3-2010

A Multi-key Pirate Decoder against Traitor Tracing Schemes

Yongdong WU
Institute for Infocomm Research

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Wu YD, Deng RH. A multi-key pirate decoder against traitor tracing schemes. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 25(2): 362–374 Mar. 2010

A Multi-Key Pirate Decoder Against Traitor Tracing Schemes

Yong-Dong Wu[1] (吴永东), *Member*, *IEEE*, and Robert H. Deng[2] (邓慧杰), *Member*, *IEEE*

[1] *Institute for Infocomm Research, A*STAR, 138632, Singapore*

[2] *School of Information Systems, Singapore Management University, 178902, Singapore*

E-mail: wydong@i2r.a-star.edu.sg; robertdeng@smu.edu.sg

Received December 15, 2008; revised October 9, 2009.

Abstract In this paper we introduce an architecture for a multi-key pirate decoder which employs decryption keys from multiple traitors. The decoder has built-in monitoring and self protection functionalities and is capable of defeating most multiple-round based traitor tracing schemes such as the schemes based on the black-box confirmation method. In particular, the proposed pirate decoder is customized to defeat the private key and the public key fully collusion resistant traitor tracing (FTT) schemes, respectively. We show how the decoder prolongs a trace process so that the tracer has to give up his effort. FTT schemes are designed to identify all the traitors. We show that decoder enables the FTT schemes to identify at most 1 traitors. Finally, assuming the decoder is embedded with several bytes of memory, we demonstrate how the decoder is able to frame innocent users at will.

Keywords broadcast encryption, multi-key pirate decoder, traitor tracing

1 Introduction

In a broadcast encryption system, each authorized user has a legal decoder embedded with a unique decryption key. A content distributor encrypts broadcast content such that only authorized users can decode the protected content with their legal decoders. However, the rogue authorized users may violate copyright protection policies by sharing their decryption keys, constructing and distributing pirate decoders — illegal devices which are not registered with the broadcaster but are able to decrypt protected content. To deter authorized users from leaking their decryption keys, the broadcaster could confiscate suspicious pirate decoders, and analyze the behaviors of the decoders in question so as to arrest the traitors①.

1.1 Pirate Decoder

In the Broadcast Encryption And Traitor tracing (BEAT for short) model, traitors disclose their own decryption keys towards the construction of a pirate decoder. Kiayias and Yung[1-2] categorized pirate decoders in two orthogonal dimensions based on their initial state and self-protection attributes. The former refers to the internal (e.g., memory) and external (e.g., system clock in software decoder) states of the pirate decoder when an input message is received for the first time, while the latter is an indication on the decoder's anti-tracing capability.

Based on the initial state attribute[1], a decoder is "resetable" or "stateless" if it can be reset to its initial state after each decryption trial; otherwise, the decoder is "history-recording" or "stateful". Thus, a "resetable" decoder cannot have any nonvolatile memory (or simply memory) to record the information of previous trials. Conversely, a "history-recording" decoder must have memory, whether it is internal or environmental, to remember previous queries submitted by the tracer and uses this history information to evade tracing.

On the other hand, according to the self-protection attribute, if a pirate decoder does not employ an internal reactive mechanism, it is called "available", otherwise, called "abrupt". The first "abrupt" method is the "aggressive action" mechanism[1] which crashes the host tracing system, or releases a virus. The weakness of this method is that it is not able to entirely prohibit tracing, especially in Virtual-Machine protection environment. The second "abrupt" method is the "shutting down" mechanism[3] which erases all internal decryption keys in order to halt the tracing process.

Regular Paper

①Traitor: an authorized user who discloses his/her personal decryption key such that unauthorized users are able to decode encrypted messages.

This "suicidal" approach apparently renders the decoder useless. If the broadcaster can disseminate messages to make permanently shut down pirate decoders, the copyright protection goal of the distributor is almost achieved. Hence, from the viewpoint of pirate, the "aggressive action" method is more suitable for software decoders, while the "shutting down" mechanism is good for hardware decoders.

The third "abrupt" method is the "blind" mechanism[4] which outputs ambiguous messages to confuse the tracer. To this end, a pirate decoder analyzes the input message, and then takes defensive action. For example, if the decoder detects invalidate ciphertexts in its input[5-7], it outputs noise. This mechanism is applicable to both software decoders and hardware decoders.

1.2 Piracy Prevention

To defeat pirate decoders so as to protect copyright, a broadcaster may employ either the pre-prevention method or the post-tracing method. The former tries to increase traitors' difficulty in producing pirate decoders while the latter attempts to identify at least one traitor from a confiscated pirate decoder.

As a pre-prevention technique, the self-enforcement method (e.g., [8-9]) binds users' credit card numbers (or other sensitive data) with their decryption keys. Self-enforcement assumes that the traitors are unwilling to disclose such sensitive personal information to others and thus makes it harder for pirates to obtain users' keys. However, crafty users may sidestep the self-enforcement mechanism by wittingly making their sensitive data obsolescent, e.g., reporting stolen of credit cards before conspiratory. Albeit its attractiveness and simplicity, there is currently very few satisfactory self-enforcement solutions.

Contrary to the few pre-prevention methods available in the literature, post-tracing schemes are abundant. Generally, three major post-tracing methods are digital fingerprinting, black-box confirmation and traitor tracing. Digital fingerprinting technology (e.g., [10-13]) embeds a unique label into the digital content so as to identify the illegal distributor. Since this technology is tightly related to watermarking, and it is orthogonal to the objective of this paper. Black-box confirmation (e.g., [5, 14-16]) is used to confirm whether or not a subset of users are traitors or not, is a time-consuming process, and therefore is ineffective when the number of traitors is large. Traitor tracing can be further classified into white-box , black-box and gray-box methods. The white-box method (e.g., [15]) assumes that the tracer knows the internal detail of the decoder; in the black-box tracing method (e.g., [17-21]), the tracer treats the decoder as a black-box, and executes trials by sending craft input to the suspect decoder and observing the output from the decoder; the gray-box method assumes partial knowledge on the internals of the decoder (e.g., the encrypted key for content for decryption in [22]). The main challenge in the white-box and gray-box methods are reverse-engineering the decoder which is beyond scope of the paper. We will focus on the black-box traitor tracing method in the following.

Most traitor tracing schemes are only effective when the number of traitors is much smaller than the number of authorized users. The notion of a fully collusion resistant traitor tracing (FTT) scheme was recently put forward by Boneh, Sahai and Waters[23], and Boneh and Waters[24]. An FTT scheme effectively identifies the traitors even if all authorized users collude in producing the pirate decoder.

In a private key FTT scheme, the tracer must know a private tracing key and therefore is assumed to be trusted by all users. Boneh, Sahai and Waters[23] introduced a primitive called private linear broadcast encryption (PLBE) and showed that any PLBE gives a private key FTT scheme, refereed to as FTT1 scheme here. However, private key traitor tracing schemes suffer from the *customer's right problem*②[25]. A public key FTT scheme solves the above problem and allows anyone to run the tracing algorithm. Boneh and Waters[24] presented a primitive called Augmented Broadcast Encryption (ABE) that is sufficient to construct public key FTT schemes, called FTT2 schemes in this paper. FTT2 schemes are resistant to an arbitrary number of colluders and are secure against adaptive adversaries.

1.3 Features of BEAT Systems

In any BEAT (Broadcast Encryption And Traitor tracing) system, a broadcaster encrypts a message M into a ciphertext C, then transmits C and some decryption tokens over a broadcast channel. A user can correctly decrypt/decode C if she/he is possession of a legal decoder. A traitor tracing algorithm in a broadcast encryption system identifies traitors from a confiscated pirate decoder. A broadcast encryption system with traitor tracing should have the following features:

1) *Large User Population.* Broadcast (e.g., encrypted satellite radio broadcast[23]) can deliver a message to a group of users simultaneously over a single broadcast channel, but it incurs much more overhead

②In a private key traitor tracing scheme, since the broadcaster knows all the decryption keys, he may create a pirate decoder to frame any user at will.

than unicast communications in terms of setting up and management cost. Hence, the larger the user group is, the more efficient utilization of the broadcast channel is. A typical broadcast application is characterized by a large number of users N (e.g., $N \geqslant 10^6$ in [3, 17]).

2) *Easy Accessibility of Pirate Decoders.* In order to start a tracing process, the tracer has to confiscate a pirate decoder. This pre-requisition can be met when pirate decoders are easily accessible.

3) *Guaranteed QoS.* QoS (Quality of Service) is important in broadcast. For example, video stream should be of high quality; otherwise, users will not subscribe and pay the service. Hence, the broadcaster can only insert a small amount of control data (e.g., the message for shutting down pirate decoders in [3]).

4) *Structured Message.* Messages transmitted in broadcast applications targeted for designated decoders which are designed to process structured messages following international or industrial standards. For example, content sent to VCD decoders follows format specified by MPEG-2. Messages transmitted in either broadcast mode or traitor tracing mode must follow the predefined structure or format, since unstructured messages may crash legal decoders and/or alert pirate decoders that traitor tracing is underway. For this reason, it seems that the traitor tracing scheme in [7] is not practical since it assumes that a pirate decoder is not able to distinguish random data from structured messages.

Table 1. Notations and Abbreviations

N	The number of users
U_j	The j-th user
$\mathcal{D}_j$	The decoder of the j-th user
n	The actual number of collusion traitors
t	The maximum number of tolerable traitors
U_{π_j}	The j-th traitor, $\pi_j < \pi_{j+1}$
$\mathbb{T}$	The set of the traitors
$\mathcal{D}$	The pirate decoder
M	Plaintext message
C	Ciphertext in broadcasting/tracing period
M_j	Output of decoder $\mathcal{D}_j$
τ	Initialization time of a decoder
ϵ	The minimal decoding probability
λ	A security parameter
BEAT	Broadcast encryption and traitor tracing
DOT	Denial of trace
QoS	Quality of service
FTT	Fully collusion resistant traitor tracing
FTT1	Private FTT[23]
FTT2	Public FTT[24]

5) *Realistic Tracing Cost.* A practical traitor tracing scheme must allow a tracer to find traitors at an affordable cost either in terms of time (e.g., within the life-time of broadcast system) or in terms of budget (e.g., at a small fraction of the operating cost of the broadcast system). It is simply not viable if a tracer has to spend 10 years or $10 millions in trying to identify a traitor. For this reason, we think it is inappropriate to evaluate tracing cost based on notions in computational complexity. For example, despite the complexity of the hybrid coloring tracing algorithm is polynomial (e.g., $O(N^3 \log^2 N)$[1]), its exact cost could be too formidable to bear by the tracer.

1.4 Our Contribution

We present a generic pirate decoder and show how the decoder can be used to foil the traitor tracing schemes including FTT1 and FTT2. Specifically:

1) We introduce an architecture for a multi-key pirate decoder which employs decryption keys from multiple traitors. The decoder has built-in monitoring and anti-tracing functionalities. The monitoring functionality detects whether or not the decoder is under investigation. Once the decoder decides that it is being traced, the anti-tracing functionality takes appropriate self-protection countermeasures. Among them, our novel denial-of-trace (DOT) mechanism forces the tracing process taking a very long time to complete such that the tracer is left no choice but gives up the arm-race game. Table 2 summarizes traitor tracing schemes which are vulnerable to the DOT attack.

Table 2. DOT Attack Effect

Schemes	Number of Trials	Trial Time[a]
CFNP[b]	$32t^8 \log N \log^2(2t^2)$	221
[5][c] [9, 14]	$100\binom{N}{t}$	700
[23]	$8\lambda N^2 \log N/\epsilon$	700
[24]	$8\lambda N^3/\epsilon^2$	1.5×10^5

[a] The trial time in years given $\tau = 1$ minute, $N = 100$, $t = 4$, $\epsilon = 0.1$, and $\lambda = 100$.

[b] CFNP means [19] + [1].

[c] To indicate the black-box confirmation algorithm since [1, 4] break the tracing algorithm in [5].

2) We tailor our generic multi-key pirate decoder to evade both FTT1 and FTT2. First we show that with our decoder, both FTT1 and FTT2 only allow the tracer to identify at most 1 traitor, instead of identifying all traitors as they are designed to be. Next, assuming the decoder is embedded with just several bytes of memory, we demonstrate how the decoder is able to mislead the tracer to frame innocent users at will. We also improve FTT1 and FTT2 to prevent this framing attack.

The rest of the paper is organized as follows. Section 2 presents the architecture of the generic

multi-key pirate decoder. DOT attacks on black-box confirmation schemes and on combinatorial-key traitor tracing schemes are given in Sections 3 and 4, respectively. Section 5 illustrates how to customize our generic pirate decoder to exploit the weaknesses of FFT1. Section 6 addresses FTT2 and its vulnerabilities. Section 7 presents a stateful version of the decoder to launch framing attack. In Section 8, we discuss DOT attack to the general traitor tracing schemes. Section 9 draws our conclusion.

2 Multi-Key Pirate Decoder

In this section, we introduce a multi-key pirate decoder. Not only can it detect the tracing activity, but also take anti-tracing action accordingly so as to defeat several traitor tracing schemes such as [3, 5, 9, 20, 23-24].

2.1 Structure of Pirate Decoder

A pirate decoder should be as useful as a genuine decoder, i.e., it is able to decrypt the broadcast data. According to the definition in [14, 18, 24], a pirate decoder is useful if it can decode protected messages with probability

$$\Pr(M' = M) \geqslant \epsilon \tag{1}$$

for any message M, the output M' of the pirate decoder, and some threshold ϵ.

As an improvement of the pirate decoders of [1, 4], Fig.1 is the structure of the present multi-key pirate decoder $\mathcal{D}$ which integrates traitors' decoder D_{π_i} with some components: MUX, RST, WATCH, ACT and NVM. Each $\mathcal{D}_{\pi_i}$ may be duplicated from an original decoder, and whose output M_{π_i} is indistinguishable from the original decoder; MUX component distributes the input ciphertext C to each traitor's decoder; RST unit is necessary to start the decoder; WATCH unit decides whether the decoder is being traced or not; ACT takes action based on the decision. Optionally, NVM unit is used to log the usage history.

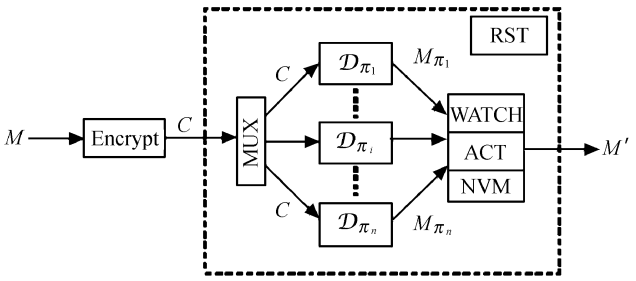

Fig.1. Multi-key pirate decoder structure. The components in the dash box constitute the pirate decoder. $\mathcal{D}_{\pi_1}, \ldots, \mathcal{D}_{\pi_n}$ are the duplicated decoders of the traitors.

Although the single-key pirate model is simple but realistic in the pirate market[26], the multi-key pirate decoder is easy in designing, practical in manufacturing with present semi-conductor technology, robust in defeating the traitor tracing algorithms, and cost-effective in comparing with the expensive broadcast subscription fee.

2.2 Workflow of Pirate Decoder

When a decoder is power-on or re-started, the RST unit is activated so as to load the program code, check the memory, and initialize the variables. In addition, RST unit may execute some delay code for the sake of DOT (denial-of-trace, see Subsection 2.3) attack. In reality, it takes some time to run the RST unit.

Next, the pirate decoder $\mathcal{D}$ is on standby status. When it receives a ciphertext C generated from message M, the MUX component distributes the ciphertext C to each traitor's decoder $\mathcal{D}_{\pi_j}$; each decoder $\mathcal{D}_{\pi_j}$ will decrypt C independently, and output M_{π_i} to the WATCH unit; WATCH unit will compare the results $M_{\pi_1}, \ldots, M_{\pi_n}$ and make a decision in accordance to the traitor-tracing scheme.

If the decision is that the ciphertext is the normal broadcast data, it will output the decrypted message (e.g., M_{π_1}). Clearly, in the broadcast period, the multi-key pirate decoder will output the original message M since all the M_{π_i} are identical. That is to say, the pirate decoder meets (1), and thus it is useful.

Conversely, if the decision indicates that the decoder is being traced, ACT may take actions to frustrate the tracing. The anti-tracing approaches include escape tracing, frame innocent users, output noise[4], erase the keys[1,3], and start the following DOT attack.

2.3 Denial of Trace

Independent from the previous three "abrupt" self-protection mechanisms introduced in Subsection 1.1, this paper presents a fourth mechanism called as DOT. DOT attack will be activated when the pirate decoder concludes that it is traced or the tracer has to reset the decoder.

Generally speaking, the important parameters which determine the performance of DOT are τ and T_{trace}, where τ, the time for either decoder initialization or intentional delay, is related to the toleration time for a legal user to start a decoder; while T_{trace}, the time to finish the tracing process, is determined by the tracing budget and the lifetime of the decoder. Thus, if the tracer resets the decoder after each trial to make sure the decoder is stateless (e.g., [23]), the maximum

number of budgeted tracing trials is

$$\omega = T_{\text{trace}}/\tau.$$

If the number of tracing trials is greater than ω, the tracer will give up tracing. Therefore, any traitor tracing algorithm for stateless decoder may be vulnerable to DOT attack if it demands for a lot of trials to accuse a traitor. In this paper, we will employ DOT attack to defeat black-box confirmation schemes, combination-key traitor tracing, FTT1 and FTT2 tracing methods.

3 DOT Attack to Black-Box Confirmation

In each trial of black-box confirmation algorithm (e.g., [5, 14-16]), a tracer selects a set $\mathbb{T}$ of at most t suspected users, and creates a specific ciphertext by which the decoders of the users in $\mathbb{T}$ should have different output from those of other users. By observing the output, the tracer can tell whether $\mathbb{T}$ does include all the traitors that cooperated to construct a given pirate decoder $\mathcal{D}$. Hence, black-box confirmation might have to go through all $\binom{N}{t}$ coalitions to do full-fledged tracing. In order to elaborate the DOT attack to black-box confirmation algorithm, let us use the scheme in [14] as an example.

3.1 Overview of [14]

3.1.1 Key Setup

The broadcaster selects a cyclic multiplication group G of order q, and two secret random polynomials

$$A(x) = a_0 + a_1 x + \cdots + a_{2t} x^{2t}, \quad \text{and}$$
$$B(x) = b_0 + b_1 x + \cdots + b_{2t} x^{2t}.$$

The public key is $PK = \texttt{KeyGen}(A(x), B(x))$ given a designated key generation algorithm $\texttt{KeyGen}(\cdot)$. When a user subscribes the broadcast service, she is securely delivered a secret key $\langle x_i, A(x_i), B(x_i)\rangle$ with a unique $x_i \in [0, q-1]$.

3.1.2 Encryption

With the public key PK, the broadcaster encrypts M ($M \in \mathsf{G}$) with a probabilistic encryption function $\mathcal{E}(\cdot)$ such that the ciphertext

$$C = \mathcal{E}(PK, M).$$

The broadcaster sends C to the users. For simplicity, we ignore the decryption function.

3.1.3 Black-Box Confirmation

After confiscating a suspicious decoder, the tracer will repeatedly confirm the suspect set of users as Fig.2.

(a) Select a suspect set $\mathbb{T} \subset \{U_1 \ldots U_N\}$, whose size is at most t. Repeat the following steps for Ω_1 times.
- Fake two random $(2t)$-degree polynomials $A_f(x)$ and $B_f(x)$ such that $\forall U_i \in \mathbb{T}, A_f(x_i) = A(x_i)$, $B_f(x_i) = B(x_i)$.
- Generate $PK_f = \texttt{KeyGen}(A'(x), B'(x))$ as a new fake public key. Clearly, only the users in subset $\mathbb{T}$ whose secret keys match both PK_f and PK.
- Encrypt a message M_f as $C_f = \mathcal{E}(PK_f, M_f)$ with the fake public key PK_f. Thus, only the users in subset $\mathbb{T}$ can decrypt C_f.

(b) If the suspect decoder decrypts C_f into M_f at a "large enough" probability, the tracer can confirm that $\mathbb{T}$ includes all the traitors.

(c) By refining $\mathbb{T}$, the tracer can arrest at least one traitor.

Fig.2. One trial in traitor tracing algorithm[14]. A full-fledged tracing algorithm needs $\binom{N}{t}$ trials.

3.2 DOT Attack to [14]

Since black-box confirmation requires a lot of trials, it is vulnerable to DOT attack in nature. For a stateless pirate decoder, the most important task in starting DOT attack is how to activate the RST unit automatically, or equally, how to detect the tracing activity. Indeed, it is easy for the present multi-key pirate decoder to finish the task with one of the following two detecting methods. For each trial, 1) if the current public key is different from the old one (i.e., the decoder should record the old public key), it means that the decoder is being traced and the tracer does not reset the decoder; or 2) if the WATCH unit observes that not all the $\{M_{\pi_i}\}_{i=1}^{t}$ are the same (i.e., $\mathbb{T}$ does not include all traitors), it concludes the pirate decoder is inspected. The first detecting method is simple while the second one is flexible in adjusting reset rate and applicable to all black-box confirmation methods. Each detecting method can instruct the ACT unit to reset the pirate decoder. Therefore the total tracing time is at least

$$T_{\text{trace}} = \Omega_1 \binom{N}{t} \tau.$$

Fig.3 shows the restriction on the user size and traitor size even if $\Omega_1 = 1$. It demonstrates that black-box confirmation is difficult in protecting broadcast content since all suitable N are very small. For example, suppose $\tau = 1$ minute, the maximum number of traitors is $t = 4$, and the budget tracing time $T_{\text{trace}} = 3$ years $\approx 1.6 \times 10^6$ minutes, the black-box tracing algorithm requires at least $W = 1.6 \times 10^6$ trials, the maximum number of users is only $N \approx 70$ which is too small to be viable in real broadcast applications. Furthermore, Ω_1 should be much larger than 1 so as to reduce the error probability, thus, N and t should be

much smaller than those shown in Fig.3.

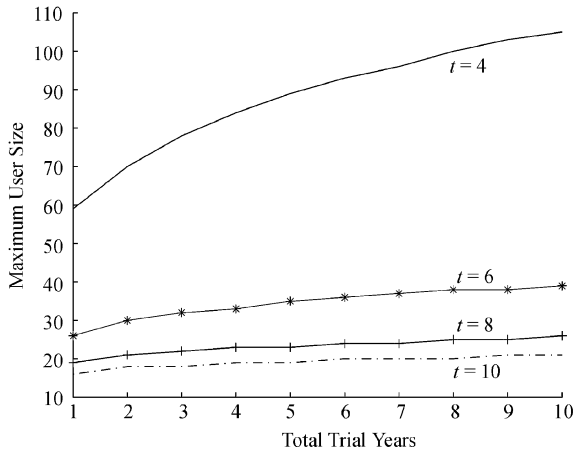

Fig.3. Relationship among the maximum user size N, the traitor size t and affordable trace time T_{trace} due to DOT attack ($\tau = 1$ minute).

3.3 Discussion on DOT Countermeasure

To defeat the "shutting down" protection of pirate decoder, the broadcaster may deliver invalid messages so as to disable all the pirate decoders forever[3]. Similarly, as a countermeasure on DOT attack, the broadcaster may send invalid messages so as to reset a pirate decoder, and hence reduce the decoding probability of pirate decoder.

Let us study the countermeasure performance with a video stream broadcast example. As in the schemes[10-12], the video stream is divided into segments of θ minutes each, and each segment is protected with a broadcast encryption scheme (e.g., [14]) independently. Since θ is determined with the capacity of the legal decoder (e.g., CPU speed or cost), network efficiency, and the content distribution attack shown in [10-12]. As a tradeoff, we select $\theta = 0.5$ minutes. Define

p_0: the probability of broadcasting an invalid message;

p_1: the resetting probability of the pirate decoder;

p_2: the decoding probability of a pirate decoder by considering the missing traffic in resetting period;

p_3: the decoding probability of a legal decoder.

Then

$$p_1 = p_0(1 - 1/\binom{N}{t}) \approx p_0$$

$$p_2 = 1 - p_1(\lceil \tau/\theta \rceil) \approx 1 - p_0(\lceil \tau/\theta \rceil) = 1 - 2p_0$$

$$p_3 = (1 - p_0) + p_0 \left(1 - \binom{N-1}{t} \Big/ \binom{N}{t}\right) \approx 1 - p_0$$

$$\Delta = p_3 - p_2 \approx p_0.$$

Therefore, p_2 is only slightly smaller than p_3 since p_0 must be small enough to guarantee the video quality. As a result, the countermeasure by sending invalid messages cannot obviously reduce the availability of the pirate decoder. Furthermore, the pirate decoder can adjust the resetting probability so as to balance the DOT attack and availability.

4 DOT Attack to CFNP Scheme

As a seminal work, Chor *et al.* addressed the broadcast encryption scheme by assigning a key-set to each user, and constructed a traitor tracing method given that the traitors' keys are known to the tracer in advance. But Chor *et al.*[17] did not explain how the tracer can obtain these keys. Fortunately, paper [19] elaborated the process of key retrieving so that it is possible to provide a full BEAT scheme. We call the full scheme as CFNP in the following.

4.1 Overview of CFNP

The broadcast encryption scheme in [19] includes two modes: single-level and multi-level. Since the multi-level mode is merely an extension of the former so as to reduce the network overhead, we just investigate the single-level mode in the following. The attack method is also applicable to the multi-level mode.

4.1.1 Setup

The broadcaster generates a key matrix $\boldsymbol{K}$ of size $l \times b$ as

$$\boldsymbol{K} = \{K_{ij}\}_{l \times b} = \begin{pmatrix} K_{11} & K_{12} & \dots & K_{1b} \\ K_{21} & K_{22} & \dots & K_{2b} \\ \vdots & & \vdots & \\ K_{l1} & K_{l2} & \dots & K_{lb} \end{pmatrix}.$$

For every row i of matrix $\boldsymbol{K}$, each user is assigned one of the keys $\{K_{i1}, \dots, K_{ib}\}$ at random. Consequently each user is assigned an l-tuple of keys, or one key from each row of the matrix $\boldsymbol{K} = \{K_{ij}\}_{l \times b}$.

4.1.2 Encryption

In order to broadcast a message M to a group of users, the broadcaster randomly divides M into l "shares" s_i, i.e.,

$$M = s_1 \oplus s_2 \oplus \cdots \oplus s_l \tag{2}$$

where $\oplus$ is an operator for bitwise exclusive OR. With an encryption scheme $\mathcal{E}(\cdot)$, the broadcaster generates a

ciphertext of M as

$$\boldsymbol{C} = \begin{pmatrix} \mathcal{E}(K_{11}, s_1) & \mathcal{E}(K_{12}, s_1) & \cdots & \mathcal{E}(K_{1b}, s_1) \\ \mathcal{E}(K_{21}, s_2) & \mathcal{E}(K_{22}, s_2) & \cdots & \mathcal{E}(K_{2b}, s_2) \\ \vdots & & \vdots & \\ \mathcal{E}(K_{l1}, s_l) & \mathcal{E}(K_{l2}, s_l) & \cdots & \mathcal{E}(K_{lb}, s_l) \end{pmatrix}. \tag{3}$$

The ciphertext matrix $\boldsymbol{C} = \{c_{ij}\}_{l \times b}$ will be broadcast to all the users. Since each user has one key for each row, upon receiving $\boldsymbol{C}$, she can obtain all the shares s_i, $i = 1, 2, \ldots, l$, and recover M with (2).

4.1.3 Black-Box Tracing

In order to find the traitors involved in the pirate decoder, a tracer will deduce the decryption keys as shown in Fig.4.

(a) **for** $i = 1$ to l,

- for $j = 1$ to $b - 1$,
  repeat the following trial for Ω_2 times
  - reset the suspicious decoder, select a random M,
  - create a matrix $\boldsymbol{C}$ with (3), and replace $c_{i1}, c_{i2}, \ldots, c_{ij}$ with random numbers.
  - send $\boldsymbol{C}$ into the suspected decoder, record the output as $\tilde{M}$. Let $\hat{p}_{ij}$ be the fraction of trial times such that $\tilde{M} = M$.
- $\forall j$, if $\hat{p}_{i(j-1)} - \hat{p}_{ij}$ is "large enough", the tracer concludes K_{ij} is present in the suspect decoder.

(b) The tracer will obtain at least l keys.

(c) Since the number of traitors is at most t, there is at least one user U who contributes at least l/t keys in the pirate decoder. The tracer identifies U is a traitor.

Fig.4. Traitor tracing algorithm in CFNP[19].

4.2 DOT Attack to CFNP

To make sure that CFNP scheme is resistant to at most t traitors, Chor *et al.*[19] selected $b = 2t^2$, and gave a theoretical lower-bound $l = 4t^2 \log N$. Meanwhile, they stated that l should be $O(t^6 \log N)$ so as to explicitly construct their schemes with [27]. Although Ω_2 is not expressed in [19], Kiayias and Yung[1] chose $\Omega_2 = b^2 \log^2 b$ such that the framing error probability is negligible. With these parameters, we can analyze the DOT attack to CFNP. Since the decoder is reset for each trial, the total time in tracing the pirate decoder $\mathcal{D}$ is

$$\begin{aligned} T_{\text{trace}} &= l \times (b-1) \times \Omega_2 \times \tau \\ &= 4t^2 \log N \times 2t^2 \times (2t^2)^2 \log^2(2t^2) \times \tau \\ &= 32t^8 \tau \log N \log^2(2t^2). \end{aligned}$$

Let us select conservative parameters $N = 100$, $t = 4$ (Note: the example values $N = 10^6$, $t = 500$ in [19]), and $\tau = 1$ minute. The total tracing time is

$$T_{\text{trace}} > 1.14 \times 10^9 \text{ minutes } \approx 213 \text{ years!}$$

4.3 Discussion on Countermeasure

Similar to the countermeasure in Subsection 3.3, the broadcaster may broadcast invalid message so as to reset the pirate decoder. Unfortunately, since this countermeasure is a two-edge sword and it may degrade the QoS for the legal users, its effect is not significant either.

One possible countermeasure is that the tracer does not reset the decoder so as to undermine DOT attack. In the original CFNP scheme, why does the tracer reset the suspect decoder before each trial such that the decoder is stateless? The answer is that the tracer may obtain a key which is not in the traitors' key-set from the non-resetting decoder. As an example, given that the pirate decoder has keys K_{11} and K_{1b}, when the tracer scans the first row, the pirate decoder may output correct message for the second trial (i.e., decryption with K_{1b}), but incorrect message for the third one (i.e., decryption with K_{11}). The tracer concludes $\hat{p}_{12} - \hat{p}_{13} = 1$, hence K_{13} is one of traitors' keys. As a result, the trace fails. With further analysis, we know that it is not useful yet if the tracer resets the pirate decoder in a batch of trials.

5 DOT Attack to FTT1

Constructed with bilinear maps in groups of composite order, FTT1 is the first fully collusion resistant tracing traitors system whose ciphertext is of size $O(\sqrt{N})$ and private key is of size $O(1)$. Nonetheless, its private key property enables the pirate decoder to detect the tracing activity. This section overviews and analyzes FFT1.

5.1 Overview of FFT1

In the private fully collusion resistant traitor tracing scheme (or FTT1 for short)[23], Boneh *et al.* proposed a Private Linear Broadcast Encryption (PLBE) primitive, and employs PLBE to design traitor tracing algorithm `Trace`$(\cdot)$.

5.1.1 PLBE Algorithm

Like other cipher systems, PLBE consists of a set of cryptographic functions `Setup`$(\cdot)$, `Encrypt`$(\cdot)$, `TrEncrypt`$(\cdot)$, and `Decrypt`$(\cdot)$ as

- `Setup`(N, λ): a probabilistic algorithm that takes as input N, and a security parameter λ. The algorithm runs in polynomial time in λ and outputs a public key

PK, a secret key TK and private keys $K_1, \ldots, K_N$, where K_i is given to user U_i.

• $\texttt{Encrypt}(PK, M)$: takes as input PK, and a message M. It outputs a ciphertext C which will be broadcast to all the users.

• $\texttt{TrEncrypt}(TK, i, M)$: takes as input TK, an integer i satisfying $1 \leqslant i \leqslant N+1$, and a message M. It outputs a ciphertext C. $\texttt{TrEncrypt}(\cdot)$ is primarily used for traitor tracing. (Note: $\texttt{TrEncryptL}(TK, 1, M)$ outputs a ciphertext distribution that is indistinguishable from the distribution generated by $\texttt{Encrypt}(PK, M)$.)

• $\texttt{Decrypt}(j, K_j, C, PK)$: takes as input a subset the private key K_j for user j, a ciphertext C, and the public key PK. The algorithm outputs a message M or $\perp$.

PLBE property: $\forall i \in [1, N+1]$, $\forall j \in [1, N]$, and all messages M:
Let $(PK, TK, (K_1, \ldots, K_N)) \leftarrow \texttt{Setup}(N, \lambda)$,
and $C \leftarrow \texttt{TrEncrypt}(TK, i, M)$.
If $j \geqslant i$ **then** $\texttt{Decrypt}(j, K_j, C, PK) = M$.

Loosely speaking, PLBE property is similar to encryption with key-chain. But PLBE can result in the following tracing algorithm.

5.1.2 Tracing Algorithm

For a suspected decoder $\mathcal{D}$, and a given $\epsilon > 0$, the tracer will perform the tracing algorithm $\texttt{Trace}(TK, \epsilon)$ as Fig.5.

Let the traitor set $\mathbb{T} = \phi$,

(a) **For** $i = 1$ to $N+1$, **do** the following:
Let $\eta = 0$. Repeat the following steps $\Omega_3 = 8\lambda(N \log N/\epsilon)$ times:
- Sample message M from the finite message space at random.
- Let $C \leftarrow \texttt{TrEncrypt}(TK, i, M)$.
- Call $\mathcal{D}$ on input C to get the output $\tilde{M}$. If $\tilde{M} = M$, $\eta \leftarrow \eta + 1$.

Let $\hat{p}_i$ be the fraction of times that $\mathcal{D}$ decrypted the ciphertexts correctly, thus, $\hat{p}_i = \eta/\Omega_3$.
If $\hat{p}_i - \hat{p}_{i+1} \geqslant \epsilon/(4N)$, add i into $\mathbb{T}$.

(b) Output $\mathbb{T}$ as the set of guilty colluders.

Fig.5. Traitor tracing algorithm in FTT1.

5.2 Attack to FTT1 Scheme

As constructed in Fig.1, a pirate decoder $\mathcal{D}$ can detect the tracing activity easily, then it may perform escaping from identifying, denying tracing, or framing innocent users.

5.2.1 Detecting Trace

Assume that the traitors divide the users into $n+1$ groups as $G_j = \{U_u \mid \pi_j < u \leqslant \pi_{j+1}\}$, $j = 1, 2, \ldots, n-1$. Specially, $G_0 = \{U_u \mid u \leqslant \pi_1\}$, and $G_n = \{U_u \mid \pi_n < u \leqslant N\}$.

• In the tracing process $\texttt{Trace}(TK, \epsilon)$, the tracer will generate a ciphertext $C = \texttt{TrEncrypt}(TK, i, M)$, and send C to the suspected decoder $\mathcal{D}$.

• The WATCH unit of $\mathcal{D}$ gets n plaintext M_{π_j} generated by traitor's decoder $\mathcal{D}_{\pi_j}$. With regard to Fig.6, the index i of examined user must be

$$i \in \begin{cases} G_0 = [1, \pi_1], & M_{\pi_1} = M_{\pi_n}, \\ G_j = (\pi_j, \pi_{j+1}], & \exists j,\ M_{\pi_j} \neq M_{\pi_{j+1}} = M_{\pi_{j+2}}, \\ G_n = (\pi_{n-1}, N], & \forall j, k,\ j \neq k,\ M_{\pi_j} \neq M_{\pi_k}. \end{cases} \tag{4}$$

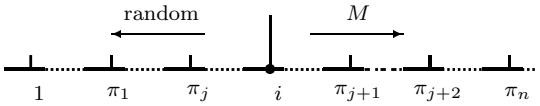

Fig.6. Determinate the range of the examined index i.

Hence, according to (4), the statement in [23] "*a broadcast to users* $\{i, \ldots, N\}$ *should reveal no non-trivial information about* i" is not true since the pirate decoder can know the interval of i.

• The ACT unit of $\mathcal{D}$ will handle all the results M_{π_j} as

$$O_p = \begin{cases} 1, & i \in G_0, \\ -1, & i \in G_j,\ j \in [1, n-1], \\ 0, & i \in G_n. \end{cases} \tag{5}$$

If $O_p = 1$, the pirate decoder cannot tell tracing from broadcasting, and regards that it is normal broadcast. In this case, it outputs the correct message M. Thus, (1) is met easily, i.e., the pirate decoder is useful. But if $O_p \neq 1$, the decoder regards it is traced, and may take protective actions against the tracer as the following Subsections.

5.2.2 Minimizing the Number of Disclosed Traitors

With the tracing algorithm in FTT1, Boneh *et al.* proved that at least one traitor will be identified theorematically. But the tracing algorithm may arrest all the traitors if they do not act properly. Therefore, from the viewpoint of traitors, it is the best strategy if the number of disclosed traitors is only one.

To this end, the traitors may select a fix victim U_j, e.g., $U_j = U_{\pi_n}$. For any input ciphertext, the pirate decoder always outputs M_j. In this case, one traitor is identified in theory. But if the tracer has no enough patience/resource due to the following "denial-of-trace" attack, no one is identified in practice.

5.2.3 DOT Attack

FTT1 is designed to defeat against stateless decoder. Hence, the RST unit should be activated before each

trial. There are two activation events. One is that the tracer activates the resetting function (e.g., by pushing a "RESET" button); the other is the decoder initiates the resetting process if $O_p \neq 1$. Any of the resetting event will start the reset process which will take initialization time τ. Recall that the initialization time is decided by the pirate decoder designer, rather than the tracer. Therefore, if the decoder is reset after each tracing trial, the total tracing time in FTT1 is at least

$$T_{\text{trace}} = \tau\Omega_3 = 8\tau\lambda N^2 \log N/\epsilon.$$

Example 1. Let us select some conservative parameters, $N = 100$, $\lambda = 500$, $\epsilon = 0.1$ (Note: Dodis *et al.* select $\epsilon = 0.01$ in [14]), $\tau = 1$ minute, a full trace process takes

$$\begin{aligned} T_{\text{trace}} &\approx 8 \times 1 \times 100 \times 100^2 \times \log 100/0.1 \\ &\approx 3.68 \times 10^8 \text{ minutes } \approx 700 \text{ years}! \end{aligned}$$

Even the binary search is used, the tracing process takes at least 32 years ! Thus, the tracing algorithm is infeasible if RST unit is performed in each tracing trial to make sure stateless. But if the decoder is stateful, it will start a framing attack which is more serious attack (see Section 6).

6 DOT Attack to FTT2

The public fully collusion traitor tracing scheme (or FTT2)[24] enables any one to trace a suspected decoder. In addition, a broadcaster can revoke any set of users. However, since FTT2 is similar to FTT1, the traitors can build a pirate decoder to exploit FTT2 with the similar methods in Section 5.

6.1 Overview of FTT2 Scheme

The cryptographic primitive in FFT2[24] is ABE (Augmented Broadcast Encryption). As PLBE, ABE can also result in a traitor tracing algorithm $\texttt{Trace}(\cdot)$.

6.1.1 ABE Algorithm

ABE consists of a set of cryptographic functions: $\texttt{Setup}(\cdot)$, $\texttt{Encrypt}(\cdot)$, and $\texttt{Decrypt}(\cdot)$. Their definitions are

• $\texttt{Setup}(N, \lambda)$: a probabilistic algorithm that takes as input N, and a security parameter λ. The algorithm runs in polynomial time in λ and outputs a public key PK and private keys $K_1, \ldots, K_N$, where K_j is given to user U_j.

• $\texttt{Encrypt}(S_{\mathcal{D}}, PK, i, M)$: takes as input a subset of users $S_{\mathcal{D}} \subseteq \{1, \ldots, N\}$, a public key PK, an integer i satisfying $1 \leqslant i \leqslant N+1$, and a message M. It outputs a ciphertext C. This algorithm encrypts a message to a set $S_{\mathcal{D}} \cap \{i, \ldots, N\}$.

• $\texttt{Decrypt}(S_{\mathcal{D}}, j, K_j, C, PK)$: takes as input a subset $S_{\mathcal{D}} \subseteq \{1, \ldots, N\}$, the private key K_j for user j, a ciphertext C, and the public key PK. The algorithm outputs a message M or $\perp$.

ABE property: for all subsets $S_{\mathcal{D}} \subseteq \{1, \ldots, N\}$, $\forall i$, $j \in \{1, \ldots, N+1\}$ (where $j \leqslant N$), and all messages M:
Let $(PK, (K_1, \ldots, K_N)) \leftarrow \texttt{Setup}(N, \lambda)$,
and $C \leftarrow \texttt{Encrypt}(S_{\mathcal{D}}, PK, i, M)$.
If $j \in S_{\mathcal{D}}$ and $j \geqslant i$, $\texttt{Decrypt}(S_{\mathcal{D}}, j, K_j, C, PK)$ is M, **otherwise**, $\perp$.

6.1.2 Tracing Algorithm

In the tracing algorithm $\texttt{Trace}(S_{\mathcal{D}}, PK, \epsilon)$, the tracer confiscates a suspected decoder $\mathcal{D}$, for a given $\epsilon > 0$ and a set $S_{\mathcal{D}}$, the tracing processing is as Fig.7.

Let suspect set $\mathbb{T} = \phi$.

(a) **For** $i = 1$ to $N+1$ **do**
Let $\eta = 0$. Repeat the following steps $\Omega_4 = 8\lambda(N/\epsilon)^2$ times:

- Sample message M from the finite message space at random.
- Let $C = \texttt{Encrypt}(S_{\mathcal{D}}, PK, i, M)$.
- Call $\mathcal{D}$ on input C to get decoder output $\tilde{M}$. If $\tilde{M} = M$, $\eta \leftarrow \eta + 1$.

Let $\hat{p}_i$ be the fraction of times that $\mathcal{D}$ decrypted the ciphertexts correctly, i.e., $\hat{p}_i = \eta/\Omega_4$.

(b) **If** $\hat{p}_i - \hat{p}_{i+1} \geqslant \epsilon/(4N)$, **then** add i to set $\mathbb{T}$.

(c) **Output** $\mathbb{T}$ as the set of guilty colluders.

Fig.7. Traitor tracing algorithm in FTT2.

6.1.3 Tracing All Traitors

As the tracing algorithm in FTT1, the above $\texttt{Trace}(S_{\mathcal{D}}, PK, \epsilon)$ may identify only one traitor. To identify all the traitors, Boneh *et al.*[24] propose to iteratively update the set $S_{\mathcal{D}}$ so as to revoke the identified traitors. Specifically, the $\texttt{TraceAll}(S_{\mathcal{D}}, PK, \epsilon)$ algorithm is shown in Fig.8.

(a) set $S_{\mathcal{D}}$ as the set of all of users;

(b) find one traitor u with $\texttt{Trace}(S_{\mathcal{D}}, PK, \epsilon)$;

(c) $\mathbb{T} \leftarrow \mathbb{T} \cup \{u\}$, and calculate $S_{\mathcal{D}} \leftarrow S_{\mathcal{D}} \backslash \{u\}$;

(d) if $\texttt{Trace}(S_{\mathcal{D}}, PK, \epsilon)$ outputs a traitor, go to step (b). Otherwise, quit.

Fig.8. Tracing all of the traitors in FTT2.

6.2 Attack to FTT2 Scheme

Since the tracing algorithm of FTT2 is similar to that of FTT1, FTT2 has all the vulnerabilities as FTT1. Especially, albeit FTT2 has `TraceAll`$(\cdot)$, it can arrest at most one traitor. For brevity, we do not repeat the attacks in Subsection 5.2 on FFT2. But we should emphasize that FTT2 is very vulnerable to DOT attack since `Trace`$(S_{\mathcal{D}}, PK, \epsilon)$ needs $8\lambda N^3/\epsilon^2$ tracing trials for linear scan, and almost $8\lambda N^2 \log N/\epsilon^2$ trials for binary scan.

Example 2. Let us estimate the time for the linear tracing with the same parameters as Example 1. $\tau = 1$ minute, $N = 100$, $\epsilon = 0.1$, $\lambda = 100$, the total tracing time is

$$\begin{aligned} T_{\text{trace}} &= N \times \Omega_4 \times \tau = N \times 8\lambda(N^2/\epsilon^2) \\ &= 8\lambda\tau N^3/\epsilon^2 = 800 \times 10^6/10^{-2} \\ &= 8 \times 10^{10} \text{ minutes} = 1.5 \times 10^5 \text{ years}! \end{aligned}$$

Thus, on average, it requires about 7.7×10^4 years to trace a traitor of a small group of users. If the trace-revoke process `TraceAll`$(S_{\mathcal{D}}, PK, \epsilon)$ is used to trace all the traitors, it costs about 7.7×10^6 years.

7 Framing Attack

In this section, we discuss the soundness of stateless decoder, and then create a stateful decoder such that the tracer is fooled to frame innocent users. Finally, we propose a countermeasure to defeat the framing attack. For simplicity, we address FTT1 only as it is straightforward to extend the attack and countermeasure to FTT2.

7.1 Stateless vs. Stateful Decoder

FTT[23] "*solves the tracing traitors problem in the stateless model, where the tracer is allowed to reset the pirate algorithm after each tracing query*". Although Boneh *et al.* explicitly stated that the pirate decoder is stateless, we think the stateless model is too strict to be practical because:

(a) *Cheap in Manufacturing.* Nowadays, it is easy to manufacture a stateful decoder. For example, a decoder with NVM③ which cost below $0.01 is stateful. In reality, many devices (e.g., RFID, smartcard, DVD player etc.) themselves have NVM to store the private data. In other words, it almost costs nothing to enable a stateless decoder into a stateful decoder.

(b) Volatile memory retains the data for some time after power-off. As far back as in 1978, researchers showed that if DRAM is cooled with liquid nitrogen, its data retention period could last up to a week. Practically, Halderman *et al.*[28] showed that encryption keys can be recovered from memory chips several minutes after power is disconnected. Hence, the decoder can read the previous result after a resetting is performed.

(c) *Necessary for Decoding.* History information is necessary in most of multimedia decoders which are compliant with International Standards (e.g., MPEG). For example, in order to decode MPEG video and MP3 audio, the decoder has to store the recently decoded pictures/chips.

(d) *Hard in Reducing.* To explain the soundness of stateless model, Boneh and Water argued that any tracing system for stateless decoders can be converted into a tracing system for stateful decoders by embedding watermarks[1]. However, the conversion algorithm[1] has three pre-requisitions:

• Partial collusion: as the scheme in [17], Kiayias and Yung's scheme guarantees to arrest a traitor when the number of traitors is below a predefined value. Since FTT1 and FTT2 assume that all the users are involved in collusion, the transform method[1] is invalid for either FTT1 or FTT2.

• Best-effort decryption: the conversion algorithm[1] requires the decoder to try its best to decrypt all the input ciphertexts. Otherwise, their method does not work. However, Boneh and Waters[24] merely required `useful decoder` which can partially decrypt ciphertexts. Thus, if a pirate decoder discards some decoded messages intentionally, (1) is still satisfied, but the tracing algorithm[1] fails. That is to say, the tracing algorithms[1] have more strict condition than FTT[23-24], thus the conversion method[1] is not suitable to either FTT1 or FTT2.

• Solid watermarking assumption: Kiayias and Yung assumed that it was possible to identify at least one traitor from a copy generated from several watermarked copies. This assumption is still questionable:

  ∗ Ergun *et al.*[29] proved that no traitor will be identified with a pirated copy if the number of the traitors is below a threshold given that the probability of implicating innocent is low;

  ∗ The international initiative SDMI (Secure Digital Music Initiative) (http://www.sdmi.org/) aimed to develop and standardize technologies that "protect the playing, storing, and distributing of digital music". Although no documentation explained the implementations of the SDMI technologies, and neither watermark embedding nor

③NVM: non-volatile memory includes read-only memory, flash memory, most types of magnetic computer storage devices (e.g., hard disks, floppy disk drives, and magnetic tape), optical disc drives, and early computer storage methods such as paper tape and punch cards. See http://en.wikipedia.org/wiki/Non-volatile_storage.

detecting software was directly accessible to challenge participants, Craver *et al.*[30] have defeated all four of the audio watermarking technologies;

∗ A bulk of papers related to watermark analysis are published annually. In fact, a lot of watermarking methods are broken, even the well-cited spread spectrum method[31] is also challenged[32].

Thus, it is still challenging to convert a tracing system for stateless decoders into a tracing system for stateful decoders, especially in case of FTT.

In all, FTT[23-24] model supposes: the colluders are willing to recruit cryptanalysts to break semantically secure broadcast encryption scheme, but they will not spend \$0.01 to change a stateless decoder into stateful decoder. *Do the traitors regard that cryptanalysts are valueless?* Clearly, this model is not sound, especially for hardware decoders.

7.2 Framing Innocent User with Stateful Decoder

According to Subsection 7.1, it is easy to design a stateful decoder. With a stateful decoder, the traitors can cheat FFT1 tracer to frame innocent users. Specifically, if only one NVM counter l_C is used to recode the number of ciphertexts received by the decoder, the pirate decoder $\mathcal{D}$ may be used to frame innocent users as Fig.9.

(a) Let $l_C = 0$.
(b) For each ciphertext received, $\mathcal{D}$ will perform steps (c)∼(f).
(c) The pirate decoder runs the detection algorithm to obtain O_p. If $O_p \neq -1$, the pirate decoder can tell tracing algorithm from broadcasting one, hence it output $\Sigma_{i=1}^{n} M_{\pi_i}/n$, and goes to step (b).
(d) But if $O_p = -1$, the pirate decoder knows that the tracing algorithm is being performed, and continues the following framing steps.
(e) Let $l_C = l_C + 1$, and $\gamma = \lfloor l_C/\Omega_3 \rfloor + 1$. Since each user is examined for Ω_3 times continuously, the decoder knows that the user $U_{\pi_1+\gamma}$ is examined.
(f) If $(\gamma \bmod 2) = 1$, output M_{π_n} which is identical to M, otherwise, $\mathcal{D}$ outputs a noise.

Fig.9. Framing attack to FTT1.

After finishing all the tracing trials, the tracer concludes: $\forall \alpha = \pi_1 + 2d - 1 \leqslant \pi_n$ for some integer d, the pirate decoder $\mathcal{D}$ can decode all the ciphertexts $\texttt{TrEncrypt}(TK, \alpha, M)$, but cannot decode any ciphertext $C = \texttt{TrEncrypt}(TK, \alpha + 1, M)$, i.e.,

$$\hat{p}_\alpha = 1 \text{ but } \hat{p}_{\alpha+1} = 0.$$

Thus,

$$\hat{p}_\alpha - \hat{p}_{\alpha+1} = 1 \geqslant \epsilon/(4N).$$

According to the tracing rule, the tracer identifies U_α as a traitor. Clearly, many users are framed. Thus, if the traitors spend only \$0.01 to install a counter, the tracing algorithm will frame innocent users at probability 1. Thus, FTT1 tracing algorithm is too fragile to be viable.

Example 3. Given $N = 10$ users, U_3 and U_8 are traitors who created a pirate decoder. In the tracing method in Subsection 5.1.2, if $3 < i \leqslant 8$, decoder $\mathcal{D}$ will find the tracing activity since the output of $\mathcal{D}_8$ is structural but $\mathcal{D}_3$ outputs random. Hence, when the tracer performs tracing algorithm, the output for $i = 4, 6$ is from D_8, while the output for $i = 5, 7$ is from D_3, hence,

$$\hat{p}_4 = \hat{p}_6 = 1 \text{ but } \hat{p}_5 = \hat{p}_7 = 0.$$

That is to say,

$$\hat{p}_4 - \hat{p}_5 = 1 > \epsilon/(4N).$$

and

$$\hat{p}_6 - \hat{p}_7 = 1 > \epsilon/(4N).$$

Thus, users U_4 and U_6 are identified as traitors wrongly.

7.3 Countermeasure on Framing Attack

In the FTT scheme, the tracer scans the users in a known pattern (linear or binary), if the tracer randomly scans the users, the pirate decoder maybe fail to framing the innocent users. That is to say, the improved tracing algorithm is as Fig.10.

(a) Let traitor set as empty $\mathbb{T} = \phi$.
(b) The algorithm repeats the following steps L times:
  (i) Sample a *structural* message M, and *select* $i \in_R [1, N]$ *randomly*.
  (ii) Let $C = \texttt{Encrypt}(S_{\mathcal{D}}, PK, i, M)$.
  (iii) Call $\mathcal{D}$ on input C, and compare the output of $\mathcal{D}$ to M.
(c) Let $\hat{p}_i$ be the fraction of times that $\mathcal{D}$ decrypted the ciphertexts correctly.
(d) If $\hat{p}_i - \hat{p}_{i+1} \geqslant \epsilon/(4N)$, then add i to set $\mathbb{T}$.

Fig.10. Countermeasure on framing attack. We think the tracing message should be structural. Otherwise, the pirate decoder will always output noise so as to defeat the tracing algorithm if the message is not structural.

With the random scan pattern, albeit the craft decoder knows the range of i, the probability of $\hat{p}_i - \hat{p}_{i+1}$ is small, and hence no user in interval $(\pi_j, \pi_{j+k}]$ is accused.

8 Discussion

8.1 Traitor Tracing under One Roof

In most, if not all, of traitor tracing schemes, the tracer will perform a trial:

• select a target event E (a suspect group of users or keys),

• create a craft ciphertext for the target,

• observe the output or action so as to obtain the occurrence rate of E.

Let us consider the tracing algorithm as throwing a biased coin. One side is more likely to come up than the other, but you do not know which and would like to find out. The obvious solution is to flip it many times and then choose the side that comes up the most. But how many times do you have to flip it to be confident that you have chosen correctly? One lower bound of trial times may be the following version of Chernoff bound[19]

$$P\Big(\frac{1}{\Omega}\sum_{i=1}^{\Omega} E_i \geqslant \beta p\Big) < \Big(\frac{e^{\beta-1}}{\beta^{\beta}}\Big)^{p^{\Omega}} \qquad (6)$$

where $\beta > 1$ is an error-tolerance parameter, and observation value $E_i = 1$ means that event E occurs, and $p = P(E)$ is the probability of event E which is related to parameters N, t and ϵ in tracing algorithm. In order to make the error of implicating the innocent user is negligible, the right side of (6) should be very small, such that Ω is selected. Therefore, Ω is very large such that the traitor tracing scheme suffers from DOT attack.

8.2 Countermeasure by Forging Reset Time

Subsection 8.1 indicates it is not easy to reduce the resetting times unless other bound algorithm is available. Is it possible to reduce the actual resetting TIME? For hardware decoder, the hope is slim if we do not reverse-engineer the decoder. But for software decoder, it is possible since the execution time is obtained from the OS (Operating System). If the pirate decoder is cheated by a faked time such that the actual resetting time is much smaller than τ, DOT attack fails.

8.3 Counter-Countermeasure

The above countermeasure is viable since the pirate decoder cannot calibrate the system clock and hence is cheated. Hence, the pirate decoder should testify OS (or other environment). Here, assume that the pirate software has no random resource since the resource is under control of the tracer, but it has a secret data R, e.g., decoding key or hard-code secret. The testation process is as Fig.11.

(a) In the resetting, the pirate decode executes a module F, and query OS the execute time, the OS may cheat it with a time t_1. Afterwards, the pirate decoder continues the resetting process. Please note the really resetting time may be smaller than τ.

(b) When the pirate decoder receives the first ciphertext C, it will generate

$$n_0 = \mathcal{H}(C \parallel R)$$

where $\mathcal{H}(\cdot)$ is a one-way function.

(c) The pirate decoder repeatedly executes the module F for i times, and queries for the execution time t_2.

(d) If $\mid t_2/t_1 - n_0 \mid > \epsilon''$ for some threshold ϵ'', the pirate decoder confirms it is cheated and delay for some time, otherwise, it works as usual.

Fig.11. Testify the environment by pirate decoder software.

9 Conclusion

This paper presents a crafty pirate decoder which can exploit several traitor tracing schemes. Concretely, it exploits the vulnerabilities of the fully collusion traitor tracing schemes. To this end, the pirate decoder detects the tracing activity, and then takes action such as starting denial-of-trace, and framing innocent users. To be frameproof tracing, we replace the deterministic trace scan into random scan. However, this improvement cannot defeat denial-of-trace attack since the property of the underlying cryptographic primitive exposes the tracing activity to the craft decoders.

References

[1] Kiayias A, Yung M. On crafty pirates and foxy tracers. In *ACM Workshop in Digital Rights Management* (*DRM 2001*), Philadelphia, USA, Nov. 5, 2002, Revised Papers, *LNCS* 2320, pp.22-39.

[2] Kiayias A, Yung M. Self protecting pirates and black-box traitor tracing. In *Proc. CRYPTO 2001*, Santa Barbara, USA, Aug. 19-23, 2001, pp.63-79.

[3] Pfitzmann B. Trails of traced traitors. In *Information Hiding Workshop*, Cambridge, UK, May 30-June 1, 1996, *LNCS* 1174, pp.49-64.

[4] Yan J J, Wu Y. An attack on a traitor tracing scheme. Technical Report No.518, Computer Lab., Univ. Cambridge, 2001.

[5] Boneh D, Franklin M. An efficient public key traitor tracing scheme. In *Proc. Crypto 1999*, Santa Barbara, USA, Aug. 15-19, 1999, *LNCS* 1666, pp.338-353.

[6] Kiayias A, Yung M. Traitor tracing with constant transmission rate. In *Proc. Eurocrypt 2002*, Amsterdam, The Netherlands, April 28-May2, 2002, *LNCS* 2332, pp.450-465.

[7] Chabanne H, Phan D H, Pointcheval D. Public traceability in traitor tracing schemes. In *Proc. EUROCRYPT 2005*, Aarhus, Denmark, May 22-26, 2005, *LNCS* 3494, pp.542-558.

[8] Dwork C, Lotspiech J, Naor M. Digital signets: Self-enforcing protection of digital content. In *Proc. STOC 1996*, Philadelphia, USA, May 22-24, 1996, pp.489-498.

[9] Naor M, Pinkas B. Efficient trace and revoke schemes. In *Proc. Financial Crypto 2000*, Anguilla, British West Indies, Feb. 20-24, 2000, *LNCS* 1962, pp.1-20.

[10] Fiat A, Tassa T. Dynamic traitor tracing. In *Proc. CRYPTO 1999*, Santa, Barbara, USA, Aug. 15-19, 1999, *LNCS* 1666, pp.354-371.
[11] Berkman O, Parnas M, Sgall J. Efficient dynamic traitor tracing. *SIAM Journal of Computing*, 2001, 30(6): 1802-1828.
[12] Safavi-Naini R, Wang Y. Sequential traitor tracing. In *Proc. Crypto 2003*, Santa Barbara, Aug. 17-21, 2000, *LNCS* 1880, pp.316-332, see also *IEEE Transactions on Information Theory*, May 2003, 49(5): 1319-1326.
[13] Zhao H V, Liu K J R. Fingerprint multicast in secure video streaming? *IEEE Trans. Image Processing*, Jan. 2006, 15(1): 12-29.
[14] Dodis Y, Fazio N, Kiayias A, Yung M. Scalable public-key tracing and revoking. In *Proc. PODC 2003*, Boston, USA, July 13-16, 2003, pp.190-199.
[15] Kiayias A, Yung M. Breaking and repairing asymmetric public-key traitor tracing. In *Proc. ACM DRM 2002*, Washington DC, USA, Nov. 18, 2002, *LNCS* 2696, pp.32-50.
[16] Tzeng W G, Tzeng Z J. A public-key traitor tracing scheme with revocation using dynamic shares. *Designs, Codes and Cryptography*, 2005, 35(1): 47-61.
[17] Chor B, Fiat A, Naor M. Tracing traitors. In *Proc. Crypto 1994*, Santa Barbara, USA, Aug. 27-31, 1994, *LNCS* 839, pp.257-270.
[18] Naor M, Pinkas B. Threshold traitor tracing. In *Proc. Crypto 1998*, Santa Barbara, USA, Aug. 23-27, 1998, *LNCS* 1462, pp.502-517.
[19] Chor B, Fiat A, Naor M, Pinkas B. Tracing traitors Chor. *IEEE Transactions on Information Theory*, May 2000, 46(3): 893-910.
[20] Kurosawa K, Desmedt Y. Optimum traitor tracing and asymmetric schemes. In *Proc. Eurocrypt 1998*, Espoo, Finland, May 31-June 4, 1998, *LNCS* 1403, pp.145-157.
[21] Dodis Y, Fazio N. Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In *Proc. the 6th Int. Workshop on Theory and Practice in Public Key Cryptology*, Miami, USA, Jan. 6-8, 2003, pp.100-115.
[22] McGregor J P, Yin Y L, Lee R B. A traitor tracing scheme based on RSA for fast decryption. In *Proc. ACNS 2005*, New York, USA, June 7-10, 2005, *LNCS* 3531, pp.56-74.
[23] Boneh D, Sahai A, Waters B. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Proc. Eurocrypt 2006*, St. Petersburg, Russia, May 28-June 1, 2006, *LNCS* 4004, pp.573-592.
[24] Boneh D, Waters B. A collusion resistant broadcast, trace and revoke system. In *Proc. ACM CCS 2006*, Alexandria, USA, Oct. 30-Nov. 3, 2006, http://eprint.iacr.org/2006/298.
[25] Qiao L, Nahrstedt K. Watermarking schemes and protocols for protecting rightful ownership and customer's rights. *J. Vis. Commun. Image Representation*, 1998, 9(3): 194-210.
[26] Tonien D, Safavi-Naini R. An efficient single-key pirates tracing scheme using cover-free families. In *Proc. ACNS 2006*, Singapore, June 6-9, 2006, *LNCS* 3989, pp.82-97.
[27] Alon N, Bruck J, Noar J, Noar M, Roth R. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Trans. Information Theory*, 1992, 38(3): 509-516.
[28] Halderman J A, Schoen S D, Heninger N, Clarkson W, Paul W, Calandrino J A. Lest we remember: Cold boot attacks on encryption keys. In *Proc. the 17th USENIX Security Symposium*, San Jose, USA, July 28-Aug. 1, 2008, pp.91-98.
[29] Ergun F, Kilian J, Kumar R. A note on the limits of collusion-resistant watermarks. In *Proc. EUROCRYPT 1999*, Prague, Czech, May 2-6, 1999, *LNCS* 1592, pp.140-149.
[30] Craver S A, Wu M, Liu B, Stubblefield A, Swartzlander B, Wallach D S, Dean D, Felten E W. Reading between the lines: Lessons from the SDMI challenge. In *Proc. the 10th USENIX Security Symposium*, Boston, USA, June 25-30, 2001, pp.353-363.
[31] Cox I J, Kilian J, Leighton T, Shamoon T. Secure spread spectrum watermarking for multimedia. *IEEE Trans. Image Processing*, 1997, 6(12): 1673-1687.
[32] Das T K, Maitra S. Cryptanalysis of correlation-based watermarking schemes using single watermarked copy. *IEEE Signal Processing Letters*, 2004, 11(4): 446-449.

Yong-Dong Wu received the B.A. and M.S. degrees from Beihang University in 1991 and 1994 respectively, and the Ph.D. degree from Institute of Automation, Chinese Academy of Sciences in 1997. He is currently a senior scientist with Cryptography and Security Department, Institute of Infocomm Research (I^2R), A*STAR, Singapore. His interests includes multimedia security, e-Business, digital right management and network security. He has published more than 10 journal papers, more than 60 conference papers, and 7 patents. His research results and proposals was incorporated in the ISO/IEC JPEG 2000 security standard 15444-8 in 2007. Dr. Wu won the Tan Kah Kee Young Inventor award in 2004 and 2005.

Robert H. Deng received his B.Eng. degree from National University of Defense Technology, China, in 1981, his M.Sc. and Ph.D. degree from Illinois Institute of Technology, Chicago, in 1983 and 1985, respectively. He is currently professor and director of SIS Research Center, School of Information Systems, Singapore Management University. Prior to this, he was principal scientist and manager of Infocomm Security Department, Institute for Infocomm Research. He has 26 patents and more than 200 technical publications in international conferences and journals in the areas of digital communications, network and distributed system security and information security. He has served as general chair, program chair, and program committee member of numerous international conferences. He received the University Outstanding Researcher Award from the National University of Singapore in 1999 and the Lee Kuan Yew Fellow for Research Excellence from the Singapore Management University in 2006.