Distrib. Comput. (2005) 17: 323–347
Digital Object Identifier (DOI) 10.1007/s00446-004-0120-5

DISTRIBUTED COMPUTING

Scalable public-key tracing and revoking

Yevgeniy Dodis[1], **Nelly Fazio**[1], **Aggelos Kiayias**[2], **Moti Yung**[3]

[1] Computer Science Department, Courant Institute of Mathematical Sciences, New York University, New York, NY, USA (e-mail: {dodis,fazio}@cs.nyu.edu)
[2] Computer Science & Engineering Department, University of Connecticut, Storrs, CT, USA (e-mail: aggelos@cse.uconn.edu)
[3] Department of Computer Science, Columbia University, New York, NY, USA (e-mail: moti@cs.columbia.edu)

Received: August 28, 2003 / Accepted: October 13, 2004
Published online: January 13, 2005 – © Springer-Verlag 2005

Abstract. Traitor tracing schemes constitute a useful tool against piracy in the context of digital content distribution. They are encryption schemes that can be employed by content providers that wish to deliver content to an exclusive set of users. Each user holds a decryption key that is fingerprinted and bound to his identity. When a pirate decoder is discovered, it is possible to trace the identities of the users that contributed to its construction. In most settings, both the user population and the set of content providers are *dynamic*, thus scalable user management and scalable provider management are crucial. Previous work on public-key traitor tracing did not address the dynamic scenario thoroughly: no efficient scalable public-key traitor tracing scheme has been proposed, in which the populations of providers and users can change dynamically over time without incurring substantial penalty in terms of system performance and management complexity. To address these issues, we introduce a formal model for Scalable Public-Key Traitor Tracing, and present the first construction of such a scheme. Our model mandates for deterministic traitor tracing and unlimited number of efficient provider and user management operations. We present a formal adversarial model for our system and we prove our construction secure, against both adversaries that attempt to cheat the provider and user management mechanism, and adversaries that attempt to cheat the traitor tracing mechanism.

Keywords: Digital Content Distribution – Traitor Tracing – Scalability – Broadcast Encryption – Multicast

1 Introduction

1.1 Motivation

An important application of global networking is digital content distribution. In a typical scenario (e.g., Pay-TV) the entities that are active are the *content providers* and the *users* that subscribe to services and receive the content. In addition, the *security manager* is the entity in the system that manages providers and users and is responsible for enforcing various operational rules. For digital content distribution services to remain economically viable in the long run, it is important to design distribution schemes with certain basic properties: (1) strong content-protection – to ensure that only current subscribers have access to the distributed content; (2) traitor-traceability – to counter illegal content reception; (3) transmission efficiency – to optimize the bandwith utilization of the communication medium; and (4) scalability – to support many content providers and a large, dynamically changing population of subscribers. Next, we elaborate on these properties.

1.1.1 Content-protection

Exclusive reception of the digital content can be achieved by employing a multi-user encryption scheme in conjunction with a subscription-based model of service. In this setting only currently subscribed users are able to recover the content successfully. From a security viewpoint, the challenge here is similar to regular encryption systems: to make sure that at any given moment the active subscriber population can access the content whereas outsiders who eavesdrop on the communication medium are incapable of recovering the plaintext content.

1.1.2 Traitor-traceability

Even if an ideally secure content protection mechanism can be realized, it cannot prevent each subscriber from illicitly share its secret information with non-members. More generally, a group of subscribers (the *traitors*) can collude to construct an illegal decryption device (a *pirate decoder*), which can then be distributed on the black market. We would like to have a mechanism by which misbehaving pirates get caught. One effective such mechanism is provided by the notion of a *traitor tracing scheme*. A traitor tracing scheme is a multi-recipient encryption system that can be used for digital content distribution, with the property that the decryption key of each user is fingerprinted. When a pirate decoder is discovered, the security manager can employ a traitor tracing algorithm to uncover the identities of the traitors. The corresponding decryption keys could then be revoked, thus making the pirate decoder useless. Therefore, a traitor tracing scheme deters subscribers of a distribution system from leaking their keys, by the mere fact that the identity of the leaking entities (traitors) can be revealed. It is also a powerful tool against unauthorized access to the content since it allows the uncovering of compromised decryption keys and thus their removal from the system. Ideally, one would like an algorithm able to recover the traitors' identities by just probing the pirate decoder in a *black-box* fashion. However, the inherent hardness of this problem [19] poses limitations on the efficiency attainable with this approach, so that some traitor tracing mechanisms operate under the assumption that it is possible to extract the actual decryption key that enables the pirate decoder to unscramble the content (*non-black-box* traceability). We will consider both variants in this work.

1.1.3 Transmission efficiency

For efficiency reasons, the distribution scheme should send as few ciphertexts as possible while allowing all the legitimate receivers to recover the plaintext content. Also, the amount of users' storage and decryption time should be as small as possible. In an efficient system the above parameters must be independent of the total number of user management operations (user additions and removals) as well as of the total number of users.

1.1.4 Scalability

In the context of digital content distribution, scalability has two facets: *server-side* and *client-side*.

Server-side scalability deals with the property that allows the population of content providers to change dynamically. Each content provider in this setting needs access to the encryption mechanism (so that it can scramble content) and access to the distribution channel. Access to the distribution channel can be handled directly using access control mechanisms that are negotiated between the manager of the communication medium and the joining content provider, and we will not focus on this aspect in this work; note that in some cases the distribution channel may be a service entirely separated from the subscription service, e.g., when the Internet is used for distribution. On the other hand, access to the encryption mechanism suggests that each content provider needs to have the encryption keys that allow *all users* of the system to get its content. If the content providers are few and closely connected to the security manager, then one may assume that the encryption keys (and perhaps the decryption keys as well) are shared among the providers and the security manager. But this scenario does not scale, since when there are many providers, the amount of keys each user has to store can become prohibitively large (thus violating transmission efficiency property). Therefore, there is a need for providers to use the same key information. If symmetric key methods are being used, a corruption of one provider among the large (sub)-group of providers immediately compromises the content of all providers in that (sub)-group, violating the content protection property. Thus, due to the fact that we deal with a large set of providers and cannot trust all of them, we need an encryption mechanism whose security is not degraded when senders are compromised. This leads to the need of employing public-key cryptography for server-side scalability.

Client-side scalability deals with the fact that we have a user populaion that is changing dynamically due to the service subscription model and security constraints. To allow for a scalable management of user accounts, keys should be easy to generate and revoke. Adversaries that control some user keys that are revoked should be incapable of reading content. We obviously need mechanisms to identify misbehaving users to allow the piracy-deterrence property while the population is dynamically changing.

The goal of this work is to investigate systems that incorporate the above basic properties.

1.2 Related work

There are two primitives that are relevant to the present work: *broadcast encryption*, introduced by Fiat and Naor, [12] and *traitor tracing*, introduced by Chor et al. [6]. Broadcast encryption deals with the problem of transmitting protected content to a population of users so that an arbitrary (chosen) subset of them can be barred from each transmission. Traitor-tracing, on the other hand, deals with the problem of transmitting protected content to a population of users so that each decryption key is fingerprinted and bound to the identity of the user. In order to achieve a tracing and revoking functionality, traitor-tracing and broadcast encryption may be employed simultaneously as they can be coupled in the following producer-consumer pair: the traitor-tracing algorithm produces a set of corrupted users and the broadcast encryption algorithm receives this as input and bars these users from future transmissions.

Since we are interested in achieving both the tracing and the revoking functionalities simultaneously, in the remaining of this subsection we will consider traitor-tracing as a base property and will describe how previous work managed to couple revocation (from broadcast schemes) with traceability methods.

The original traitor tracing schemes of Chor et al. [6] employed a probabilistic design: each user possesses a different subset of a set of keys and tracing is achieved using the properties of the key assignment. These results were later implemented with concrete combinatorial designs by [27]. Both these schemes do not possess a "Remove-user" operation. This issue was later considered in [14, 17, 23], who investigated the combination of traitor tracing schemes with efficient revocation methods. These previous schemes did not consider a scalable, long-lived setting and were rather unsuitable for the public-key setting.

A different line of work that employed algebraic (rather than combinatorial) techniques in order to achieve traceability allowed for efficient public-key traitor tracing schemes in [3, 21] (the latter introduced a public-key scheme with deterministic traceability) and in [20]; these schemes did not consider revocation of keys. The combination of revocation and traceability functionalities was considered in the work of Naor and Pinkas [25], which described several schemes in the symmetric-key setting and a public-key scheme.

The work of [25] is the most relevant to our current investigation and motivated it. In particular, revocation method number 2 of [25] provides a tracing and revoking functionality that achieves client-side scalability but it is not a public-key system and thus it is not server-side scalable; in particular, distribution of content in these schemes relies on shared keys (i.e., a private-key encryption setting). In [25] a pubic-key variation is presented as well, nevertheless it is only for a limited number of revocations (proportional to the size of the public-key) and thus it is not cient-side scalable.

Public-key traitor tracing schemes with comparable revocation capabilities as the scheme in [25] (bounded number of revocations) were also designed in [28] and [8,9]. In all these schemes the bound on the number of revocations is proportional to the ciphertext size of the system. We remark that the scheme of [8] allows for an unlimited number of revocations, but does not have transmission efficiency as the size of ciphertexts grows with the number of revocations.

We note that client-side scalability was recognized as an important issue and was considered in the context of long-lived broadcast encryption in [15]. This scheme does not operate in a server-scalable environment and its main goal is analyzing the cost of rekeying in the course of the life time of the system. Finally, scalable revoking can also be achieved in the context of multicast refresh-key [5,29]. Note that these schemes, however, do not intend to provide traitor tracing functionalities.

Regarding the nature of traceability of the present work, the condition proven in [19] regarding black-box tracebility cannot be satisfied efficiently in our setting and thus our scheme cannot support black-box traceability in an efficient manner. This problem is inherent to all algebraic schemes like ours and those in [3,21,25]. Thus, the best we can hope for is either efficient non-black-box traceability or black-box confirmation (first proposed in [3]), which is an algorithm for the black-box model that may take time exponential in the number of traitors, but it is applicable to numerous practical scenarios. Our scheme efficiently supports both of them.

An interesting aspect is the issue of "statefulness" of the receivers in a digital content broadcast scheme. A receiver that maintains a state needs to be online and observe all the transmissions of the security manager. On the other hand, a stateless receiver may arbitrarily go off-line and then reconnect to the system without having to synchronize any information with previous transmissions. Statefulness affects the revocation capability of a scheme. Previous work on revocation and traceability was consistent with the stateless receiver scenario [8,9,17,18,23,25,28]. Nevertheless, efficient long-lived system seems to require some degree of statefulness, cf. [14]. The present work takes a hybrid approach similar to that of revocation method 2 of [25] (which is in the private-key setting): receivers do not maintain state across various system operations, nevertheless, the system changes phases or periods across which a receiver should maintain state.

As a final note, we remark that scalable traitor tracing and revoking addresses a somewhat different problem than dynamic traitor tracing (introduced in [13]). This latter concept deals with a specific functionality of the traitor tracing procedure (called "dynamic"), and does not deal with dynamically changing provider and user populations (despite the name). In particular, dynamic traitor tracing deals with the setting of pirate rebroadcast and the capability to trace pirates by observing the rebroadcasted data – an interesting scenario that requires robust watermarking techniques, but that we do not consider in this work.

1.3 Our results

Given the state of the art, we notice a lack of models and systems where all the properties that constitute a scalable public-key traitor tracing scheme are achieved simultaneously. The study of such a scheme is the undertaking of the present work and our results and approach are outlined in this subsection. An earlier version of this work appeared in [10]; the present work presents detailed modeling of the primitive suggested in this earlier paper, contains all proofs and corrects a flaw in the original construction.

In this work, we introduce the first model of a scalable public-key traitor tracing scheme where an unlimited number of users can be added and removed efficiently from the system. Being based on public-key techniques, the scheme supports any number of content providers broadcasting over the same infrastructure. Based on the DDH assumption, we then present a concrete scheme meeting these requirements, while preserving the confidentiality of the broadcast from revoked users in the adaptive chosen-plaintext sense. Addition of new users does not affect the keys of users already in the system. Unlimited number of user removals is achieved by dividing the lifetime of the system into *periods*: within a period, a bounded number of user removals can be executed. When an *a priori* specified threshold is reached, a fresh period is started with a New-period operation. For efficiency, such operation does not require private channels between the system manager and the users, and its complexity is independent of the number of users in the system. Within a period, users are not required to maintain state and are stateless. With every New-period operation each user needs to update its private-key information by employing an efficient key-update operation that depends on a security parameter.

The renewal of periods is influenced by the "proactive security model" of Ostrovsky and Yung [26], where information is updated by the manager. Unlike the proactive model, though, each period has a different key, which is reminiscent of the "key insulated" model of security of Dodis et al. [11].

In a scalable scheme, adversaries can introduce adversarially-controlled users in the system, they can observe the modifications to the global public key that occur during the runtime operation of the scheme and potentially take advantage of them. We consider two types of adversaries, the ones that attempt to defeat the content protection and user-management mechanism of the system and the ones that try to elude the traceability capability. Since the adversarial goal is distinct in these two cases, we consider the following classification of the two adversaries:

- *Window adversary*: the adversary obtains the encryption-key as well as some secret keys that are subsequently revoked; the adversary remains active and observes the revocation of other users of the system (in fact we allow the adversary to adaptively select which users should be revoked an unbounded number of times). Moreover, the adversary is allowed to control arbitrarily many content providers and select the content that is scrambled by the system, except for the challenge broadcast. We show that our construction is secure against window adversaries as long as they are fully revoked in a "window" of the system's operation that has a certain length (specified as a system parameter).

- *Traceability adversary*: the adversary, as before, obtains some secret keys and constructs a pirate decryption device, employing the secret user-key information (we allow the adversary to adaptively select the identities of the traitors). We show that our construction is secure against this type of adversaries in the non-black-box traitor tracing model. Our traitor tracing algorithm is *deterministic* and recovers the identities of *all* traitors (i.e., those who contributed to the pirate-key construction). Furthermore, our scheme supports the black-box confirmation method of [3], that even allows a form of traceability in the black-box traitor tracing model.

The advantage of our scalable public-key traitor tracing scheme over previous results comes from the fact that any adversary against the content protection mechanism that is fully revoked in the specified window of the system's operation will, in fact, "expire." An expired adversary will be incapable of intercepting the scrambled content (in the semantic security sense) even if it remains active in the system after being revoked. It is the capability of our scheme to expire adversaries that allows for the enhanced functionality of an unlimited number of revocations. None of the previous public-key traitor tracing schemes with revocation capability [8,9,25,28] possessed this crucial property. Indeed, in all previous public-key schemes, if an adversary, after being revoked, could continue to observe the system operations and cause more user revocations, then she would be able to "revive" her revoked key information and use it to intercept the scrambled content again.

2 The scalable public-key tracing and revoking model

In the scalable public-key traitor tracing model, the lifetime of the system is divided into periods. A period is an administrative unit managed based on the system activity and (possibly) on time passing.

A scalable scheme is comprised of the following basic procedures:

- Setup. An initialization procedure that is executed by the security manager. It takes as input a security parameter 1^k and a *saturation limit* 1^v that is an upper bound to the number of users that can be removed within a period. It generates a master secret key MSK along with a public key PK. The security manager keeps MSK secret and publishes PK.
- Add-user. It is a key-generation procedure executed by the security manager. It takes as input the master secret key MSK and the identity i of the new user, and results in a personalized secret key SK_i which is securely communicated to user i.
- Encryption. A public encryption algorithm $\mathcal{E}$ that takes as input the public key PK and a plaintext M, and outputs a ciphertext C, to be distributed to the user population through an insecure broadcast channel.
- Decryption. A deterministic algorithm $\mathcal{D}$ that takes as input the secret key SK_i of some user i and a ciphertext C, and outputs the corresponding plaintext or $\perp$.
- Remove-user. A procedure that given a public key PK, the identity of a user i and the corresponding secret key SK_i, results in a public key PK' so that, for all messages $M, \mathcal{E}(PK', M)$ should be "incomprehensible" for the user holding the revoked secret key SK_i, while non-removed users should be capable of decrypting it. If the saturation limit has been reached, then a New-period operation has to be executed before removing user i.
- New-period. A procedure executed by the security manager to initiate a fresh period when the saturation limit is reached (a *reactive change*), or when a certain time-limit is reached (a *proactive change*). It takes as input the master secret key MSK and the current public key PK, and results in a new public key PK' and a special reset message to be transmitted over an authenticated but otherwise insecure broadcast channel. Active subscribers can interpret such reset message and update their secret key accordingly; users removed in previous periods, instead, are prevented from doing do by the security properties of the scheme (cf. Sect. 5).
- Tracing. Given the master secret key MSK, the current public key PK and access (either *black-box* or *non-black-box*, cf. Sect. 6) to a pirate decoder, this procedure identifies (at least) one of the traitor users whose keys were employed to construct the pirate decoder.

2.1 Scalability objectives

A scalable scheme should satisfy the following requirements:

- Efficient addition of unlimited number of users throughout the scheme's lifetime. Specifically, the Add-user operation should have (i) communication independent of the size of the user population, and (ii) it should not involve the existing users of the system in any way.
- Efficient revocation of the decryption capabilities of a set of users within a period, provided that the number of users to be removed is below the saturation limit. Specifically, Remove-user should have time complexity independent of the total number of active users in the system, and should only affect the public key of the system.
- Efficient introduction of a new period. The communication overhead for changing a period should be independent of the total number of active users in the system and it should not require private communication channels between the security manager and the active users (but contrary to Remove-user it will require from users to modify their secret keys – as a result, in our model users are stateless within a period and stateful across periods).
- Efficient traitor tracing of a pirate decoder. Specifically, the tracing procedure should be polynomial-time in the number of users and the number of traitors.

2.2 Formal modeling of scalable schemes

A scalable public-key traitor tracing scheme should provide two basic functionalities: on one hand, the system should be capable of revoking the decryption capabilities of "bad" users; on the other hand, it should be capable of identifying users that participate in the construction of pirate decoders. We formally model the security of revocation and tracing in Sect. 5 and Sect. 6, respectively.

3 Preliminaries

Throughout the paper, k will denote a security parameter; let q be a k-bit prime number and let $\mathcal{G}$ be a cyclic group of order q. We assume that $\mathcal{G}$ is the (multiplicative) subgroup of order q of $\mathbb{Z}_p^*$, where $q \mid (p-1)$ and p is a large prime. Alternatively, one can take as group $\mathcal{G}$ the (additive) group of points of an elliptic curve over a finite field.

Definition 1. *Consider the uniform distribution over the following sets:*

$$R \doteq \{\langle g, g', u, u' \rangle \mid g, g', u, u' \in \mathcal{G}\}$$
$$D \doteq \{\langle g, g', u, u' \rangle \mid g, g', u, u' \in \mathcal{G}, \log_g u = \log_{g'} u'\}.$$

For every 0/1-valued probabilistic polynomial-time algorithm $\mathcal{A}$ and for all $k \in \mathbb{Z}_{\geq 0}$, define the DDH *advantage of $\mathcal{A}$ against $\mathcal{G}$ at k as:*

$$\mathsf{AdvDDH}_{\mathcal{G},\mathcal{A}}(k) \doteq \Big| \Pr[\tau = 1 \mid \rho \overset{R}{\leftarrow} R; \tau \overset{R}{\leftarrow} \mathcal{A}(1^k, \rho)] - \Pr[\tau = 1 \mid \rho \overset{R}{\leftarrow} D; \tau \overset{R}{\leftarrow} \mathcal{A}(1^k, \rho)] \Big|$$

where the probability is over the random coins of $\mathcal{A}$ and the random choice of ρ from R and D, respectively.

Definition 2. *Let $\mathsf{AdvDDH}_{\mathcal{G}}(k) \doteq max_{\mathcal{A}}\mathsf{AdvDDH}_{\mathcal{G},\mathcal{A}}(k)$, where the max is over all probabilistic, polynomial-time 0/1-valued algorithms $\mathcal{A}$.*

Assumption 1 (Decisional Diffie-Hellman Assumption) *The Decisional Diffie-Hellman (*DDH*) assumption for $\mathcal{G}$ asserts that the function $\mathsf{AdvDDH}_{\mathcal{G}}(k)$ is negligible in k.*

In the following, we will also need a (weaker) assumption about the hardness of computing discrete logarithms in $\mathcal{G}$.

Definition 3. *For every probabilistic polynomial-time algorithm $\mathcal{A}$ and for all $k \in \mathbb{Z}_{\geq 0}$, define the* DLog *advantage of $\mathcal{A}$ against $\mathcal{G}$ at k as:*

$$\mathsf{AdvDLog}_{\mathcal{G},\mathcal{A}}(k) \doteq \Pr[w' = w \mid g, g' \overset{R}{\leftarrow} \mathcal{G}; w \leftarrow \log_g g'; w' \leftarrow \mathcal{A}(1^k, g, g')]$$

where the probability is over the random coins of $\mathcal{A}$ and the random choice of g, g' from $\mathcal{G}$.

Definition 4. *Let $\mathsf{AdvDLog}_{\mathcal{G}}(k) \doteq max_{\mathcal{A}}\mathsf{AdvDLog}_{\mathcal{G},\mathcal{A}}(k)$, where the max is over all probabilistic, polynomial-time algorithms $\mathcal{A}$.*

Assumption 2 (Discrete Logarithm Assumption) *The Discrete Logarithm (*DLog*) assumption for $\mathcal{G}$ asserts that the function $\mathsf{AdvDLog}_{\mathcal{G}}(k)$ is negligible in k.*

3.1 Discrete logarithm representations

Let g be a generator of $\mathcal{G}$ and let $h_0, h_1, \ldots, h_v$ be elements of $\mathcal{G}$ such that

$$h_j = g^{r_j}$$

with $j = 0, \ldots, v$ and $r_0, \ldots, r_v \in \mathbb{Z}_q$. For a certain element $y \doteq g^b$ of $\mathcal{G}$, a representation of y with respect to the base $h_0, \ldots, h_v$ is a $(v+1)$-vector

$$\boldsymbol{\delta} \doteq \langle \delta_0, \ldots, \delta_v \rangle$$

such that:

$$y = \prod_{\ell=1}^{v} h_\ell^{\delta_\ell}$$

or equivalently $\boldsymbol{\delta} \cdot \boldsymbol{r} = b$ where "$\cdot$" denotes the inner product of two vectors modulo q.

It is well known (e.g., see [4]) that obtaining representations of a given y with respect to some base $h_0, \ldots, h_v$ is as hard as the discrete logarithm problem over $\mathcal{G}$. Furthermore, it was shown in Lemma 3.2 of [3] that if some adversary is given $m < v$ random representations of some y with respect to some base, then any additional representation that can be obtained has to be a "convex combination" of the given representations (a convex combination of the vectors $\boldsymbol{\delta_1}, \ldots, \boldsymbol{\delta_m}$ is a vector $\sum_{\ell=1}^{m} \mu_\ell \boldsymbol{\delta_\ell}$ with $\sum_{\ell=1}^{m} \mu_\ell = 1$). However, our scheme makes use of a particular family of discrete logarithm representations, introduced below. In Sect. 6 we will see how Lemma 3.2 of [3] can be modified accordingly.

3.2 Leap-vectors

We introduce a new family of discrete logarithm representations, called *leap-vectors*. In what follows, $\mathbb{Z}_q^v[x]$ denotes the set of v-degree polynomials over $\mathbb{Z}_q$ and $\mathbb{Z}_q^{<v}[x]$ denotes the ring of polynomials over $\mathbb{Z}_q$ with degree less than v.

Definition 5. *Given $z_1, \ldots, z_v \in \mathbb{Z}_q$ and $P(x) \in \mathbb{Z}_q^v[x]$, the set $\mathcal{L}_{z_1,\ldots,z_v}^P$ of leap-vectors with respect to $P(\cdot)$ and the values $z_1, \ldots, z_v$, consists of all vectors $\boldsymbol{\alpha} \in \mathbb{Z}_q^{v+1}$ for which it holds that:*

$$P(0) = \boldsymbol{\alpha} \cdot \langle 1, P(z_1), \ldots, P(z_v) \rangle. \qquad (1)$$

In other words, a leap-vector with respect to $P(\cdot)$ and $z_1, \ldots, z_v$, is a representation of $g^{P(0)}$ with respect to the base

$$g, g^{P(z_1)}, \ldots, g^{P(z_v)}.$$

Given any leap-vector $\boldsymbol{\alpha} := \langle \alpha_0, \ldots, \alpha_v \rangle$ with respect to some values $z_1, \ldots, z_v$, it is possible to derive the equation

$$\alpha_0 = \left(1 - \sum_{\ell=1}^{v} \alpha_\ell\right) a_0 + \sum_{j=1}^{v} \left(\sum_{\ell=1}^{v} z_\ell^j \alpha_\ell\right) a_j$$

over the coefficients of the polynomial

$$P(x) := a_0 + a_1 x + \ldots + a_v x^v.$$

If one possesses a point $\langle x_i, P(x_i) \rangle$ of the polynomial $P(\cdot)$, it is possible to generate a leap-vector for the values $z_1, \ldots, z_v$ (provided that $x_i \notin \{z_1, \ldots, z_v\}$) using Lagrange interpolation.

Definition 6. *Given distinct $x_i, z_1, \ldots, z_v \in \mathbb{Z}_q$, and $P(\cdot) \in \mathbb{Z}_q^v[x]$, define the leap-vector $\boldsymbol{\nu}_{z_1,\ldots,z_v}^{x_i,P}$ associated to the point $\langle x_i, P(x_i) \rangle$ with respect to $P(\cdot)$ and $z_1, \ldots, z_v$ as:*

$$\boldsymbol{\nu}_{z_1,\ldots,z_v}^{x_i,P} \doteq \langle \lambda_0^{(i)} P(x_i), \lambda_1^{(i)}, \ldots, \lambda_v^{(i)} \rangle \qquad (2)$$

where

$$\lambda_0^{(i)} \doteq \prod_{j=1}^{v} \frac{x_i}{x_i - z_j} \tag{3}$$

and, for $\ell = 1, \dots, v$,

$$\lambda_\ell^{(i)} \doteq \frac{z_\ell}{z_\ell - x_i} \prod_{\substack{j=1 \\ j \neq \ell}}^{v} \frac{z_\ell}{z_\ell - z_j}. \tag{4}$$

An important property of leap-vectors is the following:

Proposition 1. *Given a polynomial $P(\cdot) \in \mathbb{Z}_q^v[x]$ and the values $z_1, \dots, z_v \in \mathbb{Z}_q$, knowledge of a leap-vector $\boldsymbol{\alpha} \in \mathcal{L}^P_{z_1,\dots,z_v}$ implies knowledge of a linear equation on the coefficients of $P(\cdot)$, linearly independent from the linear equations defined using $\langle z_1, P(z_1)\rangle, \dots, \langle z_v, P(z_v)\rangle$.*

Proof. Define

$$\boldsymbol{\pi} \doteq (P(z_1), P(z_2), \dots, P(z_v), \alpha_0)^T.$$

The constraint on the coefficients $a_0, a_1, \dots, a_v$ of the polynomial $P(\cdot)$ arising from points $\langle z_1, P(z_1)\rangle, \dots, \langle z_v, P(z_v)\rangle$ and the equation associated to the leap-vector $\boldsymbol{\alpha}$, can be represented as:

$$\boldsymbol{\pi} = \mathbf{M} \cdot \boldsymbol{a}$$

where

$$\boldsymbol{a} \doteq (a_0, a_1, \dots, a_v)^T$$

and

$$\mathbf{M} \doteq \begin{pmatrix} 1 & z_1 & \dots & z_1^v \\ 1 & z_2 & \dots & z_2^v \\ \vdots & \vdots & \vdots & \vdots \\ 1 & z_v & \dots & z_v^v \\ 1 - \sum_{j=1}^{v} \alpha_j & -\sum_{j=1}^{v} \alpha_j z_j & \dots & -\sum_{j=1}^{v} \alpha_j z_j^v \end{pmatrix}$$

Notice that matrix **M** above is obtained from a Vandermonde matrix by adding a linear combination of the first v rows to the last one. Since every Vandermonde matrix has full rank, it follows that **M** has full rank, too. Hence, the equation defined by the leap-vector $\boldsymbol{\alpha}$ is linearly independent to the equations defined by the points $\langle z_1, P(z_1)\rangle, \dots, \langle z_v, P(z_v)\rangle$. □

As a result, the possession of a leap-vector implies some knowledge about the polynomial $P(\cdot)$ *beyond* what is implied by the points $\langle z_1, P(z_1)\rangle, \dots, \langle z_v, P(z_v)\rangle$. In other words, a leap-vector is the necessary information needed to *leap* from the values $P(z_1), \dots, P(z_v)$ to the value $P(0)$.

4 The scheme

Here we present our scheme and show that it is a correct public-key system (i.e., that the public-key information allows anyone to encrypt and that a holder of one of the private-key representations can apply the decryption algorithm to recover the plaintext).

Setup. The description of a cyclic multiplicative group $\mathcal{G}$ of order q is generated. Then, two random generators $g, g' \in \mathcal{G}$ and two random polynomials $A(\cdot), B(\cdot) \in \mathbb{Z}_q^v[x]$ are selected. The parameter v will be also referred to as the *saturation limit*, whereas $m = \lfloor \frac{v}{2} \rfloor$ will be the *maximum traitor collusion size*. Define

$$A(x) := a_0 + a_1 x + \dots + a_v x^v$$
$$B(x) := b_0 + b_1 x + \dots + b_v x^v.$$

The master secret key is

$$MSK := (A(\cdot), B(\cdot))$$

and the system's public key is

$$PK := \langle g, g', g^{A(0)} g'^{B(0)}, \langle \ell, g^{A(\ell)} g'^{B(\ell)} \rangle_{\ell=1}^{v} \rangle$$

where indices $1, \dots, v$ are used as place-holders. The security manager initiates a new period by publishing PK, and setting the *saturation level* L to 0. L is a system variable known to the security manager.

Add-user. When a new user i requests to join the system, the security manager transmits (over a private channel) the tuple $SK_i := \langle x_i, A(x_i), B(x_i) \rangle$ to user i, where

$$x_i \overset{R}{\leftarrow} \mathbb{Z}_q \quad x_i \notin \{1, \dots, v\} \cup \mathcal{U}.$$

The set $\mathcal{U}$ is the user-registry containing all values x_i that were selected in previous executions of the Add-user procedure. Subsequently, the security manager records the value x_i as associated to user i and adds x_i to $\mathcal{U}$.

Encryption. The sender obtains the current public key of the system

$$PK := \langle g, g', y, \langle z_1, h_1 \rangle, \dots, \langle z_v, h_v \rangle \rangle$$

(where $y = g^{A(0)} g'^{B(0)}$ and $h_\ell = g^{A(z_\ell)} g'^{B(z_\ell)}$, for some identity z_ℓ, $\ell = 1, \dots, v$) and then employs the encryption function $\mathcal{E}$ that, given the public key PK and a plaintext $M \in \mathcal{G}$, selects a random $r \overset{R}{\leftarrow} \mathbb{Z}_q$ and sets the corresponding ciphertext to be:

$$\psi \doteq \langle g^r, g'^r, y^r M, \langle z_1, h_1^r \rangle, \dots, \langle z_v, h_v^r \rangle \rangle.$$

Decryption. The decryption algorithm $\mathcal{D}$ takes as input a secret key $SK_i = \langle x_i, A(x_i), B(x_i) \rangle$ and a ciphertext

$$\psi = \langle u, u', u'', \langle z_1, u_1 \rangle, \dots, \langle z_v, u_v \rangle \rangle.$$

$\mathcal{D}$ first computes the leap-vectors

$$\boldsymbol{\nu}_{A,i} \doteq \boldsymbol{\nu}^{x_i, A}_{z_1,\dots,z_v} \quad \boldsymbol{\nu}_{B,i} \doteq \boldsymbol{\nu}^{x_i, B}_{z_1,\dots,z_v}$$

associated to the points $\langle x_i, A(x_i) \rangle$ and $\langle x_i, B(x_i) \rangle$ with respect to the values $z_1, \dots, z_v$. Observe that, by Definition 6 (Eqs. (2) and (4)), $\boldsymbol{\nu}_{A,i}$ and $\boldsymbol{\nu}_{B,i}$ agree on all components except for the first: denoting with $(\nu_{A,i})_\ell$ (respectively $(\nu_{B,i})_\ell$) the entry in $\boldsymbol{\nu}_{A,i}$ (respectively $\boldsymbol{\nu}_{B,i}$) indexed by ℓ, it holds that $\nu_{i,\ell} \doteq (\nu_{A,i})_\ell = (\nu_{B,i})_\ell$, for $\ell = 1, \dots, v$.

The decryption algorithm returns:

$$\mathcal{D}(\psi) \doteq \frac{u''}{u^{(\nu_{A,i})_0} u'^{(\nu_{B,i})_0} \prod_{\ell=1}^{v} u_\ell^{\nu_{i,\ell}}}$$

If ψ is a properly formed ciphertext, i.e.

$$\psi = \langle g^r, g'^r, y^r M, \langle z_1, h_1^r\rangle, \ldots, \langle z_v, h_v^r\rangle\rangle$$

then, due to the properties of the leap-vector representation (Eq. (1)), we have:

$$\begin{aligned}\mathcal{D}(\psi) &= \frac{g^{rA(0)} g'^{rB(0)} M}{g^{r(\nu_{A,i})_0} g'^{r(\nu_{B,i})_0} \prod_{\ell=1}^{v} g^{r\nu_{i,\ell}A(z_\ell)} g'^{r\nu_{i,\ell}B(z_\ell)}}\\ &= M\end{aligned}$$

Remove-user. Let $i_1, \ldots, i_k$ be the identities of the users to be removed, so that $L + k \le v$. Suppose that the current public key is $PK = \langle g, g', y, \langle z_1, h_1\rangle, \ldots, \langle z_v, h_v\rangle\rangle$. The revocation procedure uses the user-registry $\mathcal{U}$ to retrieve the values $x_{i_1}, \ldots, x_{i_k}$ and modifies the current public key PK as:

$$\begin{aligned}PK :=&\langle g, g', y, \langle z_1, h_1\rangle, \ldots, \langle z_L, h_L\rangle,\\ &\langle x_{i_1}, g^{A(x_{i_1})} g'^{B(x_{i_1})}\rangle, \ldots, \langle x_{i_k}, g^{A(x_{i_k})} g'^{B(x_{i_k})}\rangle,\\ &\langle z_{L+k+1}, h_{L+k+1}\rangle, \ldots, \langle z_v, h_v\rangle\rangle.\end{aligned}$$

Finally, the saturation level is increased to $L := L + k$.

New-period. When a Remove-user operation is invoked such that the resulting saturation level L would "overflow" the saturation limit v, the security manager starts a new period. First, the security manager broadcasts a special message change period (signed, but not encrypted). Note that we assume that change period is digitally signed by the security manager so that no third parties can maliciously initiate the New-period operation.

Let $enc : \mathbb{Z}_q \to \mathcal{G}$ be an easily invertible encoding that translates a number from $\{0, \ldots, q-1\}$ into an element of $\mathcal{G}$. If $\mathcal{G}$ is the subgroup of $\mathbb{Z}_p^*$ of oder $q = \frac{p-1}{2}$, then enc can be implemented as follows: $enc(a) \doteq (a+1)^2 \bmod p$. It is easy to see that $enc(a) \in \mathcal{G}$ for any $a \in \mathbb{Z}_q$: this is because $\mathcal{G}$ is the subgroup of quadratic residues modulo p. The encoding function enc can be easily inverted as follows: given $b := enc(a)$, compute the two square roots ρ_1, ρ_2 of a modulo p and define $enc^{-1}(b) = \min\{\rho_1, \rho_2\} - 1$ where min treats ρ_1, ρ_2 as integers in $\{0, \ldots, p-1\}$.

The security manager selects $d_0, \ldots, d_v, e_0, \ldots, e_v$ uniformly at random from $\mathbb{Z}_q$ and transmits the reset message

$$\begin{aligned}\psi_{\mathsf{reset}} := \langle &\mathcal{E}(PK, enc(d_0)), \ldots, \mathcal{E}(PK, enc(d_v)),\\ &\mathcal{E}(PK, enc(e_0)), \ldots, \mathcal{E}(PK, enc(e_v))\rangle\end{aligned}$$

where PK is the current public key of the system. Let $D(\cdot)$ be the polynomial defined by $d_0, \ldots, d_v$ and let $E(\cdot)$ be the polynomial defined by $e_0, \ldots, e_v$: namely,

$$\begin{aligned}D(x) &= d_0 + d_1 x + \ldots + d_v x^v\\ E(x) &= e_0 + e_1 x + \ldots + e_v x^v.\end{aligned}$$

At this point, the security manager resets the saturation level $L := 0$, updates the two secret polynomials to be:

$$\begin{aligned}A_{\mathsf{new}}(\cdot) &:= A(\cdot) + D(\cdot) \pmod q\\ B_{\mathsf{new}}(\cdot) &:= B(\cdot) + E(\cdot) \pmod q\end{aligned}$$

and modifies the public key PK as follows:

$$PK_{\mathsf{new}} := \langle g, g', g^{A_{\mathsf{new}}(0)} g'^{B_{\mathsf{new}}(0)}, \langle \ell, g^{A_{\mathsf{new}}(\ell)} g'^{B_{\mathsf{new}}(\ell)}\rangle_{\ell=1}^{v}\rangle.$$

Upon receiving the signed change period message, user i enters a wait-mode. When the user receives the reset message ψ_{reset}, he decrypts all ciphertexts, decodes the coefficients $d_0, \ldots, d_v, e_0, \ldots, e_v$ using enc^{-1} and forms the polynomials $D(\cdot), E(\cdot)$. Then, the user modifies his secret key SK_i to be the new tuple

$$SK_i := \langle x_i, A(x_i) + D(x_i), B(x_i) + E(x_i)\rangle.$$

Intuitively, this is secure because a revoked user i will not be able to decrypt any of the ciphertext in the reset message. Therefore, the secret polynomials in the (updated) master secret key will look completely random to user i and his secret key will become useless. A formal security analysis is presented in Sect. 5.

Remark. We notice that the efficiency of the New-period operation can be improved by using hybrid encryption. In particular, instead of computing and sending $2v+2$ ciphertexts under the current public-key (which incurs a cost of $O(v^2)$ in terms of communication), the security manager may pick a random session key k, use it to encrypt the $2v+2$ coefficients via a secure one-time symmetric-key encryption scheme, and broadcast the resulting ciphertext together with $\mathcal{E}(PK, enc'(k))$ (where enc' is a suitable encoding of session keys into elements of $\mathcal{G}$). Each non-revoked user will then be able to recover the coefficients $d_0, \ldots, d_v, e_0, \ldots, e_v$ from such reset message by first recovering the session key k from the public-key ciphertext $\mathcal{E}(PK, enc'(k))$, and then using k to decrypt the symmetric-key ciphertext. This will drop the communication cost to $O(v)$. We omit the details.

5 Dealing with revocation

5.1 Model for revocation

The public-key traitor tracing scheme described in Sect. 4 withstands a more powerful type of attack than what has been considered so far in previous related work [8,9,25,28]. In our attack scenario, the adversary $\mathcal{A}$ is allowed not only to join the system up to a bounded number of times v (equal to the *saturation level*, which is fixed as a system parameter), but also to observe and even actively affect the evolution of the system, by specifying which users should be revoked and their relative order in the sequence of revocations. Notice that this type of adversary defeats all previous public-key traitor tracing schemes with fixed ciphertext size [9,25,28].

More formally, in our model the adversary interleaves, in any adaptively-chosen order, two types of queries:

- Join query: it models the subscription to the system of a user controlled by the adversary. To reply to such query, a variant of the Add-user operation is executed, in which the adversary is allowed to specify the identity for which she will get the decryption key, (whereas in a regular Add-user operation, the security manager would assign a random identity to the new user). Thus, the Join query models a more powerful adversary that could control the random choice of the security manager. Notice that, after a Join query, the adversary obtains a valid secret key capable of recovering subsequent encrypted broadcasts.

- Revoke query: it models the revocation of a user from the system. To reply to such query, a Remove-user operation is performed and $\mathcal{A}$ is given the new public key that results after the invalidation of the key corresponding to the revoked user.

The main constraint that the above attack scenario imposes on the adversary's behavior is that $\mathcal{A}$ can make at most v Join queries; no restriction is placed on the number of Revoke queries. Whenever $\mathcal{A}$ has finished collecting the amount of information she thinks she needs to maximize her chances of winning the game, the corrupted users are revoked, the adversary outputs a pair of messages and receives back the encryption of either one with equal probability.

Note that proving security under this attack scenario does not mean that a real-world implementation of our scheme would withstand only adversaries corrupting up to v users; rather, it provides provable security against adversaries controlling an unlimited number of users, as long as no more than v users are ever corrupted in a row, without the security manager discovering them and revoking their decryption keys within a single period.

To fully appreciate the novelty of the attack scenario proposed above, recall that in the adversarial model that has been considered in previous work on public-key traitor tracing [8, 9, 25, 28], the only functionality conceded to $\mathcal{A}$ was to obtain the secret key of a user which was also *simultaneously* revoked from the system. In our model, such capability, usually called *corruption*, is split into two distinct operations. This clearly allows the adversary to mount more powerful attacks, and does indeed more closely model the reality, since the security manager does not always find out about "bad" users immediately. Moreover, keeping the Join and Revoke operations distinct, allows us to impose on the adversary the (minimal) restriction of obtaining at most v secret keys, without bounding the number of Revoke queries. This constitutes a major novelty of our adversarial model: previous work required both the number of revoked users and the number of compromised secret keys (tied together by the definition of *corruption* query) to be bounded by v.

Clearly, for the challenge to the adversary not to be trivial, all the secret keys that $\mathcal{A}$ obtains through Join queries must have been rendered useless by corresponding subsequent Revoke queries. We model this necessary constraint by requiring that before asking for her challenge, $\mathcal{A}$ enters a wait-mode during which all the (at most v) users she corrupted are revoked within a window of consecutive revocations that should not get interrupted by a New-period operation.

It is interesting to point here some technical similarities of the window adversary model to a (lunch-time) Chosen Ciphertext Attack (CCA1). In particular, in a lunch-time attack the adversary, prior to obtaining the challenge, can query a decryption oracle to obtain decryptions of chosen ciphertexts; in the security proof, this introduces the technical challenge of simulating such decryption oracle. In the case of a window-adversary, the adversary can query the Join oracle to obtain valid decryption keys (that will be revoked afterwards). From a technical viewpoint, simulating the Join oracle is a technical challenge of similar nature to the task of simulating the decryption oracle of a CCA1 attacker. Indeed, in our security proof and system design we take advantage of techniques that were developed for dealing with CCA1 attacks.

5.1.1 Formal model for window adversary

We formalize the above attack scenario in terms of the window adversary attack game $\mathbf{G}^v_{\mathsf{win}}(1^k)$, played between a challenger and the adversary $\mathcal{A}$. The game consists of three stages, denoted respectively fst, snd and trd. To enable coordination between the three stages, at the end of each stage $\mathcal{A}$ is allowed to output a piece of state information (via the variable aux), which will be given as input to the next stage.

The first stage (fst) is a learning stage, in which the adversary is allowed to obtain the secret keys of at most v users and to make the system evolve via Revoke queries. At the end of this stage, all the corrupted users get revoked.

The second stage (snd) is a choosing stage, in which $\mathcal{A}$ picks two messages M_0, M_1 that she deems she will be able to distinguish in the ciphertext form.

In the third stage (trd), $\mathcal{A}$ receives a challenge ciphertext ψ^*, which consists of the encryption of either M_0 or M_1 with equal probability. The game ends with $\mathcal{A}$ outputting her best guess to whether M_0 or M_1 was encrypted.

```
1. Let $\langle PK, MSK\rangle \overset{R}{\leftarrow} \mathsf{Setup}(1^k)$
2. Let $L := 0$, $\mathsf{Corr} := \emptyset$
3. Let $state := \langle L, PK, MSK, \mathsf{Corr}\rangle$
4. $aux \overset{R}{\leftarrow} \mathcal{A}^{\mathsf{Join}(state,\cdot),\mathsf{Revoke}(1,state,\cdot)}(\mathsf{fst}, state.PK)$
5. If $L + |\mathsf{Corr}| > v$ then exit
6. For all $x_j \in \mathsf{Corr}$ do $aux \overset{R}{\leftarrow} aux||\mathsf{Revoke}(0, state, x_j)$
7. $\langle aux, M_0, M_1\rangle \overset{R}{\leftarrow} \mathcal{A}^{\mathsf{Revoke}(1,state,\cdot)}(\mathsf{snd}, aux, state.PK)$
8. $\psi^* \overset{R}{\leftarrow} \mathcal{E}(state.PK, M_{\sigma^*})$, where $\sigma^* \overset{R}{\leftarrow} \{0,1\}$
9. $\sigma \overset{R}{\leftarrow} \mathcal{A}^{\mathsf{Revoke}(state,\cdot)}(\mathsf{trd}, aux, state.PK, \psi^*)$
10. Output Success if and only if $\sigma = \sigma^*$
```

The two oracles employed above are defined as follows:

```
$\mathsf{Join}(state, x)$ :
    (i) parse $state$ as $\langle L, PK, MSK, \mathsf{Corr}\rangle$
    (ii) parse $PK$ as $\langle g, g', y, \langle z_1, h_1\rangle, \ldots, \langle z_v, h_v\rangle\rangle$
    (iii) parse $MSK$ as $(A(\cdot), B(\cdot))$
    (iv) if $x \in \{1, \ldots, v\}$, then exit
    (v) set $\mathsf{Corr} := \mathsf{Corr} \cup \{x\}$ and return $(A(x), B(x))$

$\mathsf{Revoke}(\mathsf{isOracle}, state, x)$ :
    (i) parse $state$ as $\langle L, PK, MSK, \mathsf{Corr}\rangle$
    (ii) parse $PK$ as $\langle g, g', y, \langle z_1, h_1\rangle, \ldots, \langle z_v, h_v\rangle\rangle$
    (iii) parse $MSK$ as $(A(\cdot), B(\cdot))$
    (iv) if $\mathsf{isOracle} = 1$ and $x \in \mathsf{Corr}$, then exit
    (v) if $L = v$ then a New-period operation is executed
        and $state$ is updated accordingly (i.e., $L$ is reset to 0,
        $state.MSK$ is modified by adding the randomizing polynomials and $state.PK$ changes correspondingly)
    (vi) set $L := L + 1$
    (vii) update $state.PK$ by replacing the pair $\langle z_L, h_L\rangle$ with
        $\langle x, g^{A(x)} g'^{B(x)}\rangle$
    (viii) output $state.PK$; if step (v) caused a New-period
```

operation, then also output the corresponding reset message ψ_{reset}

Few points in the above formalization are worth of comment. First, without loss of generality, we assume that the adversary never corrupts the same user twice, as there is no extra information to be gained. We also assume that $\mathcal{A}$ never revokes the users it corrupts, as they get explicitly revoked at step 6. of the attack game.

Second, note that the test at step 5. is needed to enforce the window constraint: just testing $|\mathsf{Corr}| > v$ would not have been enough to ensure that in step 6. we can revoke all the corrupted users within the same period.

Finally, note that the code for Revoke is used both as an oracle to $\mathcal{A}$ (steps 4. and 7.) and as a subroutine for the attack game (step 6.). To distinguish these two cases, we use the boolean variable isOracle.

Definition 7. *Define $\mathcal{A}$'s advantage as*

$$\mathsf{Adv}_{\mathcal{A}}(k) \doteq \mid \Pr(\sigma = \sigma^*) - 1/2 \mid$$

where the probability is over all the randomness introduced by the window attack game.

A public-key traitor tracing scheme is secure against window adversaries if for any PPT *adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}(k)$ is negligible in k .*

5.2 Security of revocation

We now formally prove that the scalable public-key traitor tracing scheme described in Sect. 4 is secure against window adversaries (as defined above). In the security proof, we will follow the same structural approach used in [9], first advocated in [7]. Starting from the actual attack scenario, we consider a sequence of hypothetical games, all defined over the same probability space. In each game, the adversary's view is obtained in different ways, but its distribution is still indistinguishable among the games.

The security of our scheme relies on the DDH assumption (Assumption 1) as shown below in Theorem 1.

Theorem 1. *Under the decisional Diffie-Hellman Assumption for $\mathcal{G}$, the scheme presented above is secure against window adversaries.*

Proof. We define a sequence of "indistinguishable" games $\mathbf{G}_0, \mathbf{G}_1, \ldots$, all operating over the same underlying probability space. Starting from the actual adversarial game $\mathbf{G}_0 \doteq \mathbf{G}^v_{\mathsf{win}}(1^k)$, we incrementally make slight modifications to the behavior of the oracles, thus changing the way the adversary's view is computed, while maintaining the views' distributions indistinguishable among 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. Recall that in each game $\mathbf{G}_j$, the goal of adversary $\mathcal{A}$ is to output $\sigma \in \{0,1\}$ which is her best guess to the bit σ^* used at step 7. of the attack game $\mathbf{G}^v_{\mathsf{win}}(1^k)$ to create the challenge ciphertext ψ^*: let T_j be the event that $\sigma = \sigma^*$ in game $\mathbf{G}_j$ (i.e., the event that the game ends with Success as output). Without loss of generality, in the following we assume that the adversary corrupts exactly v users during the attack game.

Game $\mathbf{G}_0$. Define $\mathbf{G}_0$ to be the original game $\mathbf{G}^v_{\mathsf{win}}(1^k)$.

Game $\mathbf{G}_1$. Define the "special" New-period operation to be the first one to be caused by the Revoke oracle at step 7. of the attack game. Depending on the adversary's strategy, such "special" New-period operation may not occur at all.

Game $\mathbf{G}_1$ is identical to game $\mathbf{G}_0$, except that in $\mathbf{G}_1$ the reset message output by the "special" New-period operation contains $2v + 2$ encryptions of random elements of $\mathbb{Z}_q$, rather than encryptions of the coefficients of the randomizing polynomials. This modification suggests that the secret polynomials which are contained in $state.MSK$ at the beginning of the period initiated by the "special" New-period operation are totally random, even given all the information in the adversary's view.

In Lemma 2 (whose proof is given below), we show that the chances of adversary $\mathcal{A}$ winning game $\mathbf{G}_1$ cannot be significantly better than her chances of winning game $\mathbf{G}_0$: more precisely,

$$\left|\Pr[T_1] - \Pr[T_0]\right| \leq (4v + 4)\ \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{5}$$

Game $\mathbf{G}_2$. To turn game $\mathbf{G}_1$ into game $\mathbf{G}_2$, step 8. of the attack game is modified as follows:

$$8'.\ \psi^* \leftarrow \mathcal{E}(state.PK, M), \text{where } M \overset{R}{\leftarrow} \mathcal{G}, \sigma^* \overset{R}{\leftarrow} \{0,1\}$$

Because of this change, the challenge ciphertext ψ^* no longer contains σ^*, nor does any other information in the adversary's view; therefore,

$$\Pr[T_2] = \frac{1}{2}. \tag{6}$$

In Lemma 3, proven below, we show that the adversary has almost the same chances to guess σ^* in game $\mathbf{G}_1$ and $\mathbf{G}_2$: more precisely,

$$\left|\Pr[T_2] - \Pr[T_1]\right| \leq 2\ \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{7}$$

Combining Eqs. (5), (6), and (7) together, adversary $\mathcal{A}$'s advantage can be bounded as:

$$\mathsf{Adv}_{\mathcal{A}}(k) \leq (4v + 6)\ \mathsf{AdvDDH}_{\mathcal{G}}(k).$$

□

The core of the proof of Theorem 1 is in the two lemmas that follow, Lemma 2 and Lemma 3.

5.2.1 Overview of the proof technique

Throughout the paper, we make extensive use of a technical lemma, stated and proved as Lemma 9 in [7]. For ease of reference, we report it verbatim below.

Lemma 1. *Let k,n be integers with $1 \leq k \leq n$, and let K be a finite field. Consider a probability space with random variables $\boldsymbol{\alpha} \in K^{n\times 1}, \boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)^T \in K^{k\times 1}, \boldsymbol{\gamma} \in K^{k\times 1}$, and $\boldsymbol{M} \in K^{k\times n}$, such that $\boldsymbol{\alpha}$ is uniformly distributed* *over K^n, $\boldsymbol{\beta} = \boldsymbol{M\alpha} + \boldsymbol{\gamma}$, and for $1 \le i \le k$, the first ith rows of $\boldsymbol{M}$ and $\boldsymbol{\gamma}$ are determined by $\beta_1, \dots, \beta_{i-1}$. Then, conditioning on any fixed values of $\beta_1, \dots, \beta_{k-1}$ such that the resulting matrix $\boldsymbol{M}$ has rank k, the value of β_k is uniformly distributed over K in the resulting conditional probability space.*

Our use of this technical lemma is quite uniform across the proofs to follow. In all cases, our main aim will be to prove that some quantity $\mathsf{rand} \in \mathbb{Z}_q$ looks uniformly random to the adversary, despite all the other information in the adversary's view. At a high level, our approach is organized in the following steps.

First, we consider all the randomness underlying a specific execution of the attack game. This will include, for instance, the random coins of the adversary, the randomness used in creating the challenge, etc. We then partition all the randomness in two parts: a quantity $\boldsymbol{V}$ and a vector $\boldsymbol{\alpha}$, such that conditioning on any fixed value of $\boldsymbol{V}$, $\boldsymbol{\alpha}$ is still distributed uniformly at random in the appropriate vector space (which usually will have $\mathbb{Z}_q$ as support).

Second, we consider another vector $\boldsymbol{\beta}$, whose last entry is rand, with the property that fixing a value for $\boldsymbol{V}$ and $\boldsymbol{\beta}$ also fixes the value of $\boldsymbol{\alpha}$, and thus all the information of the entire game (which in particular includes the information in the adversary's view).

Third, we define a matrix $\mathbf{M}$ (and possibly a vector $\boldsymbol{\gamma}$) describing the constraints binding vector $\boldsymbol{\alpha}$ to vector $\boldsymbol{\beta}$, thus obtaining a matrix equation of the form:

$$\boldsymbol{\beta} = \mathbf{M} \cdot \boldsymbol{\alpha} + \boldsymbol{\gamma}.$$

Finally, we make sure that the preconditions of Lemma 1 are fulfilled; it will follow that the last entry of $\boldsymbol{\beta}$ (which is the quantity of interest rand), is distributed uniformly at random in $\mathbb{Z}_q$, even conditioning on fixed values of $\boldsymbol{V}$ and of all the other entries of $\boldsymbol{\beta}$, or equivalently, conditioning on all the other information in the adversary's view.

5.2.2 Notation

In what follows, we refer to the period initiated by the tth $\mathsf{New\text{-}period}$ operation as the tth period. Also, for notational convenience, we denote with $D^t(\cdot)$ and $E^t(\cdot)$ the randomizing polynomials chosen during the tth $\mathsf{New\text{-}period}$ operation and with $d_0^t, \dots, d_v^t$ and $e_0^t, \dots, e_v^t$ the corresponding coefficients. In some cases, it will be convenient to denote these $2v + 2$ coefficients with a uniform notation; for this reason, for $j = 1, \dots, 2v + 2$, we additionally define c_j^t as follows:

$$c_j^t \doteq \begin{cases} d_{j-1}^t & \text{if } j \in \{1, \dots, v+1\} \\ e_{j-v-2} & \text{if } j \in \{v+2, \dots, 2v+2\} \end{cases}$$

Moreover, let $A^t(\cdot)$ and $B^t(\cdot)$ be the values of the secret polynomials after the changes due to the tth $\mathsf{New\text{-}period}$ operation. In other words, the system starts with period number 0, $A^0(\cdot)$ and $B^0(\cdot)$ are the polynomials initially output by the Setup algorithm and

$$A^t(\cdot) \doteq A^{t-1}(\cdot) + D^t(\cdot) \quad B^t(\cdot) \doteq B^{t-1}(\cdot) + E^t(\cdot). \tag{8}$$

Also define

$$D^{t_1,t_2}(\cdot) \doteq \sum_{t=t_1}^{t_2} D^t(\cdot) \quad E^{t_1,t_2}(\cdot) \doteq \sum_{t=t_1}^{t_2} E^t(\cdot). \tag{9}$$

5.2.3 Proofs of lemmata

Lemma 2. $\left|\Pr[T_1] - \Pr[T_0]\right| \le (4v + 4)\ \mathsf{AdvDDH}_{\mathcal{G}}(k)$.

Proof. Recall that $\mathbf{G}_1$ differs from $\mathbf{G}_0$ only in the way the reset message is computed for the "special" $\mathsf{New\text{-}period}$ operation: hence, if the adversary's strategy does not cause any $\mathsf{New\text{-}period}$ operation to occur during step 7. of the attack game, the two games are identical, so that in fact $\Pr[T_1] = \Pr[T_0]$, and the lemma immediately follows.

We now discuss the case in which the "special" $\mathsf{New\text{-}period}$ operation takes place: in particular, let $\hat{t}$ be the period initiated by this operation and $D^{\hat{t}}(\cdot)$ and $E^{\hat{t}}(\cdot)$ be the randomizing polynomials used in such $\mathsf{New\text{-}period}$ operation. We then consider the sequence of $2v + 3$ hybrid games $\mathbf{G}_{0,0}, \dots, \mathbf{G}_{0,2v+2}$, where $\mathbf{G}_{0,i}$ is defined as $\mathbf{G}_0$, except that the first i ciphertexts in the "special" reset message contain random values rather than coefficients of the randomizing polynomials $D^{\hat{t}}(\cdot)$ and $E^{\hat{t}}(\cdot)$. In other words, $\mathbf{G}_{0,0} \equiv \mathbf{G}_0$, $\mathbf{G}_{0,2v+2} \equiv \mathbf{G}_1$ and two consecutive hybrid games $\mathbf{G}_{0,i}$ and $\mathbf{G}_{0,i+1}$ differ only in that the $(i + 1)$th ciphertext of the "special" reset message contains the $(i + 1)$th coefficient in game $\mathbf{G}_{0,i}$, whereas it contains a random value in game $\mathbf{G}_{0,i+1}$. Then, to prove the lemma it suffices to show that for all $i = 0, \dots, 2v + 1$ it holds:

$$\left|\Pr[T_{0,i+1}] - \Pr[T_{0,i}]\right| \le 2\ \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{10}$$

To this aim, fix i and consider the additional games $\mathbf{G}_{0,i}^0 \equiv \mathbf{G}_{0,i}$, $\mathbf{G}_{0,i}^1$, $\mathbf{G}_{0,i}^2$, $\mathbf{G}_{0,i}^3$, $\mathbf{G}_{0,i}^4 \equiv \mathbf{G}_{0,i+1}$, defined as follows:

Game $\mathbf{G}_{0,i}^1$. It operates as $\mathbf{G}_{0,i}^0$, except that the $(i + 1)$th ciphertext in the "special" reset message is computed as:

$$\langle u, u', u'', \langle z_\ell, u^{A^{\hat{t}-1}(z_\ell)} u'^{B^{\hat{t}-1}(z_\ell)} \rangle_{\ell=1}^v \rangle$$

where $u \doteq g^r$, $u' \doteq g'^r$, $u'' \doteq u^{A^{\hat{t}-1}(0)}\ u'^{B^{\hat{t}-1}(0)} enc(c_{i+1}^{\hat{t}})$, $r \overset{R}{\leftarrow} \mathbb{Z}_q$ and $c_{i+1}^{\hat{t}}$ is either the $(i + 1)$th coefficient of the randomizing polynomial $D^{\hat{t}}(\cdot)$ (if $0 \le i \le v$) or the $(i - v)$th coefficient of $E^{\hat{t}}(\cdot)$ (if $v + 1 \le i \le 2v + 1$). Since such modification is just a syntactic change, it holds:

$$\Pr[T_{0,i}^1] = \Pr[T_{0,i}^0]. \tag{11}$$

Game $\mathbf{G}_{0,i}^2$. To turn game $\mathbf{G}_{0,i}^1$ into game $\mathbf{G}_{0,i}^2$ we make another change to the way in which the $(i + 1)$th ciphertext in the "special" reset message is computed. Namely, the value u' is now computed as $u' \doteq g'^{r'}$, for a random $r' \in \mathbb{Z}_q$ such that $r' \ne r$. In other words, in game $\mathbf{G}_{0,i}^2$ the values u and u' are nearly independent (being subject only to $r \ne r'$), whereas in game $\mathbf{G}_{0,i}^1$ they are obtained using the same value r. Therefore, using a standard reduction argument, any difference in behavior between games $\mathbf{G}_{0,i}^1$ and $\mathbf{G}_{0,i}^2$ can be used to distinguish Diffie-Hellman tuples from totally random tuples. Hence,

$$\left|\Pr[T_{0,i}^2] - \Pr[T_{0,i}^1]\right| \le \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{12}$$

Note that for simplicity here (and throughout the rest of the paper) we omit the negligible additive term that is caused by the negligibly-rare event $r = r'$.

Game $\mathbf{G}^3_{0,i}$. To define game $\mathbf{G}^3_{0,i}$, we again modify the $(i+1)$th ciphertext in the "special" reset message: specifically, the value u'' is now computed as $g^{r''}$, for a random $r'' \in \mathbb{Z}_q$.

We want to show that this modification does not alter the behavior of adversary $\mathcal{A}$ or, more precisely, that $\Pr[T^3_{0,i}] = \Pr[T^2_{0,i}]$. To this aim, we first consider all the random variables affecting the adversary's view, and then we show that they are distributed according to the same joint distribution in both games.

Let $\bar{t}$ be the total number of New-period operations that occur during the entire game, and for $t = 1, \ldots, \bar{t}$, let c^t_1, …, c^t_{2v+2} be the coefficients of the randomizing polynomials $D^t(\cdot)$ and $E^t(\cdot)$ used in the tth New-period operation. For $t = 1, \ldots, \bar{t}$, $t \neq \hat{t}$, and $j = 1, \ldots, 2v+2$, let r^t_j be the randomness used to encrypt (the encoding of) coefficient c^t_j in the tth reset message.

As for the "special" reset message (i.e., the one corresponding to $t = \hat{t}$), recall that in both game $\mathbf{G}^2_{0,i}$ and game $\mathbf{G}^3_{0,i}$ the first i ciphertexts consists of just random values s_1, …, $s_i \in \mathcal{G}$, rather than (the encoding of) the corresponding coefficients $c^{\hat{t}}_1$, …, $c^{\hat{t}}_i$. Coefficients $c^{\hat{t}}_{i+2}$, …, $c^{\hat{t}}_{2v+2}$, instead, are regularly encrypted under the public key $PK^{\hat{t}-1}$ in both games: let $r^{\hat{t}}_j$ be the randomness used in such encryptions, for $j = i+2, \ldots, 2v+2$. The ciphertext corresponding to coefficient c_{i+1} in the "special" reset message constitutes the only difference between the adversary's view in game $\mathbf{G}^2_{0,i}$ and $\mathbf{G}^3_{0,i}$. In particular, such encryption is defined in terms of the values r, r' and r'': r and r' are randomly chosen from $\mathbb{Z}_q$ in both games, whereas r'' is computed differently in the two games. For the sake of clarity, we will denote with $[r'']_2$ and $[r'']_3$ the value of such quantity in game $\mathbf{G}^2_{0,i}$ and $\mathbf{G}^3_{0,i}$, respectively. Notice that $[r'']_2$ is a linear combination of r, r' (and other quantities), whereas $[r'']_3$ is uniformly distributed in $\mathbb{Z}_q$, independently of anything else.

Define

$$\boldsymbol{W} \doteq \left(\{c^t_j, r^t_j\}^{2v+2}_{\substack{j=1 \\ t \neq \hat{t}}}, \{c^{\hat{t}}_j, s_j, r^{\hat{t}}_j\}^i_{j=1}, \{c^{\hat{t}}_j, r^{\hat{t}}_j\}^{2v+2}_{j=i+1}, r, r' \right)$$

and consider the quantity

$$\boldsymbol{V} \doteq (\mathsf{Coins}, w, \sigma^*, r^*, \boldsymbol{W})$$

where Coins represents the coin tosses of $\mathcal{A}$, $w \doteq \log_g g'$, σ^* is the random bit chosen by the challenger in step 8. of the attack game and r^* is the randomness used to create the challenge ψ^*.

The remaining randomness used during the attack game consists of the $2v+2$ coefficients of the polynomials $A^0(\cdot)$, $B^0(\cdot)$ and can be represented by a vector $\boldsymbol{\alpha}$ uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$:

$$\boldsymbol{\alpha} \doteq (a_0, a_1, \ldots, a_v, b_0, b_1, \ldots, b_v)^T.$$

Consider the vector $\boldsymbol{\beta} \in \mathbb{Z}_q^{(2v+2)\times 1}$ defined as:

$$\boldsymbol{\beta} \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \mathbf{A}_1, \ldots, \mathbf{A}_v, r'')^T$$

where $\mathbf{X}_0 \doteq A^0(0) + wB^0(0)$, $\mathbf{X}_\ell \doteq A^0(\ell) + wB^0(\ell)$ and $\mathbf{A}_\ell \doteq A^0(x_\ell)$ for $\ell = 1, \ldots, v$, and $r'' \doteq \log_g u''$.

It is clear by inspection that all the information in the adversary's view is completely determined by $\boldsymbol{V}$ and $\boldsymbol{\beta}$. In particular, the initial public key PK^0 is fixed by $\boldsymbol{\beta}$ and w; the secret keys of the corrupted users are determined by the choice of $\boldsymbol{\beta}$, Coins and w; the "special" reset message is fixed by PK^0, Coins, r'' and all the randomness in $\boldsymbol{W}$; and the resulting public key $PK^{\hat{t}}$ only depends on PK^0 and $\boldsymbol{W}$. Thus, if the distribution of $\boldsymbol{V}$ and $\boldsymbol{\beta}$ is the same in both games $\mathbf{G}^2_{0,i}$ and $\mathbf{G}^3_{0,i}$, it will follow that $\Pr[T^3_{0,i}] = \Pr[T^2_{0,i}]$. Since the definition of r'' is the only difference between game $\mathbf{G}^2_{0,i}$ and $\mathbf{G}^3_{0,i}$, and in $\mathbf{G}^3_{0,i}$ the value of $[r'']_3$ is chosen uniformly from $\mathbb{Z}_q$, independently of anything else, it suffices to show that the distribution of $[r'']_2$, conditioned on $\boldsymbol{V}$ and the first $2v+1$ entries of $\boldsymbol{\beta}$, is also uniform in $\mathbb{Z}_q$.

In game $\mathbf{G}^2_{0,i}$, the quantities in $\boldsymbol{V}$, $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are related according to the following matrix equation:

$$[\boldsymbol{\beta}]_2 = \mathbf{M} \cdot \boldsymbol{\alpha} + \boldsymbol{\gamma}$$

where $[\boldsymbol{\beta}]_2$ denotes the value of $\boldsymbol{\beta}$ in game $\mathbf{G}^2_{0,i}$ (i.e. when the value of the last entry is $[r'']_2$), $\boldsymbol{\gamma} \in \mathbb{Z}_q^{(2v+2)\times 1}$ is the vector

$$\boldsymbol{\gamma} \doteq \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ rD^{0,\hat{t}-1}(0) + wr'E^{0,\hat{t}-1}(0) + \log_g enc(c^{\hat{t}}_{i+1}) \end{pmatrix}$$

and $\mathbf{M} \in \mathbb{Z}_q^{(2v+2)\times(2v+2)}$ is the matrix

$$\mathbf{M} \doteq \begin{pmatrix} 1 & 0 & \ldots & 0 & w & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 1 & w & w & \ldots & w \\ & & \vdots & & & & \vdots & \\ 1 & v & \ldots & v^v & w & wv & \ldots & wv^v \\ 1 & x_1 & \ldots & x_1^v & 0 & 0 & \ldots & 0 \\ & & \vdots & & & & \vdots & \\ 1 & x_v & \ldots & x_v^v & 0 & 0 & \ldots & 0 \\ r & 0 & \ldots & 0 & wr' & 0 & \ldots & 0 \end{pmatrix}$$

The above matrix describes all the constraints on $\boldsymbol{\alpha}$ arising from the information in the adversary's view in game $\mathbf{G}^2_{0,i}$ (which, as noted above, can be described just by $\boldsymbol{V}$ and $[\boldsymbol{\beta}]_2$). In other words, all other constraints on $\boldsymbol{\alpha}$ are linear combination of the above, possibly using coefficients from $\boldsymbol{V}$. In particular, the constraints that the adversary can derive from knowledge of the values $B^0(x_\ell)$, $\ell = 1, \ldots, v$ (which come from the secret keys that $\mathcal{A}$ got via Join queries) can be obtained from the constraints corresponding to $\mathbf{X}_0$, $\mathbf{X}_1$, …, $\mathbf{X}_v$, $\mathbf{A}_1$, …, $\mathbf{A}_v$ and the value of w. As for Revoke queries, notice that the public key PK resulting from invalidating the secret key of an arbitrary user z during time period t, does not provide any new information about $\boldsymbol{\alpha}$ to the adversary. Indeed, PK only differs from the previous public key in that it contains the value $h_z = g^{A^t(z)} g'^{B^t(z)}$ which is determined by the quantity:

$$\mathbf{X} \doteq A^t(z) + wB^t(z) - (D^{0,t}(z) + wE^{0,t}(z)) \\ = A^0(z) + wB^0(z).$$

Since such value is just a point of the v-degree polynomial $A^0(\cdot) + wB^0(\cdot)$, which is completely fixed by the values $\mathbf{X}_0$, $\mathbf{X}_1$, ..., $\mathbf{X}_v$, it immediately follows that the constraint on $\boldsymbol{\alpha}$ induced by $\mathbf{X}$ is a linear combination of the first $v+1$ rows of $\mathbf{M}$. Similarly, the v values $u_1, \ldots, u_v$ included in the $(v+1)$th ciphertext of the "special" reset message are determined by the quantities $\mathbf{X}_{z_1}$, ..., $\mathbf{X}_{z_v}$ where, for $\ell = 1, \ldots, v$, $\mathbf{X}_{z_\ell}$ is defined as:

$$rA^{\hat{t}-1}(z_\ell) + wr'B^{\hat{t}-1}(z_\ell) - (rD^{0,\hat{t}-1}(z_\ell) + wr'E^{0,\hat{t}-1}(z_\ell))$$

or equivalently

$$\mathbf{X}_{z_\ell} \doteq rA^0(z_\ell) + wr'B^0(z_\ell).$$

Such values are just points of the v-degree polynomial

$$rA^0(\cdot) + wr'B^0(\cdot)$$

which is determined by $\mathbf{A}_1, \ldots, \mathbf{A}_v$, $B^0(x_1), \ldots, B^0(x_v)$, r, r', w and $[r'']_2$. Thus, it follows that all the constraints on $\boldsymbol{\alpha}$ induced by $\mathbf{X}_{z_1}, \ldots, \mathbf{X}_{z_v}$ are linear combinations of the rows of $\mathbf{M}$.

Moreover, $\mathbf{M}$ has rank $(2v+2)$, provided that $r \neq r'$ and $w \neq 0$, since the corrupted users x_1, ..., x_v are assumed to be distinct.

As soon as we fix a value for $\boldsymbol{V}$, vector $\boldsymbol{\gamma}$ and the first $v+1$ rows of matrix $\mathbf{M}$ are completely fixed, but $\boldsymbol{\alpha}$ is still distributed uniformly and independently at random in $\mathbb{Z}_q^{(2v+2)\times 1}$. If we additionally fix the value of the first $(v+1)$ components of $[\boldsymbol{\beta}]_2$, the initial public key PK^0 is fixed; it follows that the first identity x_1 that $\mathcal{A}$ chooses to corrupt is also fixed and thus the $(v+2)$th row of $\mathbf{M}$ is determined. Fixing also a value for $\mathbf{A}_1$ (which is the $(v+2)$th entry of $[\boldsymbol{\beta}]_2$), the value of $\mathbf{B}_1$ is fixed too, so that all the information on which the adversary can base her choice of x_2 is fixed, and thus the $(v+3)$th row of $\mathbf{M}$ is determined as well. By a similar reasoning, it follows that fixing the first $(v+i+1)$ entries of $[\boldsymbol{\beta}]_2$ determines the $(v+i+2)$th row of $\mathbf{M}$, for $i = 1, \ldots, v$. Hence, by Lemma 1, we can conclude that the conditional distribution of $[r'']_2$, with respect to $\boldsymbol{V}$ and to all other components of $[\boldsymbol{\beta}]_2$, is also uniform over $\mathbb{Z}_q$, from which it follows that

$$\Pr[T^3_{0,i}] = \Pr[T^2_{0,i}]. \tag{13}$$

Game $\mathbf{G}^4_{0,i}$. Game $\mathbf{G}^4_{0,i}$ is defined to be identical to $\mathbf{G}_{0,i+1}$. Thus, $\mathbf{G}^4_{0,i}$ differs from $\mathbf{G}^3_{0,i}$ only in that the values u and u' in the $(i+1)$th ciphertext in the "special" reset message are consistent, rather than being nearly independent, as in game $\mathbf{G}^3_{0,i}$. Namely, the values u and u' are now computed as $u \doteq g^r$ and $u' \doteq g'^r$, for the same random $r \in \mathbb{Z}_q$. It follows that any difference in behavior between games $\mathbf{G}^3_{0,i}$ and $\mathbf{G}^4_{0,i}$ can be used to distinguish Diffie-Hellman tuples from totally random tuples. Hence,

$$\left|\Pr[T^4_{0,i}] - \Pr[T^3_{0,i}]\right| \leq \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{14}$$

Combining Eqs. (11), (12), (13) and (14) we get Eq. (10), for all $i = 0, \ldots, 2v+1$; then, by definition of the hybrid sequence $\mathbf{G}_{0,0}, \ldots, \mathbf{G}_{0,2v+2}$, the thesis follows. □

Lemma 3. $\left|\Pr[T_2] - \Pr[T_1]\right| \leq 2\ \mathsf{AdvDDH}_{\mathcal{G}}(k)$.

Proof. Recall that $\mathbf{G}_2$ differs from $\mathbf{G}_1$ only in the way the challenge ciphertext ψ^* is computed: in particular, in game $\mathbf{G}_1$, ψ^* encrypts either one of the two messages M_0 and M_1 chosen by the adversary, whereas in $\mathbf{G}_2$, ψ^* encrypts a totally random message $M \overset{R}{\leftarrow} \mathcal{G}$.

To reach the thesis, we consider the sequence of games $\mathbf{G}^0_1 \equiv \mathbf{G}_1$, $\mathbf{G}^1_1$, $\mathbf{G}^2_1$, $\mathbf{G}^3_1$, $\mathbf{G}^4_1 \equiv \mathbf{G}_2$, defined below.

Game $\mathbf{G}^1_1$. It operates as $\mathbf{G}^0_1$, except that the challenge ciphertext is computed as follows:

$$\langle u^*, u'^*, u''^*, \langle z^*_\ell, u^{*A^{t^*}(z^*_\ell)} u'^{*B^{t^*}(z^*_\ell)}\rangle_{\ell=1}^{v}\rangle$$

where $u^* \doteq g^{r^*}$, $u'^* \doteq g'^{r^*}$, $u''^* \doteq u^{*A^{t^*}(0)} u'^{*B^{t^*}(0)} M_{\sigma^*}$ and $r^* \overset{R}{\leftarrow} \mathbb{Z}_q$. This syntactic change does not affect the adversary's view, and thus

$$\Pr[T^1_1] = \Pr[T^0_1]. \tag{15}$$

Game $\mathbf{G}^2_1$. To turn game $\mathbf{G}^1_1$ into game $\mathbf{G}^2_1$ we make another change to the way in which the challenge ciphertext is computed. Namely, the value u'^* is now computed as $u'^* \doteq g'^{r'^*}$, for a random $r'^* \in \mathbb{Z}_q$ such that $r'^* \neq r^*$. In other words, in game $\mathbf{G}^2_1$ the values u^* and u'^* are nearly independent (being subject only to $r^* \neq r'^*$), whereas in game $\mathbf{G}^1_1$ they are obtained using the same value r^*. Therefore, using a standard reduction argument, any difference in behavior between games $\mathbf{G}^1_1$ and $\mathbf{G}^2_1$ can be used to distinguish Diffie-Hellman tuples from totally random tuples. Hence,

$$\left|\Pr[T^2_1] - \Pr[T^1_1]\right| \leq \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{16}$$

Game $\mathbf{G}^3_1$. To define game $\mathbf{G}^3_1$, we again modify the challenge ciphertext: specifically, the value u''^* is now computed as $g^{r''^*}$, for a random $r''^* \in \mathbb{Z}_q$.

To prove that $\Pr[T^3_1] = \Pr[T^2_1]$, we first consider all the quantities that can affect event T^2_1 in game $\mathbf{G}^2_1$ and event T^3_1 in game $\mathbf{G}^3_1$, and then we show that these quantities have the same joint distribution in both games.

Let $D^{t^*}(\cdot)$ and $E^{t^*}(\cdot)$ be the randomizing polynomials used in the last $\mathsf{New\text{-}period}$ operation before the challenge ciphertext was created. (If no $\mathsf{New\text{-}period}$ occurred at all during the attack game, then let both $D^{t^*}(\cdot)$ and $E^{t^*}(\cdot)$ be just the zero polynomial.)

Let $\bar{t}$ be the total number of $\mathsf{New\text{-}period}$ operations that occured during the entire game, and for $t = 1, \ldots, \bar{t}$, let c^t_1, ..., c^t_{2v+2} be the coefficients of the randomizing polynomials $D^t(\cdot)$ and $E^t(\cdot)$ used in the tth $\mathsf{New\text{-}period}$ operation. For $t = 1, \ldots, \bar{t}$, and $j = 1, \ldots, 2v+2$, let r^t_j be the randomness used to encrypt (the encoding of) coefficient c^t_j in the tth reset message.

Observe that the challenge ciphertext ψ^* is the only value in the adversary's view which is computed differently in game

$\mathbf{G}_1^2$ and game $\mathbf{G}_1^3$. In particular, such encryption is defined in terms of the values r^*, r'^* and r''^*: r^* and r'^* are randomly chosen from $\mathbb{Z}_q$ in both games, whereas r''^* is computed differently in the two games. For the sake of clarity, we will denote with $[r''^*]_2$ and $[r''^*]_3$ the value of such quantity in game $\mathbf{G}_1^2$ and $\mathbf{G}_1^3$, respectively. Notice that $[r''^*]_2$ is a linear combination of r^*, r'^* (and other quantities), whereas $[r''^*]_3$ is uniformly distributed in $\mathbb{Z}_q$, independently of anything else.

The rest of our analysis proceeds differently depending on whether the adversary's strategy caused the "special" New-period operation to occur or not. The case in which no New-period operation occurred at step 7. of the attack game is slightly simpler, so we discuss it first.

Case 1. Consider the quantity

$$\boldsymbol{V} \doteq (\mathsf{Coins}, w, \{\{c_j^t, r_j^t\}_{j=1}^{2v+2}\}_{t=1}^{\bar{t}}, \sigma^*, r^*, r'^*)$$

where Coins represents the coin tosses of $\mathcal{A}$, $w \doteq \log_g g'$, and σ^* is the random bit chosen by the challenger in step 8. of the attack game.

The remaining randomness used during the attack game consists of the $2v+2$ coefficients of the polynomials $A^0(\cdot)$, $B^0(\cdot)$ and can be represented by a vector $\boldsymbol{\alpha}$ uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$:

$$\boldsymbol{\alpha} \doteq (a_0, a_1, \ldots, a_v, b_0, b_1, \ldots, b_v)^T.$$

Consider the vector $\boldsymbol{\beta} \in \mathbb{Z}_q^{(2v+2)\times 1}$ defined as:

$$\boldsymbol{\beta} \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \mathbf{A}_1, \ldots, \mathbf{A}_v, r''^*)^T$$

where $\mathbf{X}_0 \doteq A^0(0) + wB^0(0)$, $\mathbf{X}_\ell \doteq A^0(\ell) + wB^0(\ell)$ and $\mathbf{A}_\ell \doteq A^0(x_\ell)$ for $\ell = 1, \ldots, v$, and $r''^* \doteq \log_g u''^*$.

It is clear by inspection that all the information in the adversary's view is completely determined by $\boldsymbol{V}$ and $\boldsymbol{\beta}$. Thus, if the distribution of $\boldsymbol{V}$ and $\boldsymbol{\beta}$ is the same in both games $\mathbf{G}_1^2$ and $\mathbf{G}_1^3$, it will follow that $\Pr[T_1^3] = \Pr[T_1^2]$. Since the definition of r''^* is the only difference between game $\mathbf{G}_1^2$ and $\mathbf{G}_1^3$, and in $\mathbf{G}_1^3$ the value of $[r''^*]_3$ is chosen uniformly from $\mathbb{Z}_q$, independently of anything else, it suffices to show that the distribution of $[r''^*]_2$, conditioned on $\boldsymbol{V}$ and the first $2v+1$ entries of $\boldsymbol{\beta}$, is also uniform in $\mathbb{Z}_q$.

In game $\mathbf{G}_1^2$, the quantities in $\boldsymbol{V}$, $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are related according to the following matrix equation:

$$[\boldsymbol{\beta}]_2 = \mathbf{M} \cdot \boldsymbol{\alpha} + \boldsymbol{\gamma}$$

where $[\boldsymbol{\beta}]_2$ denotes the value of $\boldsymbol{\beta}$ in game $\mathbf{G}_1^2$ (i.e. when the value of the last entry is $[r''^*]_2$), $\boldsymbol{\gamma} \in \mathbb{Z}_q^{(2v+2)\times 1}$ is the vector

$$\boldsymbol{\gamma} \doteq \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ r^* D^{0,t^*}(0) + wr'^* E^{0,t^*}(0) + \log_g M_{\sigma^*} \end{pmatrix}$$

and $\mathbf{M} \in \mathbb{Z}_q^{(2v+2)\times(2v+2)}$ is the matrix

$$\mathbf{M} \doteq \begin{pmatrix} 1 & 0 \ldots 0 & w & 0 \ldots 0 \\ 1 & 1 \ldots 1 & w & w \ldots w \\ & \vdots & & \vdots \\ 1 & v \ldots v^v & w & wv \ldots wv^v \\ 1 & x_1 \ldots x_1^v & 0 & 0 \ldots 0 \\ & \vdots & & \vdots \\ 1 & x_v \ldots x_v^v & 0 & 0 \ldots 0 \\ r^* & 0 \ldots 0 & wr'^* & 0 \ldots 0 \end{pmatrix}$$

The above matrix $\mathbf{M}$ is square, has full rank (provided that $r^* \neq r'^*$ and $w \neq 0$) and it describes all the constraints on the $(2v+2)$ unknowns represented by $\boldsymbol{\alpha}$, that can be derived from the information in the adversary's view in $\mathbf{G}_1^2$. In particular, the fact that no New-period operation occurred during execution of step 7. of the attack game guarantees that the identities included in the public key PK^* that was used to create the challenge ciphertext ψ^* are exactly those of the users corrupted by the adversary. Hence, the constraints on $\boldsymbol{\alpha}$ arising from the last v components of the challenge ciphertext ψ^* can be obtained as linear combination of the constraints specified by $\mathbf{M}$.

As soon as we fix a value for $\boldsymbol{V}$, the first $2v+1$ entries of vector $\boldsymbol{\gamma}$ and the first $v+1$ rows of matrix $\mathbf{M}$ are completely fixed, but $\boldsymbol{\alpha}$ is still distributed uniformly and independently at random in $\mathbb{Z}_q^{(2v+2)\times 1}$. If we additionally fix the value of the first $(v+1)$ components of $[\boldsymbol{\beta}]_2$, the initial public key PK^0 is fixed; it follows that the first identity x_1 that $\mathcal{A}$ chooses to corrupt is also fixed and thus the $(v+2)$th row of $\mathbf{M}$ is determined. Fixing also a value for A_1 (which is the $(v+2)$th entry of $[\boldsymbol{\beta}]_2$), the value of B_1 is fixed too, so that all the information on which the adversary can base her choice of x_2 is fixed, and thus the $(v+3)$th row of $\mathbf{M}$ is determined as well. By a similar reasoning, it follows that fixing the first $(v+\ell+1)$ entries of $[\boldsymbol{\beta}]_2$ determines the $(v+\ell+2)$th row of $\mathbf{M}$, for $\ell = 1, \ldots, v$. In particular, fixing all the entries of $[\boldsymbol{\beta}]_2$ but the last, fixes all the information that adversary $\mathcal{A}$ sees before asking for her challenge: thus, her choice of M_0, M_1 is determined, so that the last entry of $\boldsymbol{\gamma}$ is fixed, too. Hence, by Lemma 1, we can conclude that the conditional distribution of $[r''^*]_2$, with respect to $\boldsymbol{V}$ and to all the other components of $[\boldsymbol{\beta}]_2$, is also uniform over $\mathbb{Z}_q$, from which it follows that

$$\Pr[T_1^3] = \Pr[T_1^2]. \tag{17}$$

Case 2. We now discuss the case in which the "special" New-period operation takes place: in particular, let $D^{\hat{t}}(\cdot)$ and $E^{\hat{t}}(\cdot)$ be the randomizing polynomials used in such New-period operation. Consider the quantity

$$\boldsymbol{V} \doteq \left(\mathsf{Coins}, w, \{c_j^t, r_j^t\}_{\substack{j=1 \\ t \neq \hat{t}}}^{2v+2}, \{s_j, r_j^{\hat{t}}\}_{j=1}^{2v+2}, \sigma^*, r^*, r'^*\right)$$

where Coins represents the coin tosses of $\mathcal{A}$, $w \doteq \log_g g'$, σ^* is the random bit chosen by the challenger in step 8. of the attack game, and $s_1, \ldots, s_{2v+2}$ are the random elements of $\mathcal{G}$ that are encrypted by the "special" New-period operation in place of the randomizing coefficients $d_0^{\hat{t}}$, $d_1^{\hat{t}}, \ldots, d_v^{\hat{t}}$, and $e_0^{\hat{t}}$, $e_1^{\hat{t}}, \ldots, e_v^{\hat{t}}$.

$$\boldsymbol{\alpha} \doteq (a_0, a_1, \ldots, a_v, b_0, b_1, \ldots, b_v, d_0^{\hat{t}}, d_1^{\hat{t}}, \ldots, d_v^{\hat{t}}, e_0^{\hat{t}}, e_1^{\hat{t}}, \ldots, e_v^{\hat{t}})^T$$

$$\boldsymbol{\beta} \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \hat{\mathbf{X}}_0, \hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_v, \mathbf{A}_1, \ldots, \mathbf{A}_v, \mathbf{X}_1^*, \ldots, \mathbf{X}_v^*, r''^*)^T$$

$$\boldsymbol{\gamma} \doteq \left(\begin{array}{c} 0 \\ 0 \\ \vdots \\ 0 \\ D^{0,\hat{t}-1}(0) + wE^{0,\hat{t}-1}(0) \\ D^{0,\hat{t}-1}(1) + wE^{0,\hat{t}-1}(1) \\ \vdots \\ D^{0,\hat{t}-1}(v) + wE^{0,\hat{t}-1}(v) \\ 0 \\ \vdots \\ 0 \\ r^*(D^{0,\hat{t}-1}(z_1^*) + D^{\hat{t}+1,t^*}(z_1^*)) + wr'^*(E^{0,\hat{t}-1}(z_1^*) + E^{\hat{t}+1,t^*}(z_1^*)) \\ \vdots \\ r^*(D^{0,\hat{t}-1}(z_v^*) + D^{\hat{t}+1,t^*}(z_v^*)) + wr'^*(E^{0,\hat{t}-1}(z_v^*) + E^{\hat{t}+1,t^*}(z_v^*)) \\ r^*(D^{0,\hat{t}-1}(0) + D^{\hat{t}+1,t^*}(0)) + wr'^*(E^{0,\hat{t}-1}(0) + E^{\hat{t}+1,t^*}(0)) + \log_g M_{\sigma^*} \end{array} \right)$$

$$\mathbf{M} \doteq \left(\begin{array}{cccccccccccccccc} 1 & 0 & \ldots & 0 & w & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 1 & w & w & \ldots & w & 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 \\ & & \vdots & & & & \vdots & & & & \vdots & & & & \vdots & \\ 1 & v & \ldots & v^v & w & wv & \ldots & wv^v & 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 \\ 1 & 0 & \ldots & 0 & w & 0 & \ldots & 0 & 1 & 0 & \ldots & 0 & w & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 1 & w & w & \ldots & w & 1 & 1 & \ldots & 1 & w & w & \ldots & w \\ & & \vdots & & & & \vdots & & & & \vdots & & & & \vdots & \\ 1 & v & \ldots & v^v & w & wv & \ldots & wv^v & 1 & v & \ldots & v^v & w & wv & \ldots & wv^v \\ 1 & x_1 & \ldots & x_1^v & 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 \\ & & \vdots & & & & \vdots & & & & \vdots & & & & \vdots & \\ 1 & x_v & \ldots & x_v^v & 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & 0 \\ r^* & r^*z_1^* & \ldots & r^*z_1^{*v} & wr'^* & wr'^*z_1^* & \ldots & wr'^*z_1^{*v} & r^* & r^*z_1^* & \ldots & r^*z_1^{*v} & wr'^* & wr'^*z_1^* & \ldots & wr'^*z_1^{*v} \\ & & \vdots & & & & \vdots & & & & \vdots & & & & \vdots & \\ r^* & r^*z_v^* & \ldots & r^*z_v^{*v} & wr'^* & wr'^*z_v^* & \ldots & wr'^*z_v^{*v} & r^* & r^*z_v^* & \ldots & r^*z_v^{*v} & wr'^* & wr'^*z_v^* & \ldots & wr'^*z_v^{*v} \\ r^* & 0 & \ldots & 0 & wr'^* & 0 & \ldots & 0 & r^* & 0 & \ldots & 0 & wr'^* & 0 & \ldots & 0 \end{array} \right)$$

Fig. 1. Vectors $\boldsymbol{\alpha} \in \mathbb{Z}_q^{(4v+4)\times 1}$, $\boldsymbol{\beta} \in \mathbb{Z}_q^{(4v+3)\times 1}$ and $\boldsymbol{\gamma} \in \mathbb{Z}_q^{(4v+3)\times 1}$ and the matrix $\mathbf{M} \in \mathbb{Z}_q^{(4v+3)\times(4v+4)}$ used in the last information-theoretic argument of Lemma 3. Notation: $r''^* \doteq \log_g u''^*$, $\mathbf{X}_0 \doteq A^0(0) + wB^0(0)$, $\hat{\mathbf{X}}_0 \doteq A^{\hat{t}}(0) + wB^{\hat{t}}(0)$, and for $\ell = 1, \ldots, v$, $\mathbf{X}_\ell \doteq A^0(\ell) + wB^0(\ell)$, $\hat{\mathbf{X}}_\ell \doteq A^{\hat{t}}(\ell) + wB^{\hat{t}}(\ell)$, $\mathbf{A}_\ell \doteq A^0(x_\ell)$ and $\mathbf{X}_\ell^* \doteq r^*A^{t^*}(z_\ell^*) + wr'^*B^{t^*}(z_\ell^*)$

The remaining randomness used during the attack game consists of these randomizing coefficients, along with the $2v+2$ coefficients of the polynomials $A^0(\cdot)$, $B^0(\cdot)$ and can be represented by a vector $\boldsymbol{\alpha}$ uniformly distributed in $\mathbb{Z}_q^{(4v+4)\times 1}$, given in Fig. 1.

Consider the vector $\boldsymbol{\beta} \in \mathbb{Z}_q^{(4v+3)\times 1}$ defined in Fig. 1: it is clear by inspection that all the information in the adversary's view is completely determined by $\boldsymbol{V}$ and $\boldsymbol{\beta}$. In particular, the initial public key PK^0 is fixed by $\boldsymbol{\beta}$ and w; the secret keys of the corrupted users are determined by the choice of $\boldsymbol{\beta}$, Coins and w; the "special" reset message is fixed by PK^0, Coins, and all the randomness in $\boldsymbol{V}$; the resulting public key $PK^{\hat{t}}$ only depends on $\boldsymbol{\beta}$ and w; and the adversary's choice of M_0 and M_1 is fixed by $\boldsymbol{V}$ and the first $4v+2$ entries of $\boldsymbol{\beta}$.

Thus, if the distribution of $\boldsymbol{V}$ and $\boldsymbol{\beta}$ is the same in both games $\mathbf{G}_1^2$ and $\mathbf{G}_1^3$, it will follow that $\Pr[T_1^3] = \Pr[T_1^2]$. Since the definition of r''^* is the only difference between game $\mathbf{G}_1^2$ and $\mathbf{G}_1^3$, and in $\mathbf{G}_1^3$ the value of $[r''^*]_3$ is chosen uniformly from $\mathbb{Z}_q$, independently of anything else, it suffices to show that the distribution of $[r''^*]_2$, conditioned on $\boldsymbol{V}$ and the first $4v+2$ entries of $\boldsymbol{\beta}$, is also uniform in $\mathbb{Z}_q$.

In game $\mathbf{G}_1^2$, the quantities in $\boldsymbol{V}$, $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are related according to the following matrix equation:

$$[\boldsymbol{\beta}]_2 = \mathbf{M} \cdot \boldsymbol{\alpha} + \boldsymbol{\gamma}$$

where $[\boldsymbol{\beta}]_2$ denotes the value of $\boldsymbol{\beta}$ in game $\mathbf{G}_1^2$ (i.e. when the value of the last entry is $[r''^*]_2$) and $\boldsymbol{\gamma} \in \mathbb{Z}_q^{(4v+3)\times 1}$ and $\mathbf{M} \in \mathbb{Z}_q^{(4v+3)\times(4v+4)}$ are defined in Fig. 1.

The matrix $\mathbf{M}$ in Fig. 1 describes all the constraints on the $(4v+4)$ unknowns represented by $\boldsymbol{\alpha}$, that can be derived from the information in the adversary's view in game $\mathbf{G}_1^2$. Notice that $\mathbf{M}$ includes the constraints on $\boldsymbol{\alpha}$ arising from the last v components of the challenge ciphertext ψ^*. Moreover, since we are assuming that the "special" New-period operation took place during the execution of step 7. of the attack game, and that the adversary never revokes the users she corrupts, the identities $z_1^*, \ldots, z_v^*$ specified in the public key PK^{t^*} that is used to create the challenge ciphertext are all different from the identities $x_1, \ldots, x_v$ of the corrupted users, so that $\mathbf{M}$ has full rank, provided that $r^* \neq r'^*$ and $w \neq 0$.

As soon as we fix a value for $\boldsymbol{V}$, the first $4v+2$ entries of vector $\boldsymbol{\gamma}$ and the first $2v+2$ rows of matrix $\mathbf{M}$ are completely fixed, but $\boldsymbol{\alpha}$ is still distributed uniformly and independently at random in $\mathbb{Z}_q^{(4v+4)\times 1}$. If we additionally fix the value of the first $(2v+2)$ components of $[\boldsymbol{\beta}]_2$, the initial public key PK^0 is fixed (in fact, the public key $PK^{\hat{t}}$ resulting from the "special" New-period operation gets fixed, too); it follows that the first identity x_1 that $\mathcal{A}$ chooses to corrupt is also fixed and thus the $(2v+3)$th row of $\mathbf{M}$ is determined. Fixing also a value for $\mathbf{A}_1$ (which is the $(2v+3)$th entry of $[\boldsymbol{\beta}]_2$), the value of $\mathbf{B}_1$ is fixed too, so that all the information on which the adversary can base her choice of x_2 is fixed, and thus the $(2v+4)$th row of $\mathbf{M}$ is determined as well. By a similar reasoning, it follows that fixing the first $(2v+\ell+2)$ entries of $[\boldsymbol{\beta}]_2$ determines the $(2v+\ell+3)$th row of $\mathbf{M}$, for $\ell = 1, \ldots, v$. In particular, fixing the first $3v+2$ entries of $[\boldsymbol{\beta}]_2$ fixes all the information that adversary $\mathcal{A}$ sees before asking for her challenge: thus, the identities $z_1^*, \ldots, z_v^*$, as well as the two messages M_0, M_1 chosen by $\mathcal{A}$ are determined, so that all the remaining rows of $\mathbf{M}$, as well as the last entry of $\boldsymbol{\gamma}$ get fixed, too. Hence, by Lemma 1, we can conclude that the conditional distribution of $[r''^*]_2$, with respect to $\boldsymbol{V}$ and to all other components of $[\boldsymbol{\beta}]_2$, is uniform over $\mathbb{Z}_q$, from which it follows that Eq. (17) holds in this case, too.

Game $\mathbf{G}_1^4$. Game $\mathbf{G}_1^4$ is defined to be identical to $\mathbf{G}_2$. Thus, $\mathbf{G}_1^4$ differs from $\mathbf{G}_1^3$ only in that the values u^* and u'^* in the challenge ciphertext ψ^* are consistent, rather than being nearly independent, as in game $\mathbf{G}_1^3$. Namely, the values u^* and u'^* are now computed as $u^* \doteq g^{r^*}$ and $u'^* \doteq g'^{r^*}$, for the same random $r^* \in \mathbb{Z}_q$. It follows that any difference in behavior between games $\mathbf{G}_1^3$ and $\mathbf{G}_1^4$ can be used to distinguish Diffie-Hellman tuples from totally random tuples. Hence,

$$\left|\Pr[T_1^4] - \Pr[T_1^3]\right| \leq \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{18}$$

Combining Eqs. (15), (16), (17) and (18), the thesis follows. □

6 Dealing with traceability

The goal of a tracing algorithm is to obtain the identity of at least one of the pirates who colluded in creating a given "pirate decoder" D which, as in previous work, is assumed to be stateless. In this section we present a formal model for traceability and two tracing algorithms that can be integrated within the scheme described in Sect. 4.

The first method, a *black-box algorithm*, repeatedly calls a *black-box confirmation* subroutine that, given a pirate decryption device and a subset of at most m suspected users,[1] checks whether the suspected set includes all the secret keys that were used to generate the pirate device, and if so outputs the identity of one of the traitors.

The second method, a *non-black-box algorithm*, receives as input a "valid" key extracted from a pirate device which was constructed using the keys of at most m users and deterministically recovers the identities of all the traitors.

6.1 Model for traceability

The traceability adversary operates similarly to the window adversary described in Sect. 5. Namely, after receiving the initial public key of the system, adversary $\mathcal{A}$ can interleave (in any adaptively-chosen order) up to m Join queries, upon which $\mathcal{A}$ receives the secret keys of the corresponding users (the *traitors*), and a polynomial number of Revoke queries. Notice that each Revoke will change the public key, and the adversary monitors these changes as well. Also notice that the final set of revoked users is likely very different, and typically disjoint from the set $\mathcal{T}$ of traitors. At the end, $\mathcal{A}$ outputs a pirate decoder D which presumably works well (in some sense more precisely clarified below), with the final public key $PK_{\mathcal{A}}$.

6.1.1 Formal model for traceability adversary

The above attack scenario can be formalized in terms of the following traceability adversary attack game $\mathbf{G}_{\mathsf{trt}}^m(1^k)$, played between a challenger and the adversary.

1. Let $\langle PK, MSK\rangle := \mathsf{Setup}(1^k)$
2. Let $L := 0, \mathcal{T} := \emptyset$
3. Let $state := \langle L, PK, MSK, \mathcal{T}\rangle$
4. $\mathsf{D} \overset{R}{\leftarrow} \mathcal{A}^{\mathsf{Join}(state,\cdot),\mathsf{Revoke}(1,state,\cdot)}(state.PK)$
5. If $|\mathcal{T}| > m$ then exit
6. Parse $state$ as $\langle L, PK_{\mathcal{A}}, MSK_{\mathcal{A}}, \mathcal{T}\rangle$
7. Output $\langle \mathsf{D}, PK_{\mathcal{A}}, MSK_{\mathcal{A}}, \mathcal{T}\rangle$

The definitions of the Join and Revoke oracles is the same as in Sect. 5.1, except that the role of the set Corr is now played by the set $\mathcal{T}$.

Definition 8. *For any public key PK, define the success probability of a decoder* D *as:*

$$\mathsf{Succ}_{PK}(\mathsf{D}) \doteq \Pr[M' = M \mid M \overset{R}{\leftarrow} \mathcal{G}; \psi^* \overset{R}{\leftarrow} \mathcal{E}(PK, M); M' \overset{R}{\leftarrow} \mathsf{D}(\psi^*)]$$

where the probability is over the random choice of M, the randomness used to create the challenge ciphertext ψ^* and the coin tosses of D.

Notice that the pirate decoder D expects to receive a ciphertext created under the public key $PK_{\mathcal{A}}$, but the quantity

[1] Recall, m denotes the *collusion* threshold, and should not be confused with the *revocation* threshold v defined in Sect. 4; e.g., in our schemes $m = \lfloor \frac{v}{2} \rfloor$.

$\mathsf{Succ}_{PK}(\mathsf{D})$ can be defined for any public key anyway. Clearly, if D could notice the change, then it could stop working properly: in this case we can assume that it outputs a message $M' \neq M$.

The job of the tracing algorithm is to find one or all of the (at most) m traitors whose keys were used by $\mathcal{A}$ in building D. The precise security guarantees depend on whether tracing is black-box or not. We describe both tracing methods in Sect. 6.2 and 6.3, respectively.

6.2 Black-box tracing

In the black-box model, the tracing algorithm is only allowed to query the pirate decoder D on a polynomial number of a random-looking ciphertexts, and from the plain observation of D's input/output behavior, the tracing algorithm should successfully identify (at least) one of the traitors.

This form of tracing is the most desirable, as it can be applied to any stateless pirate decoder. Similarly to previous work [3, 25, 28] though, the algorithm presented below only achieves a weaker variant of black-box tracing, called *black-box confirmation*. Informally, a black-box confirmation algorithm is a subroutine that tests whether a given set Susp of at most m suspected users does include all the traitors that cooperated to construct a given pirate decoder D and if so, it outputs at least one such pirate. On a pessimistic note, this means that our tracing algorithm might have to go through all m-element subsets of the user universe $\mathcal{U}$ to do full-fledged tracing. However, we point out that: (1) in many cases a lot of partial information about the set of corrupted users makes the search space dramatically smaller; (2) all previous public-key traitor tracing schemes suffer from the same problem; (3) as observed in [19], the problem seems to be inherent to this setting.

However, we significantly improve upon previous black-box confirmation algorithms in the following respects: (1) formal modeling of the problem; (2) our algorithm allows the adversary to *adaptively* corrupt players before building the pirate decoder; (3) our algorithm can be successfully applied to pirate decoders that work on at least an ε-fraction of correctly formed messages (rather than with probability 1), where ε is the desired threshold below which the decoder is considered "useless" (following the "threshold tracing" approach of [24]).

Definition 9 (Black-Box Confirmation Algorithm). *A* Black-Box Confirmation *(*BBC*) algorithm is a probabilistic, polynomial time oracle machine taking as oracle input a pirate decoder* D*, and as regular input a public key PK, the corresponding master secret key MSK, and a set* Susp *of suspected traitors. Its output is either a user's identity i or the special symbol* ?.

Definition 10 (ε-Black-Box Confirmation Property). *Let $\mathcal{A}$ be any probabilistic, polynomial-time adversary, and let $\langle \mathsf{D}, PK_{\mathcal{A}}, MSK_{\mathcal{A}}, \mathcal{T}\rangle$ be the output resulting from the adversary playing the traceability attack game $\boldsymbol{G}^m_{\mathsf{trt}}(1^k)$ with the challenger. A Black-Box Confirmation algorithm* BBC *satisfies* ε-Black-Box Confirmation *if, for any* PPT *adversary $\mathcal{A}$ playing the $\boldsymbol{G}^m_{\mathsf{trt}}(1^k)$ game, the following two properties hold with all but negligible probability:*

- **Confirmation:** *whenever $\mathcal{T} \subseteq$ Susp, then the output of* $\mathsf{BBC}^{\mathsf{D}}(PK_{\mathcal{A}}, MSK_{\mathcal{A}}, \mathsf{Susp})$ *is some identity $i \in \mathcal{T}$.*
- **Soundness:** *whenever* $\mathsf{BBC}^{\mathsf{D}}(PK_{\mathcal{A}}, MSK_{\mathcal{A}}, \mathsf{Susp})$ *outputs $i \neq ?$, then $i \in \mathcal{T}$.*

6.2.1 Black-box confirmation algorithm

At a high level, our black-box confirmation algorithm BBC works as follows. Based on the current set I of suspected users (initially set to Susp) and using the master secret key $MSK_{\mathcal{A}}$, it modifies the public key $PK_{\mathcal{A}}$ into a fake public key $PK(I)$. It then estimates the probability

$$\delta(I) \doteq \mathsf{Succ}_{PK(I)}(\mathsf{D})$$

by observing the behavior of D when fed with encryptions of the form $\mathcal{E}(PK(I), M)$, for random messages M. The Chernoff bound implies that the latter estimation can be done quickly and accurately (by computing statistics from repeated sampling), provided $\delta(I)$ is "large enough" (specifically, at least ε/m). Now, BBC takes any index $i \in I$, and accurately estimates $\delta(I \setminus \{i\})$. If the difference between $\delta(I)$ and $\delta(I \setminus \{i\})$ is "non-trivial" (specifically, at least $\varepsilon/2m$), it proclaims i as a traitor. Otherwise, it sets $I := I \setminus \{i\}$, and repeats the entire procedure until $I = \emptyset$ (in which case it outputs ?).

The last main detail to be filled in is how the algorithm generates the fake public key $PK(I)$. Recall from Sect. 4 that the master secret key $MSK_{\mathcal{A}}$ consists of two random polynomials over $\mathbb{Z}_q^v[x]$. Let $\bar{t}$ be the total number of New-period operations that occur during the entire game, and for $t = 1, \ldots, \bar{t}$, let $c_1^t, \ldots, c_{2v+2}^t$ be the coefficients of the randomizing polynomials $D^t(\cdot)$ and $E^t(\cdot)$ used in the tth New-period operation. For $t = 1, \ldots, \bar{t}$, and $j = 1, \ldots, 2v+2$, let r_j^t be the randomness used to encrypt (the encoding of) coefficient c_j^t in the tth reset message. By Eq. (8), $(A^{\bar{t}}(\cdot), B^{\bar{t}}(\cdot))$ denotes the master secret key corresponding to the public key $PK_{\mathcal{A}}$. Given the set I, we create two polynomials $A'(\cdot)$ and $B'(\cdot)$ uniformly distributed over $\mathbb{Z}_q^v[x]$ except they agree with $A^{\bar{t}}(\cdot)$ and $B^{\bar{t}}(\cdot)$ on points in I:

$$A'(x_s) = A^{\bar{t}}(x_s) \quad B'(x_s) = B^{\bar{t}}(x_s), \ \forall s \in I.$$

Notice that, since $|I| \leq m \leq v/2$, this creates no problem. We then create the public key $PK(I)$ as if the master secret key were $MSK' = (A'(\cdot), B'(\cdot))$ rather than $MSK_{\mathcal{A}} = (A^{\bar{t}}(\cdot), B^{\bar{t}}(\cdot))$. Specifically, we define

$$PK(I) \doteq (g, g', y', \langle z_\ell, h'_\ell \rangle_{\ell=1}^v).$$

where $y' \doteq g^{A'(0)} g'^{B'(0)}$, and $h'_\ell \doteq g^{A'(z_\ell)} g'^{B'(z_\ell)}$, for $\ell = 1, \ldots, v$.

6.2.2 Correctness of black-box tracing

The correctness of the black-box tracing algorithm described above follows from Theorem 2 and Theorem 3 stated below. Theorem 2 implies that if the decoder was useful at the start (i.e., $\mathsf{Succ}_{PK_{\mathcal{A}}}(\mathsf{D}) \geq \varepsilon$) and $\mathcal{T} \subseteq$ Susp, then the decoder cannot "notice" that $PK_{\mathcal{A}}$ was changed to $PK(\mathsf{Susp})$, i.e. $\delta(\mathsf{Susp}) \gtrsim \varepsilon$.[2] Coupled with the obvious fact that $\delta(\emptyset)$ is negligible (since M is encrypted with a totally random one-time pad), we see that there must be a time when $\delta(I)$ changes by a non-trivial amount (i.e., at least by $\varepsilon/2m$) when we remove some $i \in I$. This i will then be output by our algorithm, and since i cannot be an innocent user (by Theorem 3 below), i must be one of the traitors. This shows the confirmation property.

Theorem 2. *Under the* DDH *assumption, if* $\mathcal{T} \subseteq \mathsf{Susp}$ *and* $|\mathsf{Susp}| \leq v$*, then* $|\delta(\mathsf{Susp}) - \mathsf{Succ}_{PK_{\mathcal{A}}}(\mathsf{D})|$ *is negligible.*

Proof. We again follow the structural approach of defining a sequence of "indistinguishable" games $\mathbf{G}_0, \mathbf{G}_1, \ldots$, all operating over the same underlying probability space. Each of these games consists of the BBC algorithm sending a ciphertext ψ^* to the pirate decoder D; different games only differs in the way ψ^* is computed. In the original game $\mathbf{G}_0$, the goal of the decoder D is to output a message M' which is D's best guess at the random message M encrypted within ψ^*; for each game $\mathbf{G}_j$, let T_j be the event that $M = M'$ in $\mathbf{G}_j$.

Game $\mathbf{G}_0$: This game defines the probability $\mathsf{Succ}_{PK_{\mathcal{A}}}(\mathsf{D})$. In this game, the BBC algorithm takes as input the public key $PK_{\mathcal{A}}$, the corresponding master secret key $MSK_{\mathcal{A}}$ and a set Susp of suspected users; it then chooses a message $M \overset{R}{\leftarrow} \mathcal{G}$ and, using the public key $PK_{\mathcal{A}}$, encrypts it as follows:

$$\begin{aligned}
&E1. \quad r \overset{R}{\leftarrow} \mathbb{Z}_q\\
&E2. \quad u \leftarrow g^r\\
&E3. \quad u' \leftarrow g'^r\\
&E4. \quad u'' \leftarrow g^{A^{\bar{t}}(0)r} g'^{B^{\bar{t}}(0)r} M\\
&E5. \quad u_\ell \leftarrow g^{A^{\bar{t}}(z_\ell)r} g'^{B^{\bar{t}}(z_\ell)r}, \quad \ell = 1, \ldots, v\\
&E6. \quad \psi^* \leftarrow \langle u, u', u'', \langle z_1, u_1\rangle, \ldots, \langle z_v, u_v\rangle\rangle
\end{aligned}$$

By definition, we have that

$$\Pr[T_0] = \mathsf{Succ}_{PK_{\mathcal{A}}}(\mathsf{D}). \tag{19}$$

Game $\mathbf{G}_1$: Game $\mathbf{G}_1$ is identical to game $\mathbf{G}_0$, except that in game $\mathbf{G}_1$ steps $E4$ and $E5$ are substituted with:

$$\begin{aligned}
&E4'. \quad u'' \leftarrow u^{A^{\bar{t}}(0)} u'^{B^{\bar{t}}(0)} M\\
&E5'. \quad u_\ell \leftarrow u^{A^{\bar{t}}(z_\ell)} u'^{B^{\bar{t}}(z_\ell)}, \quad \ell = 1, \ldots, v
\end{aligned}$$

Notice that the point of these changes is just to make explicit any functional dependency of ψ^* on the quantities u and u'. Since we just made a conceptual change, it clearly holds that

$$\Pr[T_1] = \Pr[T_0]. \tag{20}$$

Game $\mathbf{G}_2$: To define game $\mathbf{G}_2$, we make more changes to the encryption algorithm as follows:

$$\begin{aligned}
&E1'. \quad r \overset{R}{\leftarrow} \mathbb{Z}_q; \quad r' \overset{R}{\leftarrow} \mathbb{Z}_q \setminus \{r\}\\
&E3'. \quad u' \leftarrow g'^{r'}
\end{aligned}$$

[2] The relation $\gtrsim$ is meant to indicate that $\delta(\mathsf{Susp})$ is greater than ε minus negligible terms.

Notice that while in game $\mathbf{G}_1$ the value u and u' are obtained using the same value r, in game $\mathbf{G}_2$ they are nearly independent, being subject only to $r \neq r'$. Therefore, using a standard reduction argument, any non-negligible difference in behavior between games $\mathbf{G}_1$ and $\mathbf{G}_2$ can be used to construct a PPT adversary able to distinguish Diffie-Hellman tuples from totally random tuples with non-negligible advantage. Hence,

$$\left|\Pr[T_2] - \Pr[T_1]\right| \leq \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{21}$$

Game $\mathbf{G}_3$: To turn game $\mathbf{G}_2$ into game $\mathbf{G}_3$, we consider the set Susp and construct the public key $PK(\mathsf{Susp})$ as described above; specifically, two random polynomials $A'(\cdot)$ and $B'(\cdot)$ are chosen such that

$$A'(x_s) = A^{\bar{t}}(x_s) \quad B'(x_s) = B^{\bar{t}}(x_s), \ \forall s \in \mathsf{Susp} \tag{22}$$

and $PK(\mathsf{Susp})$ is set to be:

$$PK(\mathsf{Susp}) \doteq \langle g, g', g^{A'(0)} g'^{B'(0)}, \langle z_\ell, g^{A'(z_\ell)} g'^{B'(z_\ell)}\rangle_{\ell=1}^{v}\rangle.$$

Then, we change steps $E4'$ and $E5'$ of the encryption algorithm of game $\mathbf{G}_2$ as follows:

$$\begin{aligned}
&E4''. \quad u'' \leftarrow u^{A'(0)} u'^{B'(0)} M\\
&E5''. \quad u_\ell \leftarrow u^{A'(z_\ell)} u'^{B'(z_\ell)}, \quad \ell = 1, \ldots, v
\end{aligned}$$

Using the technique outlined in Sect. 5.2, in Lemma 4 below, we show that

$$\Pr[T_3] = \Pr[T_2]. \tag{23}$$

Game $\mathbf{G}_4$: In game $\mathbf{G}_4$, we "undo" the changes of game $\mathbf{G}_2$, restoring lines $E1'$ and $E3'$ of the encryption oracle to their original values:

$$\begin{aligned}
&E1''. \quad r \overset{R}{\leftarrow} \mathbb{Z}_q\\
&E3''. \quad u' \leftarrow g'^r
\end{aligned}$$

Notice that in game $\mathbf{G}_4$ the value u and u' are again obtained using the same value r, whereas in game $\mathbf{G}_3$ they are nearly independent, being subject only to $r \neq r'$. Therefore, using a standard reduction argument, any non-negligible difference in behavior between games $\mathbf{G}_3$ and $\mathbf{G}_4$ can be used to construct a PPT adversary able to distinguish Diffie-Hellman tuples from totally random tuples with non-negligible advantage. Hence,

$$\left|\Pr[T_4] - \Pr[T_3]\right| \leq \mathsf{AdvDDH}_{\mathcal{G}}(k). \tag{24}$$

Finally, observe that in $\mathbf{G}_4$ the encryption of the random message M is obtained using the public key $PK(\mathsf{Susp})$: thus, game $\mathbf{G}_4$ is exactly the game which defines the probability $\delta(\mathsf{Susp})$ i.e.,

$$\Pr[T_4] = \delta(\mathsf{Susp}). \tag{25}$$

Combining Eqs. (19), (20), (21), (23), (24) and (25), we can conclude that $\mathcal{A}$ has only a negligible chance to tell whether the message M was encrypted under the public keys $PK_{\mathcal{A}}$ or $PK(\mathsf{Susp})$; more precisely:

$$|\delta(\mathsf{Susp}) - \mathsf{Succ}_{PK_{\mathcal{A}}}(\mathsf{D})| \leq 2\ \mathsf{AdvDDH}_{\mathcal{G}}(k).$$

□

Lemma 4. $\Pr[T_3] = \Pr[T_2]$

Proof. To prove the lemma, we consider all the quantities that can affect event T_2 in game $\mathbf{G}_2$ and event T_3 in game $\mathbf{G}_3$, and then we show that these quantities are distributed according to the same joint distribution in both games.

Consider the quantity:

$$\boldsymbol{V} \doteq (\mathsf{Coins}_{\mathcal{A}}, \mathsf{Coins}_{\mathsf{D}}, w, M, r, r', \{\{c_j^t, r_j^t\}_{j=1}^{2v+2}\}_{t=1}^{\bar{t}})$$

where $\mathsf{Coins}_{\mathcal{A}}$ denotes the coin tosses of $\mathcal{A}$, $\mathsf{Coins}_{\mathsf{D}}$ denotes the coin tosses of D, $w \doteq \log_g g'$, M is the random message encrypted within ψ^*, r and r' are the random values used to create ψ^*, and

$$\{\{c_j^t, r_j^t\}_{j=1}^{2v+2}\}_{t=1}^{\bar{t}}$$

represents all the randomness used in the $\bar{t}$ New-period operations that took place during the $\mathbf{G}_{\mathsf{trt}}^m(1^k)$ attack game.

The remaining randomness used during games $\mathbf{G}_2$ and $\mathbf{G}_3$ consists of the $4v+4$ coefficients of the polynomials $A^0(\cdot)$, $B^0(\cdot)$ (chosen by the Setup algorithm in step 1. of the $\mathbf{G}_{\mathsf{trt}}^m(1^k)$ attack game) and $A'(\cdot)$, $B'(\cdot)$ (used in game $\mathbf{G}_3$). This randomness can be represented with the vector

$$\boldsymbol{\alpha} \doteq (a_0, a_1, \ldots, a_v, b_0, b_1, \ldots, b_v)^T$$

which is uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$, and the vector

$$\boldsymbol{\alpha}' \doteq (a_0', a_1', \ldots, a_v', b_0', b_1', \ldots, b_v')^T$$

which is uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$, subject to the constraints arising from imposing Eq. (22).

Let $\mathcal{T} = \{t_1, \ldots, t_m\}$ be the set of traitors and set

$$\mathbf{A}_j \doteq A^{\bar{t}}(x_{t_j}) \quad \mathbf{B}_j \doteq B^{\bar{t}}(x_{t_j}),\ j = 1, \ldots, m.$$

Notice that, since $\mathcal{T} \subseteq \mathsf{Susp}$, for $j = 1, \ldots, m$, it holds that $\mathbf{A}_j = A'(x_{t_j})$ and $\mathbf{B}_j = B'(x_{t_j})$.

Consider the quantity $\bar{\boldsymbol{\beta}} \in \mathbb{Z}_q^{(v+m+1)\times 1}$ defined as:

$$\bar{\boldsymbol{\beta}} \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \mathbf{A}_1, \ldots, \mathbf{A}_m)^T$$

where $\mathbf{X}_0 \doteq A^0(0) + wB^0(0)$, and $\mathbf{X}_\ell \doteq A^0(\ell) + wB^0(\ell)$, for $\ell = 1, \ldots, v$.

It is clear by inspection that all the information in the view of the adversary $\mathcal{A}$ during the attack game $\mathbf{G}_{\mathsf{trt}}^m(1^k)$ is completely determined by $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$. In particular, the initial public key PK^0 is fixed by $\bar{\boldsymbol{\beta}}$ and w, and the secret keys of the traitors are determined by the choice of $\bar{\boldsymbol{\beta}}$, $\mathsf{Coins}_{\mathcal{A}}$ and w.

Besides the information in $\mathcal{A}$'s view, which is completely determined by the value of $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, the only other quantity affecting D's behavior is the ciphertext ψ^*. This ciphertext is computed differently in games $\mathbf{G}_2$ and $\mathbf{G}_3$: for the sake of clarity, we will denote with $[\psi^*]_2$ and $[\psi^*]_3$ the value of such quantity in game $\mathbf{G}_2$ and $\mathbf{G}_3$, respectively. We now want to show that, conditioned on all the other information in D's view, $[\psi^*]_2$ and $[\psi^*]_3$ are distributed according to the same distribution in the two games.

In game $\mathbf{G}_2$, the ciphertext $[\psi^*]_2$ sent to the decoder is completely determined by $\boldsymbol{V}$, $\bar{\boldsymbol{\beta}}$ and by the v-degree polynomial $X^{\bar{t}}(\cdot) \doteq rA^{\bar{t}}(\cdot) + wr'B^{\bar{t}}(\cdot)$. Similarly, in game $\mathbf{G}_3$, the ciphertext $[\psi^*]_3$ is completely determined by $\boldsymbol{V}$, $\bar{\boldsymbol{\beta}}$ and by the v-degree polynomial $X'(\cdot) \doteq rA'(\cdot) + wr'B'(\cdot)$. Moreover, $[\psi^*]_2$ depends on $\boldsymbol{V}$, $\bar{\boldsymbol{\beta}}$ and $X^{\bar{t}}(\cdot)$ according to the same functional dependence of $[\psi^*]_3$ upon $\boldsymbol{V}$, $\bar{\boldsymbol{\beta}}$ and $X'(\cdot)$. Therefore, to prove the lemma, it suffices to show that, conditioning on any fixed values of $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, $X^{\bar{t}}(\cdot)$ and $X'(\cdot)$ are distributed according to the same conditional probability distribution; namely, both are random polynomials over $\mathbb{Z}_q^v[x]$, subject to the constraint that their value at x_{t_j} is $r\mathbf{A}_j + wr'\mathbf{B}_j$, for $j = 1, \ldots, m$.

By Lagrange interpolation, $X^{\bar{t}}(\cdot)$ can be identified with its value at the points 0, 1, …, $v-m$, x_{t_1}, …, x_{t_m}; define

$$\mathbf{X}_\ell^{\bar{t}} \doteq X^{\bar{t}}(\ell),\ \ell = 0, \ldots, v-m$$

and

$$\mathbf{X}_{v-m+j}^{\bar{t}} \doteq X^{\bar{t}}(x_{i_j}),\ j = 1, \ldots, m.$$

Similarly, we can also identify $X'(\cdot)$ with its value at the same $v+1$ points; define

$$\mathbf{X}_\ell' \doteq X'(\ell),\ \ell = 0, \ldots, v-m$$

and

$$\mathbf{X}_{v-m+j}' \doteq X'(x_{t_j}),\ j = 1, \ldots, m.$$

As noticed above, the assumption that $\mathcal{T} \subseteq \mathsf{Susp}$ implies that for $j = 1, \ldots, m$:

$$A^{\bar{t}}(x_{t_j}) = A'(x_{t_j}) = \mathbf{A}_j, \quad B^{\bar{t}}(x_{t_j}) = B'(x_{t_j}) = \mathbf{B}_j.$$

Therefore, it follows that

$$\mathbf{X}_{v-m+j} = \mathbf{X}_{v-m+j}',\ j = 1, \ldots, m. \tag{26}$$

It only remains to be proven that, conditioning on fixed values of $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, the tuple $\mathbf{X}_0^{\bar{t}}$, …, $\mathbf{X}_{v-m}^{\bar{t}}$ and the tuple $\mathbf{X}_0'$, …, $\mathbf{X}_{v-m}'$ are distributed according to the same joint conditional distribution. (Notice that fixing a value for $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, immediately fixes a value for the tuple $\mathbf{X}_{v-m+j}^{\bar{t}}$, $j = 1, \ldots, m$, which by Eq. (26) is equal to $\mathbf{X}_{v-m+j}'$, $j = 1, \ldots, m$.)

Recall that, in game $\mathbf{G}_3$, the polynomials $A'(\cdot)$ and $B'(\cdot)$ are chosen uniformly at random from $\mathbb{Z}_q^v[x]$, independently from anything else, but subject to the constraints in Eq. (22). It follows that the polynomial $X'(\cdot) = rA'(\cdot) + wr'B'(\cdot)$ is also random in $\mathbb{Z}_q^v[x]$, subject to the constraint that its value at x_s is

$$rA^{\bar{t}}(x_s) + wr'B^{\bar{t}}(x_s),\ \forall s \in \mathsf{Susp}.$$

Therefore, conditioning on fixed values of $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, the tuple $\mathbf{X}_0', \ldots, \mathbf{X}_{v-m}'$ is distributed uniformly at random in $\mathbb{Z}_q^{(v-m+1)\times 1}$. Hence, it suffices to show that, for $\ell = 0, \ldots, v-m$, the conditional distribution of $\mathbf{X}_\ell^{\bar{t}}$ with respect to $\boldsymbol{V}$, $\bar{\boldsymbol{\beta}}$ and $\mathbf{X}_0^{\bar{t}}, \ldots, \mathbf{X}_{\ell-1}^{\bar{t}}$ is uniform over $\mathbb{Z}_q$. To this aim, fix $\ell \in \{0, \ldots, v-m\}$, and consider the following matrix equation:

$$\boldsymbol{\beta}_\ell = \mathbf{M}_\ell \cdot \boldsymbol{\alpha} + \boldsymbol{\gamma}_\ell$$

where $\boldsymbol{\beta}_\ell \in \mathbb{Z}_q^{(v+m+\ell+2)\times 1}$ is the vector

$$\boldsymbol{\beta}_\ell \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \mathbf{A}_1, \ldots, \mathbf{A}_m, \mathbf{X}_0^{\bar{t}}, \mathbf{X}_1^{\bar{t}}, \ldots, \mathbf{X}_\ell^{\bar{t}})^T,$$

$\boldsymbol{\gamma}_\ell \in \mathbb{Z}_q^{(v+m+\ell+2)\times 1}$ is the vector

$$\boldsymbol{\gamma}_\ell \doteq \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ D^{0,\bar{t}}(x_{t_1}) \\ \vdots \\ D^{0,\bar{t}}(x_{t_m}) \\ rD^{0,\bar{t}}(0) + wr'E^{0,\bar{t}}(0) \\ rD^{0,\bar{t}}(1) + wr'E^{0,\bar{t}}(1) \\ \vdots \\ rD^{0,\bar{t}}(\ell) + wr'E^{0,\bar{t}}(\ell) \end{pmatrix}$$

and $\mathbf{M}_\ell \in \mathbb{Z}_q^{(v+m+\ell+2)\times(2v+2)}$ is the matrix

$$\mathbf{M}_\ell \doteq \begin{pmatrix} 1 & 0 & \dots & 0 & w & 0 & \dots & 0 \\ 1 & 1 & \dots & 1 & w & w & \dots & w \\ & & \vdots & & & \vdots & & \\ 1 & v & \dots & v^v & w & wv & \dots & wv^v \\ 1 & x_{t_1} & \dots & x_{t_1}^v & 0 & 0 & \dots & 0 \\ & & \vdots & & & \vdots & & \\ 1 & x_{t_m} & \dots & x_{t_m}^v & 0 & 0 & \dots & 0 \\ r & 0 & \dots & 0 & wr' & 0 & \dots & 0 \\ r & r & \dots & r & wr' & wr' & \dots & wr' \\ & & \vdots & & & \vdots & & \\ r & r\ell & \dots & r\ell^v & wr' & wr'\ell & \dots & wr'\ell^v \end{pmatrix}$$

By inspection, it is possible to see that the rows of matrix $\mathbf{M}_\ell$ are linearly independent, provided that $r \neq r'$ and $w \neq 0$: thus, the rank of $\mathbf{M}_\ell$ is $v + m + \ell + 2$. As soon as we fix $\boldsymbol{V}$, vector $\boldsymbol{\gamma}_\ell$ and the first $v + 1$ rows of $\mathbf{M}_\ell$ are determined, but $\boldsymbol{\alpha}$ is still distributed uniformly and independently at random in $\mathbb{Z}_q^{(2v+2)\times 1}$. Similarly to the proof of Lemma 3, it is also possible to show that fixing the first $v + j$ entries of $\bar{\boldsymbol{\beta}}$ determines the $(v+j+1)$th row of $\mathbf{M}_\ell$, for $j = 1, \dots, m$; and that moreover, fixing the first $v + m + 1$ entries of $\bar{\boldsymbol{\beta}}$ determines all the remaining rows of $\mathbf{M}_\ell$.

Hence, by Lemma 1, we can conclude that the conditional distribution of $\mathbf{X}_\ell^{\bar{t}}$ with respect to $(\boldsymbol{V}, \bar{\boldsymbol{\beta}}, \mathbf{X}_0^{\bar{t}}, \dots, \mathbf{X}_{\ell-1}^{\bar{t}})$ is uniform over $\mathbb{Z}_q$, $\forall \ell \in \{0, \dots, v - m\}$. In other words, the value of $X^{\bar{t}}(\cdot)$ at any point is uniformly random, subject to the constraint

$$\mathbf{X}^{\bar{t}}(x_{t_j}) = r\mathbf{A}_j + wr'\mathbf{B}_j,\ \forall t_j \in \mathcal{T}.$$

Thus, $(\boldsymbol{V}, \bar{\boldsymbol{\beta}}, X^{\bar{t}}(\cdot)))$ has the same joint distribution as $(\boldsymbol{V}, \bar{\boldsymbol{\beta}}, X'(\cdot))$, completing the proof. □

We now move on to prove the soundness of the BBC algorithm, showing that it can accuse an innocent user with at most negligible probability. Informally this is true because, under the DDH assumption it is impossible to notice if the values $A'(x_i)$ and $B'(x_i)$ (which are unknown to the adversary since i is assumed to be honest), were replaced by random noise $A''(x_i)$ and $B''(x_i)$. Thus, the behavior of the decoder will be the same regardless of whether $PK(I)$ or $PK(I \setminus \{i\})$ was used to encrypt the ciphertext. Since our algorithm only accuses a user i when a sensible change occurs in the decryption capability of the pirate decoder, it follows that an innocent user will be blamed with at most negligible probability.

Theorem 3. *Under the* DDH *assumption, if* $|I| \leq v$ *and* $i \notin \mathcal{T}$*, then* $|\delta(I) - \delta(I \setminus \{i\})|$ *is negligible.*

Proof. Proceeding as in the proof of Theorem 2, we define a sequence of "indistinguishable" games $\mathbf{G}_0, \mathbf{G}_1, \dots$: for each game $\mathbf{G}_j$, let T_j be the event that decoder D correctly decrypts the challenge sent by the BBC algorithm in game $\mathbf{G}_j$.

Game $\mathbf{G}_0$: This game describes the experiment which defines the value of $\delta(I)$. In this game, the decoder D is fed with ciphertexts obtained encrypting random messages under the fake public key $PK(I)$, defined as:

$$PK(I) = \langle g, g', g^{A'(0)}g'^{B'(0)}, \langle z_\ell, g^{A'(z_\ell)}g'^{B'(z_\ell)}\rangle_{\ell=1}^v \rangle$$

where $A'(\cdot)$ and $B'(\cdot)$ are random v-degree polynomials subject to:

$$A'(x_s) = A^{\bar{t}}(x_s) \quad B'(x_s) = B^{\bar{t}}(x_s),\ \forall s \in I. \tag{27}$$

More precisely, the BBC algorithm chooses a random message M and encrypts it as follows:

$E1.\quad r \overset{R}{\leftarrow} \mathbb{Z}_q$
$E2.\quad u \leftarrow g^r$
$E3.\quad u' \leftarrow g'^r$
$E4.\quad u'' \leftarrow g^{A'(0)r}g'^{B'(0)r}M$
$E5.\quad u_\ell \leftarrow g^{A'(z_\ell)r}g'^{B'(z_\ell)r},\quad \ell = 1, \dots, v$
$E6.\quad \psi^* \leftarrow \langle u, u', u'', \langle z_1, u_1\rangle, \dots, \langle z_v, u_v\rangle\rangle$

By definition, we have that:

$$\Pr[T_0] = \delta(I). \tag{28}$$

Game $\mathbf{G}_1$: Game $\mathbf{G}_1$ is identical to game $\mathbf{G}_0$, except that in game $\mathbf{G}_1$ steps $E4$ and $E5$ are substituted with:

$E4'.\quad u'' \leftarrow u^{A'(0)}u'^{B'(0)}M$
$E5'.\quad u_\ell \leftarrow u^{A'(z_\ell)}u'^{B'(z_\ell)},\quad \ell = 1, \dots, v$

Notice that the point of these changes is just to make explicit any functional dependency of ψ^* on the quantities u and u'. Since we just made a conceptual change, it clearly holds that

$$\Pr[T_1] = \Pr[T_0]. \tag{29}$$

Game $\mathbf{G}_2$: Game $\mathbf{G}_2$ is identical to game $\mathbf{G}_1$, except that in game $\mathbf{G}_2$ steps $E1$ and $E3$ are substituted with:

$E1'.\quad r \overset{R}{\leftarrow} \mathbb{Z}_q;\quad r' \overset{R}{\leftarrow} \mathbb{Z}_q \setminus \{r\}$
$E3'.\quad u' \leftarrow g'^{r'}$

Notice that while in game $\mathbf{G}_1$ the value u and u' are obtained using the same value r, in game $\mathbf{G}_2$ they are nearly independent, being subject only to $r \neq r'$. Therefore, using a standard reduction argument, any non-negligible difference in behavior between games $\mathbf{G}_1$ and $\mathbf{G}_2$ can be used to construct a PPT adversary able to distinguish Diffie-Hellman tuples from totally random tuples with non-negligible advantage. Hence,

$$\left|\Pr[T_2] - \Pr[T_1]\right| \leq \mathsf{AdvDDH}_{\mathcal{G}}(k). \qquad (30)$$

Game $\mathbf{G}_3$: To turn game $\mathbf{G}_2$ into game $\mathbf{G}_3$, we consider the set $I \setminus \{i\}$ and construct the public key $PK(I \setminus \{i\})$: two new random v-degree polynomials $A''(\cdot)$ and $B''(\cdot)$ are chosen such that

$$A''(x_s) = A^{\bar{t}}(x_s) \quad B''(x_s) = B^{\bar{t}}(x_s),\ \forall s \in I \setminus \{i\} \qquad (31)$$

and $PK(I \setminus \{i\})$ is set to be:

$$\langle g, g', g^{A''(0)}g'^{B''(0)}, \langle z_\ell, g^{A''(z_\ell)}g'^{B''(z_\ell)}\rangle_{\ell=1}^{v}\rangle.$$

Notice that, by Eqs. (27) and (31), it holds that

$$A''(x_s) = A'(x_s) \quad B''(x_s) = B'(x_s),\ \forall s \in I \setminus \{i\}.$$

Finally, we change steps $E4'$ and $E5'$ of the encryption algorithm as follows:

$$E4''.\ \ u'' \leftarrow u^{A''(0)}u'^{B''(0)}M$$
$$E5''.\ \ u_\ell \leftarrow u^{A''(z_\ell)}u'^{B''(z_\ell)},\quad \ell = 1, \ldots, v$$

Using the technique described in Sect. 5.2, in Lemma 5 below, we show that

$$\Pr[T_3] = \Pr[T_2]. \qquad (32)$$

Game $\mathbf{G}_4$: In game $\mathbf{G}_4$, we "undo" the changes of game $\mathbf{G}_2$, restoring lines $E1$ and $E3$ of the encryption oracle to their original values:

$$E1''.\ \ r \overset{R}{\leftarrow} \mathbb{Z}_q$$
$$E3''.\ \ u' \leftarrow g'^r$$

Notice that in game $\mathbf{G}_4$ the value u and u' are again obtained using the same value r, whereas in game $\mathbf{G}_3$ they are nearly independent, being subject only to $r \neq r'$. Therefore, using a standard reduction argument, any non-negligible difference in behavior between games $\mathbf{G}_3$ and $\mathbf{G}_4$ can be used to construct a PPT adversary able to distinguish Diffie-Hellman tuples from totally random tuples with non-negligible advantage. Hence,

$$\left|\Pr[T_4] - \Pr[T_3]\right| \leq \mathsf{AdvDDH}_{\mathcal{G}}(k). \qquad (33)$$

In game $\mathbf{G}_4$, the encryption of the random message M is obtained using the public key $PK(I \setminus \{i\})$: thus, game $\mathbf{G}_4$ is exactly the game defining $\delta(I \setminus \{i\})$ i.e.,

$$\Pr[T_4] = \delta(I \setminus \{i\}). \qquad (34)$$

By Eqs. (28), (29), (30), (32), (33) and (34), we can conclude that the adversary has only a negligible chance to tell whether the message M was encrypted under $PK(I)$ or $PK(I \setminus \{i\})$; more precisely:

$$|\delta(I) - \delta(I \setminus \{i\})| \leq 2\ \mathsf{AdvDDH}_{\mathcal{G}}(k).$$

□

Lemma 5. $\Pr[T_3] = \Pr[T_2]$

Proof. To prove the lemma, we consider all the quantities that can affect event T_2 in game $\mathbf{G}_2$ and event T_3 in game $\mathbf{G}_3$, and then we show that these quantities are distributed according to the same joint distribution in both games.

Let $\bar{m} \doteq |\mathcal{T} \cap I|$, where $\mathcal{T} = \{t_1, \ldots, t_m\}$ is the set of traitors; without loss of generality assume that $\mathcal{T} \cap I = \{t_1, \ldots, t_{\bar{m}}\}$. Also, set

$$\mathbf{A}_j \doteq A^{\bar{t}}(x_{t_j}) \quad \mathbf{B}_j \doteq B^{\bar{t}}(x_{t_j}),\ j = 1, \ldots, m.$$

Notice that, since $i \notin \mathcal{T}$, for $1 \leq j \leq \bar{m}$ it also holds that:

$$\mathbf{A}_j = A'(x_{t_j}) = A''(x_{t_j}) \quad \mathbf{B}_j = B'(x_{t_j}) = B''(x_{t_j}).$$

Consider the quantity:

$$\boldsymbol{V} \doteq (\mathsf{Coins}_{\mathcal{A}}, \mathsf{Coins}_{\mathsf{D}}, w, M, r, r', \{\{c_j^t, r_j^t\}_{j=1}^{2v+2}\}_{t=1}^{\bar{t}})$$

where $\mathsf{Coins}_{\mathcal{A}}$ denotes the coin tosses of $\mathcal{A}$, $\mathsf{Coins}_{\mathsf{D}}$ denotes the coin tosses of D, $w \doteq \log_g g'$, $\mathbf{X}_\ell \doteq (A^0(\ell) + B^0(\ell))$ for $\ell = 0, \ldots, v$, M is the random message encrypted within ψ^*, r and r' are the random values used to create ψ^*, and

$$\{\{c_j^t, r_j^t\}_{j=1}^{2v+2}\}_{t=1}^{\bar{t}}$$

represents all the randomness used in the $\bar{t}$ New-period operations that took place during the attack game $\mathbf{G}_{\mathsf{trt}}^m(1^k)$.

The remaining randomness used during games $\mathbf{G}_2$ and $\mathbf{G}_3$ consists of the $6v + 6$ coefficients of the polynomials $A^0(\cdot)$, $B^0(\cdot)$ (chosen by the Setup algorithm in step 1. of the $\mathbf{G}_{\mathsf{trt}}^m(1^k)$ attack game), $A'(\cdot)$, $B'(\cdot)$ (used in game $\mathbf{G}_2$), and $A''(\cdot)$, $B''(\cdot)$ (used in game $\mathbf{G}_3$). This randomness can be represented with the vector

$$\boldsymbol{\alpha} \doteq (a_0, a_1, \ldots, a_v, b_0, b_1, \ldots, b_v)^T$$

which is uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$, the vector

$$\boldsymbol{\alpha}' \doteq (a_0', a_1', \ldots, a_v', b_0', b_1', \ldots, b_v')^T$$

which is uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$, subject to the constraints arising from imposing Eq. (27), and the vector

$$\boldsymbol{\alpha}'' \doteq (a_0'', a_1'', \ldots, a_v'', b_0'', b_1'', \ldots, b_v'')^T$$

which is uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$, subject to the constraints arising from imposing Eq. (31).

Consider the quantity $\bar{\boldsymbol{\beta}} \in \mathbb{Z}_q^{(v+\bar{m}+1)\times 1}$ defined as:

$$\bar{\boldsymbol{\beta}} \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \mathbf{A}_1, \ldots, \mathbf{A}_{\bar{m}})^T$$

where $\mathbf{X}_0 \doteq A^0(0) + wB^0(0)$, and $\mathbf{X}_\ell \doteq A^0(\ell) + wB^0(\ell)$, for $\ell = 1, \ldots, v$.

It is clear by inspection that all the information in the view of the adversary $\mathcal{A}$ during the attack game $\mathbf{G}_{\mathsf{trt}}^m(1^k)$ is completely determined by $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$. In particular, the initial public key PK^0 is fixed by $\boldsymbol{V}$, and the secret keys of the traitors are determined by the choice of $\bar{\boldsymbol{\beta}}$, $\mathsf{Coins}_{\mathcal{A}}$ and w.

Besides the information in $\mathcal{A}$'s view, the only other quantity affecting D's behavior is the ciphertext ψ^*. This ciphertext is computed differently in games $\mathbf{G}_2$ and $\mathbf{G}_3$: for the sake of clarity, we will denote with $[\psi^*]_2$ and $[\psi^*]_3$ the value of such quantity in game $\mathbf{G}_2$ and $\mathbf{G}_3$, respectively. We now want to show that, conditioned on all the other information in D's view, $[\psi^*]_2$ and $[\psi^*]_3$ are distributed according to the same distribution in the two games.

In game $\mathbf{G}_2$, the ciphertext $[\psi^*]_2$ sent to the decoder is completely determined by $\boldsymbol{V}$, $\bar{\boldsymbol{\beta}}$ and by the v-degree polynomial $X' \doteq rA'(\cdot) + wr'B'(\cdot)$. Similarly, in game $\mathbf{G}_3$, the ciphertext $[\psi^*]_3$ is completely determined by $\boldsymbol{V}, \bar{\boldsymbol{\beta}}$ and by the v-degree polynomial $X''(\cdot) \doteq rA''(\cdot)+wr'B''(\cdot)$. Moreover, $[\psi^*]_2$ depends on $\boldsymbol{V}, \bar{\boldsymbol{\beta}}$ and $X'(\cdot)$ according to the same functional dependence of $[\psi^*]_3$ upon $\boldsymbol{V}$, $\bar{\boldsymbol{\beta}}$ and $X''(\cdot)$. Therefore, to prove the lemma, it suffices to show that, conditioning on any fixed values of $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, $X'(\cdot)$ and $X''(\cdot)$ are distributed according to the same conditional probability distribution.

Recall that, in game $\mathbf{G}_2$, the polynomials $A'(\cdot)$ and $B'(\cdot)$ are chosen uniformly at random from $\mathbb{Z}_q^v[x]$, independently from anything else, but subject to the constraints in Eq. (27). It follows that the polynomial $X'(\cdot) = rA'(\cdot) + wr'B'(\cdot)$ is also random in $\mathbb{Z}_q^v[x]$, subject to the constraint that its value at x_s is

$$rA^{\bar{t}}(x_s) + wr'B^{\bar{t}}(x_s),\ \forall s \in I.$$

Similarly, in game $\mathbf{G}_3$, the polynomials $A''(\cdot)$ and $B''(\cdot)$ are chosen uniformly at random from $\mathbb{Z}_q^v[x]$, independently from anything else, but subject to the constraints in (31). It follows that the polynomial $X''(\cdot) = rA''(\cdot) + wr'B''(\cdot)$ is also random in $\mathbb{Z}_q^v[x]$, subject to the constraint that its value at x_s is

$$rA^{\bar{t}}(x_s) + wr'B^{\bar{t}}(x_s),\ \forall s \in I \setminus \{i\}.$$

In other words, the distributions of $X'(\cdot)$ and $X''(\cdot)$ only differ in that the value of $X'(\cdot)$ at x_i is fixed to be

$$\mathbf{X}'_i \doteq rA^{\bar{t}}(x_i) + wr'B^{\bar{t}}(x_i)$$

whereas the value of $X''(\cdot)$ at x_i is a random element in $\mathbb{Z}_q$. Thus, to prove that $X'(\cdot)$ and $X''(\cdot)$ have the same conditional probability distribution with respect to $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, it suffices to show that, conditioning on any fixed values of $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$, the value $\mathbf{X}'_i$ is distributed uniformly at random in $\mathbb{Z}_q$.

To this aim, consider the following matrix equation:

$$\boldsymbol{\beta} = \mathbf{M} \cdot \boldsymbol{\alpha} + \boldsymbol{\gamma}$$

where $\boldsymbol{\beta} \in \mathbb{Z}_q^{(v+\bar{m}+2)\times 1}$ is the vector

$$\boldsymbol{\beta} \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \mathbf{A}_1, \ldots, \mathbf{A}_{\bar{m}}, \mathbf{X}'_i)^T,$$

$\boldsymbol{\gamma} \in \mathbb{Z}_q^{(v+\bar{m}+2)\times 1}$ is the vector

$$\boldsymbol{\gamma} \doteq \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ D^{0,\bar{t}}(x_{t_1}) \\ \vdots \\ D^{0,\bar{t}}(x_{t_{\bar{m}}}) \\ rD^{0,\bar{t}}(x_i) + wr'E^{0,\bar{t}}(x_i) \end{pmatrix}$$

and $\mathbf{M} \in \mathbb{Z}_q^{(v+\bar{m}+2)\times(2v+2)}$ is the matrix

$$\mathbf{M} \doteq \begin{pmatrix} 1 & 0 & \ldots & 0 & w & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 1 & w & w & \ldots & w \\ & & \vdots & & & \vdots & & \\ 1 & v & \ldots & v^v & w & wv & \ldots & wv^v \\ 1 & x_{t_1} & \ldots & x_{t_1}^v & 0 & 0 & \ldots & 0 \\ & & \vdots & & & \vdots & & \\ 1 & x_{t_m} & \ldots & x_{t_{\bar{m}}}^v & 0 & 0 & \ldots & 0 \\ r & rx_i & \ldots & rx_i^v & wr' & wr'x_i & \ldots & wr'x_i^v \end{pmatrix}$$

By inspection, it is possible to see that the rows of matrix $\mathbf{M}$ are linearly independent, provided that $r \neq r'$ and $w \neq 0$: thus, the rank of $\mathbf{M}$ is $v + \bar{m} + 2$. As soon as we fix $\boldsymbol{V}$, vector $\boldsymbol{\gamma}$ and the first $v + 1$ rows of $\mathbf{M}$ are determined, but $\boldsymbol{\alpha}$ is still distributed uniformly and independently at random in $\mathbb{Z}_q^{(2v+2)\times 1}$. Similarly to the proof of Lemma 3, it is also possible to show that fixing the first $v + j$ entries of $\bar{\boldsymbol{\beta}}$ determines the $(v + j + 1)$th row of $\mathbf{M}$, for $j = 1, \ldots, \bar{m}$; and that moreover, fixing the first $v + \bar{m} + 1$ entries of $\bar{\boldsymbol{\beta}}$ determines all the remaining rows of $\mathbf{M}$.

By Lemma 1, we can conclude that the conditional distribution of $\mathbf{X}'_i$ with respect to $\boldsymbol{V}$ and $\bar{\boldsymbol{\beta}}$ is uniform over $\mathbb{Z}_q$. In other words, conditioning on the information seen by the adversary before receiving the challenge ψ^*, the value of $X'(\cdot)$ at x_i looks random over $\mathbb{Z}_q$. Thus, $(\boldsymbol{V}, \bar{\boldsymbol{\beta}}, X'(\cdot))$ has the same joint distribution as $(\boldsymbol{V}, \bar{\boldsymbol{\beta}}, X''(\cdot))$, completing the proof. □

6.3 Non-black-box tracing

In Sect. 6.3.2 we describe a non-black-box tracing algorithm which builds on the results of [3,25], but it is tailored to our family of representations. Then, in Sect. 6.3.3, we analyze its security properties in the formal model for traceability of Sect. 6.1, under a non-black-box assumption, given below as Assumption 3. Before that, however, we develop some notation.

6.3.1 Notation

Recall that in the scheme of Sect. 4, the secret key of user x_i consists of two points $A(x_i), B(x_i)$, which can be combined with the system's public key to obtain two leap-vectors to be used in the decryption algorithm. More precisely, given the current public key

$$PK \doteq \langle g, g', y, \langle z_1, h_1\rangle, \ldots, \langle z_v, h_v\rangle\rangle,$$

it is possible to construct (by Definition 6) two leap-vectors

$$\boldsymbol{\nu}_{A,i} \doteq \boldsymbol{\nu}_{z_1,\ldots,z_v}^{x_i,A} \quad \boldsymbol{\nu}_{B,i} \doteq \boldsymbol{\nu}_{z_1,\ldots,z_v}^{x_i,B}$$

where $(A(\cdot), B(\cdot))$ is the master secret key corresponding to the current public key PK. By Eqs. (2) and (4), $\boldsymbol{\nu}_{A,i}$ and $\boldsymbol{\nu}_{B,i}$ agree on the last v components; thus, under the current public key PK, user x_i's secret key can be compactly rewritten as

$$\begin{aligned}\boldsymbol{\delta}_i &\doteq \langle (\nu_{A,i})_0, (\nu_{B,i})_0, \boldsymbol{\delta}'_i\rangle \\ &\doteq \langle \lambda_0^{(i)} A(x_i), \lambda_0^{(i)} B(x_i), \langle \lambda_1^{(i)}, \ldots, \lambda_v^{(i)}\rangle\rangle,\end{aligned}$$

where $\lambda_0^{(i)}, \lambda_1^{(i)}, \ldots, \lambda_v^{(i)}$ are the Lagrange coefficients defined in Eqs. (3) and (4); recall that, for notational convenience, we use superscript (i) to make explicit that a given set of Lagrange coefficients is relative to user x_i.

Notice that such vector $\boldsymbol{\delta}_i$ is a representation of y with respect to $g, g', h_1, \ldots, h_v$; for short, when this is the case, in the following we will just say that $\boldsymbol{\delta}_i$ is a valid representation of the public key PK. Also notice that *any* such valid representation $\boldsymbol{\delta}$ of the current public key PK would work for decrypting messages encrypted with PK; for a generic valid representation

$$\boldsymbol{\delta} \doteq \langle \gamma_a, \gamma_b, \gamma_1, \ldots, \gamma_v \rangle,$$

we will denote with $\boldsymbol{\delta}'$ its last v entries:

$$\boldsymbol{\delta}' \doteq \langle \gamma_1, \ldots, \gamma_v \rangle.$$

In the non-black-box model, the tracing algorithm is assumed to be able of inspecting the content of a successful pirate decoder, and to extract the secret key hidden within it. More precisely, in designing and analyzing our non-black-box tracing algorithm, we make the following assumption:

Assumption 3 (Non-Black-Box Assumption) *Let $\mathcal{A}$ be any probabilistic, polynomial-time adversary, and let $\langle \mathsf{D}, PK_{\mathcal{A}}, MSK_{\mathcal{A}}, \mathcal{T} \rangle$ be the output resulting from the adversary playing the traceability attack game $\boldsymbol{G}^m_{\mathsf{trt}}(1^k)$ with the challenger. If D can correctly decrypt random ciphertexts encrypted using $PK_{\mathcal{A}}$ (in other words, $\mathsf{Succ}_{PK_{\mathcal{A}}}(\mathsf{D}) = 1$), then D contains a valid representation $\boldsymbol{\delta}$ of $PK_{\mathcal{A}}$, and it is possible to reverse-engineer D and extract $\boldsymbol{\delta}$.*

Assumption 3 is partially supported by Proposition 1 and it is essentially equivalent to what was previously assumed in [3]. It is also *a priori* much less restrictive than the non-black-box assumption made in [25], where the non-black-box analysis is subject to the hypothesis that the illegal key extracted from the pirate decoder is a convex linear combination of some of the traitors' keys. In fact, in Lemma 6 (whose proof is given in Sect. 6.3.3) we show that in our context, the seemingly more restrictive assumption from [25] actually follows from Assumption 3 and Assumption 2.

Lemma 6. *Let $\mathcal{A}$ be any probabilistic, polynomial-time adversary, and let $\langle \mathsf{D}, PK_{\mathcal{A}}, MSK_{\mathcal{A}}, \mathcal{T} \rangle$ be the output resulting from the adversary playing the traceability attack game $\boldsymbol{G}^m_{\mathsf{trt}}(1^k)$ with the challenger. Also let $\mathcal{T} \doteq \{t_1, \ldots, t_m\}$ and, for $j = 1, \ldots, m$, denote with $\boldsymbol{\delta}_{t_j}$ the compact representation of the secret key of user t_j with respect to the public key $PK_{\mathcal{A}}$. If the pirate decoder D output by $\mathcal{A}$ contains a valid representation $\boldsymbol{\delta}$ for the public key $PK_{\mathcal{A}}$, such that $\boldsymbol{\delta}'$ is not a linear combination of $\boldsymbol{\delta}'_{t_1}, \ldots, \boldsymbol{\delta}'_{t_m}$, then the discrete logarithm problem over $\mathcal{G}$ is solvable.*

6.3.2 Non-black-box tracing algorithm

We present a deterministic tracing algorithm that recovers, under Assumptions 2 and 3, the identities of the traitors that created the pirate key. Suppose that the content of a pirate decoder is exposed. By Assumption 3, it is possible to extract from D a valid representation $\boldsymbol{\delta}$ of the current public key $PK_{\mathcal{A}}$. Define $\{x_1, \ldots, x_n\}$ to be the set of all values assigned to the users in the system (where n denotes the total number of users in the system), and let $\boldsymbol{\delta_1}, \ldots, \boldsymbol{\delta_n}$ be the corresponding secret keys. Let $\{z_{i_1}, \ldots, z_{i_v}\}$ be the set of values of the revoked users specified in the current public key.[3] Remember that the secret key of user j with respect to the current public key can be compactly represented in the form

$$\boldsymbol{\delta_j} \doteq \langle \lambda_0^{(j)} A(x_j), \lambda_0^{(j)} B(x_j), \lambda_{i_1}^{(j)}, \ldots, \lambda_{i_v}^{(j)} \rangle$$

where $\lambda_j^{(j)}, \lambda_{i_1}^{(j)}, \ldots, \lambda_{i_v}^{(j)}$ are the Lagrange coefficients defined in Eqs. (3) and (4). Notice that, for any polynomial $P(\cdot) \in \mathbb{Z}_q^v[x]$, it holds that

$$P(0) = \lambda_0^{(j)} P(x_j) + \lambda_{i_1}^{(j)} P(x_{i_1}) + \ldots + \lambda_{i_v}^{(j)} P(x_{i_v}).$$

Consider the matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times v}$ whose jth row is $\boldsymbol{\delta'_j}$, for $j = 1, \ldots, n$, i.e.:

$$\mathbf{A} \doteq \begin{pmatrix} \lambda_{i_1}^{(1)} \ldots \lambda_{i_v}^{(1)} \\ \ldots \\ \lambda_{i_1}^{(n)} \ldots \lambda_{i_v}^{(n)} \end{pmatrix}$$

Define the identities of the traitors to be $\{t_1, \ldots, t_m\} \subseteq \{1, \ldots, n\}$. By Lemma 6 and Assumption 2, $\boldsymbol{\delta}'$ must be a linear combination of the vectors $\boldsymbol{\delta'_{t_1}}, \ldots, \boldsymbol{\delta'_{t_m}}$ obtained by projecting the traitors' secret keys $\boldsymbol{\delta_{t_1}}, \ldots, \boldsymbol{\delta_{t_m}}$ onto the last v components. It follows that $\boldsymbol{\delta}'$ also lies in the linear span of $\boldsymbol{\delta'_1}, \ldots, \boldsymbol{\delta'_n}$. More precisely, there exists a vector $\boldsymbol{\varphi}$ of Hamming weight at most m such that

$$\boldsymbol{\delta}' = \boldsymbol{\varphi} \cdot \mathbf{A}. \tag{35}$$

Consider the two matrices:

$$\mathbf{B} \doteq \begin{pmatrix} x_{i_1} \ldots x_{i_1}^v \\ \ldots \\ x_{i_v} \ldots x_{i_v}^v \end{pmatrix} \quad \mathbf{H} \doteq \begin{pmatrix} -\lambda_0^{(1)} x_1 \ldots -\lambda_1^{(1)} x_1^v \\ \ldots \\ -\lambda_0^{(n)} x_n \ldots -\lambda_0^{(n)} x_n^v \end{pmatrix}$$

It is easy to verify that $\mathbf{A} \cdot \mathbf{B} = \mathbf{H}$. Multiplying (35) by $\mathbf{B}$, we get

$$\boldsymbol{\varphi} \cdot \mathbf{H} = \boldsymbol{\delta}''$$

where

$$\boldsymbol{\delta}'' \doteq \boldsymbol{\delta}' \cdot \mathbf{B}. \tag{36}$$

Let $\mathcal{C}$ denote the linear code over $\mathbb{Z}_q^n$ that has $\mathbf{H}$ as its parity-check matrix, i.e.

$$\boldsymbol{c} \in \mathcal{C} \iff \boldsymbol{c} \cdot \mathbf{H} = \mathbf{0}.$$

Let $\lambda_1, \ldots, \lambda_n$ be the Lagrange coefficients corresponding to $\{x_1, \ldots, x_n\}$; thus, for every $P(\cdot) \in \mathbb{Z}_q^{<n}[x]$, it holds that

$$P(0) = \lambda_1 P(x_1) + \ldots + \lambda_n P(x_n).$$

In Lemma 7 (Sect. 6.3.3), we prove that $\mathcal{C}$ is a Generalized Reed-Solomon Code (GRS), with distance $(v + 1)$. For more details about Generalized Reed-Solomon Codes, see e.g. [22]. Generalized Reed-Solomon Codes can be decoded efficiently by the algorithm of Berlekamp and Welch [1]. This means

[3] Without loss of generality we are assuming that the current saturation level L is equal to v.

that, for any $e \leq m$ and any vector $\boldsymbol{\mu} \in \mathbb{Z}_q^n$, there exists (at most) a unique vector $\boldsymbol{\omega} \in \mathcal{C}$ that disagrees with $\boldsymbol{\mu}$ in at most e positions (since $\mathcal{C}$ has distance $(v+1)$ and $m = \lfloor \frac{v}{2} \rfloor$). Moreover, such unique vector $\boldsymbol{\omega} \in \mathcal{C}$ (if it exists) can be recovered in deterministic polynomial-time. We now describe how this can be exploited to reconstruct $\boldsymbol{\varphi}$ given $\boldsymbol{\delta}'$.

First, we compute an arbitrary vector $\boldsymbol{\vartheta} \in \mathbb{Z}_q^n$ that satisfies the system of equations

$$\boldsymbol{\vartheta} \cdot \mathbf{H} = \boldsymbol{\delta}''. \tag{37}$$

where $\boldsymbol{\delta}''$ is defined in Eq. (36). Note that such $\boldsymbol{\vartheta}$ can be found by standard linear algebra since Eq. (37) induces a system of v equations with n unknowns, $n > v$, and $\mathbf{H}$ contains a non-singular minor of size v. It is easy to verify that the vector

$$\boldsymbol{\omega} \doteq \boldsymbol{\vartheta} - \boldsymbol{\varphi}$$

belongs to the linear code $\mathcal{C}$; indeed,

$$\begin{aligned}\boldsymbol{\omega} \cdot \mathbf{H} &= \boldsymbol{\vartheta} \cdot \mathbf{H} - \boldsymbol{\varphi} \cdot \mathbf{H}\\ &= \boldsymbol{\delta}'' - \boldsymbol{\delta}''\\ &= \mathbf{0}.\end{aligned}$$

As a result, the vector $\boldsymbol{\vartheta}$ can be expressed as $\boldsymbol{\vartheta} = \boldsymbol{\omega} + \boldsymbol{\varphi}$.

Provided that the number of traitors is at most m, it holds that the Hamming weight of $\boldsymbol{\varphi}$ is less than or equal to m and as a result $\boldsymbol{\vartheta}$ is an n-vector that differs in at most m positions from the vector $\boldsymbol{\omega}$ (which belongs to $\mathcal{C}$): in other words, we can view $\boldsymbol{\vartheta}$ as a "partially corrupted" version of the codeword $\boldsymbol{\omega}$. Therefore, we can recover $\boldsymbol{\omega}$ from $\boldsymbol{\vartheta}$, by running the Berlekamp-Welch decoding algorithm for GRS-codes on input $\boldsymbol{\vartheta}$. At this point, $\boldsymbol{\varphi}$ can be computed as $\boldsymbol{\varphi} = \boldsymbol{\vartheta} - \boldsymbol{\omega}$.

By Eq. (35), $\boldsymbol{\varphi}$ is a vector of Hamming weight at most m, whose non-zero components correspond to the identities of the traitors; thus, the traitors' identities can be recovered as

$$\{t_1, \ldots, t_m\} \doteq \{j \in \{1, \ldots, n\} \mid \boldsymbol{\varphi}_j \neq 0\}.$$

Time-Complexity. The tracing procedure has time complexity $O(n^2)$, which can be optimized to $O(n(\log n)^2)$, if matrix operations are implemented in a more sophisticated manner, see e.g. [2]. If the number of traitors exceeds the bound m, it is still possible to extract candidate sets of potential traitors using the Guruswami-Sudan algorithm [16], which performs GRS-decoding "beyond the error-correction bound." This will work provided that the size of the traitor coalition is less than or equal to $n - \sqrt{n(n-v)}$.

6.3.3 Correctness of non-black-box tracing

Given Lemmas 6 and 7, the correctness of the non-black-box tracing algorithm described above follows from the properties of algebraic decoding of GRS codes. Thus, to conclude the argument, we now move on to the proofs of these lemmas.

Proof of Lemma 6
Let g be a generator of $\mathcal{G}$, and let $g' \doteq g^w$. Using adversary $\mathcal{A}$ described in the attack game $\mathbf{G}_{\mathsf{trt}}^m(1^k)$, we want to show how to recover the value w. In performing step 1. of $\mathbf{G}_{\mathsf{trt}}^m(1^k)$, choose two random polynomials $A^0(\cdot)$ and $B^0(\cdot)$ and set the initial public key to be

$$\langle g, g', g^{A^0(0)} g'^{B^0(0)}, \langle \ell, g^{A^0(\ell)} g'^{B^0(\ell)} \rangle_{\ell=1}^{v} \rangle.$$

The game then proceeds as described in Sect. 6.1; in particular, let $\bar{t}$ be the number of $\mathsf{New\text{-}period}$ operation occurring during the entire game. Eventually, adversary $\mathcal{A}$ outputs a pirate decoder D from which (by Assumption 3) it is possible to extract a vector

$$\boldsymbol{\delta} = \langle \gamma_a, \gamma_b, \gamma_1, \ldots, \gamma_v \rangle,$$

which is a valid representation of the final public key $PK_{\mathcal{A}}$. In formula,

$$y = g^{\gamma_a} g'^{\gamma_b} \prod_{\ell=1}^{v} h_\ell^{\gamma_\ell} \tag{38}$$

where

$$PK_{\mathcal{A}} \doteq \langle g, g', y, \langle x_{i_\ell}, h_\ell \rangle_{\ell=1}^{v} \rangle.$$

Considering discrete logarithms to the base g of Eq. (38), we get:

$$A^{\bar{t}}(0) + wB^{\bar{t}}(0) = \gamma_a + \sum_{\ell=1}^{v} A^{\bar{t}}(x_{i_\ell})\gamma_\ell + w\Big(\gamma_b + \sum_{\ell=1}^{v} B^{\bar{t}}(x_{i_\ell})\gamma_\ell\Big)$$

that can be rewritten as:

$$w\Big(\gamma_b + \sum_{\ell=1}^{v} B^{\bar{t}}(x_{i_\ell})\gamma_\ell - B^{\bar{t}}(0)\Big) = A^{\bar{t}}(0) - \gamma_a - \sum_{\ell=1}^{v} A^{\bar{t}}(x_{i_\ell})\gamma_\ell \tag{39}$$

Notice that both the right-hand side and the coefficient of w in Eq. (39) are known, so that if such coefficient is non-zero (or, equivalently, if the right-hand side of Eq. (39) is non-zero), then we can successfully recover the value of w, thus violating Assumption 2. To complete the argument, it then suffices to show that the right-hand side of (39) is zero only with negligible probability, or equivalently that:

$$\Pr[\gamma_a = \bar{\gamma}_a] = 1/q \tag{40}$$

where

$$\bar{\gamma}_a \doteq A^{\bar{t}}(0) - \sum_{\ell=1}^{v} A^{\bar{t}}(x_{i_\ell}).$$

To this aim, below we prove that, conditioning on all the other information in $\mathcal{A}$'s view, the quantity $\bar{\gamma}_a$ is uniformly distributed in $\mathbb{Z}_q$. It will follow that $\mathcal{A}$'s chances of outputting a value γ_a equal to $\bar{\gamma}_a$ are just 1 in q, proving Equation (39) and thus the lemma.

To prove that $\bar{\gamma}_a$ is distributed uniformly in $\mathbb{Z}_q$, we again make use of Lemma 1 following the same approach described in Sect. 5.2.

Consider the quantity

$$\boldsymbol{V} \doteq (\mathsf{Coins}, w, \{\{c_j^t, r_j^t\}_{j=1}^{2v+2}\}_{t=1}^{\bar{t}})$$

where Coins represents the coin tosses of $\mathcal{A}$, $w \doteq \log_g g'$, and $\{\{c_j^t, r_j^t\}_{j=1}^{2v+2}\}_{t=1}^{\bar{t}}$ represents all the randomness used in the $\bar{t}$ $\mathsf{New\text{-}period}$ operations that took place during the $\mathbf{G}_{\mathsf{trt}}^m(1^k)$ attack game.

The remaining randomness used during the attack game consists of the $2v+2$ coefficients of the polynomials $A^0(\cdot)$, $B^0(\cdot)$ and can be represented by a vector $\boldsymbol{\alpha}$ uniformly distributed in $\mathbb{Z}_q^{(2v+2)\times 1}$:

$$\boldsymbol{\alpha} \doteq (a_0, a_1, \ldots, a_v, b_0, b_1, \ldots, b_v)^T.$$

Consider the vector $\boldsymbol{\beta} \in \mathbb{Z}_q^{(v+m+2)\times 1}$ defined as:

$$\boldsymbol{\beta} \doteq (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_v, \mathbf{A}_1, \ldots, \mathbf{A}_m, \bar{\gamma}_a)^T$$

where $\mathbf{X}_0 \doteq A^0(0) + wB^0(0)$, $\mathbf{X}_\ell \doteq A^0(\ell) + wB^0(\ell)$, for $\ell = 1.\ldots, v$ and $\mathbf{A}_j \doteq A^0(t_j)$ for $j = 1, \ldots, m$.

It is clear by inspection that all the information in the view of the adversary $\mathcal{A}$ during the attack game $\mathbf{G}^m_{\mathsf{trt}}(1^k)$ is completely determined by $\boldsymbol{V}$ and $\boldsymbol{\beta}$. In particular, the initial public key PK^0 is fixed by $\boldsymbol{\beta}$ and w, and the secret keys of the traitors are determined by the choice of $\boldsymbol{\beta}$, Coins and w.

The quantities in $\boldsymbol{V}$, $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are related according to the following matrix equation:

$$\boldsymbol{\beta} = \mathbf{M} \cdot \boldsymbol{\alpha} + \boldsymbol{\gamma}$$

where $\boldsymbol{\gamma} \in \mathbb{Z}_q^{(v+m+2)\times 1}$ is the vector

$$\boldsymbol{\gamma} \doteq \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ D^{0,\bar{t}}(0) - \sum_{\ell=1}^{v} D^{0,\bar{t}}(x_{i_\ell})\gamma_\ell \end{pmatrix}$$

and $\mathbf{M} \in \mathbb{Z}_q^{(v+m+2)\times(2v+2)}$ is the matrix

$$\begin{pmatrix} 1 & 0 & \ldots & 0 & w & 0 & \ldots & 0 \\ 1 & 1 & \ldots & 1 & w & w & \ldots & w \\ & & \vdots & & & & \vdots & \\ 1 & v & \ldots & v^v & w & wv & \ldots & wv^v \\ 1 & x_{t_1} & \ldots & x_{t_1}^v & 0 & 0 & \ldots & 0 \\ & & \vdots & & & & \vdots & \\ 1 & x_{t_m} & \ldots & x_{t_m}^v & 0 & 0 & \ldots & 0 \\ 1-\sum_{\ell=1}^{v}\gamma_\ell & -\sum_{\ell=1}^{v}\gamma_\ell x_{i_\ell} & \ldots & -\sum_{\ell=1}^{v}\gamma_\ell x_{i_\ell}^v & 0 & 0 & \ldots & 0 \end{pmatrix}$$

By inspection, it is possible to see that the first $v+m+1$ rows of $\mathbf{M}$ are linearly independent, provided that $w \neq 0$. To see that the rank of $\mathbf{M}$ is indeed $v+m+2$, define $\mathbf{T} \in \mathbb{Z}_q^{m\times v}$ to be the minor of matrix $\mathbf{A}$ resulting from considering only rows $t_1, \ldots, t_m$:

$$\mathbf{T} \doteq \begin{pmatrix} \lambda_{i_1}^{(t_1)} & \ldots & \lambda_{i_v}^{(t_1)} \\ & \ldots & \\ \lambda_{i_1}^{(t_m)} & \ldots & \lambda_{i_v}^{(t_m)} \end{pmatrix}$$

It is possible to show that if the last row of $\mathbf{M}$ were in the linear span of the first $v+m+1$ rows of $\mathbf{M}$, it would follow that $\boldsymbol{\delta}'$ should belong to the linear span of the rows of $\mathbf{T}$. But since, by hypothesis, $\boldsymbol{\delta}'$ is not a linear combination of $\boldsymbol{\delta}'_{t_1}, \ldots, \boldsymbol{\delta}'_{t_m}$, the matrix $\mathbf{M}$ must have full rank.

As soon as we fix $\boldsymbol{V}$, the first $v+m+1$ entries of $\boldsymbol{\gamma}$ and the first $v+1$ rows of $\mathbf{M}$ are determined, but $\boldsymbol{\alpha}$ is still distributed uniformly and independently at random in $\mathbb{Z}_q^{(2v+2)\times 1}$. Similarly to the proof of Lemma 3, it is also possible to show that fixing the first $v+j+1$ entries of $\boldsymbol{\beta}$ determines the $(v+j+2)$th row of $\mathbf{M}$, for $j = 1, \ldots, m$; and that moreover, fixing the first $v+m+1$ entries of $\boldsymbol{\beta}$ also determines the last rows of $\boldsymbol{\gamma}$ and of $\mathbf{M}$.

Hence, by Lemma 1, we can conclude that the conditional distribution of $\bar{\gamma}_a$ with respect to $\boldsymbol{V}$ and to the first $v+m+1$ entries of $\boldsymbol{\beta}$, is uniform over $\mathbb{Z}_q$. In other words, conditioning on all the other information in $\mathcal{A}$'s view, the quantity $\bar{\gamma}_a$ is uniformly distributed over $\mathbb{Z}_q$. Equation (39), and thus the lemma, follows. □

Lemma 7. *Consider the Generalized Reed-Solomon code:*

$$\mathcal{C}' \doteq \{\langle -\frac{\lambda_1}{\lambda_0^{(1)}}P(x_1), \ldots, -\frac{\lambda_n}{\lambda_0^{(n)}}P(x_n)\rangle \mid P \in \mathbb{Z}_q^{<n-v}[x]\}.$$

It holds that

1. *$\mathcal{C} = \mathcal{C}'$.*
2. *$\mathcal{C}$ is a linear code with message-rate $(n-v)/n$ and distance $v+1$.*

Proof.
1. We only need to show that $\mathcal{C}' \subseteq \mathcal{C}$. Indeed, assuming that $\mathcal{C}'$ is a linear sub-space of $\mathcal{C}$, since $\dim(\mathcal{C}) = n-v = \dim(\mathcal{C}')$, it immediately follows that $\mathcal{C} = \mathcal{C}'$.

To prove that $\mathcal{C}' \subseteq \mathcal{C}$, notice that if $\langle c_1, \ldots, c_n\rangle \in \mathcal{C}'$, then it is of the form

$$\big\langle -\frac{\lambda_1}{\lambda_0^{(1)}}P(x_1), \ldots, -\frac{\lambda_n}{\lambda_0^{(n)}}P(x_n)\big\rangle$$

for some polynomial $P(\cdot) \in \mathbb{Z}_q^{<n-v}[x]$. We now verify that $\langle c_1, \ldots, c_n\rangle$ belongs to $\mathcal{C}$. First, notice that for $\ell = 1, \ldots, v$, multiplying $\langle c_1, \ldots, c_n\rangle$ by the ℓth column of $\mathbf{H}$ we get

$$\langle c_1, \ldots, c_n\rangle \cdot \langle -\lambda_0^{(1)}x_1^\ell, \ldots, -\lambda_0^{(n)}x_n^\ell\rangle = \sum_{i=1}^{n} \lambda_i P(x_i) x_i^\ell.$$

Now observe that

$$\sum_{i=1}^{n} \lambda_i P(x_i) x_i^\ell = 0$$

by the choice of $\lambda_1, \ldots, \lambda_n$ and the facts that $\mathrm{degree}(P) < n-v$ and $\ell \leq v$ (just consider the polynomial $Q(x) \doteq P(x)x^\ell \in \mathbb{Z}_q^{<n}[x]$). It follows that

$$\langle c_1, \ldots, c_n\rangle \cdot \mathbf{H} = \mathbf{0}.$$

2. Observe that a vector of $\mathbb{Z}_q^{n-v}$ can be encoded as the coefficients of a polynomial $P(\cdot) \in \mathbb{Z}_q^{<n-v}[x]$. The corresponding codeword of $\mathcal{C}$ will be the vector

$$\big\langle -\frac{\lambda_1}{\lambda_0^{(1)}}P(x_1), \ldots, -\frac{\lambda_n}{\lambda_0^{(n)}}P(x_n)\big\rangle.$$

To see that the distance of the linear code is $v+1$, observe that any two different codewords of $\mathcal{C}$ can agree on at most $n-v-1$ positions, or equivalently any two distinct codewords differ on at least $v+1$ positions. □

7 Conclusions and future work

We introduce the first public-key traitor tracing scheme where an unlimited number of users can be efficiently added and removed from the system. Our scheme enjoys both client-side scalability, by supporting a dynamically-changing user population, and server-side scalability, as it enables many content providers to use a common content distribution infrastructure.

We present a formal model for scalable public-key traitor tracing, and a thorough analysis of the revocation and tracing properties of our scheme against adaptive adversaries.

At a technical level, our adversarial model improves over previous modeling for public-key traitor tracing by capturing a larger class of adversaries, endowed with greater control over the system when compared to previous schemes. In particular, in our model the adversary can control an *a priori* unbounded number of user additions and removals. The main limitation of our formal model is that the adversary is supposed to be fully revoked in a "window" of the system.

We leave it as an interesting open problem to extend our results to a more general adversarial model, where the adversary is not supposed to obey the "window" constraint (while maintaining the efficiency of the scheme).

Acknowledgements. We are grateful to Antonio Nicolosi for his constructive criticism and encouraging support throughout this research. We thank one of the anonymous reviewers for pointing out a flaw in an early version [10] of our scheme, and we thank all the reviewers for their helpful comments that improved the readability and exposition of the paper.

References

1. Berlekamp ER, Welch LR: Error Correction of Algebraic Block Codes, 1986. U.S. Patent, Number 4,633,470
2. Bini D, Pan VY: Polynomial and Matrix Computations (vol. 1): Fundamental Algorithms. Birkhäuser, 1994
3. Boneh D, Franklin M: An Efficient Public Key Traitor Tracing Scheme. In: Advances in Cryptology – Crypto '99. LNCS 1666, pp 338–353. Springer, 1999. Full version available at crypto.stanford.edu/~dabo/pubs.html
4. Brands S: Rethinking Public Key Infrastructures and Digital Certificates – Building in Privacy. PhD thesis, Technical University of Eindhoven, 1999
5. Canetti R, Garay J, Itkis G, Micciancio D, Naor M, Pinkas B: Multicast Security: A Taxonomy and some Efficient Constructions. Proceedings of IEEE INFOCOM '99 2:708–716 (1999)
6. Chor B, Fiat A, Naor N: Tracing Traitors. In: Advances in Cryptology – Crypto '94 Springer, 1994, pp. 257–270. LNCS 839
7. Cramer R, Shoup V: Design and Analysis of Practical Public-Key Encryption Scheme Secure against Adaptive Chosen Ciphertext Attack. SIAM J Comput 33(1):167–226 (2003)
8. Dodis Y, Fazio N: Public-Key Broadcast Encryption for Statless Receivers. In: Digital Rights Management – DRM '02. LNCS 2696, pp. 61–80 Springer, 2002
9. Dodis Y, Fazio N: Public-Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack. In: Public Key Cryptography – PKC '03. LNCS 2567, pp. 100–115. Springer, 2003
10. Dodis Y, Fazio N, Kiayias A, Yung M: Scalable Public-Key Tracing and Revoking. In: 22nd Annual Symposium on Principles of Distributed Computing – PODC '03, pp. 190–199. ACM Press, 2003
11. Dodis Y, Katz J, Xu S, Yung M: Key-Insulated Public-Key Cryptosystems. In: Advances in Cryptology – EuroCrypt '02. LNCS 2332, pp. 65–82. Springer, 2002
12. Fiat A, Naor M: Broadcast Encryption. In: Advances in Cryptology – Crypto '93. LNCS 773, pp. 480–491. Springer, 1993
13. Fiat A, Tassa T: Dynamic Traitor Tracing. J Cryptol 14(3):211–223 (2001)
14. Gafni E, Staddon J, Yin YL: Efficient Methods for Integrating Traceability and Broadcast Encryption. In: Advances in Cryptology – Crypto '99. LNCS 1666, pp. 372–387. Springer, 1999
15. Garay A, Staddon J, Wool A: Long-Lived Broadcast Encryption. In: Advances in Cryptology – Crypto 2000. LNCS 1880, pp. 333–352. Springer, 2000
16. Guruswami V, Sudan M: Improved Decoding of Reed-Solomon and Algebraic-Geometric Codes. In: IEEE Symposium on Foundations of Computer Science, pp. 28–39, 1998
17. Halevy D, Shamir A: The LSD Broadcast Encryption Scheme. In: Advances in Cryptology – Crypto '02. LNCS 2442, pp. 47–60. Springer, 2002
18. Kiayias A, Yung M: On Crafty Pirates and Foxy Tracers. In: Digital Rights Management – DRM '01. LNCS 2320, pp. 22–39. Springer, 2001
19. Kiayias A, Yung M: Self Protecting Pirates and Black-Box Traitor Tracing. In: Advances in Cryptology – Crypto '01. LNCS 2139, pp. 63–79. Springer, 2001
20. Kiayias A, Yung M: Traitor Tracing with Constant Transmission Rate. In: Advances in Cryptology – EuroCrypt '02. LNCS 2332, pp. 450–465. Springer, 2002
21. Kurosawa K, Desmedt Y: Optimum Traitor Tracing and new Direction for Asymmetricity. In: Advances in Cryptology – EuroCrypt '98. LNCS 1403, pp. 145–157. Springer, 1998
22. MacWilliams FJ, Sloane N: The Theory of Error Correcting Codes. North Holland, Amsterdam, 1977
23. Naor D, Naor M, Lotspiech J: Revocation and Tracing Schemes for Stateless Receivers. In: Advances in Cryptology – Crypto '01. LNCS 2139, pp. 41–62. Springer, 2001
24. Naor M, Pinkas B: Threshold Traitor Tracing. In: Advances in Cryptology – Crypto '98. LNCS 1462, pp. 502–517. Springer, 1998
25. Naor M, Pinkas B: Efficient Trace and Revoke Schemes. In: Financial Cryptography – FC 2000. LNCS 1962, pp. 1–20. Springer, 2000. Full version available at www.wisdom.weizmann.ac.il/~ naor/onpub.html
26. Ostrovsky R, Yung M: How to Withstand Mobile Virus Attacks. In: Logrippo L (ed) Proceedings of the 10th Annual ACM Symposium on Principles of Distributed Computing, pp. 51–60, Montréal, Québec, Canada, August 1991. ACM Press
27. Stinson DR, Wei R: Combinatorial Properties and Constructions of Traceability Schemes and Frameproof Codes. SIAM J Discrete Math 11(1):41–53 (1998)
28. Tzeng WG, Tzeng ZJ: A Public-Key Traitor Tracing Scheme with Revocation Using Dynamics Shares. In: Public Key Cryptography – PKC '01. LNCS 1992, pp. 207–224. Springer, 2001
29. Wallner D, Harder E, Agee R: Key Management for Multicast: Issues and Architectures. Available at ftp://ftp.ietf.org/rfc/rfc2627.txt, 1997