This is the full version of the extended abstract which appears in
Information Security Conference—ISC'07 (9 october 2007 – 12 october 2007, Chili)
J. Garay Ed., Springer-Verlag, LNCS 4779, pages 71-88.

Traitor Tracing with Optimal Transmission Rate

Nelly Fazio[1]*, Antonio Nicolosi[2]*, and Duong Hieu Phan[3]**

[1] IBM Almaden Research Center
nfazio@us.ibm.com
[2] New York University and Stanford University
nicolosi@scs.stanford.edu
[3] France Telecom R&D and University of Paris 8
duonghieu.phan@orange-ftgroup.com

Abstract. We present the first traitor tracing scheme with efficient black-box traitor tracing in which the ratio of the ciphertext and plaintext lengths (the *transmission rate*) is asymptotically 1, which is optimal. Previous constructions in this setting either obtained constant (but not optimal) transmission rate [KY02b], or did not support black-box tracing [CPP05]. Our treatment improves the standard modeling of black-box tracing by additionally accounting for pirate strategies that attempt to escape tracing by purposedly rendering the transmitted content at lower quality. Our construction relies on the *decisional bilinear Diffie-Hellman* assumption, and attains the same features of public traceability as (a repaired variant of) [CPP05], which is less efficient and requires non-standard assumptions for bilinear groups.

Keywords: Traitor Tracing, Constant Transmission Rate, Fingerprint Codes, Bilinear Maps.

1 Introduction

Traitor tracing schemes constitute a very useful tool against piracy in the context of digital content distribution. They are multi-recipient encryption schemes that can be employed by *content providers* that wish to deliver copyrighted material to an exclusive set of users. Each user holds a decryption key that is fingerprinted and bound to his identity. If a group of subscribers (the *traitors*) collude to construct an illegal device (the *pirate decoder*), the *security manager* can run a specialized traitor tracing algorithm to uncover the source of the leakage. Therefore, a traitor tracing scheme deters subscribers of a distribution system from leaking information by the mere fact that the identities of the leaking entities can then be revealed.

The first formal definition of traitor tracing scheme appears in Chor *et al.* [CFN94,CFNP00], whose construction requires storage and decryption complexity $O(t^2 \log^2 t \log(n/t))$ and communication complexity $O(t^3 \log^4 t \log(n/t))$, where n is the size of the universe of users and t is an upper bound on the number of traitors. Stinson and Wei later suggested in [SW98] explicit combinatorial construction that achieve better efficiency for small values of t and n.

The work of [NP98,CFNP00] introduced the notion of *threshold* traitor tracing scheme, where the tracing algorithm is only required to guarantee exposure of the traitors' identities for pirate decoders whose decryption probability is better than a given threshold β. The scheme of [NP98] achieves storage complexity $O(t/\beta \log(t/\varepsilon))$, where ε is the probability of successfully tracing one of the traitors. Moreover, the scheme has communication complexity linear in t and constant decryption complexity.

In [BF99], Boneh and Franklin present an efficient public-key traitor tracing scheme with deterministic t-tracing based on an algebraic approach. Its communication, storage and decryption complexities are all $O(t)$. The authors also introduce the notion of *non-black-box traceability*: given a "valid" key extracted from a pirate device (constructed using the keys of at most t users), recover the identity of at least one traitor. This is in contrast with the notion of *black-box tracing* (on which we focus in this paper), where the traitor's identity can be uncovered by just observing the pirate decoder's replies on "well crafted" ciphertexts.

* Research conducted in part while visiting ENS, Paris and Stanford University, CA.
** Research conducted in part while at ENS, Paris and University College London.

More recently, Boneh *et al.* [BSW06,BW06] proposed traitor tracing schemes that withstand any number of traitors (*full traceability*), while requiring a sub-linear ciphertext length ($O(\sqrt{n})$). In [Pfi96], Pfitzmann introduces the notion of *asymmetric* traitor tracing. In this model, tracing uncovers some secret information about the traitor that was *a priori* unknown to the system manager. Thus, the result of the tracing algorithm provides actual evidence of the treachery. Further results in this direction are in [KD98,KY02c,KY02a].

Alternative traitor tracing solutions [FT01,BPS00,SW03] have also been proposed to fight leakage of the decrypted content, rather then leakage of the decryption capabilities.

As originally observed in [GSY99], traitor tracing scheme are most useful when combined with a revocation scheme; such trace-and-revoke approach consists in first uncovering the compromised decryption keys and then revoking their decryption capabilities, thus rendering the corresponding pirate decoder useless [NP00,TT01,NNL01,DF02,DF03,KHL03,DFKY05,BW06].

Constant Transmission Rate. All proposals mentioned so far result into schemes that are not quite communication-efficient: the length of each ciphertext is (at least) t times longer than the embedded plaintext. As pointed out by Kiayias and Yung in [KY02b], an important problem in designing practical traitor tracing schemes is to ensure a low *transmission rate*, defined as the asymptotic ratio of the size of ciphertexts over the size of plaintexts, while at the same time minimize the *secret-* and the *public-storage* rates, similarly defined as the asymptotic ratio of the size of user-keys and of public-keys over the size of plaintexts.[1] Under this terminology, the transmission rate of all the above mentioned solutions is linear w.r.t. the maximal number t of traitors, whereas in [KY02b], Kiayias and Yung show that if the plaintexts to be distributed are large (which is the case for most applications of traitor tracing, such as distribution of multimedia content), then it is possible to obtain ciphertexts with constant expansion rate. Their solution is based on collusion-secure fingerprint codes [BS98,Tar03] and its parameters are summarized in Figure 1.

Besides the clear benefit in terms of communication efficiency, schemes with constant transmission rate also enjoy efficient black-box traceability, while schemes with linear transmission rate are inherently more limited in this regard [KY01] (*e.g.,* the black-box traitor tracing of [BF99] takes time proportional to $\binom{n}{t}$).

In [CPP05], Chabanne *et al.* extend the setting of [KY02b] with the notion of *public traceability*: Whereas traditional tracing algorithms require knowledge of the system's secret information, in a scheme with public traceability everyone can run the tracing algorithm. In this paper, we also consider *local public traceability*, whereby public information suffices to carry out the preliminary phase of tracing, which requires interaction with the pirate decoder, and results in an encoding of the traitor's identity that can be decoded with a master key. This separation of tasks ensures that the system's secret information is only needed for off-line operations (*i.e.*, user registration and possibly the final phase of tracing), thus improving the overall security of the system by allowing for safer storage solutions.

The work of [PST06] describes a traitor tracing scheme with constant (but not optimal) transmission rate and (full) public traceability based on Identifiable Parent Property (IPP) codes. Figure 1 also reports on these two schemes. One could think that traitor tracing schemes with linear transmission rate (*e.g.* [BF99]) could easily be turned into schemes with constant transmission rate by means of hybrid encryption: To send a large message, pick a random session key, encrypt it with the given traitor tracing scheme, and append a symmetric encryption of the message under the chosen anonymous session key. This approach, however, suffers from a simple yet severe untraceable pirate strategy: Just decrypt the session key and make it available to the "customers" on the black market, *e.g.,* via anonymous e-mail, or via text-messaging from a pre-paid cellphone. Clearly, when a traitor tracing scheme is used to encrypt the content directly, this "re-broadcasting" strategy becomes much less appealing for would-be pirates, because of the higher costs and exposure risks associated with running a high-bandwidth darknet.

[1] We adopt a terminology slightly different from the one of [KY02b], which uses the term *ciphertext/user-key/public-key* rates, for what we called *transmission/secret-storage/public-storage* rates. Moreover, in [KY02b] *transmission rate* refers to the sum of the all the three rates. Our choice is of course mostly a matter of taste: we prefer the terminology of this paper as it makes more evident the role played by each quantity in a concrete implementation of the system.

	Transmission Rate	Secret-Storage Rate	Public-Storage Rate	BB Tracing	Public Traceability	Hardness Assumption
[BF99]	$2t+1$	$2t$	$2t+1$	$\times$	$\times$	DDH
[KY02b]	3	2	4	$\surd *$	$\times$	DDH
[CPP05]	1	2	1	$\times$	$\times$	$\mathsf{DBDH}^2\text{-}\mathsf{E} \wedge \mathsf{DBDH}^1\text{-}\mathsf{M}$
[PST06]	7	1	1	$\surd$	full	DDH
Repaired [CPP05]	3	2	6	$\surd$	local	$\mathsf{DBDH}^2\text{-}\mathsf{E} \wedge \mathsf{DBDH}^1\text{-}\mathsf{M}$
Our Scheme	1	2	10	$\surd$	local	DBDH

Fig. 1. Comparison of rates (*transmission*, *secret-* and *public-storage* rates) and tracing features (*black-box tracing* and *public traceability*) between existing schemes and our construction. The "*" in the row labeled "[KY02b]" refers to the fact that the scheme of [KY02b] can support black-box tracing using the tracing algorithm that we describe in Appendix C.2. The row labeled "[PST06]" refers to instantiating their generic construction with ternary IPP codes and ElGamal-style encryption. The row labeled "Repaired [CPP05]" refers to the variant of the scheme of [CPP05] that we suggest in Appendix C.3 to support black-box tracing.

Our Contributions. We present the first public-key traitor tracing scheme with efficient black-box traitor tracing and local public traceability in which the transmission rate is asymptotically 1, which is optimal. Encryption involves the same amount of computation as in [CPP05]; decryption is twice as fast. We also considerably simplify the computational hardness requirements, relying just on the DBDH assumption—much weaker and more widely accepted than the non-standard bilinear assumptions employed in [CPP05].

Our treatment improves the standard modeling of black-box tracing by additionally accounting for pirate strategies that attempt to escape tracing by purposedly rendering the transmitted content at lower quality (*e.g.* by dropping every other frame from the decrypted video-clip, or skipping few seconds from the original audio file).

As additional contribution, we point out and resolve an issue in the black-box tracing of [KY02b] (which was also independently addressed in a revised version of their work [KY06]). We then show that [CPP05], which extends [KY02b] and inherits its tracing mechanism, inherits in fact the above-mentioned problem, too. In this case, however, fixing the black-box functionality requires changes that intrinsically conflict with the optimizations put up by [CPP05] to achieve optimal transmission rate. In other words, [CPP05] can either provide optimal transmission rate with only non-black-box tracing, or support local public traceability with sub-optimal transmission rate, but cannot achieve both at the same time.

Organization. Section 2 introduces the tools needed for our construction. Section 3 defines the syntactic, security, and traceability properties of traitor tracing schemes. We present our new traitor tracing scheme and its security analysis in Section 4, and Section 5 discusses a concrete choice of parameters. In the Appendix, we point out a flaw in the tracing algorithms of [KY02b] and [CPP05] and propose fixes.

2 Preliminaries

The security properties of our construction hinge upon the decisional bilinear Diffie-Hellman assumption (DBDH) for $(\mathcal{G}_1, \mathcal{G}_2)$. We refer the reader to Appendix A. for the relevant definitions.

Collusion-Secure Codes. Collusion-secure codes [BS98] provide a powerful tool against illegal redistribution of fingerprinted material in settings satisfying the following *Marking Assumption*: 1) it is possible to introduce small changes to the content at some discrete set of locations (the *marks*), while preserving the "quality" of the content being distributed; but 2) it is infeasible to apport changes to a mark without rendering the entire content "useless" *unless* one possesses two copies of the content that differ at that mark. Below, we include a formalization of the notion of collusion-secure codes, adapted from [BS98].

Definition 1. Let Σ be a finite alphabet, and $n, v \in \mathbb{Z}_{\geq 0}$. An (n, v)-code over Σ is a set of n v-tuples of symbols of Σ: $\mathcal{C} = \{\omega^{(1)}, \ldots, \omega^{(n)}\} \subseteq \Sigma^v$.

Definition 2. Let T be a subset of indices in $[1,n]$. The set of undetectable positions for T is: $R_T = \{\ell \in [1,v] \mid (\forall i,j \in T).[\omega_\ell^{(i)} = \omega_\ell^{(j)}]\}$.

Notice that for each $i \in T$, the projection of each codeword $\omega^{(i)}$ over the undetectable positions for T is the same; we denote this common projected sub-word as $\omega_{|R_T}$. By the Marking Assumption, any "useful" copy of the content created by the collusion of the users in T must result in a tuple $\bar{\omega}$ whose projection over R_T is also $\omega_{|R_T}$. This is captured by the following:

Definition 3. The set of feasible codewords for T is: $F_T = \{\bar{\omega} \in (\Sigma \cup \{?\})^v \mid \bar{\omega}_{|R_T} = \omega_{|R_T}\}$.

Definition 4. Let $\varepsilon > 0$ and $t \in \mathbb{Z}_{\geq 0}$. $\mathcal{C}$ is an (ε, t, n, v)-collusion-secure code over Σ if there exists a probabilistic polynomial-time algorithm $\mathcal{T}$ such that for all $T \subseteq [1,n]$ of size $\mid T \mid \leq t$, and for all $\bar{\omega} \in F_T$, it holds that: $\Pr[\mathcal{T}(r_\mathcal{C}, \bar{\omega}) \in T] \geq (1-\varepsilon)$, where the probability is over the random coins $r_\mathcal{C}$ used in the construction of the (n,v)-code $\mathcal{C}$, and over the random coins of $\mathcal{T}$.

3 Public-Key Traitor Tracing Scheme with Public Traceability

Definition 5 (Public-Key Traitor Tracing Scheme). A public-key traitor tracing scheme is a 5-tuple of probabilistic polynomial-time algorithms ($\mathbf{Setup}$, $\mathbf{Reg}$, $\mathbf{Enc}$, $\mathbf{Dec}$, $\mathbf{Trace}$), where:

$\mathbf{Setup}$**:** On input a security parameter 1^κ, a collusion threshold 1^t, and a bound n on the maximum number of users, returns a public key pk along with some master secret information msk (*cf.* $\mathbf{Reg}$ and $\mathbf{Trace}$);
$\mathbf{Reg}$**:** Given msk and a user index $i \in [1,n]$, outputs a "fingerprinted" user key sk_i;[2]
$\mathbf{Enc}$**:** On input key pk and a message m (from the appropriate message space $\mathcal{M}$, implicitly described by pk), returns a (randomized) ciphertext ψ;
$\mathbf{Dec}$**:** On input a user key sk_i and a ciphertext ψ, recovers the message encrypted within ψ;
$\mathbf{Trace}$**:** Given the master secret key msk, the public key pk, and black-box access to a "pirate" decoder capable of inverting the $\mathbf{Enc}(\mathsf{pk}, \cdot)$ functionality, returns the user index of one of the traitors that contributed his/her user key for the realization of the pirate decoder, or the special user index 0 upon failure.

For correctness, decryption with any user key output by $\mathbf{Reg}$ should "undo" encryption:

$$\Pr\left[\mathbf{Dec}(\mathsf{sk}_i, \mathbf{Enc}(\mathsf{pk}, m)) = m \left| \begin{array}{c} (\mathsf{pk}, \mathsf{msk}) \stackrel{R}{\leftarrow} \mathbf{Setup}(1^\kappa, 1^t, n), m \stackrel{R}{\leftarrow} \mathcal{M}, \\ i \stackrel{R}{\leftarrow} [1,n], \mathsf{sk}_i \stackrel{R}{\leftarrow} \mathbf{Reg}(\mathsf{msk}, u) \end{array} \right.\right] = 1,$$

where the probability is over the random coins of $\mathbf{Setup}$, $\mathbf{Reg}$, $\mathbf{Enc}$, $\mathbf{Dec}$, and over the random selection of m from $\mathcal{M}$ and of i from $[1,n]$.

Definition 6 (Full/Local Public Traceability). A public-key traitor tracing scheme is said to support: 1) *public traceability* if the $\mathbf{Trace}$ algorithm can be implemented without the master secret key msk; or 2) *local public traceability* if the $\mathbf{Trace}$ algorithm can be split in an on-line phase, in which the pirate decoder can be queried without knowledge of the secret key, and an off-line phase, without access to the pirate decoder, that can retrieve the identity of the traitor from the master secret key and the output of the publicly executable on-line phase.

Requirements on the Encryption Functionality. For security, encryption of distinct messages under a traitor tracing scheme should look indistinguishable to any efficient algorithm that is allowed to pick the two messages based on the public key of the system, but without knowledge of any user key:

[2] Equivalently, we can think of $\mathbf{Setup}$ as outputting a vector of user keys, one per each user in the system; we will refer to either formalization interchangeably.

Definition 7 (Indistinguishability under Chosen-Plaintext Attack). A public-key traitor tracing scheme satisfies ε_{ind}-indistinguishability if, for any pair of probabilistic polynomial-time algorithms $(\mathcal{A}_1, \mathcal{A}_2)$, it holds that:

$$\Pr\left[\mathcal{A}_2(\mathsf{state}, \psi^*) = b^* \;\middle|\; \begin{array}{c} (\mathsf{pk}, \mathsf{msk}) \xleftarrow{R} \mathbf{Setup}(1^\kappa, 1^t, n), \\ (m_0, m_1, \mathsf{state}) \xleftarrow{R} \mathcal{A}_1(\mathsf{pk}), \\ b^* \xleftarrow{R} \{0,1\}, \psi^* \xleftarrow{R} \mathbf{Enc}(\mathsf{pk}, m_{b^*}) \end{array}\right] \leq \frac{1}{2} + \varepsilon_{\text{ind}},$$

where the probability is over b^*, and the random coins of $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathbf{Setup}$, and $\mathbf{Enc}$.

Requirements on the Tracing Functionality. Existing literature usually models black-box traceability as the ability to "extract" the identity of (at least) one traitor from pirate decoders that *correctly* invert the decryption algorithm (under appropriate efficiency and success probability constraints). This approach, however, is often criticized because it leaves the way open for pirate decoders that decrypt ciphertexts into plaintexts that closely resemble (but are not identical to) the original plaintexts. For example, in the context of media distribution, the pirate could purposedly remove few frames from the original video clip, or play the correct audio file at a lower sampling rate. Such pirates could still attract a share of the black market, and since they actually do not correctly invert the encryption functionality, the scheme's traceability guarantees often would do not apply to them. To account for pirate strategies of this sort, we allow traitors to specify a notion of "resemblance" in the form of a polynomial-time reflexive, symmetric binary relation $\mathcal{R}$ over plaintexts, with $\mathcal{R}(m, m') = 1$ if customers would accept m' as a reasonable replacement for m.[3] The only semantic constraint on $\mathcal{R}$ is that it shall not be so lax as to deem random[4] plaintexts similar to fixed ones, *i.e.*, the quantity $p_{\mathcal{R}} \doteq \max_{m \in \mathcal{M}} \Pr[\mathcal{R}(m, m') = 1 \mid m' \xleftarrow{R} \mathcal{M}]$ shall be negligible (otherwise tracing is impossible, since a keyless decoder could simply output a random plaintext as a "reasonable" decryption of any ciphertext). Similarly, tracing needs only be effective against efficient decoders $\mathcal{D}$ whose success probability $p_{\mathcal{D}} \doteq \Pr[\mathcal{R}(m, \mathcal{D}(\mathbf{Enc}(\mathsf{pk}, m))) = 1 \mid m \xleftarrow{R} \mathcal{M}]$ is non-negligible.

Definition 8. A public-key traitor tracing scheme is $\varepsilon_{\text{trac}}$-traceable if for any probabilistic polynomial-time traitor strategy $\mathcal{A}$, it holds that:

$$\Pr\left[\mathbf{Trace}^{\mathcal{D}(\cdot)}(\mathsf{pk}, \mathsf{msk}) \notin T \;\middle|\; \begin{array}{c} (\mathsf{pk}, \mathsf{msk}) \xleftarrow{R} \mathbf{Setup}(1^\kappa, 1^t, n), \\ (\mathcal{D}, \mathcal{R}) \xleftarrow{R} \mathcal{A}(\mathsf{pk})^{\mathbf{Reg}(\mathsf{msk}, \cdot)} \end{array}\right] \leq \varepsilon_{\text{trac}}$$

where $\mathcal{M}$ is the message space, $T \subseteq [1, n]$ is the set of up to t indices on which $\mathcal{A}$ queried the $\mathbf{Reg}(\mathsf{msk}, \cdot)$ oracle, $\mathcal{D}$ and $\mathcal{R}$ both run in probabilistic polynomial-time and are such that $p_{\mathcal{D}}$ is non-negligible and $p_{\mathcal{R}}$ is negligible, and the probability is over the coins of $\mathbf{Setup}$, $\mathbf{Reg}$, $\mathcal{A}$, $\mathcal{D}$ and $\mathbf{Trace}$.

Notice that Definition 8 subsumes the case that the traitor strategy $\mathcal{A}$ only produces a "good" pirate decoder $\mathcal{D}$ with a low (but non-negligible) probability: indeed, any such strategy can be "boosted" by simply keeping executing $\mathcal{A}$ on fresh random coins, until the pirate decoder $\mathcal{D}$ that $\mathcal{A}$ outputs is a good one (which can be efficiently tested by estimating $\mathcal{D}$'s decryption capability on the encryption of a random plaintext).

4 Public-Key Traitor Tracing with Public Traceability, Black-Box Tracing and Optimal Transmission Rate

Similarly to the schemes of [KY02b] and [CPP05], our construction is based on the use of an (ε, t, n, v)-collusion-secure code $\mathcal{C}$ over the alphabet $\{0, 1\}$ (*cf.* Definition 4). At a high level, the idea is to first define

[3] Alternatively, the resemblance relation $\mathcal{R}$ could be specified as a parameter of the scheme in the definition of the $\mathbf{Trace}$ algorithm.

[4] For the sake of simplicity, in this paper we discuss only the case of random sampling from $\mathcal{M}$, but the treatment generalizes to the case of other plaintext distribution with high min-entropy.

a two-user sub-scheme resilient against a single traitor, and then "concatenate" v instantiations of this sub-scheme according to the code $\mathcal{C}$; in particular, each user $i \in [1, n]$ is associated to a codeword $\omega^{(i)}$ in $\mathcal{C}$, and given decryption key $\mathsf{sk}_i \doteq (K_{1,\omega_1^{(i)}}, \ldots, K_{v,\omega_v^{(i)}})$, where $\omega_j^{(i)}$ is the j-th bit of the codeword $\omega^{(i)}$, and $K_{j,0}, K_{j,1}$ are the keys for the j-th instantiation of the basic two-user sub-scheme. Although the overall architecture that we follow is well-known, achieving optimal transmission rate along these lines requires solving a number of technical problems, on which we elaborate in Section 4.4.

4.1 Our Two-User Sub-Scheme

Setup: Given a security parameter 1^κ, the algorithm works as follows:

Step 1: Generate a κ-bit prime q, two groups $\mathcal{G}_1$ and $\mathcal{G}_2$ of order q, and an admissible bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$. Choose an arbitrary generator $P \in \mathcal{G}_1$.

Step 2: Pick random elements $a, b, c \in \mathbb{Z}_q^*$, and set $Q \doteq aP, R \doteq bP, h \doteq e(P, cP)$. Compute two linearly independent vectors (α_0, β_0) and (α_1, β_1) in $\mathbb{Z}_q$ such that $b\alpha_\sigma + a\beta_\sigma = c \bmod q$, for $\sigma \in \{0, 1\}$. The private key of the security manager is set to be $\mathsf{msk} \doteq (a, b, \alpha_0, \beta_0, \alpha_1, \beta_1)$.

Step 3: For $\sigma \in \{0, 1\}$, let $A_\sigma \doteq \alpha_\sigma R$ and $B_\sigma \doteq \beta_\sigma P$. Choose a universal hash function $H : \mathcal{G}_2 \to \{0, 1\}^\kappa$, and set the public key of the scheme to be the tuple

$$\mathsf{pk} \doteq (q, \mathcal{G}_1, \mathcal{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1).^5$$

The associated message space is $\mathcal{M} \doteq \{0, 1\}^\kappa$.

Reg: For $\sigma \in \{0, 1\}$, the secret key of user σ is set to be $\mathsf{sk}_\sigma \doteq \alpha_\sigma$. Notice that $cP = \alpha_\sigma R + \beta_\sigma Q$ and hence $h = e(P, cP) = e(P, \alpha_\sigma R) \cdot e(Q, \beta_\sigma P) = e(P, A_\sigma) \cdot e(Q, B_\sigma)$, for $\sigma \in \{0, 1\}$.

Enc: Given pk, anybody can encrypt a message $m \in \mathcal{M}$ by first selecting a random $k \in \mathbb{Z}_q$ and then creating the ciphertext $\psi \doteq \langle U, V, W \rangle \in \mathcal{G}_2 \times \mathcal{G}_1 \times \mathcal{M}$ where

$$U \doteq e(P, R)^k, \quad V \doteq kQ, \quad W \doteq m \oplus H(h^k)$$

Dec: Given a ciphertext $\psi = \langle U, V, W \rangle$, user σ computes $h^k = U^{\alpha_\sigma} \cdot e(V, B_\sigma)$ and recovers $m = W \oplus H(h^k)$. Correctness of the decryption algorithm is clear by inspection.

Trace: To trace a decoder $\mathcal{D}$ with resemblance relation, feed $\mathcal{D}$ with the "illegal" ciphertext $\hat{\psi} \doteq \langle e(P, R)^{k'}, kQ, \hat{m} \oplus H(e(P, A_\sigma)^{k'} e(Q, B_\sigma)^k) \rangle$, for random $\sigma \in \{0, 1\}$, $k, k' \in \mathbb{Z}_q$, $\hat{m} \in \mathcal{M}$. If the output m^* of $\mathcal{D}$ satisfies $\mathcal{R}(\hat{m}, m^*) = 1$, then return σ as the traitor's identity; otherwise, pick fresh random $\sigma \in \{0, 1\}, k, k' \in \mathbb{Z}_q, \hat{m} \in \mathcal{M}$ and repeat.

Before moving on to the security and traceability of our two-user scheme in the sense of Definitions 7 and 8 (*cf. Section 3*), we remark that **Trace** does not require knowledge of the master secret key msk, and thus it supports full public traceability (*cf.* Definition 6). Also, notice that decryption requires only one pairing computation.

4.2 Indistinguishability under Chosen-Plaintext Attack

Theorem 9. *Under the* DBDH *assumption for* $(\mathcal{G}_1, \mathcal{G}_2)$*, the scheme in Section 4.1 is secure w.r.t. indistinguishability under chosen-plaintext attack (*cf. *Definition 28 and Definition 7).*

[5] Note that there is no need to explicitly include h in the public key, as it can be derived as $h = e(P, A_\sigma) \cdot e(Q, B_\sigma)$. Caching the value of h, however, is recommendable when public storage is not at a premium, as that would save two pairing computations during encryption.

Proof. To a contradiction, let us assume that the scheme does not satisfy Definition 7 *i.e.*, there is an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that, given the public key $\mathsf{pk} = (q, \mathcal{G}_1, \mathcal{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$, can break the scheme with non-negligible advantage ε_{ind}. We then construct an algorithm $\mathcal{B}$ (whose running time is polynomially related to $\mathcal{A}$'s) that breaks the DBDH assumption with probability $\varepsilon_{\mathsf{DBDH}} = \varepsilon_{\text{ind}}$.

Algorithm $\mathcal{B}$ is given as input an instance (P', xP', yP', zP', h') of the DBDH problem in $(\mathcal{G}_1, \mathcal{G}_2)$; its task is to determine whether $h' = e(P', P')^{xyz}$, or h' is a random element in $\mathcal{G}_2$. $\mathcal{B}$ proceeds as follows:

Setup: $\mathcal{B}$ sets $P \doteq xP'$ and $Q \doteq P'$. Then, $\mathcal{B}$ picks $r \xleftarrow{R} \mathbb{Z}_q^*$, and sets $R \doteq rQ$. $\mathcal{B}$ now chooses $\beta_0, \beta_1 \xleftarrow{R} \mathbb{Z}_q^*$ and computes $B_0 \doteq \beta_0 P$ and $B_1 \doteq \beta_1 P$. Then, $\mathcal{B}$ sets $A_0 \doteq zP'$ and $h \doteq e(P, A_0) \cdot e(Q, B_0)$. Finally, $\mathcal{B}$ sets $A_1 \doteq A_0 + \beta_0 Q - \beta_1 Q$, so that in fact $h = e(P, A_\sigma) \cdot e(Q, B_\sigma)$, for $\sigma \in \{0, 1\}$, as required. $\mathcal{B}$ can now set $\mathsf{pk} \doteq (q, \mathcal{G}_1, \mathcal{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$ and send it to $\mathcal{A}_1$.

Challenge: $\mathcal{A}_1$ outputs two messages m_0, m_1 on which it wishes to be challenged, along with some state state to be passed to $\mathcal{A}_2$. To prepare the ciphertext, $\mathcal{B}$ picks random $b^* \in \{0, 1\}$, and sets

$$U \doteq e(P, yP')^r \ (= e(P, R)^y), V \doteq yP' \ (= yQ), W \doteq m_{b^*} \oplus H(h' \cdot e(yP', xP')^{\beta_0}).$$

(Notice that this implicitly defines $k = y$.) Then, $\mathcal{B}$ sends $\mathcal{A}_2$ the challenge ciphertext $\psi^* \doteq (U, V, W)$, along with the state information state.

Guess: Algorithm $\mathcal{A}_2$ outputs a guess $b' \in \{0, 1\}$. $\mathcal{B}$ returns 1 if $b' = b^*$ and 0 otherwise.

If $h' = e(P', P')^{xyz}$, then $\mathcal{A}_2$ gets a valid encryption of m_{b^*}, since (as we verify below) in this case the input to the hash function in the computation of W is just h^k:

$$\begin{aligned} h' \cdot e(yP', xP')^{\beta_0} &= e(P', P')^{xyz} \cdot e(yP', \beta_0(xP')) = e(xP', zP')^y \cdot e(P', \beta_0(xP'))^y \\ &= e(P, A_0)^y \cdot e(Q, B_0)^y = [e(P, A_0) \cdot e(Q, B_0)]^y = h^y = h^k, \end{aligned}$$

as required by the encryption algorithm. Therefore, in this case $\mathcal{A}$ will successfully guess $b' = b^*$ with probability $\varepsilon_{\text{ind}} + 1/2$.

On the other hand, when h' is a random element of $\mathcal{G}_2$, the input to H is a random value, independent of any other information in the adversary's view. Since H is chosen from a universal hash function family, its output is also (almost) uniformly random in $\{0, 1\}^\kappa$, so that the value of W (and hence the whole challenge ciphertext ψ^*) is completely independent from m_{b^*}. Thus, in this case $b' = b^*$ holds with probability $1/2$.

It follows that adversary $\mathcal{B}$ breaks the DBDH assumption with non-negligible advantage $\varepsilon_{\mathsf{DBDH}} = \varepsilon_{\text{ind}}$, contradicting our hardness assumption. □

4.3 Traceability

To assess the effectiveness of the **Trace** algorithm from Section 4.1, we start with some observations about the illegal ciphertexts that **Trace** uses in querying the decoder $\mathcal{D}$:

Definition 10 (Valid and Probe Ciphertexts). Let $\sigma \in \{0, 1\}$, $\hat{m} \in \mathcal{M}$, $\hat{U} \in \mathcal{G}_1$, $\hat{V} \in \mathcal{G}_2$, $\hat{W} = \hat{m} \oplus H(\hat{U}^{\alpha_\sigma} e(\hat{V}, B_\sigma))$, and $\hat{\psi} = \langle \hat{U}, \hat{V}, \hat{W} \rangle$. We say that the ciphertext $\hat{\psi}$ is:

- **valid**, if $\hat{U} = e(P, R)^k, \hat{V} = kQ$, for some $k \in \mathbb{Z}_q$;
- σ-**probe**, if $\hat{U} = e(P, R)^{k'}, \hat{V} = kQ$, for distinct $k, k' \in \mathbb{Z}_q$.

Lemma 11 (Indistinguishability of Valid *vs.* Probe Ciphertexts). *Under the* DBDH *assumption for* $(\mathcal{G}_1, \mathcal{G}_2)$*, given the public key* $\mathsf{pk} = (q, \mathcal{G}_1, \mathcal{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$ *and the secret key* $\mathsf{sk}_\tau = \alpha_\tau$ *of user* $\tau \in \{0, 1\}$ *(where* $A_\tau = \alpha_\tau R$*), it is infeasible to distinguish a valid ciphertext from a* τ*-probe.*

Proof. For simplicity, assume $\tau = 0$. We proceed by contradiction: assume there is an adversary $\mathcal{A}$ that, given the public key $\mathsf{pk} = (q, \mathcal{G}_1, \mathcal{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$ and the secret key α_0 of user 0, can distinguish valid ciphertexts from probes with probability ε. We then construct an algorithm $\mathcal{B}$ (whose running time is polynomially related to $\mathcal{A}$'s) that breaks the DBDH assumption with probability $\varepsilon_{\mathsf{DBDH}} = \varepsilon$.

Algorithm $\mathcal{B}$ is given as input an instance (P', xP', yP', zP', h') of the DBDH problem in $(\mathcal{G}_1, \mathcal{G}_2)$; its task is to determine whether $h' = e(P', P')^{xyz}$ or h' is a random element in $\mathcal{G}_2$. $\mathcal{B}$ proceeds as follows:

Setup: $\mathcal{B}$ lets $P \doteq xP'$, $Q \doteq P'$, $R \doteq yP'$, chooses $\alpha_0, \beta_0, \beta_1 \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes $A_0 \doteq \alpha_0 R$, $B_0 \doteq \beta_0 P$ and $B_1 \doteq \beta_1 P$. $\mathcal{B}$ also sets $A_1 \doteq A_0 + \beta_0 Q - \beta_1 Q$, which implicitly defines $h = e(P, A_0) \cdot e(Q, B_0) = e(P, A_1) \cdot e(Q, B_1)$. $\mathcal{B}$ now defines $\mathsf{pk} \doteq (q, \mathcal{G}_1, \mathcal{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$. Then, $\mathcal{B}$ prepares a challenge ciphertext $\hat{\psi} \doteq \langle \hat{U}, \hat{V}, \hat{W} \rangle$ by setting $\hat{U} \doteq h'$, $\hat{V} \doteq zP'(= zQ$, thus implicitly defining $k = z)$ and $\hat{W} \doteq \hat{m} \oplus H(\hat{U}^{\alpha_0} e(\hat{V}, B_0))$, for $\hat{m} \stackrel{R}{\leftarrow} \mathcal{M}$. At this point, $\mathcal{B}$ feeds $\mathcal{A}$ with pk, $\hat{\psi}$, and α_0.
Attack: $\mathcal{A}$ returns her guess to whether $\hat{\psi}$ is a valid ciphertext or a probe (w.r.t. the public key pk).
Break: $\mathcal{B}$ outputs yes or no accordingly.

If $h' = e(P', P')^{xyz}$, then $\mathcal{A}$ gets a valid ciphertext since $h' = e(xP', yP')^z = e(P, R)^z$, consistently with the value of $\hat{V} = zQ$, as required by the encryption algorithm. Otherwise, h' is a random value in $\mathcal{G}_2$, of the form $h' = e(P, R)^{k'}$, for some k' totally independent from $k = z$, and thus $\hat{\psi}$ is a 0-probe. Therefore, $\mathcal{B}$ breaks the DBDH assumption with the same advantage as $\mathcal{A}$'s *i.e.*, $\varepsilon_{\mathsf{DBDH}} = \varepsilon$. □

An important consequence of Lemma 11 is that pirate decoders created by user τ reply to τ-probes with an m^* such that $\mathcal{R}(\hat{m}, m^*) = 1$ with non-negligible probability $\hat{p}_{\mathcal{D}}$:

Corollary 12. *Let $\mathcal{D}$, $\mathcal{R}$ be the pirate decoder and resemblance relation output by a traitor strategy $\mathcal{A}$ based on the user key α_τ, such that $p_{\mathcal{D}}$ is non-negligible and $p_{\mathcal{R}}$ is negligible (*cf.* Definition 8). Let $\hat{\psi}$ be a τ-probe for a message $\hat{m} \stackrel{R}{\leftarrow} \mathcal{M}$. Under the* DBDH *assumption, $\hat{p}_{\mathcal{D}} \doteq \Pr[\mathcal{R}(\hat{m}, m^*) = 1 \mid m^* \stackrel{R}{\leftarrow} D(\hat{\psi})]$ is non-negligible.*

Proof. To a contradiction, assume $\hat{p}_{\mathcal{D}}$ to be negligible. We then construct an efficient algorithm $\mathcal{B}$ that, given pk and the secret key α_τ of a single user, distinguishes valid ciphertexts from τ-probes as follows: on input a ciphertext $\hat{\psi} = \langle \hat{U}, \hat{V}, \hat{W} \rangle$, $\mathcal{B}$ computes $\hat{m} \doteq \hat{W} \oplus H(\hat{U}^{\alpha_\tau} \cdot e(\hat{V}, B_\tau))$ from α_τ and $\hat{\psi}$. Notice that this value $\hat{m}$ is correct regardless of whether $\hat{\psi}$ is a valid ciphertext or a τ-probe. Then, $\mathcal{B}$ feeds $\mathcal{D}$ with $\hat{\psi}$, getting back a value m^*. If $\mathcal{R}(\hat{m}, m^*) = 1$, then $\mathcal{B}$ concludes that $\hat{\psi}$ must be valid; otherwise, $\mathcal{B}$ concludes that $\hat{\psi}$ is a τ-probe. In other words, $\mathcal{B}$ "interpolates" between the experiment defining probabilities $p_{\mathcal{D}}$ and $\hat{p}_{\mathcal{D}}$, so that $\mathcal{B}$'s advantage in discerning valid ciphertext from τ-probes is clearly $p_{\mathcal{D}} - \hat{p}_{\mathcal{D}}$. But if $\hat{p}_{\mathcal{D}}$ were negligible, such algorithm $\mathcal{B}$ would violate the statement of Lemma 11, proving our argument. □

The next lemma addresses the case of pirate decoders fed with probes of the "wrong type":

Lemma 13. *Replacing $\hat{\psi}$ with a $(1-\tau)$-probe in the setting of Corollary 12, $\Pr[\mathcal{R}(\hat{m}, m^*) = 1]$ is negligible.*

Proof. We start with the observation that if we could somehow remove the message $\hat{m}$ from the pirate decoder's view, then our thesis would follow immediately, since $\hat{m}$ would then be independent from the message m^* output by $\mathcal{D}$, and hence, by definition of $p_{\mathcal{R}}$, $\mathcal{R}(\hat{m}, m^*) = 1$ would hold with probability $p_{\mathcal{R}}$, which is negligible.

In fact, $\hat{m}$ occurs in $\mathcal{D}$'s view only in the third component of the $(1-\tau)$-probe $\hat{\psi} \doteq \langle \hat{U}, \hat{V}, \hat{W} \rangle$, as $\hat{W} = \hat{m} \oplus H(\hat{U}^{\alpha_{1-\tau}} e(\hat{V}, B_{1-\tau}))$, so it suffices to show that $\hat{U}^{\alpha_{1-\tau}} e(\hat{V}, B_{1-\tau})$ is indistinguishable from random in $\mathcal{D}$'s view. Since B_0, B_1 both appear in the public key pk of the system, this boils down to proving that $\mathcal{D}$ cannot distinguish $\hat{U}^{\alpha_{1-\tau}}$ from random. It also holds that $\hat{U}^{\alpha_{1-\tau}} = e(P, R)^{k'\alpha_{1-\tau}} = e(P, A_{1-\tau})^{k'}$, so that the task faced by $\mathcal{D}$ is to tell $e(P, A_{1-\tau})^{k'}$ apart from random, given $e(P, R)$, $e(P, A_{1-\tau})$, and $\hat{U} = e(P, R)^{k'}$. But this is just the DDH problem for group $\mathcal{G}_2$, whose hardness is implied by the DBDH assumption.

The above argument can be easily rephrased along the lines of the reductions described in the proofs of Theorem 9 and Lemma 11; we refrain from doing so due to space limitations. □

Theorem 14. *Under the* DBDH *assumption for* $(\mathcal{G}_1, \mathcal{G}_2)$*, our* $\mathbf{Trace}$ *algorithm has a negligible traceability error.*

Proof. Let $\mathcal{D}$, $\mathcal{R}$ be the pirate decoder and resemblance relation on which the $\mathbf{Trace}$ algorithm is being run, and let τ be the traitor index. Corollary 12 guarantees that $\mathbf{Trace}$ will on average terminate after $2/p_{\mathcal{D}}$ queries to $\mathcal{D}$. Upon termination, $\mathbf{Trace}$'s output will be wrong only if it happens that $\mathcal{D}$ replies to a $(1-\tau)$-probe $\hat{\psi}$ with an m^* satisfying $\mathcal{R}(\hat{m}, m^*) = 1$, *i.e.*, $\Pr[\mathbf{Trace}^{D(\cdot)}(pk, \perp) \notin T] = \Pr[\hat{\psi}$ is a $(1-\tau)$-probe $\mid \mathcal{R}(\hat{m}, m^*) = 1]$, which by Corollary 12, Lemma 13, and Bayes' theorem is easily seen to equal $p_{\mathcal{R}}/(p_{\mathcal{D}} + p_{\mathcal{R}})$, which is negligible. □

4.4 Our Multi-User Scheme

As mentioned at the beginning of Section 4, a common approach to extending a two-user construction to the multi-user setting is to concatenate several instantiations (say, v) of the basic two-user scheme. Tracing in the resulting multi-user scheme can then be performed iteratively as a sequence of v stages; in each stage, the pirate decoder is queried with ciphertexts that are valid in all v components, except for one, which instead is crafted according to the $\mathbf{Trace}$ algorithm of the two-user construction. In this way, if the decoder does not have both sub-keys for the component currently under testing, it will be unable to tell that the ciphertext is invalid, and so the tracing procedure of the two-user subscheme will determine which of the two sub-keys the decoder holds for that component.

Since tracing requires the ability to set up each component of the ciphertext independently of all the others, it may seem necessary to use completely unrelated instantiations of the two-user sub-scheme for each component. This is done, for example, in [KY02b]. (*cf.* Appendix B.2). Having independent components, however, clearly leads to a multi-user scheme with the same transmission rate as the underlying basic two-user scheme, and so it would not help us attaining optimal transmission rate. In fact, the scheme of [CPP05] (*cf.* Appendix B.4) manages to get transmission rate 1 by sacrificing component independence, and instead using component-instances all very closely related to each other. As we show in Appendix C.3, though, their scheme does not support black-box traceability.

To solve this tension between transmission rate and black-box traceability, we move from the observation that, at each stage, it suffices that a single component can be appropriately set up independently from the rest; the remaining $v-1$ can all be closely related to each other. Therefore, ciphertexts in our construction include a "special" position ℓ, where encryption is performed with instance of our two-user scheme that is specific to the ℓ-th component; the remaining $(v-1)$ positions, instead, are encrypted using a "shared" two-user scheme.

To prevent pirate decoders from selectively ignoring the "special" position (which is the only part of the ciphertext that encodes tracing information), we follow the approach proposed in [KY02b], by which the encryption algorithm preliminarily processes the plaintext with an *All-Or-Nothing* transform (AONT) [Riv97,Boy99,CDH$^+$00]. This will force decoders to decrypt *all* blocks of the ciphertext, since ignoring even a single one would result in missing at least one block of the AONT-transformed plaintext, so that, by the properties of AONT's, such decoders would fail to recover *any* information about the original plaintext being transmitted. We remark that reliance on AONT's to force the pirate to include (at least) one key for each component was suggested in [KY02b], but later dismissed by the authors in [KY06] as ineffective for the black-box setting, since it cannot prevent cropping of the plaintext once it has been decrypted. However, we believe their critique to be misleading, since traitor strategies in which the pirate decoder tampers with the decrypted plaintexts are dealt with the use of the resemblance relation $\mathcal{R}$ (see discussion in Section 3), while AONT's prevent the pirate from learning anything about the plaintext if even a single block cannot be decrypted.

For the sake of clarity, we first describe the scheme without explicitly mentioning the AONT pre-processing, and later discuss the details regarding the use of AONT's.

Setup: Given the security parameters 1^κ, 1^t and ε, the algorithm works as follows:

Step 1: Generate a κ-bit prime q, two groups $\mathcal{G}_1$ and $\mathcal{G}_2$ of order q, and an admissible bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$. Generate an (ε, t, n, v)-collusion-secure code $\mathcal{C} = \{\omega^{(1)}, \ldots, \omega^{(n)}\}$.

Step 2a: Generate v independent copies of the 2-user scheme described in Section 4.1 (call these copies the *special* schemes). In particular, for $j = 1, \ldots, v$, let P_j be a generator of $\mathcal{G}_1$; pick random elements $a_j, b_j, c_j \in \mathbb{Z}_q^*$, and set $Q_j \doteq a_j P_j$, $R_j \doteq b_j P_j$, $h_j \doteq e(P_j, c_j P_j)$. Also, for $j = 1, \ldots, v$, compute linearly independent vectors $(\alpha_{j,0}, \beta_{j,0})$, $(\alpha_{j,1}, \beta_{j,1}) \in \mathbb{Z}_q^2$ such that $b_j \alpha_{j,\sigma} + a_j \beta_{j,\sigma} = c_j \bmod q$, for $\sigma \in \{0, 1\}$.

Step 2b: Generate one more independent copy of the 2-user scheme of Section 4.1, in which we additionally select v values for h (call this the *shared* scheme). At a high level, the shared scheme can be thought of as v parallel copies of the 2-user scheme of Section 4.1, sharing the same values P, Q and R. More precisely, draw $P \stackrel{R}{\leftarrow} \mathcal{G}_1$, $a, b \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, and set $Q \doteq aP$, and $R \doteq bP$; then, for each $j = 1, \ldots, v$, select $\bar{c}_j \in \mathbb{Z}_q^*$ and set $\bar{h}_j \doteq e(P, \bar{c}_j P)$. Also, for each $j = 1, \ldots, v$, compute two linearly independent vectors $(\bar{\alpha}_{j,0}, \bar{\beta}_{j,0})$, $(\bar{\alpha}_{j,1}, \bar{\beta}_{j,1})$ in $\mathbb{Z}_q^2$ such that $b\bar{\alpha}_{j,\sigma} + a\bar{\beta}_{j,\sigma} = \bar{c}_j \bmod q$, for $\sigma \in \{0, 1\}$.

Step 2c: The master secret key msk of the security manager is set to be:

$$((a_j, b_j, (\alpha_{j,0}, \beta_{j,0}, \alpha_{j,1}, \beta_{j,1}))_{j=1,\ldots,v}, a, b, (\bar{\alpha}_{j,0}, \bar{\beta}_{j,0}, \bar{\alpha}_{j,1}, \bar{\beta}_{j,1})_{j=1,\ldots,v})$$

Step 3: For $j = 1, \ldots, v$ and $\sigma \in \{0, 1\}$, let $A_{j,\sigma} \doteq \alpha_{j,\sigma} R_j$, $B_{j,\sigma} \doteq \beta_{j,\sigma} P_j$, $\bar{A}_{j,\sigma} \doteq \bar{\alpha}_{j,\sigma} R$ and $\bar{B}_{j,\sigma} \doteq \bar{\beta}_{j,\sigma} P$. Choose a universal hash function $H : \mathcal{G}_2 \to \{0,1\}^\kappa$, and set pk to:[6]

$$(H, (P_j, Q_j, R_j, A_{j,0}, B_{j,0}, A_{j,1}, B_{j,1}), P, Q, R, (\bar{A}_{j,0}, \bar{B}_{j,0}, \bar{B}_{j,1}))$$

for all $j = 1, \ldots, v$. The associated message space is $\mathcal{M} \doteq (\{0,1\}^\kappa)^v$.

Reg: For each user i, the security manager first retrieves the corresponding codeword $\omega_i \in \mathcal{C}$ and sets his/her secret key to: $\mathsf{sk}_i \doteq ((\alpha_{j,\omega_j^{(i)}})_{j=1,\ldots,v}, (\bar{\alpha}_{j,\omega_j^{(i)}})_{j=1,\ldots,v})$. Notice that, for $j = 1, \ldots, v$, it holds that:

$$c_j P_j = \alpha_{j,\omega_j^{(i)}} R_j + \beta_{j,\omega_j^{(i)}} Q_j \quad \text{and hence} \quad h_j = e(P_j, A_{j,\omega_j^{(i)}}) \cdot e(Q_j, B_{j,\omega_j^{(i)}}),$$
$$\bar{c}_j P = \bar{\alpha}_{j,\omega_j^{(i)}} R + \bar{\beta}_{j,\omega_j^{(i)}} Q \quad \text{and hence} \quad \bar{h}_j = e(P, \bar{A}_{j,\omega_j^{(i)}}) \cdot e(Q, \bar{B}_{j,\omega_j^{(i)}}).$$

Enc: Given pk, anybody can encrypt a message $m = (m^{(1)} \| \ldots \| m^{(v)}) \in \mathcal{M}$ as follows:

First, select $\ell \stackrel{R}{\leftarrow} \{1, \ldots, v\}$ and $k_\ell \stackrel{R}{\leftarrow} \mathbb{Z}_q$, and compute the special component of the ciphertext $(U_\ell, V_\ell, W_\ell) \in \mathcal{G}_2 \times \mathcal{G}_1 \times \{0,1\}^\kappa$, where $U_\ell \doteq e(P_\ell, R_\ell)^{k_\ell}$, $V \doteq k_\ell Q_\ell$ and $W_\ell \doteq m^{(\ell)} \oplus H(h_\ell^{k_\ell})$.

Then, select $k \stackrel{R}{\leftarrow} \mathbb{Z}_q$, and compute the remaining pieces of the ciphertext as: $(U, V, W_1, \ldots, W_{\ell-1}, W_{\ell+1}, \ldots, W_v)$, where $U \doteq e(P, R)^k$, $V \doteq kQ$, and $W_j \doteq m^{(j)} \oplus H(\bar{h}_j^k)$, for $j = 1, \ldots, v$, $j \neq \ell$. The ciphertext is set to be the tuple $\psi \doteq \langle \ell, U_\ell, V_\ell, U, V, W_1, \ldots, W_v \rangle$.

[6] The shared scheme is not used for tracing, so $\bar{A}_{j,1}$ can be safely omitted ($\bar{A}_{j,0}$ is included only so that $\bar{h}_j$ can be computed.)

Dec: Given a ciphertext $\psi = \langle \ell, U_\ell, V_\ell, U, V, W_1, \ldots, W_v \rangle \in \mathbb{Z} \times (\mathcal{G}_2 \times \mathcal{G}_1)^2 \times \mathcal{M}$, u_i computes for each $j = 1, \ldots, v, j \neq \ell$:

$$h_\ell^{k_\ell} = (U_\ell)^{\alpha_{\ell,\omega_\ell^{(i)}}} \cdot e(V_\ell, B_{\ell,\omega_\ell^{(i)}}) \quad \text{and} \quad \bar{h}_j^k = (U)^{\bar{\alpha}_{j,\omega_j^{(i)}}} \cdot e(V, \bar{B}_{j,\omega_j^{(i)}})$$

recovers $m^{(\ell)} = W_\ell \oplus H(h_\ell^{k_\ell})$ and $m^{(j)} = W_j \oplus H(\bar{h}_j^k)$ (for $j \in \{1, \ldots, v\} \setminus \{\ell\}$) and outputs $m \doteq (m^{(1)} \| \ldots \| m^{(v)})$.

Trace: Given pk, anybody can extract the "traitor codeword" $\hat{\omega} \doteq (\hat{\omega}^{(1)}, \ldots, \hat{\omega}^{(v)}) \in \{0,1\}^v$ from a decoder $\mathcal{D}$ by making $O(v)$ queries to $\mathcal{D}$. At a high level, the idea is to iteratively derive each $\hat{\omega}^{(\ell)}$ by feeding $\mathcal{D}$ with an invalid ciphertext that looks valid in the "shared" components, but is actually a *probe* (in the sense of Section 4.3) on the ℓ-th "special" component. In this way, if $\mathcal{D}$ contains only one of the two user-keys for the ℓ-th "special" two-user component (say, $\alpha_{\ell,\tau^{(\ell)}}$), its reply will reveal the value of $\tau^{(\ell)}$. More in detail, to extract $\tau^{(\ell)}$ from $\mathcal{D}$, **Trace** queries $\mathcal{D}$ with ciphertexts of the form $\hat{\psi}^{(\ell)} \doteq \langle \ell, \hat{U}_\ell, \hat{V}_\ell, U^{(\ell)}, V^{(\ell)}, W_1^{(\ell)}, \ldots, \hat{W}_\ell^{(\ell)}, \ldots, W_v^{(\ell)} \rangle$, where $k_\ell, k'_\ell, k^{(\ell)} \xleftarrow{R} \mathbb{Z}_q$, $\hat{m}^{(\ell)} = \hat{m}_1^{(\ell)}, \ldots, \hat{m}_v^{(\ell)}$ is drawn at random from $\mathcal{M}$, $\sigma^{(\ell)}$ is a random bit, $W_j^{(\ell)} \doteq \hat{m}_j^{(\ell)} \oplus H(h_j^{k^{(\ell)}})$ for each $j = 1, \ldots, v, j \neq \ell$, and

$$\hat{U}_\ell \doteq e(P_\ell, R_\ell)^{k'_\ell} \qquad \hat{V}_\ell \doteq k_\ell Q_\ell \qquad U^{(\ell)} \doteq e(P, R)^{k^{(\ell)}} \qquad V^{(\ell)} \doteq k^{(\ell)} Q$$
$$\hat{W}_\ell^{(\ell)} \doteq \hat{m}_\ell^{(\ell)} \oplus H(e(P_\ell, A_{\ell,\tau^{(\ell)}})^{k'_\ell} \cdot e(\hat{V}_\ell, B_{\ell,\tau^{(\ell)}})).$$

Let $m^{*(\ell)} \doteq (m_1^{*(\ell)} \| \ldots \| m_v^{*(\ell)})$ be the plaintext output by $\mathcal{D}$ when fed with the ciphertext $\hat{\psi}^{(\ell)}$. If $\mathcal{R}(\hat{m}^{(\ell)}, m^{*(\ell)}) = 1$, then set $\hat{\omega}^{(\ell)} = \sigma^{(\ell)}$; otherwise, pick fresh random $k_\ell, k'_\ell, k^{(\ell)}$ from $\mathbb{Z}_q$, $\hat{m}^{(\ell)}$ from $\mathcal{M}$, $\sigma^{(\ell)}$ from $\{0,1\}$, and repeat, until either $\mathcal{R}(\hat{m}^{(\ell)}, m^{*(\ell)}) = 1$, or the iteration has failed some fixed polynomial number of time, in which case $\hat{\omega}^{(\ell)}$ is set arbitrarily.

After this process has been repeated for $\ell = 1, \ldots, v$, the resulting "traitor codeword" $\hat{\omega}$ is handed to the tracer, who (knowing the random coins $r_{\mathcal{C}}$ used in generating $\mathcal{C}$) can run it through the tracing algorithm $\mathcal{T}(r_{\mathcal{C}}, \cdot)$ of the collusion-secure code $\mathcal{C}$, thus obtaining a value in $\{1, \ldots, n, 0\}$, which is the output of **Trace**.

Remark 15. Since the **Trace** algorithm needs msk only in the off-line phase, which does not access the pirate decoder and is much less computation-intensive,[7] our multi-user scheme supports local public traceability.

Remark 16. We bound the number of trials that **Trace** performs to extract each bit $\hat{\omega}^{(\ell)}$ because a pirate decoder holding both keys for position ℓ could cause the test $\mathcal{R}(\hat{m}^{(\ell)}, m^{*(\ell)}) = 1$ to fail with probability 1. A suitable value for this bound is $O(1/p_{\mathcal{D}})$, where $p_{\mathcal{D}}$ is the success probability (over random valid ciphertexts) of the decoder under tracing, which can be efficiently estimated using Chernoff bounds.

Remark 17. Notice that the size of the message blocks can be shrunk to any $\kappa' \leq \kappa$, by choosing a universal hash function $H : \mathcal{G}_2 \to \{0,1\}^{\kappa'}$. This is possible as long as $\kappa' > \log v + \log(1/\varepsilon) = O(\log t + \log\log(n/\varepsilon) + \log(1/\varepsilon))$, which ensures that, during tracing, the probability of a hash collision in any of the v components of the scheme is bounded by ε. For a typical choice of parameters ($n = 2^{30}$, $\varepsilon = 2^{-30}$, $t = 30$), κ' can be chosen as low as 64 bits.

[7] For the scheme of [Tar03], for example, such computation consists just of a matrix-vector multiplication.

Pre-Processing Messages with AONT's. An AONT is an efficient, unkeyed, randomized transformation, with the property that it is hard to invert unless the *entire* output is known. (For a formal definition, see [Boy99,CDH$^+$00].) As for specific instantiations, Boyko showed in [Boy99] that the *Optimal Asymmetric Encryption Padding* (OAEP)[BR94] can be proven secure as an AONT in the Random Oracle Model. In [CDH$^+$00], Canetti *et al.* described constructions in the standard model based on the notion of *Exposure-Resilient Functions*.

For our purposes, it suffices to think of an AONT as a length-preserving algorithm $\mathrm{AONT}(m; r)$, where $m \in (\{0,1\}^\kappa)^{v-1}$ is the message to be processed and r is an additional random value, of the same length as each message block *i.e.*, $|r| = \kappa$. In what follows, we denote by $M \stackrel{R}{\leftarrow} \mathrm{AONT}(m)$ the process of selecting a random r from $\{0,1\}^\kappa$ and setting $M \leftarrow \mathrm{AONT}(m; r)$. The resulting AONT-transformed message $M = (M_1, \ldots, M_v)$ is an element of $(\{0,1\}^\kappa)^v$, so that it can be encrypted with the **Enc** algorithm described above. We can thus define a multi-user scheme with AONT pre-processing by modifying the **Enc** and **Dec** algorithms as:

$$\mathbf{Enc}'(m) \doteq \mathbf{Enc}(\mathrm{AONT}(m)) \qquad \mathbf{Dec}'(\psi) \doteq \mathrm{AONT}^{-1}(\mathbf{Dec}(\psi))$$

Notice that the use of AONT pre-processing in the full-blown scheme implies an expansion in the message size by roughly a factor $1 + 1/v$, which still results in an asymptotical unitary ciphertext-to-plaintext ratio.

4.5 Indistinguishability under Chosen-Plaintext Attack

In this section, we assess the security of the multi-user scheme of Section 4.4. (For lack of space, we defer all proofs for this section to Appendix D.)

We start by verifying the intuition that AONT pre-processing does not hurt security:

Lemma 18. *If the multi-user scheme without AONT pre-processing is secure w.r.t. indistinguishability under chosen-plaintext attack, then the multi-user scheme with AONT pre-processing is secure w.r.t. the same notion.*

Next, we observe that the security of the multi-user scheme from Section 4.4 can be reduced (via a hybrid argument) to the security of the two-user scheme from Section 4.1:

Lemma 19. *If our two-user scheme is secure w.r.t. indistinguishability under chosen-plaintext attack, then our multi-user scheme without AONT pre-processing is secure w.r.t. the same notion.*

In light of Theorem 9, our main security theorem follows immediately from Lemmata 18 and 19:

Theorem 20. *Under the* DBDH *assumption for* $(\mathcal{G}_1, \mathcal{G}_2)$*, the scheme in Section 4.4 is secure w.r.t. indistinguishability under chosen-plaintext attack.*

4.6 Traceability

Similarly to the case of the 2-user scheme of Section 4.1, the traceability of our multi-user scheme (with AONT pre-processing) is based on the notions of *valid* and *probe* ciphertexts:

Definition 21. Let $\ell \in [1, v]$, $\sigma \in \{0,1\}$, $\hat{m} \in \mathcal{M}$, $\hat{M} = (\hat{M}_1, \ldots, \hat{M}_v) \stackrel{R}{\leftarrow} \mathrm{AONT}(\hat{m})$, $\hat{U}_\ell \in \mathcal{G}_2$, $\hat{V}_\ell \in \mathcal{G}_1$, $k \in \mathbb{Z}_q$, $U = e(P, R)^k$, $V = kQ$, $W_j = \hat{M}_j \oplus H(h_j^k)$ $(j = 1, \ldots, v,\ j \neq \ell)$, $\hat{W}_\ell = \hat{M}_\ell \oplus H(\hat{U}_\ell^{\alpha_{\ell,\sigma}} e(\hat{V}_\ell, B_{\ell,\sigma}))$, and $\hat{\psi} = \langle \ell, \hat{U}_\ell, \hat{V}_\ell, U, V, W_1, \ldots, \hat{W}_\ell, \ldots, W_v \rangle$. We say that the ciphertext $\hat{\psi}$ is:

- **valid**, if $\hat{U}_\ell = e(P_\ell, R_\ell)^{k_\ell}$, $\hat{V}_\ell = k_\ell Q_\ell$, for some $k_\ell \in \mathbb{Z}_q$;
- (ℓ, σ)-**probe**, if $\hat{U}_\ell = e(P_\ell, R_\ell)^{k'_\ell}$, $\hat{V}_\ell = k_\ell Q_\ell$, for distinct $k_\ell, k'_\ell \in \mathbb{Z}_q$.

Our analysis is organized as follows. Let T denote the set of indices of the t traitors. Lemma 22 proves the computational indistinguishability of valid ciphertexts *vs.* (ℓ, τ^ℓ)-probes when only the τ^ℓ subkey is known for position ℓ. It follows (Corollary 23) that pirate decoders must decrypt such (ℓ, τ^ℓ)-probes correctly (w.r.t. the chosen resemblance relation). Lemma 24 then shows that instead $(\ell, 1-\tau^\ell)$-probes cannot be properly decrypted, and Lemma 25 combines Corollary 23 and Lemma 24 to argue that the chances that the ℓ-th stage of tracing fails to extract the correct bit $\hat{\omega}^{(\ell)} = \tau^\ell$ from $\mathcal{D}$ are negligible, which implies the overall traceability of our scheme (Theorem 26).

Lemma 22 (Indistinguishability of Valid *vs.* Probe Ciphertexts). *Under the* DBDH *assumption for* $(\mathcal{G}_1, \mathcal{G}_2)$*, given the public key* $\mathsf{pk} = (q,\, \mathcal{G}_1,\, \mathcal{G}_2,\, e,\, H,\, P_j, Q_j, R_j,\, (A_{j,0},\, B_{j,0},\, A_{j,1},\, B_{j,1})_{j=1,\ldots,v},\, P, Q, R,\, (\bar{A}_{j,0},\, \bar{B}_{j,0},\, \bar{B}_{j,1})_{j=1,\ldots,v})$ *and the secret keys* $\mathsf{sk}_i \doteq ((\alpha_{j,\omega_j^{(i)}})_{j=1,\ldots,v},\, (\bar{\alpha}_{j,\omega_j^{(i)}})_{j=1,\ldots,v})$ *for each* $i \in T$*, it is infeasible to distinguish valid ciphertexts from* (ℓ, τ^ℓ)*-probes, if the codewords of all traitors in* T *have bit* τ^ℓ *at position* ℓ*.*

Proof. Since the ℓ-th "special" sub-schemes is completely independent from the rest of our construction, the thesis follows as a simple reduction to Lemma 11. □

Corollary 23. *Let* $\mathcal{D}$*,* $\mathcal{R}$ *be the pirate decoder and resemblance relation output by a traitor strategy* $\mathcal{A}$ *based on the user keys of the traitors in* T*, such that* $p_{\mathcal{D}}$ *is non-negligible and* $p_{\mathcal{R}}$ *is negligible (*cf. *Definition 8). Assume the codewords of all the traitors in* T *have bit* τ^ℓ *at position* ℓ*, and let* $\hat{\psi}$ *be an* (ℓ, τ^ℓ)*-probe for a message* $\hat{m} \stackrel{R}{\leftarrow} \mathcal{M}$*. Under the* DBDH *assumption,* $\hat{p}_{\mathcal{D}} \doteq \Pr[\mathcal{R}(\hat{m}, m^*) = 1 \mid m^* \stackrel{R}{\leftarrow} D(\hat{\psi})]$ *is non-negligible.*

Proof. Reduces to Lemma 22 exactly as Corollary 12 reduces to Lemma 11.

Lemma 24. *Replacing* $\hat{\psi}$ *with an* $(\ell, 1-\tau^\ell)$*-probe in the setting of Corollary 23,* $\Pr[\mathcal{R}(\hat{m}, m^*) = 1]$ *is negligible, if the AONT employed in the system is secure.*

Proof. The argument described in the proof of Lemma 13 implies that the AONT-transformed message block $\hat{M}_\ell$ is computationally hidden from the pirate decoder's view. By the properties of AONT's, the whole original message $\hat{m}$ is then also computationally hidden from $\mathcal{D}$, so that in fact $\hat{m}$ is just a random message independent from the output m^* of $\mathcal{D}$, and hence $\mathcal{R}(\hat{m}, m^*) = 1$ holds with probability $p_{\mathcal{R}}$, which is negligible. □

Lemma 25. *Consider the* ℓ*-th stage of the* **Trace** *algorithm, when the tracer queries the decoder* $\mathcal{D}$ *with* (ℓ, σ)*-probes for random* $\sigma \in \{0, 1\}$*. If all codewords of the traitors in* T *have bit* τ^ℓ *in the* ℓ*-th position, then the* ℓ*-th stage will terminate setting* $\hat{\omega}^\ell = 1 - \tau^\ell$ *with negligible probability.*

Proof. The assumption that $\mathcal{D}$ does not contain both keys for position ℓ implies, by Corollary 23, that the ℓ-th stage of **Trace** will on average terminate after $2/p_{\mathcal{D}}$ queries to $\mathcal{D}$. Upon termination, **Trace**'s output will be wrong only if it happens that $\mathcal{D}$ replies to an $(\ell, 1-\tau^\ell)$-probe $\hat{\psi}$ with an m^* satisfying $\mathcal{R}(\hat{m}, m^*) = 1$, which by Corollary 23, Lemma 24, and Bayes' theorem is easily seen to equal $p_{\mathcal{R}}/(p_{\mathcal{D}} + p_{\mathcal{R}})$, which is negligible. □

Theorem 26. *Under the* DBDH *assumption for* $(\mathcal{G}_1, \mathcal{G}_2)$*, the multi-user* **Trace** *algorithm from Section 4.4 has a negligible traceability error.*

Proof. Let $\hat{\omega} = (\hat{\omega}^{(1)}, \ldots, \hat{\omega}^{(v)})$ be the "traitor codeword" recovered at the end of the publicly traceable phase of **Trace** (*cf.* Section 4.4). By the union bound, Lemma 25 implies that $\hat{\omega}$ will be correct in all positions ℓ where all traitors show the same bit, except with negligible probability. By the collusion resistance of the code $\mathcal{C}$ underlying the key assignment of $\mathbf{Setup}$, the codeword-tracing algorithm $\mathcal{T}$ (*cf.* Definition 4) will then be able to tie such traitor codeword $\hat{\omega}$ to the identity of one of the traitors in T (except with negligible probability ε), as required. □

Remark 27. As noted above, by employing AONT's, the security and tracing capabilities of our multi-user scheme follow almost directly from those of the embedded "special" sub-scheme. In fact, even if we were to suppress the shared sub-scheme (*e.g.*, by setting $W_j = M_j$, for $j = 1, \ldots, v, j \neq \ell$), the multi-user scheme would still be secure and tracing would still be possible (thanks also to the random rotation of the special position ℓ between 1 and v). Using the shared sub-scheme, however, reinforces the semantic security of the scheme, though at the cost of a greater computational load, due to the larger number of pairing computations needed for encryption and decryption.

5 Space and Time Parameters in a Concrete Instantiation

Existing constructions of constant-rate traitor tracing schemes (including ours) are based on the use of collusion-secure fingerprint codes[8] [BS98,Tar03], and in particular are applicable for messages of size proportional to the length of the code, which in the case of the optimal codes due to Tardos [Tar03] is $O(t^2(\log n + \log \frac{1}{\varepsilon}))$. For a typical choice of parameters, *e.g.* user population $n = 2^{30}$, tracing error probability $\varepsilon = 2^{-30}$ and traceable threshold $t = 30$, the resulting code length is about 5 million bits. Instantiating our construction with such codes yields a scheme with plaintext and ciphertext of size 41 MBytes. (The ciphertext size is equal to the plaintext size, as the additive overhead is less than 1 KByte.) These values are well within the range of multimedia applications, since 41 MBytes roughly corresponds to 33 seconds of DVD-quality (high-resolution) video, 4 minutes of VCD-quality (low-resolution) video and 25–50 minutes of audio. The resulting public and secret keys roughly require respectively 1.5GByte and 206 MBytes. Although quite large, such a public key could be stored in commodity hardware (*e.g.,* it would fit in the hard disk of an iPod), whereas user secret keys could be kept in Secure Digital memory cards, like those commonly available for PDAs.

Another important issue for a concrete instantiation is the rate at which encrypted content can be processed. In our scheme, decryption requires one paring per 1024 bits of content, which, using the PBC Library [Lyn] on a desktop PC, takes approximately 16 msec. However, in our context, the pairings to be computed all have one of their two input-points in common: as reported in [BBS04], pre-processing in similar settings more than halves the computation time, so that one easily gets in the order of 128 pairings/sec, corresponding to a near-CD-quality audio rate of 128 Kbits/sec. More specialized software implementations [BGhCS04] of the pairing operation can further reduce its computational cost to around 3 msec; whereas hardware implementations, even under conservative assumptions on the hardware architecture [KMPB05], can obtain running time below 1 msec, attaining the 1Mbits/sec data rate needed for VCD-quality video.

6 Conclusion

We present the first public-key traitor tracing scheme with efficient black-box tracing and optimal transmission rate. Our treatment improves the standard modeling of black-box tracing by additionally accounting for pirate strategies that attempt to escape tracing by purposedly rendering the transmitted content at lower quality (*e.g.* by dropping every other frame from the decrypted video-clip, or skipping few seconds from the original audio file). We also point out and resolve an issue in the black-box traitor tracing mechanism of both the previous schemes in this setting [KY02b,CPP05]. Our construction is based on the decisional bilinear Diffie-Hellman assumption, and additionally provides the same features of public traceability as (a repaired version of) [CPP05], which is less efficient and requires non-standard assumptions for bilinear groups.

[8] [PST06] actually employs IPP codes, but similar considerations on code length and message size apply to such codes as well.

References

BBS04. S.L.M. Barreto, L. Ben, and M. Scott, *Efficient Implementation of Paring-Based Cryptosystems*, Journal of Cryptology **17** (2004), no. 4, 321–334.

BF99. D. Boneh and M. Franklin, *An Efficient Public Key Traitor Tracing Scheme*, Advances in Cryptology—Crypto '99 (Heidelberg), Springer, 1999, LNCS 1666. Full version available at `crypto.stanford.edu/~dabo/pubs.html`, pp. 338–353.

BGhCS04. P. Barreto, S. Galbraith, hEigeartaigh C., and M. Scott, *Efficient Pairing Computation on Supersingular Abelian Varieties*, Available at `http://eprint.iacr.org/2004/375`, 2004.

Boy99. V. Boyko, *On the Security Proprties of the OAEP as an All-or-Nothing Transform*, Advances in Cryptology—Crypto '99 (Heidelberg), Springer, 1999, LNCS 1666, pp. 503–518.

BPS00. O. Berkman, M. Parnas, and J. Sgall, *Efficient Dynamic Traitor Tracing*, Proceedings of the 11th Symposium on Discrete Algorithms, 2000, pp. 586–595.

BR94. M. Bellare and P. Rogaway, *Optimal Asymmetric Encryption–How to Encrypt with RSA*, Advances in Cryptology—EuroCrypt '94 (Heidelberg), Springer, 1994, LNCS 950, pp. 92–111.

BS98. D. Boneh and J. Shaw, *Collusion-Secure Fingerprinting for Digital Data*, IEEE Transactions on Information Theory **44** (1998), no. 5, 1897–1905.

BSW06. D. Boneh, A. Sahai, and B. Waters, *Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys*, Advances in Cryptology—Proceedings of EUROCRYPT '06 (Heidelberg), Springer, 2006, LNCS 4004, pp. 573–592.

BW06. D. Boneh and B. Waters, *A Collusion Resistant Broadcast, Trace and Revoke System*, Computer and Communication Security—CCS'06 (New York), ACM Press, 2006, pp. 211–220.

CDH$^+$00. R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai, *Exposure-Resilient Functions and All-or-Nothing Transform*, Advances in Cryptology—EuroCrypt '00 (Heidelberg), Springer, 2000, LNCS 1807, pp. 453–469.

CFN94. B. Chor, A. Fiat, and N. Naor, *Tracing Traitors*, Advances in Cryptology—Crypto '94 (Heidelberg), Springer, 1994, LNCS 839, pp. 257–270.

CFNP00. B. Chor, A. Fiat, N. Naor, and B. Pinkas, *Tracing Traitors*, IEEE Transaction on Information Theory **46** (2000), no. 3, 893–910.

CPP05. H. Chabanne, D.H. Phan, and D. Poitcheval, *Public Traceability in Traitor Tracing Schemes*, Advances in Cryptology—EuroCrypt '05 (Heidelberg), Springer, 2005, LNCS 3494, pp. 542–558.

DF02. Y. Dodis and N. Fazio, *Public-Key Broadcast Encryption for Statless Receivers*, Digital Rights Management—DRM '02 (Heidelberg), Springer, 2002, LNCS 2696, pp. 61–80.

DF03. ______, *Public-Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack*, Public Key Cryptography—PKC '03 (Heidelberg), Springer, 2003, LNCS 2567, pp. 100–115.

DFKY05. Y. Dodis, N. Fazio, A. Kiayias, and M. Yung, *Scalable Public-Key Tracing and Revoking*, Journal of Distributed Computing **17** (2005), no. 4, 323–347.

FT01. A. Fiat and T. Tassa, *Dynamic Traitor Tracing*, Journal of Cryptology **14** (2001), no. 3, 211–223.

GSY99. E. Gafni, J. Staddon, and Y. L. Yin, *Efficient Methods for Integrating Traceability and Broadcast Encryption*, Advances in Cryptology—Crypto '99 (Heidelberg), Springer, 1999, LNCS 1666, pp. 372–387.

KD98. K. Kurosawa and Y. Desmedt, *Optimum Traitor Tracing and new Direction for Asymmetricity*, Advances in Cryptology—EuroCrypt '98 (Heidelberg), Springer, 1998, LNCS 1403, pp. 145–157.

KHL03. C.H. Kim, Y.H. Hwang, and P.J. Lee, *An Efficient Public Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack*, Advances in Cryptology - Asiacrypt 2003 (Heidelberg), Springer, 2003, LNCS 2894, pp. 359–373.

KMPB05. T. Kerins, W.P. Marnane, E.M. Popovici, and P.S.L.M. Barreto, *Efficient Hardware for the Tate Paring Calculation in Characteristic Three*, Cryptography Hardware and Embedded Systems—CHES '05 (Heidelberg), Springer, 2005, LNCS 3659, pp. 412–426.

KY01. A. Kiayias and M. Yung, *Self Protecting Pirates and Black-Box Traitor Tracing*, Advances in Cryptology—Crypto '01 (Heidelberg), Springer, 2001, LNCS 2139, pp. 63–79.

KY02a. ______, *Breaking and Repairing Asymmetric Public-Key Traitor Tracing*, Digital Rights Management—DRM '02 (Heidelberg), Springer, 2002, LNCS 2696, pp. 32–50.

KY02b. ______, *Traitor Tracing with Constant Transmission Rate*, Advances in Cryptology—EuroCrypt '02 (Heidelberg), Springer, 2002, LNCS 2332, pp. 450–465.

KY02c. K. Kurosawa and T. Yoshida, *Linear Code Implies Public-Key Traitor Tracing*, Public Key Cryptography—PKC '02 (Heidelberg), Springer, 2002, LNCS 2274, pp. 172–187.

KY06. A. Kiayias and M. Yung, *Copyrighting Public-key Functions and Applications to Black-box Traitor Tracing*, Full revised version of [KY02b]. Available at: `http://eprint.iacr.org/2006/458/`, 2006.

Lyn. B. Lynn, *PBC Library*, Available at `http://crypto.stanford.edu/pbc/`.

NNL01. D. Naor, M. Naor, and J. Lotspiech, *Revocation and Tracing Schemes for Stateless Receivers*, Advances in Cryptology—Crypto '01 (Heidelberg), Springer, 2001, LNCS 2139, pp. 41–62.

NP98. M. Naor and B. Pinkas, *Threshold Traitor Tracing*, Advances in Cryptology—Crypto '98 (Heidelberg), Springer, 1998, LNCS 1462, pp. 502–517.

NP00. ______, *Efficient Trace and Revoke Schemes*, Financial Cryptography—FC 2000 (Heidelberg), Springer, 2000, LNCS 1962. Full version available at `www.wisdom.weizmann.ac.il/~naor/onpub.html`, pp. 1–20.

Pfi96. B. Pfitzmann, *Trials of Traced Traitors*, Information Hiding (Heidelberg), Springer, 1996, LNCS 1174, pp. 49–64.

PST06. D.H. Phan, R. Safavi-Naini, and D. Tonien, *Generic construction of hybrid public key traitor tracing with full-public-traceability*, International Colloquim on Automata, Languages, and Programming—ICALP'06, 2006, pp. 264–275.

Riv97. R. Rivest, *All-or-Nothing Encryption and the Package Transform*, Fast Softaware Encryption, 1997.

Sho04. V. Shoup, *Sequences of Games: A Tool for Taming Complexity in Security Proofs*, Manuscript. Available at `shoup.net/papers/games.pdf`, 2004.

SW98. D. R. Stinson and R. Wei, *Combinatorial Properties and Constructions of Traceability Schemes and Frameproof Codes*, SIAM Journal on Discrete Mathematics **11** (1998), no. 1, 41–53.

SW03. R. Safavi-Naini and Y. Wang, *Sequential Traitor Tracing*, IEEE Transactions on Information Theory **49** (2003), no. 5, 1319–1326.

Tar03. G. Tardos, *Optimal Probabilistic Fingerprint Codes*, Proceedings of the 35th Symposium on Theory of Computing—STOC'03 (New York), ACM Press, 2003, pp. 116–125.

TT01. W.G. Tzeng and Z.J. Tzeng, *A Public-Key Traitor Tracing Scheme with Revocation Using Dynamics Shares*, Public Key Cryptography—PKC '01 (Heidelberg), Springer, 2001, LNCS 1992, pp. 207–224.

A Bilinear Maps and Intractability Assumptions

A.1 Bilinear Maps

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two groups of order q, for some large prime q. In our construction, we will make use of a bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$, satisfying the following properties:

Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathcal{G}_1$ and all $a, b \in \mathbb{Z}_q$;
Non-degeneracy: The map does not send all pairs in $\mathcal{G}_1 \times \mathcal{G}_1$ to the unit in $\mathcal{G}_2$;
Computable: There is an efficient algorithm to compute $e(P, Q)$ for any elements $P, Q \in \mathcal{G}_1$.

A bilinear map satisfying the three above properties is said to be an *admissible bilinear map*. Throughout the paper, we view $\mathcal{G}_1$ as an additive group and $\mathcal{G}_2$ as a multiplicative group. We remark that since $\mathcal{G}_1, \mathcal{G}_2$ are groups of prime order and e is non-degenerated, $e(P, P)$ generates $\mathcal{G}_2$ whenever P generates $\mathcal{G}_1$. It follows that $e(P, \cdot)$ is an isomorphism from $\mathcal{G}_1$ into $\mathcal{G}_2$. Typical examples of constructions of admissible bilinear maps satisfying the above properties are based on the modified Weil and Tate pairings (*cf. e.g.,* [BF99]).

A.2 Assumptions for Our Scheme

DBDH (the decisional bilinear Diffie-Hellman problem in $(\mathcal{G}_1, \mathcal{G}_2)$):
Given (P, aP, bP, cP, h) for random $P \in \mathcal{G}_1$, $a, b, c \in \mathbb{Z}_q$ and $h \in \mathcal{G}_2$, output yes if $h = e(P, P)^{abc}$ and no otherwise.

Definition 28 (DBDH **Assumption).** The DBDH problem is $\varepsilon_{\mathsf{DBDH}}$-hard in $(\mathcal{G}_1, \mathcal{G}_2)$ if, for all probabilistic polynomial-time algorithms $\mathcal{A}$, we have

$$|\Pr[\mathcal{A}(P, aP, bP, cP, h) = yes \mid P \xleftarrow{R} \mathcal{G}_1, a, b, c \xleftarrow{R} \mathbb{Z}_q, h = e(P, P)^{abc}] -$$
$$\Pr[\mathcal{A}(P, aP, bP, cP, h) = yes \mid P \xleftarrow{R} \mathcal{G}_1, a, b, c \xleftarrow{R} \mathbb{Z}_q, h \xleftarrow{R} \mathcal{G}_2]| < \varepsilon_{\mathsf{DBDH}}$$

where the probability is over the random selection of P from $\mathcal{G}_1$, of a, b, c from $\mathbb{Z}_q$, and over $\mathcal{A}$'s random coins.

A.3 Assumption for the Schemes of [KY02b]

DDH (the decisional Diffie-Hellman problem in $\mathcal{G}$):
Given (P, aP, bP, S) for random $P \in \mathcal{G}$, $a, b \in \mathbb{Z}_q$ and $S \in \mathcal{G}$, output yes if $S = abP$ and no otherwise.

A.4 Assumptions for the Schemes of [CPP05]

DBDH^2-E (the extended decisional bilinear Diffie-Hellman problem):
Given $(P, aP, bP, cP, ab^2P, h)$ for random $P \in \mathcal{G}_1$, $a, b, c \in \mathbb{Z}_q$ and $h \in \mathcal{G}_2$, output yes if $h = e(P, P)^{cb^2}$ and no otherwise.

DBDH^1-M (the modified decisional bilinear Diffie-Hellman problem in $\mathcal{G}_1$):
Given(P, aP, bP, S) for random $P \in \mathcal{G}_1$, $a, b \in \mathbb{Z}_q$ and $S \in \mathcal{G}_1$, output yes if $S = ab^2P$ and no otherwise.

B The Public-Key Traitor Tracing Schemes of [KY02b] and [CPP05]

B.1 The Two-User Sub-Scheme of [KY02b]

Setup: Given a security parameter 1^κ, the algorithm works as follows:

Step 1: Generate a κ-bit prime q and a group $\mathcal{G}$ of order q in which the DDH problem is difficult. Let P be a generator of $\mathcal{G}$.[9]

Step 2: Pick random elements $a, c \in \mathbb{Z}_q^*$, and set $Q \doteq aP$, $Z \doteq cP$. The private key of the security manager is set to be the pair $\mathsf{msk} \doteq (a, c)$.

Step 3: Choose a universal hash function $H : \mathcal{G} \to \{0,1\}^\kappa$, and set the public key as $\mathsf{pk} \doteq (q, \mathcal{G}, H, P, Q, Z)$. The message space is $\mathcal{M} \doteq \{0,1\}^\kappa$.

Reg: The security manager selects two linearly independent vectors (α_0, β_0), $(\alpha_1, \beta_1) \in \mathbb{Z}_q^2$ such that $\alpha_\sigma + a\beta_\sigma = c \bmod q$, for $\sigma \in \{0, 1\}$. This implies: $Z = cP = \alpha_\sigma P + \beta_\sigma Q$, for $\sigma \in \{0, 1\}$. The secret key of user σ is then set to be $\mathsf{sk}_\sigma \doteq (\alpha_\sigma, \beta_\sigma)$, for $\sigma \in \{0, 1\}$.

Enc: Given pk, anybody can encrypt a message $m \in \mathcal{M}$ by first selecting a random $k \in \mathbb{Z}_q$ and then creating the ciphertext $\psi \doteq \langle U, V, W \rangle \in \mathcal{G}^2 \times \mathcal{M}$ where

$$U \doteq kP, \quad V \doteq kQ, \quad W \doteq m \oplus H(kZ)$$

Dec: Given a ciphertext $\psi = \langle U, V, W \rangle \in \mathcal{G}^2 \times \mathcal{M}$, user σ computes $kZ = \alpha_\sigma U + \beta_\sigma V$ and recovers $m = W \oplus H(kZ)$.

Trace: To trace a decoder $\mathcal{D}$ back to the identity of the traitor, the security manager picks two distinct random values $k, k' \in \mathbb{Z}_q$, along with a random $\hat{m} \in \mathcal{M}$, and feeds $\mathcal{D}$ with the "illegal" ciphertext $\hat{\psi} \doteq \langle k'P, kQ, \hat{m} \rangle$. If the output of $\mathcal{D}$ is $\hat{m} \oplus H(k'\alpha_\sigma P + k\beta_\sigma Q)$, then the algorithm returns the identity σ as the traitor; otherwise it outputs 0.

In [KY02b], the authors show that the above two-user scheme is secure and traceable (for up to 1 traitor) in the sense of Definitions 7 and 8 under the DDH assumption (*cf.* Appendix A.3).

B.2 The Multi-User Scheme of [KY02b]

Setup: Given security parameters 1^κ, 1^t and ε, the algorithm works as follows:

Step 1: Generate a κ-bit prime q and a group $\mathcal{G}$ in which the DDH problem is difficult.[10] Generate an (ε, t, n, v)-collusion-secure code $\mathcal{C} = \{\omega^{(1)}, \ldots, \omega^{(n)}\}$ over $\{0, 1\}$.

[9] Even though [KY02b] used the multiplicative notation, we use here the additive notation for the sake of consistency with the rest of the paper (*cf.* Footnote 10).

[10] Even though [KY02b] used the multiplicative notation, we use here the additive notation for the sake of consistency with the rest of the paper. Notice, however, that $\mathcal{G}$ should not be identified with the group $\mathcal{G}_1$ used elsewhere in this paper, and in particular $\mathcal{G}$ should not be equipped with a bilinear map, for that would violate the required hardness of the DDH problem in $\mathcal{G}$.

Step 2: For each $j = 1, \ldots, v$, let P_j be a generator of $\mathcal{G}$, pick random $a_j, c_j \in \mathbb{Z}_q^*$, and set $Q_j \doteq a_j P_j$, $Z_j \doteq c_j P_j$. For each $j = 1, \ldots, v$, compute two linearly independent vectors $(\alpha_{j,0}, \beta_{j,0})$, $(\alpha_{j,1}, \beta_{j,1})$ in $\mathbb{Z}_q^2$ such that $\alpha_{j,\sigma} + a\beta_{j,\sigma} = c_j \bmod q$, for $\sigma \in \{0,1\}$. The private key of the security manager is set to be $\mathsf{msk} \doteq (a_j, \alpha_{j,0}, \beta_{j,0}, \alpha_{j,1}, \beta_{j,1})_{j=1,\ldots,v}$.

Step 3: Choose a universal hash function $H : \mathcal{G} \to \{0,1\}^\kappa$, and set the public key to $\mathsf{pk} \doteq (q, \mathcal{G}, H, (P_1, Q_1, Z_1), \ldots, (P_v, Q_v, Z_v))$. The message space is $\mathcal{M} \doteq (\{0,1\}^\kappa)^v$.

Reg: For each user i, the security manager first retrieves the corresponding codeword $\omega^{(i)} \in \mathcal{C}$, and then, for each $j = 1, \ldots, v$, gives u_i one of the two pairs $(\alpha_{j,0}, \beta_{j,0})$ or $(\alpha_{j,1}, \beta_{j,1})$, according to the value of $\omega_j^{(i)}$ (the j-th bit of the codeword $\omega^{(i)}$). The secret key of user i is then set to be $\mathsf{sk}_i \doteq (\alpha_{j,\omega_j^{(i)}}, \beta_{j,\omega_j^{(i)}})_{j=1,\ldots,v}$. Notice that, for $j = 1, \ldots, v$, $Z_j = c_j P_j = \alpha_{j,\omega_j^{(i)}} P_j + \beta_{j,\omega_j^{(i)}} Q_j$.

Enc: Given pk, anybody can encrypt a message $m = (m^{(1)} \| \ldots \| m^{(v)}) \in \mathcal{M}$ by first selecting random $k_1, \ldots, k_v \in \mathbb{Z}_q$ and then creating a ciphertext $\psi \doteq (\langle U_1, V_1, W_1 \rangle, \ldots, \langle U_v, V_v, W_v \rangle) \in (\mathcal{G}^2 \times \{0,1\}^\kappa)^v$ where $U_j \doteq k_j P_j$, $V_j \doteq k_j Q_j$ and $W_j \doteq m^{(j)} \oplus H(k_j Z_j)$, $j = 1, \ldots, v$.

Dec: Given a ciphertext $\psi = (\langle U_1, V_1, W_1 \rangle, \ldots, \langle U_v, V_v, W_v \rangle)$, user i computes $k_j Z_j = \alpha_{j,\omega_j^{(i)}} U_j + \beta_{j,\omega_j^{(i)}} V_j$ and recovers $m^{(j)} = W_j \oplus H(k_j Z_j)$, for $j = 1, \ldots, v$.

Trace: To trace a decoder $\mathcal{D}$ back to the identity of one of the traitors, the security manager prepares an illegal ciphertext $\hat{\psi} \doteq (\hat{\psi}_1, \ldots, \hat{\psi}_v)$, where each $\hat{\psi}_j$ is constructed as in the tracing algorithm from Appendix B.1 (*i.e.*, $\hat{\psi}_j \doteq \langle k'_j P_j, k_j Q_j, \hat{m}_j \rangle$, for random $k_j, k'_j \xleftarrow{R} \mathbb{Z}_q$ and $\hat{m}_j \xleftarrow{R} \{0,1\}^\kappa$). Let $m \doteq (m^{(1)} \| \ldots \| m^{(v)})$ be the plaintext output by $\mathcal{D}$ when fed with the ciphertext $\hat{\psi}$.

The security manager forms a "traitor codeword" $\hat{\omega} \doteq (\hat{\omega}^{(1)}, \ldots, \hat{\omega}^{(v)}) \in \{0, 1, \text{'?'}\}^v$, where each $\hat{\omega}^{(j)}$ is derived from $m^{(j)}$ as in the tracing algorithm for the two-user scheme (*i.e.*, $\hat{\omega}^{(j)} \doteq \sigma_j$ if $m^{(j)} = \hat{m}_j \oplus H(k'_j \alpha_{j,\sigma_j} P_j + k_j \beta_{j,\sigma_j} Q_j)$ (for $\sigma_j = \{0,1\}$), or $\hat{\omega}^{(j)} \doteq$ '?' otherwise).

At this point, the "traitor codeword" $\hat{\omega}$ is run through the tracing algorithm $\mathcal{T}(r_\mathcal{C}, \cdot)$ of the collusion-secure code $\mathcal{C}$ (where $r_\mathcal{C}$ are the random coins used by the security manager in generating $\mathcal{C}$). Finally, **Trace** outputs whichever value in $\{1, \ldots, n, 0\}$ returned by $\mathcal{T}(r_\mathcal{C}, \hat{\omega})$.

B.3 The Two-User Sub-Scheme of [CPP05]

Setup: Given a security parameter 1^κ, the algorithm works as follows:

Step 1: Generate a κ-bit prime q, two groups $\mathcal{G}_1$ and $\mathcal{G}_2$ of order q, and an admissible bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$. Let P be a generator of $\mathcal{G}_1$ and set $g \doteq e(P, P)$.

Step 2: Pick random elements $a, c \in \mathbb{Z}_q^*$, and set $Q \doteq aP$, $h \doteq g^c$. The private key of the security manager is set to be the pair $\mathsf{msk} \doteq (a, c)$.

Step 3: The security manager selects two linearly independent vectors (α_0, β_0) and (α_1, β_1) in $\mathbb{Z}_q^2$ such that $\alpha_\sigma + a\beta_\sigma = c \bmod q$, for $\sigma \in \{0,1\}$. It chooses a universal hash function $H : \mathcal{G}_2 \to \{0,1\}^\kappa$, and set the public key of the scheme to be the tuple $\mathsf{pk} \doteq (q, \mathcal{G}_1, \mathcal{G}_2, e, H, g, P, Q, h, \alpha_0 P, \beta_0 P, \alpha_1 P, \beta_1 P)$. The message space is $\mathcal{M} \doteq \{0,1\}^\kappa$.

Reg: The secret key of user σ is set to be $\mathsf{sk}_\sigma \doteq (\alpha_\sigma)$. Notice that: $cP = \alpha_\sigma P + \beta_\sigma Q$ and hence $e(P, cP) = e(P, \alpha_\sigma P) \cdot e(Q, \beta_\sigma P) = e(P, A_\sigma) \cdot e(Q, B_\sigma)$, for $\sigma \in \{0,1\}$.

Enc: Given pk, anybody can encrypt a message $m \in \mathcal{M}$ by first selecting a random $k \in \mathbb{Z}_q$ and then creating the ciphertext $\psi \doteq \langle U, V, W \rangle \in \mathcal{G}_1^2 \times \mathcal{M}$ where

$$U \doteq kP, \quad V \doteq k^2 Q, \quad W \doteq m \oplus H(h^{k^2})$$

Dec: Given a ciphertext $\psi = \langle U, V, W \rangle$, user σ computes $h^{k^2} = e(U, \alpha_\sigma U) \cdot e(V, B_\sigma)$ and recovers $m = W \oplus H(h^{k^2})$.

Trace: To trace a decoder $\mathcal{D}$ back to the identity of the traitor, the tracer picks two distinct random values $k, k' \in \mathbb{Z}_q$, along with a random $\hat{m} \in \mathcal{M}$, and feeds $\mathcal{D}$ with the "illegal" ciphertext $\hat{\psi} \doteq \langle k'P, k^2Q, \hat{m}\rangle$. If the output of $\mathcal{D}$ is $\hat{m} \oplus H(e(\alpha_\sigma P, P)^{k'^2} \cdot e(\beta_\sigma P, Q)^{k^2})$, then the algorithm returns the identity σ as the traitor; otherwise it outputs 0.

In [CPP05], the above two-user scheme is proven secure and traceable (for up to 1 traitor) in the sense of Definitions 7 and 8 under two non-standard assumptions for bilinear groups, respectively called DBDH^2-E and DBDH^1-M in [CPP05] (*cf.* Appendix A.4).

B.4 The Multi-User Scheme of [CPP05]

We now describe the multi-user scheme[11] of [CPP05], which is based on the use of bilinear maps. The key difference from the multi-user scheme of [KY02b] is the idea of *proxy quantity*: the security manager selects the master secret key roughly as in [KY02b], but now some secret information is removed from the users' secret keys and a derived value (the proxy quantity) is lifted to the public key.

These public proxy quantities are sufficient to decrypt and contain less information about the master secret key. This makes it (seemingly) safe to reuse the same parameters P and Q (in the public key) and the same randomness k (in the ciphertext) for all v components of the multi-user scheme. This (seemingly) results in a significant bonus, as it allows for considerably shorter public keys and ciphertexts.

Setup: Given the security parameters 1^κ, 1^t and ε, the algorithm works as follows:

Step 1: Generate a κ-bit prime q, two groups $\mathcal{G}_1$ and $\mathcal{G}_2$ of order q, and an admissible bilinear map $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$. Let P be a generator of $\mathcal{G}_1$ and set $g \doteq e(P, P)$.
Generate an (ε, t, n, v)-collusion-secure code $\mathcal{C} = \{\omega^{(1)}, \ldots, \omega^{(n)}\}$ over $\{0, 1\}$.

Step 2: Pick random elements $a, c_j \in \mathbb{Z}_q^*$ $(j = 1, \ldots, v)$, and set $Q \doteq aP$, $h_j \doteq g^{c_j}$, $j = 1, \ldots, v$. For each $j = 1, \ldots, v$, compute two linearly independent vectors $(\alpha_{j,0}, \beta_{j,0})$, $(\alpha_{j,1}, \beta_{j,1})$ in $\mathbb{Z}_q^2$ such that $\alpha_{j,\sigma} + a\beta_{j,\sigma} = c_j \bmod q$, for $\sigma \in \{0, 1\}$. The private key of the security manager is set to be $\mathsf{msk} \doteq (a, (\alpha_{j,0}, \beta_{j,0}, \alpha_{j,1}, \beta_{j,1})_{j=1,\ldots,v})$.

Step 3: For $j = 1, \ldots, v$ and $\sigma \in \{0, 1\}$, let $A_{j,\sigma} \doteq \alpha_{j,\sigma}P$ and $B_{j,\sigma} \doteq \beta_{j,\sigma}P$. Choose a universal hash function $H : \mathcal{G}_2 \to \{0, 1\}^\kappa$, and set the public key to: $\mathsf{pk} \doteq (q, \mathcal{G}_1, \mathcal{G}_2, e, H, P, Q, (h_j, A_{j,0}, B_{j,0}, A_{j,1}, B_{j,1})_{j=1,\ldots,v})$. The message space is $\mathcal{M} \doteq (\{0, 1\}^\kappa)^v$.

Reg: For each user i, the security manager retrieves the corresponding codeword $\omega^{(i)} \in \mathcal{C}$, and sets the secret key of user i to be: $\mathsf{sk}_i \doteq (\alpha_{j,\omega_j^{(i)}})_{j=1,\ldots,v}$. Notice that, for $j = 1, \ldots, v$, $c_jP = \alpha_{j,\omega_j^{(i)}}P + \beta_{j,\omega_j^{(i)}}Q$ and hence, $h_j = e(P, c_jP) = e(P, \alpha_{j,\omega_j^{(i)}}P) \cdot e(Q, \beta_{j,\omega_j^{(i)}}P) = e(P, A_{j,\omega_j^{(i)}}) \cdot e(Q, B_{j,\omega_j^{(i)}})$.

Enc: Given pk, anybody can encrypt a message $m = (m^{(1)} \| \ldots \| m^{(v)}) \in \mathcal{M}$ by first selecting a random $k \in \mathbb{Z}_q$ and then creating a ciphertext $\psi \doteq \langle U, V, (W_1, \ldots, W_v)\rangle \in \mathcal{G}_1^2 \times \mathcal{M}$, where $U \doteq kP$, $V \doteq k^2Q$ and $W_j \doteq m^{(j)} \oplus H(h_j^{k^2})$, $j = 1, \ldots, v$.

Dec: Given a ciphertext $\psi = \langle U, V, (W_1, \ldots, W_v)\rangle \in \mathcal{G}_1^2 \times \mathcal{M}$, user i computes (for $j = 1, \ldots, v$) the mask $h_j^{k^2} = e(U, \alpha_{j,\omega_j^{(i)}}U) \cdot e(V, B_{j,\omega_j^{(i)}})$ and then recovers each $m^{(j)}$ as $m^{(j)} = W_j \oplus H(h_j^{k^2})$.

Trace: Although [CPP05] present a tracing algorithm only for their two-user scheme, the authors suggested therein that their multi-user scheme inherits the tracing capabilities of [KY02b]. In particular, we sketch here the obvious necessary modifications to the **Trace** algorithm in Appendix B.2: the illegal ciphertext has the form $\hat{\psi} \doteq \langle k'P, k^2Q, (\hat{m}_1, \ldots, \hat{m}_v)\rangle$, where $k, k' \stackrel{R}{\leftarrow} \mathbb{Z}_q$, and each $\hat{m}_j$ is random in $\{0, 1\}^\kappa$; and the "traitor codeword" $\hat{\omega} \doteq (\hat{\omega}^{(1)}, \ldots, \hat{\omega}^{(v)})$, is constructed from $\mathcal{D}$'s response $m \doteq (m^{(1)} \| \ldots \| m^{(v)})$ by defining each $\hat{\omega}^{(j)} \in \{0, 1, \text{'?'}\}$ as in the tracing for the two-user scheme

[11] In [CPP05], the authors present two schemes with the same parameters. For conciseness, here we only report the second scheme, which was claimed to also support local public traceability.

(*i.e.,* $\hat{\omega}^{(j)} \doteq \sigma_j$ if $m^{(j)} = \hat{m}_j \oplus H(e(\alpha_{j,\sigma_j} P, P)^{k'^2} \cdot e(\beta_{j,\sigma_j} P, Q)^{k^2})$ (for $\sigma_j = \{0,1\}$), or $\hat{\omega}^{(j)} \doteq$ '?' otherwise).

C On the Query Complexity of Black-Box Tracing in [KY02b]

Appendix B.2 reports the multi-user scheme of [KY02b], which includes a black-box tracing algorithm making a single query to the pirate decoder $\mathcal{D}$. Below we show that such algorithm is broken, and we present a simple traitor strategy that allows a coalition of just $2 < t$ users to escape tracing with probability 1. We also propose a variation of their black-box tracing algorithm, which requires v queries but is successful in tracing up to the desired threshold of traitors, thus suggesting that the query complexity of black-box tracing in [KY02b] is higher than what claimed therein.

C.1 A Simple Untraceable Traitor Strategy

Consider the coalition of 2 users, which for simplicity we will suppose associated with the first two codewords $\omega^{(1)}, \omega^{(2)}$ of $\mathcal{C}$. Since $\omega^{(1)} \neq \omega^{(2)}$, they must differ in at least one of their v bits, say the first bit.

This means that by pooling their secret keys, the two traitors can construct a pirate decoder $\mathcal{D}$ containing both user-keys $(\alpha_{1,0}, \beta_{1,0}), (\alpha_{1,1}, \beta_{1,1})$ for the two-user sub-scheme associated to index 1, plus at least one user-key for each of the remaining $(v-1)$ components. When given a ciphertext $\psi \doteq \langle \psi_1, \ldots, \psi_v \rangle$, $\mathcal{D}$ starts by decrypting ψ_1 twice: once using $(\alpha_{1,0}, \beta_{1,0})$, and then again using $(\alpha_{1,1}, \beta_{1,1})$. If the two resulting plaintexts coincide, then $\mathcal{D}$ decrypts the rest of ψ and output the resulting message; otherwise, $\mathcal{D}$ can conclude that it is being traced, and can just output a predetermined message (*e.g.,* the all-zero message).

Notice that $\mathcal{D}$ perfectly decrypts ciphertext distributed according to $\mathbf{Enc}(\mathsf{pk}, \cdot)$ since, by correctness of decryption, $\mathcal{D}$'s "integrity" check will always pass on a valid ciphertext. Moreover, $\mathcal{D}$ escapes tracing with probability 1, since the $\mathbf{Trace}$ algorithm of [KY02b] prepares the invalid ciphertext $\hat{\psi}$ by concatenating invalid ciphertexts $\hat{\psi}_j$ for each of the v components of the scheme. This will result in different decryptions of $\hat{\psi}_1$ under $(\alpha_{1,0}, \beta_{1,0})$ and $(\alpha_{1,1}, \beta_{1,1})$, and thus $\mathcal{D}$ will reply with a plaintext containing no information about the identities of the traitors.

C.2 The Fix

The problem with the $\mathbf{Trace}$ algorithm of [KY02b] is that it implicitly assumed that pirate decoders would decrypt each component of the ciphertext independently from each other, which clearly does not need to be the case. Bearing this in mind, the fix is immediate: it suffices for $\mathbf{Trace}$ to iteratively query the decoder with v ciphertexts, each constructed to be invalid in just one component, but valid elsewhere. Now, the independence of the v component sub-schemes implies that $\mathcal{D}$ will be unable to tell valid and invalid ciphertexts apart, unless it possesses both user-keys for the single sub-scheme "under testing." As a consequence, $\mathbf{Trace}$ will end up extracting a traitor codeword from $\mathcal{D}$ with at most t unreadable marks '?', and thus the tracing algorithm $\mathcal{T}(\cdot,\cdot)$ of the collusion-secure code $\mathcal{C}$ will successfully recover the identity of one of the traitor (with probability $1-\varepsilon$).

C.3 Consequences for the Multi-User Scheme of [CPP05]

Being based on the techniques of [KY02b], the multi-user scheme of [CPP05] inherits the problem pointed out in Appendix C.1. As it turns out, however, in this case the consequences are more severe. In particular, the easy fix that we proposed for the scheme of [KY02b] in Appendix C.2 does not apply: interestingly, the higher correlation between the parameters used in the v components of the scheme of [CPP05], which proved crucial to attain optimal transmission rate, at the same time poses a serious impediment to black-box tracing.

Indeed, ciphertexts in the multi-user scheme of [CPP05] (*cf.* Appendix B.4) have the form $\psi \doteq \langle kP, k^2Q, (W_1, \ldots, W_v)\rangle$, in which the same "randomization" values kP, k^2Q are used for all the v two-user sub-schemes. Hence, it is not possible to make the ciphertext invalid in just one component, while preserving its validity in the remaining $(v-1)$ ones (which was the idea behind our fix in Appendix C.2). Therefore, it seems that the scheme of [CPP05], as given, does not support black-box tracing. Since the notion of local public traceability is only meaningful in the black-box setting, this also voids the claimed traceability features of the multi-user scheme of [CPP05].

To salvage black-box tracing and local public traceability, one could modify the scheme of [CPP05] and revert to the "parallel" composition of sub-schemes (exactly as in [KY02b]), thus "undoing" the optimization that enabled short ciphertexts. The resulting scheme, however, would just be a variant of [KY02b] with the same parameters, but with the additional need of bilinear maps and reliance on non-standard bilinear-related assumptions.

As a result, it seems appropriate to regard the multi-user scheme of [CPP05] as a scheme with optimal transmission rate, but with only non-black-box tracing and no public traceability features.

D Proofs from Section 4.5

D.1 Proof of Lemma 18

Lemma. If the multi-user scheme without AONT pre-processing is secure w.r.t. indistinguishability under chosen-plaintext attack (*cf.* Theorem 9), then the multi-user scheme with AONT pre-processing is secure w.r.t. the same notion.

Proof. The proof is by a straightforward reduction argument: given any efficient adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, having advantage ε in attacking the multi-user scheme with AONT pre-processing, we construct an adversary $\mathcal{B}$, with essentially the same running time as $\mathcal{A}$'s, having the same advantage ε in attacking the multi-user scheme without AONT pre-processing.

Adversary $\mathcal{B}$ just forwards $\mathcal{A}_1$ the public key for the scheme that it wants to attack. $\mathcal{A}_1$ will reply with two messages $m_0 \doteq (m_0^{(1)}, \ldots, m_0^{(v)})$ and $m_1 \doteq (m_1^{(1)}, \ldots, m_1^{(v)})$ on which to be challenged. Then $\mathcal{B}$ applies the all-or-nothing transform to both messages, obtaining $m'_0 \overset{R}{\leftarrow} \mathrm{AONT}(m_0)$ and $m'_1 \overset{R}{\leftarrow} \mathrm{AONT}(m_1)$. $\mathcal{B}$ then submits m'_0 and m'_1 to its challenger, and gets back a challenge ciphertext ψ^*. Notice that ψ^* is also a valid challenge ciphertext for $\mathcal{A}$, and so $\mathcal{B}$ directly forwards it to $\mathcal{A}_2$ as challenge (along with any state information that $\mathcal{A}_1$ might have output). Finally, $\mathcal{B}$ outputs whichever bit b' is returned by b.

Since $\mathcal{B}$ perfectly simulates the attack game that adversary $\mathcal{A}$ expects, $\mathcal{B}$'s advantage against the scheme without AONT pre-processing equals ε, completing the proof. □

D.2 Proof of Lemma 19

Lemma. If the two-user scheme in Section 4.1 is secure w.r.t. indistinguishability under chosen-plaintext attack (*cf.* Theorem 9), then the multi-user scheme in Section 4.4 is secure w.r.t. the same notion.

Proof. For the sake of clarity, in the security proof, we will follow the structural approach advocated in [Sho04]. Starting from the actual attack scenario (as defined in Definition 7), we consider a sequence of hypothetical games, all defined over the same underlying probability space. In each game, the adversary's view is obtained in a slightly different way, but its distribution is maintained (computationally) indistinguishable across the games. In the last game, it will be clear that the adversary has (at most) a negligible advantage;

by the indistinguishability of any two consecutive games, it will follow that also in the original game the adversary's advantage is negligible.

Fix any efficient adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, along with its random tape. Fix also the randomness used by the challenger in the execution of the $\mathbf{Setup}$ and $\mathbf{Enc}$ algorithms, and the random bit b^* used in creating the challenge ciphertext ψ^*. In each game $\mathbf{G}_i$, the goal of adversary is to guess such bit b^*. Let b' be the random variable denoting the bit output by $\mathcal{A}_2$ at the end of the game, and denote with S_i the event that $b' = b^*$ in game $\mathbf{G}_i$.

Game $\mathbf{G}_0$. Define $\mathbf{G}_0$ to be the original game as described in Definition 7.

Game $\mathbf{G}_1$. This game is identical to game **Game $\mathbf{G}_0$**, except that the $\mathbf{Enc}$ algorithm in $\mathbf{G}_1$ is modified so that the "special component" of the ciphertext is computed as follows:

$$k_\ell \xleftarrow{R} \mathbb{Z}_q,\ U_\ell \leftarrow e(P_\ell, R_\ell)^{k_\ell},\ V_\ell \leftarrow k_\ell Q_\ell,\ \boxed{W_\ell \xleftarrow{R} \{0,1\}^\kappa}$$

In other words, rather than being set as $W_\ell \leftarrow m_{b^*}^{(\ell)} \oplus H(h_\ell^{k_\ell})$, in game $\mathbf{G}_1$ W_ℓ is a random κ-bit value.

Claim (1). $|\Pr[S_0] - \Pr[S_1]| \leq 2\varepsilon^{(1)}$, where $\varepsilon^{(1)}$ is the advantage of some efficient adversary attacking the security of the 2-user scheme from Section 4.1.

The proof of this is by a standard reduction argument, by which any non-negligible difference in behavior between game $\mathbf{G}_0$ and $\mathbf{G}_1$ can be used to construct an efficient adversary $\mathcal{B}^{(1)}$ successfully attacking the security of the 2-user scheme from Section 4.1. More precisely, $\mathcal{B}^{(1)}$ gets in input a 2-user public key $(\tilde{P}, \tilde{Q}, \tilde{R}, \tilde{A}_0, \tilde{B}_0, \tilde{A}_1, \tilde{B}_1)$, and proceeds as follows:

Setup: To create the public key for the multi-user scheme to be fed to $\mathcal{A}_1$, $\mathcal{B}^{(1)}$ proceeds exactly according to the corresponding key generation algorithm, except that, for the parameters corresponding to the ℓ-th special component,[12] $\mathcal{B}^{(1)}$ uses the values from the public key that it received as its own input:

$$P_\ell \leftarrow \tilde{P},\ Q_\ell \leftarrow \tilde{Q},\ R_\ell \leftarrow \tilde{R},\ A_{\ell,0} \leftarrow \tilde{A}_0,\ B_{\ell,0} \leftarrow \tilde{B}_0,\ A_{\ell,1} \leftarrow \tilde{A}_1,\ B_{\ell,1} \leftarrow \tilde{B}_1$$

$\mathcal{B}^{(1)}$ then sends $\mathcal{A}_1$ the resulting multi-user public key.

Challenge: $\mathcal{A}_1$ outputs two messages $m_0 \doteq (m_0^{(1)} \| \ldots \| m_0^{(v)}), m_1 \doteq (m_1^{(1)} \| \ldots \| m_1^{(v)})$ on which it wishes to be challenged, along with some state τ to be passed to $\mathcal{A}_2$. Now $\mathcal{B}^{(1)}$, in turn, has to choose two messages, $\tilde{m}_0$ and $\tilde{m}_1$, for its own challenge. So $\mathcal{B}^{(1)}$ chooses $b^* \in \{0,1\}$ at random, sets $\tilde{m}_{b^*} \doteq m_{b^*}^{(\ell)}$, and picks $\tilde{m}_{1-b^*} \xleftarrow{R} \{0,1\}^\kappa$. At this point, $\mathcal{B}^{(1)}$ is given a challenge ciphertext $\tilde{\psi} \doteq \langle \tilde{U}, \tilde{V}, \tilde{W} \rangle$, where $\tilde{U} \doteq e(\tilde{P}, \tilde{R})^{\tilde{k}}$, $\tilde{V} \doteq \tilde{k}\tilde{Q}$, $\tilde{h} = e(\tilde{P}, \tilde{A}_0) \cdot e(\tilde{Q}, \tilde{B}_0)$ and $\tilde{W} \doteq \tilde{m}_{\tilde{b}} \oplus H(\tilde{h}^{\tilde{k}})$. Recall that $\mathcal{B}^{(1)}$'s job is to guess the bit $\tilde{b}$ that was used to create its challenge. To this end, $\mathcal{B}^{(1)}$ prepares a challenge ciphertext ψ^* for $\mathcal{A}_2$ by faithfully running the $\mathbf{Enc}$ algorithm on the message $m_{b^*} \doteq (m_{b^*}^{(1)} \| \ldots \| m_{b^*}^{(v)})$, except that, for the special component, rather than choosing a random k^ℓ and properly encrypting the message block $m_{b^*}^{\ell}$, $\mathcal{B}^{(1)}$ uses the values contained in its own challenge $\tilde{\psi}$:

$$U_\ell \doteq \tilde{U},\ V_\ell \doteq \tilde{V},\ W_\ell \doteq \tilde{W}.$$

Then, $\mathcal{B}^{(1)}$ sends $\mathcal{A}_2$ the challenge ciphertext $\psi^* \doteq \langle \ell, U_\ell, V_\ell, U, V, W_1, \ldots, W_v \rangle$ so computed, along with the state information τ.

Guess: Algorithm $\mathcal{A}_2$ outputs a guess $b' \in \{0,1\}$, which $\mathcal{B}^{(1)}$ also gives in output as its own guess to $\tilde{b}$.

[12] Notice that the value of ℓ is fixed within this proof, since we fixed the randomness for $\mathbf{Enc}$ across the games.

It should be clear by inspection that adversary $\mathcal{B}^{(1)}$ 'interpolates' between games $\mathbf{G}_0$ and $\mathbf{G}_1$ for $\mathcal{A}$, in the sense that if $b^* = \tilde{b}$, then the view of adversary $\mathcal{A}$ is computed exactly as in $\mathbf{G}_0$, whereas if $b^* = 1 - \tilde{b}$, then the computation proceeds according to $\mathbf{G}_1$. Thus, it holds that:

$$\Pr[S_0] = \Pr[b' = b^* \mid b^* = \tilde{b}] \qquad \text{and} \qquad \Pr[S_1] = \Pr[b' = b^* \mid b^* = 1 - \tilde{b}].$$

Now, let $\varepsilon^{(1)}$ be adversary $\mathcal{B}^{(1)}$'s advantage in guessing $\tilde{b}$: $\varepsilon^{(1)} \doteq |\Pr[b' = \tilde{b}] - 1/2|$. Splitting the probability according to the event space partition $(b^* = \tilde{b}) \vee (b^* = 1 - \tilde{b})$, we get

$$\begin{aligned}
\Pr[b' = \tilde{b}] &= \Pr[b' = \tilde{b} \mid b^* = \tilde{b}] \cdot \Pr[b^* = \tilde{b}] + \Pr[b' = \tilde{b} \mid b^* = 1 - \tilde{b}] \cdot \Pr[b^* = 1 - \tilde{b}] \\
&= \frac{1}{2}(\Pr[b' = \tilde{b} \mid b^* = \tilde{b}] + \Pr[b' = \tilde{b} \mid b^* = 1 - \tilde{b}]) \\
&= \frac{1}{2}(\Pr[b' = \tilde{b} \mid b^* = \tilde{b}] + 1 - \Pr[b' = 1 - \tilde{b} \mid b^* = 1 - \tilde{b}]) \\
&= \frac{1}{2} + \frac{1}{2}(\Pr[b' = b^* \mid b^* = \tilde{b}] - \Pr[b' = b^* \mid b^* = 1 - \tilde{b}]) \\
&= \frac{1}{2} + \frac{1}{2}(\Pr[S_0] - \Pr[S_1])
\end{aligned}$$

It thus follows that $|\Pr[S_0] - \Pr[S_1]| = 2|\Pr[b' = \tilde{b}] - 1/2| = 2\varepsilon^{(1)}$, as claimed. □

Game $\mathbf{G}_i$, $2 \le i \le \ell$. This game is identical to game **Game $\mathbf{G}_{i-1}$**, except that the **Enc** algorithm in $\mathbf{G}_{i-1}$ is modified so that W_{i-1}, rather than properly encrypting the message block $m_{b^*}^{(i-1)}$, is chosen as a random κ-bit value:

$$\boxed{W_{i-1} \stackrel{R}{\leftarrow} \{0,1\}^\kappa}$$

Claim (i). $|\Pr[S_{i-1}] - \Pr[S_i]| \le 2\varepsilon^{(i)}$, where $\varepsilon^{(i)}$ is the advantage of some efficient adversary attacking the security of the 2-user scheme from Section 4.1.

Again, we will prove the claim by showing how any non-negligible difference in behavior between game $\mathbf{G}_{i-1}$ and $\mathbf{G}_i$ can be used to construct an efficient adversary $\mathcal{B}^{(i)}$ successfully attacking the security of the 2-user scheme from Section 4.1.

More precisely, $\mathcal{B}^{(i)}$ gets in input a 2-user public key $(\tilde{P}, \tilde{Q}, \tilde{R}, \tilde{A}_0, \tilde{B}_0, \tilde{A}_1, \tilde{B}_1)$, and proceeds as follows:

Setup: To create the public key for the multi-user scheme to be fed to $\mathcal{A}_1$, $\mathcal{B}^{(i)}$ proceeds exactly according to the corresponding key generation algorithm, except that for the parameters corresponding to the "shared scheme," $\mathcal{B}^{(i)}$ bases its computations on the values included in the 2-user public key that it received as its own input:

$$\begin{array}{ll}
P \leftarrow \tilde{P},\ Q \leftarrow \tilde{Q},\ R \leftarrow \tilde{R} & \\
\bar{A}_{i,0} \leftarrow \tilde{A}_0,\ \bar{B}_{i,0} \leftarrow \tilde{B}_0,\ \bar{A}_{i,1} \leftarrow \tilde{A}_1,\ \bar{B}_{i,1} \leftarrow \tilde{B}_1 & \\
\bar{\beta}_{j,0} \stackrel{R}{\leftarrow} \mathbb{Z}_q, \quad \bar{B}_{j,0} \leftarrow \bar{\beta}_{j,0}\tilde{P} & (j = 1, \ldots, v,\ j \neq i) \\
\bar{\alpha}_{j,0} \stackrel{R}{\leftarrow} \mathbb{Z}_q, \quad \bar{A}_{j,0} \leftarrow \bar{\alpha}_{j,0}\tilde{R} & (j = 1, \ldots, v,\ j \neq i) \\
\bar{\beta}_{j,1} \stackrel{R}{\leftarrow} \mathbb{Z}_q, \quad \bar{B}_{j,1} \leftarrow \bar{\beta}_{j,1}\tilde{P} & (j = 1, \ldots, v,\ j \neq i) \\
\bar{A}_{j,1} \leftarrow \bar{A}_{j,0} + \bar{\beta}_{j,0}\tilde{Q} - \bar{\beta}_{j,1}\tilde{Q} & (j = 1, \ldots, v,\ j \neq i)
\end{array}$$

(Notice that the last set of positions guarantee that, for all values of j, it holds that:

$$e(P, \bar{A}_{j,0}) \cdot e(Q, \bar{B}_{j,0}) = e(P, \bar{A}_{j,1}) \cdot e(Q, \bar{B}_{j,1}),$$

so that in fact we can define $\bar{h}_j \doteq e(P, \bar{A}_{j,\sigma}) \cdot e(Q, \bar{B}_{j,\sigma})$, for $\sigma \in \{0,1\}, j = 1, \ldots, v$, as in the actual **Setup** algorithm for the multi-user scheme (*cf.* Section 4.4).)

$\mathcal{B}^{(1)}$ then sends $\mathcal{A}_1$ the resulting multi-user public key.

Challenge: $\mathcal{A}_1$ outputs two messages $m_0 \doteq (m_0^{(1)} \| \ldots \| m_0^{(v)}), m_1 \doteq (m_1^{(1)} \| \ldots \| m_1^{(v)})$ on which it wishes to be challenged, along with some state τ to be passed to $\mathcal{A}_2$. Now $\mathcal{B}^{(1)}$, in turn, has to choose two messages, $\tilde{m}_0$ and $\tilde{m}_1$, for its own challenge. So $\mathcal{B}^{(1)}$ chooses $b^* \in \{0,1\}$ at random, sets $\tilde{m}_{b^*} \doteq m_{b^*}^{(i-1)}$, and picks $\tilde{m}_{1-b^*} \stackrel{R}{\leftarrow} \{0,1\}^\kappa$. At this point, $\mathcal{B}^{(1)}$ is given a challenge ciphertext $\tilde{\psi} \doteq \langle \tilde{U}, \tilde{V}, \tilde{W} \rangle$, where $\tilde{U} \doteq e(\tilde{P}, \tilde{R})^{\tilde{k}}$, $\tilde{V} \doteq \tilde{k}\tilde{Q}$, $\tilde{h} = e(\tilde{P}, \tilde{A}_0) \cdot e(\tilde{Q}, \tilde{B}_0)$ and $\tilde{W} \doteq \tilde{m}_{\tilde{b}} \oplus H(\tilde{h}^{\tilde{k}})$. Recall that $\mathcal{B}^{(1)}$'s job is to guess the bit $\tilde{b}$ that was used to create its challenge. To this end, $\mathcal{B}^{(1)}$ prepares a challenge ciphertext ψ^* for $\mathcal{A}_2$ as follows:

$$
\begin{aligned}
& k_\ell \stackrel{R}{\leftarrow} \mathbb{Z}_q,\ U_\ell \leftarrow e(P_\ell, R_\ell)^{k_\ell},\ V_\ell \leftarrow k_\ell Q_\ell,\ W_\ell \stackrel{R}{\leftarrow} \{0,1\}^\kappa \\
& U \leftarrow \tilde{U},\ V \leftarrow \tilde{V},\ W_{i-1} \leftarrow \tilde{W} \\
& W_j \stackrel{R}{\leftarrow} \{0,1\}^\kappa, \quad (j = 1, \ldots, i-2) \\
& W_j \leftarrow m_{b^*}^{(j)} \oplus H(\tilde{U}^{\bar{\alpha}_{j,0}} \cdot e(\tilde{V}, \bar{B}_{j,0})), \quad (j = i, \ldots, v,\ j \neq \ell)
\end{aligned}
$$

Notice that, for $j = i, \ldots, v, j \neq \ell$, the W_j values computed by $\mathcal{B}^{(i)}$ are proper encryptions of the corresponding $m_{b^*}^{(j)}$, since:

$$
\begin{aligned}
W_j &\leftarrow m_{b^*}^{(j)} \oplus H(\tilde{U}^{\bar{\alpha}_{j,0}} \cdot e(\tilde{V}, \bar{B}_{j,0})) \\
&= m_{b^*}^{(j)} \oplus H((e(\tilde{P}, \tilde{R})^{\tilde{k}})^{\bar{\alpha}_{j,0}} \cdot e(\tilde{k}\tilde{Q}, \bar{B}_{j,0})) \\
&= m_{b^*}^{(j)} \oplus H((e(\tilde{P}, \bar{\alpha}_{j,0}\tilde{R}) \cdot e(\tilde{Q}, \bar{B}_{j,0}))^{\tilde{k}}) \\
&= m_{b^*}^{(j)} \oplus H((e(P, \bar{A}_{j,0}) \cdot e(Q, \bar{B}_{j,0}))^{\tilde{k}}) \\
&= m_{b^*}^{(j)} \oplus H(\bar{h}_j^{\tilde{k}}).
\end{aligned}
$$

At this point, $\mathcal{B}^{(i)}$ sends $\mathcal{A}_2$ the challenge ciphertext $\psi^* \doteq \langle \ell, U_\ell, V_\ell, U, V, W_1, \ldots, W_v \rangle$ so computed, along with the state information τ.

Guess: Algorithm $\mathcal{A}_2$ outputs a guess $b' \in \{0,1\}$, which $\mathcal{B}^{(1)}$ also gives in output as its own guess to $\tilde{b}$.

Before arguing about the success probability of $\mathcal{B}^{(i)}$, notice that, by the definitions of games $\mathbf{G}_{i-1}$ and $\mathbf{G}_i$, the challenge ciphertexts that adversary $\mathcal{A}$ is given in both games have the same overall structure: they are completely random in the first few W_j components (as well as in the special component W_ℓ), whereas they are properly formed in the last few W_j components, $j \neq \ell$. The only difference is in the position where such "transition" from "random W_j" to "properly formed W_j" takes place: between indices $(i-2, i-1)$, in the case of game $\mathbf{G}_{i-1}$; and between indices $(i-1, i)$,[13] in the case of game $\mathbf{G}_{i-1}$.

It should then be clear, by the way adversary $\mathcal{B}^{(i)}$ prepares the challenge ciphetext ψ^* for adversary $\mathcal{A}$, that $\mathcal{B}^{(i)}$ effectively 'interpolates' between games $\mathbf{G}_{i-1}$ and $\mathbf{G}_i$ for $\mathcal{A}$, in the sense that: if $b^* = \tilde{b}$, then W_{i-1} is properly formed, and the view of adversary $\mathcal{A}$ is computed exactly as in game $\mathbf{G}_{i-1}$; whereas if $b^* = 1 - \tilde{b}$, then W_{i-1} is completely random, so that $\mathcal{A}$'s view is distributed as in game $\mathbf{G}_1$. It thus follows that:

$$
\Pr[S_{i-1}] = \Pr[b' = b^* \mid b^* = \tilde{b}] \qquad \text{and} \qquad \Pr[S_i] = \Pr[b' = b^* \mid b^* = 1 - \tilde{b}].
$$

[12] Clarify that ℓ has been fixed when we fixed the randomness for **Enc** across the games.

[13] For $i = \ell$, the transition is actually between indices $(\ell-1, \ell+1)$, since we are dealing with the special component W_ℓ separately.

Now, let $\varepsilon^{(i)}$ be adversary $\mathcal{B}^{(i)}$'s advantage in guessing $\tilde{b}$: $\varepsilon^{(i)} \doteq |\Pr[b' = \tilde{b}] - 1/2|$. Splitting the probability according to the event space partition $(b^* = \tilde{b}) \vee (b^* = 1 - \tilde{b})$, we get

$$\begin{aligned}
\Pr[b' = \tilde{b}] &= \Pr[b' = \tilde{b} \mid b^* = \tilde{b}] \cdot \Pr[b^* = \tilde{b}] + \Pr[b' = \tilde{b} \mid b^* = 1 - \tilde{b}] \cdot \Pr[b^* = 1 - \tilde{b}] \\
&= \frac{1}{2}(\Pr[b' = \tilde{b} \mid b^* = \tilde{b}] + \Pr[b' = \tilde{b} \mid b^* = 1 - \tilde{b}]) \\
&= \frac{1}{2}(\Pr[b' = \tilde{b} \mid b^* = \tilde{b}] + 1 - \Pr[b' = 1 - \tilde{b} \mid b^* = 1 - \tilde{b}]) \\
&= \frac{1}{2} + \frac{1}{2}(\Pr[b' = b^* \mid b^* = \tilde{b}] - \Pr[b' = b^* \mid b^* = 1 - \tilde{b}]) \\
&= \frac{1}{2} + \frac{1}{2}(\Pr[S_{i-1}] - \Pr[S_i])
\end{aligned}$$

It thus follows that $|\Pr[S_{i-1}] - \Pr[S_i]| = 2|\Pr[b' = \tilde{b}] - 1/2| = 2\varepsilon^{(i)}$, as claimed. □

Game $\mathbf{G}_i$, $\ell + 1 \le i \le v$. This game is identical to game **Game $\mathbf{G}_{i-1}$**, except that the **Enc** algorithm in $\mathbf{G}_{i-1}$ is modified so that W_i, rather than properly encrypting the message block $m^{(i)}_{b^*}$, is chosen as a random κ-bit value:

$$\boxed{W_i \stackrel{R}{\leftarrow} \{0,1\}^\kappa}$$

Claim (i). $|\Pr[S_{i-1}] - \Pr[S_i]| \le 2\varepsilon^{(i)}$, where $\varepsilon^{(i)}$ is the advantage of some efficient adversary attacking the security of the 2-user scheme from Section 4.1.

The proof of this is by a reduction argument completely analogous to the one used in proving the claims for the cases $2 \le i \le \ell$, the only difference being a notational one, since now the reduction will embed the challenge from the 2-user scheme into the component W_i (rather than W_{i-1}).[14] □

To conclude the proof, observe that, in game $\mathbf{G}_v$, all the W_j components in the challenge ciphertext $\psi^* \doteq \langle \ell, U_\ell, V_\ell, U, V, W_1, \ldots, W_v \rangle$ are just drawn at random from $\{0,1\}^\kappa$, so that no information about the random bit b^* is present in adversary $\mathcal{A}$'s view. It follows that the probability of a correct guess $b' = b^*$ by $\mathcal{A}$ in game $\mathbf{G}_v$ is just $1/2$, *i.e.*:

$$\Pr[S_v] = \frac{1}{2}$$

Combining the last equation with the intermediate results from Claims 1–v, we can conclude that

$$\Pr[S_0] \le \frac{1}{2} + 2v\varepsilon^{\text{2-user}}_{\text{ind}},$$

where $\varepsilon^{\text{2-user}}_{\text{ind}}$ is an upper bound on the advantage of any efficient adversary attacking the security of the 2-user scheme from Section 4.1, which is negligible by the hypothesis of the lemma, completing the proof. □

E A Comparison with [BSW06,BW06]

Recently, Boneh *et al.* [BSW06,BW06] proposed traitor tracing schemes that withstand any number of traitors (*full traceability*), while requiring a sub-linear ciphertext length ($O(\sqrt{n})$). While the schemes of [BSW06,BW06] are the most efficient ones supporting full collusion, they are not well suited for the more practical case of small number of traitors (say, logarithmic in the size of the entire user population). Indeed, in this case, the ciphertext in these schemes still contains $O(\sqrt{n})$ elements. In our scheme, assuming the number of traitors t is logarithmic in the number of users n, the ciphertext has poly-logarithmic

[14] The reason for this notational change is just to "jump" over the special component ℓ, which is treated separately in game $\mathbf{G}_1$.

length $v = O(t^2(\log n + \log \frac{1}{\varepsilon}))$ = $O(\log^3 n)$, which is asymptotically superior to the $O(\sqrt{n})$-ciphertexts of [BSW06,BW06].

More importantly, the tracing algorithms of [BSW06,BW06] require $O(n^2)$ decryption queries to the pirate decoder, whereas our scheme employs $O(v) = O(\log^3 n)$ decryption queries, and is completely parallelizable.

In brief, the schemes of [BSW06,BW06] are preferable in case of full collusions, whereas our scheme has advantages in term of efficiency and of complexity of black-box tracing when the number of traitors is logarithmic.