Cost-effective Private Linear Key Agreement with Adaptive CCA Security from Prime Order Multilinear Maps and Tracing Traitors

Mriganka Mandal and Ratna Dutta
Department of Mathematics, Indian Institute of Technology Kharagpur, Kharagpur, 721302, India

Keywords: Broadcast Encryption, Traitor Tracing, Multilinear Maps, Indistinguishability Obfuscation.

Abstract: *Private linear key agreement* (PLKA) enables a group of users to agree upon a common session key in a broadcast encryption (BE) scenario, while *traitor tracing* (TT) system allows a tracer to identify conspiracy of a troop of colluding pirate users. This paper introduces a *key encapsulation* mechanism in BE that provides the functionalities of both PLKA and TT in a unified *cost-effective* primitive. Our PLKA based traitor tracing offers a solution to the problem of achieving *full collusion resistance* property and *public traceability* simultaneously with significant efficiency and storage compared to a sequential improvement of the PLKA based traitor tracing systems. Our PLKA builds on a *prime order* multilinear group setting employing indistinguishability obfuscation ($i\mathcal{O}$) and pseudorandom function (PRF). The resulting scheme has a fair communication, storage and computational efficiency compared to that of *composite* order groups. Our PLKA is *adaptively chosen ciphertext attack* (CCA)-secure and based on the hardness of the multilinear assumption, namely, the Decisional Hybrid Diffie-Hellman Exponent (DHDHE) assumption in standard model and so far a plausible improvement in the literature. More precisely, our PLKA design significantly reduces the ciphertext size, public parameter size and user secret key size. We frame a traitor tracing algorithm with *shorter* running time which can be executed *publicly*.

1 INTRODUCTION

A *private linear key agreement* (PLKA) under key encapsulation framework requires the broadcaster to broadcast a common message, called header, for a specific type of user sets $[i] \in \mathcal{S}$ where $\mathcal{S} = \{[1], \ldots, [N]\} \subset 2^{[N]}$ and $[i] = \{1, \ldots, i\}$ is the collection of users. Each user is assigned a private key by a group manager (GM). The GM is a trusted third party and the role of a broadcaster may be played by the GM or by a seperate entity depending on applications. The header along with the user's pre-assigned private key enables users in $[i]$ to extract a session key common to all the users in $[i]$. On the other hand, a PLKA based broadcast encryption (BE) empowers a content broadcaster to broadcast an encrypted message under a common session key for $[i] \in \mathcal{S}$ so that a user $u \in [i]$ can decrypt the ciphertext using his private key. The users outside $[i]$ obtain nothing even if they collude for both the key encapsulation model and broadcast model of PLKA. The first construction for PLKA was designed by (Boneh et al., 2006; Boneh and Waters, 2006) followed by a number of works (Garg et al., 2010; Boneh and Zhandry, 2014; Nishimaki et al., 2016).

Consider a traditional cable TV system where the broadcaster broadcasts a classified digital content encrypted under a publicly known key to a set of legitimate users. Each legitimate user, having a valid private key embedded within a set-top box provided by the GM, can successfully decrypt and recover the classified content. Any user, who has paid to get his private key from the GM, might make a reprint to resell his private key or even publish it on the Internet. This allows unauthorized users to decrypt the classified content without having a legal authorization, causing the broadcaster a massive financial loss. Consequently, the broadcaster will attempt to identify those rouge user.

A *Traitor tracing* (TT) system is devised to aid content broadcasters to identify conspiracy of defrauders who create a *pirate decoder* box. A coalition of traitors might make a conspiracy to create the pirate decoder containing an arbitrarily complex and even obfuscated malicious program and is capable of decrypting the encrypted digital content. The traitors might alter their private keys in such a way that the altered keys cannot be linked with their

Mandal, M. and Dutta, R.
Cost-effective Private Linear Key Agreement with Adaptive CCA Security from Prime Order Multilinear Maps and Tracing Traitors.
DOI: 10.5220/0006845503560363
In *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications (ICETE 2018) - Volume 2: SECRYPT*, pages 356-363
ISBN: 978-989-758-319-3

Table 1: Comparison summary of communication, storage and other functionality.

Scheme	Group Type	$\lvert\mathsf{PP}\rvert$	$\lvert\mathsf{sk}_u\rvert$	$\lvert\mathsf{CT}\rvert$	TRA	Complexity Assumptions
(Boneh and Waters, 2006)	composite, BL	$9\sqrt{N}+5$	$(\sqrt{N}+1)$ in $\mathbb{G}$	$6\sqrt{N}$ in $\mathbb{G}$, $\sqrt{N}$ in $\mathbb{G}_T$	public	D3DH, DHSD, BSD
(Boneh et al., 2006)	composite, BL	$4\sqrt{N}+3$	1 in $\mathbb{G}$	$5\sqrt{N}$ in $\mathbb{G}$, $\sqrt{N}$ in $\mathbb{G}_T$	secret	D3DH, DHSD, BSD
(Garg et al., 2010)	prime, BL	$4\sqrt{N}+1$	$(\sqrt{N}+1)$ in $\mathbb{G}$	$6\sqrt{N}$ in $\mathbb{G}$, $\sqrt{N}$ in $\mathbb{G}_T$	public	D3DH, XDH
(Boneh and Zhandry, 2014)	–	$\mathsf{poly}(\log N,\eta)$	η	$\mathsf{poly}(\log N,\eta)$	public	$i\mathcal{O}$ & FE security
(Nishimaki et al., 2016)−I	–	$\mathsf{poly}(\eta)$	$\mathsf{poly}(n)$	$\mathsf{poly}(n,\lvert m\rvert)$	public	$i\mathcal{O}$ & FE security
(Nishimaki et al., 2016)−II	–	$\mathsf{poly}(\log n)$	$\mathsf{poly}(n)$	$\lvert m\rvert+\mathsf{poly}(\log n)$	public	$i\mathcal{O}$ security
(Garg et al., 2016)	composite, ML	$\mathsf{poly}(\log N)$	$\mathsf{poly}(\log N)$	$\mathsf{poly}(\log N)$	public	FE security
Ours	prime, ML	$\mathsf{poly}(\log N,\eta)$	1 in $\mathbb{G}_{\vec{\rho}}$	2 in $\mathbb{G}_{\vec{\rho}}$, 3η, $\log(N)$	public	DHDHE and $i\mathcal{O}$ security

$\lvert\mathsf{PP}\rvert$ = public parameter size, $\lvert\mathsf{sk}_u\rvert$ = user secret key size, $\lvert\mathsf{CT}\rvert$ = ciphertext size, TRA = traceability, BL = bilinear, ML = multilinear, FE = functional encryption, D3DH = Decision (modified) 3-party Diffie-Hellman, DHSD = Diffie-Hellman Subgroup Decision, BSD = Bilinear Subgroup Decision, XDH = External Diffie-Hellman, DHDHE = Decisional Hybrid Diffie-Hellman Exponent assumptions, $\mathbb{G}$ = Bilinear source group, $\mathbb{G}_T$ = Bilinear target group, $\mathbb{G}_{\vec{\rho}}$ = Multilinear intermediate group, n = arbitrary bit-length of user identity, $\lvert m\rvert$ = message-bit length, N = total number of users in the system and, η = security parameter.

original private keys. A traitor tracing system runs an efficient *tracing algorithm* that interacts with the pirate decoder considering it as a *black-box oracle* and outputs at least one identity of the traitors in the coalition who was involved to create the malicious program using his own private key. Pirate cable TV, set-top decoders, encrypted satellite radio, pirate decryption software posted on the Internet etc. are few examples of pirate decoder box.

A naive approach to address this problem is the following. For a system having N users, the broadcaster broadcasts N ciphertext under N different public keys whereby a legitimate user can decrypt the ciphertext corresponding to his own secret key. Consequently, given any pirate decoder, it is easy to pinpoint at least one traitor whose secret key is used to fabricate the pirate decoder. However, this solution is inefficient as the ciphertext size is linear in N. Although a PLKA system has the capability of fraud detection, it is not always possible to switch a general BE scheme into a tracing scheme. Designing a PLKA traitor tracing, with shorter size ciphertext, public parameter and the user secret key is a challenging task.

Related Work. Traitor tracing was formally introduced by (Chor et al., 1994), followed by a several works in different flavors (Kiayias and Yung, 2001; Boneh and Waters, 2006; Boneh et al., 2006; Garg et al., 2010; Boneh and Zhandry, 2014; Nishimaki et al., 2016; Garg et al., 2016).

In 2001, (Kiayias and Yung, 2001) proposed t-collusion resistant tracing mechanism with ciphertext size linear in t. A collusion of at most t-users are allowed to construct a pirate decoder in such system. The *first fully collusion resistant* PLKA with traitor tracing was proposed by (Boneh and Waters, 2006; Boneh et al., 2006) in composite order bilinear group with sublinear size parameters. Later, (Garg et al., 2010) developed a similar variant on prime order bilinar group setting. Depending on the tracing authority, traitor tracing systems fall into two categories – (a) *publicly traceable* that does not require any secret inputs except the public parameter in the tracing algorithm (Boneh and Waters, 2006; Garg et al., 2010; Boneh and Zhandry, 2014; Nishimaki et al., 2016; Garg et al., 2016), and (b) *secretly traceable* which uses a secret tracing key to identify rogue users (Boneh et al., 2006; Kiayias and Yung, 2001). In 2014, (Boneh and Zhandry, 2014) constructed a fully collusion resistant PLKA traitor tracing with public traceability utilizing the constrained pseudorandom functions (cPRF*s*) and indistinguishability obfuscation ($i\mathcal{O}$). All the aforementioned PLKA schemes use the *Hybrid Coloring* tracing approach of (Kiayias and Yung, 2001). Adopting $i\mathcal{O}$, (Nishimaki et al., 2016) exhibited that a PLKA traitor tracing is an immediate consequence of functional encryption (FE). In (Garg et al., 2016), a FE scheme is designed in *composite* order asymmetric multilinear group setting without $i\mathcal{O}$ and provides another indirect construction of traitor tracing. None of the schemes (Nishimaki et al., 2016; Garg et al., 2016) provide explicit construction of PLKA traitor tracing. As pointed out by (Garg et al., 2010), the communication, storage, and computational efficiency of *prime* order groups are much higher compared to that of *composite* order group. Our main focus in this work is to build a PLKA traitor tracing scheme over *prime* order multilinear groups (Coron et al., 2015; Gentry et al., 2015) achieving order-of-magnitude improvements in efficiency and storage without any security breach.

Our Contribution. We design a PLKA construction coupling pseudorandom function (PRF) of (Goldreich et al., 1986) with indistinguishability obfuscation ($i\mathcal{O}$) and adopting multilinear maps over *prime* order group. Note that several recent attacks have broken many assumptions on known multilinear maps (Coron et al., 2015; Gentry et al., 2015). Recently, (Gu, 2015) constructed a new variant of the multilinear maps which seemed to thwart known attacks. We skillfully integrate the tracing mechanism of (Kiayias and Yung, 2001) in our PLKA, yielding the *first fully collusion resistant* and *publicly*

Table 2: Comparative summary of computation and tracing time.

PLKA	Pairing	Exponentiation	Product	Running Time of Tracing Algorithm
(Boneh and Waters, 2006)	$3\sqrt{N}+4$ (bilinear)	$3N+(N+15)\sqrt{N}+4$	$4N+5\sqrt{N}+4$	$O(N^3)$
(Boneh et al., 2006)	$2\sqrt{N}+3$ (bilinear)	$2N+10\sqrt{N}+1$	$N+3\sqrt{N}+4$	$O(N^3)$
(Garg et al., 2010)	$\sqrt{N}+8$ (bilinear)	$3N+24\sqrt{N}$	$3\sqrt{N}+11$	$O(N^3)$
Ours	2 (multilinear)	$3N+8$	$2N+3$	$\mathsf{poly}((\log N)^2, \eta)$

N = total number of users in the system, η = security parameter.

traceable PLKA traitor tracing in key encapsulation framework over *prime* order multilinear group setting with tracing algorithm having *shorter* running time. We summarize below our main findings in this work:

• Our PLKA construction significantly reduces the parameter sizes as exhibited by Table 1. The public parameter size in our construction is *polylogarithmic* in N while the ciphertext size is *logarithmic* in N. Here, N is the total number of users in the system. More interestingly, user secret key is a single multilinear group element in our PLKA.

• We emphasize that our scheme is *adaptively chosen ciphertext attack* (CCA)-secure under the Decisional Hybrid Diffie-Hellman Exponent (DHDHE)-assumption in *standard security* model and relies on $\mathsf{i}O$ security. Note that recently $\mathsf{i}O$ is aggregately constructible from the puncturable secret key functional encryption (Kitagawa et al., 2018). Our tracing algorithm enables to trace the conspiracy of an arbitrary number of defrauders using the public parameter only. On a more positive note, we have shown that although we follow the tracing approach of (Kiayias and Yung, 2001), the run time of our tracing algorithm is $\mathsf{poly}((\log N)^2, \eta)$, where η is the security parameter. However, the running time of tracing algorithms is $O(N^3)$ for all the existing PLKA traitor tracing schemes based on *Hybrid Coloring* tracing mechanism of (Kiayias and Yung, 2001). In sum, we achieve a *publicly traceable* and *fully collusion resistant* traitor tracing scheme with *shorter* running time.

• The PLKA design of (Boneh and Waters, 2006; Boneh et al., 2006; Garg et al., 2010) uses bilinear maps while that of (Boneh and Zhandry, 2014) is constructed using the security of $\mathsf{i}O$ and cPRF*s* (Boneh and Waters, 2013). The work of (Nishimaki et al., 2016; Garg et al., 2016) are based on FE. Coupling $\mathsf{i}O$ with the one way function, (Nishimaki et al., 2016) constructed a FE scheme and furnished an idea to transform it into a traitor tracing scheme. They set up with the exponentially large identity space and embedded user's arbitrary information in their secret key. As a result, the user identity bit-length become arbitrarily large. As shown in Table 1, the size of ciphertext and the user secret key in their works grow with the identity bit-length which is arbitrarily large, and also the ciphertext size depends on the message-bit length. The size of the parameters in our PLKA construction are independent of identity bit-length as well as the message-bit length. Our PLKA has similar parameter sizes as that of the PLKA of (Boneh and Zhandry, 2014) which stance upon four cPRF*s* in generic forms showing only the input-output behavior. Additionally, the work of (Boneh and Zhandry, 2014) utilizes the multilinear map based cPRF of (Boneh and Waters, 2013) which are themselves based on multilinear maps that requires at least $O(\log N)$ symmetric multilinear pairing operations which are known to be very expensive. In contrast, we use only two PRF*s* of (Goldreich et al., 1986) which are efficient due to their inherent tree structures.

• Table 2 shows the computation comparison in terms of number of pairings, exponentiations, multiplications and run time of the tracing algorithm. We exclude (Garg et al., 2016; Nishimaki et al., 2016; Boneh and Zhandry, 2014) from Table 2 as suitable FE schemes and multiparty key exchange protocols are the primary requirements in these works rather than direct constructions for traitor tracing. To trace all the traitors, (Nishimaki et al., 2016) proposed an *oracle jump finding* (OJF) problem and showed that any PLKA is sufficient for traitor tracing employing OJF problem. However, to run the tracing algorithm, the works of (Nishimaki et al., 2016) requires the total number q of traitors belonging to the pirate decoder $\mathcal{D}$ as an extra input and run time of OJF algorithm is $\mathsf{poly}(\log N, q, \eta)$ which is faster than our PLKA construction. For the bounded collusion resistant schemes, q is publicly known. In many real life scenarios, the tracing algorithm is given *black-box* interactions with $\mathcal{D}$ and finding q at prior not always possible. Unlike this, our tracing algorithm does not require any prior knowledge of parameters like q and runs in $\mathsf{poly}((\log N)^2, \eta)$ time using only the public parameter as the inputs.

2 PRELIMINARY

Notation. Let, $[j] = \{1, \ldots, j\}$ be the set of all positive integers from 1 to j. Given any set S, $x \in_R S$ stands for x drawn uniformly at random from S. For a randomized algorithm RandA, $y \leftarrow \mathsf{RandA}(z)$ represents output by RandA on input z. The equivalence relation over a set is denoted by $\equiv$. A probabilistic polynomial time algorithm is denoted by PPT and η is the security parameter.

Definition 1. (Negligible Function) *A function $\Psi : \mathbb{N} \to \mathbb{R}$ is said to be negligible in N, if for every positive integer c there exists an integer N_c such that $|\Psi(N)| < \frac{1}{N^c}$ for all $N > N_c$.*

Definition 2. (Chernoff Bound) *Let, $\mathcal{X} = \sum_{i=1}^{n} \mathcal{X}_i$, where $\mathcal{X}_i$ independent random variables for $i = 1,\ldots,n$. Let $\mathcal{X}_i = 1$ with probability p_i, $\mathcal{X}_i = 0$ with probability $1-p_i$ and $\mu = E(\mathcal{X}) = \sum_{i=1}^{n} p_i$ is the expectation. Then, $Pr[|\mathcal{X}-\mu| \geq a] \leq 2e^{\frac{-2a^2}{n}}$, where $a = \mu\delta$ is an arbitrary constant and $0 < \delta < 1$.*

Definition 3. (Pseudorandom Function (PRF)**)** *A PRF (Blum and Micali, 1984) is a function denoted by $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, that can be computed by a deterministic polynomial time algorithm which on input a fixed but randomly chosen key $k \in \mathcal{K}$ and any point $x \in \mathcal{X}$, outputs $\mathsf{PRF}(k,x) \in \mathcal{Y}$ such that $\mathsf{PRF}(k,\cdot)$ is indistinguishable from a random function.*

Henceforth, $\mathsf{PRF}_k(\cdot)$ refers to $\mathsf{PRF}(k,\cdot)$ for a random key $k \in \mathcal{K}$.

Definition 4. (Indistinguishability Obfuscator) *A uniform probabilistic polynomial time machine $\mathsf{i}\mathcal{O}$ for a circuit class $\{\mathcal{C}_\eta\}$, with circuits of size at most η, is called an indistinguishability obfuscator ($\mathsf{i}\mathcal{O}$) (Kitagawa et al., 2018) if it amuses the following properties.*

• Functionality Preserving: *For all security parameters $\eta \in \mathbb{N}$, for all circuit $C \in \{\mathcal{C}_\eta\}$ and for all inputs x, $\mathsf{i}\mathcal{O}(\eta,C)$ preserves the functionality of the circuit C under the obfuscation, i.e., $Pr[\forall x, C'(x) = C(x) : C' \leftarrow \mathsf{i}\mathcal{O}(\eta,C)] = 1$.*

• Indistinguishability: *For all pairs of probabilistic polynomial time adversaries $\mathcal{A} = (\mathcal{D}_1,\mathcal{D}_2)$, there exists a negligible function $\zeta(\eta)$ such that, if $Pr[\forall x, C_0(x) = C_1(x) : (C_0,C_1,\sigma) \leftarrow \mathcal{D}_1(\eta)] > 1-\zeta(\eta)$ then $|Pr[\mathcal{D}_2(\sigma,\mathsf{i}\mathcal{O}(\eta,C_0)) = 1] - Pr[\mathcal{D}_2(\sigma,\mathsf{i}\mathcal{O}(\eta,C_1)) = 1]| < \zeta(\eta)$.*

Note that if no confusion arises, we will omit η as an input to $\mathsf{i}\mathcal{O}$ and as a subscript for $\mathcal{C}$.

2.1 Asymmetric Multilinear Map and Complexity Assumption

A (leveled) asymmetric multilinear map $\mathsf{a}\mathcal{MM} = (\mathsf{a}\mathcal{MM}.\mathsf{Setup}, e_{\vec{\vartheta}_1,\vec{\vartheta}_2})$ of (Coron et al., 2015; Gentry et al., 2015) consists of the following two algorithms.

• (aPPM)$\leftarrow$ $\mathsf{a}\mathcal{MM}.\mathsf{Setup}(1^\eta,\vec{\rho})$: It takes as input the security parameter 1^η and sets up $\vec{\rho}$-leveled linear map, where $\vec{\rho}$ is some positive vector of length $\kappa+1$. It outputs a description of all possible groups $\mathbb{G}_{\vec{\vartheta}}$ for all the vectors $\vec{\vartheta} \in (\mathbb{N}\cup\{0\})^{\kappa+1}$ with the restriction that $\vec{\vartheta} \leq \vec{\rho}$ (with component-wise comparison). For all such vectors $\vec{\vartheta}$, it outputs the canonical generators $g_{\vec{\vartheta}} \in \mathbb{G}_{\vec{\vartheta}}$. Let $\vec{e}_i$, $i = 0,\ldots,\kappa$ be the i-th standard basis vector, with 1 at position i and 0 elsewhere. Define $\mathbb{G}_{\vec{e}_i}$ as the i-th source group, $\mathbb{G}_{\vec{\rho}}$ as the target group, and rest of $\mathbb{G}_{\vec{\vartheta}}$ as the intermediate groups and all the groups have same large prime order $p > 2^\eta$. As there are uncountable numbers of such vectors, it is hard to publish all. Instead, one can publish a public parameter $\mathsf{aPPM} = (\kappa, g_{\vec{e}_0},\ldots,g_{\vec{e}_\kappa})$ consisting of only source groups' canonical generators.

• ($g^{ab}_{\vec{\vartheta}_1+\vec{\vartheta}_2}$)$\leftarrow$ $e_{\vec{\vartheta}_1,\vec{\vartheta}_2}(g^a_{\vec{\vartheta}_1}, g^b_{\vec{\vartheta}_2})$: On input elements $g^a_{\vec{\vartheta}_1} \in \mathbb{G}_{\vec{\vartheta}_1}$, $g^b_{\vec{\vartheta}_2} \in \mathbb{G}_{\vec{\vartheta}_2}$ with $\vec{\vartheta}_1 + \vec{\vartheta}_2 \leq \vec{\rho}$, $\vec{\vartheta}_1,\vec{\vartheta}_2 \in_R (\mathbb{N}\cup\{0\})^{\kappa+1}$, for all $a,b \in_R \mathbb{Z}_p$ and it outputs an element of $\mathbb{G}_{\vec{\vartheta}_1+\vec{\vartheta}_2}$ such that $e_{\vec{\vartheta}_1,\vec{\vartheta}_2}(g^a_{\vec{\vartheta}_1}, g^b_{\vec{\vartheta}_2}) = g^{ab}_{\vec{\vartheta}_1+\vec{\vartheta}_2}$.

Note that we often omit the subscripts and just write e. We can also generalize e to multiple inputs as $e(h^{(1)},h^{(2)},\ldots,h^{(\zeta)}) = e(h^{(1)},e(h^{(2)},\ldots,h^{(\zeta)}))$. The following assumption is from (Boneh et al., 2014).

> – It runs the algorithm $\mathsf{a}\mathcal{MM}.\mathsf{Setup}(1^\eta,2\vec{\rho})$ to generate $\mathsf{aPPM} = (\kappa, g_{\vec{e}_0},\ldots,g_{\vec{e}_\kappa})$ and e is the description of the multilinear map
>
> – It picks random t and ξ from $\mathbb{Z}_p$ and computes $V = g^t_{\vec{\rho}}$, $\Gamma_0 = (g_{\vec{e}_0})^\xi$, $\Gamma_1 = (g_{\vec{e}_1})^{\xi^2}$, ..., $\Gamma_{\kappa-1} = (g_{\vec{e}_{\kappa-1}})^{\xi^{2^{\kappa-1}}}$, $\Gamma_\kappa = (g_{\vec{e}_\kappa})^{\xi^{2^{\kappa+1}}}$
>
> – It sets $T_0 = (g_{2\vec{\rho}})^{t\xi^{2^\kappa}}$, $T_1 = R \in_R \mathbb{G}_{2\vec{\rho}}$
>
> – It returns $\chi_\mu = (e, \mathsf{aPPM}, \Gamma_0,\ldots, \Gamma_{\kappa-1},\Gamma_\kappa, V, T_\mu)$

Figure 1: κ-DHDHE instance generator $\mathcal{G}_\mu^{\kappa-\mathsf{DHDHE}}$.

κ-Decisional Hybrid Diffie-Hellman Exponent Assumption (κ-DHDHE). The κ-DHDHE problem is to guess $\mu \in \{0,1\}$ given $\chi_\mu = (e, \mathsf{aPPM}, \Gamma_0,\ldots, \Gamma_\kappa, V, T_\mu)$ generated by the generator $\mathcal{G}_\mu^{\kappa-\mathsf{DHDHE}}$ shown in Figure 1.

Definition 5. (κ-DHDHE **Assumption**) *The κ-DHDHE assumption is that $\mathsf{Adv}_{\mathcal{B}}^{\kappa-\mathsf{DHDHE}}(\eta)$ is at most negligible for all PPT algorithms $\mathcal{B}$.*

2.2 Hybrid Coloring

A *Hybrid Coloring* of the user population, introduced by (Kiayias and Yung, 2001), is a partition of the total number of users $[N]$ in a broadcast encryption (BE) system. A random ciphertext C_R induces a *Hybrid Coloring* over $[N]$ as follows.

• Let $\mathcal{D}$ be a pirate decoder (PD) box. We define an *equivalence relation* over the user secret key space as follows: $\forall\ u, u' \in [N]$, $\mathsf{pk}_u \equiv \mathsf{pk}_{u'}$ *iff* $Pr\left[\mathcal{D}(1^\eta, \mathsf{pk}_u, C_R) \neq \mathcal{D}(1^\eta, \mathsf{pk}_{u'}, C_R)\right] \leq \varepsilon$, *where* ε *is a negligible quantity and* pk_u *and* $\mathsf{pk}_{u'}$ *are the secret key of* u *and* u' *respectively.*

• Assume that C_m be a ciphertext corresponding to a valid message m. Then, with overwhelming high probability $\mathcal{D}(1^\eta, \mathsf{pk}_u, C_m) = \mathcal{D}(1^\eta, \mathsf{pk}_u, C_m)$ for all $u, u' \in [N]$. In that case, we get a unique equivalence class. Consequently, all the users will get the same color. Let Ciphr_R be the set of all random ciphertexts such that for all $C' \in \mathsf{Ciphr}_R$, C' induces a unique equivalence class. Then, the set of all valid ciphertexts constitute a subset of Ciphr_R.

• A BE scheme induces a *Hybrid Coloring* if there exist an algorithm that produces a ciphertext C such that C induces a partition over the user population.

One important observation regarding the tracing algorithm of (Kiayias and Yung, 2001) is formally stated by the following lemma.

Lemma 1. (Kiayias and Yung, 2001) *The tracing procedure using the* Hybrid Coloring *has time complexity* $O(N^3 \log^2 N)$ *and identify a traitor with high probability.*

3 OUR PLKA TRACING SCHEME

Our PLKA consists of three randomized algorithms PLKA.Setup, PLKA.Enc, PLKA.Dec and an external tracing algorithm $\mathsf{PLKA.Trace}^{\mathcal{D}}$ which are described below.

• $(\mathsf{plparams}, (\mathsf{plsk}_1, \ldots, \mathsf{plsk}_N)) \leftarrow \mathsf{PLKA.Setup}(\eta, \kappa)$: The group manager (GM) takes as input the length κ of the identities along with the security parameter η and proceeds as follows. The identity space is $\mathcal{ID} = \{0,1\}^\kappa \setminus \{0^\kappa\}$ and the total number of users the system can allow is $N = (2^\kappa - 1)$.

(i) The GM first constructs $\vec{\rho} = (1, \ldots, 1)$, a $(\kappa+1)$-length vector with all 1's, and runs the setup algorithm $\mathsf{a}\mathcal{MM}.\mathsf{Setup}(1^\eta, 2\vec{\rho})$ for the multilinear map described in section 2.1 to generate the public parameter $\mathsf{aPPM} = (\kappa, g_{\vec{e}_0}, \ldots, g_{\vec{e}_\kappa})$ where $g_{\vec{e}_i}$ is the canonical generator of the i-th source group $\mathbb{G}_{\vec{e}_i}$ for $0 \leq i \leq \kappa$ and $\mathbb{G}_{2\vec{\rho}}$ is the target group. All the groups have the same large prime order $p > 2^\eta$. It generates the canonical generators $g_{\vec{\rho}}$ and $g_{2\vec{\rho}}$ of the groups $\mathbb{G}_{\vec{\rho}}$ and $\mathbb{G}_{2\vec{\rho}}$ respectively by the repeated multilinear pairing operations using aPPM.

(ii) Two GGM tree (Goldreich et al., 1986) based secure pseudorandom functions $\mathsf{PRF}_{\mathsf{rand}} : \{0,1\}^{2\eta} \rightarrow \{0, \ldots, N\}$ and $\mathsf{PRF}_{\mathsf{auth}} : \{0,1\}^{2\eta} \times [N] \rightarrow \{0,1\}^\eta$ are selected by the GM where rand, auth are keys randomly chosen from the key space $\mathcal{K} = \{0,1\}^\eta$. It also picks $\mathsf{PRG} : \{0,1\}^\eta \rightarrow \{0,1\}^{2\eta}$, the length doubling pseudorandom generator (Blum and Micali, 1984).

(iii) The GM chooses $\xi, \tau \in_R \mathbb{Z}_p$, sets the programs $\mathsf{PT}_{\mathsf{Enc}}$ (Figure 2), $\mathsf{PT}_{\mathsf{Dec}}$ (Figure 3) and obfuscate these to generate obfuscated programs $\widetilde{\mathsf{PT}}_{\mathsf{Enc}} = \mathsf{i}\mathcal{O}(\mathsf{PT}_{\mathsf{Enc}})$, $\widetilde{\mathsf{PT}}_{\mathsf{Dec}} = \mathsf{i}\mathcal{O}(\mathsf{PT}_{\mathsf{Dec}})$ respectively using a secure indistinguishability obfuscator $\mathsf{i}\mathcal{O}$. The program $\mathsf{PT}_{\mathsf{Enc}}(j \in [N], t \in \mathbb{Z}_p, s \in \{0,1\}^\eta)$ has $(\mathsf{PRF}_{\mathsf{rand}}, \mathsf{PRF}_{\mathsf{auth}}, (\xi,\ \tau), \kappa, g_{\vec{\rho}}, g_{2\vec{\rho}})$ hard-coded in it and runs on input j, t, s to generate a header-session key pair $(\mathsf{Hdr} = (r \in \{0,1\}^{2\eta}, C_1 \in [N], C_2 \in \{0,1\}^\eta, C_3 \in \mathbb{G}_{\vec{\rho}}, C_4 \in \mathbb{G}_{\vec{\rho}}), K_{\mathsf{PLKA}} = (g_{2\vec{\rho}})^{t\xi^{2^\kappa}})$.

Inputs: $j \in [N], t \in \mathbb{Z}_p, s \in \{0,1\}^\eta$

Constants: $\mathsf{PRF}_{\mathsf{rand}}$, $\mathsf{PRF}_{\mathsf{auth}}$, (ξ, τ), κ, $g_{\vec{\rho}}$, $g_{2\vec{\rho}}$

1. Compute:
   (a) $r = \mathsf{PRG}(s)$
   (b) $C_1 = (\mathsf{PRF}_{\mathsf{rand}}(r) + j) \bmod (N+1)$
   (c) $C_2 = \mathsf{PRF}_{\mathsf{auth}}(r, C_1)$
   (d) $C_3 = (g_{\vec{\rho}})^t$ and $C_4 = (g_{\vec{\rho}})^{t\left\{\tau + \sum_{i=1}^{j} \xi^{2^\kappa - i}\right\}}$
2. Set: $K_{\mathsf{PLKA}} = (g_{2\vec{\rho}})^{t\xi^{2^\kappa}}$
3. Output: $(\mathsf{Hdr} = (r, C_1, C_2, C_3, C_4), K_{\mathsf{PLKA}})$

Figure 2: The program $\mathsf{PT}_{\mathsf{Enc}}$.

On the other hand, the program $\mathsf{PT}_{\mathsf{Dec}}(\mathsf{Hdr}, u \in [N], \mathsf{plsk}_u \in \mathbb{G}_{\vec{\rho}})$ has $\mathsf{PRF}_{\mathsf{rand}}, \mathsf{PRF}_{\mathsf{auth}}, (\xi, \tau), \kappa, g_{\vec{\rho}}, g_{2\vec{\rho}}$ hard-coded in it and runs on inputs Hdr, u, plsk_u to generate the correct session key K_{PLKA}. The obfuscated programs $\widetilde{\mathsf{PT}}_{\mathsf{Enc}}$ and $\widetilde{\mathsf{PT}}_{\mathsf{Dec}}$ behave in a similar manner as $\mathsf{PT}_{\mathsf{Enc}}$ and $\mathsf{PT}_{\mathsf{Dec}}$ respectively. That is, on the same input, $\mathsf{PT}_{\mathsf{Enc}}$ and $\widetilde{\mathsf{PT}}_{\mathsf{Enc}}$ generate the same output. Similarly, $\mathsf{PT}_{\mathsf{Dec}}$ and $\widetilde{\mathsf{PT}}_{\mathsf{Dec}}$ provide the same output on the same input. Note that in step $1(b)$ of $\mathsf{PT}_{\mathsf{Enc}}$, from the GGM tree based construction $\mathsf{PRF}_{\mathsf{rand}}(r)$ is an η-bit string which is converted to an integer and added to j modulo $(N+1)$ to generate header component C_2. Similarly, in step $1(a)$ of $\mathsf{PT}_{\mathsf{Dec}}$, to recover j from the header component C_1 we consider the integer representation of the η-bit string $\mathsf{PRF}_{\mathsf{rand}}(r)$.

(iv) The GM finally publishes the private linear public parameter plparams= $(\mathsf{PRF}_{\mathsf{rand}}$, $\mathsf{PRF}_{\mathsf{auth}}$,

Inputs: $\mathsf{Hdr} = (r \in \{0,1\}^{2\eta},\ C_1 \in [N],\ C_2 \in \{0,1\}^{\eta},\ C_3 \in \mathbb{G}_{\vec{\rho}},\ C_4 \in \mathbb{G}_{\vec{\rho}}),\ u \in [N],\ \mathsf{plsk}_u \in \mathbb{G}_{\vec{\rho}}$

Constants: $\mathsf{PRF}_{\mathsf{rand}}, \mathsf{PRF}_{\mathsf{auth}}, (\xi,\tau), \kappa, g_{\vec{\rho}}, g_{2\vec{\rho}}$

1. Compute:
   (a) $j = (C_1 - \mathsf{PRF}_{\mathsf{rand}}(r)) \bmod (N+1)$
   (b) $x = \mathsf{PRG}(\mathsf{PRF}_{\mathsf{auth}}(r, C_1))$
   (c) $y = (g_{\vec{\rho}})^{\tau\xi^u}$
2. Check that $(u \leq j) \wedge (x = \mathsf{PRG}(C_2)) \wedge (y = \mathsf{plsk}_u)$
   (a) If check fails, output $\perp$ and stop
   (b) Otherwise, compute:
      i. $\Lambda_{2^\kappa - i + u} = (g_{\vec{\rho}})^{\xi^{2^\kappa - i + u}}$ for all $i \in [j], i \neq u$ and $\Lambda_u = (g_{\vec{\rho}})^{\xi^u}$
      ii. $K_{\mathsf{PLKA}} = \dfrac{e(\Lambda_u, C_4)}{e\left((\mathsf{plsk}_u \cdot \prod_{\substack{i=1 \\ i\neq u}}^{j} \Lambda_{2^\kappa - i + u}), C_3\right)}$
3. Output: K_{PLKA}

Figure 3: The program $\mathsf{PT}_{\mathsf{Dec}}$.

$\mathsf{PRG}, \widetilde{\mathsf{PT}}_{\mathsf{Enc}}, \widetilde{\mathsf{PT}}_{\mathsf{Dec}})$. For each user $u \in [N]$, it computes the user secret key $\mathsf{plsk}_u = (g_{\vec{\rho}})^{\tau\xi^u}$ and sends plsk_u to user u through a secure communication channel between the GM and the user u.

• $(\mathsf{Hdr}, K_{\mathsf{PLKA}}) \leftarrow \mathsf{PLKA.Enc}(\mathsf{plparams}, j \in [N])$: On input an integer $j \in [N]$ and the public parameter $\mathsf{plparams}$, the encryptor executes the following steps.

(i) It chooses elements $t \in_R \mathbb{Z}_p$ and $s \in_R \{0,1\}^{\eta}$.

(ii) It generates $(\mathsf{Hdr} = (r, C_1, C_2, C_3, C_4), K_{\mathsf{PLKA}})$ by running the program $\widetilde{\mathsf{PT}}_{\mathsf{Enc}}$, extracted from $\mathsf{plparams}$, on input $(j \in [N],\ t \in \mathbb{Z}_p,\ s \in \{0,1\}^{\eta})$, where $\mathsf{Hdr} = (r, C_1, C_2, C_3, C_4)$ is the ciphertext header and K_{PLKA} is the session key for all the users in the set $[j]$.

(iii) Finally, it publishes Hdr as the ciphertext and keeps K_{PLKA} as secret to itself.

•$(K_{\mathsf{PLKA}} \vee \perp) \leftarrow \mathsf{PLKA.Dec}(\mathsf{plparams}, u \in [N], \mathsf{plsk}_u, \mathsf{Hdr} = (r, C_1, C_2, C_3, C_4))$: A user $u \in [N]$ uses secret key $\mathsf{plsk}_u = (g_{\vec{\rho}})^{\tau\xi^u}$ to recover the session key K_{PLKA} from the ciphertext header $\mathsf{Hdr} = (r, C_1, C_2, C_3, C_4)$ as follows.

(i) It runs the program $\widetilde{\mathsf{PT}}_{\mathsf{Dec}}$, extracted from $\mathsf{plparams}$, on input $(\mathsf{Hdr} = (r, C_1, C_2, C_3, C_4), u, \mathsf{plsk}_u)$.

(ii) If it passes all the checking conditions in step 2 of the program $\widetilde{\mathsf{PT}}_{\mathsf{Dec}} = i\mathcal{O}(\mathsf{PT}_{\mathsf{Dec}})$ in Figure 3, it gets the correct key K_{PLKA} as the output; otherwise gets $\perp$.

• $\mathbb{T}^{\mathsf{TTS}} \leftarrow \mathsf{PLKA.Trace}^{\mathcal{D}}(\mathsf{plparams}, \varepsilon)$: The tracer takes as input the public parameter $\mathsf{plparams}$, a parameter ε which is polynomially related to the security parameter η. It runs the $\mathsf{Trace}^{\mathcal{D}}$ program of Algorithm 1, on input the public parameter $\mathsf{plparams}$ and the parameter ε. It outputs the set of users $\mathbb{T}^{\mathsf{TTS}} \subseteq \{1, \ldots, N\}$ as the traitor users.

Algorithm 1: Traitor tracing program $\mathsf{Trace}^{\mathcal{D}}$.

```
Input: plparams, ε
for i = 0 to N do
    success ← 0
    for j = 1 to 2(log N / ε)^2 do
        (Hdr^(i), K_PLKA^(i)) ← PLKA.Enc(plparams, i)
        K_PLKA^(i') ← D(Hdr^(i))
        if K_PLKA^(i) = K_PLKA^(i') then
            success ← success + 1
        end if
    end for
    Y_i^obsrv ← success
end for
return T^TTS = { i : Y_i^obsrv − Y_{i−1}^obsrv ≥ 4(log N)^2 / ε }
```

Correctness and the proof of our tracing algorithm is shown in the Theorem 2.

Correctness. Let, $u, j \in [N]$ and $1 \leq u \leq j \leq N$. Let, $(\mathsf{plparams}, (\mathsf{plsk}_1, \ldots, \mathsf{plsk}_N)) \leftarrow \mathsf{PLKA.Setup}(\eta, \kappa)$, where $\mathsf{plparams} = (\mathsf{PRF}_{\mathsf{rand}}, \mathsf{PRF}_{\mathsf{auth}}, \mathsf{PRG}, \widetilde{\mathsf{PT}}_{\mathsf{Enc}}, \widetilde{\mathsf{PT}}_{\mathsf{Dec}})$ and $\mathsf{plsk}_u = (g_{\vec{\rho}})^{\tau\xi^u}$. Let $(\mathsf{Hdr}, K_{\mathsf{PLKA}} = (g_{2\vec{\rho}})^{t\xi^{2^\kappa}}) \leftarrow \mathsf{PLKA.Enc}(\mathsf{plparams}, j \in [N])$, where $\mathsf{Hdr} = (r, C_1, C_2, C_3, C_4)$ with

$$C_1 = (\mathsf{PRF}_{\mathsf{rand}}(r) + j) \bmod (N+1),\ C_3 = (g_{\vec{\rho}})^t,$$
$$C_2 = \mathsf{PRF}_{\mathsf{auth}}(r, C_1),\ C_4 = (g_{\vec{\rho}})^{t\left\{\tau + \sum_{i=1}^{j} \xi^{2^\kappa - i}\right\}}.$$

A user u, with its secret key $\mathsf{plsk}_u = (g_{\vec{\rho}})^{\tau\xi^u}$ runs $\mathsf{PLKA.Dec}(\mathsf{plparams}, u, \mathsf{plsk}_u, \mathsf{Hdr})$. If u passes all the conditions in step 2 of the program in Figure 3 in executing the program $\widetilde{\mathsf{PT}}_{\mathsf{Dec}}$ in $\mathsf{plparams}$, then we show below that u can recover the correct session key $K_{\mathsf{PLKA}} = (g_{2\vec{\rho}})^{t\xi^{2^\kappa}}$ by extracting C_3 and C_4 from Hdr and proceeding as follows.

As, $\Lambda_{2^\kappa - i + u} = (g_{\vec{\rho}})^{\xi^{2^\kappa - i + u}}$ and $\Lambda_u = (g_{\vec{\rho}})^{\xi^u}$ are given in $\mathsf{PT}_{\mathsf{Dec}}$, we have

$$e(\Lambda_u, C_4) \Big/ e\left(\mathsf{plsk}_u \cdot \prod_{\substack{i=1 \\ i \neq u}}^{j} \Lambda_{2^\kappa - i + u}, C_3\right)$$

$$= \frac{e\left((g_{\vec{\rho}})^{\xi^u}, (g_{\vec{\rho}})^{t\left\{\tau + \sum_{i=1}^{j} \xi^{2^\kappa - i}\right\}}\right)}{e\left((g_{\vec{\rho}})^{\tau\xi^u} \cdot \prod_{\substack{i=1 \\ i \neq u}}^{j} (g_{\vec{\rho}})^{\xi^{2^\kappa - i + u}}, (g_{\vec{\rho}})^t\right)}$$

$$= \frac{(g_{2\vec{\rho}})^{\xi^u t \sum_{i=1}^{j} \xi^{2^\kappa - i}}}{(g_{2\vec{\rho}})^{t \sum_{\substack{i=1 \\ i \neq u}}^{j} \xi^{2^\kappa - i + u}}} = (g_{2\vec{\rho}})^{t\xi^{2^\kappa}} = K_{\mathsf{PLKA}}$$

Remark 1. *As the set system $\mathcal{S} = \{[1], \ldots, [N]\}$ has only a polynomial number of recipient sets in it, according to (Boneh and Zhandry, 2014), the selective and the adaptive security are equivalent.*

4 SECURITY ANALYSIS

Theorem 1. (Security of Indistinguishability) *Assuming secure* $\mathsf{i}O$, *our* PLKA *scheme, presented in section 3, achieves adaptive* CCA*-security under the* κ-DHDHE *assumption.*

Proof. Due to limited space, proof is available in the full version (Mandal and Dutta, 2018).

Theorem 2. (Security of Traceability) *Suppose that our* PLKA *scheme, presented in section 3, is adaptive* CCA*-secure. Then, the publicly traceable* $\mathsf{PLKA.Trace}^{\mathcal{D}}$ *algorithm outputs identity of all the traitors.*

Proof. Assume that at the beginning the adversary $\mathcal{A}$ outputs a pirate decoder box $\mathcal{D}$. For $i = 0, \ldots, N$ construct the experiment TrExp_i of Figure 4 using the *Hybrid Coloring* mechanism shown in section 2.2. Let $p_i = \Pr[\mathcal{H}_i = \mathsf{success}]$ be the success probability in the above experiment TrExp_i for $i = 0, \ldots, N$. Clearly, $p_0 = 0$, whereas $p_N = 1$ and hence $|p_N - p_0| = 1$.

Consider that user $j \in [N]$ is not a traitor user. Then, the secret key plsk_j of user j is not embedded into the pirate decoder box $\mathcal{D}$. Note that if plsk_k is embedded into $\mathcal{D}$ for some $k < j$, then $\mathcal{H}_j = \mathcal{H}_k =$ success and consequently $|p_j - p_k| = 0$. On the other hand, if $j \in [N]$ is the least positive integer such that plsk_j is embedded into $\mathcal{D}$, then $\mathcal{H}_j =$ success but $\mathcal{H}_k =$ failure for $1 \leq k \leq j-1$. In this case, $|p_j - p_k| \geq \frac{1}{N}$. However, the adversary $\mathcal{A}$, who has formed the pirate decoder box $\mathcal{D}$, can not distinguish the ciphertext headers $\mathsf{Hdr}^{(j)}$ and $\mathsf{Hdr}^{(j-1)}$ without having the knowledge of plsk_j, even if $\mathcal{A}$ has the secret key plsk_k for $1 \leq k \leq j-1$. As a result, the difference between the success probability in the experiment TrExp_{j-1} and in the experiment TrExp_j is negligible in the total number of user N. Therefore, $|p_{j-1} - p_j|$ is negligible in N.

(i) The tracer generates header-session key pair $(\mathsf{Hdr}^{(i)}, K_{\mathsf{PLKA}}{}^{(i)}) \leftarrow \mathsf{PLKA.Enc}(\mathsf{plparams}, i)$, where plparams is the public parameter generated using PLKA.Setup algorithm of our PLKA scheme.
(ii) Then, tracer interacts with the pirate decoder $\mathcal{D}$, giving $\mathsf{Hdr}^{(i)}$ as an input to $\mathcal{D}$, and in return tracer will get $K^{(i')}_{\mathsf{PLKA}} \leftarrow \mathcal{D}(\mathsf{Hdr}^{(i)})$. Here, $\mathcal{D}$ acts as a *black-box* oracle for this interaction.
(iii) Finally, tracer sets the success or failure $\mathcal{H}_i$ as follows

$$\mathcal{H}_i = \begin{cases} \mathsf{success} & \text{if } K_{\mathsf{PLKA}}{}^{(i)} = K^{(i')}_{\mathsf{PLKA}} \\ \mathsf{failure} & \text{otherwise} \end{cases}$$

Figure 4: Tracing Experiment TrExp_i for $i = 0, \ldots, N$.

Since $|p_N - p_0| = 1$, by the triangular inequality there must exists at least one user $i_t \in [N]$ such that $|p_{i_t} - p_{i_t - 1}| \geq \frac{1}{N}$. So that the success probability difference between the two experiments TrExp_{i_t} and $\mathsf{TrExp}_{i_t - 1}$ is at least $\frac{1}{N}$ which is non-negligible. Let the advantage of breaking the indistinguishability security of our PLKA scheme is $\varepsilon = \mathsf{Adv}^{\mathsf{CCA-PLKA}}_{\mathcal{A}}(\eta)$. If $|p_{i_t} - p_{i_t - 1}| \geq \frac{1}{N} \geq \varepsilon$, then this indicate that plsk_{i_t} is embedded into $\mathcal{D}$ with probability at least ε and hence the user i_t must be a traitor. Observe that user $i_t - 1$ can not be a traitor. If both i_t and $i_t - 1$ are traitors, then $\mathcal{H}_{i_t} =$ success as well as $\mathcal{H}_{i_t - 1} =$ success, as $\mathcal{D}$ having $\mathsf{plsk}_{i_t - 1}$ can return correct session keys corresponding to both $\mathsf{Hdr}^{(i_t)}$ and $\mathsf{Hdr}^{(i_t - 1)}$. Note that $\mathcal{D}$ can decrypt the ciphertext header $\mathsf{Hdr}^{(j)}$ for any $j > i_t - 1$ if $\mathsf{plsk}_{i_t - 1}$ is embedded in $\mathcal{D}$.

To ensure perfectly that the user i_t is a traitor user, one has to to repeat the experiment TrExp_{i_t} more than a single time. Consider that for each $i = 0, \ldots, N$, the tracer repeats the experiment TrExp_i *independently* up to $\mathfrak{R}$ trials. We define a random variable $\mathcal{Y}_i$ as total number of success that were returned by $\mathcal{D}$ during $\mathfrak{R}$ trials of the experiment TrExp_i. If i_t is a traitor user, then for one trial $|p_{i_t} - p_{i_t - 1}| \geq \varepsilon$. Therefore, for $\mathfrak{R}$ trials the *expected* difference between the random variable $\mathcal{Y}_{i_t}$ and $\mathcal{Y}_{i_t-1}$ is at least $\varepsilon\Re$. To perfectly ensure that the user i_t is a traitor user, we have to make sure that the observed values of the random variables $\mathcal{Y}_k$, denoted by $\mathcal{Y}_k^{\mathsf{obsrv}}$, is sufficiently closed to their expected values $\mu_k = p_k\Re$ for $k = i_t, i_t - 1$. Using the *Chernoff bound*, we obtain the following relation between $\mathcal{Y}_k^{\mathsf{obsrv}}$ and its expected value $\mu_k = p_k\Re$ for $k = i_t, i_t - 1$, taking $\delta = \frac{1}{2}$, and setting $a = \frac{\varepsilon\Re}{2}$:

$$Pr\left[|\mathcal{Y}_k^{\mathsf{obsrv}} - \mu_k| \geq \frac{\varepsilon\Re}{2}\right] \leq 2(e)^{\frac{-\varepsilon^2\Re}{2}} = 2(N^{\frac{1}{\log N}})^{\frac{-\varepsilon^2\Re}{2}} \leq 2N^{-\log N}$$

if $\Re \geq 2(\frac{\log N}{\varepsilon})^2$. Observe that this probability is negligible in N using the Definition 1, as $\log N$ is an positive function.

Again from the *Chernoff bound*, we can write $\mu_k - \frac{\varepsilon\Re}{2} \geq \mathcal{Y}_k^{\mathsf{obsrv}} \geq \mu_k + \frac{\varepsilon\Re}{2}$. Hence, $\mathcal{Y}_k^{\mathsf{obsrv}} \geq \mu_k + \frac{\varepsilon\Re}{2}$ and $-\mathcal{Y}_k^{\mathsf{obsrv}} \geq -\mu_k + \frac{\varepsilon\Re}{2}$. If i_t is a traitor, then for i_t and $i_t - 1$, the difference between two observed values $\mathcal{Y}_{i_t}^{\mathsf{obsrv}}$ and $\mathcal{Y}_{i_t-1}^{\mathsf{obsrv}}$ (repeat each up to $\Re$ times) is given by

$$(\mathcal{Y}_{i_t}^{\mathsf{obsrv}} - \mathcal{Y}_{i_t-1}^{\mathsf{obsrv}}) \geq \mu_{i_t} + \frac{\varepsilon\Re}{2} - \mu_{i_t-1} + \frac{\varepsilon\Re}{2}$$
$$\geq \varepsilon\Re + (\mu_{i_t} - \mu_{i_t-1}) \geq \varepsilon\Re + (p_{i_t} - p_{i_t-1})\Re \geq 2\varepsilon\Re$$

Hence, for the traitor user i_t, the difference between $\mathcal{Y}_{i_t}^{\mathsf{obsrv}}$ and $\mathcal{Y}_{i_t-1}^{\mathsf{obsrv}}$ is at least $2\varepsilon\Re$, where $\Re \geq 2(\frac{\log N}{\varepsilon})^2$. The complete tracing mechanism is given in Algorithm 1.

5 CONCLUSION

In this work, we have designed an *adaptively* CCA-secure PLKA traitor tracing scheme, under the *prime* order multilinear group setting, which is *fully collusion resistance* and *publicly traceable*. Our construction is proven to be secure under the hardness of standard DHDHE-assumption. More precisely, our design significantly reduces the *parameter sizes* and the *tracing time* which are so far a plausible improvement in the literature.

REFERENCES

Blum, M. and Micali, S. (1984). How to generate cryptographically strong sequences of pseudorandom bits. *SIAM journal on Computing*, 13(4):850–864.

Boneh, D., Sahai, A., and Waters, B. (2006). Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Advances in Cryptology-EUROCRYPT 2006*, pages 573–592. Springer.

Boneh, D. and Waters, B. (2006). A fully collusion resistant broadcast, trace, and revoke system. In *Proceedings of the 13th ACM CCS*, pages 211–220. ACM.

Boneh, D. and Waters, B. (2013). Constrained pseudorandom functions and their applications. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 280–300. Springer.

Boneh, D., Waters, B., and Zhandry, M. (2014). Low overhead broadcast encryption from multilinear maps. In *Advances in Cryptology–CRYPTO 2014*, pages 206–223. Springer.

Boneh, D. and Zhandry, M. (2014). Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *Advances in Cryptology–CRYPTO 2014*, pages 480–499. Springer.

Chor, B., Fiat, A., and Naor, M. (1994). Tracing traitors. In *Annual International Cryptology Conference*, pages 257–270. Springer.

Coron, J.-S., Lepoint, T., and Tibouchi, M. (2015). New multilinear maps over the integers. In *Advances in Cryptology–CRYPTO 2015*, pages 267–286. Springer.

Garg, S., Gentry, C., Halevi, S., and Zhandry, M. (2016). Functional encryption without obfuscation. In *TCC*, pages 480–511. Springer.

Garg, S., Kumarasubramanian, A., Sahai, A., and Waters, B. (2010). Building efficient fully collusion-resilient traitor tracing and revocation schemes. In *Proceedings of the 17th ACM CCS*, pages 121–130. ACM.

Gentry, C., Gorbunov, S., and Halevi, S. (2015). Graph-induced multilinear maps from lattices. In *TCC*, pages 498–527. Springer.

Goldreich, O., Goldwasser, S., and Micali, S. (1986). How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807.

Gu, C. (2015). An improved multilinear map and its applications. *International Journal of Information Technology and Web Engineering (IJITWE)*, 10(3):64–81.

Kiayias, A. and Yung, M. (2001). On crafty pirates and foxy tracers. In *Security and Privacy in Digital Rights Management*, pages 22–39. Springer.

Kitagawa, F., Nishimaki, R., and Tanaka, K. (2018). Obfustopia built on secret-key functional encryption. *EUROCRYPT 2018*.

Mandal, M. and Dutta, R. (2018). Cost-effective private linear key agreement with adaptive CCA security from prime order multilinear maps and tracing traitors. Cryptology ePrint Archive, Report 2018/508. https://eprint.iacr.org/2018/508.

Nishimaki, R., Wichs, D., and Zhandry, M. (2016). Anonymous traitor tracing: How to embed arbitrary information in a key. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 388–419. Springer.