciphertext $c = m^e\ mod\ n$, then the plaintext *m* can be reconstructed as $c^d\ mod\ n$.

The secret key *d* can be split in three different ways: additive, multiplicative and Euclidean splitting [4].

Additive Splitting

The secret *d* can be additively split into two shares (d_1, d_2) where

- d_1 is a random element in $Z_{\phi(n)} - \{0\}$;
- d_2 is computed as $d_2 = d - d_1 mod\ \phi(n)$.

Consider to the basic scheme, given an identifier *ID* and the corresponding δ_{ID}, the value of d_{ID} is computed as $d_{ID} \cong d - \delta_{ID}$ mod $\phi(n)$. d_{ID} is secretly placed inside the decoder and δ_{ID} is required by the decoder as an input to decrypt a ciphertext *c*. Remarking that $d = \delta_{ID} + d_{ID}\ mod\ \phi(n)$, the decryption process is performed as follows.

- compute $c_0 = c^{\delta_{ID}}\ mod\ n$
- compute $c_1 = c^{d_{ID}}\ mod\ n$
- compute the message $m = c_0 c_1 mod\ n$

Multiplicative Splitting

In the multiplicative method, the secret *d* is split into two shares (d_1, d_2) as follows.

- d_1 is a random element in ${Z_{\phi(n)}}^*$, where ${Z_{\phi(n)}}^* = Z_{\phi(n)} - \{1\}$;
- d_2 is computed as $d_2 = \frac{d}{d_1} mod\ \phi(n)$.

This splitting method is implemented to the basic scheme as follows. Given an identifier *ID* and the corresponding δ_{ID}, the value of d_{ID} is computed as $d_{ID} \cong d/\delta_{ID}$ mod $\phi(n)$. d_{ID} is secretly placed inside the decoder and δ_{ID} is used as an input in the decryption process. As *d* can be reconstructed as a multiplication of δ_{ID} and d_{ID}, then to decrypt a ciphertext *c*, the decoder performs the following protocol:

- compute $c_0 = c^{\delta_{ID}}\ mod\ n$
- compute the message $m = {c_0}^{d_{ID}}\ mod\ n$

Euclidean Splitting

With the additive splitting, two half exponentiations are performed in a parallel way, while the multiplicative splitting prescribes a serial operation. Euclidean splitting combines these two methods and leads to parallel and sequential operations in the decryption process. In this method, *d* is split into two shares (d_1, d_2) through the following stages.

1. d_1 is a random element in $\{0.1\}^K$, $d_1 \neq 0$, for some parameter K;
2. $d_2 = d_{2,h}||d_{2,1}$ is computed as $d_{2,h} = \lfloor\frac{d}{d_1}\rfloor$ and $d_{2,1} = d\ mod\ d_1$

Thus, *d* can be revealed as $d = d_1 . d_{2,h} + d_{2,1}$.

In the implementation, given an identifying value δ_{ID}, the value d_{ID} is defined as $d_{ID} = ({d_{ID}}^1, {d_{ID}}^2)$, where ${d_{ID}}^1 = \lfloor d/\delta_{ID} \rfloor$ and ${d_{ID}}^2 = d\ mod\ \delta_{ID}$. With input δ_{ID}, to decrypt *c*, the decoder performs the following protocol.

1. compute $c_0 = c^{\delta_{ID}}\ mod\ n$;
2. compute the message $m = {c_0}^{{d_{ID}}^1} c^{{d_{ID}}^2}\ mod\ n$.

2.3 Security Analysis

The schemes presented above require the highest level obfuscation and tamper-resistant technique. If the value of d_{ID} is obtained, then the information of δ_{ID} allows the reconstruction of the secret key *d*. In the decryption process, the schemes involve modular exponentiations, with an exponent other than d_{ID}, i.e. δ_{ID}, to compute c_0. However, δ_{ID} is not a sensitive value so that the value of c_0 does not reveal any sensitive information.

An attackers is supposedly a legitimate user. The attacker may attempt to construct an untraceable decoder, or at least a decoder that does not trace his identity. A possible way to obtain such a decoder is by recovering d_{ID} and then *d* from δ_{ID}. Since the attacker has a decryption software, he can use it to produce chosen-ciphertext attacks. Consequently, the fundamental cryptosystem has to meet the idea of unbreakability under chosen-ciphertext attacks. In addition, there is no size recommendation for δ_{ID}, but it must be unique. However, d_{ID} is recommended to be at least of the size of $n^{1/2}$ [4].

Collusion attacks does not apply in this scenario. A collusion attack occurs when a coalition of users attempt to generate an untraceable decoder using their personal keys. However, the only personal key that is owned and knowledged by a user is his *ID*. Knowledge of some δ_{ID} does not provide useful information to any coalition of users as δ_{ID} is unrelated to the secret decryption key *d*.

3. Schemes to Counter Leaked Decrypted Content Based Traitors

Another piracy scenario is rebroadcasting the decrypted content. A traitor may first subscribes to the system. As an authorized user, the traitor can legally decrypt protected content in the system. The traitor then delivers the decrypted content to his own group of consumers for profit purposes. This scenario, however, is expensive because it needs the establishment of a self-governing broadcasting infrastructure. In addition, rebroadcasting a decrypted content has a higher risk of being found out. Nevertheless, this piracy strategy must also be anticipated.

An approach for tracing traitor who rebroadcast content is to embed a distict mark for every single user. This approach, however, needs high bandwidth for delivering different copies to different users. To minimize this requirement, a watermark could be allocated to a group of users, instead of to an individual user. With this watermark allocation scenario, the tracing scheme can be made effective when it is run dynamically.

3.1 Dynamic Tracing Schemes

Dynamic tracing scheme [6] allows content provider to discover all traitors using less bandwidth. In this system, content is partitioned into sequential segments. A watermarking method, such as spread-spectrum technique [7], can be used to embed one of q marks inside each segment, hence resulting q versions of a segment. q is referred to as the *watermarking alphabet size*. Watermarking algorithm is supposed to be robust and the embedded marks are unchangeable. In every cycle of broadcasting, a group of users is divided into q disjoint subgroups and every subgroup is given a version of a segment. The subgroups are modified in each interval using rebroadcasted content. Whenever one of the distributed versions is rebroadcasted, it indicates that the corresponding subgroup contains a traitor. The system then replaces the allocation versions to the subgroup, and starting a new cycle. This scheme assumes the existence of an efficient group key management system that allows content provider to efficiently modify groups and to firmly distribute the assigned version. Finally, the gathered information allows the scheme to find and detach all traitors.

To control which version users receive for every segment, the scheme uses the following setting [2].

- every user has a unique symmetric key in common with the **center**, the source of content and its watermarked copies.
- If user i is to get version l of segment j, then prior every segment transmission the center sends an individually encrypted transmission to user i containing key $K_l{}^j$. All such keys are generated randomly.
- The center then transmits multiple versions of the *j*th segment, where version *l* is encrypted under key $K_l{}^j$.

Implementation of this scheme requires two broadcasting elements: particular transmission key for each segment and multiple broadcasting versions of every segment. The later is a high overhead element because it multiplies the total bandwidth by the number of versions.

Some mechanisms can be utilized to reduce the overhead. First of all, instead of using individually encrypted transmission, one can use broadcast encryption schemes [1]. Next, between segments, it is not necessary to change keys for all users. The change is needed when a set of users is divided into two or more subsets, or some sets are united. Furthermore, the transmission of multiple versions of a segment is expensive. To reduce bandwidth overhead, marking may not be implemented in the whole content. For example, even if only 10 % of a movie is watermarked and protected, the pirate experiences problems. A pirate copy that misses 10 % of the movie will not be valuable.

The dynamic schemes of Fiat and Naor [6] were improved by Berkman *et al* [8] who used an undirected graph to represent their algorithms. In each cycle, the algorithms partition the group of users into disjoint subgroups, and give all users in the same subgroup a common version of the current segment from a set of version *C*. If a subgroup is assigned version $c \in S$, the subgroup is said to be *colored* by *c*. If a pirate redistributes color *c*, then the algorithm receive the color *c* as an *answer*. This answer shows that one of the *c*-colored users must be a traitor. A traitor can be *identified* if only a single user is colored by *c*, and *c* is an answer.

Consider an undirected graph *G=(V,E)*, where *V* is the sets of all vertices and *E* is the set of all edges. Each vertex represents a subgroup of users, and every single user be affiliated with exactly one vertex. If there is an edge *(X,Y)*, then the subgroup $X \cup Y$ has a traitor in several previous cycles, and the *answer* was the color of some subset $Z \subseteq X \cup Y$. A vertex *I* represents the subgroup of innocent users, that is the subgroup of users which is unknown to contain a traitor at the present stage.

The basic algorithm to trace the traitors is as follows [8]:

1. Begin with a graph *G=(V,E)* with *I=U*, *V={I}*, E=ϕ, and the number of disjoint edges, *t=0*.
2. Repeat forever:
   a. Find a vertex *X* that contains a traitor.
   b. If *X=I*, split *I* into two new vertices of (almost) equal size, and connect them by an edge. Set *I* = ϕ and *t = t+1*.
   c. Otherwise, let *Y* be the vertex that is connected to *X* by an edge. Set $I = I \cup Y$, split *X* into two vertices of (almost) equal size, and connect them by an edge.

Dynamic tracing schemes have two drawbacks. Firstly, modifying groups and assigning marks to users in every cycle relies on the rebroadcasted content (a.k.a *feedback* from the channel). If there is no feedback from the channel, no regrouping will occur and so the system is susceptible to a *delayed rebroadcast attack*. In such an attack, the attackers may not rebroadcast a version immediately, but rather record and rebroadcast it with some delay, and thus, the broadcaster has no information and keep the mark allocation unchanged. Ultimately, the system fails to trace any traitor. Secondly, the dynamic tracing needs high real-time computation for regrouping the users and assigning marks to subgroups. As a consequent, the length of a segment cannot be short. Safavi-Naini [9] proposed a sequential tracing scheme to overcome these shortcomings.

3.2. Sequential Tracing Schemes

The sequential tracing scheme [9] operates the same scenario as the dynamic tracing, but uses a distinc mark allocation method. In this scheme, the channel feedback is only utilized for tracing traitors, while marks allocation in each interval is undertaken based on a predefined table no matter the channel feedback, so that the system is protected against the postponed rebroadcast attack. Although rebroadcast is postponed until the entire content is transfered, at least one traitor will be identified. Traitors are traced sequentially, means that when a traitor is found, he is removed from the system and the process continues to find the remaining traitors. Additionally, all computations on group key management are performed as pre-computation. These mechanisms, therefore, will minimize real-time computation.

Initially, content is partitioned into segments. A *q*-ary watermarking system, $\mathrm{W} = \{1,2,\ldots,\mathrm{q}\}$, is utilized to develop *q* versions of each segment. These versions are distributed to the group of users $U = \{u_1, u_2, \ldots, u_N\}$ based on a *mark allocation table M*. The mark allocation table $M = (m_{ij})$ is an $N \times L$ array over *W*, where m_{ij} is the mark assigned to the user u_i in segment *j* and *L* is the *convergence length* of the tracing algorithm, that is, the number of steps required by the algorithm to identify all traitors.

3.2.1 Mark Allocation Table

The mark allocation table *M* is constructed as follows [9]. Let $\mathrm{W} = \{1,2,\ldots,\mathrm{q}\}$ be the set of marks, *b* and *m* be integers where $b \le q$. Consider a set of functions $\phi = \{\phi_{ij} | 1 \le i \le b, 1 \le j \le m\}$ where $\phi_{ij}: W \to W$. These functions are utilized to develop *b* row blocks of *M*; each consists of *q* rows and $L = m + 1$ columns. Let M_0 and $\phi_{ij}(M_0)$ be the following $q \times 1$ matrices:

$$M_0 = \begin{pmatrix} 1 \\ 2 \\ \vdots \\ q \end{pmatrix} \qquad \phi_{ij}(M_0) = \begin{pmatrix} \phi_{ij}(1) \\ \phi_{ij}(2) \\ \vdots \\ \phi_{ij}(q) \end{pmatrix}$$

The mark allocation table *M* is defined as follows.

$$M = \begin{pmatrix} M_0 & \phi_{11}(M_0) & \phi_{12}(M_0) & \dots & \phi_{1m}(M_0) \\ M_0 & \phi_{21}(M_0) & \phi_{22}(M_0) & \dots & \phi_{2m}(M_0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ M_0 & \phi_{b1}(M_0) & \phi_{b2}(M_0) & \dots & \phi_{bm}(M_0) \end{pmatrix}$$

In *j*-th time interval, the broadcaster uses the *j*-th column of *M* to allocate marks to users. To achieve the tracing ability, the function $\phi_{ij}: W \to W$ must satisfy the following properties.

1. For a fixed *j* and a pair of the first indices (i_1, i_2), if $i_1 \neq i_2$, $\phi_{i_1 j}(x) \neq \phi_{i_2 j}(x)$ for all $x \in W$.
2. For a pair of the first indices (i_1, i_2) and a pair of second indices (j_1, j_2) with $j_1 \neq j_2$, if $\phi_{i_1 j_1}(x_1) = \phi_{i_2 j_1}(x_2)$, then $\phi_{i_1 j_2}(x_1) \neq \phi_{i_2 j_2}(x_2)$, for all distinct $x_1, x_2 \in W$.

The set function ϕ produces a mark allocation table consisting of $N = \frac{(p-1)^2}{2}$ rows and $L = 1 + \frac{(p-1)}{2}$ columns. The table can be used to trace *t* traitors, where $t = \lfloor \frac{-1+\sqrt{3+2p}}{2} \rfloor$. Therefore, given the number of users *N*, to trace at most *t* traitors, *p* must be chosen such that $p \ge \max\left(1 + \sqrt{2N}, 2t^2 + 2t - 1\right)$

3.2.2 Tracing Algorithm

A group of traitors T may choose one of their versions and rebroadcast it. The tracer intercepts the rebroadcast, extracts the mark, and adds it to a *feedback sequence*, F. Consider the set

$$W_j(T) = \{f_j | f_j \in \{m_{ij} | u_i \in T\}\}$$

The feedback sequence $F = \{f_1, f_2, \dots, f_L\}$ is called *c-feedback sequence* if there exists $T \subseteq U$, $|T| \leq t$, such that $f_j \in W_j(T)$ for $j = 1,2, \dots, L$.

In step j, f_j is received from the channel. The rows that have f_j in their j-th position will be incremented. Let F_j denotes the subset of the first j elements of F and $\rho(F_j, u_i)$ the number of matches between F_j and u_i. A tracing function A is defines as follows.

$$A(F_j) = \{u_i | \rho(F_j, u_i) = t + 1\}$$

That is, when a row reaches $t + 1$ matches, the corresponding user is identified as a traitor. The mark allocation table M and the tracing function A define a sequential t-traceability scheme, that is, a sequential traitor tracing scheme which is able to trace at most t traitors, for $N = bq$ users with convergence length is $L = m + 1$, and $t = \lfloor \frac{-1+\sqrt{5+4m}}{2} \rfloor$. The tracing algorithm identifies one of the t traitors in $t^2 + 1$ steps, and all traitors in at most $t^2 + t$ steps.

4. Discussion on Previous Schemes

The obvious difference between tracing schemes to counter leaked decryption key and leaked decrypted content is the thing that they focus to allocate to users. The former allocate the different personal key to each user, while the latter focus on mark allocation for each user or group of users. Though they have the same ultimate goals: capture at least one traitor, their mechanisms work in different perspectives. The former tracing schemes can be utilized to identify a traitor who contributes on construction of a pirate decoder, but cannot identify traitors who redistribute the decrypted content. In this case content decrypted by all users may have the same perceptibility. Conversely, the latter schemes can be employed to trace traitors who rebroadcast the decrypted content, but not those who construct an illegal decoder as all users may use the same key to decrypt the content.

Tracing traitors may be implemented statically or dynamically. In a static scheme, keys or marks are assigned only once and remain unchange during the lifetime of the content. This approach is suitable if entire content is delivered in one package, such as DVD movie. Only when a black-market copy is discovered, the tracing and incrimination algorithms are performed. However, performance in such a rigid setting is less efficient and less useful as there are few effective countermeasures. The only recourse is the legal action post-factum. In most cases, the static schemes can only be promised to capture one traitor as the keys present in a decoder might all belong to only one of the traitors.

A dynamic scheme replaces keys or marks allocation at particular intervals of the content lifetime to anticipate the real-time action of a pirate. This approach is appropriate if content is delivered online, such as in case like a pay TV broadcast. The pirate may rebroadcasts the content, such as, on the internet. To enable tracing, the scheme assumes online feedback from the pirate subscribers to the content provider. The provider can see the current pirate broadcast and adapt its watermark distribution in the next segments to trace the traitors efficiently. The dynamic scheme allows immediate disconnection of the traitors and is able to trace all traitors.

A further improvement of the dynamic scheme, such as the sequential tracing ~\cite{safavi-nainietal2003}, is a hybrid scheme. The hybrid scheme integrates the existing approaches: allocating the keys or marks statically and tracing traitors dynamically. This scheme can solve the problem of postponed rebroadcast attack and high real-time computations in the dynamic scheme.

The dynamic tracing scheme and its improver, the sequential scheme, promise a more efficient traitor tracing. However, they are designed to counter the traitors who make use of the leaked decryption content, which is a likely rare piracy strategy. On the other hand, the piracy strategy that make use of the leaked decryption key, which is the most likely to rise, is mostly countered by static schemes. Inspired by this issue, an new idea to improve the traitor tracing scheme is come.

5. Proposed Scheme

At glance, the improvement notion is to combine the dynamic and static schemes in a more general objective. This notion makes use of both personal key and watermark allocations. Suppose $U = \{u_1, u_2, \dots, u_N\}$ is the set of users and *W={1,2,...,q}* is the set of the marks. Each user u_i is assigned a unique personal key $P(u_i)$. The protected content is divided into segments and q-ary watermarking system is used to produce q versions of each segment. The set of users is partitioned into q disjoint subsets, where each subset is entitled by a different version of the segment. By this setting, upon confiscation a pirate decoder, the traitor can be discovered by identifying the personal keys utilized to build the decoder. Similarly, when a rebroadcasted content is found, the broadcaster can be identified by extracting mark from the content.

The above setting, however, is feasible when content is broadcasted online. How about when entire content is delivered once in one package? In this case, the combination of both personal key and watermark can still be used, but tracing mechanism cannot be said dynamic. Tough all keys and marks allocation procedures can be proceed as the previous setting, the tracing and incrimination algorithms are activated only when a black-market copy is discovered. Nevertheless, the improvement setting can be utilized to counter both piracy strategies that make use of the leaked decryption key and the leaked decrypted content.

In the initial stage, content provider allocates a unique personal key for each user and a unique version of each segment of the content for each subgroup of users. The key allocation is done statically. In this scheme, if the number of users is in the form $N = 2^x$, for any positive integer x, then there is no way for traitors to develop an untraceable pirate decoder, because any coalition will results a key that relates to an existing user. In this case, however, a group of traitors may erroneously incriminate or may intentionally counterfeit an innocent user. Regardless the traitors' motivation, they yield a key that relates to the secret decryption key d. On the other hands, such a collusion attacks does not work in the scheme, because the only personal key that is owned and knowledged by a user is his *ID*. Knowledge of some δ_{ID} is useless for any coalition as δ_{ID} is unrelated to the decryption key d. Therefore, we prefer to adopt this *ID* based key allocation mechanism.

Content's version allocation is also done statically using a mark allocation table as the sequential tracing scheme. The table utilized by the sequential scheme is constructed based on the determination of the maximum number of traitors. As a consequence, a mark allocation table can be used to trace only a certain number of traitors. When the number of traitors increases, a new table needs to be developed. Therefore, though the mark allocation table makes the sequential tracing scheme robust to a *delayed rebroadcast attack* and able to minimize real-time computation, it is less practical as the number of traitors is dynamically changed in the real application. We need to find out how it is possible to allocate marks without knowledge of the number of traitors.

Utilizing the *ID* based key allocation mechanism, the encryption-decryption stage can be done symmetrically or asymmetrically. Regardless the encryption key, the key allocation mechanism only focuses on splitting the secret decryption key. With this properties, the scheme is flexible and more applicable. Though the key and marks allocation are static, the traitor tracing needs to be dynamic. Dynamic tracing enables the system to immediately disconnect a user from further content distribution once he is identified as a traitor.

6. Conclusion

We have classified some existing traitor racing schemes based on the piracy strategies they are aimed to counter. They have the same purposes, but work in different perspectives. The schemes that are designed to counter the leaked decryption keys can identify a naughty user who involve on developing a pirate decoder, but cannot trace traitors who share the decrypted content. Conversely, the schemes that are focused to counter the leaked decrypted content have capability to trace legal users who redistribute the decrypted content, but not those who construct an illegal decoder.

We proposed to improve traitor tracing schemes by combining the dynamic and static schemes. Our scheme is flexible and more applicable. The key and marks are allocated statically, while tracing traitors is undertaken dynamically.

References

[1] A. Fiat and M. Naor, “Broadcast encryption,” *Adv. Cryptol. - Crypto ’93*, vol. 773, pp. 480–491, 1993.

[2] B. Chor, A. Fiat, M. Naor, and B. Pinkas, “Tracing traitors,” *IEEE Trans. Inf. Theory*, vol. 46, no. 3, pp. 893–910, 2000.

[3] N. Fazio, “On Cryptographic Techniques for Digital Rights Management,” 2006.

[4] M. Joye and T. Lepoint, “Traitor tracing schemes for protected software implementations,” *Proc. 11th Annu. ACM Work. Digit. rights Manag. - DRM ’11*, p. 15, 2011.

[5] M. O. Rabin, “Digitalized Signatures and Public-Key Functions as Intractable as Factorization.” p. 20, 1979.

[6] A. Fiat, “Dynamic traitor tracing,” *J. Cryptol.*, vol. 14, no. 3, pp. 211–223, 2001.

[7] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, “Secure spread spectrum watermarking for images, audio and video,” *Proc. 3rd IEEE Int. Conf. Image Process.*, vol. 3, pp. 185–206, 1996.

[8] O. Berkman, M. Parnas, and J. Sgall, “Efficient Dynamic Traitor Tracing.” pp. 1–21, 2000.

[9] R. Safavi-Naini and Y. Wang, “Sequential traitor tracing,” *IEEE Trans. Inf. Theory*, vol. 49, no. 5, pp. 1319–1326, 2003.

9 772503 250015