Breaking and Repairing Asymmetric Public-Key Traitor Tracing

Aggelos Kiayias* Moti Yung†

Abstract

Traitor tracing schemes are a very useful tool for preventing piracy in digital content distribution systems. A traitor tracing procedure allows the system-manager to reveal the identities of the subscribers that were implicated in the construction of a pirate-device that illegally receives the digital content (called traitors). In an important variant called "asymmetric" traitor tracing, the system-manager is not necessarily trusted, thus the tracing procedure must produce *undeniable* proof of the implication of the traitor subscribers. This *non-repudiation* property of asymmetric schemes has the potential to significantly increase the effectiveness of the tracing procedure against piracy.

In this work, we break the two previous proposals for efficient asymmetric public-key traitor tracing, by showing how traitors can evade the proposed traitor tracing procedures. Then, we present a new efficient Asymmetric Public-Key Traitor Tracing scheme for which we prove its traceability in detail (in the non-black-box model); to the best of our knowledge this is the first such scheme. Our system is capable of proving the implication of *all* traitors that participate in the construction of a pirate-key. We note that even though we break the earlier schemes we employ some of their fundamental techniques and thus consider them important developments towards the solution.

1 Introduction.

The secure distribution of a digital content stream to an *exclusive* set of subscribers is an important problem that has many applications in the entertainment industry. The typical setting is that of Pay-TV: the content distributor transmits scrambled streams of video of the channel line-up that are received by the subscribers using a decoder device (e.g. a cable-box). The digital content should be encrypted in such a way so that eavesdroppers are incapable of intercepting the stream. On the other hand, each legitimate subscriber possesses a decryption mechanism (essentially: a cryptographic decryption key) that enables him/her to receive the content.

One major problem faced by administrators of such systems is "piracy": the illegal reception of the scrambled content that is made possible by taking advantage of "insider information." Current encryption mechanisms are strong enough to ensure that an eavesdropper is incapable of inverting the scrambling method used in the broadcast to the legitimate subscribers. However, illegal reception of the digital content can still occur if some of the legitimate users of the system leak (some of) their key information to a third party. Such users are called *traitors* and a third party that uses subscriber-key information for illegal data reception is called a *pirate*.

The traditional approach to tackle the problem of piracy is to hide the decryption-key information from all legitimate subscribers of the system. This can be a quite tricky problem,

*CS&E Department, University of Connecticut, Storrs, CT, USA aggelos@cse.uconn.edu

†Columbia University, NY, USA moti@cs.columbia.edu

since every legitimate user should possess enough key-information to enable the reception of the scrambled digital content. Hiding the key from the user can be achieved by employing tamper resistant hardware (so that the key-information is concealed, but still can be used in a black-box fashion), or, in the software-based setting, program obfuscation can be employed instead. Hardware tamper resistance can provide satisfactory solutions in terms of security. However the cost of constructing and distributing tamper resistant decoders to each subscriber of the system is prohibitive for many digital content providers. Especially in an Internet-based setting it would be extremely desirable that no system-specific physical device should be required and that the decoder software should be downloadable from the distributor's web-site. This suggests that a software obfuscation technique would be more appropriate, however there are no cryptographically strong methods to achieve general software obfuscation and, further, there are indications that it is actually impossible (see [28, 2]), so one has to rely in ad-hoc methods. As a result, no key-concealment method is a complete panacea: there will always be at least one savvy malicious subscriber that will compromise the resistance (hardware tamper-resistance or software obfuscation) of the decoder. Even worse, once the the decoder is broken once, the information on how to repeat this can be distributed to thousands of users using the Internet, thus dramatically scaling up the damages to the content-distributors [1].

Traitor Tracing schemes: An alternative approach to piracy prevention in digital content distribution systems was proposed by Chor, Fiat and Naor under the framework of Traitor Tracing Schemes (TTS) [8] (see also [9]). In a TTS, each subscriber has a decryption key that is associated with his identity (it can be thought of as a fingerprinted decryption key). Malicious subscribers (traitors) might again try to leak their personal key information to a pirate. However, in a traitor-tracing scheme the distributor (or the authorities) possess a "traitor tracing" procedure that, given the pirate decoder, is capable of recovering the identities of at least one of the traitors. Even though the existence of such a mechanism cannot eliminate piracy, it can effectively deter users from leaking their personalized keys to a pirate. The probabilistic constructions of [8] were followed by more explicit combinatorial designs [30], and later by [13, 22], who also considered the combination of traitor tracing schemes with efficient revocation methods (cf. broadcast encryption, [12]).

Public Key Traitor Tracing: Traitor Tracing Schemes have been introduced in the symmetric encryption setting, under the basic assumption that the content distributor coincides with the administrator of the secure broadcasting infrastructure. However it is highly desirable to divide these roles. In public-key traitor tracing, there is one authority that is responsible for the broadcasting infrastructure (which we call the system-manager) and several, non-trusted, content-distributors that may take advantage of the public-key encryption procedure (published by the system-manager) to distribute content to the subscribers of the system. This setting is illustrated in figure 1. Public-key TTSs were presented in [20, 4, 16].

Asymmetric Traitor Tracing: A shortcoming of the traitor tracing procedure, as it is achieved in all the schemes mentioned above, is the fact that the system-manager does not obtain undeniable *proof* for the implication of a certain set of subscribers in the construction of a pirate device. In these schemes, the system-manager knows all the key information distributed to the users and as a result, if it is malicious, it can implicate an innocent user in the

[1] Industry assessment of the risk involved in such exposures was demonstrated recently by the $ 1 billion lawsuit filed (and currently kept on hold) by Canal Plus (owned by Media giant Vivendi Universal) against the NDS Group (controlled by News Corp. of Rupert Murdoch), alleging that top engineers hired by NDS broke into the security system of Canal Plus Pay-TV service and exposed sensitive user-key information to thousands of potential malicious users by publishing this information on the web [1].

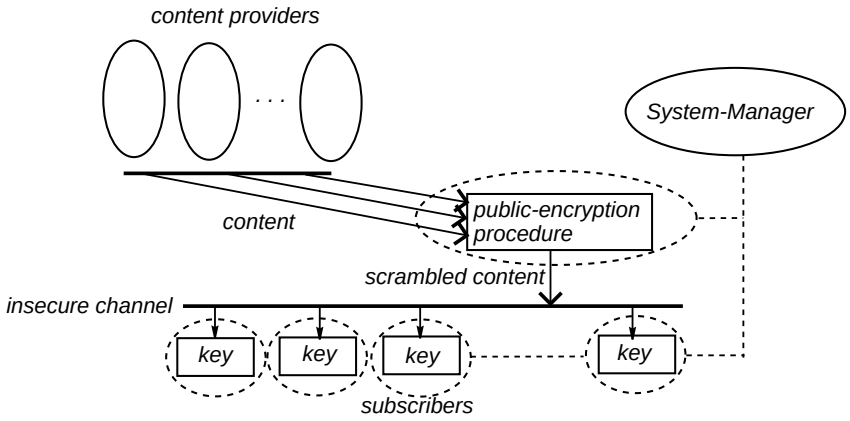

Figure 1: The Setup of a Public-key Traitor Tracing Scheme

construction of a pirate device. As a result, these schemes do not support *non-repudiation*: subscribers can always deny their implication in the construction of a pirate-device and claim that it was the malicious system-manager that implicated them instead (e.g. some malicious employee of the system-manager) . Of course this does not prohibit the system-manager from taking uni-lateral measures against such users (e.g. disconnect them from the service); however if non-repudiation was possible, this would drastically increase the effectiveness of the traitor tracing scenario against piracy: in a traitor-tracing scheme with non-repudiation, where the tracing procedure produces solid proof for the implication of the traitors, the system-manager might press criminal charges against subscribers that leak their key-information, thus significantly lowering the commercial viability of piracy. Note that in such a scheme tracing becomes a much more challenging task: on the one hand the system-manager should know enough information about the subscriber keys to execute the tracing algorithm; on the other hand the system-manager should be oblivious to a significant portion of a subscriber's personalized-key (otherwise the system-manager would be capable of implicating innocent users in the construction of a pirate-device). This asymmetry of knowledge assuring non-repudiation, gave these schemes the name "asymmetric traitor tracing."

The existence of asymmetric traitor tracing schemes was shown by Pfitzmann [25], who also introduced the setting of *asymmetric traitor tracing*. Nevertheless, this first scheme, was merely a plausibility result that employed generic secure function evaluation techniques (that are completely impractical). Later, the problem was also studied in the context of fingerprinting [26, 27]. A "somewhat asymmetric" public-key traitor tracing scheme was presented in [20], using a threshold mechanism to ensure the non-repudiation property (i.e. the capability to implicate innocent users was divided to a number of authorities, who had to collude in order to break the asymmetry of the scheme). This is not a real solution to asymmetry (since it was not as originally defined in the harder model of a *single* authority which performs the management functions). Further, the underlying traitor tracing mechanism of the scheme of [20] was later [29, 4] shown to fail against collusions of traitors which include more than a single user. Thus, the problem of efficient (single-authority) public-key traitor tracing remained open, until the recent introduction of two proposals by Watanabe et al. [31] and Komaki et al. [19].

Our results: In the present work we show the following:

- First, we break the two previous proposals of [31] and [19] for efficient asymmetric public-key traitor tracing: we show that for both schemes it is possible for the traitors to evade tracing.
- Second, we present an efficient Asymmetric Public-Key Traitor Tracing scheme of which we prove its traceability in detail (in the non-black box traitor tracing model). Our scheme is capable of proving the implication of *all* traitors that participate in the construction of a pirate-key. As a result our proposal is the first efficient Asymmetric Public-Key Traitor Tracing Scheme. In fact our scheme is comparable in efficiency to previous non-asymmetric traitor tracing schemes as illustrated in figure 2.

	Ciphertext Size	User-Key Size	Encryption-Key Size	System Type	Asymmetry
[8] scheme 1	$\mathcal{O}(v^4 \log n)$	$\mathcal{O}(v^2 \log n)$	$\mathcal{O}(v^2 \log n)$	generic	NO
[4]	$2v+1$	$\mathcal{O}(1)$	$2v+1$	public-key	NO
Our Scheme	$2v+2$	$\mathcal{O}(1)$	$2v+2$	public-key	YES

Figure 2: Comparison of our public-key traitor-tracing scheme with previous work. Note that n is the number of users and v is a parameter of the system, that denotes the maximum size of a traitor collusion that the system can withstand.

Remark: We would like to note that even though we actually break the schemes of [31] and [19] (demonstrating the high level of care which is needed to design such subtle systems), their contributions are nevertheless significant, since we have learned a lot from their underlying mechanisms. In particular they introduced the idea of using Oblivious Polynomial Evaluation [23], as the basic building block to achieve asymmetry in public-key traitor tracing schemes; something that we also take advantage in our scheme. Thus, the earlier works represent important steps towards the development of the efficient asymmetric scheme.

2 Preliminaries

We work in a multiplicative cyclic group $\mathcal{G}$ of large prime order over which solving the Decisional Diffie Hellman (DDH) Problem is hard:

Definition 1 DDH. *Let $g \in \mathcal{G}$ be a generator. Consider triples of the form* R, *$\langle g^a, g^b, g^c \rangle$ with $a, b, c <$ order(g) and triples of the form* D, *$\langle g^a, g^b, g^{ab} \rangle$ with $a, b <$ order(g). A predicate solves the DDH problem if it can distinguish the collection* D *from the collection* R.

The DDH-Assumption *for $\mathcal{G}$ suggests that any predicate that solves the DDH problem has distinguishing probability negligible in* $\log(\text{order}(g))$.

For example $\mathcal{G}$ can be the subgroup of order q of $\mathbf{Z}_p^*$, where $q \mid p-1$ and p, q are large primes. In the following g will denote a generator of $\mathcal{G}$. Note that arithmetic in the exponents is performed in the finite field $\mathbf{Z}_q$.

Let $h_0, h_1, \ldots, h_v$ be random elements of $\mathcal{G}$ so that $h_j := g^{r_j}$ for $j = 0, \ldots, v$. For a certain element $y := g^b$ of $\mathcal{G}$ a representation of y with respect to the base $h_0, \ldots, h_v$ is a $(v+1)$-vector $\vec{\delta} := \langle \delta_0, \ldots, \delta_v \rangle$ such that $y = h_0^{\delta_0} \ldots h_v^{\delta_v}$, or equivalently $\vec{\delta} \cdot \vec{r} = b$ where $\cdot$ denotes the inner

product between two vectors. It is well known (see e.g. [6]) that obtaining representations of a given y w.r.t. some given base $h_0, \ldots, h_v$ is as hard as the discrete-log problem over $\mathcal{G}$.

2.1 Oblivious Polynomial Evaluation

A tool that is instrumental in obtaining the non-repudiation property of our scheme is Oblivious Polynomial Evaluation (OPE). An OPE protocol involves two parties, the sender S, who possesses a secret polynomial $P \in \mathbf{Z}_q[x]$, and the receiver R who possesses a secret value $\alpha \in \mathbf{Z}_q$. An OPE protocol allows the receiver to compute the evaluation of the sender's polynomial P over its secret value α (i.e. compute $P(\alpha)$) in such a way so that:

- The sender S cannot extract any non-trivial information about the value α.
- The receiver R cannot extract any information about the polynomial P, other than what can be trivially extracted from the value $P(\alpha)$.

Oblivious Polynomial Evaluation was introduced by Naor and Pinkas in [23], where an efficient construction with two communication flows was presented. The security of the scheme was based on Oblivious Transfer and an intractability assumption related to the Polynomial Reconstruction Problem (see e.g. [14, 18]). Later, in [7], it was shown that efficient OPE protocols can be based on the generic assumption of Oblivious Transfer.

Here, we will assume a two communication flow protocol (such as those of [23, 7]) where $\{OPE\}(\alpha)$ denotes the data transmitted by the receiver R to the sender S in the first flow, and $\{OPE\}(P(\alpha))$ denotes the data transmitted by the sender to the receiver in the second communication flow. According to the properties of Oblivious Polynomial Evaluation as described above, $\{OPE\}(\alpha)$ does not yield any non-trivial information about the value α, and $\{OPE\}(P(\alpha))$ contains enough information for the receiver to compute $P(\alpha)$ but not any further non-trivial information about the secret polynomial P.

The variant of Oblivious Polynomial Evaluation that we will use has two additional properties. (1) it is malleable: i.e. given $\{OPE\}(\alpha)$ the sender can easily compute $\{OPE\}(\alpha+\alpha')$, for a given (e.g., random) α' and $+$ an operation in the underlying finite field; and (2) it is performed over a publicly committed value, namely α can be thought of as a private key whose public key is publicly known. Indeed, protocols such as [23] can have these properties due to their structure and using generic techniques. In addition, an OPE protocol can be designed directly to achieve these properties very efficiently [17]. This scheme is robust and exploits zero-knowledge proofs to assure that the receiver is acting on the committed value. In particular we show how it is possible for the receiver to convince the sender that the submitted string $\{OPE\}(\alpha)$ is (i) properly formed, and (ii) it agrees with the public commitment value. Further, the scheme is simulatable in the sense that one can produce protocol transcripts on any given public commitment value. Our design will exploit the above properties in a crucial way.

2.2 Asymmetric Public-Key Traitor Tracing Schemes

An Asymmetric Public-Key Traitor Tracing scheme involves the following entities: the system-manager, who is responsible for administrating the system, issuing subscriber information, and tracing pirate devices; the subscribers, or users, of the system; the channel-providers who use the system to distribute scrambled data to the set of subscribers; and finally, the "judge"

who verifies that certain subscribers have been implicated in the construction of a pirate-device. Note that the judge is not assumed to be a trusted party, in fact any interested party can play the role of the judge. Nevertheless, we use this terminology to emphasize that the tracing procedure should produce solid evidence for the implication of the traitor users in the construction of a pirate-device.

The description of an asymmetric public-key traitor-tracing scheme is comprised of the following procedures and requirements:

- Join. A protocol between the system-manager and a new user that introduces the new user as a subscriber to the system. The join procedure will result in a personalized key for each new subscriber. The join procedure is a critical component in the context of asymmetric schemes: on the one hand the subscriber should commit to his/her key in a non-repudiable way; on the other hand, the system-manager should be oblivious to a portion of the subscriber key (to prevent a malicious system-manager from implicating an innocent user).
- Encryption. A probabilistic procedure that can be used by any third party to send encrypted messages to the set of users.
- Decryption. An algorithm that can be used by any user, in combination with his/her secret-key, to decrypt a message.
- Traitor-Tracing and Trial. An algorithm that given the contents of a pirate-decoder can be used by the system-manager to reveal the identities of the traitor users that participated in the construction of the decoder by revealing their keys. The algorithm generates non-repudiated information which can be verified in a trial by a judge; see figure 3.

The security of the asymmetric public-key traitor tracing scheme can be modeled by employing standard notions of security pertaining to public-key encryption. In particular, we remark that definitions of semantic-security against passive or chosen-ciphertext adversaries can be employed to model security in our setting without any modifications.

Further, the scheme is "asymmetric" if it satisfies the following properties: (i) *frameproof*: the system-manager is incapable of implicating innocent users in the construction of a pirate decoder; (ii) *direct non-repudiation*: tracing should produce indisputable proof for the implication of the traitors in the construction of the pirate decoder; such proof should be impossible to forge by the system-manager and any interested third party (the judge) can check its validity *without the participation of the subscribers of the system* (hence the name *direct* non-repudiation).

3 Flaws in proposed Asymmetric Public-Key Schemes

In this section we demonstrate that the two previous proposals of asymmetric public-key traitor tracing schemes are flawed.

3.1 The scheme of [31] is Flawed

Let us briefly describe the scheme; for more details the reader is referred to [31]. Following the join procedure, each legitimate user i of the system possesses a value $f(z_i, \alpha_i)$ of a bi-

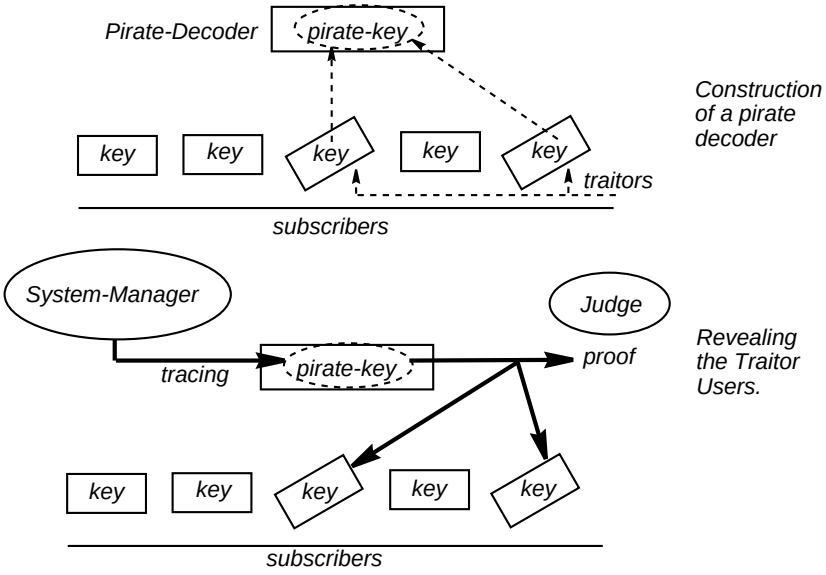

Figure 3: Traitor Tracing in an Asymmetric Scheme

variate polynomial $f(x,y) = f_1(x) + yf_2(y)$ where $f_1(x) := a_0 + a_1x + \ldots + a_kx^k$ and $f_2(x) = b_0 + b_1x + \ldots + b_kx^k$ are two secret random polynomials of $\mathbf{Z}_q[x]$ (selected by the system-manager during the initialization of the scheme). The join procedure is executed so that the α_i value is not revealed to the authority but nevertheless the authority obtains a commitment of the user to this value (to be used as a proof in case of piracy).

The public-key of the scheme is $\langle g, h_{0,0}, \ldots, h_{0,k}, h_{1,0}, \ldots, h_{1,k}\rangle$ so that $h_{0,i} := g^{a_i}$ and $h_{1,i} := g^{b_i}$ for $i = 1, \ldots, k$. Encryption in this scheme works as follows: given a message $s \in \mathcal{G}$ the sender selects random $r, x_1, \ldots, x_k \in \mathbf{Z}_q$ and computes:

$$\langle g^r, s \cdot h_{1,0}^r, h_{0,0}^r, \ldots, h_{0,k}^r, (x_1, \prod_{j=0}^{k} h_{1,j}^{rx_1^j} = g^{rf_2(x_1)}), \ldots, (x_k, \prod_{j=0}^{k} h_{1,j}^{rx_k^j} = g^{rf_2(x_k)})\rangle$$

The receiver, a user that possesses $f(z_i, \alpha_i) = f_1(z_i) + \alpha_i f_2(z_i)$, decrypts a ciphertext $\langle G, G', G_0, \ldots, G_k, (x_1, G_1'), \ldots, (x_k, G_k')\rangle$ as follows: first the value $\gamma := (G^{f(z_i,\alpha_i)} / \prod_{j=0}^{k} G_j^{z_i^j})^{\alpha_i^{-1}}$ is computed. Note that the value γ equals $g^{rf_2(z_i)}$. Then the user computes the Lagrange coefficients $\lambda, \lambda_1, \ldots, \lambda_k$ so that $\lambda f(z_i) + \lambda_1 f(x_1) + \ldots + \lambda_k f(x_k) = f(0)$ for any polynomial f of degree at most k (note that this is only possible if $z_i \notin \{x_1, \ldots, x_k\}$ but this can only happen with negligible probability); observe that $\lambda f_2(z_i) + \lambda_1 f_2(x_1) + \ldots + \lambda_k f_2(x_k) = f_2(0)$. Subsequently the user computes the plaintext s by evaluating $G'/(\gamma^\lambda \prod_{j=1}^{k} (G_j')^{\lambda_j})$.

In [31] it is shown that it is computationally hard for a collusion of up to k users to compute another decryption key $\langle z, \alpha, f(z,\alpha)\rangle$. Nevertheless this is not sufficient to ensure tracing. Attacks in previous traitor tracing schemes (in particular against the scheme of [20]) used techniques to combine user-key information in an arbitrary fashion thus disabling "direct" tracing by merely observing the keys found inside the pirate-decoder (see [29, 4]). Watanabe et al. [31] claim that such techniques do not seem to apply in their scheme: "*On the other hand, it seems [that such techniques are] not applicable to the threshold-decryption-based scheme such*

as [Yoshida et al] and ours, since a session key can be computed by combining $k+1$ shares using the Lagrange interpolation, and simple convex combination of the personal keys of k traitors does not lead to the pirate key." (page 400, [31]).

Here we show that this assumption is, in fact, false:

Claim 1. Any collusion of traitors of more than a single user, can generate keys that are not traceable in the scheme of [31]. Such keys are random linear combinations of the traitors' keys.

The Break. The break depends on the following fact that we show about the [31] scheme: Given t user-keys $\langle z_i, \alpha_i, f(z_i, \alpha_i)\rangle$ the vector $\langle \delta_0, \ldots, \delta_k, \delta'_0, \ldots, \delta'_k, \Delta\rangle :=$

$$\langle \sum_{\ell=1}^{t} \mu_\ell, \sum_{\ell=1}^{t} \mu_\ell z_\ell, \ldots, \sum_{\ell=1}^{t} \mu_\ell z_\ell^k, \sum_{\ell=1}^{t} \mu_\ell \alpha_\ell, \sum_{\ell=1}^{t} \mu_\ell \alpha_\ell z_\ell, \ldots, \sum_{\ell=1}^{t} \mu_\ell \alpha_\ell z_\ell^k, \sum_{\ell=1}^{t} \mu_\ell f(z_\ell, \alpha_\ell)\rangle$$

can also be used as a key, where $\mu_1, \ldots, \mu_t$ are random elements of $\mathbf{Z}_q$. This can be seen as follows: given the ciphertext $\langle G, G', G_0, \ldots, G_k, (x_1, G'_1), \ldots, (x_k, G'_k)\rangle$, a pirate device employing a key of the above form, computes $\gamma := G^{\Delta} / \prod_{j=0}^{k} G_j^{\delta_j}$. Observe that $\gamma = g^{r \sum_{\ell=1}^{t} \mu_\ell \alpha_\ell f_2(z_\ell)}$.

Next the pirate device needs to compute values $\lambda, \lambda_1, \ldots, \lambda_k$ so that $\lambda \sum_{\ell=1}^{t} \mu_\ell \alpha_\ell f(z_\ell) + \lambda_1 f(x_1) + \ldots + \lambda_k f(x_k) = f(0)$ for any polynomial $f \in \mathbf{Z}_q[x]$ of degree at most k. If such values can be computed, then the message can be recovered by the pirate device by computing $G' / (\gamma^{\lambda} \prod_{j=1}^{k} (G'_j)^{\lambda_j})$.

To complete the description of our break, we show how the values $\lambda, \lambda_1, \ldots, \lambda_k$ can be computed using only the information provided in the pirate-key and the current ciphertext. First observe that if A is a $(k+1) \times (k+1)$-matrix so that its i-th row is equal to $\langle 1, x_i, \ldots, x_i^k\rangle$ for $i = 1, \ldots, k$ and its $(k+1)$-th row equals $\langle \delta'_0, \ldots, \delta'_k\rangle$ it holds that A is non-singular with very high probability provided that $z_1, \ldots, z_t$ do not belong to $\{x_1, \ldots, x_k\}$ (something that can only happen with negligible probability). Then, the system $A \cdot \langle b_0, \ldots, b_k\rangle^T = \langle f(x_1), \ldots, f(x_k), \sum_{\ell=1}^{t} \mu_\ell \alpha_\ell f(z_\ell)\rangle^T$ is solvable for any polynomial $f \in \mathbf{Z}_q[x]$ of degree at most k and defines the coefficients of $f(x)$. It is an immediate conclusion that $b_0 = f(0)$ can be defined as a linear combination of the values $\langle f(x_1), \ldots, f(x_k), \sum_{\ell=1}^{t} \mu_\ell \alpha_\ell f(z_\ell)\rangle$ and the coefficients $\lambda, \lambda_1, \ldots, \lambda_k$ of the linear combination depend only on the matrix A, which is accessible to the pirate device given the pirate-key and the ciphertext.

As a result, a pirate, using keys of the above form, produces pirate-devices that the tracing procedure of the scheme of [31] (which is merely based on checking whether the key(s) found in a pirate-device are equal to some of the subscriber keys) is incapable to trace.

We remark that in [31], a "black-box traitor tracing" algorithm is presented as well. It is based on "black-box confirmation" (as defined by [4]). Such traitor-tracing methods require *exponential-time* in the number of traitors, and are, therefore, *not* practical for many scenarios.

3.2 The Tracing Scheme of [19] requires Exponential Time

Let us briefly describe the scheme; for details the reader is referred to [19]. Following the Join procedure, each legitimate user i of the system obtains a point of a random secret polynomial $f(x) := a_0 + a_1 x + \ldots + a_{2k-1} x^{2k-1}$ (this polynomial is privately selected by the system-manager during the initialization of the scheme). As a result, each subscriber will obtain a point $\langle d_i, f(d_i)\rangle$. Note that this point is not known to the system-manager; in fact this is the crucial point that is used by [19] to show the asymmetry property of their scheme. The public-key of the system is set to $\langle g, g^{a_0}, \ldots, g^{a_{2k-1}}\rangle$ and the encryption is defined as follows: a sender encrypts a message s by $\langle g^r, s \cdot (g^{a_0})^r, (g^{a_1})^r, \ldots, (g^{a_{2k-1}})^r\rangle$. A ciphertext $\langle G, G', G_1, \ldots, G_{2k-1}\rangle$

can be decrypted by any subscriber of the system (that possesses a point $\langle d_i, f(d_i)\rangle$ of f) as follows:

$$s = G' \prod_{j=1}^{2k-1} G_j^{d_i^j} / G^{f(d_i)}$$

It is proven in [19] that a coalition of less than $2k$ traitors is incapable of constructing another subscriber key $\langle d, f(d)\rangle$. However it is possible to construct vectors that can be used as keys by taking linear combinations of the form

$$\langle \sum_{\ell=1}^{t} \mu_\ell, \sum_{\ell=1}^{t} \mu_\ell d_\ell, \ldots, \sum_{\ell=1}^{t} \mu_\ell d_\ell^{2k-1}, \sum_{\ell=1}^{t} \mu_\ell f(d_\ell)\rangle$$

Such vectors can be constructed by a traitor collusion of t subscribers and it is not apparent how tracing can be achieved in this case. This fact is mentioned in the paper and it is claimed it is possible to trace those combinations to the users that created them by using coding-theoretic techniques.

Claim 2. Tracing in the scheme of [19] requires exponential time.

Justification. The coding theoretic methods that Komaki et al. ([19]) claim they can use for tracing in their scheme, (note that they do not present a concrete tracing algorithm), require the tracer to *know* the points $\langle d_i, f(d_i)\rangle$ assigned to the users during the Join protocol. However, these same values are also required to be unknown in order to achieve the claimed asymmetry/non-repudiation. This fact went unnoticed in [19], and renders the proposed traceability procedure exponential time, as the tracer will have to use in the decoding algorithm *all* possible values of the underlying finite field $\mathbf{Z}_q$ which is exponentially large (the size of an element in the underlying finite field coincides with the security parameter of the system).

4 The New Scheme

In this section we present our public-key asymmetric traitor tracing scheme. In order to achieve the asymmetry property it is necessary to use a basic underlying mechanism that can provide non-repudiation. To this effect, we assume that every user u possesses a digital signature mechanism sign_u that allows him/her to sign messages. The signature of user u on a message M will be denoted by $\mathsf{sign}_u(M)$. Any interested party can verify the signature of user u on a message M by running the publicly available verification algorithm verify_u.

Initialization. The system-manager selects one random polynomial $Q_1(x) = a_0 + a_1 x + \ldots a_{2v} x^{2v}$ over $\mathbf{Z}_q$ and a random $b \in \mathbf{Z}_q$ and sets $y = g^{a_0}$ and $h_0 = g, h_1 = g^{-a_1}, \ldots, h_{2v} = g^{-a_{2v}}, h' = g^{-b}$. The tuple $\langle y, h_0, \ldots, h_{2v}, h'\rangle$ is published as the public-key of the system. Let $Q(x, y) := Q_1(x) + by$.

Join. The join procedure is a protocol executed by the system-manager and a new user u that wants to obtain the subscription service. The goal of the Join protocol is to allow the user u to compute a point $\langle z_u, \alpha_u, Q(z_u, \alpha_u)\rangle$ of the bi-variate polynomial Q so that: z_u is randomly selected by the system manager, and $\alpha_u = \alpha_u^C + \alpha_c^R$ where α_u^C is a value selected and committed by the user, and the value α_u^R is randomly selected by the system manager. The commitment of the user u to the value α_u^C will be of the form $\langle C_u = g^{\alpha_u^C}, \mathsf{sign}_u(C_u)\rangle$.

The join protocol can be implemented by employing an instantiation of a Malleable OPE over a committed value as specified in subsection 2.1.

After the completion of the Join procedure, the user's secret personal key will be set to the vector $\vec{\kappa}_u := \langle Q(z_u, \alpha_u), z_u, z_u^2, \ldots, z_u^{2v}, \alpha_u \rangle$ (note that the user u does not need to store the whole $\vec{\kappa}_u$ as this can be recovered from the values $z_u, \alpha_u, Q(z_u, \alpha_u)$ as needed; as a result the storage space needed for the secret-key is not proportional to v — however the working space in the receiver should be proportional to v).

The following proposition asserts that the join procedure allows the user to compute a valid secret-key of the system, that is not known (in its entirety) by the system-manager; instead, the system-manager holds a non-repudiable commitment of the user to the secret portion of the user's secret-key.

Proposition 2 *The key $\vec{\kappa}_u$ computed by user u is a representation of y w.r.t. the base $h_0, \ldots, h_{2v}, h'$.*

Proof. If $\vec{\kappa}_u = \langle Q(z_u, \alpha_u), z_u, z_u^2, \ldots, z_u^{2v}, \alpha_u \rangle$, observe that

$$(h_0)^{Q(z_u,\alpha_u)}(h_1)^{z_u} \ldots (h_{2v})^{z_u^{2v}}(h')^{\alpha_u} = g^{Q(z_u,\alpha_u) - a_1 z_u - \ldots - a_{2v} z_u^{2v} - b\alpha_u} = g^{a_0} = y$$

□

Next note that due to the properties of the OPE we are assured that if the join protocol terminates successfully, user u is committed to the secret-key $\vec{\kappa}_u$ with overwhelming probability; if the protocol is aborted by the system-manager then the user u cannot compute any representation of y w.r.t the base $h_0, \ldots, h_{2v}, h'$. Further, note that due to the malleability and security of the OPE variant, it follows that an oracle to an invocation of the registration procedure can be simulated by giving a truly random point of the bivariate polynomial Q. This suggests that a malicious adversary cannot create subscribers for which he controls the point over which the system's polynomial Q is evaluated.

The next proposition, assures us that based on the hardness of the discrete logarithm problem, certain limitations on the structure of the pirate keys are imposed. This approach follows the one of Boneh and Franklin [4].

Proposition 3 *Suppose there exists an adversary, that given the public-key $\langle y, h_0, \ldots, h_{2v}, h' \rangle$ of the system and $t < 2v + 2$ random values $\langle z_i, \alpha_i, Q(z_i, \alpha_i) \rangle$ of the bivariate polynomial Q, it outputs a representation of y w.r.t. the base $h_0, \ldots, h_{2v}, h'$ denoted by $\vec{K} = \langle \delta_0, \ldots, \delta_{2v}, \delta' \rangle$ that is not a linear combination of the vectors $\langle Q(z_i, \alpha_i), z_i, \ldots, z_i^{2v}, \alpha_i \rangle$. Then the discrete-log problem over $\mathcal{G}$ is solvable.*

Proof. Let $\langle g, G \rangle$ be an instance of the discrete-log problem over the group $\mathcal{G}$. Consider the following algorithm that uses the adversary as follows: first we select $z_1, \ldots, z_t$, $\alpha_1, \ldots, \alpha_t, a_0, a_1, \ldots, a_{2v}, b$ at random from $\mathbf{Z}_q$. We set $Q_1(x) = a_0 + a_1 x + \ldots a_{2v} x^{2v}$, and $Q(x, y) = Q_1(x) + by$. Then we select a $(2v+2)$-tuple $\langle b_0, b_1, \ldots, b_{2v}, b' \rangle$ at random, from the (right-)kernel of the matrix

$$\begin{pmatrix} Q(z_1, \alpha_1) & z_1 & \ldots & z_1^{2v} & \alpha_1 \\ Q(z_2, \alpha_2) & z_2 & \ldots & z_2^{2v} & \alpha_2 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ Q(z_t, \alpha_t) & z_t & \ldots & z_t^{2v} & \alpha_t \end{pmatrix}$$

Observe that since $t < 2v+2$ and the above matrix is of full rank with very high probability (as the z_i's are assumed distinct, and the α_i's are random). Thus, it follows that its right-kernel contains q^{2v+2-t} vectors. The system-manager gives to the adversary the public-key

$$\langle y, h_0, h_1, \ldots, h_{2v}, h' \rangle = \langle g^{a_0}, gG^{b_0}, g^{-a_1}G^{b_1}, \ldots, g^{-a_{2v}}G^{b_{2v}}, g^{-b}G^{b'} \rangle$$

Then, we give to the adversary the values $\langle Q(z_i, \alpha_i), z_i, \alpha_i \rangle$. Finally the adversary outputs a representation $\langle \delta_0, \ldots, \delta_{2v}, \delta' \rangle$ such that it is not a linear combination of the vectors $\langle Q(z_i, \alpha_i), z_i, \ldots, z_i^{2v}, \alpha_i \rangle_{i=1}^{t}$. Now observe that the matrix below is also of full rank, and that its right-kernel contains $q^{2v+2-t-1}$ vectors.

$$\begin{pmatrix} Q(z_1,\alpha_1) & z_1 & \ldots & z_1^{2v} & \alpha_1 \\ Q(z_2,\alpha_2) & z_2 & \ldots & z_2^{2v} & \alpha_2 \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ Q(z_t,\alpha_t) & z_t & \ldots & z_t^{2v} & \alpha_t \\ \delta_0 & \delta_1 & \ldots & \delta_{2v} & \delta' \end{pmatrix}$$

It follows that the probability that $\langle \delta_0, \delta_1, \ldots, \delta_{2v}, \delta' \rangle \cdot \langle b_0, b_1, \ldots, b_{2v}, b' \rangle$ equals to 0 is at most $1/q$. Finally observe that: $y = h_0^{\delta_0} \ldots h_{2v}^{\delta_{2v}} (h')^{\delta'}$ and as a result

$$\log_g G = (b_0\delta_0 + \ldots b_v\delta_v + b'\delta')^{-1}(a_0 + a_1\delta_1 + \ldots + a_{2v}\delta_{2v} + b\delta' - \delta_0)$$

This completes the proof. □

Encryption. Any (non-trusted) channel provider can use the encryption function to distribute content to the set of subscribers. The encryption operation is defined as follows: the channel provider obtains the public-key of the system, $\mathsf{pk} := \langle y, h_0, \ldots, h_{2v}, h' \rangle$. A plaintext M is encrypted as follows: $\langle y^r \cdot M, h_0^r, \ldots, h_{2v}^r, (h')^r \rangle$, where r is a random integer less than q. So the (probabilistic) encryption function is defined as follows:

$$\mathcal{E}(\mathsf{pk}, M) = \langle y^r \cdot M, h_0^r, \ldots, h_{2v}^r, (h')^r \rangle$$

As a result, $\mathcal{E}$, is an extended ElGamal encryption, for which one can easily show:

Proposition 4 *The encryption function of the system is semantically secure under the Decisional Diffie Hellman Assumption over the group $\mathcal{G}$.*

The proof extends the standard semantic security of ElGamal encryption to discrete-log representations of arbitrary length, see e.g. [4]. We remark that semantic security against a chosen ciphertext security can be achieved also, following the techniques of [10], as it was demonstrated in [4].

Decryption. Any ciphertext can be decrypted using a representation of y w.r.t. the base $h_0, \ldots, h_{2v}, h'$. Given a ciphertext $\tilde{G} := \langle G, G_0, G_1, \ldots, G_{2v}, G' \rangle$ and a representation $\vec{\kappa} := \langle \delta_0, \ldots, \delta_{2v}, \delta' \rangle$ the decryption function is defined as follows:

$$\mathcal{D}(\tilde{G}, \vec{\kappa}) := G/((G')^{\delta'} \prod_{j=0}^{2v} (G_j)^{\delta_j})$$

It is easy to verify that the decryption operation inverts the encryption function. Indeed, for a ciphertext $\tilde{G} = \langle y^r \cdot M, h_0^r, \ldots, h_{2v}^r, (h')^r \rangle$ and a representation $\vec{\kappa} = \langle \delta_0, \ldots, \delta_{2v}, \delta' \rangle$ of y w.r.t. $\langle h_0, \ldots, h_{2v}, h' \rangle$, it holds that,

$$\mathcal{D}(\tilde{G}, \vec{\kappa}) = \frac{y^r \cdot M}{((h')^r)^{\delta'} \prod_{j=0}^{2v} (h_j^r)^{\delta_j}} = \frac{y^r M}{y^r} = M$$

Traitor Tracing. It is clear from the definition of the decryption function that any representation of y w.r.t. the base $h_0, \ldots, h_{2v}, h'$ can be used as a decryption-key. Under the security of the encryption function, the set of all possible decryption keys is identified with the set of all representations of y. A malicious coalition of users (traitors) is capable of producing arbitrary representations but, under the hardness of the discrete-logarithm problem, proposition 3 suggests that these representations can only be linear combinations of the secret-keys the malicious coalition possesses. In particular if $u_1, \ldots, u_t$ constitute a malicious collusion of users then they can compute representations of y of the form $\sum_{\ell=1}^{t} \mu_\ell \vec{\kappa}_{u_\ell}$ where $\mu_1, \ldots, \mu_t$ are random elements of $\mathbf{Z}_q$.

In order to achieve asymmetric traitor tracing, the problem that needs to be resolved is defined in figure 4.

Input. A vector $\vec{K}$ of the form $\sum_{\ell=1}^{t} \mu_\ell \vec{\kappa}_{u_\ell}$, where $\{u_1, \ldots, u_t\} \subseteq \{1, \ldots, n\}$ is the set of traitor users and n is the number of all users in the system. Recall that $\vec{\kappa}_u = \langle Q(z_u, \alpha_u), z_u, \ldots, z_u^{2v}, \alpha_u \rangle$. Also part of the input are integers $z_1, \ldots, z_n$, that define a portion of the secret-key $\vec{\kappa}_u$ of every user (recall that during the Join protocol these values are selected by the system-manager).
Output. The indices $\{u_1, \ldots, u_t\}$ and the values $\{\mu_1, \ldots, \mu_t\}$.

Figure 4: The Traitor Tracing Problem

Below we present an efficient algorithmic construction for the Traitor Tracing problem based on Decoding of Algebraic Codes. We are motivated by the work of [4] and [24], that presented similar techniques for traitor tracing based on linear codes decoding. For an introduction to Coding Theory the reader is referred to [21]. First, the tracer, defines the following $(n \times 2v)$-matrix:

$$H := \begin{pmatrix} z_1 & \ldots & z_1^{2v} \\ z_2 & \ldots & z_2^{2v} \\ \vdots & \ldots & \vdots \\ z_n & \ldots & z_n^{2v} \end{pmatrix}$$

Note that the number of users of the system is typically much larger than the parameter v, and as a result we assume that $n > 2v$. Let $\mathcal{C}$ define the code over $\mathbf{Z}_q^n$ that has H as a parity-check matrix (i.e. for all $\vec{c} \in \mathcal{C}$ it holds that $\vec{c} \cdot H = \mathbf{0}$).

Now let $\lambda_1, \ldots, \lambda_n$ be the Lagrange coefficients so that $\lambda_1 g(z_1) + \ldots + \lambda_n g(z_n) = g(0)$, for all $g \in \mathbf{Z}_q[x]$ with degree$(g) < n$.

Lemma 5 *It holds that 1.* $\mathcal{C} = \{\langle \lambda_1 M(z_1), \ldots, \lambda_n M(z_n) \rangle \mid M \in \mathbf{Z}_q[x], \text{degree}(M) < n - 2v\}$. *2.* $\mathcal{C}$ *is a linear code with message-rate* $(n - 2v)/n$ *and distance* $2v + 1$.

Proof. 1. Let $\mathcal{C}'$ denote the linear space in the right-hand-side of the equality above. If $\langle c_1, \ldots, c_n \rangle \in \mathcal{C}'$, it is of the form $\langle \lambda_1 M(z_1), \ldots, \lambda_n M(z_n) \rangle$. Then it is easy to verify that

$\langle c_1, \ldots, c_n \rangle$ belongs to $\mathcal{C}$: indeed it holds that $\langle c_1, \ldots, c_n \rangle \cdot \langle z_1^\ell, \ldots, z_n^\ell \rangle = \sum_{i=1}^n \lambda_i M(z_i) z_i^\ell$, for any $\ell = 1, \ldots, 2v$. Now observe that $\sum_{i=1}^n \lambda_i M(z_i) z_i^\ell = 0$ by the choice of $\lambda_1, \ldots, \lambda_n$ (and the fact that degree$(M) < n - 2v$). Since $\langle z_1^\ell, \ldots, z_n^\ell \rangle$ is the ℓ-th column of H, it follows that $\langle c_1, \ldots, c_n \rangle \cdot H = \mathbf{0}$. This shows that $\mathcal{C}' \subseteq \mathcal{C}$. On the other hand observe that $\dim(\mathcal{C}) = n - 2v = \dim(\mathcal{C}')$. Since $\mathcal{C}'$ is a linear sub-space of $\mathcal{C}$ and it has the same dimension, it follows that $\mathcal{C} = \mathcal{C}'$.

Item *2*, is straightforward from item *1*: in particular a vector of $\mathbf{Z}_q^{n-2v}$ can be encoded as the coefficients of a polynomial $M \in \mathbf{Z}_q[x]$ of degree less than $n - 2v$. The corresponding codeword of $\mathcal{C}$ will be the vector $\langle \lambda_1 M(z_1), \ldots, \lambda_n M(z_n) \rangle$. To see that the distance of the linear code is $2v + 1$ observe that any two different codewords of $\mathcal{C}$ can agree on at most $n - 2v - 1$ positions, or equivalently any two distinct codewords differ on at least $2v + 1$ positions. □

Next we show that $\mathcal{C}$ is a linear code that allows efficient error-correction. In particular it is clear from lemma 5 that $\mathcal{C}$ is a Generalized Reed-Solomon Code (for the definition of Generalized Reed-Solomon Codes, see [21]). Generalized Reed-Solomon Codes can be decoded efficiently by the algorithm of Berlekamp and Welch [3]. This means that for any vector $\vec{x} \in \mathbf{Z}_q^n$ for which there exists a vector $\vec{w} \in \mathcal{C}$ that disagrees with $\vec{x}$ in at most e positions with $e \leq \frac{n-(n-2v)}{2} = v$, it holds that $\vec{w}$ is unique with this property ($\mathcal{C}$ is a maximum-distance-separable code) and the vector $\vec{w}$ can be recovered in deterministic polynomial-time.

Let us now proceed to describe the tracing procedure. Given $\vec{K} = \sum_{\ell=1}^t \mu_\ell \vec{\kappa}_{u_\ell}$, denote $\vec{K} = \langle K_0, K_1, \ldots, K_{2v}, K' \rangle$; recall that $\{u_1, \ldots, u_t\}$ is the set of traitor users. The tracer concentrates on the $(2v)$-vector $\vec{\eta} = \langle K_1, \ldots, K_{2v} \rangle$. By the definition of $\vec{\eta}$ it holds that there exists a vector $\vec{\nu} = \langle \nu_1, \ldots, \nu_n \rangle$ with $\nu_{u_\ell} = \mu_\ell$ for all $\ell = 1, \ldots, t$ and $\nu_i = 0$ for $i \notin \{u_1, \ldots, u_t\}$, with the property $\langle \nu_1, \ldots, \nu_n \rangle \cdot H = \vec{\eta}$. It is immediate that the recovery of $\vec{\nu}$ yields the solution to the traitor tracing problem as defined in figure 4.

The tracer computes an arbitrary vector $\vec{\delta}$ that satisfies the system of equations $\vec{\delta} \cdot H = \vec{\eta}$. Note that such $\vec{\delta}$ can be found by standard linear algebra since $\vec{\delta} \cdot H = \vec{\eta}$ is a system of $2v$ equations with n unknowns, $n > 2v$, and H contains a non-singular minor of size $2v$. It is easy to verify that the vector $\vec{w} := \vec{\delta} - \vec{\nu}$ belongs to the linear code $\mathcal{C}$: indeed, $\vec{w} \cdot H = \vec{\delta} \cdot H - \vec{\nu} \cdot H = \vec{\eta} - \vec{\eta} = \mathbf{0}$. As a result the vector $\vec{\delta}$ can be expressed as $\vec{\delta} = \vec{w} + \vec{\nu}$.

Provided that $t \leq v$ it holds that the Hamming weight of $\vec{\nu}$ is less or equal to v and as a result $\vec{\delta}$ is a n-vector that differs in at most v positions from the vector $\vec{w}$ that belongs in $\mathcal{C}$. Due to the properties of the linear code $\mathcal{C}$ it holds that $\vec{w}$ will be the unique vector of $\mathcal{C}$ with this property, and furthermore $\vec{w}$ can be recovered in deterministic polynomial-time if we feed $\vec{\delta}$ to the decoding procedure for $\mathcal{C}$ (which is essentially the Berlekamp-Welch algorithm, [3]). The recovery of $\vec{w}$, immediately will result in the recovery of $\vec{\nu} = \vec{\delta} - \vec{w}$. As mentioned above the recovery of $\vec{\nu}$ solves the traitor tracing problem.

Efficiency. Our tracing algorithm has time complexity $\mathcal{O}(n^2)$, if the Berlekamp-Welch algorithm is implemented in the straightforward manner; more efficient implementations are possible that reduce the time-complexity to $\mathcal{O}(n(\log n)^2)$. We remark that it is possible to trace even if the number of traitors exceeds the bound v; this can be done by employing the decoding algorithm of Guruswami and Sudan [14] that will produce a *list* of possible sets of traitor users, provided that the size of the traitor collusion is less or equal to $n - \sqrt{n(n-2v)}$.

The Trial. The system-manager obtains the output of the tracing procedure on the pirate key $\vec{K} = \langle K_0, K_1, \ldots, K_{2v}, K' \rangle$ with $\vec{K} = \sum_{\ell=1}^t \mu_\ell \vec{\kappa}_{u_\ell}$ where $\{u_1, \ldots, u_t\} \subseteq \{1, \ldots, n\}$ is the set of traitor users (note that due to proposition 3 the pirate-key is ensured to be of this form). The output of the tracing is the vector $\vec{\nu} = \langle \nu_1, \ldots, \nu_n \rangle$ where $\nu_{u_\ell} = \mu_\ell$ for $\ell = 1, \ldots, t$

and $\nu_i = 0$ for all $i \in \{1,\ldots,n\} - \{u_1,\ldots,u_t\}$. The system-manager has to prove to the judge that the users $\{u_1,\ldots,u_t\}$ were implicated in the construction of the pirate key $\vec{K}$. This can be done as follows: the system-manager transmits to the judge the vector $\vec{\nu}$, the pirate key $\vec{K}$, the values $\alpha^R_{u_1},\ldots,\alpha^R_{u_t}$ and the commitments of the implicated subscribers $C_{u_1};\mathsf{sign}_{u_1}(C_{u_1}),\ldots\ C_{u_t};\mathsf{sign}_{u_t}(C_{u_t})$ that were generated when the traitors joined the system as subscribers. The judge verifies all signatures using the publicly known verification algorithms $\mathsf{verify}_{\mathsf{u_1}},\ldots,\mathsf{verify}_{\mathsf{u_t}}$.

Then, the judge examines whether the claim of the system-manager regarding the implication of the users $\{u_1,\ldots,u_t\}$ is correct; this is done as follows: the judge tests whether

$$\prod_{\ell=1}^{t}(C_{u_\ell}g^{\alpha^R_{u_\ell}})^{\nu_{u_\ell}} \stackrel{?}{=} g^{K'}$$

if the test passes, then the judge concludes that indeed the users $\{u_1,\ldots,u_t\}$ were implicated in the construction of the pirate key $\vec{K}$.

Theorem 6 *1. If $\{u_1,\ldots,u_t\}$ are the traitor users whose keys have been used in the construction of the pirate-key $\vec{K}$, the judge will verify this fact using the information provided by the tracer.*
2. Under the security of the underlying malleable OPE, if the system-manager can implicate an innocent user in the construction of a pirate-key $\vec{K}$ it follows that the discrete-log problem over $\mathcal{G}$ is solvable with overwhelming probability.

Proof. 1. First, observe that $K' = \sum_{\ell=1}^{t}\mu_\ell\alpha_{u_\ell}$. Because of the properties of the tracing procedure it holds that $\nu_{u_\ell} = \mu_\ell$ for $\ell = 1,\ldots,t$. Since $C_u = g^{\alpha^C_u}$ for all $u \in \{1,\ldots,n\}$, it holds that $\prod_{\ell=1}^{t}(C_{u_\ell}g^{\alpha^R_{u_\ell}})^{\nu_{u_\ell}} = g^{K'}$ (recall that $\alpha_{u_\ell} = \alpha^C_{u_\ell} + \alpha^R_{u_\ell}$).

2. Suppose that the system-manager convinces the judge that users $u_0,u_1,\ldots,u_t$ are implicated in the pirate-key $\vec{K}$ but $\vec{K} = \sum_{\ell=1}^{t}\mu_\ell\vec{\kappa_{u_\ell}}$ (i.e. the key of user u_0 is not among the ones that are used in the definition of $\vec{K}$). Since the judge agrees to the implication of the users $u_0,u_1,\ldots,u_t$ it holds that $\prod_{\ell=0}^{t}(C_{u_\ell}g^{y_\ell})^{\nu_{u_\ell}} = g^{K'}$, where K', $\vec{\nu}$, $y_0,\ldots,y_t$ are supplied by the system-manager. Below we show how to use such a cheating system manager to solve the discrete log problem over $\mathcal{G}$.

Given a challenge for the discrete-logarithm problem $\langle g, G\rangle$ over $\mathcal{G}$, we simulate the Join protocol for the users $u_0,\ldots,u_t$. On the one hand, we execute the Join protocol exactly as defined for users $u_1,\ldots,u_t$; on the other hand, we simulate the join protocol so that the public commitment value of user u_0, denoted by C_{u_0} equals G. (recall that the Join protocol — based on the underlying OPE — is "simulatable", namely, we can run simulations of the Join protocol on a given public commitment $C_u = g^x$ for which we do not know the discrete log mod g, with overwhelming probability of success). Subsequently, we construct a pirate-key $\vec{K}$ based on the values $z_{u_1},\alpha_{u_1},Q(z_{u_1},\alpha_{u_1}),\ldots,z_{u_t},\alpha_{u_t},Q(z_{u_t},\alpha_{u_t})$ (which we know, from the output of the Join protocol for the users $u_1,\ldots,u_t$), and we give this value to the system-manager. Then, we obtain the values $x_0,x_1,\ldots,x_t,y_0,\ldots,y_t$ and K' by simulating the system-manager; it follows that the discrete-logarithm of G can be computed as

$$\log_g(G) = (x_0)^{-1}(K' - x_1\alpha_{u_1} - \ldots - x_t\alpha_{u_t} - y_0x_0 - \ldots y_tx_t)(\mathrm{mod}q)$$

This completes the proof. □

Black-Box Traitor Tracing. Black-box traceability is an important enhancement of the traitor tracing procedure, where the tracer is capable of recovering the identities of the traitor users using merely black-box access to the pirate-decoder. Not surprisingly our scheme is not likely to satisfy this property (in an efficient way) as it belongs in the family of public-key traitor tracing schemes that includes the schemes of [20, 4] that cannot support this desirable enhanced traceability property in an efficient way as shown in [15]. A weaker form of black-box traitor tracing, called black-box confirmation, that can be potentially applied in this family of schemes was presented in [5]; this technique can give a black-box traitor tracing algorithm that has exponential running-time in the number of traitors and is applicable to our scheme as well.

Practical Considerations. Broadcasting streams of digital content using solely the encryption function of a traitor tracing scheme can be quite expensive. A content distributor takes the most out of such a scheme, if it broadcasts short-lived session keys (suitable for a block-cipher such as the AES [11]) to all subscribers using the encryption function of the TTS, and then in each session uses the block-cipher to scramble the digital content stream. Sessions should be *short* so that users are discouraged from distributing the (not fingerprinted) session-keys. There is an evident trade-off between the degree of protection against piracy and the efficiency of the digital content distribution scheme with respect to the session-length parameter. Finding an optimal value for this parameter is important for a practical implementation of a traitor tracing scheme in a certain context; it involves risk assessment, weighing damages against the cost of repetitive distribution.

Note that we assumed a system-manager which performs honestly in subscribing users (during the Join protocol). In case this is not sufficient, we can require the system-manager to sign the Join protocol transcript and keep the corresponding private record. This will enable complaints against the system-manager to be solved in court as well.

References

[1] *Canal Plus files $ 1 billion lawsuit on News Corp arm*, Reuters, 03.12.02, 7:46 PM ET. (Also http://www.wired.com/news/politics/0,1283,51005,00.html).

[2] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan and Ke Yang, *On the (Im)possibility of Obfuscating Programs*, CRYPTO 2001.

[3] Elwyn R. Berlekamp and L. Welch, *Error Correction of Algebraic Block Codes.* U.S. Patent, Number 4,633,470 1986.

[4] Dan Boneh and Matthew Franklin, *An Efficient Public Key Traitor Tracing Scheme*, CRYPTO 1999.

[5] Dan Boneh and Matthew Franklin, *An Efficient Public Key Traitor Tracing Scheme*, manuscript, full-version of [4], 2001.

[6] Stefan Brands, *Rethinking Public Key Infrastructures and Digital Certificates – Building in Privacy*, Ph.D. thesis, Technical University of Eindhoven, 1999.

[7] Yan-Cheng Chang and Chi-Jen Lu, *Oblivious Polynomial Evaluation and Oblivious Neural Learning*, Asiacrypt 2001.

[8] Benny Chor, Amos Fiat, and Moni Naor, *Tracing Traitors*, CRYPTO 1994.

[9] Benny Chor, Amos Fiat, Moni Naor, and Benny Pinkas, *Tracing Traitors*, IEEE Transactions on Information Theory, Vol. 46, no. 3, pp. 893-910, 2000.

[10] Ronald Cramer and Victor Shoup, *A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack*, CRYPTO 1998.

[11] J. Daemen and V. Rijmen, *The design of Rijndael- AES the advanced encryption standard*, Springer Verlag, 2002.

[12] Amos Fiat and Moni Naor, *Broadcast Encryption* CRYPTO 1993.

[13] Eli Gafni, Jessica Staddon and Yiqun Lisa Yin, *Efficient Methods for Integrating Traceability and Broadcast Encryption*, CRYPTO 1999.

[14] Venkatesan Guruswami and Madhu Sudan, *Improved Decoding of Reed-Solomon and Algebraic-Geometric Codes.* In the Proceedings of the 39th Annual Symposium on Foundations of Computer Science, IEEE Computer Society, pp. 28–39, 1998.

[15] Aggelos Kiayias and Moti Yung, *Self Protecting Pirates and Black-Box Traitor Tracing*, CRYPTO 2001.

[16] Aggelos Kiayias and Moti Yung, *Traitor Tracing with Constant Transmission Rate*, Eurocrypt 2002.

[17] Aggelos Kiayias and Moti Yung, *Robust Malleable Oblivious Polynomial Evaluation*, manuscript.

[18] Aggelos Kiayias and Moti Yung, *Cryptographic Hardness Based on the Decoding of Reed-Solomon Codes*, ICALP 2002.

[19] Hirotaka Komaki, Yuji Watanabe, Goichiro Hanaoka, and Hideki Imai, *Efficient Asymmetric Self-Enforcement Scheme with Public Traceability*, Public Key Cryptography 2001.

[20] K. Kurosawa and Y. Desmedt, *Optimum Traitor Tracing and Asymmetric Schemes*, Eurocrypt 1998.

[21] F. J. MacWilliams and N. Sloane, *The Theory of Error Correcting Codes.* North Holland, Amsterdam, 1977.

[22] Dalit Naor, Moni Naor and Jeffrey B. Lotspiech *Revocation and Tracing Schemes for Stateless Receivers*, CRYPTO 2001.

[23] Moni Naor and Benny Pinkas, *Oblivious Transfer and Polynomial Evaluation.* In the Proceedings of the 31th ACM Symposium on the Theory of Computing, 1999. (Full version available from authors).

[24] Moni Naor and Benny Pinkas, *Efficient Trace and Revoke Schemes* , In the Proceedings of Financial Crypto '2000, Anguilla, February 2000.

[25] Birgit Pfitzmann, *Trials of Traced Traitors*, Information Hiding Workshop, Spring LNCS 1174, pp. 49-63, 1996.

[26] Birgit Pfitzmann and Matthias Schunter, *Asymmetric Fingerprinting*, Eurocrypt 1996.

[27] Brigitt Pfitzmann and M. Waidner, *Asymmetric fingerprinting for larger collusions*, in proc. ACM Conference on Computer and Communication Security, pp. 151–160, 1997.

[28] Tomas Sander and Christian F. Tschudin, *On Software Protection via Function Hiding*, Information Hiding 1998.

[29] Douglas Stinson and Ruizhong Wei, *Key preassigned traceability schemes for broadcast encryption*, In the Proceedings of SAC'98, Lecture Notes in Computer Science 1556, Springer Verlag, pp.144–156, 1998.

[30] Douglas R. Stinson and Ruizhong Wei, *Combinatorial Properties and Constructions of Traceability Schemes and Frameproof Codes*, SIAM J. on Discrete Math, Vol. 11, no. 1, 1998.

[31] Yuji Watanabe, Goichiro Hanaoka and Hideki Imai *Efficient Asymmetric Public-Key Traitor Tracing without Trusted Agents*, CT-RSA 2001.