Some Advances in Broadcast Encryption and Traitor Tracing

Duong Hieu Phan

DI/ENS – École Doctorale de
Sciences Mathématiques de Paris Centre

LAGA, Département Mathématiques
Université de Paris 8

Some Advances in Broadcast Encryption and Traitor Tracing

THÈSE D'HABILITATION

présentée pour l'obtention du

Diplôme d'Habilitation à Diriger des Recherches de l'École normale supérieure

(Spécialité Informatique)

par

Duong Hieu Phan

Soutenue publiquement le 19 Novembre 2014 devant le jury composé de

Michel Abdalla ..... *Examinateur*
Claude Carlet ..... *Examinateur*
Jean-Sébastien Coron ..... *Rapporteur*
Louis Goubin ..... *Examinateur*
Marc Joye ..... *Rapporteur*
Aggelos Kiayias ..... *Rapporteur*
David Pointcheval ..... *Directeur de recherches*
Jacques Stern ..... *Examinateur*

Travaux effectués au sein de LAGA, UMR 7539, Université Paris 8 - Université Paris 13
et de l'Équipe de Cryptographie de l'École normale supérieure

Remerciements

Le travail présenté dans ce mémoire est un travail collectif, je tiens à remercier tous ceux qui m'ont accompagné dans l'aventure de la recherche depuis de nombreuses années.

Mes remerciements les plus chaleureux vont à David Pointcheval pour son soutien sans faille depuis le jour où il m'a accueilli comme premier thésard. Ses encouragements ont été décisifs à l'aboutissement de cette thèse !

J'adresse ma grande reconnaissance aux membres du jury : Michel Abdalla, Claude Carlet, Louis Goubin, Jacques Stern et en particulier aux rapporteurs : Aggelos Kiayias, Jean-Sébastien Coron et Marc Joye pour leur patience et leurs suggestions. Je suis honoré que la thèse ait été revue par de tels experts.

Un grand et amical merci à Phong Nguyen, qui m'a fait découvrir la recherche en cryptographie en France (pour la première fois lors d'un match de football au stade de Hanoï il y a une vingtaine d'années) et qui a toujours été de bon conseils depuis.

Mon tout premier cours de cryptographie en France a été avec Jacques Stern, je le remercie de m'avoir m'accueilli ensuite dans le laboratoire de l'ENS, c'est le point de départ de mon petit chemin en cryptographie.

La relecture de cette thèse est un travail dur, je tiens à remercier Olivier Billet, Malika Izabachène, David Pointcheval, et en particulier mon épouse Thuy-Linh.

Depuis plus de dix ans, j'ai eu le plaisir de collaborer avec de nombreux chercheurs brillants. Je tiens tout d'abord à remercier Trinh Viet Cuong, mon thésard, Damien Stehlé, mon ami de DEA, et mes co-auteurs : Michel Abdalla, Olivier Billet, Angelo De Caro, Hervé Chabanne, Benoît Chevallier-Mames, Alexander W. Dent, Yvo Desmedt, Nelly Fazio, Philippe Guillot, San Ling, John Malone-Lee, Helger Lipmaa, Gregory Neven, Antonio Nicolosi, Ngo Quang Hung, Nguyen Thanh Thuy, Abdelkrim Nimour, David Pointcheval, Reihaneh Safavi-Naini, Siamak Fayyaz Shahandashti, Nigel P. Smart, Ron Steinfeld, Mario Strefler, Dongvu Tonien et Takahiro Yamanoi.

L'Université Paris 8, l'ENS, France Telecom et l'University College London sont d'excellents environnements de recherche dans lesquels j'ai eu la chance de travailler. Je tiens à remercier mes collègues Claude Carlet, Farid Mokrane, Philippe Guillot, Sihem Mesnager, Wolfgang Schmid, Nadia El Mrabet, Serge Larrivière, Maarten Bullynck, Mariou Benoît, Marie-José Durand-Richard, Damien Vergnaud, Dario Catalano, Vadim Lyubashevsky, David Naccache, Pierre-Alain Fouque, Emmanuel Bresson, Louis Granboulan, Céline Chevalier, Hoeteck Wee, Oded Regev, Sorina Ionica, Elizabeth Quaglia, Angelo De Caro, Itai Dinur, Fabrice Ben Hamouda, Mario Cornejo, Alain Passelègue, Olivier Blazy, Henri Gilbert, Olivier Billet, Thomas Peyrin, Matt Robshaw, Yannick Seurin, Ryad Benadjila, Côme Berbain, Gilles Macario-Rat, David Arditti, Jonathan Etrog, Stanislas Francfort, Marc Girault, Sébastien Canard, Jacques Traoré, Brigitte Vallée, Fabien Laguillaumie, Ayoub Otmani et Marine Minier.

Une partie de cette thèse a été réalisée dans le cadre du projet ANR BEST dont je suis pilote pour l'Université Paris 8. Cécile Delerablée, Pascal Paillier, Aurore Guillevic, Renaud Dubois, Jean-Bernard Fischer et Régis Bevan ont eux aussi contribué à la thèse à travers des discussions très fructueuses.

Je suis redevable aux nombreuses personnes qui m'ont permis de conduire mon activité de recherche dans les meilleures conditions. Ainsi, je remercie les membres du département de mathématiques de l'Université Paris 8 et du département d'informatique de l'École normale superieure, et tout particulièrement Emmanuelle Najjar, Jöelle Isnard et Valérie Mongiat qui répondent toujours présentes au moindre problème.

Je remercie Lionel Schwartz et Phan Ha Duong de m'avoir impliqué dans plusieurs projets avec le Vietnam. Une grande partie de cette thèse a été rédigée au Vietnam cet été 2014, lors

de ma visite à VIASM. Je remercie Ngo Bao Chau, Nguyen Huu Du et Le Tuan Hoa pour leur accueil et aussi aux chercheurs vietnamiens avec qui j'ai eu des discussions très constructives, notamment Vu Duc Thang, Le Minh Ha, Nguyen Quoc Khanh, Nguyen Duy Lan, Nguyen Manh Ha, Le Trieu Phong, Le Duc Phong, Le Thanh Ha, Hoang Viet Tung, Vu Ha Van, et les étudiants Ngo Manh Cuong et Do Xuan Thanh. L'aventure continue, on se donne rendez-vous à l'ASIACRYPT 2016 au Vietnam !

Mes pensées le plus profondes vont à ma famille ; à mon père Đình Diệu qui est une source d'inspiration dès mon enfance, à ma mère Xuân Hương pour son soutien continue, à mes soeurs Quỳnh Dương et Hà Dương et leur famille pour m'avoir toujours soutenu depuis mes premiers pas en France. Enfin, mes plus tendres mercis à Thùy Linh et à Đình Khánh, la seule personne citée dans cette thèse qui ne sait pas encore lire, cette thèse est à toi, mon petit.

Abstract

In this thesis, we consider a generalization of the encryption from "one-to-one" to "one-to-many" communication. The objective is to allow a center to send secret messages to a large number of receivers. The security notions in "one-to-many" communications need to be extended beyond the notion of confidentiality in "one-to-one" encryption to meet practical requirements. Two main functionalites are studied: traitor tracing which identifies malicious users who leak their secrets to a pirate and broadcast encryption which prevents non-legitimate or revoked users from decrypting broadcasted information.

In the first part of this thesis, we focus on combinatorial schemes. Our objective is to design solutions that support both the functionalities of broadcast encryption and traitor tracing against various pirate strategies. In one direction, we introduce a trace&revoke code and a tracing technique called "shadow group testing" to deal with "smart" pirates. In another direction, we propose a method to integrate revocation into some code-based schemes.

The second part discusses the techniques for constructing algebraic schemes. We first extend some well-known schemes, in particular the pairing-based BGW one, in order to enhance the security and to capture new properties. We then propose the first lattice-based traitor tracing of which the security is based on the hardness of the Learning With Errors problem. We finally consider the combination of algebraic and combinatorial methods and propose an optimal ciphertext rate traitor tracing scheme.

Finally, in the third part of the thesis, we propose an extended attack model, namely Pirates 2.0, that goes beyond the formalism of the conventional attacks. We also propose some generalized primitives for broadcast encryption and traitor tracing to fit new practical requirements such as multi-channel and decentralized broadcast encryption.

Keywords: Provable security, broadcast encryption, traitor tracing, lattice, pairings, combinatoric.

Résumé

Nous considérons dans cette thèse une généralisation du chiffrement au cas d'utilisateurs multiples, à savoir la diffusion de données chiffrées. Cette généralisation du chiffrement introduit deux nouveaux problèmes au-delà de la confidentialité : comment le centre peut-il identifier les abonnés malhonnêtes (qui fabriquent des décodeurs pirates et sont appelés traîtres) et comment le centre peut-il révoquer les abonnés malhonnêtes sans avoir besoin de mettre à jour les paramètres du système.

Dans un premier temps, nous prenons l'approche combinatoire dans le but de construire des schémas qui supportent à la fois la traçabilité et la révocation. Nous avons en particulier introduit un nouveau type de code, nommé trace&revoke code, et la technique de "shadow group testing" pour contrer les pirates "intelligents". Nous avons en outre proposé une méthode pour intégrer la révocation à quelques schémas de traçage de traîtres fondés sur les codes.

Dans un deuxième temps, nous suivons l'approche algébrique. Tout d'abord, en considérant les schémas fondés sur les couplages sur des courbes elliptiques, nous renforçons la sécurité du schéma de Boneh-Gentry-Waters et le rendons dynamique. Nous étudions ensuite l'application des réseaux euclidiens et proposons un schéma de traçage de traîtres dont la sécurité est assurée sous l'hypothèse bien connue de LWE (Learning with errors).

La dernière partie de la thèse est consacrée à la présentation d'un nouveau type d'attaque en collaboration publique, appelé attaque Pirates 2.0 et quelques extensions du modèle de diffusion de données qui répondent aux exigences pratiques comme les schémas décentralisés ou les schémas multi-canaux.

Mots-clés : Sécurité prouvée, diffudion de donées chiffrées, traçage de traîtres,réseaux euclidiens,couplages sur des courbes elliptiques.

Contents

I Some Advances in Broadcast Encryption and Traitor Tracing **1**

1 Introduction **3**
1.1 Broadcast Encryption & Traitor Tracing . . . 3
1.2 Provable Security: a Rigorous Analysis of Security in Cryptographic Systems . . 4
1.3 Security Notions for Broadcast Encryption & Traitor Tracing . . . 5
1.3.1 Definitions . . . 6
1.3.2 Security notions . . . 7
1.4 Short Overview of Broadcast Encryption & Traitor Tracing . . . 8

2 Combinatorial Approach **11**
2.1 Tree-based Constructions . . . 11
2.1.1 Brief Description of the Subset-Cover Framework . . . 11
2.1.2 Complete Subtree Scheme . . . 12
2.1.3 Subset Difference Scheme . . . 12
2.2 Code based Traitor tracing . . . 13
2.2.1 IPP codes . . . 14
2.2.2 Tardos' construction . . . 15
2.2.3 Code-based traitor tracing . . . 15
2.3 Black-Box Trace & Revoke Codes [NPP13] . . . 16
2.3.1 The construction . . . 17
2.3.2 Summary . . . 19
2.4 Trace&Revoke from linear codes . . . 19

3 Algebraic Approach **25**
3.1 From ElGamal encryption to multi-receiver encryption, trator tracing and revoke schemes . . . 26
3.1.1 Boneh-Franklin method for traitor tracing [BF99b] . . . 26
3.1.2 Naor-Pinkas method for revocation [NP00] . . . 27
3.2 Dealing with Full Collusion . . . 27
3.2.1 Broadcast encryption: BGW scheme . . . 27
3.2.2 Traitor Tracing: BSW scheme . . . 29
3.3 Some variants of BGW . . . 30
3.3.1 Adaptive CCA Security with Constant-size Secret Keys and Ciphertexts [PPSS13] . . . 30
3.3.2 BGW in Multi-Channel setting [PPT13] . . . 31
3.4 Lattice-based Approach: ℓ-LWE and Projective Sampling [LPSS14] . . . 33
3.4.1 Tracing traitors . . . 35
3.4.2 Hardness of k-LWE . . . 36
3.4.3 Hardness proof of Boneh-Freeman with exponential loss . . . 36

3.4.4 Our reduction with polynomial loss . . . 37
3.4.5 Public traceability . . . 37
3.5 Optimal transmission rate in Traitor tracing . . . 38
3.5.1 Constant Transmission Rate in Traitor Tracing . . . 38
3.5.2 Optimal Transmission Rate [FNP07b] . . . 39
3.5.3 Message Tracing with Optimal Transmission Rate [PPS12b] . . . 40
3.5.4 2-user Anonymous Broadcast Encryption . . . 41

4 Discussions and Perspectives **43**
4.1 Extended Attack Models . . . 43
4.2 Generalised Primitives . . . 44
4.3 Some Remarks and Open Problems . . . 45
4.4 Perspectives . . . 46

II Curriculum vitæ & publications **49**

III Appendix: Articles **59**

A Black-box Trace&Revoke Codes **61**
A.1 Introduction . . . 61
A.2 Revocable Codes . . . 64
A.3 Traceable Codes . . . 67
A.4 Trace&Revoke Codes . . . 72
A.5 Constructions of black-box Trace and Revoke with (r, s)-disjunct matrices . . . 74
A.6 Discussions . . . 82
A.7 Appendix: Basic Definitions . . . 83

B Traitor Tracing with Optimal Transmission Rate **87**
B.1 Introduction . . . 87
B.2 Preliminaries . . . 90
B.3 Public-Key Traitor Tracing Scheme with Public Traceability . . . 90
B.4 Public-Key Traitor Tracing with Public Traceability, Black-Box Tracing and Optimal Transmission Rate . . . 92
B.5 Space and Time Parameters in a Concrete Instantiation . . . 102
B.6 Conclusion . . . 102
B.7 Appendix . . . 103

C Hardness of k-LWE and Applications in Traitor Tracing **115**
C.1 Introduction . . . 115
C.2 Preliminaries . . . 117
C.3 New lattice tools . . . 121
C.4 A lattice-based public-key traitor tracing scheme . . . 126
C.5 Projective sampling and public traceability . . . 128
C.6 Appendix . . . 130
C.7 Traitor Tracing . . . 130
C.8 Basic results on lattices . . . 136
C.9 Missing proofs of Section C.3 . . . 136
C.10 Missing proof of Section C.5 . . . 141

D Adaptive CCA Broadcast Encryption with Constant-Size Secret Keys and Ciphertexts **143**
D.1 Introduction . . . 143
D.2 Preliminaries . . . 146
D.3 CCA from Generic Transforms? . . . 147
D.4 An Efficient Selective CCA Broadcast Encryption . . . 148
D.5 Inclusive-Exclusive Broadcast Encryption . . . 150
D.6 Achieving Adaptive CCA Security . . . 151
D.7 Concluding Remarks . . . 154
D.8 Appendix . . . 155

E Message Tracing with Optimal Ciphertext Rate **165**
E.1 Definitions . . . 168
E.2 A Generic Construction from PKE . . . 171
E.3 A Construction With Shorter Keys . . . 175
E.4 Conclusion . . . 181
E.5 Appendix . . . 181

F Traitors Collaborating in Public: Pirates 2.0 **185**
F.1 Introduction . . . 185
F.2 Formalization of Pirates 2.0 . . . 188
F.3 Pirates 2.0 and the Subset-Cover Framework . . . 190
F.4 Pirates 2.0 and Code Based Schemes . . . 196
F.5 Conclusion . . . 197

G Identity-Based Traitor Tracing **199**
G.1 Introduction . . . 199
G.2 Preliminaries . . . 200
G.3 Identity-Based Traitor Tracing . . . 201
G.4 The Scheme . . . 203
G.5 Security Results . . . 206
G.6 Appendix: Waters' HIBE with Asymmetric Pairings . . . 210

H Multi-Channel Broadcast Encryption **213**
H.1 Introduction . . . 213
H.2 Multi-Channel Broadcast Encryption . . . 216
H.3 Preliminaries . . . 218
H.4 Multi-Channel Broadcast Encryption I – MCBE_1 . . . 220
H.5 Multi-Channel Broadcast Encryption II – MCBE_2 . . . 223
H.6 Conclusion . . . 227

I Decentralized Dynamic Broadcast Encryption **229**
I.1 Introduction . . . 229
I.2 Definitions . . . 231
I.3 Generic Decentralized Broadcast Encryption . . . 236
I.4 Tree-based Subgroup Key Exchange . . . 238
I.5 Concrete Instantiations . . . 241
I.6 Appendix . . . 242

Bibliography **253**

Part I

Some Advances in Broadcast Encryption and Traitor Tracing

Chapter 1

Introduction

1.1 Broadcast Encryption & Traitor Tracing

The oldest goal of cryptography is to allow parties to communicate in a secured manner over an insecure channel which might be under adversarial control. Nowadays, confidentiality remains one of the main goals, besides authentication and integrity. Almost all standard protocols for confidentiality, termed encryption, are implemented in a "one-to-one" communication framework: a sender encrypts the message and sends the ciphertext to a receiver who has the secret key to decrypt the ciphertext. The objective, from a security point of view, is to prevent an outside attacker (who observes and may be able to interact with the system) to break the confidentiality, *i.e.* to recover some information about the original message. The situation will not be exactly the same when one generalises "one-to-one" to "one-to-many" communication, aka multi-receiver encryption where the sender needs to send a secret message to many receivers. One might think that the trivial solution consisting of sharing a common secret key among all legitimate receivers would be sufficient. However, this is not the case, mainly because the security notions in "one-to-many" communications need to be extended to meet practical requirements. As the old saying goes: when a secret is known by more than one person, it is not a secret anymore. Therefore, if a common secret key is shared among all the receivers, then one of the receivers can give it to the adversary. Consequently, on the one hand, the confidentiality of the whole system is totally broken and on the other hand, we have no idea who the source of secret leakage is and we can not detect and exclude this dishonest user (commonly called a traitor), since all the receivers have the same secret key.

In "one-to-many" communications, there are new fundamental security requirements for the security to deal with access control and traceability.

- Access control assures that only legitimate or targeted users have the right to decrypt the message. The resulting schemes are generally called *broadcast encryption* (BE in short). In practical applications such as pay-TV, the targeted set is often very large and contain almost all users except some non-paying ones (who should be revoked from the system), the targeted set is implicitly determined via the revoked set and the corresponding system is commonly called a *revoke scheme*.

- While access control is quite natural to be considered in broadcast encryption, traceability is really a new property which is "orthogonal" to the main objectives of classical cryptographic systems. Since one cannot totally prevent receivers from leaking their secret keys in "one-to-many" communications, we should discourage them from doing this. In fact, when a user joins the system and commits himself to respect the security requirement by not revealing any secret information; and if the user knows that the source of any secret in-

formation leakage will be detected, then dishonest users are deterred from revealing their secret. A multi-receiver encryption scheme with the ability to trace traitors is called a *traitor tracing* (TT in short).

More formally, in "one-to-many" communications, two main primitives have been studied: broadcast encryption prevents non-legitimate users from decrypting broadcasted information and traitor tracing discourages malicious users from leaking their secrets to a pirate. In these primitives, each user receives a decryption box, called a decoder, containing the secrets that help to decrypt the broadcasted ciphertext. Broadcast encryption enables the center (*i.e.* broadcaster) to choose any set of legitimate users to decrypt the broadcasted information. Traitor tracing provides a way of embedding different secrets into each user's decryption box so that even if several traitors collude to produce a pirate decoder, the authority will still have the capacity to trace at least one of them.

Before giving an overview of broadcast encryption and traitor tracing, let us discuss some impacts in practice of these primitives.

Practical impact. Among many cryptographic primitives which have been proposed since the introduction of public-key encryption, broadcast encryption and traitor tracing have received quite a lot of attention due to their practical impact, especially in pay-TV and in positioning systems. In the context of pay-TV, piracy has an increasingly alarming and direct impact on the revenues of broadcasters. According to AEPOC ("Association Européenne pour la Protection des Oeuvres et services Cryptés"), about € 1 billion are spent in the EU alone to acquire pirate equipments every year. Another report from Datamonitor estimates that between 2004 and 2010, the loss for broadcast operators would have been around € 681 millions for a € 3.2 billions benefit over the same period. Recent years have witnessed the emergence of a growing global black economy based on piracy. In the context of positioning system, we can look at GALILEO European project to build a global navigation satellite system (whose cost is estimated at € 3.4 billions). This project aims to be an alternative to the American GPS and broadcast encryption schemes are at the core of Galileo to operate group management i.e. to allow or deny access to some of its services. Depending on the practical requirements, the prime objective might be very different from one system to another:

- in the commercial domain, the quality of service is more important than security: the overall goal is to maintain a good quality of service while minimising the financial loss due to piracy;
- in the military domain, the safety and preservation of governmental interests are the prime concerns. If a technical solution cannot achieve a sufficient level of security with respect to that purpose, it will not be implemented.

Due to a large number of potential multi-receiver scenarios, it is highly unlikely that a single solution will fit them all. This motivates a trade-off between efficiency parameters and security levels. Our goal is to construct schemes which are flexible enough to fit a variety of scenarios in a way that is optimal (or close to optimal) and of which the security levels are rigorously investigated. For the latter, we provide a quick overview of the vast domain of provable security.

1.2 Provable Security: a Rigorous Analysis of Security in Cryptographic Systems

Cryptography has a very long history and the traditional way to design a cryptographic protocol is by "trial and error": a protocol is proposed and one tries to break it; if the protocol resists

all attacks for a while then it is considered secure. Unfortunately, history has taught us that this is not an appropriate way: many protocols (for example the Chor-Rivest scheme) have been broken many years after they were believed secure. Only about 30 years ago, a fundamental and radical idea was proposed by Goldwasser and Micali [GM84], followed by Blum, Micali [BM84] and Yao [Yao82], suggesting that the security could be proved under standard and well believed complexity theoretic assumptions, *e.g.* the assumed hardness of factoring. The methodology they proposed has come to be known as provable security.

Security notions have been defined for cryptographic primitives. In our context of broadcast encryption, we will mainly focus on confidentiality. The main goal for any encryption scheme is secrecy: ideally, such a notion means that a ciphertext should not reveal any information about the plaintext, no matter how powerful the adversary is. This had been defined under the term "perfect secrecy" [Sha49], but also showed to be impossible unless one uses one-time pad, which is a symmetric encryption that uses a secret key as long as the message to be encrypted. That is, if one wants either to use a small symmetric key in order to protect many plaintexts or a long message, or to use an asymmetric encryption, such perfect secrecy is impossible.

To overcome this theoretical impossibility which has no real practical impact (since adversaries are computationally limited), several security notions have thereafter been defined, namely the polynomial security [GM84], *a.k.a.* indistinguishability of ciphertexts or semantic security. The semantic security intuitively means that no *polynomially* bounded adversary can extract any information about the plaintext from the ciphertext. Indistinguishability was indeed defined in the basic scenario only, where the adversary only has access to public information (in the public-key encryption setting, the adversary can thus encrypt any plaintext of its choice, hence the name of chosen-plaintext attacks, denoted CPA.) Naor and Yung [NY90] introduce the notion of chosen-ciphertext attacks. However, they restrict the chosen-ciphertext attacks to be non-adaptive, in the sense that the decryption queries can not depend on the challenge ciphertext (*a.k.a. lunchtime attacks*, denoted CCA1.) Rackoff and Simon [RS91] extend this notion, with an unlimited access to the decryption oracle (excepted on the challenge ciphertext), denoted CCA2, and provide a candidate with non-interactive zero-knowledge proofs of knowledge. By now, the widely admitted appropriate security level for asymmetric encryption is the chosen-ciphertext security (IND-CCA2, or CCA) which is actually the semantic security against adaptive chosen-ciphertext attacks. In order to achieve semantic security, even in the basic chosen-plaintext scenario, the encryption algorithm must be probabilistic, which means that a given plaintext (with a fixed public key) could be encrypted in many different ways (at least 2^k different ciphertexts if 2^{-k} is the expected security level).

In the context of multi-receiver encryption, the most desired security level remains the semantic security against adaptive chosen-ciphertext attacks. However, the adversarial model needs to be extended. Indeed, the adversaries can have access to encryption/decryption oracles but they can also corrupt legitimate users. In fact, in classical (single-receiver) encryption, if the receiver is corrupted then the system collapses but in multi-receiver encryption, it is required that even if a pirate corrupts many receivers, the security should still hold for the remaining users. In the third part of this introduction, we will discuss the security notions for broadcast encryption and traitor tracing which are the basis of our results.

1.3 Security Notions for Broadcast Encryption & Traitor Tracing

The main goal of a BE scheme is to enable the sender of a message to choose any subset of users (called the *target set* or the *privileged set*) to which the message will be encrypted. The target set can be directly determined by the sender or can be implicitly determined via its complement

- the revoked set. In the latter case, the resulting scheme is called a revoke scheme.

Theoretically, it requires N bits to uniquely identify a subset of a set of size N. However, if the size r of the revoked set is small, it is sufficient to identify the revoked users, which can be done using $r \log N$ bits. The same technique applies if the target set is small. In practice, we should notice that the target set is quite stable (for example, in pay-TV, the target set is almost stable during the whole month) and we only need to communicate the modification of the target set between two periods. It was often sufficient to consider group key distribution where one user is added to or removed from the target group: it corresponds to a broadcast encryption where one user is added or removed from the target set. In general, it is widely accepted that the size of the description of the target set is not taken into account when broadcast encryption schemes are compared.

We now give formal definitions for BE schemes as key encapsulation mechanisms and define the relevant security notions.

1.3.1 Definitions

Broadcast encryption is conventionally formalised as *broadcast encapsulation* in which a session key is produced and this session key is required to be indistinguishable from random, under the adversarial view. Such a scheme can provide public encryption functionality in combination with a symmetric encryption through the hybrid encryption (a.k.a. KEM-DEM) paradigm [CS03]. We hence use the terms encryption and encapsulation, key header and ciphertext interchangeably.

Definition 1.3.1 [Broadcast Encryption Scheme] a (public-key) *dynamic* broadcast encapsulation scheme is a tuple of four algorithms $\mathsf{BE} = (\mathsf{Setup}, \mathsf{Join}, \mathsf{Encaps}, \mathsf{Decaps})$ where:

- $\mathsf{Setup}(1^k)$ outputs $(\mathsf{msk}, \mathsf{ek})$ containing the master secret key and the (initial) encryption key;
- $\mathsf{Join}(\mathsf{msk}, i)$ outputs the key pair (sk_i, pk_i) for user i, and updates system parameters to include the information of the users i (by appending pk_i to ek and sk_i to msk).
- $\mathsf{Encaps}(\mathsf{ek}, S)$ for a set of users S outputs (H, K) containing a ciphertext (a.k.a. key header) and a session key (for a revoke scheme, replace S with R.)
- $\mathsf{Decaps}(\mathsf{ek}, sk_i, S, H)$ outputs K if $i \in S$ (or $i \notin R$ in a revoke scheme) and $\perp$ otherwise.

In some static schemes, the setup algorithm takes as input N as the number of users and returns the secret keys for all users. This can be made to fit our definition by defining msk to contain the concatenation of the user secret keys and defining Join to simply return the i-th key contained in msk.

The correctness requirement is that for all subset S of users and for any $i \in S$:
If $[(\mathsf{msk}, \mathsf{ek}) \leftarrow \mathsf{Setup}(1^k), K_i \leftarrow \mathsf{Join}(\mathsf{msk}, i), (H, K) \leftarrow \mathsf{Encaps}(\mathsf{ek}, S)]$ then $\mathsf{Decaps}(K_i, S, H) = K$. For revoke schemes, the definition is the same except that S is replaced with R, and we require that $i \notin R$.

We now discuss traitor tracing and more generally, trace&revoke schemes.

Definition 1.3.2 [Trace&Revoke Scheme] A trace&revoke encapsulation scheme is a broadcast encapsulation scheme with an additional tracing algorithm $\mathsf{Trace}^{\mathbb{D}}(R_{\mathbb{D}}, \mathsf{pk}, \mathsf{msk})$: the traitor tracing algorithm interacts in a black-box manner with a pirate decoder $\mathbb{D}$ that is built from a certain set T of traitors. The algorithm takes as input a subset $R_{\mathbb{D}} \subset [N]$ (suppose that, at the time of tracing, there are N users in the system and $R_{\mathbb{D}}$ can be adversarially chosen), the public key pk, the master key msk, and outputs a set $T_{\mathbb{D}} \subseteq [N]$. Precisely, under the conditions:

- There are at most t traitors: $|T| \leq t$;
- The minimal revoked set does not contain all the traitors: $T \not\subseteq R_{\mathbb{D}}$, or equivalently $S_{\mathbb{D}} = ([N] - R_{\mathbb{D}})$ contains at least one traitor;
- $\mathbb{D}$ is "efficient" to decrypt ciphertexts (*i.e.* decrypts with some non-negligible probability) for some revoked sets R that include the minimal revoked set $R_{\mathbb{D}}$ but do not contain all the traitors ($R_{\mathbb{D}} \subseteq R$ but $T \not\subseteq R$);

then the tracing algorithm outputs at least one traitor in $S_{\mathbb{D}}$, *i.e.* $\emptyset \neq T_{\mathbb{D}} \subseteq T \cap S_{\mathbb{D}}$.

The above definition captures both the functionalities of revoking users and tracing traitors in a general black-box model. However, there are many others models for tracing such as *non-black-box tracing*, *single-key black box tracing* [BF99b] models and tracing for stateful pirates [KY02b]. The objective of the tracing procedure could also be relaxed in some situation where it might be sufficient for the authority to disable pirate decoders. We would refer to the Kiayias and Pehliganoglu's book [KP10] for an overview of different types of tracing games.

A traitor tracing scheme is in fact a trace&revoke scheme without the possibility to revoke users, namely the target set is always set to be the whole set of users. The combination of traceability and revocability is challenging and they are often studied in separated ways. In our works, on the one hand, we continue to investigate these properties in independent ways and on the other hand, we try to combine them to achieve trace&revoke schemes.

1.3.2 Security notions

We define the strongest security model, namely the adaptive CCA security game for a dynamic broadcast encryption $\mathrm{Exp}_k^{aBE-CCA}$. The game involves five phases:

1. Setup. The environment runs $\mathsf{Setup}(1^k)$ to initialise the system and gives the adversary the encryption key. (For symmetric schemes, the adversary receives access to an encryption oracle instead.)
2. Query 1. The adversary has access to an $\mathsf{ODecaps}$, an OJoin and an $\mathsf{OCorrupt}$ oracles. He can query these oracles adaptively to: decrypt a chosen ciphertext; join new users to the system; corrupt a subset of them and receive all their secrets. (In a BE scheme that is not dynamic, there is no Join oracle as all users are created during setup.)
3. Challenge. The adversary outputs a set S of receivers it wants to attack. S must not contain any user for which the adversary has already obtained the decryption keys. The environment obtains $(H, K) \leftarrow \mathsf{Encaps}(\mathsf{ek}, S)$. It flips a coin $b \xleftarrow{\$} \{0,1\}$ and sets $K_b = K$, $K_{1-b} \xleftarrow{\$} \mathcal{K}$. Then it returns (S, H, K_0, K_1) to the adversary.
4. Query 2. This is the same as the key query 1 phase, except that the adversary cannot corrupt users in S.
5. Guess. The adversary outputs his guess bit b'. If he has corrupted users in the set S or queried the decryption oracle on the challenge header, the experiment aborts and outputs $\perp$. The experiment outputs 1 if $b = b'$, else 0.

We define the advantage of the adversary as

$$\mathrm{Adv}_k^{aBE-CCA}(\mathcal{A}) = |\Pr[\mathrm{Exp}_{N,k}^{aBE-CCA} = 1] - \frac{1}{2}|.$$

Definition 1.3.3 A BE scheme is (t, ϵ, q_D, q_K)-secure, if for any t-time adversary $\mathcal{A}$ who makes a total of q_D decryption queries and q_K key queries $\mathrm{Adv}_k^{aBE-CCA}(\mathcal{A}) < \epsilon$.

In some security models, the adversary is given only one key, which is either the correct key or a random element from the key set. We call this version the "real-or-random" (ROR) game, and the experiment described above the "left-or-right" (LOR) game, in analogy to the security definitions for encryption in [BDJR97]. We can show in the same fashion that the two notions are essentially equivalent, with a factor 2-loss in the reduction from LOR to ROR.

We dropped the requirement that $S' \subset S$ for the decryption oracle from [BGW05]. The notions are equivalent in the static security model used in the BGW paper, because the adversary already knows all the decryption keys for users not in S and can therefore decrypt all the other messages himself. The restriction seems artificial and is probably an artifact of the security proof.

In the CCA1 version, the adversary has access to the decryption oracle only before the challenge phase. In the CPA version, the adversary does not have access to the decryption oracle.

1.4 Short Overview of Broadcast Encryption & Traitor Tracing

In the next chapters, we will discuss the most relevant techniques in designing a BE or TT scheme and our contributions. Here we only give a short overview of the domain.

Broadcast encryption has first been described by Fiat and Naor in [FN93]. BE has not received much attention until the last decade, when Naor, Naor, and Lotspiech presented their (symmetric-key) subset-cover framework along with a security model and a security analysis [NNL01]. Since then, many BE schemes have been proposed and the subset cover framework has become the basis for many subsequent proposals, including [DF03] which proposes the first public key broadcast encryption.

Boneh, Gentry, and Waters [BGW05] are first to propose a fully collusion-resistant public key broadcast encryption in which the ciphertext size is constant. They proposed two schemes, respectively CPA and CCA secure, both in the selective model of security where the adversary is required to choose the corrupted users before the set up.

Adaptive security is proposed by [GW09] where the authors give several schemes which achieve adaptive CPA security, including two broadcast encryption schemes and two identity-based broadcast encryption (IBBE) schemes, one of them achieves constant-size ciphertexts in the random oracle model. The schemes proposed in [Wat09] and [LSW10], respectively a broadcast encryption and a revocation scheme, are the only secure schemes under static assumptions (as opposed to the so called q-based ones). [LSW10] also proposes an identity-based revocation scheme which is proved selective CPA secure.

Dynamic broadcast encryption is proposed in [DPP07] where they design CPA secure schemes that are only partially adaptive secure. Strictly speaking, their scheme is a *revocation* scheme in which the set of revoked users is selected at the time of encryption, and in turn, any user outside of the revoked set is able to decrypt. [Del07] proposes identity-based broadcast encryption and gives a selective CPA secure scheme. Based on BGW scheme, we propose in [PPSS13] a constant size adaptive CCA secure *inclusive-exclusive* broadcast encryption scheme which can act both as a broadcast encryption and as a revocation scheme at the same time.

The first formal definition of traitor tracing scheme appears in Chor *et al.* [CFN94b, CFNP00] in which the construction requires storage, decryption time complexity of $O(t^2 \log^2 t \log(N/t))$ and communication complexity of $O(t^3 \log^4 t \log(N/t))$, where N is the size of the users and t is the upper bound on the number of traitors. Stinson and Wei latter suggest

in [SW98a] explicit combinatorial constructions that achieve better efficiency for small values of t and N.

In [BF99b], Boneh and Franklin present an efficient public-key traitor tracing scheme with deterministic t-tracing based on an algebraic approach. Its communication, storage and decryption complexities are all $O(t)$. The authors also introduce the notion of *non-black-box traceability*: given a "valid" key extracted from a pirate device (constructed using the keys of at most t users), recover the identity of at least one traitor. This is in contrast with the notion of *black-box tracing* where the traitor's identity can be uncovered by only observing the pirate decoder's replies on "well crafted" ciphertexts. Unfortunately, Kiayias and Yung [KY01c] show that black-box tracing cannot be efficient (say, in poly-time) in this type of scheme whenever the number of traitors is superlogarithmic. The Boneh-Franklin scheme can however achieve black-box confirmation: given a superset of the traitors, it is guaranteed to find at least one traitor and no innocent suspect is incriminated. Boneh *et al.* [BSW06b, BW06b] propose traitor tracing schemes that withstand any number of traitors (*full traceability*) while requiring a sub-linear ciphertext length ($O(\sqrt{N})$). Very recently, Boneh and Zhandry [BZ14] propose a fully collusion resistant scheme with poly-log size parameters. It relies on indistinguishability obfuscation [GGH$^+$13c], of which security foundation remains to be studied and practicality remains to be exhibited.

In [Pfi96], Pfitzmann introduces the notion of *asymmetric* traitor tracing. In this model, the tracer uncovers some secret information about the traitor that was *a priori* unknown to the system manager. Thus, the result of the tracing algorithm provides evidence of the treachery. Further results in this direction are in [KD98b, KY02d, KY02a]. We put forth the notion of public traceability, i.e., the possibility of running tracing procedure on public information. Some schemes [CPP05a, PSNT06b, BW06b, BZ14, LPSS14] achieve public traceability and some others achieve a stronger notion than public traceability, namely the non-repudation, but the setup in these schemes require some interactive protocol between the center and each user such as a secure 2-party computation protocol in [Pfi96], a commitment protocol in [PW97] or an oblivious polynomial evaluation in [WHI01, KWHI01, KY02a].

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

A class of schemes relying on the use of collusion secure codes [BS95, BS98, Tar03] has been introduced by Kiayias and Yung in [KY02c]. These code-based schemes enjoy many nice and desirable properties: they support black-box tracing and the ratio between the ciphertexts and the plaintexts is constant. However, since these schemes use collusion secure codes for both the ciphertext and the key used in the decoders, the sizes of the ciphertexts and keys are quite large. Another drawback of [KY02c, CPP05a] comes from the use of an all-or-nothing transform (AONT [Riv97]) to prevent deletion of keys from the pirate decoders as a way to escape the tracing procedure based on the underlying collusion secure code. Based on robust codes [SNW03b, Sir07a, BN08b, Nui09], Boneh and Naor [BN08b] and us [BP08] independently improve the Sirvent scheme to make the ciphertext size constant. These schemes become quite competitive but their drawbacks remain the large private key size and their weak resistance to collaborative attacks [BP09].

Recently, in [LPSS14], we introduce the first lattice based traitor tracing in the bounded model in which the security relies on the hardness of a new variant of the LWE problem, called k-LWE.

As originally observed in [GSY99], traitor tracing schemes are most useful when combined with revocation schemes; such trace&revoke approach consists in first uncovering the compromised decryption keys and then revoking their decryption capabilities, thus making pirate decoders useless. We can name some schemes in this category [NP00, TT01, NNL01, DF02, DF03, KHL03, DFKY05, BW06b, NPP13]. The construction of practical trace&revoke schemes still

remains a challenge.

Organisation of the thesis. Overall, broadcast encryption and traitor tracing schemes can be categorised into two main classes: combinatoric schemes and algebraic schemes. The first chapter is devoted to combinatorial schemes; the second one deals with algebraic schemes and also with schemes that combine both algebraic and combinatorial structures. In each chapter, we present the state of the art and our contributions. Finally, the third chapter is devoted to discussing about new attack models, generalised models of broadcast encryption and perspectives for our future works.

Chapter 2

Combinatorial Approach

Combinatorial broadcast encryption schemes are mainly based on a tree structure or on a fingerprinting code. Tree-based schemes support revocation but have limited capacity dealing with tracing traitors, while code based ones provide traceability but not revocation. Our objective is to propose methods that can support both traceability and revocation. In one direction, we introduce a trace&revoke code and in another direction, we integrate revocation into some code based schemes. We also propose efficient code based schemes with optimal transmission rate but because these schemes require a combination with some algebraic structures, we postpone the presentation to the end of the next chapter on algebraic schemes.

In Sections 2.1 and 2.2 we present the state of the art in constructing combinatorial schemes and then, in Sections 2.3 and 2.4, we present our contributions.

2.1 Tree-based Constructions

The subset-cover framework proposed by Naor, Naor, and Lotspiech in [NNL01] is a powerful tool to design efficient trace&revoke systems. It captures the ideas in previously proposed traitor tracing systems and forms the basis of the so called NNL scheme used in the content protection system for HD-DVDs known as AACS [AACa].

2.1.1 Brief Description of the Subset-Cover Framework

The subset-cover framework is a powerful mean to capture several trace&revoke designs. It encompasses several traitor tracing schemes proposed to date and maybe even more importantly, serves as the basis for two of the most efficient trace&revoke schemes: the complete subtree scheme and the subset difference scheme.

In the subset-cover framework, the set $\mathtt{N}$ of users in the system is covered by a collection of subsets $\mathtt{S}_i$ such that $\cup_i \mathtt{S}_i \supset \mathtt{N}$ and $\mathtt{S}_i \cap \mathtt{N} \neq \emptyset$. This covering is not a partition of $\mathtt{N}$ and the sets $\mathtt{S}_i$ rather overlap. To every subset $\mathtt{S}_i$ corresponds a long term secret key L_i, and every user that belongs to $\mathtt{S}_i$ is provided with this secret key—or in an equivalent way, with some materials that allow him to derive this secret key. Therefore, every user u of the system is given a collection of long term keys $\{L_{i_k}\}$ that together form his secret key which we denote by sk_u.

In order to broadcast a content M, the center uses a standard hybrid scheme: a session key K is first drawn randomly and used to encrypt the content (with an encryption scheme E') before being encapsulated under multiple long term keys (with another encryption scheme E). The long term keys L_{i_k}, $k = 1, \ldots, l$ are chosen so that the corresponding subsets $\mathtt{S}_{i_1}, \ldots, \mathtt{S}_{i_l}$ only cover the set of users entitled to decrypt. Therefore, the center broadcasts ciphertexts of

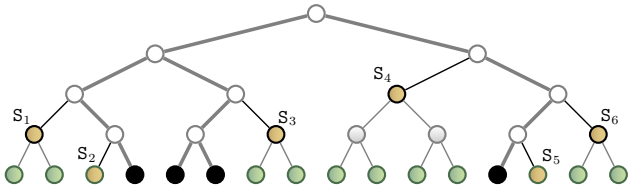


Figure 2.1: Complete subtree: leaves correspond to users, $\mathtt{S}_1, \ldots, \mathtt{S}_6$ consist of the covering that excludes revoked users (in black) while allowing other users to decrypt.This is derived from the Steiner tree associated to the set of revoked users $\mathtt{R}$.

the form:

$$\Big[(i_1, E_{L_{i_1}}(K)),\ (i_2, E_{L_{i_2}}(K)),\ \ldots,\ (i_l, E_{L_{i_l}}(K)) \parallel E'_K(M) \Big]$$

To decrypt, a valid decoder for user u performs the following sequence of operations: it first looks for an index i_j in the first element of each of the l couples $(i_k, E_{i_k}(K))$ in turn such that $\mathtt{S}_{i_j} \subset sk_u$. If no index corresponds, the decoder does not decrypt; otherwise, the decoder retrieves the corresponding long term key L_{i_j} and uses it to decrypt the associated encrypted session key $E_{i_j}(K)$ and then decrypts the payload $E'_K(M)$.

Since the system is built to handle revoked users, let us also denote by $\mathtt{R}$ the set of revoked users in the system at any point in time. In order to prevent them (individually but also together as a collusion) from accessing the encrypted content $E'_K(M)$, the collection $\mathtt{S}_{i_1}, \ldots, \mathtt{S}_{i_l}$ is specially crafted so that:

$$\bigcup_{k=1}^{l} \mathtt{S}_{i_k} = \mathtt{N} \setminus \mathtt{R} \ .$$

2.1.2 Complete Subtree Scheme

In this scheme, the users correspond to the leaves of a complete binary tree whereas the collection of subsets $\mathtt{S}_i$ exactly corresponds to all the possible subtrees in the complete tree. When $|\mathtt{N}| = 2^n$, the complete binary tree is of length n and there are exactly n subtrees that contain a given leaf. Figure 2.1 shows a covering using six subsets of twelve users that excludes four revoked users (depicted in black). This subset scheme complies with the bifurcation property since any subset (or equivalently any subtree of the complete binary tree) can be split into two subsets of equal size (the two subtrees rooted at the two descendants of the root of the original subtree). Regarding the key assignment, each user represented by a leaf u in the complete binary tree is provided with the keys L_i associated to the nodes i on the path from the leaf u to the root.

Covering algorithm. In the case of the complete subtree, the covering used to exclude the $r = |\mathtt{R}|$ revoked users from $\mathtt{N}$ is the collection of subsets that hang off the Steiner tree of the revoked leaves. (The Steiner tree of the revoked leaves is the minimal subtree of the complete binary tree that connects all the revoked leaves to the root and it is unique.) Since any user only knows the keys from its leaf to the root and since this path is included in the Steiner tree for revoked users, these users cannot decrypt anymore. This algorithm produces covers of size $O(r \log(N/r))$.

2.1.3 Subset Difference Scheme

The subset difference scheme has been introduced to lower the number of subsets required to partition the set of legitimate users $\mathtt{N} \setminus \mathtt{R}$. It improves the above presented complete subtree scheme by a factor of $\log(N/r)$ in terms of bandwidth usage for the headers.

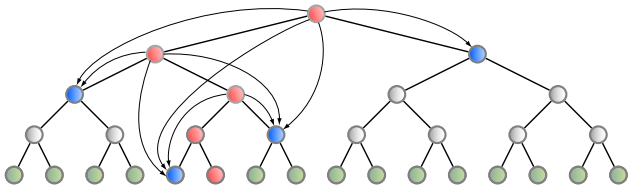


Figure 2.2: Key assignment. User u receives all the labels $\mathsf{LABEL}_{i,j}$ such that i is a parent of j and i is on the path from the leaf of u to the root.

To attain this level of performance, the number of possible subsets has been tremendously increased. Remember that $\mathtt{S}_i$ denotes the full binary subtree of the complete binary tree rooted at node i. Now, for each node j in $\mathtt{S}_i$ different from i, let us denote by $\mathtt{S}_{i,j}$ the binary subtree rooted at node i of which the full binary subtree rooted at node j has been removed. (See examples in Figure 2.3.) A user will need to know all the keys $L_{i,j}$ such that he belongs to the subtree rooted at i but not to the subtree rooted at j. However, it would be impossible for each device to store such a large number of long term keys. This is why a key derivation procedure has been designed to allow the derivation of most of the $O(N)$ long term keys: a user only needs to store $O(\log^2(N))$ keys. Each node i in the full binary tree is first assigned a random label LABEL_i, then labels $\mathsf{LABEL}_{i,j}$ together with their corresponding long term keys $L_{i,j}$ are deduced (in a pseudo-random way) from label LABEL_i. The key derivation procedure then works as follows: from each LABEL_i, a pseudo-random value $\mathsf{LABEL}_{i,j}$ is obtained for each sub-node j using the tree based construction proposed by Goldreich, Goldwasser, and Micali [GGM84]; from this value $\mathsf{LABEL}_{i,j}$, a long term key $L_{i,j}$ is eventually deduced (in a pseudo-random way). Each user is then provided with labels $\mathsf{LABEL}_{i,j}$ for all nodes i that are on the path from the leaf that represents the user to the root and all nodes j hanging off this path as described on Fig. 2.2. This key assignment ensures that every user in the subtree rooted at node i but not in the subtree rooted at node j is able to derive $L_{i,j}$ while every user in the subtree rooted at node j is not able to derive $L_{i,j}$.

Covering algorithm. The covering algorithm works by maintaining a subtree T of the Steiner tree of $\mathtt{R}$ and removes nodes from it at each step:

`init:` Make T the Steiner tree of $\mathtt{R}$.

`select:` If there is only one leaf v_k in T and it is not the root (or node 0), add the subset $\mathtt{S}_{0,k}$ and return. If there is only the root in T, return. Otherwise, select two leaves v_{j_1} and v_{j_2} from T so that their least common ancestor v does not contain any other leaves of T than v_{j_1} and v_{j_2}. Call v_{i_1} and v_{i_2} the children of v such that v_{i_1} is the ancestor of v_{j_1} and v_{i_2} the ancestor of v_{j_2}. Then, if $v_{i_1} \neq v_{j_1}$, add $\mathtt{S}_{i_1,j_1}$ to the partition and similarly if $v_{i_2} \neq v_{j_2}$, add $\mathtt{S}_{i_2,j_2}$ to the partition. Remove all the descendants of v from T, which makes v a leaf of T. Reiterate the step 'select'.

An example output of this procedure is shown in Figure 2.3.

2.2 Code based Traitor tracing

Fingerprinting with collusion secure codes allows one to identify a digital document among several copies of it by embedding a fingerprint (a codeword). Such an identification scheme must be resilient to collusions of traitors trying to remove their fingerprints so as to escape identification. Therefore, collusion secure codes share some properties with traitor tracing; However, the main assumption here (called the marking assumption) is that the traitors from a collusion are only able to identify the positions where the digits from their respective codewords differ; such positions are called *detectable positions*. This assumption especially makes sense

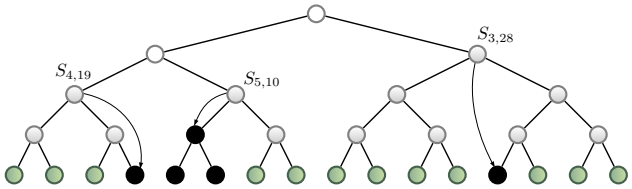


Figure 2.3: Subset difference: leaves correspond to users and black nodes are not able to derive the necessary information to decrypt. Therefore $\mathtt{S}_{4,19}$ prevents user 19 from decrypting, $\mathtt{S}_{5,10}$ prevents users 20 and 21 from decrypting, and $\mathtt{S}_{3,28}$ prevents user 28 from decrypting. All other users are able to decrypt.

with fingerprinting data: apart from the codewords, the documents are identical, and it is easy to uncover places where two copies of a document differ.

Among the first constructions are the identifiable parent property (IPP) codes introduced in [CFN94b]. These codes are defined over large alphabets and can be obtained from linear codes or from combinatorial constructions. If the condition that a traitor is always correctly identified in IPP can be relaxed, i.e. tracing algorithm may fail with some negligible probability, then more efficient construction can be achieved. Randomized collusion secure codes, which can be seen as "relaxed" binary IPP codes, have first been proposed by Boneh and Shaw in [BS95]. These codes are more efficient than linear codes based IPP codes. In Boneh-Shaw codes, the length of the codewords is $O(N^3 \log(N/\epsilon))$ for fully-collusion resistant codes and $O(c^4 \log(N/\epsilon))$ for codes resisting collusions of at most c traitors. Tardos latter introduces a new construction in [Tar03] and proves that the size of its codewords is optimal: a length of $O(c^2 \log(N/\epsilon))$ is enough to resist collusions of at most c traitors. This obviously gives fully-collusion secure codes of length $O(N^2 \log(N/\epsilon))$.

We will first give a definition of an IPP code, then a description of Tardos' construction and finally explain the general framework of constructing traitor tracing schemes which relies on any IPP code, including the most important case of collusion secure code.

2.2.1 IPP codes

Let $\mathcal{Q}$ be an alphabet set containing q symbols. If $C = \{w_1, w_2, \ldots, w_N\} \subset \mathcal{Q}^\ell$, then C is called a q-ary code of size N and length ℓ. Each $w_i \in C$ is called a codeword and we write $w_i = (w_{i,1}, w_{i,2}, \ldots, w_{i,\ell})$ where $w_{i,j} \in \mathcal{Q}$ is called the j^{th} component of the codeword w_i.

We define *descendants* of a subset of codewords as follows. Let $X \subset C$ and $u = (u_1, \ldots, u_\ell) \in \mathcal{Q}^\ell$. The word u is called a descendant of X if for any $1 \leq j \leq \ell$, the j^{th} component u_j of u is equal to a j^{th} component of a codeword in X. In this case, codewords in X are called *parent codewords* of u. For example, $(3, 2, 1, 3)$ is a descendant of the three codewords $(3, 1, 1, 2)$, $(1, 2, 1, 3)$ and $(2, 2, 2, 2)$. We denote by $\mathsf{Desc}(X)$ the set of all descendants of X. For a positive integer c, denote by $\mathsf{Desc}_c(C)$ the set of all descendants of subsets of up to c codewords. Codes with identifiable parent property (IPP codes) are defined below.

Definition 2.2.1 A code C is called c-IPP if, for any $u \in \mathsf{Desc}_c(C)$, there exists $w \in C$ such that for any $X \subset C$, if $|X| \leq c$ and $u \in \mathsf{Desc}(X)$ then $w \in X$.

In a c-IPP code, given a descendant $u \in \mathsf{Desc}_c(C)$, we can always identify at least one of its parent codewords. It is also required that the tracing is *error-free* and a traitor is always correctly identified. There are many constructions [SW98d, SSW01b, TM05] of c-IPP codes.

Binary c-IPP codes (with more than two codewords) do not exist, thus in any c-IPP code, the alphabet size $q \geq 3$. However, if we relax the condition on error-free tracing then we can obtain

binary codes which are called collusion secure codes. Therefore, in collusion secure codes, there is an error parameter that specifies the probability that the tracing algorithm fails to output the correct parent codeword. As mentioned, the most efficient codes are Tardos's collusion secure codes.

2.2.2 Tardos' construction

We now briefly describe the generation of a Tardos collusion secure code as proposed in [Tar03]. We additionally describe the associated tracing procedure.

Code generation. In order to generate a code for N users that resists c-collusions, set the length $\ell = 100c^2 \log(\frac{N}{\epsilon})$ where ϵ is the false-positive error probability (*i.e.* the probability that an innocent user is accused) of the tracing algorithm and randomly draw a sequence of probabilities p_i as follows:

$$p_i = \sin^2(r_i), \qquad i \in [\![1, \ell]\!]$$

where r_i is randomly drawn from $[t, \pi/2 - t]$ and $0 < t < \pi/4$ is chosen so that $300\, c \sin^2 t = 1$.

Each binary codeword w of the code is then constructed by setting its i-th digit to be either '1' or '0' according to probability p_i, that is: $\Pr[w_i = 1] = p_i$.

Tracing procedure. The authority traces a subset of traitors from a collusion (of at most c traitors) that has produced a binary word v by computing an accusation sum Z_w for each possible codeword w via:

$$Z_w = \sum_{i=1}^{\ell} v_i \cdot \left(\bar{w_i} \sqrt{\frac{1 - p_i}{p_i}} + (\bar{w_i} - 1) \sqrt{\frac{p_i}{1 - p_i}} \right) \ ,$$

where $\bar{w_i}$ is the bit w_i viewed as an integer. Then, users corresponding to codewords w such that $Z_w > 20\, c \log(\frac{N}{\epsilon})$ are declared traitors. Tardos proves that the probability of false-negative alarms (*i.e.* the probability that no traitor is found) is then $\epsilon^{c/4}$.

2.2.3 Code-based traitor tracing

At a high level, the idea is to first define a q-user sub-scheme which is resilient against a single traitor, and then "concatenate" v instantiations of this sub-scheme according to the q-ary IPP code $\mathcal{C}$; in particular, each user $i \in [1, n]$ is associated to a codeword $\omega^{(i)}$ in $\mathcal{C}$, and given the decryption key $\mathsf{sk}_i := (k_{1,\omega_1^{(i)}}, \ldots, k_{v,\omega_v^{(i)}})$, where $\omega_j^{(i)}$ is the j-th bit of the codeword $\omega^{(i)}$, and $k_{j,0}, \ldots, k_{j,q-1}$ are the keys for the j-th instantiation of the basic 2-user sub-scheme. The session key K is decomposed into random sub-keys as $K = K_1 \oplus K_2 \cdots \oplus K_l$ and then each K_i is encrypted with each of the $k_{i,j}$ to form a sub-ciphertext $c_{i,j}$. The whole ciphertext contains all sub-ciphertexts and the decryption is realised in a natural way: each user i decrypts sub-ciphertext $c_{j,\omega_j^{(i)}}$ with its secret key $k_{j,\omega_j^{(i)}}$ to get K_j for any $j = 1, \ldots l$ and finally gets K. Here is an example of a traitor tracing with 3-ary IPP code.

Key assignment :

Table 0	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	...	$k_{0,\ell}$
Table 1	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	...	$k_{1,\ell}$
Table 2	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$	$k_{2,4}$	$k_{2,5}$	...	$k_{2,\ell}$

Codeword i	1	2	0	2	0	...	1
user i	$k_{1,1}$	$k_{2,2}$	$k_{0,3}$	$k_{2,4}$	$k_{0,5}$	...	$k_{1,\ell}$

Encryption :

Session Key	$K_1\oplus$	$K_2\oplus$	$K_3\oplus$	$K_4\oplus$	$K_5\oplus$	...	$\oplus K_\ell$	$= K$
Ciphertext	$c_{0,1}$	$c_{0,2}$	$c_{0,3}$	$c_{0,4}$	$c_{0,5}$	...	$c_{0,\ell}$	
	$c_{1,1}$	$c_{1,2}$	$c_{1,3}$	$c_{1,4}$	$c_{1,5}$	...	$c_{1,\ell}$	
	$c_{2,1}$	$c_{2,2}$	$c_{2,3}$	$c_{2,4}$	$c_{2,5}$	...	$c_{2,\ell}$	

Construction of traitor tracing with robust fingerprinting codes [BP08]. Independently from Boneh-Naor [BN08b], we consider an efficient way to construct a traitor tracing from robust fingerprinting codes [BP08]: instead of decomposing the session key K into l parts, we simply decompose it into u parts, for some u much smaller than l. This helps us to reduce the ciphertext size from $O(lq)$ to $O(uq)$. However, under this encryption, if the adversary erases some position in his codeword then he still can decrypt a large part of the ciphertexts and with fingerprinting codes, one cannot trace back the traitors. This requires us to use robust fingerprinting codes which exactly deals with adversaries who can erase some parts of their codewords. This requires a stronger definition of a feasible set: $\mathrm{FS}^\star(w_1,\ldots,w_t) = \{w \in \{0,1\}^n \,|\, \forall i \in [n] \;\; : (w[i] = \star) \vee (\exists j \in [t] : w[i] = w_j[i])\}$. Robust fingerprinting codes are constructed by Safavi-Naini and Wang [SNW03b] and Sirvent [Sir07a]. Nuida [Nui09] gives the most efficient construction to date.

2.3 Black-Box Trace & Revoke Codes [NPP13]

NNL schemes, though described as trace&revoke schemes, work better for revocation than for tracing traitors. In fact, the tracing works well if we suppose that the decoder is naive, i.e. it decrypts (with some non-negligible probability) all the ciphertexts as it can, without any strategy. For smarter decoders, the scheme may not be able to identify a traitor but achieve a medium goal of making the pirate box useless by finding a "pattern" that does not allow decryption using the pirate box but still allows broadcasting to the legitimate users.

In practice, we certainly cannot assume that a decoder will accept to decrypt any signal because the pirate might be able to distinguish a normal ciphertext from an abnormal ciphertext which is probably only used in a tracing procedure. The pirate surely prefers an imperfect decoder that decrypts "almost" all ciphertexts and is untraceable rather than a perfect but traceable decoder. For example, considering the Complete Subtree scheme, let us first notice that the complete subtree scheme can be casted in terms of a binary code as follows: there is a codeword for each leaf of the N-leaf full binary tree $\mathcal{T}$. The code length is $\ell = 2N - 1$, each position (i.e. coordinate) of the code corresponds to a node of the tree. For each codeword w, 1 is put in a position if and only if the corresponding node is on the path from w to the root. We will refer this code as the *CS-code* (see figure 2.3). The pirate decoder can employ the following strategy: it does not decrypt any weight-1 signal where the 1 is in the position of a traitor leaf node. Under this strategy, the CS-code is not traceable, unless with error probability of at least 1/2 because no tracing algorithm can distinguish a traitor (a leaf node) from its sibling in the full binary tree. Note that the sibling may very well be a non-traitor. The CS-Code cannot deal with this type of pirate strategy because the code has a rigid structure where each position plays a specific role and corresponds to a subset of users of different sizes.

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
S_1	1	1	1	1	1	1	1	1
S_2	1	1	1	1	0	0	0	0
S_3	0	0	0	0	1	1	1	1
S_4	1	1	0	0	0	0	0	0
S_5	0	0	1	1	0	0	0	0
S_6	0	0	0	0	1	1	0	0
S_7	0	0	0	0	0	0	1	1
S_8	1	0	0	0	0	0	1	0
S_9	0	1	0	0	0	0	0	0
S_{10}	0	0	1	0	0	0	0	0
S_{11}	0	0	0	1	0	0	0	0
S_{12}	0	0	0	0	1	0	0	0
S_{13}	0	0	0	0	0	1	0	0
S_{14}	0	0	0	0	0	0	1	0
S_{15}	0	0	0	0	0	0	0	1

Figure 2.4: Complete Subtree Scheme (for 8 users) can be viewed as a binary code with high structure (each subtree corresponds to a line which defines a binary codeword)

Our objective is to propose probabilistic constructions of codes where all the code positions have the same role and thus the strategy of refusing to decrypt some positions has no significant impact on the tracing algorithm. Our probabilistic constructions, described in the next sections, can deal with the above pirate strategy against CS-Code for that reason.

However, the pirate strategy can certainly be smarter than rejecting some position(s) of the code. For example, for a probabilistic code where the codewords are chosen independently from the same distribution and all positions play the same role, a non-trivial pirate can estimate the (Hamming) weight of signals used in broadcast encryption and refuse to decrypt a ciphertext that corresponds to a signal containing too few or too many 1s. This pirate strategy, called the "weight-limited pirate", is formalised as follows:

Definition 2.3.1 [Weight-Limited Decoder] A **Weight-Limited Decoder** is a decoder that only decrypts signals c with Hamming weight in an interval $[a, b]$.

It seems to us that for probabilistic constructions of codes where all the code positions have the same role, it is hard for a pirate to employ any other strategy than the Weight-Limited Decoder because the codewords look random and the most important information seems to be the Hamming weight. We therefore focus on Weight-Limited Decoder and construct a scheme in which in the tracing procedure we randomly sample tracing signals that have the same weight as in the ciphertext.

2.3.1 The construction

We first indicate a simple connection between traceability of codes with the so-called *disjunct matrices*, a classical combinatorial object originally used in group testing, which has "built-in" tracing capability. Roughly speaking, a r-disjunct matrix is a binary matrix satisfying the following property: given the (boolean) union of at most r unknown columns of the matrix, we can identify *all* the unknown columns. This concept is used to design non-adaptive group tests in the following sense: there is a set of at most r *positive items* in a population of N items

and the rest of the items are *negative*; we must identify the positives using as few non-adaptive "tests" as possible; each test is a subset of items; a test returns positive iff at least one positive item is contained in the test. In the original group testing application [Dor43], each item is a blood sample, and a test is a pool of blood samples which indicates if any sample in the pool is positive for syphilis. That application explains the "positive" and "negative" terms.

The problem with disjunct matrices is that they have no "built-in" efficient revocation capability. Indeed, disjunct matrices or equivalently *cover-free families* [EFF85] have been used for traitor tracing in [TSN06]. However, by following the tracing framework of [BF99b] it cannot be used for revocation. We deal with this problem by considering a combinatorial object called (r, s)-*disjunct matrices* which retains the tracing-capability of disjunct matrices while also supports revocation. Intuitively, a matrix $\mathbf{M}$ is said to be (r, s)-disjunct if for an arbitrary set R of up to r columns of $\mathbf{M}$, there is a set $I \subseteq [\ell]$ of at most s rows which eliminates R, or equivalently, covers $N - R$. It is not hard to see that (r, s)-disjunct matrices, while attain efficient revocation capability, also retain the traceability of disjunct matrices. It turns out that (r, s)-disjunct matrices are equivalent to exclusive set systems (ESS for short), first introduced by Aiello et al. [ALO98] under the name *complement cover families* and independently latter by Kumar and Russell [KR03]. In [GSY99], the authors consider traitor tracing for exclusive set systems but only in white-box model where the pirate key is supposed to be known. This somewhat looses the main advantage of supporting black-box tracing in code-based systems.

Our main contribution is to present good (r, s)-disjunct matrices which allow for black-box tracing and efficient revocation.

In fact, we generate a matrix $\mathbf{M} \in \mathcal{M}(N, b, n)$ with $\ell = bn$ rows and N columns, and independently generate columns of $\mathbf{M}$ where each column of $\mathbf{M}$, viewed as a subset of $[\ell]$, is chosen by picking uniformly (with probability $1/b$) and exactly one bin from each part. In particular, each column of $\mathbf{M}$ has exactly n elements.

We can think of each column as a "ball" and each part is a collection of b bins. The distribution $\mathcal{M}(N, b, n)$ is defined by throwing N balls to b bins belonging to a part, and by repeating that experiment n times, one for each part. This type of matrix distribution is used in constructing compressed sensing matrices. The resulting random matrix can also be thought of as the incidence matrix of concatenating a random code of length n with the identity code [NPR12]. The idea is to choose a matrix $\mathbf{M}$ at random from $\mathcal{M}(N, b, n)$ with suitably chosen parameters n and b, and show that $\mathbf{M}$ is (r, s)-disjunct with high probability.

Tracing smart pirate with shadow group testing If we consider naïve pirates who decrypt any ciphertext as they can then we are done. Indeed, the properties of (r, s)-disjunct matrices directly allow us to do both revocation (as for r-disjunct matrices) and tracing (by sending special weight-1 tracing signals to the pirate decoder and then we can cover the union vector of the traitors' codewords, which is sufficient for tracing). However, it is a challenging problem to deal with a smart pirate, namely the Weight-Limited Decoder as discussed above. We first remark that if the tracing algorithm works for a weight-limited decoder with interval $[a, a]$, then it *a fortiori* works for a weight-limited decoder with interval $[u, v]$, for any $u \leq a \leq v$. Therefore, the most interesting case is a singleton interval. The main problem for tracing procedure is that we can now ask the decoder random queries of the same weight, say a, with normal ciphertexts. The point is that, instead of identifying a traitor, we can only identify a vector $w \in \{0, 1\}^\ell$ that is contained in the union of all traitors' codewords and contains at least one traitor's codeword. Identifying $w = (w_1, \cdots, w_\ell)$ is then equivalent to identifying all the coordinates i of w for which $w_i = 1$. Thus, there is a subset $U \subseteq [\ell]$ of at most D unknown coordinates that we want to identify. We need to query the pirate decoder with weight-a signals c to identify U. Each query c is the characteristic vector of a subset of size a of $[\ell]$. So we think of each query as an a-subset

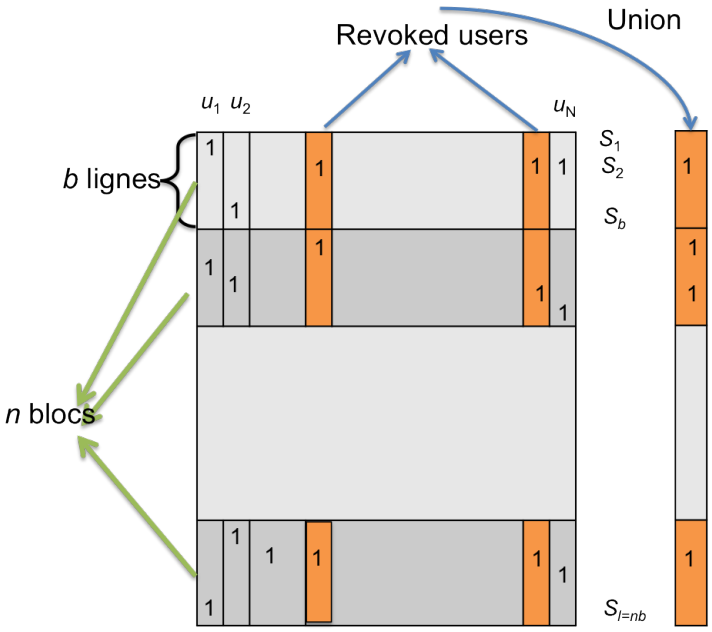


Figure 2.5: Construction of (r, s)-disjunct matrices

A of $[\ell]$. The decoder is able to decrypt query A if and only if there is at least one traitor whose codeword intersects A. In other words, each query A is a *group test* for the "positives" U in the population $[\ell]$. The queries then form a matrix which, interestingly, is also a disjunct matrix. We thus have a group testing problem "inside" another group testing problem. We refer to the "inner" group tests as the *shadow tests*, because they are not used to directly identify the traitors: they are rather used to identify the shadow U of the traitors.

The resulting code yields a trace&revoke scheme with private key size and ciphertext length $O((t+r)\log(N/(t+r)))$ for N users, at most r revoked users and at most t traitors. The constants hidden in the big-O are small (≤ 8). This randomized construction yields a key assignment scheme where users independently pick their keys from the same distribution and all keys have the same role. Thus, unlike the complete-subtree method which leads to a highly asymmetric key assignment making it suitable for a more relaxed tracing model (called semi-BB in the comparison table 2.3.2) but unable to dealing with tracing a traitor for smart pirate decoders, our code has better "built-in" support for traceability against non-trivial pirate strategies.

2.3.2 Summary

Table 2.3.2 summarizes some known results on combinatorial constructions.

2.4 Trace&Revoke from linear codes

Linear codes can be used for traitor tracing. It is shown [SSW00] that any $(n, k, \Delta)_q$-code is an q-ary c-IPP code with $\Delta > n(1 - 1/c^2)$. However, it is not known how we can achieve both tracing and revoking ability with linear codes. We propose a method for this goal by generalising the previous framework of constructing traitor tracing from IPP codes. We then instantiate the generic scheme with concrete linear codes, namely Reed-Solomon code and Porat-Rothschild codes.

In fact, we generalise the code based traitor tracing in section 2.2.3 in two aspects:

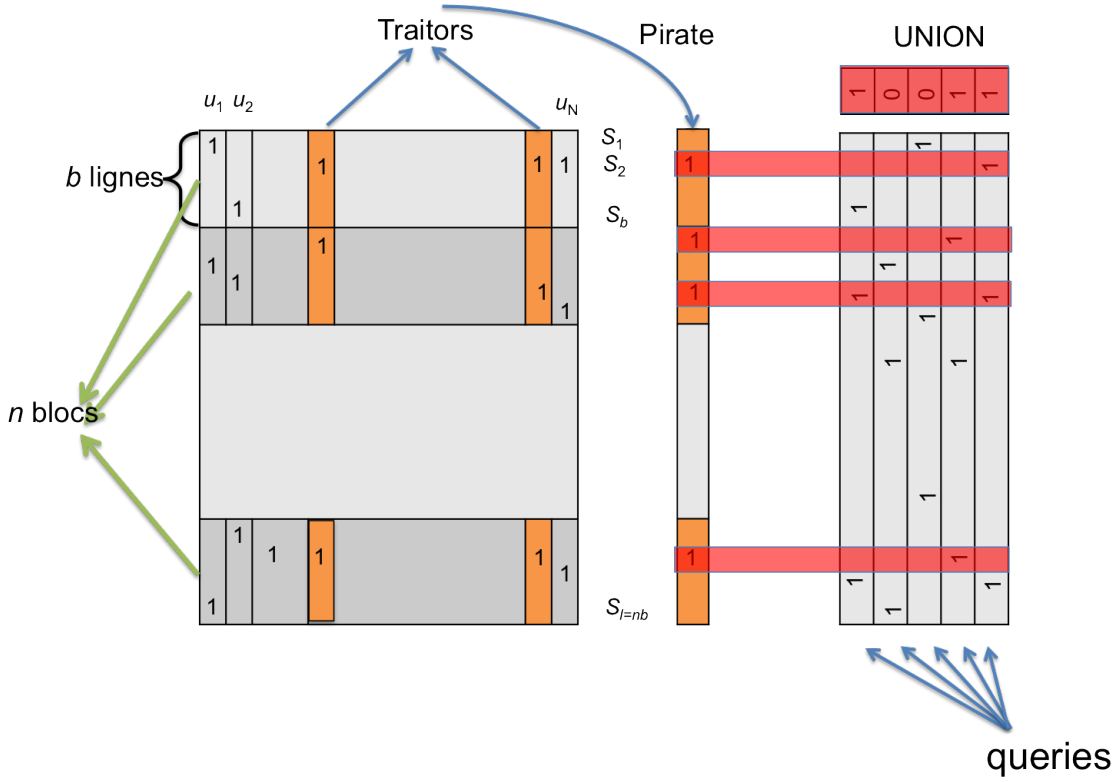


Figure 2.6: The shadow technique

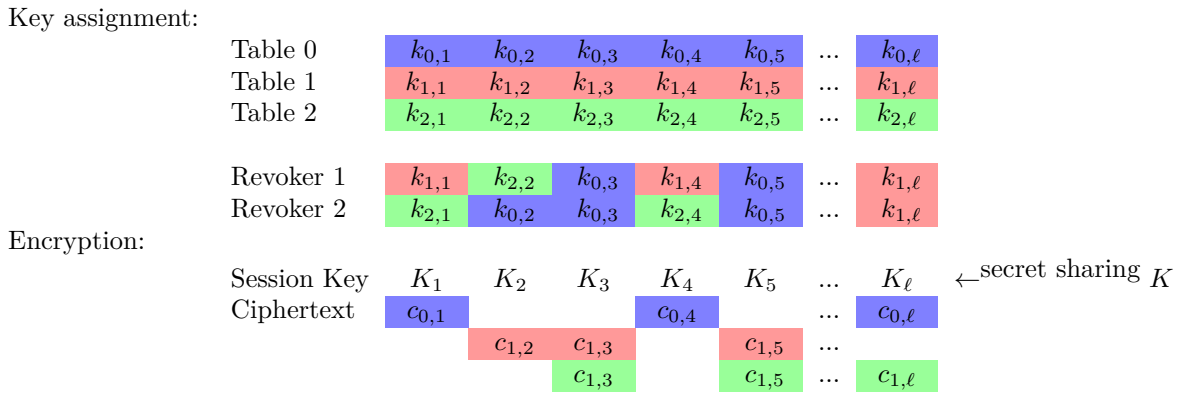
Paper	ℓ	s	k (keys per user)	Constraints	Traceability	Eff. dec?
[KR03]	$O(s^3(Ns)^{r/s}\log N)$	$s \geq r$	$O(s^3(Ns)^{r/s}\log N)$	N large	???	No
[ALO98]	$\frac{N2^c}{(c-1)}$	$s = O(r\log_c(N/r))$	$2n$	any $c \geq 2$	???	No
[GSY99]	$\left(\frac{r\log N}{\log r}\right)^2$	$s = \left(\frac{r\log N}{\log r}\right)^2$	$\left(\frac{r\log N}{\log r}\right)^2$	Com	White-Box	No
[KRS99]	$\frac{r^3\log N}{\log r}$	$s = \frac{r^3\log N}{\log r}$	$\frac{r^3\log N}{\log r}$		???	Yes
[NNL01]	$2N$	$s = O(r\log(N/r))$	$\log N$		semi-BB	Yes
[NNL01]	$N\log N$	$s = 2r$	$\log^2 N$	Com	semi-BB	Yes
[HS02]		$s = O(r)$	$\log^{1+\epsilon} N$	Com	semi-BB	Yes
[GST04]		$s = O(r)$	$O(\log N)$	Com	semiBB	Yes
[JHC+05]		$s = \frac{r}{p+1} + \frac{N-r}{c}$	$O(c^{p+1})$	for any p, c, Com	No	Yes
[GRW06]	$\mathcal{P}oly(r,s)\binom{N}{r}^{1/s}$	$s = r\log(N/r)$	$\mathcal{P}oly(r,\log N)$		???	No
[GRW06]	$O\left(rs\binom{N}{r}^{1/s}\right)$	$2r$	$\mathcal{P}oly(r)N^{1/2}$		???	Yes
Ours	$8r^2\log(N/r)$	$s = 4r\log(N)$	$k = 2r\log(N/r)$		Black-Box	Yes

Table 2.1: Known results on combinatorial constructions. "Com": the security is based on computational assumption. "???": not considered in the paper. "semi-BB": tracing can either trace a traitor or output a partition that the pirate decoder cannot decrypt.

- In each position, we do not encrypt the sub-key K_j with all the keys $k_{i,j}$. Instead, we encrypt K_j with a subset of the keys $k_{i,j}$. By this way, we can revoke users by not encrypting K_j with the key $k_{i,j}$ if the user i has been revoked.

- By "revoking" the key $k_{i,j}$ of a revoked user in a sub q-user scheme, legitimate users are also affected because the key $k_{i,j}$ is shared among many users and some of them might also be revoked. The decomposition of the session key K as $K = K_1 \oplus K_2 \cdots \oplus K_l$ does not work anymore. However, we can show that legitimate users still get sufficiently large number of non-revoked sub-keys and therefore, if we decompose the session key K with an appropriate secret sharing then non-revoked users can still decrypt.

We illustrate our modification of the code based traitor tracing in section 2.2.3 to achieve a code based trace&revoke as follows:



The analysis in this section is based on our on-going work and we would give some details of our solution.

Let C be a $(n, k, \Delta)_q$-code, over an alphabet Σ of q symbols. A *mixture* $S = (S_1, \ldots, S_n)$ over Σ^n is a sequence of n subsets of Σ, i.e. $S_i \subseteq \Sigma$. Given a vector $\mathbf{w} = (w_1, \ldots, w_n) \in \Sigma^n$, the *agreement* between $\mathbf{w}$ and a mixture S is defined to be the number of positions $i \in [n]$ for which $w_i \in S_i$:

$$\text{AGR}(\mathbf{w}, S) = \sum_{i=1}^{n} \mathbf{1}_{w_i \in S_i}.$$

We consider the following broadcast system. For each $i \in [n]$ and each symbol $a \in \Sigma$, there is a key $k_{(i,a)}$. There are $N = q^k$ users in the system. Each user corresponds to a codeword $\mathbf{c} \in C$, where the user is given n keys $k_{(i,c_i)}$.

Let s be a secret message to be broadcasted. Let $s_1, \ldots, s_n$ be the shares of a $(\rho n, n)$-secret sharing scheme. At least ρn shares are needed to recover the secret s. Let S be a mixture over Σ^n. We *broadcast using mixture* S by encrypting each s_i with all the $|S_i|$ keys $k_{i,a}$ for $a \in S_i$. Thus, if anyone is able to decrypt the message, that person has to possess at least ρn keys from separate sets S_i of the mixture S.

Revocation To revoke r users $R = \{\mathbf{c}_j \mid j \in [r]\}$, where $\mathbf{c}_j \in C$ are codewords, we do the following. Define

$$R[i] = \cup_{j \in [r]} \{\mathbf{c}_j[i]\}.$$

And, we broadcast using the following mixture:

$$S = (S_1, \ldots, S_n) = (\Sigma - R[1], \ldots, \Sigma - R[n]).$$

Now, we need to make sure that for every user $\mathbf{w} \notin R$, $\mathbf{w}$ is able to decrypt the message, which means we want

$$\text{AGR}(\mathbf{w}, S) \geq \rho n.$$

What property must the code satisfy for this to happen? Note that $\mathbf{w}$ shares at most $n - \Delta$ positions with any codeword in R. Hence, there are at least $n - r(n - \Delta)$ positions i for which $w_i \notin R_i$. In other words, $\text{AGR}(\mathbf{w}, S) \geq n - r(n - \Delta)$. Thus, a non-revoked user can decrypt if

$$n - r(n - \Delta) \geq \rho n$$

which is equivalent to

$$\Delta \geq n \cdot \left(1 - \frac{1 - \rho}{r}\right).$$

Tracing Suppose we are broadcasting using some mixture S. Let T be the set of traitors. Now, using the blackbox method described in [CFN94b], we obtain a set F of at least ρn keys k_{i,a_i} for which $a_i \in S_i$ for each such key.

Naturally, we will view F as a mixture $(F_1, \ldots, F_n)$ which has ρn singletons ($F_i = \{a_i\}$ if $k_{i,a_i} \in F$) corresponding to the keys and the rest are empty sets. Let $\mathbf{w}$ be the codeword which agrees with F in the most number of positions. We want $\mathbf{w} \in T$.

Let t be the (maximum) number of traitors. We know that there must be one traitor $\mathbf{c}$ who contributed at least $\rho n/t$ keys to F. Thus, it is sufficient to ensure that, for every user $\mathbf{u} \notin T$, $\text{AGR}(\mathbf{u}, F) < \rho n/t$. But we know that $\text{AGR}(\mathbf{u}, \mathbf{c}) \leq n - \Delta$ for any traitor $\mathbf{c} \in T$. Hence,

$$\text{AGR}(\mathbf{u}, F) \leq \sum_{\mathbf{c} \in T} \text{AGR}(\mathbf{u}, \mathbf{c}) \leq t(n - \Delta).$$

Thus, it is sufficient that

$$t(n - \Delta) < \rho n/t,$$

which is equivalent to

$$\Delta > n(1 - \rho/t^2).$$

Trace&revoke So, for the system to be able to trace&revoke, we need a $(n, k, \Delta)_q$-code in which

$$\Delta > n \cdot \max\left\{\left(1 - \frac{1 - \rho}{r}\right), (1 - \rho/t^2)\right\} = n \cdot \left(1 - \min\{\frac{1 - \rho}{r}, \rho/t^2\}\right).$$

The number of keys per user is n. The broadcast key size is at most nq We just proved the following theorem.

Theorem 2.4.1 *Let $\rho \in (0, 1)$ be an arbitrary real number. Let $r, t \leq N$ be postive integers. Suppose there exists a $(n, k, \delta)_q$-code for which*

$$\delta > 1 - \min\left\{\frac{1 - \rho}{r}, \frac{\rho}{t^2}\right\},$$

and $q^k \geq N$. Then, there exists a trace&revoke system which can support up to r revoked users and t traitors in which the user key size is n and the broadcast key size is $s \leq nq$.

Example 2.4.2 (Using a code meeting GV-bound) *Let's pick $\rho = 1/2$.*
Set $d = \max\{2r, 2t^2\}$. As we have seen above from the application of Porat-Rothschild derandomization of the Gilbert-Varshamov bound, we can explicitly construct a code with relative distance $\delta = \Delta/n > 1 - 1/d$ where $q = \Theta(d)$, $n = O(d \log N)$.

The number of keys per user is

$$n = O\left(\max\{2r, 2t^2\}\log N\right)$$

which is probably not too bad. The broadcast key size

$$s \leq qn = O\left((\max\{2r, 2t^2\})^2 \log N\right)$$

is bad.

Example 2.4.3 (Using RS-code) *Again pick $\rho = 1/2$ and set $d = \max\{2r, 2t^2\}$. The RS-code has $\delta = \frac{n-k+1}{n} = 1 - \frac{k}{n} + \frac{1}{n}$. In this case, if we choose $n = kd$ then $\rho > 1 - 1/d$. Hence, to use RS-code, we need to pick $q \geq n = kd$ such that $q^k \geq N$ or, equivalently, $n \log q \geq d \log N$.*

For example, we can pick $q = n \approx \frac{2d\log N}{\log(d\log N)}$ and $k \approx \frac{\log N}{\log q}$. In this case, the number of keys per user is

$$n = O\left(\frac{2d\log N}{\log(d\log N)}\right)$$

and the broadcast key size is

$$s = O(n^2).$$

Or, if we only want to reduce the number of keys per user we can do something extravagant such as picking $q = 2^n$. In this case, we have

$$n = \sqrt{d\log N}$$

and

$$s = n2^n = \sqrt{d\log N}N^d.$$

Final remark. The parameter ρ characterises in fact the trade-off between the capacity of tracing and the capacity of revoking. Indeed:

- When $\rho = 1$, the above code is the Tracing traitor system in [CFN94b]. The (n, n)-secret sharing could be very efficiently implemented (the xor of n parts).
- When $1 < \rho n < n$, as shown above, we could combine the functionalities of an ESS system (for revocation) and a black-box tracing against any pirate strategy. However, we should note that the $(\rho n, n)$-secret sharing makes the scheme less efficient than in the cases where $\rho n = 1$ or $\rho n = n$.
- When $\rho = \frac{1}{n}$, the above code is an ESS system. The $(1, n)$-secret sharing becomes trivial. Each singleton $S_i = \{a_i\}$ defines a subset covering all users who have the key k_{i,a_i}. This corresponds to the case we consider in the previous section and the results lead to a $(N, 4r^2\log^2 N, r, 4r^2\log^2 N)-$disjunct matrix. As shown in the previous section, the code can be used for revocation in a very efficient way and the shadow technique helps us to trace weight-limited decoders. Even though the tracing complexity is expensive, the resulting system enjoys the nice properties of an ESS system with constant decryption time complexity at the receiver.

Chapter 3

Algebraic Approach

While most combinatoric schemes deal with bounded collusions (the number of revoked users and the number of traitors have been assumed to be below some threshold), algebraic schemes can deal with both bounded collusions and full collusions. However, the situation is quite different when it comes to broadcast encryption and traitor tracing:

Broadcast encryption: Boneh, Gentry, and Waters [BGW05] are first to propose a fully collusion-resistant public key broadcast encryption in which the ciphertext size is constant. They proposed two schemes, respectively CPA and CCA secure, both in the selective model of security. Dynamic fully collusion-resistant broadcast encryption is proposed in [DPP07] where the authors designed CPA secure schemes that were only partially adaptive secure. In brief, full collusion broadcast encryption can be made quite practical.

Traitor Tracing: The first non-trivial fully collusion resistant scheme is proposed by Boneh et al. [BSW06b]. However, its ciphertext size is still large ($\Omega(\sqrt{N})$, where N is the total number of users) and it relies on pairing groups of composite order. Very recently, Boneh and Zhandry [BZ14] propose a fully collusion resistant scheme with poly-log size parameters. This scheme relies on indistinguishability obfuscation [GGH$^+$13c] of which security foundation remains to be studied and practicality remains to be investigated. Unsurprisingly, the most efficient schemes are in the bounded collusion model where the number of malicious users is limited. The most efficient algebraic traitor tracing schemes remain the Boneh-Franklin scheme, Naor-Pinkas scheme and their variants.

In this chapter, we first summarise in Sections 3.1 and 3.2 the main techniques for constructing algebraic schemes, in bounded collusion model and full collusion model. We then present our contributions:

- We first consider in Section 3.3 some extensions of the BGW scheme in extended security model and in multi-channel setting.
- We then introduce in Section 3.4 the first lattice based traitor tracing in the bounded model in which the security is based on a new variant of the Learning With Errors problem (LWE) problem, called k-LWE.
- We finally propose in Section 3.5 optimal ciphertext rate traitor tracing schemes that extend the Kiayias-Yung strategy for integrating combinatorial methods with algebraic methods.

3.1 From ElGamal encryption to multi-receiver encryption, trator tracing and revoke schemes

Desmedt and Kurosawa are the first to propose a method to transform Elgamal encryption into a traitor tracing but their scheme was latter pointed out to be insecure. Boneh and Franklin then propose another transformation that is based on a representation problem and linear space tracing; Naor and Pinkas propose a revoke method that is based on polynomial interpolation. The main idea in the construction of these schemes is to modify the form of the public key in ElGamal encryption in such a way that it corresponds to many different secret keys. At this stage, it suffices to distribute each secret key to a user and we will get a multi-receiver encryption. The traceability is much more difficult to achieve though. We will briefly describe the main ideas in Boneh-Frabklin and Naor-Pinkas traitor tracing schemes. Let us first recall the ElGamal encryption.

Setup: On input the security parameter λ, return a λ-bit prime q, a group G of order q, and a randomly chosen generator $g \in G$.

Key setup Generate $\alpha \leftarrow \mathbb{Z}_q$. Set $\mathsf{sk} \leftarrow \alpha$ and $\mathsf{pk} \leftarrow y = g^\alpha$.

Encryption: Given a message $m \in G$, randomly choose $r \leftarrow \mathbb{Z}_q$ and output the ciphertext $(g^r, y^r m)$

Decryption: Given a ciphertext (c_1, c_2), return c_2/c_1^α.

3.1.1 Boneh-Franklin method for traitor tracing [BF99b]

In addition to the element $y = g^\alpha$ as in ElGamal scheme, one also chooses a vector $(h_1, \ldots, h_{2t})$ of $2t$ (where t is the bound on the number of traitors) random elements in G, say $h_i = g^{r_i}$, for $r_i \leftarrow \mathbb{Z}_q$. The public key is then set to be $\mathsf{pk} \leftarrow (y, h_1, \ldots, h_{2t})$. This allows the center to represent the same y in different ways in the basis $(h_1, \ldots, h_{2t})$. Indeed, by knowing the discrete logarithm of y and of h_i to the base g, it is easy for the center to generate a random representation $(\alpha_1, \ldots, \alpha_{2t})$ of y in the basis $(h_1, \ldots, h_{2t})$ such that: $y = h_1^{\alpha_1} \ldots h_{2t}^{\alpha_{2t}}$. Each individual key is a representation $(\alpha_1, \ldots, \alpha_{2t})$ which allows the user to compute y^r from $(h_1^r, \ldots, h_{2t}^r)$. The encryption and the decryption work then in a natural manner: by adding $(h_1^r, \ldots, h_{2t}^r)$ to the ciphertext, any legitimate user (who holds a representation) can compute y^r and recover the plaintext.

We now discuss the traceability. Boneh and Franklin show that, unless breaking the discrete logarithm problem, the only way for the adversary to produce a new representation of y is to linearly combine its known representations. This leads to the idea of using linear error-correcting code for tracing. Indeed, consider any linear error-correcting code A (codewords generated by the columns of A) that can correct up to k errors and its parity check matrix H. If we associate each user to a row of H then from any linear combination of up to k corrupted rows of H, one can trace back the corrupted rows. This is derived directly from the error-correcting property: given d which is a linear combination of up to k corrupted rows of H, i.e. $d = wB$ for an unknown vector w of weight $\leq k$; the goal is to find w. We can do this by first computing any v satisfying $vB = d$ by linear algebra; we know then $v - w$ is a codeword of A and thus v is deviated from a codeword with at most k errors; the correction of the error of v will directly provide us with the "error" w.

In fact, Boneh and Franklin use for A as a Reed-Solomon code corresponding to a Vandermonde matrix and the white-box tracing follows the above intuition. A more challenging point is the black-box tracing where one does not know any pirate key d. The black-box tracing, which

is in fact quite expensive, relies on the black-box confirmation: given a superset of the traitors, it is guaranteed to find at least one traitor and no innocent suspect is incriminated.

3.1.2 Naor-Pinkas method for revocation [NP00]

At a high level, the main idea is to use a *t-out-of-N* secret sharing and the scheme can revoke up to $t-1$ users. For simplicity, we can suppose that the number of revoked users r is equal to $t-1$ (if the effective number of revoked users is less than $t-1$ then we can add "dummy" users to the revoked list). The system works as follows: a secret is divided into N shares and each user who joins the system receives a share; the ciphertext contains $t-1$ shares that cover all the revoked users; each non-revoked user adds its share to have t shares that can decrypt the ciphertext while the revoked users only get $t-1$ shares in total and cannot decrypt even if they all collude. In order to implement this idea, Naor and Pinkas use secret sharing in the exponent and randomise the ciphertext.

More formally, the authority chooses a global polynomial P of degree $t-1$ in the setup, then chooses and publishes a random element x_i for each user i. The secret key for user i is $P(x_i)$ and all the values $g^{P(x_i)}$ and $g^{P(0)}$ are published. To revoke a set of users $(1, 2, ..., t-1)$, a broadcaster chooses a random element r then sets the session key $K = g^{rP(0)}$. The ciphertext is composed of $t-1$ elements $g^{rP(x_j)}, j \in 1, \ldots, t-1$. Each non-revoked user has in possession t shares and can perform a polynomial interpolation in the exponent to recover the session key, while the revoked users have at most $t-1$ shares and get no information from the ciphertext. It is worth noticing that the Naor-Pinkas method can be combined with the Boneh-Franklin method to achieve a trace&revoke scheme.

3.2 Dealing with Full Collusion

We recall that, in full collusion of broadcast or traitor tracing schemes, the maximum number of corrupted users is unbounded. As we mentioned at the beginning of the chapter, we will see that full collusion broadcast encryption schemes are quite practical while full collusion traitor tracing schemes remain inefficient.

3.2.1 Broadcast encryption: BGW scheme

We can explain the idea at a high level as follows: suppose that there are $2N+1$ slots on a line and there is only a hole at the position $N+1$. Each user is attributed a slot and a user at the slot $N-i$ is capable to push the hole exactly i positions to the left.

Now, for the encryption, the center simply puts the balls into the positions of the non-revoked users. In the decryption phase, each user i pushes to move the hole to his position: if there is a ball in the hole then the decryption succeeds (see Figure 3.1). The use of pairings allows the center to limit the capacity of the users: each user i can only move the hole by exactly i positions.

How is this implemented? We can imagine that each slot i is attributed an element $g_i = g^{\alpha^i}$ and the magic element g^{N+1} corresponds to the hole at position $N+1$. We can rewrite this element for any set of users $S \subseteq [N]$ as:

$$g_{N+1} = \Pi_{j \in S} g_{N+1-j+i} / \Pi_{j \in S, j \neq i} g_{N+1-j+i}$$

In order to exploit this representation, one has to work in a bilinear group: the randomised session key (with a random r) is set to be $e(g, g_{N+1})^r$. We can then rewrite it as:

$$e(g, g_{N+1})^r = e(g_i, (\Pi_{j \in S} g_{N+1-j+i})^r) / e(g^r, \Pi_{j \in S, j \neq i} g_{N+1-j+i})$$

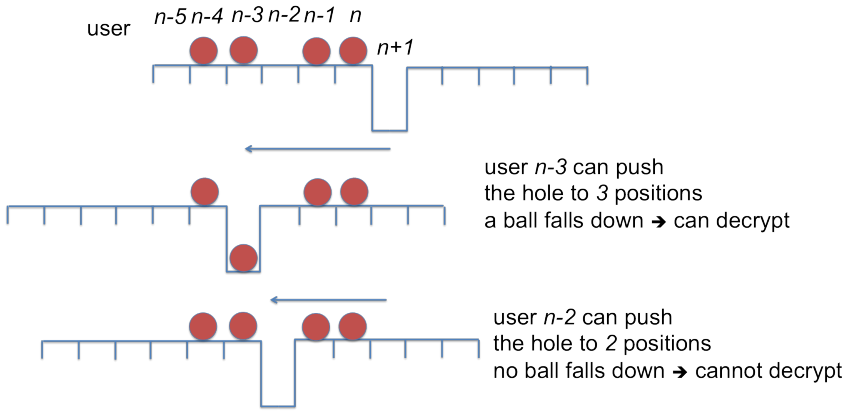


Figure 3.1: High-level view of the BGW method.

Actually, if the public key contains $g, g_1, \ldots, g_{2N}$ (so that the encryption is publicly computable) and the ciphertext contains $(g^r, (\Pi_{j\in S} g_{N+1-j+i})^r)$, then the session key can be publicly computed. The main idea is to make the session key individually computable. This can be achieved by adding an element $v = g^\gamma$ to the public key and give each user a secret key g_i^γ. The session key is then computed as:

$$e(g, g_{N+1})^r = e(g_i, (v\Pi_{j\in S} g_{N+1-j+i})^r)/e(g^r, g_i^\gamma \Pi_{j\in S, j\neq i} g_{N+1-j+i})$$

It is not hard to see that if the ciphertext contains $g^r, (v\Pi_{j\in S} g_{N+1-j+i})^r$, then only users $i \in S$ can recover the session key by adding its secret key g_i^γ to the element $\Pi_{j\in S, j\neq i} g_{N+1-j+i}$. We can imagine the way we put the balls in the positions in S is characterised by $(v\Pi_{j\in S} g_{N+1-j+i})^r$. Each user i can then move the hole by i positions by using the secret key g_i^γ to compute $g_i^\gamma \Pi_{j\in S, j\neq i} g_{N+1-j+i}$: only users in S can put a ball in the hole and recover the session key. If v is not used then each user can move the hole to any position and everyone can decrypt. However, by using the element v, each user is forced to use its secret key to move the hole and in consequence, each user can only move the hole by exactly i positions. As some of our works are based on this scheme, we give a detailed description of the scheme as follow:

$\mathsf{Setup}(\lambda)$**:** Let $\mathbb{G}$ be a bilinear group of prime order p. The algorithm first picks a random generator $g \in \mathbb{G}$ and a random scalar $\alpha \in \mathbb{Z}_p$. It computes $g_i = g^{\alpha^i} \in \mathbb{G}$ for $i = 1, 2, \ldots, n, n+2, \ldots, 2n$. Next, it picks a random scalar $\gamma \in \mathbb{Z}_p$ and sets $v = g^\gamma \in \mathbb{G}$.
The public key is $\mathsf{EK} = (g_1, \ldots, g_N, g_{N+2}, \ldots, g_{2n}, v)$, whereas the private decryption key of user $i \in \{1, \ldots, n\}$ is $d_i = v^{\alpha^i}$. These decryption keys are sent by the $\mathsf{Extract}$ algorithm.

$\mathsf{Encrypt}(S, \mathsf{EK})$**:** Pick a random scalar $r \in \mathbb{Z}_p$ and set $K = e(g_{N+1}, g)^r$, where $e(g_{N+1}, g)$ can be computed as $e(g_N, g_1)$ from EK. Next, set: $\mathsf{Hdr} = (g^r, (v \cdot \prod_{j\in S} g_{N+1-j})^r)$ and output (Hdr, K).

$\mathsf{Decrypt}(S, \mathsf{Hdr}, i, d_i, \mathsf{EK})$**:** Parse $\mathsf{Hdr} = (C_1, C_2)$, output

$$K = e(g_i, C_2)/e(d_i \cdot \prod_{j\in S, j\neq i} g_{N+1-j+i}, C_1)$$

Inversion technique. Delerablée, Paillier, and Pointcheval introduced the inversion technique which helps to construct the first dynamic revoke scheme [DPP07]. The main idea is to generate a value x_u to each user u and then, in the decryption phase, to force the user to multiply a group element by $\frac{1}{x_r - x_u}$ for each revoked user r. This means that every revoked user has to

divide by 0 during decryption, which leads to a procedure failure. The construction requires a pairing in prime-order groups, but it does not matter whether the pairing is symmetric or not. The security guarantee is actually stronger than the static security level: The adversary is allowed to corrupt users immediately before they join. The master secret key and the user secret keys consist of a scalar value and one group element from each of the base groups $\mathbb{G}_1, \mathbb{G}_2$, the encryption key consists of one group element from each of the base groups and one from the target group $\mathbb{G}_T$. For each user, a scalar, an element from the base group $\mathbb{G}_2$ and an element from the target group are added to the encryption key. The ciphertext consists of one element from each of the base groups plus a scalar and an element from $\mathbb{G}_2$ for each revoked user.

Another modification of the public-key scheme makes it non-dynamic, but allow to achieve constant-size ciphertexts at the expense of linear size decryption keys.

3.2.2 Traitor Tracing: BSW scheme

We recall that a trivial way to construct a multi-receiver scheme is to encrypt independently to each user: the ciphertext contains many components, each of them is an individual encryption of the session key for a user. We can then apply the linear tracing technique which consists of replacing step by step each component by a random element: when we modify a component, it only affects one user and therefore, if the pirate decoder decrypts differently from one step to the next, we can detect a traitor. Now, in order to achieve a sub-linear size ciphertext, we should deal with "dependent" component problems because a component in the ciphertext must be related to many users. The task of the tracer becomes quite challenging because each time the ciphertext is modified, many users are affected and it is not easy to detect the traitor. Boneh, Sahai and Waters [BSW06b] introduce a method to deal with this problem in which they took benefit from the bilinearlity of pairings to arrange the users in a matrix. In such a way, each user is identified by a couple of parameters: row position and column position. The main idea that allows to use the linear tracing technique is to produce a probe ciphertext for any position (i, j) in such a way that: any user at position (x, y) will be able to decrypt the message if and only if $(x > i)$ or $(x = i, y \geq j)$. This requires to randomise row components and column components in a smart way. In fact, Boneh, Sahai and Waters need to use pairings on composite order groups, say $e : G \times G \to G_T$ such that G is of composite order pq and if g_p is an element from the order p subgroup (which is called G_p) and g_q is an element from the order q subgroup (which is called G_q), then $e(g_p, g_q) = 1$. Each ciphertext contains $m = \sqrt{N}$ "well-formed" row components in G and m "well-formed" column components in G. The user (x, y) can successfully decrypt if the row component and the column component are of type (well-formed,well-formed). Now, the tracer produces a probe ciphertext in which:

- Column ciphertext components are well formed in both G_p and G_q subgroups for columns greater than or equal to j ; well formed in the G_q subgroup but random in the G_p subgroup for a column that is less than j.
- Row ciphertext components are completely random for rows less than i ; well formed elements in the G_q subgroup for rows greater than i; and are well formed in both subgroups for row i.

We can see that, the ciphertext structure will lead to restrictions on the decryption: it can be successful for a user (x, y) if and only if $x > i$ or $x = i, y \geq j$. Indeed:

- If $x > i$ then, because the row ciphertext components are well formed elements in the G_q subgroup, even if the column ciphertext components are randomized with an element in G_p, the randomized part is cancelled out and the (row component, column component) for (x, y) looks like (well-formed, well-formed).

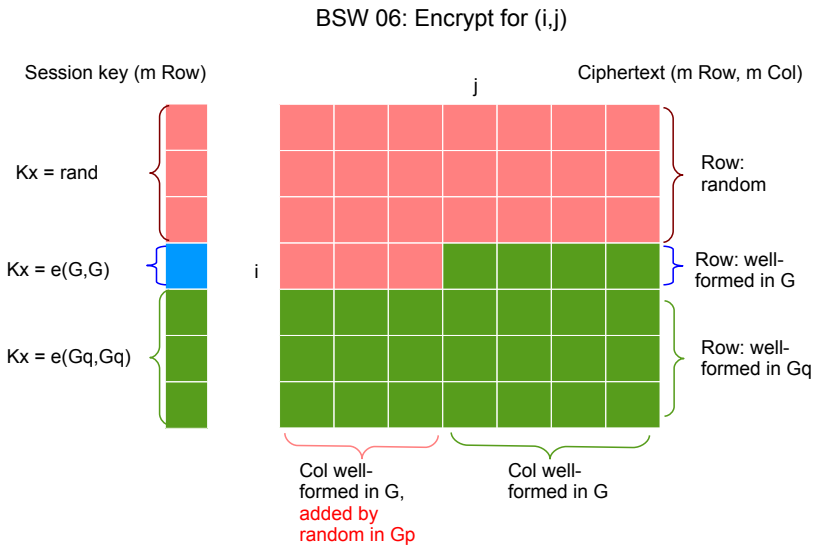


Figure 3.2: High-level view of the BSW's method.

- If $x = i$ and $y \leq j$ then the (row component, column component) for (x, y) looks like (well-formed, well-formed).
- If $x = i$ and $y < j$ or $x < i$ then the (row component, column component) are randomized either in G_q or in G_p and are thus not of type (well-formed, well-formed) and the decryption fails.

The idea is summarized in Figure 3.2. However, the detailed description is quite complicated and the security analysis requires some assumptions, in particular the Bilinear Subgroup Decision Assumption which requires that: given $g_p, g_q \in G$, a random order p element in G_T is indistiguishable from a random element in G_T. We remark however that, because the ciphertext size is of $O(\sqrt{N})$, the scheme is more about theoretical interest than of practical interest.

Trace&Revoke: Boneh-Waters scheme. Boneh and Waters [BW06b] further improve the BSW scheme by providing broadcast functionality. Their scheme is a combination of the BGW scheme and the BSW scheme, in which they make use of the broadcast technique from the BGW and of the tracing technique from the BSW scheme.

3.3 Some variants of BGW

3.3.1 Adaptive CCA Security with Constant-size Secret Keys and Ciphertexts [PPSS13]

BGW is clearly a very interesting broadcast encryption (*i.e.* an inclusive scheme) but it still has some inconveniences: i) the security is considered in the selective model and ii) it is not trivial for the scheme to efficiently revoke users (*i.e.* not an exclusive scheme) when the number of revoked users is relatively small compared to the total number of users. Our objective is to make BGW more secure and more flexible by transforming it into an *inclusive-exclusive* broadcast encryption scheme. Our contribution is realised in several steps:

- We first propose an efficient dynamic broadcast encryption scheme (called OurBE) and prove that it is selective CCA secure based on the widely-used bilinear Diffie-Hellman exponent (BDHE) assumption and a universal one-way hash function (UOWHF). The proposed scheme has constant-size ciphertexts (only two group elements), constant-size secret keys (only one group element), and a public key which increases linearly with the number of users in the system. Our scheme is a variant of the selective CPA secure BGW. The modification from BGW is minimal in the sense that our scheme has exactly the same ciphertext and secret key sizes as those of the BGW schemes and is proved secure under the same assumptions, plus the relatively weak UOWHF assumption. The minor difference is that our scheme has one extra element in the linearly-growing public key.
- We then propose an *inclusive-exclusive* broadcast encryption scheme which can simultaneously work both as a broadcast encryption and a revocation scheme, as it has the flexibility to specify either the target set or the revoked set at the time of encryption. The time complexity (in encryption and in decryption) can be made proportional to the size of the target set or to the size of the revoked set. The ciphertext and the secret key still contain respectively two and one group elements. We need to add one group element per user to the already linear size public key of the BGW scheme though.
- Finally, we show that it is possible to prove the adaptive CCA security of our scheme under generalised versions of existing assumptions. In particular, we propose generalised versions of the BDHE and the knowledge-of-exponent (KEA) assumptions, and prove that both hold in the generic group model. Under these assumptions which are reasonable generalisations of accepted assumptions, we can achieve the highest level of security with highly-efficient parameters. Namely, OurBE is provably adaptive CCA secure with constant-size ciphertexts and secret keys, and it is the first scheme to achieve such properties.

Comparison. Table 3.1 summarises our comparison between the broadcast encryption schemes that that are at least adaptive CPA or selective CCA secure in the full collusion model. The schemes in the literature with constant-size ciphertexts include a selective CCA secure scheme from [BGW05], and three adaptive CPA (ACPA) secure ones from [GW09] and [Wat09]. The schemes that do not have constant-size ciphertexts include adaptive CPA secure schemes from [DF02], [GW09] (identity-based) and [LSW10] (revocation scheme), and adaptive CCA (ACCA) secure schemes from [PPS12a].

We list plain and identity-based (IB) broadcast encryption (BE) and revocation (R) schemes. The schemes from [DF02] and [PPS12a] are generic schemes based on (hierarchical) identity-based encryption ((H)IBE), and encryption schemes (implemented under DDH), respectively. Since (H)IBE can be based on various assumptions, we simply use it in parentheses in the table. All other schemes are explicit proposals based on various bilinear Diffie-Hellman assumptions, sometimes relying upon extra assumptions such as strong unforgeability (SUF), pseudo-random functions (PRF), and the random oracle model (ROM).

For further details on parameters, we would refer to [DDG13] where our scheme has been implemented and suggested for a standardisation.

3.3.2 BGW in Multi-Channel setting [PPT13]

The context. We focus on the pay-TV scenario, in which users own decoders to decode only the channels they have subscribed to. In this context, the broadcaster sends several channels at the same time to different groups of users or target sets.

Unfortunately, previous broadcast encryption models only dealt with a single content and a single target set at a time. This was a reasonable goal but not quite suitable for pay-TV in

Scheme		O($\lvert sk_i \rvert$)	O($\lvert H \rvert$)	Security	Assumption
[DF02]	BE	$\log n$	$r \log \frac{N}{r}$	ACCA1	(IBE)
	BE	$\log^{1+\epsilon} n$	$\frac{r}{\epsilon}$	ACCA1	(HIBE)
[BGW05]	BE	**1**	**1**	SCCA	n-BDHE, SUF
[GW09]	BE	**1**	**1**	ACPA	n-BDHES, PRF, ROM
	BE	**1**	s	ACPA	n-BDHES, PRF
	IBBE	**1**	**1**	ACPA	n-BDHES, PRF, ROM
	IBBE	**1**	$\sqrt{s}$	ACPA	n-BDHES, PRF
[Wat09]	BE	n	**1**	ACPA	dBDH, dLin
[LSW10]	R	**1**	r	ACPA	dBDH, dLin
[PPS12a]	BE	**1**	$r \log \frac{N}{r}$	**ACCA**	DDH
	BE	**1**	r	**ACCA**	DDH
OurBE	BE	**1**	**1**	SCCA	n-BDHE, UOWHF
				ACCA	n-OBDHE, GKEA, UOWHF

O($\lvert \cdot \rvert$): order of size, n, s, r: number of total, targeted, revoked users.

Table 3.1: Comparison of adaptive or CCA secure broadcast encryption schemes

practice. In fact, television systems contain many channels with different sets of privileged users. One could argue that this scenario is covered by the usual systems via the use of independent broadcast encryption schemes for each channel. However, this results in a very inefficient scheme: the bandwidth or header size grows linearly in the number of channels, which could be very large; when the users zaps to another channel, one has to start from scratch and wait for the reception of the new appropriate header, which can take some time unless the decoder stores all the headers all the time.

These two problems of the limited bandwidth and limited zapping time lead to new efficiency criteria with a common solution: a broadcast encryption with a short *global header*. Our new primitive MCBE (Multi-Channel Broadcast Encryption) addresses these problems. In the following, we show that it is possible to achieve this goal in an optimal way by proposing a scheme with constant-size global header, independently of the number of channels.

The technique. In the BGW scheme, we recall that each channel can be interpreted as slots in a line and each user owns one of the slots and the user can decrypt if he/she can move the hole so that a ball falls in the hole. The trivial solution for multi-channel problem consist in making many parallel lines, each corresponding to a channel and we can encrypt line by line for each channel. Then, because the lines are independently treated, each user of a channel can move the hole in the corresponding line. However this increases the ciphertext size linearly to the number of channels. We propose a method to combine all the lines together and this reduces the ciphertext size. The main idea is that, during decryption, each user can cancel out all the balls in all lines except in the line that corresponds to the channel he/she would like to decrypt. This implicitly reduces the multi-channel framework to a single-channel one while preserving the constant ciphertext size.

Due to constraints between the various target sets, we introduce the *dummy-helper technique* that helps proving security. We eventually propose two constructions derived from the BGW

scheme. They are private broadcast encryption schemes with the following properties:

- The first construction is, asymptotically, very competitive with the BGW scheme. In fact, it achieves constant-size headers, while the private decryption key size remains linear in the number of the channels that a user has subscribed to. In addition, it is fully collusion resistant against basic selective adversaries, *i.e.* the adversaries who can only ask corruption queries to get the decryption keys of users in the selective security model (the challenge target set is announced before the setup of the global parameters). This is also the security level that the original BGW scheme achieves and our security proof holds under the standard assumption $\mathsf{n}-\mathsf{BDHE}$, as in the original BGW scheme.

- The second construction is an improvement of the previous one in order to resist strong selective adversaries who have the power of basic selective adversaries plus unlimited access to encryption and decryption queries, while keeping the parameter sizes and computational assumptions unchanged. To this aim, we introduce the *dummy-helper technique* and make use of a *random oracle* [BR93]. Our scheme is more efficient than the CCA version of the BGW scheme but our dummy-helper technique actually works in the random oracle model only.

Dummy-helper technique. In the multi-channel setting, because the session keys of all channels are compacted in only one ciphertext, there exists an implicit relationship between the session keys of the channels which could be known by the simulator without the entire knowledge of the master key. By introducing the *dummy-helper technique*, which consists of adding a new channel for one additional dummy user, we get the following interesting properties:

1. it gives our simulator the possibility to decrypt this channel and get the corresponding session key which is sufficient for the simulator to derive the other session keys and successfully answer any decryption queries.

2. by eventually publishing the decryption key of the dummy user, the center introduces a channel that can be decoded by all the users registered in the system (to send the program or ads).

We implement this dummy-helper technique in the random oracle model. It is worth noting that, despite the more complex setting of multi-channel broadcast encryption, the security is achieved under the standard assumption $\mathsf{n}-\mathsf{BDHE}$ as in the BGW scheme.

3.4 Lattice-based Approach: ℓ-LWE and Projective Sampling [LPSS14]

Since the pioneering work of Ajtai [Ajt96b], there have been a number of proposals of cryptographic schemes with security relying on the worst-case hardness of standard lattice problems, such as the decision Gap Shortest Vector Problem with polynomial gap (see the surveys [MR09, Reg10]). These schemes enjoy unmatched security guarantees: security relies on *worst-case* hardness assumptions for problems expected to be *exponentially hard* to be solved (with respect to the lattice dimension n), even with quantum computers. At the same time, they often enjoy great asymptotic efficiency, as the basic operations are matrix-vector multiplications in dimension $\widetilde{O}(n)$ over a ring of cardinality $\leq \mathcal{P}oly(n)$. A breakthrough result in that field is the introduction of the Learning With Errors problem by Regev [Reg05, Reg09], who shows it to be at least as hard as worst-case lattice problems and exploited it to derive an elementary encryption scheme. Gentry et al. shows in [GPV08] that Regev's scheme may be adapted so

that a center can generate a large number of secret keys for the same public key. As a result, the latter encryption scheme, called dual-Regev, can be naturally extended into a multi-receiver encryption scheme. We build traitor tracing schemes from this dual-Regev LWE-based encryption scheme which also enjoys public traceability. To show that we can trace the traitors, we extend the LWE problem and introduce the k-LWE problem, in which k hint vectors (the leaked keys) are given out.

The k-LWE problem. We extensively use q-ary lattices. The q-ary lattice associated to $A \in \mathbb{Z}_q^{m\times n}$ is defined as $\Lambda^{\perp}(A) = \{\vec{x} \in \mathbb{Z}^m : \vec{x}^t \cdot A = \vec{0} \bmod q\}$. It has dimension m, and a basis can be computed in polynomial-time from A. For $\vec{u} \in \mathbb{Z}_q^m$, we define $\Lambda_{\vec{u}}^{\perp}(A)$ as the coset $\{\vec{x} \in \mathbb{Z}^m : \vec{x}^t \cdot A = \vec{u}^t \bmod q\}$ of $\Lambda^{\perp}(A)$.

The k-LWE problem can be interpreted as a dual of the k-SIS problem introduced by Boneh and Freeman [BF11]. Intuitively, in both k-LWE and k-SIS, it is given as input $A \in \mathbb{Z}_q^{m\times n}$ along with k small hints $\vec{x}_1, \ldots, \vec{x}_k \in \mathbb{Z}^m$ s.t $\vec{x}_i A = \vec{0} \mod q$. The k-SIS solver is required to output a new vector $\vec{x}$, linearly independent from $\vec{x}_1, \ldots, \vec{x}_k$ such that $\vec{x}A = \vec{0} \mod q$, while the k-LWE solver is required to distinguish between

$$\frac{1}{q} \cdot U\Big(\mathrm{Im}\Big(A\Big)\Big) + \nu_\alpha^m \quad \text{and} \quad \frac{1}{q} \cdot U\Big(\mathrm{Span}_{i\le k}\Big(\vec{x}_i\Big)^{\perp}\Big) + \nu_\alpha^m.$$

where $\mathrm{Im}(A) = \{A\vec{s} : \vec{s} \in \mathbb{Z}_q^n\} \subseteq \mathbb{Z}_q^m$, $U(X)$ denotes the uniform distribution over X, $\mathrm{Span}(X)$ denotes the set of all linear combinations of elements of X, ν_α denotes the one-dimensional Gaussian distribution with standard deviation α.

These problems look hard and their hardness will be discussed latter. At the first glance, we see that one can linearly combine the hints to get a new hint and the k-SIS essentially says that this is the only way the adversary can get a new solution for the SIS problem. Regarding the k-LWE, once these hint vectors are revealed, it becomes easy to distinguish the left distribution from the uniform distribution: take one of the vectors $\vec{x}_i$, get a challenge sample $\vec{y}$ and compute $\langle \vec{x}_i, \vec{y} \rangle$; if $\vec{y}$ is a sample from the left distribution, then the centered residue is expected to be of small size which is $\ll 1$ for standard parameter settings; on the other hand, if $\vec{y}$ is sampled from the uniform distribution, then $\langle \vec{x}, \vec{y} \rangle$ should be uniform. The definition of k-LWE handles this issue by replacing $U(\mathbb{Z}_q^m)$ by $U(\mathrm{Span}_{i\le k}(\vec{x}_i)^{\perp})$.

The Scheme. The scheme is designed for a given security parameter n, a number of users N and a collusions of maximum size t. It then involves several parameters q, m, α, S. These parameters are set so that the scheme is correct (decryption works properly on honestly generated ciphertexts) and secure (semantically secure encryption and possibility to trace traitors). In particular, we set $S = \mathrm{Diag}(\sigma, \ldots, \sigma, \sigma', \ldots, \sigma') \in \mathbb{R}^{m\times m}$ where $\sigma' > \sigma$ and their respective numbers of iterations are set so that t-LWE is hard to solve.

`Setup`. The trusted authority applies the algorithm in [AP11] to generate a pair $(A, T) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}^{m\times m}$ such that $A \in \mathbb{Z}_q^{m\times n}$ and $T \in \mathbb{Z}^{m\times m}$ such that the distribution of A is within statistical distance $2^{-\Omega(n)}$ from $U(\mathbb{Z}_q^{m\times n})$; the rows of T form a basis of $\Lambda^{\perp}(A)$; each row of T has norm $\le 3mq^{n/m}$. We additionally sample $\vec{u}$ uniformly in $\mathbb{Z}_q^n$. Matrix T will be part of the tracing key tk, whereas the public key is $pk = (A, u)$.

Each user $\mathcal{U}_i$ for $i \le N$ obtains a secret key sk_i from the trusted authority, as follows. The authority executes the GPV algorithm using the basis of $\Lambda^{\perp}(A)$ which consists of the rows of T, and the standard deviation matrix S. The authority obtains a sample $\vec{x}_i$ from $D_{\Lambda_{-\vec{u}}^{\perp}(A),S}$. The standard deviations $\sigma' > \sigma$ may be chosen as small as $3mq^{n/m}\sqrt{(2m+4)/\pi}$. The user secret

key is $\vec{x}_i \in \mathbb{Z}^m$. Using the Gaussian tail bound and the union bound, we have $\|\vec{x}_i\| \leq \sqrt{m}\sigma'$ for all $i \leq N$, with probability $\geq 1 - N \cdot 2^{-\Omega(m)}$.

The tracing key tk consists of the matrix T and all pairs $(\mathcal{U}_i, sk_i)$.

`Encrypt`. The encryption algorithm is exactly the 1-bit encryption scheme from [GPV08, Se. 7.1], which we recall, for readability.[1] The plaintext and ciphertext domains are $\mathcal{P} = \{0,1\}$ and $\mathcal{C} = \mathbb{Z}_q^{m+1}$ respectively, and:

$$\texttt{Enc} : M \mapsto \left[\frac{\vec{u}^t}{A}\right] \cdot \vec{s} + \vec{e} + \left[\frac{M \cdot \lfloor q/2 \rfloor}{\vec{0}}\right], \quad \text{where } \vec{s} \hookleftarrow U(\mathbb{Z}_q^n) \text{ and } \vec{e} \hookleftarrow \lfloor \nu_{\alpha q} \rceil^{m+1}.$$

(note that we use $(A|B)$ or $\left[\frac{A}{B}\right]$ to denote the horizontal concatenation of A and B).

As explained in [GPV08], this scheme is semantically secure under chosen plaintext attacks (IND-CPA), under the assumption that LWE is hard to solve.

`Decrypt`. To decrypt a ciphertext $\vec{c} \in \mathbb{Z}_q^{m+1}$, user $\mathcal{U}_i$ uses its secret key $\vec{x}_i$ and evaluates the following function `Dec` from $\mathbb{Z}_q^{m+1}$ to $\{0,1\}$: Map $\vec{c}$ to 0 if $\langle (1\|\vec{x}_i), \vec{c} \rangle \bmod q$ is closer to 0 than $\pm\lfloor q/2 \rfloor$.

If $\vec{c}$ is a honestly generated ciphertext of a plaintext $M \in \{0,1\}$, we have $\langle (1\|\vec{x}_i), \vec{c} \rangle = \langle (1\|\vec{x}_i), \vec{e} \rangle + M \cdot \lfloor q/2 \rfloor \bmod q$, where $\vec{e} \hookleftarrow \lfloor \nu_{\alpha q} \rceil^{m+1}$. It can be shown that the latter has magnitude $\leq 2\sqrt{m}\alpha q \|(1\|\vec{x}_i)\|$ with probability $1 - 2^{-\Omega(n)}$ over the randomness of $\vec{e}$. This is $\leq 3m\alpha q\sigma'$ for all i, with probability $\geq 1 - N \cdot 2^{-\Omega(n)}$. To ensure the correctness of the scheme, it suffices to set $q \geq 4m\alpha q\sigma'$. Note that other constraints will be added to enable tracing.

3.4.1 Tracing traitors

We now present a black-box confirmation algorithm `Trace`.[2] It is given access to an oracle $\mathcal{O}^{\mathcal{D}}$ that provides black-box access to a decryption device $\mathcal{D}$. It takes as inputs the tracing key $tk = (T, (\mathcal{U}_i, (1\|\vec{x}_i))_{i \leq N})$ and a set of suspect users $\{\mathcal{U}_{i_1}, \ldots, \mathcal{U}_{i_k}\}$ of cardinality $k \leq t$, where t is the *a priori* bound on any collusion size. Wlog, we may consider that $k = t$ and $i_j = j$ for all $j \leq k$.

Algorithm `Trace` gathers information about which keys have been used to build decoder $\mathcal{D}$ by feeding different carefully designed distributions to oracle $\mathcal{O}^{\mathcal{D}}$. We consider the following $t+1$ distributions $Tr_0, \ldots, Tr_t$ over $\mathcal{C} = \mathbb{Z}_q^{m+1}$:

$$Tr_i = U\left(\mathrm{Span}((1\|\vec{x}_1), \ldots, (1\|\vec{x}_i))^{\perp}\right) + \lfloor \nu_{\alpha q} \rceil^{m+1}.$$

The first distribution Tr_0 is the uniform distribution, whereas the last distribution Tr_t is meant to be computationally indistinguishable from $\texttt{Enc}(0)$. We define p_∞ as the probability $\Pr[\mathcal{O}^{\mathcal{D}}(\vec{c}, M) = 1]$ that the decoder can decrypt the ciphertexts, over the randomness of $M \hookleftarrow U(\{0,1\})$ and $\vec{c} \hookleftarrow \texttt{Enc}(M)$. We define p_i as the probability the decoder decrypts the signals in Tr_i, for $i \in [0,t]$:

$$p_i = \Pr_{\substack{\vec{c} \hookleftarrow Tr_i \\ M \hookleftarrow U(\{0,1\})}} \left[\mathcal{O}^{\mathcal{D}}\left(\vec{c} + \left[\frac{M \cdot \lfloor q/2 \rfloor}{\vec{0}}\right], M\right) = 1\right].$$

A gap between p_{i-1} and p_i is meant to indicate that $\mathcal{U}_i$ is a traitor. The tracing works if the pirate can not distinguish between Tr_{i-1} and Tr_i, unless the user i is a traitor. This condition directly comes from the hardness of the k-LWE which we will now discuss.

[1] As usual, the encryption algorithm may be used to encapsulate session keys which are then fed into an efficient data encapsulation mechanism to encrypt the data.

[2] Note that minimal access is equivalent to standard access in our context: since the plaintext domain is small, plaintext messages can be tested exhaustively.

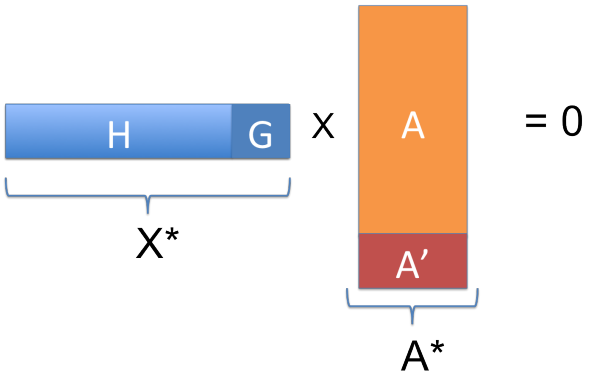
3.4.2 Hardness of k-LWE

We will briefly explain the main idea to show that the security of traitor tracing relies on the hardness of LWE, which is known to be at least as hard as standard worst-case lattice problems [Reg09, Pei09, BLP+13].

3.4.3 Hardness proof of Boneh-Freeman with exponential loss

The general framework for a hardness proof is a transformation from a SIS or LWE instance to a k-SIS or k-LWE instance. More precisely, the main steps are:

- Input: SIS or LWE instance corresponding to A
- From A, construct A^* along with k hints for A^*
- Give A^* and k hints to a k-SIS or k-LWE solver
- Based on a k-SIS or k-LWE solution for A^*, derive a SIS or LWE solution for A



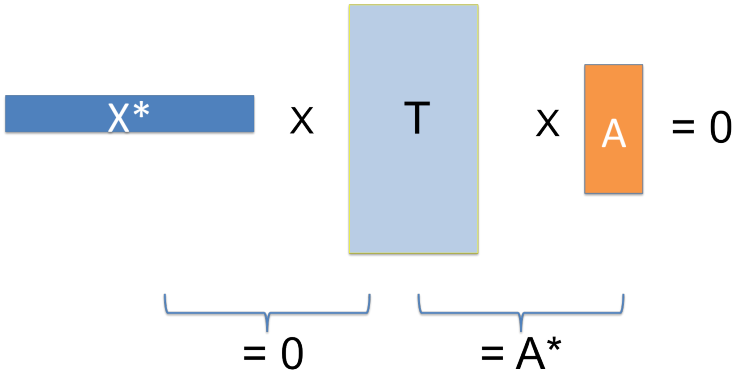
In the Boneh-Freeman approach, by extending the LWE matrix A (by k lines) to a larger matrix A^*, one can sample k hint vectors $\vec{x}_1^*, \ldots, \vec{x}_k^*$ (which forms the matrix X^* in the above figure) in the q-ary lattice $\Lambda^\perp(A^*) = \{\vec{b} : \vec{b}^t A^* = \vec{0} \bmod q\}$. The simulator then receives a solution from k-SIS solver $\vec{x}^* = (h \mid\mid g)$ s.t $\vec{x}^* \times A^* = 0 \mod q$ and $\vec{x}^*$ is linearly independent to $\vec{x}_1^*, \ldots, \vec{x}_k^*$, it can derive a solution $\vec{x}$ for the SIS problem: $\vec{x} \times A = 0 \mod q$, *i.e.*, $(\vec{x} \mid\mid 0) \times A^* = 0 \mod q$ which can be computed as

$$(\vec{x} \mid\mid 0) = (h \mid\mid g)\det(G) - (H \mid\mid G)(\det(G)G^{-1})g$$

Unfortunately, with this approach, the reduction involves a multiplication by the cofactor matrix $\det(G) \cdot G^{-1}$ over $\mathbb{Z}$ of a $k \times k$ full-rank submatrix G of the hint vectors matrix. Even though the entries of G are small, the entries of its cofactor matrix are almost as large as $\det G$, which is exponential in k. The Boneh-Freeman reduction also applies to the k-LWE case but the transformation from a LWE sample with respect to A to a k-LWE sample with respect to A^* also involves a multiplication by the cofactor matrix $\det(G) \cdot G^{-1}$. This leads to an "exponential noise blowup" which restrains the applicability range to $k \leq \widetilde{O}(1)$ if one wants to rely on the hardness of LWE with noise rate $1/\alpha \leq \mathcal{P}oly(n)$ (otherwise, LWE is not exponentially hard to solve).

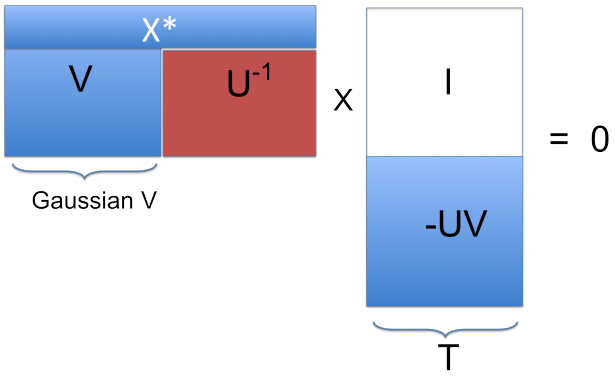
3.4.4 Our reduction with polynomial loss

We do not directly extend the matrix A but rather introduce a small transformation matrix T such that it is easy to generate a Gaussian X^* (k hints matrix): $X^* \times T = 0$. We can thus transform LWE$(A, A\vec{s}+\vec{e})$ into $k-$LWE$(TA, T(A\vec{s}+\vec{e}))$. We consequently avoid the "exponential noise blowup" because $T\vec{e}$ is of polynomial size in $\vec{e}$.



The main step in constructing such a transformation matrix is via the sampling of a Gaussian X with a small basis of $ker(X)$. It consists of sampling a $k \times m$ Gaussian matrix X along with a small unimodular matrix U such that $X \times U = I_k \,||\, 0$. We notice that X is the first k rows of U^{-1}. This tool (which is explained in details in the paper C) will allow us to define the matrix T and X^* as required:

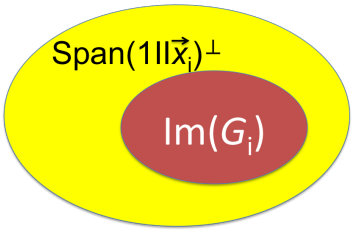
1. Sampling a Gaussian matrix V
2. Define X^* as the first k rows of $V \,||\, U^{-1}$, *i.e.,* the first k rows of V concatenated with X.



3.4.5 Public traceability

The full functionality of black-box tracing in both the Boneh-Franklin scheme and our scheme are of high complexity as both schemes rely on the black-box confirmation: given a superset of the traitors, it is guaranteed to find at least one traitor and no innocent user is incriminated. Boneh and Franklin leave the improvement of the black-box tracing as an interesting open problem. We showed that in lattice setting, the black-box tracing can be accelerated by running the tracing procedure in parallel on untrusted machines. This is a direct consequence of the public traceability property, i.e., the possibility to run the tracing procedure on public information, that our scheme enjoys. We note that almost all traitor tracing systems require the tracing key to be kept secret. Some schemes [CPP05a, PSNT06b, BW06b, BZ14] achieve public traceability and

some others achieve an even stronger security notion, namely the non-repudation. However, the setup in these schemes require some interactive protocols between the center and each user: a secure 2-party computation protocol in [Pfi96], a commitment protocol in [PW97] or an oblivious polynomial evaluation in [WHI01, KWHI01, KY02a].



In order to obtain public traceability and inspired from the notion of projective hash family [CS02], we introduce a new notion of *projective sampling family* in which each sampling function is keyed and, with a projection of the key on a well chosen space, one can simulate the sampling function in a computationally indistinguishable way. The construction of a set of projective sampling families from k-LWE allows us to publicly sample the tracing signals.

The main idea is to associate a public matrix G_i (which is characterised by a projection of $\vec{x}_i$ on a hyper plan H) to each secret key $\vec{x}_i$. It is then hard to distinguish $U(\mathrm{Span}(1\|\vec{x}_i)^\perp) + \nu_\alpha^m$ from $\mathrm{Im}(G_k) + \nu_\alpha^m$ and we can publicly sample a signal in $U(\mathrm{Span}(1\|\vec{x}_i)^\perp) + \nu_\alpha^m$ from pubic information G_i. We finally show that it is hard to distinguish $U(\mathrm{Span}_{i=1}^{j}(1\|\vec{x}_i)^\perp) + \nu_\alpha^m$ from $\mathrm{Im}(G_1) \cap \ldots \cap \mathrm{Im}(G_j) + \nu_\alpha^m$, for any$1 \leq j \leq k$ and therefore, we can publicly sample tracing signals from $G_1, \ldots, G_k$

3.5 Optimal transmission rate in Traitor tracing

3.5.1 Constant Transmission Rate in Traitor Tracing

All algebraic proposals mentioned so far result in schemes that are not quite communication-efficient: the length of each ciphertext is (at least) t times or $\sqrt{N}$ longer than the embedded plaintext.

As pointed out by Kiayias and Yung in [KY02c], an important problem in designing practical traitor tracing schemes is to ensure a low *transmission rate*, defined as the asymptotic ratio of the size of ciphertexts over the size of plaintexts, while at the same time minimize the *secret-key* and the *public-storage* rates, similarly defined as the asymptotic ratio of the size of user-keys and of public-keys over the size of plaintexts. Under this terminology, the transmission rate of all the above mentioned solutions is linear w.r.t. the maximal number t of traitors or sub-linear in the number of users, whereas in [KY02c] Kiayias and Yung show that if the plaintexts to be distributed are large (which is the case in most traitor tracing applications such as distribution of multimedia content), then it is possible to obtain ciphertexts with constant expansion rate.

In addition to the clear benefit in terms of communication efficiency, schemes with constant transmission rate also enjoy efficient black-box traceability, while schemes with linear transmission rate are inherently more limited in this regard [KY01c] (*e.g.* the black-box traitor tracing of [BF99b] takes time proportional to $\binom{n}{t}$).

Depending on the specific application, one may decide to use a scheme with constant transmission rate or a scheme with linear/sub-linear transmission rate: if one wants to directly distribute large messages, the first category is much more suitable; however if one simply wants to exchange a session key of is relatively small size, the second category enjoys better efficiency parameters.

One could think that traitor tracing schemes with linear transmission rate (*e.g.* [BF99b]) could easily be turned into schemes with constant transmission rate by means of hybrid encryption: to send a large message, pick a random session key, encrypt it with the given traitor tracing scheme, and append a symmetric encryption of the message under the chosen anonymous session key. This approach suffers however from a simple yet severe untraceable pirate strategy: decrypt the session key and make it available to the "customers" on the black market, *e.g.* via anonymous e-mail, or via text-messaging from a pre-paid cellphone. Clearly, when a traitor tracing scheme is directly used to encrypt the content, this "re-broadcasting" strategy becomes much less appealing for would-be pirates, because of the higher costs and exposure risk associated with running a high-bandwidth darknet.

The technique. The solution of Kiayias and Yung is to integrate combinatoric with algebraic methods: first construct an algebraic traitor tracing sub-scheme for two users then use collusion-secure fingerprint codes [BS98, Tar03] to combine l-sub schemes, where l is the length of codewords. The message is then decomposed into l sub-messages and each of them is encrypted by a sub-scheme. Tracing in the resulting multi-user scheme can then be performed iteratively as a sequence of l stages. In each stage, the pirate decoder is queried with ciphertexts that are valid in all l components, except for one which is crafted according to the tracing algorithm of the 2-user construction. In this way, if the decoder does not have both sub-keys for the component currently being tested, it will be unable to tell that the ciphertext is invalid, and so the tracing procedure of the 2-user subscheme will determine which of the two sub-keys the decoder holds for that component. The tracer can thus get the pirate codeword and identify a traitor from the tracing property of the embedded collusion-secure fingerprint code. In order to prevent the adversary to replay some parts of the message, Kiayias and Yung propose to apply an all-or-nothing-transform (AONT) to all the sub-messages. We propose two new techniques to optimise the transmission rate and to make the schemes of this type more practical:

- We first introduce in [FNP07b] a technique to reduce the transmission rate from 3 in Kiayias and Yung to 1 (thus optimal). However, we still need to use an AONT.
- We then propose in [PPS12b] a method to avoid AONT while preserving the optimal transmission rate. The use of AONT has some shortcomings due to the receiver having to keep the whole bunk of sub-ciphertexts to launch the decryption procedure. By avoiding AONT, the proposed scheme can be used for message tracing. Our technique requires an efficient construction of a 2-user anonymous broadcast encryption.

3.5.2 Optimal Transmission Rate [FNP07b]

As mentioned above, a common approach to extending a 2-user construction to a multi-user setting is to concatenate several instantiations (say, v) of the basic 2-user scheme. In Kiayias-Yung scheme, since tracing requires the ability to set up each component of the ciphertext to be independent from all the others, it may seem necessary to use completely unrelated instantiations of the 2-user sub-scheme for each component. Having independent components, however, clearly leads to a multi-user scheme with the same transmission rate as the underlying basic 2-user scheme, and so it would not help us attaining optimal transmission rate. In fact, our scheme at Eurocrypt '05 [CPP05a] manages to get transmission rate 1 by sacrificing component independence, and by using component-instances all very closely related to each other instead but the resulting scheme does not support black-box traceability.

To resolve the tension between transmission rate and black-box traceability, we move from the observation that, at each stage, it is sufficient that a single component can be appropriately

and independently set up from the rest; the remaining $v-1$ can all be closely related to each other. Therefore, ciphertexts in our construction include a "special" position ℓ, where encryption is performed with the instance of our 2-user scheme that is specific to the ℓ-th component; the remaining $(v-1)$ positions, instead, are encrypted using a "shared" 2-user scheme.

To prevent pirate decoders from selectively ignoring the "special" position (which is the only part of the ciphertext that encodes tracing information), we follow the approach proposed in [KY02c] by which the encryption algorithm preliminarily processes the plaintext with an AONT [Riv97, Boy99, CDH+00]. This will force decoders to decrypt *all* blocks of the ciphertext, ignoring even a single one would result in missing at least one block of the AONT-transformed plaintext, so that, by the properties of AONT's, such decoders would fail to recover *any* information about the original plaintext being transmitted. The scheme is described in Appendix B.

3.5.3 Message Tracing with Optimal Transmission Rate [PPS12b]

We propose the term "message-based traitor tracing" as a generic term that subsumes earlier variants and emphasize on the fact that we do not trace from pirate decoders but from the information embedded in the content. Fiat and Tassa are the first to consider message-based traitor tracing in [FT99], they develop *dynamic traitor tracing* to deal with pirates that rebroadcast decrypted content. They assume that there is a real-time feedback from the broadcast content to the center, so that the watermarks can be adapted to the feedback. Safavi-Naini and Wang [SNW03a] note that in this setting, dynamic TT can be prevented by delaying the rebroadcasting of the content. To take this counter-measure into account, they proposed *sequential traitor tracing*, where the mark allocation is precomputed, but users are removed according to the feedback received. They constructed a sequential TT scheme by combining error-correcting codes and watermarking. Jin and Lotspiech [JL07] claim that protection should not increase the bandwidth by more than 10 %. To solve this problem, they proposed to extend the tracing procedure over several movies (using "inner" and "outer" codes) and assumed that the pirates will not drop any block. Their sequence key block scheme permits the revocation of users after they have been traced through the rebroadcasted messages. Kiayias and Pehlivanoglu [KP09] show that the sequence key block scheme only allowed to trace and to revoke a limited number of users, and proposed a message-trace-and-revoke scheme without this limitation.

We remark that, without aiming to optimize the ciphertext rate, the use of AONT can be avoided by using robust collusion-secure code which allows the pirate to drop a fraction of the positions. This is used in [Sir07a, BP08, BN08b] to reduce the ciphertext size.However, in order to get optimal ciphertext rate in [FNP07b], AONT is compulsory, otherwise the pirate could simply drop the particular block to defeat the tracing procedure. We also notice that in all the above methods for classical tracing, each user finally gets the same plaintext and if a user redistributes this plaintext, we have no way to trace back the traitor from the distributed message. Therefore, the necessary condition for message tracing is that each user receives a different (marked) version of the plaintext. However, when the plaintext is different for each user, one cannot apply AONT for a whole fixed plaintext, otherwise all but at most one user can decrypt. The use of AONT for message tracing is thus irrelevant. Fortunately, we can still use the method of doubling one particular block by finding a way to hide this block. Our method consists in using a 2-user anonymous broadcast encryption scheme and then randomly permutting the blocks. With a 2-user anonymous broadcast encryption scheme, the pirate cannot detect any difference between an encryption for both users (which is used for all blocks but the particular block) and an encryption for one of the two users that is used for the particular protected block. Combining with the permutation of the blocks, we can show that the pirate is prevented from detecting the particular protected block. Moreover, beyond the optimisation of

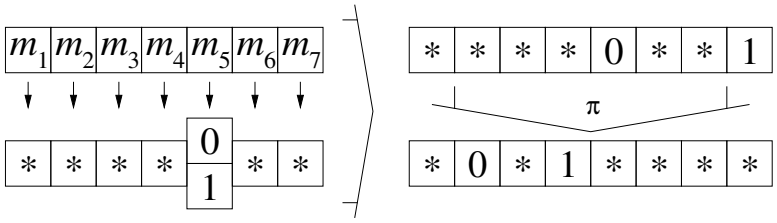


Figure 3.3: Hiding a mark at position 5 in a sequence of 7 blocks.

the ciphertext rate, by not using AONT, our scheme also enjoys the property of the sequential decryption via robust collusion secure codes as in [BP08, BN08b]: the user can sequentially decrypt the sub-ciphertexts, and does not need to wait to receive the whole ciphertext and to apply the AONT transform to start the decryption procedure.

We describe a generic construction that accomplishes this by encrypting a message consisting of n sequences of ℓ blocks, each in such a way that in sequence i, $\ell - 1$ blocks can be decrypted by both users; these blocks are not used for tracing. The remaining block is duplicated using different marks and encrypted at two positions $v_0[i], v_1[i]$, each time for one key only: the message at position $v_1[i]$ can only be decrypted by users with key 0, and the message at position $v_0[i]$ only by users with key 1. By doing this, the ciphertext will have a length of $(1 + 1/l)$-times the length of the message, plus the overhead for encryption.

3.5.4 2-user Anonymous Broadcast Encryption

We first remark that the naive way to construct a message tracing scheme is to use a PKE scheme Π and assign user keys according to the codewords of the collusion-secure code $\mathcal{T}$. If the codewords have length n, we need $2n$ instances of the PKE scheme. The main disadvantage of the PKE-based construction is the length of the user keys, which must contain one PKE key for each block. To achieve shorter keys, we use a primitive that allows encryption to either of the two users or to both of them: 2-user broadcast encryption.

Our message-traceable encryption scheme makes use of codes where the bits of the codewords are embedded in a message by doubling some parts of it, the so-called *protected blocks*. Because we do not want the adversary to learn which parts of the message contain bits of the codeword, we need a broadcast encryption scheme where a user cannot tell whether a block is destined only for his key or for both keys, a 2-user anonymous broadcast encryption (2ABE).

This requires the symmetric cipher used with this construction to be weakly robust [ABN10], since one of the decapsulated keys will be either $\perp$ or an unusable key. The construction uses one instance of the 2ABE scheme Π per bit of the codeword, encrypting $\ell+1$ messages at a time in one sequence, with the target sets determined by the positions v, w where the watermarks are embedded. In this construction, the lengths of the EK and USK are n times that of Π, and to encrypt a sequence of ℓ blocks we need $\ell + 1$ Π key-encapsulations plus $\ell + 1$ symmetrically encrypted message blocks.

A 2-user Anonymous Broadcast Encryption Scheme. We consider the 2-key 1-copyrighted public-key encryption scheme of Kiayias and Yung [KY02c] as a 2-user 1-collusion-secure anonymous broadcast encryption scheme (2ABE). For clarity of exposition, we model the scheme as a KEM.

Let G be a group of prime order q, with a generator g. The public parameters consist of (G, q, g). Since we consider the 2-user case, we drop the N parameter:

- $\mathsf{Setup}(1^\kappa)$ picks $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_q^\times$. For the two user-keys, one chooses $d'_0, d'_1 \in \mathbb{Z}_q$, and sets $\mathsf{usk}_u \stackrel{\mathsf{def}}{=} (d_u = \alpha - d'_u \cdot \beta, d'_u)$, for $u = 0, 1$. The encryption key is $\mathsf{ek} \stackrel{\mathsf{def}}{=} \{(f = g^\alpha, h = g^\beta), \mathsf{upk}_0 = h^{d'_0}, \mathsf{upk}_1 = h^{d'_1}\}$.
- $\mathsf{Encaps}(\mathsf{ek}, S; r)$ where $r \in \mathbb{Z}_q^\times$
  - if $S = \{0, 1\}$ then $c = (g^r, h^r), K = f^r$
  - else if $S = \{u\}$ then $r' \xleftarrow{\$} \mathbb{Z}_q^\times$, and $c = (g^r, h^{r'}), K = (f/\mathsf{upk}_u)^r \times \mathsf{upk}_u^{r'}$
- $\mathsf{Decaps}(\mathsf{usk}_u, c)$ computes $K = c_0^{d_u} c_1^{d'_u}$. This is equal to $g^{rd_u} \times h^{r'd'_u} = (f/\mathsf{upk}_u)^r \times h^{r'}$. In the latter encryption case, one gets the same session key. In the former encryption case, since $r' = r$, one also gets f^r.

This is a broadcast encryption, because when $S = \{u\}$ user $1 - u$ decapsulates differently. Anonymity comes from the fact that a ciphertext is either a Diffie-Hellman pair when $S = \{0, 1\}$, or a random pair in the other case.

The security analysis of our 2ABE and traitor tracing are referred to the appendix E. We note that the construction of an efficient anonymous BE in the general case remains an open problem.

Chapter 4

Discussions and Perspectives

Provable Security remains a young branch of research in cryptography. It often happens that the formalised security notions do not capture all the scenarios. In some cases, new security models need to be formalised to deal with new attacks and in some other cases, new primitives need to be introduced to better address practical demands.

In Chapter 2 and Chapter 3, we have dealt with constructions of BE/TT schemes. In this final chapter, we discuss further security models and generalisations of the BE/TT. We conclude the thesis with perspectives for further works.

4.1 Extended Attack Models

In encryption, the standard security notion is semantic security against chosen-ciphertext attacks in which the secret key is assumed to be totally hidden from adversaries' view. This assumption is not realistic in many scenarios, especially in the presence of side-channel attacks. Therefore, new security models have been proposed, namely key-leakage resilience. Since then, there have been a lot of works devoted to this new model and new methods have been proposed for proving the security under this attack model.

The same situation occurs in broadcast encryption and traitor tracing. There are new attack models that go beyond the standard formalisations and that have impacts in practice, in particular Pirate Evolution [KP07] and Pirates 2.0. *Pirate evolution* [KP07] is an attack concept against a trace&revoke scheme that exploits the combined properties of the functionalities of tracing and revocation in a tree-based scheme. By using a set of users' keys, the pirate produces an initial pirate decoder and then, whenever this pirate decoder is seized, the pirate evolves the decoder to a new one which can successfully decrypt ciphertexts. The same step can be repeated again and again, allowing to produce more and more decoders. This kind of attack has real impact on NNL schemes, which forces the designer to downgrade the efficiency of the original schemes to avoid them.

Pirates 2.0. We introduce a new concept of attacks against traitor tracing schemes [BP09]. We call attacks of this type Pirates 2.0 as they result from traitors collaborating together *in a public way*. In other words, traitors do not secretly collude but display part of their secret keys in a public place; pirate decoders are then built from this public information. The distinguishing property of Pirates 2.0 attacks is that traitors only contribute partial information about their secret key material, which suffices to produce (possibly imperfect) pirate decoders while allowing them to remain anonymous. The side-effect is that traitors can publish their contributed information without the risk of being traced; giving *strong incentives* to some of the legitimate users to become traitors. This allows coalitions to attain very large sizes, this scenario has been

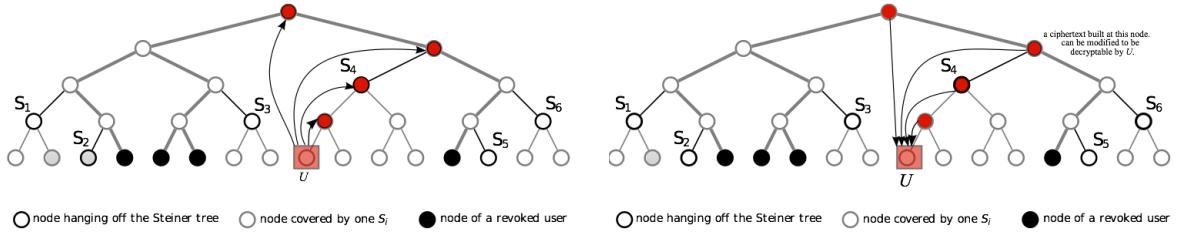


Figure 4.1: Compairison between Asano's method and our method.

deemed unrealistic in some previously considered models of coalitions.

We propose a generic model to deal with this new threat, which we use to assess the security of some of the most practical traitor tracing schemes, namely tree-based and code-based schemes. We exhibit several Pirates 2.0 attacks against these schemes, providing new theoretical insights with respect to their security. We also describe practical attacks against various instances of these schemes. The main characteristics of our new Pirates 2.0 threat are as follows:

- Anonymity Guarantee: traitors who participate in a Pirates 2.0 attack are provided with a guarantee (through the exhibition of a mathematical proof) that *they cannot be traced by the authority.*
- Partial Contributions: Traitors never need to reveal their whole secret key.
- Public Collusions: Traitors operate in a public environment, they publish secret data from their decoders.
- Large Coalitions: Traitors take part in unusually large coalitions.
- Dynamic Coalitions: Traitors can come into action only when necessary.

Counter-measures for Pirate Evolution and Pirates 2.0 [PT11, PT13]. Several methods have been proposed to limit the impacts of Pirate Evolution and Pirates 2.0. [JL09] proposes a new method of protection against pirate evolution attacks in the subset cover framework and [ZZ11, Dd11] propose some methods to deal with Pirates 2.0 but their methods significantly decrease the efficiency of the original schemes. We also propose in [PT11] a method to deal with both types of attack. The main idea is to perform in two steps: the first step is to combine all the sub-keys into a unique key which is based on the idea of Asano in [Asa02] (which has then been extended in [AK05]); the second step is to make the user key undecomposible which is not achieved in Asano's scheme but we can achieve this feature with the help of Wildcarded WIBE (WIBE) [ABC$^+$11], via a mechanism of delegating ciphertexts to the suitable leaf where the user has the key to decrypt. Our method can be seen thus as a top-down method while Asano's method can be seen as a bottom-up method (a key at a leaf can be used to derive a key at a higher node to decrypt the corresponding ciphertext and therefore when the intermediate key is leaked, it is un traceable), see Figure 4.1. All these methods can however deal with particular forms of Pirates 2.0. A step forward dealing with the general form of Pirates 2.0 is via key-leakage resilience that we propose in [PT13].

4.2 Generalised Primitives

In recent years, more and more generalised primitives for encryption have been introduced, the most general one being Functional Encryption. At first, it may seem a redundant direction

because we already have Multi-Party Computation (MPC) [Yao82] protocols allowing users to compute and learn arbitrary functions of their joint messages without revealing anything except for the final answer. However, multi-party computation is normally an interactive protocol while one of the advantages of a typical encryption scheme is that it is non-interactive. In fact, functional encryption combines the benefits of the two worlds: define any kind of access control to the data while optimise low communication round complexity.

The same situation can be observed when comparing Functional Encryption and Broadcast Encryption. While Broadcast Encryption is formally a particular case of Functional Encryption, it plays an independent role because it has its own practical impacts and because the general solution for Functional Encryption becomes inefficient when one simplifies it to fit the broadcast encryption framework. An interesting step would be to generalise Broadcast Encryption to some new primitives that have practical impact and that are not too general to loose efficiency. In this direction, we proposed the Multi-channel BE that has been described in Chapter 3 and Decentralised model for Broadcast Encryption.

Decentralised Dynamic Broadcast Encryption [PPS12a]. A broadcast encryption system generally involves three types of entities: the group manager that manages the membership, the encryptor that encrypts the data to the registered users according to a specific policy (the target set), and the users that decrypt the data if they are authorized by the policy. Public-key broadcast encryption can be seen as removing the special role of the encryptor by allowing anybody to send encrypted data. We go a step further in the decentralisation process by removing the group manager: the initial setup of the group, as well as the addition of further members to the system, do not require any central authority. Our construction makes use of well-known primitives and can be considered as an extension of the subset-cover framework. It allows for efficient concrete instantiations with parameter sizes that match those of the subset-cover constructions, while at the same time achieving the highest security level in the standard model under the DDH assumption.

4.3 Some Remarks and Open Problems

On Full Collusion in TT As mentioned, there is currently no efficient fully collusion resistant traitor tracing, even with the minimal one-wanness security level. Very recently, Boneh and Zhandry [BZ14] have proposed a scheme with poly-log size parameters. This scheme relies on indistinguishability obfuscation [GGH$^+$13c] of which the security foundation remains to be studied and practicality remains to be investigated. An efficient construction of a fully collusion resistant traitor tracing scheme remains one of the main open problems in multi-user cryptography.

On Full Collusion in BE In Section 3.3.1, we propose an efficient fully collusion resistant broadcast encryption scheme which is a variant of the BGW scheme. The sizes of the secret keys and ciphertexts in this scheme are asymptotically optimal, i.e., constant. Considering only the standard assumptions, our scheme provides shorter ciphertexts than the known CCA secure schemes. Considering the extended assumptions, our scheme is the first scheme to achieve constant size secret keys and ciphertexts along side with adaptive CCA security, see Table 3.1. However, the problem of designing adaptive CCA schemes that achieve constant size secret keys and ciphertexts under standard assumptions remains open.

On Lattice-based Schemes The proposed lattice-based traitor tracing scheme resists Chosen Plaintext Attacks. It seems quite challenging to devise such an CCA-secure scheme under lattice hardness assumptions. Intuitively, in a traitor tracing scheme the users own parts

of a master secret (e.g., each user owns a short vector in a shared lattice, or a discrete log representation with respect to a shared set of group elements), and we attempt to prevent traitors from gaining knowledge of more than their share of the secret information. This requirement seems to be in opposition with the underlying design of all known lattice-based CCA-secure encryption schemes [Pei09, PW08, CHKP10, ABB10a, ABB10b], as the receiver uses the full secret information (a short basis of lattice) to verify the well-formedness of the ciphertext it decrypts.

Our scheme can be directly adapted to broadcast to a small group of users (the tracing signals may be used for sending messages to users who own one of these keys, as if they were suspect keys). This possibility of broadcasting to a small set of the sub-keys can then be combined with the complete subtree framework in [NNL01] to deal with revocation (in which each tree node is associated with a sub-key $\vec{x}_i$ and the key of each user is a set of sub-keys on the path from the root to the user). However, this leads to an inefficient scheme that can only handle a small set of the revoked users (while maintaining security). An algebraic construction of a lattice-based trace and revoke scheme is a challenging open problem.

On Additional Features The construction of an efficient Decentralised BE with constant round complexity interaction or of an efficient anonymous BE remains open problems.

4.4 Perspectives

Our objective is to deepen and broaden the subject of this HdR to the more general case of multi-user communications. We will continue to work in part on encryption but also to focus on group signature, functional encryption, distributed cryptography and applications that combine different primitives, in particular electronic voting. The main directions are:

Security versus efficiency: Sometimes we have to sacrifice the effectiveness of a scheme in favour of its security level. The first objective of our project is to consider a balance between security and efficiency according to the requirements of practical applications. Firstly, we study the trade-off between security and efficiency in new attack models and security notions which might not reach the maximum level of security but would reflect the practical requirements. Secondly, we continue to improve the effectiveness of the schemes with a high level of security, in particular by using some recently developed tools being non-committing encryption techniques and lossy trapdoor functions. Finally, we continue the work started in the master thesis of Manh Cuong Ngo to propose a scheme of electronic voting which would better meet the practical requirements by minimising the intervention of the authorities (via the introduction of a new concept of traceability in electronic voting).

Security against quantum attacks: The ultimate implementation of quantum machine would make many cryptographic schemes vulnerable. For cryptographic propose, there are actually some algorithmic problems which remain unresolved by quantum machines, *e.g.* decoding of linear codes, problems on Euclidean networks and the LWE problem. The efficiency of the schemes that are based on those problems is still quite limited. Many interesting directions have been explored to improve the efficiency of these schemes. For example, an interesting direction, initiated by Lyubashevsky-Peikert-Regev, is to consider the LWE problem in a ring of integers of cyclotomic extensions (Ring-LWE). One of our objectives is to improve the efficiency of our lattice based traitor tracing scheme (described in Chapter 3) by adapting it into Ring-LWE setting. This is a challenge in itself because

in the multi-user case in particular, the pirate is strengthened when we work in Ring setting. Indeed, once the pirate gets a key, it can generate several other keys via the ring automorphisms. Another objective we are working on is to consider the difficulty of the LWE problem in other algebraic extensions than the cyclotomic extension.

Relationship between the primitives: Relationship between primitives is an important area that helps us to better understand the challenge of designing these primitives. Regarding the multi-receiver encryption, it is shown that a construction of a traitor tracing scheme generically leads to a group signature [KY03] and a traitor tracing scheme can be generically constructed from a collusion secure code and a symmetric encryption, as described in Chapter 2. The opposite cases are however neither disproved nor confirmed. It is also interesting to note that relationships exist not only between multi-user communication schemes but also between a multi-user scheme and a "one-to-one" scheme. Our goal is to study in more details the relationships between the multi-user primitives and those between these primitives and other more traditional primitives such as encryption, signature and identification.

Part II

Curriculum vitæ & publications

Curriculum vitæ

État civil

Duong Hieu PHAN, né le 12 juin 1978 à Hanoi (Vietnam), marié, 1 enfant (Dinh-Khanh, né le 4 juin 2009).

Formation

2002 – 2005 Doctorat dirigé par David Pointcheval, sur *Sécurité et efficacité de schémas cryptographiques.* Diplôme obtenu le 16 septembre 2005 – *mention Très Honorable.*

2001 – 2002 DEA Algorithmique à l'Université Paris 7 et ENS – *mention Bien.*

1996 – 2001 Licence et Maîtrise d'Informatique – Ecole Polytechnique de Hanoi – *mention Très Bien.*

1993 – 1996 Lycée annexe de l'*École normale supérieure de Hanoi*, spécialité Mathématique – *mention Très Bien.*

Cursus professionnel

2007 – présent Maître de conférences à l'Université Paris 8

2006 – 2007 Ingénieur de recherche à France Télécom R&D, Issy-les-Moulineaux.

2005 – 2006 Post-doc à l'University College London.

2002 – 2005 Allocataire de recherche à l'Ecole normale supérieure.

Valorisation de la recherche

2016 General co-Chair d'**Asiacrypt 2016**

2006 Membre du Comité d'organisation de**Vietcrypt 2006**

2013– présent Membre du *Steering Committee* d'**Asiacrypt**

2014 Membre du comité de programme d'**Asiacrypt '14**
Membre du comité de programme d'**Africacrypt '14**

2010 Membre du comité de programme de **PKC '10**

2009 Membre du comité de programme d'**Eurocrypt '09**

2007 Membre du comité de programme de **ProvSec '07**
Membre du comité de programme de **WISA '07**

2006 Membre du comité de programme de **VietCrypt '06**
Membre du comité de programme d'**InsCrypt'06**
Membre du comité de programme de **RIVF'06**

2005 Membre du comité de programme de **CISC '05**

Séminaires invités (depuis 2006) : Congrès SMF-VMS de Mathématiques France / Vietnam (2012), Vietnam Institute for Advanced Study in Mathematics (2012, 2013), Université Nationale du Vietnam (2010, 2012), Ecole normale supérieure (2009, 2012), Université Paris 13 (2010), Université Paris 8 (2007, 2014), Université de Caen (2007, 2009, 2013), University of Bristol, UK (2006), University College London, UK (2006 et 2007), France Telecom Paris et Caen (2007, 2008), Telecom Paris (2007).

Prime : PES 2009 (Prime d'Excellence Scientifique du Ministère de l'Education Supérieure et de la Recherche)

Jurys et rapports de thèses

Membre de jury de thèse

2013 Mario Strefler (Ecole normale supérieure)
Diffusion chiffrée avec traçage de traîtres.

2013 Amar SIAD (Université Paris 8)
Protocoles de génération des clés pour le chiffrement basé sur de l'identité.

Encadrement de la recherche

2009–2013 Thèse de Doctorat (co-direction) :
Viet-Cuong Trinh

2011–2013 Stage postdoctoral (encadrement partiel) :
Siamak Fayyaz Shahandashti, Université Paris 8 & ENS
Elizabeth A. Quaglia, Université Paris 8 & ENS

2012–2013 Stage de Master 2 (encadrement partiel)
Ngo Manh-Cuong, Ecole Polytechnique

Enseignement

2013 –2014
Cours/TD de Master 2 "Sécurité prouvable en cryptographie", *Université Paris 8*
Cours/TD de Licence de Combinatoire à l'*Université Paris 8*

2012 – 2013

Cours/TD de Master 2 “Sécurité prouvable en cryptographie”, *Université Paris 8*
Cours/TD de Licence de Combinatoire à l’ *Université Paris 8*

2011– 2012

Cours/TD de Master 2 “Sécurité prouvable en cryptographie ”, *Université Paris 8*

2010 – 2011

Cours/TD de Master 2 “Sécurité prouvable en cryptographie ”, *Université Paris 8*

2009 – 2010

Cours/TD de Master 2 “Sécurité prouvable en cryptographie ”, *Université Paris 8*
Cours/TD de Master 1 “Théorie de la complexité”, *Université Paris 8*
Cours/TD de Licence “Algèbre linéaire ”, *Université Paris 8*
Cours/TD de Licence “Combinatoire”, *Université Paris 8*

2008 – 2009

Cours/TD de Master 2 “Sécurité prouvable en cryptographie ”, *Université Paris 8*
Cours/TD de Master 1 “Théorie de la complexité”, *Université Paris 8*
Cours/TD de Licence “Aanalyse ”, *Université Paris 8*

2007 – 2008

Cours/TD de Master 2 “Sécurité prouvable en cryptographie ”, *Université Paris 8*
Cours/TD de Master 1 “Théorie de la complexité”, *Université Paris 8*
Cours/TD de Licence “Algèbre linéaire ”, *Université Paris 8*
TD de Licence “Mathématiques générales”, *Université Paris 8*

2004 – 2005

TD “Les bases de la programmation et de l’algorithmique”, *École polytechnique* (1ère année)
TD “La programmation en C”, *École Nationale Supérieure de Techniques Avancées* (2ième année)

2003 – 2004

TD “Les bases de la programmation et de l’algorithmique”, *École polytechnique* (1ère année)
TD “La programmation en C”, *École Nationale Supérieure de Techniques Avancées* (2ième année)

Personal Publications

Refereed papers

1. San Ling, Duong Hieu Phan, Damien Stehlé and Ron Steinfeld, *Hardness of k-LWE and Applications in Traitor Tracing*, **Advances in Cryptology - CRYPTO 2014**, LNCS 8616, pages 315-334, Springer-Verlag, 2014.

2. Hung Q. Ngo, Duong Hieu Phan and David Pointcheval, *Black-box Trace & Revoke Codes*, **Algorithmica**, Springer, vol. 67, no. 3, Pages 418-448, 2013.

3. Duong Hieu Phan, David Pointcheval, Siamak F Shahandashti and Mario Strefler, *Adaptive CCA Broadcast Encryption with Constant-Size Secret Keys and Ciphertexts*, **In IJIS - International Journal of Information Security**, vol. 12,, no. 4, Pages 251-265, 2013.

4. Duong Hieu Phan, David Pointcheval and Viet Cuong Trinh, *Multi-Channel Broadcast Encryption*, **ASIACCS 2013, ACM Symposium on Information, Computer and Communications Security**, ACM Press, Pages 277-286, 2013.

5. Philippe Guillot, Abdelkrim Nimour, Duong Hieu Phan and Viet Cuong Trinh, *Optimal Public Key Traitor Tracing Scheme in Non-Black Box Model*, **AFRICACRYPT 2013**, LNCS 7918, pages 140-155, Springer-Verlag, 2013.

6. Duong Hieu Phan and Viet Cuong Trinh, *Key-Leakage Resilient Revoke Scheme Resisting Pirates 2.0 in Bounded Leakage Model*, **AFRICACRYPT 2013**, LNCS 7918, pages 342-358, Springer-Verlag, 2013.

7. Michel Abdalla, Angelo De Caro and Duong Hieu Phan, *Generalized Key Delegation for Wildcarded Identity-Based and Inner-Product Encryption*, **IEEE-TIFS, IEEE Transactions on Information Forensics & Security**, Volume 7, Issue: 6, Pages 1695 - 1706, 2012.

8. Duong Hieu Phan, David Pointcheval and Mario Strefler, *Message Tracing with Optimal Ciphertext Rate*, **LatinCrypt 2012**, LNCS 7533, pages 56-77, Springer-Verlag, 2012.

9. Duong Hieu Phan, David Pointcheval and Mario Strefler, *Decentralised Dynamic Broadcast Encryption*, **SCN 2012**, LNCS 7485, pages 166-183, Springer-Verlag, 2012.

10. Duong Hieu Phan, David Pointcheval and Mario Strefler, *Security Notions for Broadcast Encryption*, **ACNS 2011**, LNCS 6715, pages 377-394, Springer-Verlag, 2011.

11. Duong Hieu Phan and Viet-Cuong Trinh, *Identity-Based Trace and Revoke Schemes*, **ProvSec 2008**, LNCS 5339, pp. 133–148, Springer.

12. Olivier Billet and Duong Hieu Phan, *Traitors Collaborating in Public: Pirates 2.0*, **Advances in Cryptology - EUROCRYPT 2009**, LNCS 5479, pages 189-205, Springer-Verlag, 2009.

13. Olivier Billet and Duong Hieu Phan, *Efficient Traitor Tracing from Collusion Secure Codes*, **The 3rd International Conference on Information Theoretic Security-ICITS 2008**, LNCS 5155, pages 171-182, Springer-Verlag, 2008.

14. Yvo Desmedt and Duong Hieu Phan, *A CCA Secure Hybrid Damgaard's ElGamal Encryption*, **ProvSec 2008,**, LNCS 5324, pages 68-92, Springer-Verlag, 2008.

15. Yvo Desmedt, Helger Lipmaa and Duong Hieu Phan, *Hybrid DamgN̆rd Is CCA1-Secure under the DDH Assumption*, **The 7th International Conference on Cryptology and Network Security–CANS 2008**, LNCS 5339, pages 18-30, Springer-Verlag, 2008.4284, pp. 332–347.

16. Nelly Fazio, Antonio Nicolosi and Duong Hieu Phan, *Traitor Tracing with Optimal Transmission Rate*, **10th International Conference on Information Security – ISC 2007**, LNCS 4779, pages 71-88, Springer-Verlag, 2007.

17. Michel Abdalla, Alex Dent, John Malone-Lee, Gregory Neven, Duong Hieu Phan and Nigel Smart., *Identity-Based Traitor Tracing*, **Public Key Cryptography - PKC 2007**, LNCS 4450, pages 361-376, Springer-Verlag, 2007.

18. Duong Hieu Phan, *Traitor Tracing for Stateful Pirate Decoders with Constant Ciphertext Rate*, **Vietcrypt 2006**, LNCS 4341, pages 354-365, Springer-Verlag, 2006.

19. Duong Hieu Phan, Rei Safavi-Naini and Dongvu Tonien, *Generic Construction of Hybrid Public Key Traitor Tracing with Full-Public-Traceability*, **33rd International Colloquium on Automata, Languages and – ICALP 2006**, LNCS 4052, pages 264–275, Springer-Verlag, 2006.

20. Hervé Chabanne and Duong Hieu Phan and David Pointcheval, *Public Traceability in Traitor Tracing Schemes*, **Advances in Cryptology – EUROCRYPT 2005**, LNCS 3494, pages 542–558, Springer-Verlag, 2005.

21. Benoît Chevallier-Mames and Duong Hieu Phan and David Pointcheval, *Optimal Asymmetric Encryption and Signature Paddings*, **ACNS '05**, LNCS 3531, pages 254–268, Springer-Verlag, 2005.

22. Duong Hieu Phan and David Pointcheval, *OAEP 3-Round: A Generic and Secure Asymmetric Encryption Padding*, **Advances in Cryptology – ASIACRYPT 2004**, LNCS 3329, pages 63–77, Springer-Verlag, 2004.

23. Duong Hieu Phan and David Pointcheval, *On the Security Notions for Public-Key Encryption Schemes*, **SCN 2004**, LNCS 3352, pages 33–47, Springer-Verlag, 2004.

24. Duong Hieu Phan and David Pointcheval, *About the Security of Ciphers (Semantic Security and Pseudo-Random Permutations)*, **SAC 2004**, LNCS 3357, pages 185–200, Springer-Verlag, 2004.

25. Duong Hieu Phan and David Pointcheval, *Chosen-Ciphertext Security without Redundancy*, **Advances in Cryptology – ASIACRYPT 2004**, LNCS 2894, pages 1–18, Springer-Verlag, 2003.

26. Duong Hieu Phan and David Pointcheval, *A Comparison between two Methods of Security Proof*, **RIVF 2003**, Proceeding of RIVF, pages 1–18

27. Thanh Thuy Nguyen, Duong Hieu Phan and Yamanoi Takahiro, *Some Preliminary Results on the Stableness of Extended F-rule Systems.*, **Journal of Advanced Computational Intelligence**, Pages 252-259, Vol.7 No.3, 2003.

Patent Applications

1. Olivier Billet and Duong Hieu Phan, *Traceable System for Encrypting/Decrypting Broadcast Digital Data, WO2009053605*

2. Olivier Billet and Duong Hieu Phan, *Obtaining Derived Values Depending on a Secret Master Value, EP2153575*

Part III

Appendix: Articles

Appendix A

Black-box Trace&Revoke Codes

Algorithmica 2013

[NPP13] with Hung Q. Ngo and David Pointcheval

Abstract : *We address the problem of designing an efficient broadcast encryption scheme which is also capable of tracing traitors. We introduce a code framework to formalize the problem. Then, we give a probabilistic construction of a code which supports both traceability and revocation. Given N users with at most r revoked users and at most t traitors, our code construction gives rise to a Trace&Revoke system with private keys of size $O((r+t)\log N)$ (which can also be reduced to constant size based on an additional computational assumption), ciphertexts of size $O((r+t)\log N)$, and $O(1)$ decryption time. Our scheme can deal with certain classes of pirate decoders, which we believe are sufficiently powerful to capture practical pirate strategies.*

In particular, our code construction is based on a combinatorial object called (r,s)-disjunct matrix, which is designed to capture both the classic traceability notion of disjunct matrix and the new requirement of revocation capability. We then probabilistically construct (r,s)-disjunct matrices which help design efficient Black-Box Trace & Revoke systems. For dealing with "smart" pirates, we introduce a tracing technique called "shadow group testing" that uses (close to) legitimate broadcast signals for tracing. Along the way, we also proved several bounds on the number of queries needed for black-box tracing under different assumptions about the pirate's strategies.

A.1 Introduction

In many real-world applications such as Pay-TV, satellite radio, and the distribution of copyright-protected materials, a content provider needs to broadcast digital information to a specific set of users (e.g., subscribers) who were given key(s) for decrypting the content. Two natural requirements arise in such setting. First, the broadcast system should be able to painlessly *revoke* the receiving rights of an arbitrary subset of subscribers, probably because they unsubscribed from the service or violated some rules. This is the essence of the *broadcast encryption* problem [Ber91, FN93]. Second, some users might collude to build a pirate decoder and distribute it, for a fee or not. Or, a pirate might achieve similar effects via hacking accounts of legitimate users. Either way, such users are called *traitors*. It is thus desirable for the system to be able to trace (and then revoke) at least one traitor by examining the pirate decoder. This is the *traitor*

tracing problem [CFN94b]. All in all, broadcast encryption systems which are capable of both tracing and revoking would be widely useful.

Many existing works studied the two problems separately, leading to inefficiency when applying the solution to one problem to cope with the other. For example, *collusion-secure codes* [BS95] provide a powerful tool against the illegal distribution of fingerprinted material in settings satisfying the so-called *Marking Assumption* [BS95, Tar03]. Though designed for fingerprinting large digital objects, these codes have been widely applied in the context of multi-user encryption for tracing traitor, motivated by the work of Kiayias and Yung [KY02c] in which a black-box tracing scheme with constant ciphertext rate was proposed. The schemes in [FNP07b, Sir07a, BN08b, BP08] belong to this class.

A drawback of employing collusion-secure codes for multi-user encryption is the resulting relatively large key size. For the marking assumption to be valid, each user – characterized by a codeword – should be assigned one out of two (or one out of many, with larger alphabet such as in IPP codes [SSW00]) keys for each position of the codeword. Indeed, the lower bound established in [Tar03] shows that the code length must be $\Omega(t^2 \log(N/\epsilon))$ for a system of N users with at most t traitors and with the failure probability ϵ of the tracing procedure. This means the private key sizes of users cannot be improved beyond this bound. The construction in [Tar03] has a code length of about $100t^2 \log(N/\epsilon)$, making the schemes inefficient for general practical usage.

Due to the quadratic dependency on the number of traitors, code-based schemes are only suitable for applications where the number of traitors is relatively small, say $t = O(\log N)$. For example, in the Pay-TV application the users' secret keys are stored in tamper-resistant smartcards, making it difficult and time-consuming to recover a key: recovering one key in a tamper-resistant smartcard does not necessarily help speed up the recover of another key in another tamper-resistant smartcard. In such applications, it is probably reasonable to assume a small value of t. However, even when the number of traitors is small, collusion-secure codes can still be quite inefficient. For instance, for a system of four millions users and at most $t = 22$ traitors, Tardos' code induces a system where each user's private key is composed of more than one million sub-keys.

Recently, schemes based on collusion-secure codes allowing erasure have been made more practical in terms of ciphertext size (as small as a constant, as shown in [BN08b, BP08]). However, in this case the code length and thus the private key size should be even longer.

A relatively small number of proposed systems, called *trace&revoke* systems [NP00, NNL01, BW06b], do address both traitor tracing and broadcast encryption. These schemes can roughly be classified into three categories: schemes based on some forms of polynomial interpolation [NP00, DF03], schemes in the tree-based subset-cover framework [NNL01, HS02, DF02], and pairing-based schemes that support full collusion [BW06b].

However, there are still fundamental shortcomings. The schemes in [NP00, DF03] have to fix an a-priori bound of the size of the collusion and as soon as the pirate could collect more than this bound of keys, from traitors or revoked users, it can totally break the schemes (by reconstructing the master secret key). The schemes [NNL01, DF02] are rather efficient but they can only deal with a non-standard notion of traitor tracing where the tracer can either identify a traitor or render pirate decoder useless (this is mentioned in [NNL01]). The scheme [BW06b] can deal with full collusion but with a large ciphertext size of $O(\sqrt{N})$ even when there is no traitor in the system.

The major objective of our paper is to propose a new code-based framework for constructing black-box Trace&Revoke systems which have small private key *and* ciphertext sizes. We will deal with *black-box tracing*, which means the traitors are traced by simple interactions with the pirate decoder. The decoder can be reset as many times as we want, i.e. it is stateless.

Contributions and paper outline. Section A.2 formalizes the notion of revocable codes, and efficient conjunction revocable codes in particular. The Complete-Subtree scheme [NNL01] is a type of conjunction revocable code. The decoding possibility of a user (i.e. a codeword) is captured by a *predicate*, which is a binary relation indicating whether a codeword is capable of decrypting a signal.

Section A.3 formalizes the traceability of codes. We introduce two notions in order to formalize the capability and the strategy of pirate decoders:

- Each pirate decoder, produced by a collusion C, is associated to a word in a *Useful Feasible Set of* C. Roughly, this set is defined such that if a signal can be decrypted by every member of C then it can also be decrypted by a word in the useful feasible set. This notion formalizes useful pirate decoders because the pirate decoder should assure the "minimum" capability of the collusion: when all members in the collusion can decrypt a signal, the pirate decoder should also be able to decrypt that same signal. However, Useful Feasible Set alone does not capture "smart" decoders in the sense that it does not use any strategy: whenever it can decrypt, it will decrypt.

- The pirate decoder could of course choose an anti-tracing strategy by refusing to decrypt signals that it considers abnormal or coming from a tracing procedure. We introduce the notion of *qualified signals* to formalize the anti-tracing strategies of pirates.

We borrow the notion of useful feasible set from collusion-secure codes to formalize the capability of the colluders. We note that, in collusion-secure codes, a word in the feasible set is also associated to a perfect decoder and in order to deal with smarter pirate decoders, one should use a collusion-secure code with an all-or-nothing transform [KY02c] or one should use a robust collusion-secure codes that allow erasure [Sir07a, BP08, BN08b]. In our setting, the pirate's strategy is formalized by the notion of qualified signals.

In Section A.3, we also indicate a simple connection between traceability of codes with the so-called *disjunct matrices*, a classic combinatorial object which has "built-in" tracing capability.

The problem with disjunct matrices is that they have no "built-in" efficient revocation capability. Indeed, disjunct matrices or equivalently *cover-free families* have been used for traitor tracing in [TSN06]. However, by following the tracing framework of [BF99b] it cannot be used for revocation. We deal with this problem in Section A.4 by introducing a new combinatorial object called (r, s)-*disjunct matrices*, which retains the tracing-capability of disjunct matrices while also supports revocation. As disjunct matrices have applications in diversely many areas [DH00], we believe that the new and stronger notion of (r, s)-disjunct matrices will be widely applicable as well.

Section A.5 is the heart of the paper, where we present a method for constructing good (r, s)-disjunct matrices which allow for tracing and efficient revocation. The resulting code yields a Trace&Revoke scheme with private key size and ciphertext length $O((t + r) \log(N/(t + r)))$ for N users, at most r revoked users and at most t traitors. The constants hidden in the big-O are small (≤ 8). This randomized construction yields a key assignment scheme where users pick their keys independently from the same distribution and all keys have the same role. Thus, unlike the complete-subtree method which leads to a highly asymmetric key assignment making it not suitable for tracing smart pirate decoders, our code has better "built-in" support for traceability against non-trivial pirate strategies.

Rigorously, we deal with non-trivial pirates (that are characterized by some qualified predicates). For a probabilistic code where codewords are picked independently from the same distribution and all keys used in encryption have the same role, a non-trivial pirate can estimate the number of keys used in a normal encryption and can refuse to decrypt a ciphertext that contains too few or too many keys that lies outside its estimated interval. This strategy of pirate,

called weight-limited pirate, is formalized under interval qualified predicate. To cope with this strategy of pirate decoder, we introduce a tracing technique called *shadow group testing* that uses (close to) legitimate broadcast signals for tracing. In particular, in one setting, we consider general pirate decoders that can used any strategy. We show that the problem of deciding whether a given signal is a legitimate broadcast signal (here, broadcast signal is a ciphertext targeted to all users) or a tracing signal is **NP**-hard. We thus establish that the shadow group testing technique can be used in conjunction with our code to construct revocable codes that are traceable against non-trivial pirates (modulo the computational hardness), in the conventional stateless setting where the pirate decoder could be resettable. This does not show that our trace&revoke code can be used to deal with any pirate decoder but it can be seen as a step toward this goal.

For tracing a particular type of pirate decoder which only decrypts signals of certain weights, we prove upper and lower bounds on the number of tests needed for a variant of group testing where each test must consist of a given number of items.

Last but not least, we prove upper- and lower-bounds for the number of black-box queries necessary in the information-theoretic limit when the pirate only decrypts signals which are legitimate broadcast signals (and when it has the keys). This result applies to arbitrary Trace&Revoke system, not just code-based ones like ours, as long as the pirate decryption assumption is valid.

Section A.6 discusses the questions of how to optimize code lengths (using multi-user tracing families), private key size (down to a constant using Asano' method [Asa02]), and how to deal with unbounded number of traitors/revoked users.

A.2 Revocable Codes

A.2.1 General settings

Broadcast encryption (BE) schemes enable the sender of a message to specify a subset of the users the message will be sent to, called the *target set* or the *privileged set*. The complement of the target set is called the *revoked set*. To revoke the receiving rights of some desired subset of users, a BE scheme typically generates three pieces of data: (a) the *Id Header*, which is a bit-string that unambiguously identifies the target set/revoked set; (b) the *Key Header*, which encapsulates a session key for the privileged users; and (c) the *Message Body*, which contains the payload encrypted with the session key.

In what follows, we describe a BE scheme based on codes. Roughly speaking, each user is associated with a "codeword" which determines the private key(s) assigned to that user. To revoke a subset of users, a code-based BE scheme generates a "signal" c which is a word (not necessarily a codeword) from which the Key Header will be constructed. The signal c will have to be "compatible" with the codewords assigned to privileged users and "incompatible" with revoked users so that only privileged users can decode. The notation of compatibility is captured by a Boolean predicate associated with the code. The formal definitions of broadcast encapsulation, public-key encryption and secret sharing schemes are given in Appendix A.7.

Definition A.2.1 [(ℓ, N)-Code] Given a finite *alphabet* Σ, and positive integers ℓ and N, an (ℓ, N)-*code* Γ is an N-subset of Σ^ℓ, namely $\Gamma \subseteq \Sigma^\ell$ and $|\Gamma| = N$. Members of Σ^ℓ are called *words*. Members of Γ are called *codewords*. The quantity ℓ is called the *length*, and N the *size* of the code.

In order to build a BE scheme, we associate a codeword w to each user. Henceforth, without loss of generality we use codewords to identify users. Given a set $R \subseteq \Gamma$ of revoked users, the code-based BE scheme broadcasts by first generates a signal $c \in \Sigma^\ell$ which is not necessarily a

codeword. The signal will be used to generate the session key for broadcasting. A user w is "compatible" with a signal w iff a corresponding predicate is true:

Definition A.2.2 [Predicate] We associate a code Γ with a predicate $\mathsf{D} : \Sigma^\ell \times \Sigma^\ell \to \{0,1\}$. The boolean value $\mathsf{D}(w,c)$ indicates whether the word/user w is compatible with the signal c. The practical semantic is that user w is able to recover the content associated with the signal c iff $\mathsf{D}(w,c) = 1$.

Given signal c, the predicate D specifies the target set (the set $\{w \mid w \in \Gamma \wedge \mathsf{D}(w,c) = 1\}$), or equivalently the revoked set. A subset of revoked users might somehow be able to collude, and combine their keys to recover the content. By combining their codewords, the colluders can generate new words (not necessarily codewords) which potentially can be used to decode signals which were not meant to be decodable by any one of them. We formalize this capability of a collusion by the following notion.

Definition A.2.3 [Feasible Set] A collusion C of users can produce new words from their own codewords. This derivation of new words depends on the structure of the code. The set of words that can be derived from a subset C of codewords is called the **feasible set**, and is denoted by $\mathsf{F}(C;\Gamma)$. When there is no ambiguity, we omit Γ and use $\mathsf{F}(C)$ to denote the feasible set.

We can now define the notion of revocable code which is a basic building block for BE schemes.

Definition A.2.4 [(Efficiently) Revocable Code] Let $\Gamma = \{w^1, \cdots, w^N\}$ be an (ℓ, N)-code. The code Γ is called r**-revocable** if there is a predicate D such that for all $R \subseteq \Gamma$ of size $|R| \leq r$, there exists a signal $c \in \Sigma^\ell$ satisfying the following conditions: $\forall u \in \Gamma - R : \mathsf{D}(u,c) = 1$ and $\forall v \in \mathsf{F}(R) : \mathsf{D}(v,c) = 0$. If, in addition, there is a poly-time algorithm Rev that, given R, outputs a signal c satisfying the above condition, then the code is said to be **efficiently** r**-revocable**.

A.2.2 Conjunction Codes and Broadcast Encryption

In order to clarify the above formalism, we now present a specific family of binary revocable codes called *k-conjunction codes* and a BE scheme based on this family. For any two vectors $u, c \in \{0,1\}^\ell$, let $u \wedge c$ (resp. $u \vee c$) denote the bitwise AND (resp. OR) of two vectors u and c. Let $\mathbf{w}_H(c)$ denote the Hamming weight of any word $c \in \{0,1\}^\ell$.

Let $k \leq \ell$ be a positive integer, a k-conjunction code is a subset of $\{0,1\}^\ell$ with the following associated predicate and feasible set.

Definition A.2.5 [Predicate for k-Conjunction Codes] For any $u, c \in \{0,1\}^\ell$, the predicate $\mathsf{D}_k(u,c) := (\mathbf{w}_H(u \wedge c) \geq k)$ is called the *k-conjunction predicate.*

Definition A.2.6 [Feasible Set for k-Conjunction Codes] For any set $C = \{u^1, \cdots, u^c\} \subseteq \Gamma$. Define

$$\mathsf{F}(C) = F(C;\Gamma) = \left\{ w \in \{0,1\}^\ell \quad s.t \quad \forall i \in [l], w_i \in \{0\} \cup \bigcup_{j=1}^{c} \{u_i^j\} \right\}.$$

Each word $w = (w_i)_{i=1}^\ell \in \{0,1\}^\ell$ can be thought of as a subset of $[\ell]$: the subset of indices i for which $w_i = 1$. Then, the above definitions can be translated as: $\mathsf{D}_k(u,c) = 1$ iff the intersection of u and c has size at least k, and $F(C)$ is the collection of all subsets of the union $\bigcup_{c \in C} c$.

The notion of revocable codes can now be made more precise for this family.

Definition A.2.7 [(Efficiently) Revocable k-Conjunction Code] A binary (ℓ, N)-code Γ is called (r, s)-**Revocable k-Conjunction** if for all $R \subseteq \Gamma$ and $|R| \leq r$, there exists a signal $c \in \{0,1\}^\ell$, with $\mathbf{w}_H(c) \leq s$ satisfying the following conditions: $\forall u \in \Gamma - R : \mathsf{D}_k(u, c) = 1$ and $\forall v \in \mathsf{F}(R) : \mathsf{D}_k(v, c) = 0$. If in addition there is a polynomial time algorithm Rev that computes c given R, then the code is said to be efficiently revocable.

Example A.2.8 [The Complete-subtree scheme as a 1-conjunctive code] The Complete-Subtree scheme [NNL01] can be roughly described as follows. Imagine a full binary tree $\mathcal{T}$ with N nodes. Each user is associated with a leaf of the three. To each tree node there corresponds a distinguished encryption key. A user is given the set of keys corresponding to all the internal nodes from the leaf to the root of the tree. To revoke a subset R of users (i.e. leaves of the tree), we first construct a minimal subtree $\mathcal{T}'$ of $\mathcal{T}$ that spans R and the root. Let K be the set of nodes of $\mathcal{T}$ that are "hanging off" of $\mathcal{T}'$. Then, it is easy to see that each non-revoked user has some key in the set K, and no revoked user has any key in the set K. The system can then use the set K of keys to encrypt the broadcast message, whose length will be proportional to $|K|$, which can be shown to be $O(|R| \log(N/|R|))$.

The above key assignment scheme can be casted in terms of a 1-conjunction code as follows. There is a codeword for each leaf of the N-leaf full binary tree $\mathcal{T}$. The code length is $\ell = 2N - 1$, each position (i.e. coordinate) of the code corresponds to a node of the tree. For each codeword w, there is a 1 in a position if the corresponding node is on the path from w to the root. We will refer to this code the *CS-code*.

Following the results in [NNL01] and our brief description above, the following proposition is straightforward.

Proposition A.2.9 For any $r \in [N]$, the CS-Code is a 1-conjunction $(r, r \log(N/r))$-revocable $(2N - 1, N)$-code.

We now present a broadcast encryption scheme that implements our above predicate for a general k-conjunction binary code. The new trick is to combine the secret sharing with the k-conjunction code such that only legitimate users possesses sufficient shares to be able to decrypt. This is a generalization of the previous schemes [NNL01] where the secret sharing is not involved. In fact, under our construction, the previous schemes [NNL01] correspond to the case $k = 1$ and the 1-out-of-m secret sharing becomes trivial.

Definition A.2.10 [BE from Conjunction Codes] Let us be given a generator of Efficiently (r, s)-Revocable k-Conjunction binary (ℓ, N)-Codes, a secret sharing scheme $\mathcal{SSS}$, and a secure public-key encryption scheme $\mathcal{PKE}$. We build a BE scheme Π that can revoke up to r users in the following way.

- $\mathsf{Setup}(1^\lambda, N)$
  1. Run the code generating algorithm on (N, k, r) to obtain an Efficiently (r, s)-Revocable k-Conjunction (ℓ, N)-Code Γ.
  2. Run $\mathcal{PKE}.\mathsf{Setup}(1^\lambda)$ to get the public parameters param for the encryption scheme;
  3. For $i = 1, \ldots, \ell$, run the key generation algorithm $\mathcal{PKE}.\mathsf{KeyGen}(\mathsf{param})$ to get the pair $(\mathsf{dk}_i, \mathsf{ek}_i)$.
  4. Set $\mathsf{MSK} = (\Gamma, \{\mathsf{dk}_i\})$, and $\mathsf{EK} = \{\mathsf{ek}_i\}$.
  5. For $i = 1, \ldots, N$, the user i is associated with the codeword $w^i \in \Gamma$, we write $w^i = w^i_1 \ldots w^i_\ell$ and set $\mathsf{usk}_i \leftarrow \{\mathsf{dk}_j / w^i_j = 1, j = 1, \ldots, \ell\}$.

- $\mathsf{Encaps}(\mathsf{EK}, R)$:

  1. For a revoked set R of size at most r, since the code Γ is efficiently (r, s)-revocable, one can find out a signal c of weight at most s, such that $\mathsf{D}_k(u, c) = 0$, for any $u \in F(R)$, and $\mathsf{D}(u, c) = 1$ for any $u \in [N] - R$. We denote by $m = \mathbf{w}_H(c)$ this weight;
  2. Denote by $i_1, \ldots, i_m$ the positions of m 1-bits in c, *i.e.,* $c_{i_j} = 1$, for $j = 1, \ldots, m$;
  3. Call $\mathsf{Share}(\kappa, k, m)$, a k-out-of-m secret sharing scheme of a κ-bit secret.
     The $\mathsf{Share}(\kappa, k, m)$ algorithm outputs a secret $K \xleftarrow{\$} \{0,1\}^\kappa$ and m shares $s_1, \ldots s_m$;
  4. Set $e_{i_j} = \mathcal{PKE}.\mathsf{Enc}(\mathsf{pk}_{i_j}, s_j)$, for $j = 1, \ldots, m$.
  5. Output K and $H = (c, (e_{i_j}), j = 1, \ldots, m)$.

- $\mathsf{Decaps}(\mathsf{usk}_j, R, H)$:

  1. If j is in $[N] - R$, then $\mathsf{D}_k(w^j, c) = 1$. This means $\mathbf{w}_H(w^j \wedge c) \geq k$.
  2. Denote by $i_1, \ldots, i_k$ the positions of the first k 1-bit in $w^j \wedge c$.
  3. Compute $s_j = \mathcal{PKE}.\mathsf{Dec}(\mathsf{sk}_{i_j}, e_{i_j})$, for $j = 1, \ldots, k$. With the $\mathsf{Combine}$ algorithm on these k shares, reconstruct K.

The case $k = 1$ is the simplest case: the secret sharing scheme simply consists in choosing a random $K \xleftarrow{\$} \{0,1\}^\kappa$, and then to set $s_i = K$ for all i.

A.3 Traceable Codes

Traitors are users who collude to build (and distribute) a *pirate decoder*. The goal of *traitor tracing schemes* is to allow an authority to trace back, from a pirate decoder, at least one codeword which was used and thus at least one traitor. More precisely, let $T \subset \Sigma^\ell$ denote the set of (codewords of) at most t traitors. From this set of codewords the "pirate" produces a *pirate decoder* $\mathbb{D}$ that efficiently decrypts some broadcast signals $c \in \{0,1\}^\ell$. We view $\mathbb{D}$ as boolean function on predicates $c \in \{0,1\}^\ell$: $\mathbb{D}(c) = 1$ means $\mathbb{D}$ correctly decrypts c, and $\mathbb{D}(c) = 0$ otherwise.

The main task of the traitor tracing scheme is to identify at least one codeword in T (i.e. one traitor) by "examining" the pirate decoder. In this paper we are only concerned with the commonly used "blackbox tracing" model, where the tracer can only query the decoder function $\mathbb{D}$ [CFN94b, KY02c]. In reality, querying $\mathbb{D}$ is roughly equivalent to the act of sending a broadcast signal to the physical decoding device, examining its output (whether it decrypts the content correctly), and then resetting the device. While it is true that some devices may not be stateless (say, some data is stored in a ROM), the blackbox tracing model is still a reasonably practical model.

Our plan is as follows. We first define the notion of a traceable code. The code is designed such that from a word w belonging to the "useful feasible set" of words, the code allows us to trace back at least one traitor. Recall that the feasible set $\mathsf{F}(T)$ is the set of all words that the traitors can derive. (We can roughly think of $\mathsf{F}(T)$ as the set of decryption keys that the traitors can derive from their private keys.) The useful feasible set $\mathsf{UF}(T)$ is a subset of the feasible set, whose meaning is define below. Then, we give a specific family of 1-conjunction traceable codes based on disjunct matrices. Finally, we describe how we might possibly get a hold of a word from the $\mathsf{UF}(T)$. This task is highly dependent on the "anti-tracing" strategy used by the pirate, and thus we will model the anti-tracing strategy with the notion of "qualified signals."

A.3.1 Traceable Codes

The feasible set models the collection of all words which can be "constructed" by the traitors from their codewords. A word in the feasible set $\mathsf{F}(T)$ may or may not be useful for decrypting broadcast contents. We consider a word w from a collusion T *useful* if it satisfies the following condition: for any signal, if every user in T is able to decrypt it, then w should be also able to decrypt it.

Definition A.3.1 [Useful Feasible Set] The **useful feasible set**, denoted by $\mathsf{UF}(T;\Gamma)$ (or $\mathsf{UF}(T)$) from a collusion T is the subset of words w in $\mathsf{F}(T)$ such that, for any signal $c \in \Sigma^\ell$, if $\mathsf{D}(u,c) = 1, \forall u \in T$, then $\mathsf{D}(w,c) = 1$.

It follows from the definition that $T' \subseteq T$ implies $\mathsf{UF}(T') \subseteq \mathsf{UF}(T)$. To see this, consider a word $w \in \mathsf{UF}(T') \subseteq \mathsf{F}(T)$ and an arbitrary signal c. If $\mathsf{D}(u,c) = 1 \forall u \in T$, then $\mathsf{D}(u,c) = 1 \forall u \in T'$, which in turns implies $\mathsf{D}(w,c) = 1$. Hence, $w \in \mathsf{UF}(T)$. Roughly, adding more traitors to a traitor set leads to more useful words. Intuitively, the pirate should at least be able to decrypt signals that all traitors are able of decrypting. Hence, the above definition is in some sense the weakest requirement of being useful.

Useful Feasible Set of 1-conjunction codes To illustrate the concept of useful feasible set, let us characterize the useful feasible sets of 1-conjunction codes. Let Γ be a 1-conjunction (ℓ, N)-code, and $T \subseteq \Gamma$ be an arbitrary set of traitors. Recall that the alphabet is binary in this case. Each word $w = (w_i)_{i=1}^{\ell} \in \{0,1\}^\ell$ can naturally be viewed as a subset of $[\ell]$: the set of all positions $i \in [\ell]$ for which $w_i = 1$. This way, the intersection and union of words are also words in $\{0,1\}^\ell$. Also, we can write $w \subset v$ for two words $w, v \in \{0,1\}^\ell$ without ambiguity.

Proposition A.3.2 Let $T \subseteq \Gamma$ be an arbitrary non-empty set of codewords of a 1-conjunction (ℓ, N)-code. Then,

$$\mathsf{UF}(T) = \{w \in \mathsf{F}(T) \mid \exists u \subset T, u \subseteq w\}.$$

In words, a word w in the feasible set $\mathsf{F}(T)$ is useful if w contains some member of T.

Proof: The fact that $\{w \in \mathsf{F}(T) \mid \exists u \subset T, u \subseteq w\} \subseteq \mathsf{UF}(T)$ is straightforward from definitions. We prove the converse. Assume to the contrary that there is some $w \in \mathsf{UF}(T)$ what does not contain any $u \in T$. Let c be a signal, viewed as a subset of $[\ell]$, constructed by collecting arbitrarily one member from each of the set $u \setminus w$ for each $u \in T$. Then, $\mathsf{D}_1(u,c) = 1$ for all $u \in T$ yet $\mathsf{D}_1(w,c) = 0$ because $w \cap c = \emptyset$. This is a contradiction. ∎

We are now ready to formalize the notion of traceable codes.

Definition A.3.3 [(Efficiently) Traceable Code] An (ℓ, N)-code Γ is **t-traceable** if from any collusion $T \subset \Gamma$ of size at most t, and any word w in the useful feasible set $\mathsf{UF}(T)$, there is an algorithm that on input w outputs a codeword in T. This algorithm is a *tracing algorithm*. If there is a polynomial time tracing algorithm, then the code is said to be *efficiently traceable*.

We have not specified how one might be able to construct a traceable code, efficiently or not, even in the 1-conjunction code case. Given a generic 1-conjunction code, and a word $w \in \mathsf{UF}(T)$, we can construct a set T_w of candidate codewords which are all codewords u which contains w. However, we can not be sure which of the words in T_w belong to the traitor set T. We will enlist the help of disjunct matrices to construct 1-conjunction traceable codes.

A.3.2 Traceable codes from disjunct matrices

The classic combinatorial structure allowing for a very common type of tracing is the so-called *disjunct matrices* [DH00]. Roughly speaking, an r-disjunct matrix is a binary matrix satisfying the following property: given the (boolean) union of at most r unknown columns of the matrix we can identify *all* the unknown columns in time linear in the size of the matrix. Disjunct matrices turn out to be very useful in constructing efficiently traceable 1-conjunction codes, so we formally define and discuss their properties next.

Let $\mathbf{M}$ be an $\ell \times N$ binary matrix. As in the previous section, we will also view each column of $\mathbf{M}$ as a subset of $[\ell]$, In particular, the set of columns of $\mathbf{M}$ is a family of subsets of $[\ell]$. Similarly, the rows of $\mathbf{M}$ form a family of subsets of $[N]$.

Definition A.3.4 [r-Disjunct Matrix] An $\ell \times N$ binary matrix $\mathbf{M}$ is said to be **r-disjunct** if no column (viewed as a subset of $[\ell]$) is contained in the union of any r other columns.

This concept is equivalent to the notion of r-cover-free set family [EFF85]. Disjunct matrices are used to design non-adaptive group tests in the following sense. There is a set of at most r *positive items* in a population of N items. The rest of the items are *negatives*. We must identify the positives using as few non-adaptive “tests" as possible. Each test is a subset of items. A test returns positive iff at least one positive item is contained in the test. In the original group testing application [Dor43], each item is a blood sample, and a test is a pool of blood samples which indicates if any sample in the pool is positive for syphilis. That application explains the “positive" and “negative" terms.

Associate each column of an $\ell \times N$ binary matrix $\mathbf{M}$ with an item. Each row of $\mathbf{M}$ represents a test, which consists of all columns with a 1 on the row. The test outcome vector is precisely the union of the positive columns, where 1 represents positive test outcome and 0 negative. It is well-known [DH00] that, if $\mathbf{M}$ is r-disjunct, then there is a $O(\ell N)$-time algorithm that identifies *all* the positives given the test outcome vector. The algorithm eliminates all items that participate in a negative test. The remaining items are identified as positives. The matrix is r-disjunct if and only if this elimination procedure returns the correct positive set, for an arbitrary set of at most r positives.

The following proposition explains a slightly stronger capability of disjunct matrices. The proposition allows for a disjunct matrix to identify a subset of positives when the outcome vector which is not necessarily an exact union of some positives.

Proposition A.3.5 Let $\mathbf{M}$ be an r-disjunct matrix with dimension $\ell \times N$. Let $\mathbf{M}^{(j)}$ denote the jth column of $\mathbf{M}$. Let T be any (unknown) subset of at most r columns of $\mathbf{M}$. Let $\emptyset \neq S \subset T$ and $w \in \{0,1\}^\ell$ be a vector such that $\bigcup_{j \in S} \mathbf{M}^{(j)} \subseteq w \subseteq \bigcup_{j \in T} \mathbf{M}^{(j)}$. Then, from the vector $\mathbf{w}$ we can identify a set of columns U in time $O(\ell N)$ such that $S \subseteq U \subseteq T$.

Proof: Let $\mathbf{M} = (m_{ij})$, and $w = (w_i)_{i=1}^{\ell}$. Remove any column j such that $m_{ij} = 1$ and $w_i = 0$ for some $i \in [\ell]$. Let U be the set of remaining columns. We claim that $S \subseteq U \subseteq T$. First, consider any column $j \notin T$. By the definition of r-disjunctness, column $\mathbf{M}^{(j)}$ is not contained in the union of columns in T. In particular, $\mathbf{M}^{(j)}$ is not contained in $\mathbf{w}$. Thus, there is some row $i \in [\ell]$ such that $m_{ij} = 1$ and $w_i = 0$. Column j is thus removed by the algorithm. We conclude that $U \subseteq T$. Next, consider any column $j \in S$. Since $\mathbf{M}^{(j)} \subseteq \mathbf{w}$ column j is not removed. Consequently, $S \subseteq U$ as desired. ∎

We can also think of an $\ell \times N$ binary matrix as an (ℓ, N)-code where the codewords are defined to be the columns of the matrix. The following corollary follows from Propositions A.3.5 and A.3.2.

Corollary A.3.6 Let Γ be the collection of columns of an $\ell \times N$ t-disjunct matrix. Then, Γ is an efficiently t-traceable 1-conjunction code.

A.3.3 Black-box Traceability and Decoders' Strategies

So far, we have defined traceable codes and specified how to obtain efficiently traceable 1-conjunction codes from disjunct matrices. Traceable codes allow us to pin-point at least one traitor from any given useful feasible word. So the remaining question is how to get a hold of a useful feasible word.

Our ability to identify a useful feasible word depends intimately on the "anti-tracing" strategy adopted by the pirate (decoder). The anti-tracing strategy attempts to decrypt some signal while ignores others. We call the set of signals that the pirate (decoder) attempts to decrypt the "qualified signals," and this set is modeled with a relation as in the following definition.

Definition A.3.7 [Qualified signals] Let $T \subset \Gamma$ be the set of traitors. Let $\mathcal{Q}$ to be the binary relation $\mathcal{Q}$ over $T \subset \Gamma$ and signals $c \in \{0,1\}^\ell$ defined by $\mathcal{Q}(T,c) = 1$ if the pirate decoder attempts to decrypt broadcast messages associated with the signal c, and $\mathcal{Q}(T,c) = 0$ otherwise.

Recall that $\mathbb{D}$ denotes the pirate decoder, which is a boolean function, where $\mathbb{D}(c) = 1$ iff the decoder successfully decode signal c. For any subset T of words, define $\mathsf{D}(T,c) = \wedge_{u \in T} D(u,c)$. We certainly can not hope to trace pirate decoders that do not decode any signal at all. Our aim is to be able to trace decoders which decode signals that it aims to decode with a non-negligible probability.

Definition A.3.8 [Effective Decoder] A pirate decoder $\mathbb{D}$ is called $(\mathcal{Q}, p)$-effective if for any signal $c \in \Sigma^\ell$, $\Pr[\mathbb{D}(c) = 1 \mid \mathsf{D}(T,c) = 1 \text{ and } \mathcal{Q}(T,c) = 1] \geq p$, where T is the set of codewords (traitors) used to build the decoder.

We will consider black-box tracing procedures, which trace traitors by simple interactions with the pirate decoder. We assume that the decoder can be reset as many time as one wants, thus is it stateless and can be modeled with the function $\mathbb{D}$ as described earlier. However, the decoder can have a specific strategy $\mathcal{Q}$. The only thing that is going for us is that the pirate decoder has to be $(\mathcal{Q}, p)$-effective.

Definition A.3.9 [Black-box (Efficiently) Traceable Code] An (ℓ, N)-code Γ is **$(\mathcal{Q}, p, \delta)$-blackbox t-traceable** if there exists a tracing algorithm Trace such that, for any collusion T of size at most t, and any $(\mathcal{Q}, p)$-effective decoder $\mathbb{D}$, the tracing algorithm $\mathsf{Trace}^{\mathbb{D}}$, with oracle access to the decoder, outputs a traitor in T with probability at least δ. If there is a tracing algorithm which runs in time $\text{poly}(N)$, then the code is said to be *$(\mathcal{Q}, p, \delta)$-blackbox efficiently t-traceable code*. In particular, such an algorithm can only issue $\text{poly}(N)$ queries to the decoder.

To illustrate the above concepts, let us consider several anti-tracing strategies.

Example A.3.10 [Naive Decoder] We first consider the case when the pirate has no strategy at all. The following decoder was called a "perfect decoder" in [BN08b]. A **naive decoder** is a decoder that tries to decrypt any word c with no strategy: $\mathcal{Q}(T,c) = 1$ for any collusion T and any signal c.

This is of course the weakest adversary for a tracing algorithm. For example, the CS-Code [NNL01] described in Example A.2.8 and the disjunct matrix-based code described in Corollary A.3.6 are both black-box efficiently traceable. Assuming the decoder $\mathbb{D}$ is naive, the

tracing algorithm works as follows. It queries the decoder $\mathbb{D}$ with all weight-1 signals $c \in \{0,1\}^\ell$. These are the signals with 1 in some coordinate and 0s elsewhere. By repeating the queries many times, we can amplify the success probability (to be more than δ, see Lemma A.3.11 below) of identifying the set of positions $i \in [\ell]$ at which some codeword in T has a 1. From these positions, we obtain a useful feasible word w for which $u \subseteq w$ for every codeword $u \in T$. This useful feasible word w is precisely the union of the traitor codewords. Hence, the set of all traitors can be traced if the code is based on a t-disjunct matrix.

For the CS-code, we apply the tracing algorithm described in [NNL01]. Recall that in the CS-code codewords are constructed from leaves of a full binary tree with N leaves. Each codeword is of length $\ell = 2N-1$, one position for each node in the tree. A codeword has 1s in the positions corresponding to the nodes from the associated leaf up to the root. Now, from the feasible word w above, we know of all the paths from the traitors to the root and thus we can easily identify the traitors.

The following simple lemma shows us how to amplify the success probability of a tracing procedure. Each query to a pirate decoder is some signal c, and we would like to know whether $\mathbb{D}(c) = 1$ or 0 with high confidence.

Lemma A.3.11 Consider a probabilistic pirate decoder that, for each query c, if it is able to decrypt c then it gives the correct answer $\mathbb{D}(c) = 1$ with probability at least p, and if it cannot decrypt c then it always outputs $\mathbb{D}(c) = 0$. Suppose we want to issue q different queries $c_1, \cdots, c_q$ to this pirate decoder. Then, by repeating *each* query $O\left(\frac{\ln\left(\frac{q}{1-\delta}\right)}{\ln\left(\frac{1}{1-p}\right)}\right)$ times, we will obtain all q correct answers with probability more than δ. (If $p = 1$ then each query is issued once only, for a total of q queries.)

Proof: Suppose we repeat each query m times, and outputs 1 only if at least one of the m copies of the query returns $\mathbb{D}(c) = 1$. Then, we will be wrong with probability at most $(1-p)^m$. Hence, for q different queries $c_1, \cdots, c_q$ we will be wrong on some of them with probability at most $q(1-p)^m$ by the union bound. By picking $m = O\left(\frac{\ln\left(\frac{q}{1-\delta}\right)}{\ln\left(\frac{1}{1-p}\right)}\right)$ we then can ensure that the probability that we are wrong is less than $1-\delta$. ∎

We have been relatively brief in the above description on the naive decoder because the decoder is too simple to spend much space on. In practice, we certainly cannot assume that a decoder will accept to decrypt any signal, even if it could. For example, against the CS-code the pirate decoder can employ the following strategy: it does not decrypt any weight-1 signal where the 1 is in the position of a traitor leaf node. Under this strategy, the CS-Code is not blackbox-traceable, unless with error probability greater the 1/2 because no tracing algorithm can distinguish between a traitor (a leaf node) and its sibling in the full binary tree. Note that the sibling may very well be a non-traitor. The CS-Code cannot deal with this type of pirate's strategy because the code has a rigid structure where each position in the code plays a specific role and corresponds to a subset of users of different sizes. For probabilistic constructions of codes where all the code positions have the same role, the strategy of refusing to decrypt some position has no significant impact on the tracing algorithm. Our probabilistic constructions, described in the next sections, can deal with the above pirate's strategy against CS-Code for that reason.

However, the pirate's strategy can certainly be smarter than rejecting some position(s) of the code. For example, for a probabilistic code where the codewords are chosen independently from the same distribution and all positions play the same role, a non-trivial pirate can estimate the (Hamming) weight of signals used in broadcast encryption and can refuse to decrypt a ciphertext

that correspond to a signal containing too few or too many 1s. This strategy of pirate, called the "weight-limited pirate," is formalized as follow:

Definition A.3.12 [Weight-Limited Decoder] A **Weight-Limited Decoder** is a decoder that only decrypts signals c with Hamming weight in an interval $[a, b]$: $\mathcal{Q}(T, c) := (\mathbf{w}_H(c) \in [a, b])$.

If the tracing algorithm works for a weight-limited decoder with interval $[a, a]$, then it *a fortiori* works for a weight-limited decoder with interval $[u, v]$, for any $u \leq a \leq v$. Therefore, the most interesting case is a singleton interval. We will denote $\mathcal{Q}_a$ the Weight-Limited Strategy $\mathcal{Q}$ for the singleton interval $[a, a]$. The following simple proposition should help clarify the concepts of weight-limited decoder and blackbox efficiently traceable codes.

Proposition A.3.13 The column set of a t-disjunct $\ell \times N$ matrix is a 1-conjunction code which is $(\mathcal{Q}_1, p, \delta)$-blackbox efficiently t-traceable code, where the number of queries the tracer issues to the decoder is $O\left(\ell \frac{\ln\left(\frac{\ell}{1-\delta}\right)}{\ln\left(\frac{1}{1-p}\right)}\right)$.

Proof: We issue ℓ different weight-1 queries (with repetitions) to a $(\mathcal{Q}_1, p)$-effective decoder $\mathbb{D}$. From Lemma A.3.11, the total number of queries issued is $O\left(\ell \frac{\ln\left(\frac{\ell}{1-\delta}\right)}{\ln\left(\frac{1}{1-p}\right)}\right)$. From the results of these queries, we will be able to recover a useful feasible word w which is the boolean union of all the traitors participated in constructing the decoder. From Proposition A.3.5, we can identify all the traitors in $O(\ell N)$ time. ▌

A.4 Trace&Revoke Codes

A.4.1 1-Conjunction Trace&Revoke Codes and (r, s)-disjunct matrices

Sections A.2 and A.3 defined and presented basic examples of revocable and traceable codes. This section defines *trace&revoke codes* which are capable of both revoking users and tracing traitors. Roughly, in a trace and revoke code, one can still trace traitors even after revoking a set of users. The codes we discuss from this section on will be 1-conjunction codes.

As usual, we interchangeably think of $w \in \{0, 1\}^\ell$ as a subset of $[\ell]$. In particular, for any position $j \in [\ell]$, we write $j \in w$ iff $w_j = 1$ and $j \notin w$ otherwise. Let Γ be an (ℓ, N)-code. For any subset $P \subset [\ell]$, let $\Gamma_P \subseteq \Sigma^P$ denote the restrictions of all codewords in Γ onto positions in P; namely $\Gamma_P = \{w|_P \ : \ w \in \Gamma\}$, where $w|_P$ denotes the projection of w onto positions in P.

Let $R \subseteq \Gamma$ be any set of codewords, then $P(R)$ denotes the positions $i \in [\ell]$ for which $w_i = 1$ for some $w \in R$. Let $\bar{P}(R) = [\ell] - R$. Then, define $\Gamma_{\bar{R}} = (\Gamma - R)_{\bar{P}(R)}$. In words, $\Gamma_{\bar{R}}$ is the set of all codewords not in R restricted to the positions not in $P(R)$. We can think of $\Gamma_{\bar{R}}$ as Γ with R being "modded out."

Definition A.4.1 [1-Conjunction Trace&Revoke Code] An (ℓ, N)-code Γ is called an (efficient) 1-conjunction (r, s, t)-trace&revoke if:

1. Γ is a 1-conjunction (efficiently) (r, s)-revocable code;

2. For any subset $R \in \Gamma$ of at most r codewords, the code $\Gamma_{\bar{R}}$ is a (efficiently) t-traceable code.

The main intuition behind the definition should be clear: after revoking users in R, the remaining codewords form a t-traceable code so that we can trace after revoke. But how do we construct (r, s, t)-trace&revoke codes? We will impose an extra constraint on disjunct matrices to add revoking capability to disjunct matrices.

In Section A.3.2, we motivated the use of disjunct matrix for tracing. However, general disjunct matrices do not necessarily have efficient revocation capability. For example, the identity matrix is r-disjunct for any r, but it would represent a horrible 1-conjunction revoke code because a broadcast signal must have weight $\Omega(N)$ leading to large broadcast messages.

This section introduces "(r, s)-disjunct matrices" that retain the traceability of disjunct matrices yet also attain efficient revocation capability. Constructions of good (r, s)-disjunct matrices are presented in Section A.5.

Let $R \subseteq [N]$ be a non-empty subset of columns of a binary matrix $\mathbf{M}$ with ℓ rows and N columns. A row set $I \subseteq [\ell]$ is said to *eliminate* R if the union of the rows in I is precisely $[N] - R$.

Definition A.4.2 [(r, s)-disjunct] An $\ell \times N$ matrix $\mathbf{M}$ is said to be (r, s)-disjunct if it satisfies the following property. Given an arbitrary set R of up to r columns of $\mathbf{M}$, there is a set $I \subseteq [\ell]$ of at most s rows which eliminates R.

Finally, we show how the new notion of disjunct matrices leads to 1-conjunction trace and revoke codes.

Theorem A.4.3 For any $(r + t, s)$-disjunct matrix $\mathbf{M}$ with ℓ rows and N columns. Then, the set of columns of $\mathbf{M}$ forms a 1-conjunction (r, s, t)-trace&revoke code.

Proof: Let Γ denote the set of columns of $\mathbf{M}$. Let R be any subset of columns of $\mathbf{M}$ with size $|R| \leq r < r + t$. Then, there is a subset $I \subseteq [\ell]$ of at most s rows that eliminates R because $\mathbf{M}$ is $(r + t, s)$-disjunct. Let $c \in \{0, 1\}^\ell$ be the characteristic vector of I, i.e. $c_i = 1$ for $i \in I$ and $c_i = 0$ for $i \in [\ell] - I$. Then, c is a signal for which $\mathsf{D}_1(c, w) = 0$ for all $w \in R$ and $\mathsf{D}_1(c, w) = 1$ for all $w \notin R$. Furthermore, the Hamming weight of c is at most s. Consequently, Γ is an (r, s)-revocable code.

Next, we show that for any set R of at most r columns of $\mathbf{M}$, $\Gamma_{\bar{R}}$ is t-traceable. Let $\mathbf{M}_{\bar{R}}$ denote the matrix obtained from $\mathbf{M}$ by removing all columns in R and all rows in $P(R)$. Then, by Corollary A.3.6 it is sufficient to show that $\mathbf{M}_{\bar{R}}$ is t-disjunct. Let T be an arbitrary set of at most t columns of $\mathbf{M}_{\bar{R}}$, which by extension is also a set of at most t columns of $\mathbf{M}$. Let w be a column of $\mathbf{M}_{\bar{R}}$ not in T. If w – as a column of $\mathbf{M}_{\bar{R}}$ is contained in the union of columns in T, then w – as a column of $\mathbf{M}$ – is contained in the union of all columns in $R \cup T$ because all positions in $P(R)$ are covered by columns in R. This means $\mathbf{M}$ is not $(r + t)$-disjunct; and in particular $\mathbf{M}$ is not $(r + t, s)$-disjunct, a contradiction! ∎

A.4.2 Trace&revoke schemes from 1-conjunction blackbox trace&revoke codes

Finally, we incorporate the notion of blackbox tracing and pirate strategy $\mathcal{Q}$ into the code definition.

Definition A.4.4 [Black-box 1-Conjunction Trace&Revoke Code] An (ℓ, N)-code Γ is $(r, s, \mathcal{Q}, p, \delta, t)$-*blackbox (efficient) trace&revoke* if

1. Γ is a 1-conjunction (efficient) (r, s)-revocable code, and

2. For any set R of at most r codewords, the code $\Gamma_{\bar{R}}$ is a 1-conjunction $(\mathcal{Q}, p, \delta)$-blackbox (efficiently) t-traceable code.

Given such a blackbox trace and revoke code, we can transform it into a trace and revoke system in a similar fashion to the the one from a revocable code to a broadcast encryption in Definition A.2.10. The details (definition of a trace&revoke scheme and the transformation) can be found in Appendix A.7 (Definitions A.7.2 and A.7.6).

A.5 Constructions of black-box Trace and Revoke with (r, s)-disjunct matrices

We have shown that (r, s)-disjunct matrices give rise to trace and revoke codes. In this section we first show how to construct (r, s)-disjunct matrices probabilistically. Then, we use these matrices to construct blackbox trace and revoke codes.

A.5.1 Constructions of 1-Conjunction (r, s, t)-trace&revoke codes

A distribution of binary matrices.

Let N, b, n be arbitrary positive integers. Let $\ell = nb$, and $\mathcal{M}(N, b, n)$ denote the distribution of $\ell \times N$ binary matrices generated as follows. Partition the set $[\ell]$ into n *parts*, each part has b "bins." The parts are $P_1 = \{1, \cdots, b\}$, $\cdots$, $P_n = \{(n-1)b + 1, \cdots, nb\}$.

To generate a matrix $\mathbf{M} \in \mathcal{M}(N, b, n)$ with $\ell = bn$ rows and N columns, we generate columns of $\mathbf{M}$ independently in the following way. Each column of $\mathbf{M}$, viewed as a subset of $[\ell]$, is chosen by picking uniformly (with probability $1/b$) exactly one bin from each part. In particular, each column of $\mathbf{M}$ has exactly n elements.

We can think of each column as a "ball," and each part is a collection of b bins. The distribution $\mathcal{M}(N, b, n)$ is defined by throwing N balls to b bins belonging to a part, and repeat that experiment n times, one for each part. This type of matrix distribution is used in constructing compressed sensing matrices. The resulting random matrix can also be thought of as the incidence matrix of concatenating a random code of length n with the identity code [NPR12].

Construction of (r, s)-disjunct matrices.

Given two integer parameters $1 \leq r < N$, our goal is to (randomly) construct a $\ell \times N$ binary matrix $\mathbf{M}$ which is (r, s)-disjunct with s as small as possible. The idea is to choose a matrix $\mathbf{M}$ at random from $\mathcal{M}(N, b, n)$ with suitably chosen parameters n and b, and show that $\mathbf{M}$ is (r, s)-disjunct with high probability.

Theorem A.5.1 Let $1 \leq r < N$ be given positive integers. Let z, b, n be positive integers such that $r < b$ and $z \mid n$. Let $\mathbf{M}$ be a matrix chosen from the distribution $\mathcal{M}(N, b, n)$. (Recall that $\mathbf{M}$ has $\ell = nb$ rows and N columns. And, each column of $\mathbf{M}$ has weight exactly n.) Then, with probability at least

$$1 - \left(\frac{Ne}{r}\right)^r N^{n/z}(r/b)^n$$

the matrix $\mathbf{M}$ satisfies both of the following conditions:

(i) let R be an arbitrary set of at most r columns of $\mathbf{M}$. Then there is a set I of rows which eliminates R, where $|I| \leq zb$ and $I \subseteq \{zb(i-1) + 1, \cdots, zbi\}$ for some $i \in \{1, \cdots, n/z\}$. In particular, $\mathbf{M}$ is (r, zb)-disjunct.

(ii) finding I given R takes time at most $O(\ell N)$.

Proof: Recall that $[\ell]$ is partitioned into n parts, each part has b "bins": $P_j = \{(j-1)b + 1, \cdots, jb\}$, $j \in [n]$. Fix a set R of at most r columns of $\mathbf{M}$. Let $\overline{R} \subseteq [\ell]$ be the union of columns in R. Define

$$I = P_1 \cup \cdots \cup P_z \setminus \overline{R}.$$

In other words, I is the set of bins in the first z parts which contain none of the columns in R. In each of the first z parts, the columns in R can be in at most r bins. Hence, $z(b-r) \leq |I| \leq zb$.

We bound the probability that I does not eliminate R, which happens if some column in $[N]-R$ belongs to no bin in I. A fixed column in $[N]-R$ belongs to no bin in I with probability at most $(r/b)^z$. Hence, by the union bound the probability that some column in $[N]-R$ belongs to no bin in I is at most $(N-r)(r/b)^z < N(r/b)^z$. In other words, I does not eliminate R with probability at most $N(r/b)^z$.

Now, if we define I to be

$$I = P_{z+1} \cup \cdots \cup P_{2z} \setminus \overline{R},$$

then by the same reasoning the probability that I does not eliminate R is also at most $N(r/b)^z$. The same conclusion holds for the next group of z parts, and so forth. Since n parts can be partitioned in to n/z groups of z parts each, and they are all independent, the probability that R cannot be eliminated by any one of these I is at most $(N(r/b)^z)^{n/z} = N^{n/z}(r/b)^n$.

Finally, by the union bound over all choices of R (including $R = \emptyset$) we conclude that $\mathbf{M}$ does not satisfy property (i) with probability at most

$$\sum_{j=0}^{r} \binom{N}{j} N^{n/z}(r/b)^n \leq \left(\frac{Ne}{r}\right)^r N^{n/z}(r/b)^n.$$

Property (ii) follows straightforwardly from the above analysis, because we can simply check each block of z consecutive parts, one by one, and verify if I satisfies the desired property. ∎

Corollary A.5.2 [Concrete parameters for an (r, s, t)-trace&revoke code] For any $1 \leq r+t < N$, there exists an efficient 1-conjunction (r, s, t)-trace&revoke (ℓ, N)-code of length $\ell = 2(2(r+t)^2 + r + t)(\log_2 N + 1)$, where $s = (4r + 4t + 2)(\log_2 N + 1)$.

The above corollary was obtain from Theorem A.4.3 and Theorem A.5.1 by setting $n = (r+t)\log_2(N^2 e/(r+t))$, $b = 2(r+t)+1$, and $z = n/(r+t)$. However, the corollary only shows the existence of such codes, it does not give an efficient strategy for constructing such codes. There are several directions one can take.

- Deterministically, in exponential time we can easily construct a matrix satisfying all conditions in the theorem with the parameters in the corollary because the theorem shows that such a matrix exists.
- Probabilistically, by slightly worsen some parameters, the theorem implies that we can construct probabilistically a good (r, s)-disjunct matrix with overwhelmingly large probability. For example, by setting $n = (r+t)\log_2(N^2 e/(r+t))$, $b = 4(r+t)$, and $z = n/(r+t)$, the probability that a random matrix from $\mathcal{M}(N, b, n)$ satisfies all properties in the theorem is at least $1 - \left(\frac{(r+t)}{N^2 e}\right)^{r+t}$. In this case, we obtain with extremely high probability an efficient 1-conjunction $(r+t, 8(r+t)(\log_2 N + 1))$-revocable code of size N and length $\ell = 8(r+t)^2(\log_2 N + 1)$.

- If desired, we can obtain a deterministic construction with a Las Vegas algorithm by repeating the above experiment independently multiple times. In each iteration, we can check in time $O(N^{O((r+t))})$ whether the randomly selected matrix satisfies the properties.

A.5.2 Combinatorial Group Testing with Prescribed-Weight Tests

From Corollary A.5.2, we knows how to construct a trace and revoke code where after revoking at most r users R, we can identify at least one out of t traitors if we have access to a useful feasible word of the code $\Gamma_{\bar{R}}$. The identification of a useful feasible word, of course, depends intimately on the anti-tracing strategy of the pirate decoder. In this section, we develop the *shadow group testing* technique for identifying a useful feasible word when the pirate decoder is a limited-weight decoder (recall Definition A.3.12). As we have observed earlier, it is sufficient to deal with constant-weight decoder strategy $\mathcal{Q}_a$. To describe the shadow group testing technique, we first derive some results regarding group testing where all pools have the same given size.

The non-adaptive case. Given positive integers r, z, N where $r + z \leq N$, the following gives upper and lower bounds the optimal number of non-adaptive tests for identifying $\leq r$ unknown items from the population of N items, where each test must have weight exactly z. Furthermore, the proof also presents two methods for constructing the tests achieving the upper-bound, one deterministic and the other randomized.

Theorem A.5.3 Let $\mathbf{M}$ be an $\ell \times N$ matrix which is r-disjunct with a uniform row weight of z. Then,

$$\ell \geq \frac{N-r}{z} \cdot e^{zr/N}. \tag{A.1}$$

Given parameters $r + z \leq N$, there is a deterministic algorithm which constructs a $\ell \times N$ matrix which is r-disjunct with a uniform row weight of z, where

$$\ell = \frac{N-r}{z} e^{\frac{rz}{N-r-z+1}} \left(1 + r + \ln z + r \ln\left(\frac{(N-z)}{r}\right)\right). \tag{A.2}$$

Furthermore, by choosing weight-z rows uniformly at random, we can also construct such a matrix with success probability $\geq 1 - \epsilon$, for any given $\epsilon \in (0, 1)$, where

$$\ell = O\left(\frac{N}{z} r \ln\left(\frac{N}{\epsilon r}\right) e^{\frac{rz}{N-r-z+1}}\right). \tag{A.3}$$

Proof: Let X be the set of all pairs (j, A) where $A \subset [N]$ is an r-subset of $[N]$, and $j \in [N] - A$. Note that $|X| = \binom{N}{r}(N-r) = N\binom{N-1}{r}$.

To prove the lower bound (A.1), fix an $\ell \times N$ r-disjunct matrix $\mathbf{M} = (m_{ij})$ with constant row weight z. A row i of $\mathbf{M}$ is said to *mask* a member $(j, A) \in X$ if row i "intersects" column j (i.e. $m_{ij} = 1$) and does not intersect *any* column $j' \in A$ (i.e. $m_{ij'} = 0, \forall j' \in A$). In order for $\mathbf{M} = (m_{ij})$ to be r-disjunct, every member of X must be masked by some row. A weight-z row masks exactly $z\binom{N-z}{r}$ members of X. Thus,

$$\begin{aligned} \ell &\geq \frac{(N-r)\binom{N}{r}}{z\binom{N-z}{r}} = \frac{N-r}{z} \cdot \frac{N}{N-z} \cdot \frac{N-1}{N-z-1} \cdots \frac{N-r+1}{N-z-r+1} \\ &> \frac{N-r}{z} \cdot \left(\frac{N}{N-z}\right)^r = \frac{N-r}{z} \cdot \frac{1}{(1-z/N)^r} > \frac{N-r}{z} \cdot e^{zr/N} \end{aligned}$$

We next derive an upperbound, we present two methods of constructing constant row-weight disjunct matrices, one deterministic and one probabilistic.

The deterministic construction works by casting the problem as an instance of the SET COVER problem and using the well-known greedy algorithm for SET COVER. The construction problem can be viewed as a set cover instance as follows. The universe to be covered is X. Each "set" is represented by a weight-z row $\mathbf{s}$. The elements in the universe that belong to the set $\mathbf{s}$ are precisely the members of X which $\mathbf{s}$ masks. Thus, each "set" $\mathbf{s}$ contains exactly $z\binom{N-z}{r}$ elements. A member $(j, A) \in X$ is covered by exactly $\binom{N-r-1}{z-1}$ sets. A classic result by Lovasz [Lov75] (and independently by Chvatal [Chv79]) implies that the greedy algorithm finds a set cover for X of size at most

$$\begin{aligned}
\ell &\leq \frac{\binom{N}{z}}{\binom{N-r-1}{z-1}}\left(1+\ln\left(z\binom{N-z}{r}\right)\right)\\
&= \frac{N-r}{z}\cdot\frac{N}{N-r}\cdot\frac{N-1}{N-r-1}\cdots\frac{N-z+1}{N-r-z+1}\left(1+\ln\left(z\binom{N-z}{r}\right)\right)\\
&< \frac{N-r}{z}\left(\frac{N-z+1}{N-r-z+1}\right)^{z}\left(1+\ln z+r\ln\left(\frac{e(N-z)}{r}\right)\right)\\
&= \frac{N-r}{z}\left(1+\frac{r}{N-r-z+1}\right)^{z}\left(1+r+\ln z+r\ln\left(\frac{(N-z)}{r}\right)\right)\\
&< \frac{N-r}{z}e^{\frac{rz}{N-r-z+1}}\left(1+r+\ln z+r\ln\left(\frac{(N-z)}{r}\right)\right).
\end{aligned}$$

This fact can also be seen from the *dual-fitting* analysis of the greedy algorithm for SET COVER. This set cover is exactly the set of rows of the r-disjunct matrix we are looking for. The final expression might seem a little unwieldy. Note, however, that compared to the lower bound (A.1), we are only off by a factor of about $O(r\ln(N/r))$. For most meaningful ranges of z and r, the factor $\left(1+r+\ln z+r\ln\left(\frac{(N-z)}{r}\right)\right)$ can safely be thought of as $O(r\ln(N/r))$. Last but not least, if $rz = O(N)$ then $e^{\frac{rz}{N-r-z+1}} = O(1)$ and the number of rows l is not exponential. Also, when $rz = \Theta(N)$ the overall cost is $l = O(r^2\log(N/r))$, matching the best known bound for disjunct matrices. This optimality only applies when we are free to choose z in terms of N and r; in particular, when we have this freedom we will pick $z = \Theta(N/r)$.

The probabilistic construction works as follows. We think of members of X as "coupons" and the weight-z row vectors as boxes of coupons. Each box has precisely $z\binom{N-z}{r}$ different coupons in it. We want to collect as few boxes as possible to have a complete coupon collection. Let's pick the boxes uniformly at random, one by one. The probability that a given coupon is chosen in each round is $\frac{\binom{N-r-1}{z-1}}{\binom{N}{z}}$. Hence, by the union bound, after ℓ rounds the probability that at least one coupon is not collected is at most

$$|X|\left(1-\frac{\binom{N-r-1}{z-1}}{\binom{N}{z}}\right)^{\ell} \leq N\binom{N-1}{r}e^{-\ell\frac{\binom{N-r-1}{z-1}}{\binom{N}{z}}}.$$

This is an upper bound on our failure probability. If we want a guarantee of at most $\epsilon < 1$ failure probability, then we can simply choose ℓ such that

$$N\binom{N-1}{r}e^{-t\frac{\binom{N-r-1}{z-1}}{\binom{N}{z}}} \leq \epsilon.$$

Similar to the above analysis,

$$\ell = O\left(\frac{N}{z} r \ln\left(\frac{N}{\epsilon r}\right) e^{\frac{rz}{N-r-z+1}}\right)$$

is sufficient. ▌

The adaptive case. One might hope that adaptive tests may help overcome the $\tilde{\Omega}(e^{rz/N})$-barrier. Unfortunately, such is not the case.

Theorem A.5.4 Any adaptive group testing strategies for a population of N items with less than r positives in which each test has weight z must use $\Omega(e^{rz/N})$ tests. Furthermore, there exists a randomized testing strategy which uses, in expectation, $e^{rz/(N-r)} + N - z$ tests.

Proof: We will show that if there are $\ell < \frac{1}{2}e^{rz/N}$ adaptive tests, then there are at least two different sets of positive items which give identical (adaptive) test results and thus the adaptive tests cannot distinguish between these two sets of positive items.

Each adaptive test is a z-subset $F_i \subseteq [N]$. Consider any sequence of ℓ adaptive tests $F_1, \cdots, F_\ell$, where $\ell < \frac{1}{2}e^{rz/N}$. We will show that there are two different r-sets of items S and T such that S intersects all of the F_i and T also intersects all of the F_i. Thus, if S were the set of positives then all of the tests return positive. And, if T were the set of positives then all of the tests also return positive. Consequently, ℓ tests are not sufficient to distinguish between S and T. (Remark: to be a little more rigorous, we could model the adaptive test strategy as a binary tree, where each node represents a z-subset, and the two children of a node correspond to whether a test turns positive or not. Here, the sequence $F_1, \cdots, F_\ell$ corresponds to the all-positive branch of the tree.)

We use the probabilistic method. We pick a subset S of size r of $[N]$ uniformly at random, and show that the probability that S intersects all of the F_i is at least $2/\binom{N}{z}$, which would establish the claim.

For a fixed F_i, we have

$$\Pr[S \cap F_i = \emptyset] = \frac{\binom{N-z}{r}}{\binom{N}{r}} = \frac{N-z}{N} \cdot \frac{N-z-1}{N-1} \cdots \frac{N-z-r+1}{N-r+1} \leq \left(\frac{N-z}{N}\right)^r < e^{-zr/N}.$$

By the union bound,

$$\Pr\left[S \cap F_i = \emptyset, \text{ for some } i\right] < \ell \cdot e^{-rz/N} \leq \frac{1}{2}.$$

Thus,

$$\Pr\left[S \cap F_i \neq \emptyset, \text{ for all } i\right] > \frac{1}{2} \geq \frac{2}{\binom{N}{r}}.$$

For the upper-bound, we can pick a random test T (of size z) until the test returns negative. Let S be the set of positive items, then

$$\Pr[T \text{ returns negative}] = \Pr[S \cap T = \emptyset] \geq \frac{\binom{N-r}{z}}{\binom{N}{z}}.$$

Hence, the expected number of random tests we need is at most

$$\frac{\binom{N}{z}}{\binom{N-r}{z}} = \frac{N}{N-r} \cdots \frac{N-z+1}{N-r-z+1} \leq \left(1 + \frac{r}{N-r}\right)^z \leq e^{zr/(N-r)}.$$

After the negative test T is obtained, we can then construct new tests by replacing a negative item in T with an item outside of it one at a time until we found all of the positive items. ▌

A.5.3 The shadow group testing technique and 1-Conjunction $\mathcal{Q}_a$-blackbox trace&revoke codes

When the pirate decoder only answers queries with weight a (this is the meaning of the $\mathcal{Q}_a$ qualified signals), we will only use weight-a queries to trace because other signals do not give reliable answers. The following theorem explains how we can trace such pirate decoders.

Theorem A.5.5 [$\mathcal{Q}_a$ tracing with shadow group tests] Let $1 \leq t < N$ and ℓ be integers, and let Γ be the 1-conjunction code formed by the columns of a t-disjunct matrix with ℓ rows and N columns. Additionally, suppose the union of any t codewords has weight at most D. Then, the code Γ is also a $(\mathcal{Q}_a, 1, 1, t)$-traceable code where the number of queries the tracer issues to the decoder is at most $O\left(\frac{\ell}{a} D \log\left(\frac{\ell}{t}\right) e^{\frac{Da}{\ell - D - a + 1}}\right)$.

Proof: Thanks to Proposition A.3.5, instead of identifying a traitor, we can just identify a vector $w \in \{0,1\}^\ell$ that is contained in the union of all traitors' codewords and that contains at least one traitor's codeword. Such a vector is in the useful feasible set (see Proposition A.3.2) of the code.

Identifying $w = (w_1, \cdots, w_\ell)$ is equivalent to identifying all the coordinates i of w for which $w_i = 1$. Thus, there is a subset $U \subseteq [\ell]$ of at most D unknown coordinates that we want to identify. We need to query the pirate decoder with weight-a signals c to identify U. Each query c is the characteristic vector of a subset of size a of $[\ell]$. So we think of each query as an a-subset A of $[\ell]$. The decoder is able to decrypt query A if and only if there is at least one traitor whose codeword intersects A. In other words, each query A is a *group test* for the "positives" U in the population $[\ell]$. We thus have a group testing problem "inside" another group testing problem. We refer to the "inner" group tests as the *shadow tests*, because they are not used to identify the traitors directly; rather, they are used to identify the shadow U of the traitors.

Finally, we directly apply Theorem A.5.3 and use the non-adaptive (but deterministic) construction in the theorem (equation (A.2)) to attain the desired number of queries. ▌

Now, we incorporate the shadow group testing technique to construct blackbox trace and revoke codes. From Corollary A.5.2 and the discussion after that we know how to construct a 1-conjunction (r, s, t)-trace&revoke code Γ. This code can easily be turned into a trace&revoke system as described in Section A.4.2. Let R be any set of at most r traitors, we know that $\Gamma_{\bar{R}}$ is a 1-conjunction t-traceable code. In fact, from the construction of the $(r + t, s)$-disjunct matrix that leads to the (r, s, t)-trace&revoke code, we know that the Hamming weight of each codeword is $n = O((r + t) \log(N/(r + t)))$; hence, the Hamming weight of the union of at most t columns is $D = tn = O(t(r + t) \log(N/(r + t)))$. Now, if the pirate decoder applies the $\mathcal{Q}_a$ anti-tracing strategy, then we will only trace with weight-a signals using the shadow group test technique from Theorem A.5.5. The number of queries the tracer needs is going to be $q = O\left(\frac{\ell}{a} D \log\left(\frac{\ell}{t}\right) e^{\frac{Da}{\ell - D - a + 1}}\right)$.

The question is, which value of weight a makes the most sense to the pirate? The answer to this question depends on how we broadcast after revoking the users in R. Looking back at the proof of Theorem A.5.1, we chose a subset I that eliminates R where the size of I is at most bz. In fact, once I eliminates R we can add more elements to I so that $|I| = bz$. We use the set I to construct a broadcast signal c; I is the support set of c. Hence, all broadcast signals will have weight $s = bz$ (before or after revoking R). In other words, it will only make sense for the pirate decoder with the anti-tracing strategy $\mathcal{Q}_a$ to set $a = s = \Theta((r + t) \log(N/(r + t)))$. Also recall that $\ell = \Theta((r + t)^2 \log(N/(r + t)))$. Hence, $\ell/a = O(r + t)$, and $e^{\frac{Da}{\ell - D - a + 1}} = O((N/(r + t))^{O(t)})$. Combining with Lemma A.3.11, we obtain the following corollary.

Corollary A.5.6 For any $1 \leq r+t < N$, there exists an $(r, s, \mathcal{Q}_s, p, \delta, t)$-blackbox trace&revoke code Γ, where $s = O((r+t)\log N)$ is the weight of broadcast signals, and the number of queries issued to the pirate decoder is $O\left(q \cdot \frac{\ln\left(\frac{q}{1-\delta}\right)}{\ln\left(\frac{1}{1-p}\right)}\right)$, where

$$q = O\left(t(t+r)^2 \log(N/(r+t)) \log\left(\frac{(r+t)^2 \log((N/(r+t))}{t}\right) \cdot \left(\frac{N}{r+t}\right)^{O(t)}\right).$$

A.5.4 Toward Traceability Against Arbitrary Pirate Decoders

We now discuss the case of arbitrary pirate decoders. Our goal is to be able to deal with any qualified predicate $\mathcal{Q}$. We propose a slightly modified version of the main scheme described in Corollary A.5.6 for this aim. However, we have to introduce an additional assumption: the pirate decoder should decrypt (with non-negligible probability) broadcast signals that correspond to the case of non-revocation, i.e., when $R = \emptyset$. The tracing process in our Trace&Revoke scheme is thus not in the standard model but it seems still practical as it requires the pirate decoder to be able to decrypt ciphertexts in the most usual case (at least in some applications) where there is no detected malicious users in the system. Anyway, our scheme satisfies both the definition of a traitor tracing scheme (in which the revoked set is by definition an empty set) for arbitrary pirate decoders and the definition of a revoke system (in which the scheme can revoke users and does not require a tracing procedure).

As shown in Theorem A.5.3, we can ask random queries to the pirate decoder, the number of queries is asymptotically similar to the deterministic case. Each random query consists of a set of $\beta 4r$ rows randomly chosen from ℓ rows of the matrix $\mathbf{M}$. In our code based trace&revoke scheme (Definition A.7.6 in Appendix A.7), the number of chosen rows in a ciphertext (the weight of the signal c) varies from $4r$ (when there is no revoked users, $4r$ rows from any block cover all users) to $8r\log(N/r)$ (when there are r revoked users).

Assume the pirate knows our strategy of choosing rows in the encapsulation and *if* it can detect whether a query – though corresponding to an encapsulation for some R – cannot be an output of the $\mathsf{Encaps}(\mathsf{EK}, R)$, then it will not decrypt. The key point here is to issue (random) tracing queries so that it is computationally hard for the pirate to distinguish between a given tracing query and an output of $\mathsf{Encaps}(\mathsf{EK}, \emptyset)$ (i.e. broadcast signals in the non-revoke mode).

We now described the modified scheme. To impose the computational hardness on the pirate, we first permute randomly the rows of the matrix $\mathbf{M}$ to obtain a matrix $\mathbf{M}^*$. We will use the matrix $\mathbf{M}^*$ as the code, instead of $\mathbf{M}$, as the $(\ell, N)-$code in our encryption in Definition A.7.6 (appendix A.7). We also slightly change the $\mathsf{Encaps}(\mathsf{EK}, R)$ in Definition A.7.6 when there is no revoked users, *i.e.,* $R = \emptyset$. In this case, $4r$ rows from a block cover all users and the $\mathsf{Encaps}(\mathsf{EK}, \emptyset)$ procedure will normally use these rows for encryption. However, in this case, we will add $(\beta - 1)4r$ more random rows and let $\mathsf{Encaps}(\mathsf{EK}, \emptyset)$ procedure uses the total $\beta 4r$ rows for encryption. Note that the $(\beta - 1)4r$ additional rows have no impact on the validity of the broadcast because the original $4r$ rows already covered all users.

We argue that no pirate decoder can distinguish a broadcast signal and a signal in the tracing procedure. In fact, in order to distinguish between a broadcast signal constructed this way and a random set of $\beta 4r$ rows, the pirate must be able to tell whether a given set A of $\beta 4r$ rows contains a subset $B \subseteq A$ of $4r$ rows which might come from the same row block. We next argue that, even if the pirate has as many as $b = 4r = \omega(t)$ codewords, it is computationally hard to detect whether a given set of $\beta 4r$ rows contains $4r$ rows from the same block.

Consider the case where the matrix $\mathbf{M}^*$ is not public and the pirate decoder is resettable, thus can be reset to the initial state after each query. The pirate has a set T of $b = 4r$ codewords (each codeword is a column of the matrix $\mathbf{M}^* = (m_{ij})$). Each row i in the set A of βb rows of

the random query corresponds to a subset of the set T: the subset of traitors j in T for which $m_{ij} = 1$. The problem of detecting whether there are b rows of A belonging to the same block thus becomes precisely an instance of the **NP**-hard EXACT COVER problem [GJ79] which is like SET COVER but each element in the universe is to be covered exactly once. To cover b columns we do not need more than b rows, and hence a solution to the EXACT COVER problem with additional empty rows will be a solution to the pirate's problem. Conversely, a solution to the pirate's problem obviously is a solution to the EXACT COVER instance. It is not hard to derive a reduction from 3-SAT to EXACT COVER. (See, e.g., [BM08].) Furthermore, if we start from 3-SAT-5 (known to be **NP**-hard [Fei98]) where each clause has at most 3 literals and each literals appear in at most 5 clauses, then we obtain instances of EXACT COVER where the number of sets is bounded by a constant (β in our case) times the size of the universe. Therefore, there is no known algorithm in polynomial time of $b = 4r$ that can solve this problem, even when β is a constant (about 15 or so).

We should however notice that our assumption requires in fact the average case hardness. On the other hand, we also notice that the pirate possesses only t users' keys but not b keys, and $b = 4r = \omega(t)$. Therefore, we believe that our assumption is reasonable.

A.5.5 Trace&Revoke in the Information-Theoretic Limit

In the tracing procedure of our scheme, as well as in the tracing procedures of almost all known schemes in the literature, we rely on the fact that the pirate accepts to decrypt the tracing queries, as it is hard to distinguish a query in tracing mode and a normal ciphertext. A natural question is, without any computational assumption about the pirate, can we still trace?

To be concrete, assume the pirate can always verify whether a query comes from $\mathsf{Encaps}(\mathsf{EK}, R)$, for some R, and that the pirate decoder only decrypts if and only if this is indeed the case. We can prove that, for this powerful pirate, the number of queries can no longer be $\mathrm{poly}(N)$ for most reasonable values of r and t. In particular, with $q \leq \binom{N}{t}(1 - \frac{t}{N})/\binom{r}{t}$ queries it is not possible to always identify correctly at least one traitor. Moreover, in this model of pirate, we present a randomized tracing strategy that matches this bound in expectation. Note that our lower-bounds apply to any Trace&Revoke systems given the powerful pirate assumption.

Considering a trace&revoke schemes against the pirates who decrypt, if possible, a ciphertext if and only if this ciphertext is an output of the algorithm $\mathsf{Encaps}(\mathsf{EK}, R)$ for some R. We then have a lower bound of the number of queries for the tracing algorithm.

Theorem A.5.7 Let $t \leq r < N$ such that $t + r < N$. If we pose at most $q \leq \left(\binom{N}{t} - 2\right)/\binom{r}{t}$ queries, then it is not possible to always identify correctly the entire traitor set T. If we pose at most $q \leq \frac{\binom{N}{t}}{\binom{r}{t}}(1 - \frac{t}{N})$ queries, then it is not possible to always identify correctly at least one traitor.

Proof: Let $c = \mathsf{Encaps}(\mathsf{EK}, R)$ for some R. Let T be the set of traitors. If $T \subseteq R$ then no traitor can decode c. If $T \setminus R \neq \emptyset$, then some traitor will be able to decode. Thus, the blackbox query can be represented by subsets $R \subseteq [N], |R| \leq r$. A query returns YES if $T \setminus R \neq \emptyset$ and NO otherwise.

Let us first consider the case when we want to identify all traitors in T, $|T| \leq t$. Suppose we pose q adaptive (!) black-box queries $F_1, \cdots, F_q$. Recall that each query can be represented by a set $F_i \subseteq [N]$, where $|F_i| \leq r$. We will prove that if $q \leq \frac{\binom{N}{t}-2}{\binom{r}{t}}$, then there are two distinct traitor sets T_1 and T_2 such that the series of queries $F_1, \cdots, F_q$ all return YES, and thus no algorithm can distinguish between T_1 and T_2.

To this end, fix *arbitrary* $F_1, \cdots, F_q$ and pick a set $T \subseteq [N]$ of size t uniformly at random. For each $i \in [q]$, $\Pr[T \subset F_i] \leq \frac{\binom{r}{t}}{\binom{N}{t}}$. Thus, by the union bound $\Pr_T[\text{some query } F_i \text{ returns NO}] \leq q\frac{\binom{r}{t}}{\binom{N}{t}}$. Put it another way,

$$\Pr_T[\text{all queries } F_i \text{ return YES}] \geq 1 - q\frac{\binom{r}{t}}{\binom{N}{t}} \geq \frac{2}{\binom{N}{t}}.$$

Thus, there exist two different sets T_1, T_2 such that all queries return YES.

Next, we consider the case when at least one traitor in T need to be identified. Our strategy is to show that, for an arbitrary query sequence $F_1, \cdots, F_q$, there are potential traitor sets $T_1, \cdots, T_k$, all of size t, satisfying the following conditions: (i) for each T_j, all queries F_i returns YES, (ii) there is no single user that belongs to all the T_j. Thus, the identification of some user in some T_j will be *wrong* for some other $T_{j'}$.

To guarantee (ii), we simply pick $k = \binom{N-1}{t-1} + 1$. Because $\binom{N-1}{t-1}$ is the maximum number of t-subsets of $[N]$ which contain a given element. To guarantee (ii), we use the probabilistic argument as above: pick T uniformly at random.

$$\Pr_T[\text{all queries } F_i \text{ return YES}] \geq 1 - q\frac{\binom{r}{t}}{\binom{N}{t}} \geq \frac{t}{N} = \frac{\binom{N-1}{t-1}}{\binom{N}{t}}.$$

The last inequality holds because $q \leq \frac{\binom{N}{t}}{\binom{r}{t}}(1 - \frac{t}{N})$. ▌ The following upper-bound is asymptotically as good as the above lower-bounds for most practically meaningful ranges of r and t. However, it is an upper-bound in expectation only.

Theorem A.5.8 Let $t \leq r < N$ such that $t + r \leq N$. There is an adaptive strategy which uses on average at most $q = \frac{\binom{N}{t}}{\binom{r}{t}} + N - r$ queries.

Proof: We pose random "queries" R of size r to the blackbox decoder. (In reality, the queries are actually $\mathsf{Encaps}(\mathsf{EK}, R)$) until the answer is **no**. The expected number of such queries is the inverse of the probability that R contains the traitor set T, which is

$$\frac{\binom{N}{r}}{\binom{N-t}{r-t}} = \frac{N}{r}\frac{N-1}{r-1}\cdots\frac{N-t+1}{r-t+1} = \frac{\binom{N}{t}}{\binom{r}{t}}.$$

After some R containing T is found, we can test each $j \in R$ individually as follows. Fix $j' \in [N] - R$ Let $R' = R - \{j\} \cup \{j'\}$. If the answer to query R' is YES then $j \in T$. This way, we will be able to identify all members of T with an additional $N - r$ queries. ▌

A.6 Discussions

Optimization of the Code Length.

Given N and $r < N$, we presented a randomized construction of $\ell \times N$ (r, s)-disjunct matrices with $s = O(r \log(N/r)$ and $\ell = O(r^2 \log(N/r)$. Note that, even for r-disjunct matrices which do not address revocation, no other known construction, including randomized ones, has asymptotically smaller number of rows. It is also known that $\ell = \Omega(r^2 \log_r N)$ for any r-disjunct matrix.

Hence, if we use this matrix for broadcasting then the broadcaster's key size (or the total number of keys that will be attributed to users) is $O(r^2 \log(N/r))$. In this setting, we can trace back all "active" traitors (the traitors included in the pirate's codeword).

We can reduce the broadcaster's key size by using a related notion called *multiple user tracing* (MUT) families. Given positive integers $u \leq r$, an (r, u)-*MUT family* is a non-adaptive group testing matrix which, given the test outcomes imposed by an arbitrary set of $v \leq r$ positives, there is a decoding algorithm that outputs at least $\min(u, v)$ out of the v positives. It is known [AA07] that, given N, r, u, an $\ell \times N$ (r, u)-MUT matrix exists with $\ell = O\left((r + u^2) \log N\right)$. There is a method for constructing the MUT matrix so that even sublinear-time decoding is also reachable. Hence, we can use MUT families to design broadcast codes which can be used for tracing (assuming the naive pirate) up to $\sqrt{r}$ of the traitors while keeping the broadcaster's key size $O(r \log N)$. Note that in Complete Subtree, the broadcaster's key size is $2N - 1$.

Optimization of the Private Key Size.

In the above construction, the users' private key size is linear in the weight of its associated codeword. We can optimize this, for the case of 1−conjunction revocable code, by using Asano's method [Asa02]. The users' private key size then becomes constant. In this case, the security should also be based on a computational assumption that the RSA inversion is hard. This optimization is described in Appendix A.7 (Definition A.7.5).

On the *a priori*-Bounds of Revoked Users and Traitors.

Our scheme assumed an *a priori*-bounds r, t of revoked users and traitors. If the bound is unknown, a natural way to get around the problem is to "stack" on top of each other (r, s)-disjunct matrices for different values of r. This way, the resulting matrix will serve as (r, s)-codes for different r. The sacrifice is in code length. In the encryption mode, depending on the number of revoked users, we can use the appropriate matrix.

Acknowledgments

This work was supported in part by the French ANR-09-VERS-016 BEST Project, and by NSF CCF-1161196.

A.7 Appendix: Basic Definitions

Definition A.7.1 [Broadcast Encapsulation] A broadcast encapsulation scheme is a 3-tuple of algorithms $\mathcal{DBE} = (\mathsf{Setup}, \mathsf{Encaps}, \mathsf{Decaps})$:

- $\mathsf{Setup}(1^k, N)$, where k is the security parameter, and N the number of users, it generates the global parameters param of the system (omitted in the following); and returns a master secret key MSK and an encryption key EK. It also generates users' keys upk_i, for $i = 1, \ldots, N$.

- $\mathsf{Encaps}(\mathsf{EK}, R)$ takes as input a revoked set R and outputs a key header H and a session key $K \in \{0,1\}^k$.

- $\mathsf{Decaps}(\mathsf{usk}_i, R, H)$ takes as input the revoked set R and a user secret key. If $i \in [N] - R$, outputs the session key K.

The correctness requirement is that for any revoked set R and for any user $i \in [N] - R$ then the decapsulation algorithm gives back the ephemeral session key.

Definition A.7.2 [Trace&Revoke Encapsulation] A trace&revoke encapsulation scheme is a broadcast encapsulation scheme with an additional tracing algorithm $\mathsf{Trace}^{\mathbb{D}}(R_{\mathbb{D}}, \mathsf{pk}, \mathsf{msk})$: the traitor tracing algorithm interacts in a black-box manner with a pirate decoder $\mathbb{D}$ that is built from a certain set T of traitors. The algorithm takes as input a subset $R_{\mathbb{D}} \subset [N]$ (could be adversarially chosen), the public key pk, the master key msk and outputs a set $T_{\mathbb{D}} \subseteq [N]$.

More precisely, under the conditions:

- there are at most t traitors: $|T| \leq t$;
- The minimal revoked set does not contain all the traitors: $T \not\subseteq R_{\mathbb{D}}$, or equivalently $S_{\mathbb{D}} = ([N] - R_{\mathbb{D}})$ contains at least a traitor;
- $\mathbb{D}$ is "efficient" to decrypt ciphertexts (*i.e.,* decrypts with some non-negligible probability) for some revoked sets R that include the minimal revoked set $R_{\mathbb{D}}$ but do not contain all the traitors ($R_{\mathbb{D}} \subseteq R$ but $T \not\subseteq R$);

then the tracing algorithm outputs at least a traitor in $S_{\mathbb{D}}$, *i.e.,* : $\emptyset \neq T_{\mathbb{D}} \subseteq T \cap S_{\mathbb{D}}$.

Definition A.7.3 [Public-Key Encryption Scheme] $\mathcal{PKE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{Setup}(1^k)$, where k is the security parameter, generates the global parameters param of the system;
- $\mathsf{KeyGen}(\mathsf{param})$ generates a pair of keys, the public (encryption) key ek and the associated private (decryption) key dk;
- $\mathsf{Enc}(\mathsf{ek}, m; r)$ produces a ciphertext c on the input message m and the public key ek, using the random coins r (we may omit r when the notation is obvious);
- $\mathsf{Dec}(\mathsf{dk}, c)$ decrypts the ciphertext c under the private key dk. It outputs the plaintext, or $\perp$ if the ciphertext is invalid.

Definition A.7.4 [Secret Sharing Scheme] $\mathcal{SSS} = (\mathsf{Share}, \mathsf{Combine})$:

- $\mathsf{Share}(k, m, n)$, outputs a secret bit string K of length k, as well as n shares $s_1, \ldots, s_n$, so that any m of them will allow to recover K.
- $\mathsf{Combine}(\{(i, s_i)\})$, from m pairs (i, s_i), it recovers the bit string K.

The correctness requirement is that from any m-subset of $\{(i, s_i)\}$ generated by $\mathsf{Share}(k, m, n)$, the $\mathsf{Combine}$ algorithm outputs the bit string K generated by Share. Furthermore, the bit string K must be perfectly uniformly distributed.

Definition A.7.5 [Constant Size Private Key] Suppose there exists an algorithm that generates $1-$Conjunction $(r, s)-$Revocable $(\ell, N)-$Code.We build a BE scheme Π that can revoke up to r users in the following way.

- $\mathsf{Setup}(1^\lambda, N)$
  1. Run the Code generating algorithm on (N, r, d) to obtain a $1-$Conjunction (r, s) Revocable $(\ell, N)-$Code Γ.
  2. Generate two large primes of the same size p, q and publish $M = pq$

3. Generate ℓ pairs $(\mathsf{dk}_i, \mathsf{ek}_i), i = 1, \ldots, \ell$ such that $e_k d_k = 1(\mod (p-1)(q-1))$;
4. Choose a random $X \xleftarrow{\$} \mathbb{Z}_M^*$
5. Set $\mathsf{MSK} = (\Gamma, X, \{\mathsf{dk}_i\})$, $\mathsf{EK} = (N, \{\mathsf{ek}_i\})$, and $Reg = \emptyset$.

- $\mathsf{Extract}(\mathsf{MSK}, i)$

1. The user i is associated with the codeword $w^i \in \Gamma$.
2. Set $\mathsf{usk}_i \leftarrow X^{\prod_{j=1}^{\ell} \mathsf{dk}_j^{w_j^i}}$; $\mathsf{upk}_i \leftarrow i$; $Reg \leftarrow Reg \cup \{i\}$.

- $\mathsf{Encaps}(\mathsf{EK}, R)$:

1. The revoked set R should contain at most r users;
2. Because Γ is 1−conjunction (r, s)−revocable, one can find out an word c such that $\mathsf{D}_1(R, c) = 0$ and $\mathsf{D}_1(u, c) = 1$ for any $u \in [N] - R$, and $m = H(c) \leq d$.
3. Denote by $i_1, \ldots, i_m$ the positions of m bits 1 in c, *i.e.,* $c_{i_j} = 1$, for $j = 1, \ldots, m$
4. Set $e_{i_j} = X^{\mathsf{dk}_{i_j}}$, for $j = 1, \ldots, m$.
5. Output $\mathcal{K}_e$ and $Header = (c, (e_{i_j}), j = 1, \ldots, m)$.

- $\mathsf{Decaps}(\mathsf{usk}_j, R, Header)$:

1. If j is in $[N] - R$, then $\mathsf{D}_1(w^j, c) = 1$. There exists thus an index $1 \leq z \leq m$ such that $c_{i_z} = w_{i_z}^j = 1$
2. Compute $s_{i_z} = \mathsf{usk}_j^{\prod_{s=1, s\neq i_z}^{\ell} \mathsf{ek}_j^{w_s^i}}$. From the s_{i_z}, reconstruct $\mathcal{K}_e$

Definition A.7.6 [Trace&Revoke System from 1-Conjunction Trace&Revoke Codes] Let us be given a generator of $(r, s, \mathcal{Q}, p, \delta, t, \tau)$-blackbox Trace&Revoke 1-Conjunction (ℓ, N)-Codes, and a secure public-key encryption scheme $\mathcal{PKE}$. We build a Trace&Revoke encapsulation scheme Π that can revoke up to r users, and tracing traitor for a pirate decoder having up to t traitors' keys, in the following way.

- $\mathsf{Setup}(1^\lambda, N)$

1. Run the code generating algorithm on $(\mathcal{Q}, N, k, r, t, s)$ to obtain an $(r, s, \mathcal{Q}, p, \delta, t, \tau)$-blackbox Trace&Revoke 1-Conjunction (ℓ, N)-Codes.
2. Run $\mathcal{PKE}.\mathsf{Setup}(1^\lambda)$ to get the public parameters param for the encryption scheme;
3. For $i = 1, \ldots, \ell$, run the key generation algorithm $\mathcal{PKE}.\mathsf{KeyGen}(\mathsf{param})$ to get the pair $(\mathsf{dk}_i, \mathsf{ek}_i)$.
4. Set $\mathsf{MSK} = (\Gamma, \{\mathsf{dk}_i\})$, and $\mathsf{EK} = \{\mathsf{ek}_i\}$.
5. For $i = 1, \ldots, N$, the user i is associated with the codeword $w^i \in \Gamma$: we set $\mathsf{usk}_i \leftarrow \{\mathsf{dk}_j / w_j^i = 1, j = 1, \ldots, \ell\}$.

- $\mathsf{Encaps}(\mathsf{EK}, R)$:

1. For a revoked set R of size at most r, since the code Γ is efficiently (r, s)-revocable, one can find out a signal c of weight at most s, such that $\mathsf{D}_1(u, c) = 0$, for any $u \in F(R)$, and $\mathsf{D}_1(u, c) = 1$ for any $u \in [N] - R$. We denote by $m = \mathbf{w}_H(c)$ this weight;
2. Denote by $i_1, \ldots, i_m$ the positions of m 1-bits in c, *i.e.,* $c_{i_j} = 1$, for $j = 1, \ldots, m$;

3. Choose a random session key $K \xleftarrow{\$} \{0,1\}^\kappa$.
4. Set $e_{i_j} = \mathcal{PKE}.\mathsf{Enc}(\mathsf{pk}_{i_j}, K)$, for $j = 1, \ldots, m$.
5. Output K and $H = (c, (e_{i_j}), j = 1, \ldots, m)$.

- $\mathsf{Decaps}(\mathsf{usk}_j, R, H)$:

  1. If j is in $[N] - R$, then $\mathsf{D}_1(w^j, c) = 1$. This means $\mathbf{w}_H(w^j \wedge c) \geq 1$ and there exists thus i_j in $w^j \wedge c$
  2. Compute $K = \mathcal{PKE}.\mathsf{Enc}(\mathsf{sk}_{i_j}, e_{i_j})$.

- $\mathsf{Trace}^{\mathbb{D}}(R_{\mathbb{D}}, \mathsf{pk}, \mathsf{msk})$: Running the tracing algorithm for the code, each time the tracer asks a qualified query c to the pirate decoder, we do as follows: run $\mathsf{Encaps}(\mathsf{EK}, R)$ but directly use the signal c (in fact, the revoked set R in this case corresponds to the set of the users that cannot decrypt c) and query the pirate decoder on the R, H. If the pirate decoder exactly recovers the session key K, we return 1 to the tracer for the code, and otherwise we return 0.

It is straightforward that if the pirate decoder $R_{\mathbb{D}}$ answers all ciphertexts constructed from qualified signal c for $\mathcal{Q}$, then the tracing procedure can be directly reduced to the tracing for the code, and thus we can identify traitors, as in the $(r, s, \mathcal{Q}, p, \delta, t, \tau)$-blackbox Trace&Revoke.

In the particular case of Q_a, the fact that the pirate decoder $R_{\mathbb{D}}$ answers all ciphertexts constructed from qualified signal c for $\mathcal{Q}_a$ implies that the pirate decoder decrypts all ciphertext with a header H containing a encapsulations of the session key, each is encrypted by a key at a row of the matrix.

Appendix B

Traitor Tracing with Optimal Transmission Rate

ISC 2011
[FNP07b] with Nelly Fazio, and Antonio Nicolosi

Abstract : *We present the first traitor tracing scheme with efficient black-box traitor tracing in which the ratio of the ciphertext and plaintext lengths (the* transmission rate*) is asymptotically 1, which is optimal. Previous constructions in this setting either obtained constant (but not optimal) transmission rate [KY02c], or did not support black-box tracing [CPP05a].*

Our treatment improves the standard modeling of black-box tracing by additionally accounting for pirate strategies that attempt to escape tracing by purposedly rendering the transmitted content at lower quality.

Our construction relies on the decisional bilinear Diffie-Hellman *assumption, and attains the same features of public traceability as (a repaired variant of) [CPP05a], which is less efficient and requires non-standard assumptions for bilinear groups.*

B.1 Introduction

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

The first formal definition of traitor tracing scheme appears in Chor *et al.* [CFN94b, CFNP00], whose construction requires storage and decryption complexity $O(t^2 \log^2 t \log(n/t))$ and communication complexity $O(t^3 \log^4 t \log(n/t))$, where n is the size of the universe of users and t is an upper bound on the number of traitors. Stinson and Wei later suggested in [SW98a] explicit combinatorial construction that achieve better efficiency for small values of t and n.

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

In [BF99b], Boneh and Franklin present an efficient public-key traitor tracing scheme with deterministic t-tracing based on an algebraic approach. Its communication, storage and decryption complexities are all $O(t)$. The authors also introduce the notion of *non-black-box traceability*: given a "valid" key extracted from a pirate device (constructed using the keys of at most t users), recover the identity of at least one traitor. This is in contrast with the notion of *black-box tracing* (on which we focus in this paper), where the traitor's identity can be uncovered by just observing the pirate decoder's replies on "well crafted" ciphertexts. More recently, Boneh *et al.* [BSW06b, BW06b] proposed traitor tracing schemes that withstand any number of traitors (*full traceability*), while requiring a sub-linear ciphertext length ($O(\sqrt{n})$). In [Pfi96], Pfitzmann introduces the notion of *asymmetric* traitor tracing. In this model, tracing uncovers some secret information about the traitor that was *a priori* unknown to the system manager. Thus, the result of the tracing algorithm provides actual evidence of the treachery. Further results in this direction are in [KD98b, KY02d, KY02a].

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

As originally observed in [GSY99], traitor tracing scheme are most useful when combined with a revocation scheme; such trace-and-revoke approach consists in first uncovering the compromised decryption keys and then revoking their decryption capabilities, thus rendering the corresponding pirate decoder useless [NP00, TT01, NNL01, DF02, DF03, KHL03, DFKY05, BW06b].

Constant Transmission Rate. All proposals mentioned so far result into schemes that are not quite communication-efficient: the length of each ciphertext is (at least) t times longer than the embedded plaintext. As pointed out by Kiayias and Yung in [KY02c], an important problem in designing practical traitor tracing schemes is to ensure a low *transmission rate*, defined as the asymptotic ratio of the size of ciphertexts over the size of plaintexts, while at the same time minimize the *secret-* and the *public-storage* rates, similarly defined as the asymptotic ratio of the size of user-keys and of public-keys over the size of plaintexts.[1] Under this terminology, the transmission rate of all the above mentioned solutions is linear w.r.t. the maximal number t of traitors, whereas in [KY02c], Kiayias and Yung show that if the plaintexts to be distributed are large (which is the case for most applications of traitor tracing, such as distribution of multimedia content), then it is possible to obtain ciphertexts with constant expansion rate. Their solution is based on collusion-secure fingerprint codes [BS98, Tar03] and its parameters are summarized in Figure B.1.

Besides the clear benefit in terms of communication efficiency, schemes with constant transmission rate also enjoy efficient black-box traceability, while schemes with linear transmission rate are inherently more limited in this regard [KY01c] (*e.g.,* the black-box traitor tracing of [BF99b] takes time proportional to $\binom{n}{t}$).

In [CPP05a], Chabanne *et al.* extend the setting of [KY02c] with the notion of *public traceability*: Whereas traditional tracing algorithms require knowledge of the system's secret information, in a scheme with public traceability everyone can run the tracing algorithm. In this

[1] We adopt a terminology slightly different from the one of [KY02c], which uses the term *ciphertext/user-key/public-key* rates, for what we called *transmission/secret-storage/public-storage* rates. Moreover, in [KY02c] *transmission rate* refers to the sum of the all the three rates. Our choice is of course mostly a matter of taste: we prefer the terminology of this paper as it makes more evident the role played by each quantity in a concrete implementation of the system.

	Trans. Rate	S-Storage Rate	P-Storage Rate	BB Tracing	Public Traceability	Hardness Assumption
BF[BF99b]	$2t+1$	$2t$	$2t+1$	$\times$	$\times$	DDH
KY[KY02c]	3	2	4	$\surd*$	$\times$	DDH
CPP[CPP05b]	1	2	1	$\times$	$\times$	DBDH^2-E $\wedge\mathsf{DBDH}^1$-M
PST[PSNT06c]	7	1	1	$\surd$	full	DDH
Repaired CPP	3	2	6	$\surd$	local	DBDH^2-E $\wedge\mathsf{DBDH}^1$-M
Our Scheme	1	2	10	$\surd$	local	DBDH

Figure B.1: Comparison of rates (*transmission*, *secret-* and *public-storage* rates) and tracing features (*black-box tracing* and *public traceability*) between existing schemes and our construction. The "*" in the row labeled "[KY02c]" refers to the fact that the scheme of [KY02c] can support black-box tracing using the tracing algorithm that we describe in Appendix B.7.3. The row labeled "[PSNT06b]" refers to instantiating their generic construction with ternary IPP codes and ElGamal-style encryption. The row labeled "Repaired [CPP05a]" refers to the variant of the scheme of [CPP05a] that we suggest in Appendix B.7.3 to support black-box tracing.

paper, we also consider *local public traceability*, whereby public information suffices to carry out the preliminary phase of tracing, which requires interaction with the pirate decoder, and results in an encoding of the traitor's identity that can be decoded with a master key. This separation of tasks ensures that the system's secret information is only needed for off-line operations (*i.e.*, user registration and possibly the final phase of tracing), thus improving the overall security of the system by allowing for safer storage solutions.

The work of [PSNT06b] describes a traitor tracing scheme with constant (but not optimal) transmission rate and (full) public traceability based on Identifiable Parent Property (IPP) codes. Figure B.1 also reports on these two schemes. One could think that traitor tracing schemes with linear transmission rate (*e.g.* [BF99b]) could easily be turned into schemes with constant transmission rate by means of hybrid encryption: To send a large message, pick a random session key, encrypt it with the given traitor tracing scheme, and append a symmetric encryption of the message under the chosen anonymous session key. This approach, however, suffers from a simple yet severe untraceable pirate strategy: Just decrypt the session key and make it available to the "customers" on the black market, *e.g.,* via anonymous e-mail, or via text-messaging from a pre-paid cellphone. Clearly, when a traitor tracing scheme is used to encrypt the content directly, this "re-broadcasting" strategy becomes much less appealing for would-be pirates, because of the higher costs and exposure risks associated with running a high-bandwidth darknet.

Our Contributions. We present the first public-key traitor tracing scheme with efficient black-box traitor tracing and local public traceability in which the transmission rate is asymptotically 1, which is optimal. Encryption involves the same amount of computation as in [CPP05a]; decryption is twice as fast. We also considerably simplify the computational hardness requirements, relying just on the DBDH assumption—much weaker and more widely accepted than the non-standard bilinear assumptions employed in [CPP05a].

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

As additional contribution, we point out and resolve an issue in the black-box tracing of [KY02c] (which was also independently addressed in a revised version of their work [KY06]).

We then show that [CPP05a], which extends [KY02c] and inherits its tracing mechanism, inherits in fact the above-mentioned problem, too. In this case, however, fixing the black-box functionality requires changes that intrinsically conflict with the optimizations put up by [CPP05a] to achieve optimal transmission rate. In other words, [CPP05a] can either provide optimal transmission rate with only non-black-box tracing, or support local public traceability with sub-optimal transmission rate, but cannot achieve both at the same time.

Organization. Section B.2 introduces the tools needed for our construction. Section B.3 defines the syntactic, security, and traceability properties of traitor tracing schemes. We present our new traitor tracing scheme and its security analysis in Section B.4, and Section B.5 discusses a concrete choice of parameters. In the Appendix, we point out a flaw in the tracing algorithms of [KY02c] and [CPP05a] and propose fixes.

B.2 Preliminaries

The security properties of our construction hinge upon the decisional bilinear Diffie-Hellman assumption (DBDH) for $(\mathbb{G}_1, \mathbb{G}_2)$. We refer the reader to Appendix B.7.1. for the relevant definitions.

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

Definition B.2.1 Let Σ be a finite alphabet, and $n, v \in \mathbb{Z}_{\geq} 0$. An (n, v)-code over Σ is a set of n v-tuples of symbols of Σ: $\mathcal{C} = \{\omega^{(1)}, \ldots, \omega^{(n)}\} \subseteq \Sigma^v$.

Definition B.2.2 Let T be a subset of indices in $[1, n]$. The set of undetectable positions for T is: $R_T = \{\ell \in [1, v] \mid (\forall i, j \in T).[\omega_\ell^{(i)} = \omega_\ell^{(j)}]\}$.

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

Definition B.2.3 The set of feasible codewords for T is: $F_T = \{\bar{\omega} \in (\Sigma \cup \{?\})^v \mid \bar{\omega}_{|R_T} = \omega_{|R_T}\}$.

Definition B.2.4 Let $\varepsilon > 0$ and $t \in \mathbb{Z}_{\geq} 0$. $\mathcal{C}$ is an (ε, t, n, v)-collusion-secure code over Σ if there exists a probabilistic polynomial-time algorithm $\mathcal{T}$ such that for all $T \subseteq [1, n]$ of size $\mid T \mid \leq t$, and for all $\bar{\omega} \in F_T$, it holds that: $\Pr[\mathcal{T}(r_{\mathcal{C}}, \bar{\omega}) \in T] \geq (1 - \varepsilon)$, where the probability is over the random coins $r_{\mathcal{C}}$ used in the construction of the (n, v)-code $\mathcal{C}$, and over the random coins of $\mathcal{T}$.

B.3 Public-Key Traitor Tracing Scheme with Public Traceability

Definition B.3.1 [Public-Key Traitor Tracing Scheme] A public-key traitor tracing scheme is a 5-tuple of probabilistic polynomial-time algorithms (**Setup**, **KeyDer**, Encaps, Decaps, **Trace**), where:

Setup: On input a security parameter 1^κ, a collusion threshold 1^t, and a bound n on the maximum number of users, returns a public key pk along with some master secret information msk (*cf.* **KeyDer** and **Trace**);

KeyDer: Given msk and a user index $i \in [1, n]$, outputs a "fingerprinted" user key sk_i;[2]

Encaps**:** On input key pk and a message m (from the appropriate message space $\mathcal{M}$, implicitly described by pk), returns a (randomized) ciphertext ψ;

Decaps**:** On input a user key sk_i and a ciphertext ψ, recovers the message encrypted within ψ;

Trace: Given the master secret key msk, the public key pk, and black-box access to a "pirate" decoder capable of inverting the $\mathsf{Encaps}(\mathsf{pk}, \cdot)$ functionality, returns the user index of one of the traitors that contributed his/her user key for the realization of the pirate decoder, or the special user index 0 upon failure.

For correctness, decryption with any user key output by **KeyDer** should "undo" encryption:

$$\Pr\left[\mathsf{Decaps}(\mathsf{sk}_i, \mathsf{Encaps}(\mathsf{pk}, m)) = m \;\middle|\; \begin{array}{c}(\mathsf{pk}, \mathsf{msk}) \xleftarrow{\$} \mathbf{Setup}(1^\kappa, 1^t, n), m \xleftarrow{\$} \mathcal{M}, \\ i \xleftarrow{\$} [1, n], \mathsf{sk}_i \xleftarrow{\$} \mathbf{KeyDer}(\mathsf{msk}, u)\end{array}\right] = 1,$$

where the probability is over the random coins of **Setup**, **KeyDer**, Encaps, Decaps, and over the random selection of m from $\mathcal{M}$ and of i from $[1, n]$.

Definition B.3.2 [Full/Local Public Traceability] A public-key traitor tracing scheme is said to support: 1) *public traceability* if the **Trace** algorithm can be implemented without the master secret key msk; or 2) *local public traceability* if the **Trace** algorithm can be split in an on-line phase, in which the pirate decoder can be queried without knowledge of the secret key, and an off-line phase, without access to the pirate decoder, that can retrieve the identity of the traitor from the master secret key and the output of the publicly executable on-line phase.

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

Definition B.3.3 [Indistinguishability under Chosen-Plaintext Attack] A public-key traitor tracing scheme satisfies ε_{ind}-indistinguishability if, for any pair of probabilistic polynomial-time algorithms $(\mathcal{A}_1, \mathcal{A}_2)$, it holds that:

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

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

Requirements on the Tracing Functionality. Existing literature usually models black-box traceability as the ability to "extract" the identity of (at least) one traitor from pirate decoders that *correctly* invert the decryption algorithm (under appropriate efficiency and success probability constraints). This approach, however, is often criticized because it leaves the way

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

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

Definition B.3.4 A public-key traitor tracing scheme is $\varepsilon_{\text{trac}}$-traceable if for any probabilistic polynomial-time traitor strategy $\mathcal{A}$, it holds that:

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

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

Notice that Definition B.3.4 subsumes the case that the traitor strategy $\mathcal{A}$ only produces a "good" pirate decoder $\mathcal{D}$ with a low (but non-negligible) probability: indeed, any such strategy can be "boosted" by simply keeping executing $\mathcal{A}$ on fresh random coins, until the pirate decoder $\mathcal{D}$ that $\mathcal{A}$ outputs is a good one (which can be efficiently tested by estimating $\mathcal{D}$'s decryption capability on the encryption of a random plaintext).

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

Similarly to the schemes of [KY02c] and [CPP05a], our construction is based on the use of an (ε, t, n, v)-collusion-secure code $\mathcal{C}$ over the alphabet $\{0, 1\}$ (*cf.* Definition B.2.4). At a high level, the idea is to first define a two-user sub-scheme resilient against a single traitor, and then "concatenate" v instantiations of this sub-scheme according to the code $\mathcal{C}$; in particular, each user $i \in [1, n]$ is associated to a codeword $\omega^{(i)}$ in $\mathcal{C}$, and given decryption key $\mathsf{sk}_i \doteq (K_{1,\omega_1^{(i)}}, \ldots, K_{v,\omega_v^{(i)}})$, where $\omega_j^{(i)}$ is the j-th bit of the codeword $\omega^{(i)}$, and $K_{j,0}, K_{j,1}$ are the keys for the j-th instantiation of the basic two-user sub-scheme. Although the overall architecture that we follow is well-known, achieving optimal transmission rate along these lines requires solving a number of technical problems, on which we elaborate in Section B.4.4.

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

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

B.4.1 Our Two-User Sub-Scheme

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

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

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

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

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

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

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

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

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

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

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

Before moving on to the security and traceability of our two-user scheme in the sense of Definitions B.3.3 and B.3.4 (*cf. Section B.3*), we remark that **Trace** does not require knowledge of the master secret key msk, and thus it supports full public traceability (*cf.* Definition B.3.2). Also, notice that decryption requires only one pairing computation.

B.4.2 Indistinguishability under Chosen-Plaintext Attack

Theorem B.4.1 Under the DBDH assumption for $(\mathbb{G}_1, \mathbb{G}_2)$, the scheme in Section B.4.1 is secure w.r.t. indistinguishability under chosen-plaintext attack (*cf.* Definition B.7.1 and Definition B.3.3).

Proof: To a contradiction, let us assume that the scheme does not satisfy Definition B.3.3 *i.e.*, there is an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that, given the public key

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

$\mathsf{pk} = (q, \mathbb{G}_1, \mathbb{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$, can break the scheme with non-negligible advantage ε_{ind}. We then construct an algorithm $\mathcal{B}$ (whose running time is polynomially related to $\mathcal{A}$'s) that breaks the DBDH assumption with probability $\varepsilon_{\mathsf{DBDH}} = \varepsilon_{\text{ind}}$.

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

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

$\mathcal{B}$ can now set $\mathsf{pk} \doteq (q, \mathbb{G}_1, \mathbb{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$ and send it to $\mathcal{A}_1$.

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

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

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

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

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

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

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

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

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

B.4.3 Traceability

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

Definition B.4.2 [Valid and Probe Ciphertexts] Let $\sigma \in \{0, 1\}$, $\hat{m} \in \mathcal{M}$, $\hat{U} \in \mathbb{G}_1$, $\hat{V} \in \mathbb{G}_2$, $\hat{W} = \hat{m} \oplus H(\hat{U}^{\alpha_\sigma} e(\hat{V}, B_\sigma))$, and $\hat{\psi} = \langle \hat{U}, \hat{V}, \hat{W} \rangle$. We say that the ciphertext $\hat{\psi}$ is:

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

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

Lemma B.4.3 [Indistinguishability of Valid *vs.* Probe Ciphertexts] Under the DBDH assumption for $(\mathbb{G}_1, \mathbb{G}_2)$, given the public key $\mathsf{pk} = (q, \mathbb{G}_1, \mathbb{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$ and the secret key $\mathsf{sk}_\tau = \alpha_\tau$ of user $\tau \in \{0,1\}$ (where $A_\tau = \alpha_\tau R$), it is infeasible to distinguish a valid ciphertext from a τ-probe.

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

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

Setup: $\mathcal{B}$ lets $P \doteq xP'$, $Q \doteq P'$, $R \doteq yP'$, chooses $\alpha_0, \beta_0, \beta_1 \xleftarrow{\$} \mathbb{Z}^*q$ and computes $A_0 \doteq \alpha_0 R$, $B_0 \doteq \beta_0 P$ and $B_1 \doteq \beta_1 P$. $\mathcal{B}$ also sets $A_1 \doteq A_0 + \beta_0 Q - \beta_1 Q$, which implicitly defines $h = e(P, A_0) \cdot e(Q, B_0) = e(P, A_1) \cdot e(Q, B_1)$. $\mathcal{B}$ now defines $\mathsf{pk} \doteq (q, \mathbb{G}_1, \mathbb{G}_2, e, H, P, Q, R, A_0, B_0, A_1, B_1)$. Then, $\mathcal{B}$ prepares a challenge ciphertext $\hat{\psi} \doteq \langle \hat{U}, \hat{V}, \hat{W} \rangle$ by setting $\hat{U} \doteq h'$, $\hat{V} \doteq zP' (= zQ$, thus implicitly defining $k = z)$ and $\hat{W} \doteq \hat{m} \oplus H(\hat{U}^{\alpha_0} e(\hat{V}, B_0))$, for $\hat{m} \xleftarrow{\$} \mathcal{M}$. At this point, $\mathcal{B}$ feeds $\mathcal{A}$ with pk, $\hat{\psi}$, and α_0.

Attack: $\mathcal{A}$ returns her guess to whether $\hat{\psi}$ is a valid ciphertext or a probe (w.r.t. the public key pk).

Break: $\mathcal{B}$ outputs yes or no accordingly.

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

An important consequence of Lemma B.4.3 is that pirate decoders created by user τ reply to τ-probes with an m^* such that $\mathcal{R}(\hat{m}, m^*) = 1$ with non-negligible probability $\hat{p}_\mathcal{D}$:

Corollary B.4.4 Let $\mathcal{D}, \mathcal{R}$ be the pirate decoder and resemblance relation output by a traitor strategy $\mathcal{A}$ based on the user key α_τ, such that $p_\mathcal{D}$ is non-negligible and $p_\mathcal{R}$ is negligible (*cf.* Definition B.3.4). Let $\hat{\psi}$ be a τ-probe for a message $\hat{m} \xleftarrow{\$} \mathcal{M}$. Under the DBDH assumption, $\hat{p}_\mathcal{D} \doteq \Pr[\mathcal{R}(\hat{m}, m^*) = 1 \mid m^* \xleftarrow{\$} D(\hat{\psi})]$ is non-negligible.

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

in discerning valid ciphertext from τ-probes is clearly $p_{\mathcal{D}} - \hat{p}_{\mathcal{D}}$. But if $\hat{p}_{\mathcal{D}}$ were negligible, such algorithm $\mathcal{B}$ would violate the statement of Lemma B.4.3, proving our argument. ▮

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

Lemma B.4.5 Replacing $\hat{\psi}$ with a $(1-\tau)$-probe in the setting of Corollary B.4.4, $\Pr[\mathcal{R}(\hat{m}, m^*) = 1]$ is negligible.

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

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

The above argument can be easily rephrased along the lines of the reductions described in the proofs of Theorem B.4.1 and Lemma B.4.3; we refrain from doing so due to space limitations. ▮

Theorem B.4.6 Under the DBDH assumption for $(\mathbb{G}_1, \mathbb{G}_2)$, our **Trace** algorithm has a negligible traceability error.

Proof: Let $\mathcal{D}$, $\mathcal{R}$ be the pirate decoder and resemblance relation on which the **Trace** algorithm is being run, and let τ be the traitor index. Corollary B.4.4 guarantees that **Trace** will on average terminate after $2/p_{\mathcal{D}}$ queries to $\mathcal{D}$. Upon termination, **Trace**'s output will be wrong only if it happens that $\mathcal{D}$ replies to a $(1-\tau)$-probe $\hat{\psi}$ with an m^* satisfying $\mathcal{R}(\hat{m}, m^*) = 1$, *i.e.*, $\Pr[\mathbf{Trace}^{D(\cdot)}(pk, \perp) \notin T] = \Pr[\hat{\psi} \text{ is a } (1-\tau)\text{-probe} \mid \mathcal{R}(\hat{m}, m^*) = 1]$, which by Corollary B.4.4, Lemma B.4.5, and Bayes' theorem is easily seen to equal $p_{\mathcal{R}}/(p_{\mathcal{D}} + p_{\mathcal{R}})$, which is negligible. ▮

B.4.4 Our Multi-User Scheme

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

Since tracing requires the ability to set up each component of the ciphertext independently of all the others, it may seem necessary to use completely unrelated instantiations of the two-user sub-scheme for each component. This is done, for example, in [KY02c]. (*cf.* Appendix B.7.2). Having independent components, however, clearly leads to a multi-user scheme with the same transmission rate as the underlying basic two-user scheme, and so it would not help us attaining

optimal transmission rate. In fact, the scheme of [CPP05a] (*cf.* Appendix B.7.2) manages to get transmission rate 1 by sacrificing component independence, and instead using component-instances all very closely related to each other. As we show in Appendix B.7.3, though, their scheme does not support black-box traceability.

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

To prevent pirate decoders from selectively ignoring the "special" position (which is the only part of the ciphertext that encodes tracing information), we follow the approach proposed in [KY02c], by which the encryption algorithm preliminarily processes the plaintext with an *All-Or-Nothing* transform (AONT) [Riv97, Boy99, CDH+00]. This will force decoders to decrypt *all* blocks of the ciphertext, since ignoring even a single one would result in missing at least one block of the AONT-transformed plaintext, so that, by the properties of AONT's, such decoders would fail to recover *any* information about the original plaintext being transmitted. We remark that reliance on AONT's to force the pirate to include (at least) one key for each component was suggested in [KY02c], but later dismissed by the authors in [KY06] as ineffective for the black-box setting, since it cannot prevent cropping of the plaintext once it has been decrypted. However, we believe their critique to be misleading, since traitor strategies in which the pirate decoder tampers with the decrypted plaintexts are dealt with the use of the resemblance relation $\mathcal{R}$ (see discussion in Section B.3), while AONT's prevent the pirate from learning anything about the plaintext if even a single block cannot be decrypted.

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

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

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

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

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

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

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

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

to:[6]

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

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

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

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

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

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

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

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

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

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

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

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

Let $m^{*(\ell)} \doteq (m_1^{*(\ell)} \| \ldots \| m_v^{*(\ell)})$ be the plaintext output by $\mathcal{D}$ when fed with the ciphertext $\hat{\psi}^{(\ell)}$. If $\mathcal{R}(\hat{m}^{(\ell)}, m^{*(\ell)}) = 1$, then set $\hat{\omega}^{(\ell)} = \sigma^{(\ell)}$; otherwise, pick fresh random $k_\ell, k'_\ell, k^{(\ell)}$ from $\mathbb{Z}q$, $\hat{m}^{(\ell)}$ from $\mathcal{M}$, $\sigma^{(\ell)}$ from $\{0,1\}$, and repeat, until either $\mathcal{R}(\hat{m}^{(\ell)}, m^{*(\ell)}) = 1$, or

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

the iteration has failed some fixed polynomial number of time, in which case $\hat{\omega}^{(\ell)}$ is set arbitrarily.

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

Remark B.4.7 Since the **Trace** algorithm needs msk only in the off-line phase, which does not access the pirate decoder and is much less computation-intensive,[7] our multi-user scheme supports local public traceability.

Remark B.4.8 We bound the number of trials that **Trace** performs to extract each bit $\hat{\omega}^{(\ell)}$ because a pirate decoder holding both keys for position ℓ could cause the test $\mathcal{R}(\hat{m}^{(\ell)}, m^{*(\ell)}) = 1$ to fail with probability 1. A suitable value for this bound is $O(1/p_{\mathcal{D}})$, where $p_{\mathcal{D}}$ is the success probability (over random valid ciphertexts) of the decoder under tracing, which can be efficiently estimated using Chernoff bounds.

Remark B.4.9 Notice that the size of the message blocks can be shrunk to any $\kappa' \leq \kappa$, by choosing a universal hash function $H : \mathbb{G}_2 \to \{0,1\}^{\kappa'}$. This is possible as long as $\kappa' > \log v + \log(1/\varepsilon) = O(\log t + \log\log(n/\varepsilon) + \log(1/\varepsilon))$, which ensures that, during tracing, the probability of a hash collision in any of the v components of the scheme is bounded by ε. For a typical choice of parameters ($n = 2^{30}$, $\varepsilon = 2^{-30}$, $t = 30$), κ' can be chosen as low as 64 bits.

Pre-Processing Messages with AONT's.

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

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

$$\mathsf{Encaps}'(m) \doteq \mathsf{Encaps}(\mathrm{AONT}(m)) \qquad \mathsf{Decaps}'(\psi) \doteq \mathrm{AONT}^{-1}(\mathsf{Decaps}(\psi))$$

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

B.4.5 Indistinguishability under Chosen-Plaintext Attack

In this section, we assess the security of the multi-user scheme of Section B.4.4. (For lack of space, we defer all proofs for this section to Appendix B.7.4.)

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

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

Lemma B.4.10 If the multi-user scheme without AONT pre-processing is secure w.r.t. indistinguishability under chosen-plaintext attack, then the multi-user scheme with AONT pre-processing is secure w.r.t. the same notion.

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

Lemma B.4.11 If our two-user scheme is secure w.r.t. indistinguishability under chosen-plaintext attack, then our multi-user scheme without AONT pre-processing is secure w.r.t. the same notion.

In light of Theorem B.4.1, our main security theorem follows immediately from Lemmas B.4.10 and B.4.11:

Theorem B.4.12 Under the DBDH assumption for $(\mathbb{G}_1, \mathbb{G}_2)$, the scheme in Section B.4.4 is secure w.r.t. indistinguishability under chosen-plaintext attack.

B.4.6 Traceability

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

Definition B.4.13 Let $\ell \in [1, v]$, $\sigma \in \{0,1\}$, $\hat{m} \in \mathcal{M}$, $\hat{M} = (\hat{M}_1, \ldots, \hat{M}_v) \xleftarrow{\$} \mathrm{AONT}(\hat{m})$, $\hat{U}_\ell \in \mathbb{G}_2$, $\hat{V}_\ell \in \mathbb{G}_1$, $k \in \mathbb{Z}_q$, $U = e(P,R)^k$, $V = kQ$, $W_j = \hat{M}_j \oplus H(h_j^k)$ $(j = 1, \ldots, v,\ j \neq \ell)$, $\hat{W}_\ell = \hat{M}_\ell \oplus H(\hat{U}_\ell^{\alpha_{\ell,\sigma}} e(\hat{V}_\ell, B_{\ell,\sigma}))$, and $\hat{\psi} = \langle \ell, \hat{U}_\ell, \hat{V}_\ell, U, V, W_1, \ldots, \hat{W}_\ell, \ldots, W_v \rangle$. We say that the ciphertext $\hat{\psi}$ is:

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

Our analysis is organized as follows. Let T denote the set of indices of the t traitors. Lemma B.4.14 proves the computational indistinguishability of valid ciphertexts *vs.* (ℓ, τ^ℓ)-probes when only the τ^ℓ subkey is known for position ℓ. It follows (Corollary B.4.15) that pirate decoders must decrypt such (ℓ, τ^ℓ)-probes correctly (w.r.t. the chosen resemblance relation). Lemma B.4.16 then shows that instead $(\ell, 1-\tau^\ell)$-probes cannot be properly decrypted, and Lemma B.4.17 combines Corollary B.4.15 and Lemma B.4.16 to argue that the chances that the ℓ-th stage of tracing fails to extract the correct bit $\hat{\omega}^{(\ell)} = \tau^\ell$ from $\mathcal{D}$ are negligible, which implies the overall traceability of our scheme (Theorem B.4.18).

Lemma B.4.14 [Indistinguishability of Valid *vs.* Probe Ciphertexts] Under the DBDH assumption for $(\mathbb{G}_1, \mathbb{G}_2)$, given the public key $\mathsf{pk} = (q,\ \mathbb{G}_1,\ \mathbb{G}_2,\ e,\ H,\ P_j, Q_j, R_j,\ (A_{j,0},\ B_{j,0},\ A_{j,1},\ B_{j,1})_{j=1,\ldots,v},\ P, Q, R,\ (\bar{A}_{j,0},\ \bar{B}_{j,0},\ \bar{B}_{j,1})_{j=1,\ldots,v})$ and the secret keys $\mathsf{sk}_i \doteq ((\alpha_{j,\omega_j^{(i)}})_{j=1,\ldots,v}, (\bar{\alpha}_{j,\omega_j^{(i)}})_{j=1,\ldots,v})$ for each $i \in T$, it is infeasible to distinguish valid ciphertexts from (ℓ, τ^ℓ)-probes, if the codewords of all traitors in T have bit τ^ℓ at position ℓ.

Proof: Since the ℓ-th "special" sub-schemes is completely independent from the rest of our construction, the thesis follows as a simple reduction to Lemma B.4.3. ▌

Corollary B.4.15 Let $\mathcal{D}$, $\mathcal{R}$ be the pirate decoder and resemblance relation output by a traitor strategy $\mathcal{A}$ based on the user keys of the traitors in T, such that $p_{\mathcal{D}}$ is non-negligible and $p_{\mathcal{R}}$ is negligible (*cf.* Definition B.3.4). Assume the codewords of all the traitors in T have bit τ^ℓ at position ℓ, and let $\hat{\psi}$ be an (ℓ, τ^ℓ)-probe for a message $\hat{m} \xleftarrow{\$} \mathcal{M}$. Under the DBDH assumption, $\hat{p}_{\mathcal{D}} \doteq \Pr[\mathcal{R}(\hat{m}, m^*) = 1 \mid m^* \xleftarrow{\$} D(\hat{\psi})]$ is non-negligible.

Proof: Reduces to Lemma B.4.14 exactly as Corollary B.4.4 reduces to Lemma B.4.3. ▮

Lemma B.4.16 Replacing $\hat{\psi}$ with an $(\ell, 1 - \tau^\ell)$-probe in the setting of Corollary B.4.15, $\Pr[\mathcal{R}(\hat{m}, m^*) = 1]$ is negligible, if the AONT employed in the system is secure.

Proof: The argument described in the proof of Lemma B.4.5 implies that the AONT-transformed message block $\hat{M}_\ell$ is computationally hidden from the pirate decoder's view. By the properties of AONT's, the whole original message $\hat{m}$ is then also computationally hidden from $\mathcal{D}$, so that in fact $\hat{m}$ is just a random message independent from the output m^* of $\mathcal{D}$, and hence $\mathcal{R}(\hat{m}, m^*) = 1$ holds with probability $p_{\mathcal{R}}$, which is negligible. ▮

Lemma B.4.17 Consider the ℓ-th stage of the **Trace** algorithm, when the tracer queries the decoder $\mathcal{D}$ with (ℓ, σ)-probes for random $\sigma \in \{0, 1\}$. If all codewords of the traitors in T have bit τ^ℓ in the ℓ-th position, then the ℓ-th stage will terminate setting $\hat{\omega}^\ell = 1 - \tau^\ell$ with negligible probability.

Proof: The assumption that $\mathcal{D}$ does not contain both keys for position ℓ implies, by Corollary B.4.15, that the ℓ-th stage of **Trace** will on average terminate after $2/p_{\mathcal{D}}$ queries to $\mathcal{D}$. Upon termination, **Trace**'s output will be wrong only if it happens that $\mathcal{D}$ replies to an $(\ell, 1 - \tau^\ell)$-probe $\hat{\psi}$ with an m^* satisfying $\mathcal{R}(\hat{m}, m^*) = 1$, which by Corollary B.4.15, Lemma B.4.16, and Bayes' theorem is easily seen to equal $p_{\mathcal{R}}/(p_{\mathcal{D}} + p_{\mathcal{R}})$, which is negligible. ▮

Theorem B.4.18 Under the DBDH assumption for $(\mathbb{G}_1, \mathbb{G}_2)$, the multi-user **Trace** algorithm from Section B.4.4 has a negligible traceability error.

Proof: Let $\hat{\omega} = (\hat{\omega}^{(1)}, \ldots, \hat{\omega}^{(v)})$ be the "traitor codeword" recovered at the end of the publicly traceable phase of **Trace** (*cf.* Section B.4.4). By the union bound, Lemma B.4.17 implies that $\hat{\omega}$ will be correct in all positions ℓ where all traitors show the same bit, except with negligible probability. By the collusion resistance of the code $\mathcal{C}$ underlying the key assignment of **Setup**, the codeword-tracing algorithm $\mathcal{T}$ (*cf.* Definition B.2.4) will then be able to tie such traitor codeword $\hat{\omega}$ to the identity of one of the traitors in T (except with negligible probability ε), as required. ▮

Remark B.4.19 As noted above, by employing AONT's, the security and tracing capabilities of our multi-user scheme follow almost directly from those of the embedded "special" sub-scheme. In fact, even if we were to suppress the shared sub-scheme (*e.g.*, by setting $W_j = M_j$, for $j = 1, \ldots, v, j \neq \ell$), the multi-user scheme would still be secure and tracing would still be possible (thanks also to the random rotation of the special position ℓ between 1 and v). Using the shared sub-scheme, however, reinforces the semantic security of the scheme, though at the cost of a greater computational load, due to the larger number of pairing computations needed for encryption and decryption.

B.5 Space and Time Parameters in a Concrete Instantiation

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

Another important issue for a concrete instantiation is the rate at which encrypted content can be processed. In our scheme, decryption requires one paring per 1024 bits of content, which, using the PBC Library [Lyn] on a desktop PC, takes approximately 16 msec. However, in our context, the pairings to be computed all have one of their two input-points in common: as reported in [BLS04], pre-processing in similar settings more than halves the computation time, so that one easily gets in the order of 128 pairings/sec, corresponding to a near-CD-quality audio rate of 128 Kbits/sec. More specialized software implementations [BGhCS04] of the pairing operation can further reduce its computational cost to around 3 msec; whereas hardware implementations, even under conservative assumptions on the hardware architecture [KMPB05], can obtain running time below 1 msec, attaining the 1Mbits/sec data rate needed for VCD-quality video.

B.6 Conclusion

We present the first public-key traitor tracing scheme with efficient black-box tracing and optimal transmission rate. Our treatment improves the standard modeling of black-box tracing by additionally accounting for pirate strategies that attempt to escape tracing by purposedly rendering the transmitted content at lower quality (*e.g.* by dropping every other frame from the decrypted video-clip, or skipping few seconds from the original audio file). We also point out and resolve an issue in the black-box traitor tracing mechanism of both the previous schemes in this setting [KY02c, CPP05a]. Our construction is based on the decisional bilinear Diffie-Hellman assumption, and additionally provides the same features of public traceability as (a repaired version of) [CPP05a], which is less efficient and requires non-standard assumptions for bilinear groups.

[8] [PSNT06b] actually employs IPP codes, but similar considerations on code length and message size apply to such codes as well.

B.7 Appendix

B.7.1 Bilinear Maps and Intractability Assumptions

Bilinear Maps

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

Bilinearity: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and all $a, b \in \mathbb{Z}q$;

Non-degeneracy: The map does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to the unit in $\mathbb{G}_2$;

Computable: There is an efficient algorithm to compute $e(P, Q)$ for any elements $P, Q \in \mathbb{G}_1$.

A bilinear map satisfying the three above properties is said to be an *admissible bilinear map*. Throughout the paper, we view $\mathbb{G}_1$ as an additive group and $\mathbb{G}_2$ as a multiplicative group. We remark that since $\mathbb{G}_1, \mathbb{G}_2$ are groups of prime order and e is non-degenerated, $e(P, P)$ generates $\mathbb{G}_2$ whenever P generates $\mathbb{G}_1$. It follows that $e(P, \cdot)$ is an isomorphism from $\mathbb{G}_1$ into $\mathbb{G}_2$. Typical examples of constructions of admissible bilinear maps satisfying the above properties are based on the modified Weil and Tate pairings (*cf. e.g.,* [BF99b]).

Assumptions for Our Scheme

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

Definition B.7.1 [DBDH Assumption] The DBDH problem is $\varepsilon_{\mathsf{DBDH}}$-hard in $(\mathbb{G}_1, \mathbb{G}_2)$ if, for all probabilistic polynomial-time algorithms $\mathcal{A}$, we have

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

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

Assumption for the Schemes of [KY02c]

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

Assumptions for the Schemes of [CPP05a]

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

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

B.7.2 The Public-Key Traitor Tracing Schemes of [KY02c] and [CPP05a]

The Two-User Sub-Scheme of [KY02c]

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

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

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

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

KeyDer: The security manager selects two linearly independent vectors $(\alpha_0, \beta_0), (\alpha_1, \beta_1) \in \mathbb{Z}_q^2$ such that $\alpha_\sigma + a\beta_\sigma = c \bmod q$, for $\sigma \in \{0,1\}$. This implies: $Z = cP = \alpha_\sigma P + \beta_\sigma Q$, for $\sigma \in \{0,1\}$. The secret key of user σ is then set to be $\mathsf{sk}_\sigma \doteq (\alpha_\sigma, \beta_\sigma)$, for $\sigma \in \{0,1\}$.

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

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

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

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

In [KY02c], the authors show that the above two-user scheme is secure and traceable (for up to 1 traitor) in the sense of Definitions B.3.3 and B.3.4 under the DDH assumption (*cf.* Appendix B.7.1).

The Multi-User Scheme of [KY02c]

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

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

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

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

[9] Even though [KY02c] used the multiplicative notation, we use here the additive notation for the sake of consistency with the rest of the paper (*cf.* Footnote 10).

[10] Even though [KY02c] used the multiplicative notation, we use here the additive notation for the sake of consistency with the rest of the paper. Notice, however, that $\mathbb{G}$ should not be identified with the group $\mathbb{G}_1$ used elsewhere in this paper, and in particular $\mathbb{G}$ should not be equipped with a bilinear map, for that would violate the required hardness of the DDH problem in $\mathbb{G}$.

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

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

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

Trace: To trace a decoder $\mathcal{D}$ back to the identity of one of the traitors, the security manager prepares an illegal ciphertext $\hat{\psi} \doteq (\hat{\psi}_1, \ldots, \hat{\psi}_v)$, where each $\hat{\psi}_j$ is constructed as in the tracing algorithm from Appendix B.7.2 (*i.e.,* $\hat{\psi}_j \doteq \langle k'_j P_j, k_j Q_j, \hat{m}_j \rangle$, for random $k_j, k'_j \xleftarrow{\$} \mathbb{Z}q$ and $\hat{m}_j \xleftarrow{\$} \{0,1\}^\kappa$). Let $m \doteq (m^{(1)} \| \ldots \| m^{(v)})$ be the plaintext output by $\mathcal{D}$ when fed with the ciphertext $\hat{\psi}$.

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

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

The Two-User Sub-Scheme of [CPP05a]

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

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

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

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

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

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

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

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

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

In [CPP05a], the above two-user scheme is proven secure and traceable (for up to 1 traitor) in the sense of Definitions B.3.3 and B.3.4 under two non-standard assumptions for bilinear groups, respectively called DBDH^2-E and DBDH^1-M in [CPP05a] (*cf.* Appendix B.7.1).

The Multi-User Scheme of [CPP05a]

We now describe the multi-user scheme[11] of [CPP05a], which is based on the use of bilinear maps. The key difference from the multi-user scheme of [KY02c] is the idea of *proxy quantity*: the security manager selects the master secret key roughly as in [KY02c], but now some secret information is removed from the users' secret keys and a derived value (the proxy quantity) is lifted to the public key.

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

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

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

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

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

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

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

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

[11] In [CPP05a], the authors present two schemes with the same parameters. For conciseness, here we only report the second scheme, which was claimed to also support local public traceability.

Trace: Although [CPP05a] present a tracing algorithm only for their two-user scheme, the authors suggested therein that their multi-user scheme inherits the tracing capabilities of [KY02c]. In particular, we sketch here the obvious necessary modifications to the **Trace** algorithm in Appendix B.7.2: the illegal ciphertext has the form $\hat{\psi} \doteq \langle k'P, k^2Q, (\hat{m}_1, \ldots, \hat{m}_v)\rangle$, where $k, k' \xleftarrow{\$} \mathbb{Z}q$, and each $\hat{m}_j$ is random in $\{0,1\}^\kappa$; and the "traitor codeword" $\hat{\omega} \doteq (\hat{\omega}^{(1)}, \ldots, \hat{\omega}^{(v)})$, is constructed from $\mathcal{D}$'s response $m \doteq (m^{(1)} \| \ldots \| m^{(v)})$ by defining each $\hat{\omega}^{(j)} \in \{0, 1, \text{'?'}\}$ as in the tracing for the two-user scheme (*i.e.,* $\hat{\omega}^{(j)} \doteq \sigma_j$ if $m^{(j)} = \hat{m}_j \oplus H(e(\alpha_{j,\sigma_j}P, P)^{k'^2} \cdot e(\beta_{j,\sigma_j}P, Q)^{k^2})$ (for $\sigma_j = \{0,1\}$), or $\hat{\omega}^{(j)} \doteq$ '?' otherwise).

B.7.3 On the Query Complexity of Black-Box Tracing in [KY02c]

Appendix B.7.2 reports the multi-user scheme of [KY02c], which includes a black-box tracing algorithm making a single query to the pirate decoder $\mathcal{D}$. Below we show that such algorithm is broken, and we present a simple traitor strategy that allows a coalition of just $2 < t$ users to escape tracing with probability 1. We also propose a variation of their black-box tracing algorithm, which requires v queries but is successful in tracing up to the desired threshold of traitors, thus suggesting that the query complexity of black-box tracing in [KY02c] is higher than what claimed therein.

A Simple Untraceable Traitor Strategy

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

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

Notice that $\mathcal{D}$ perfectly decrypts ciphertext distributed according to $\mathsf{Encaps}(\mathsf{pk}, \cdot)$ since, by correctness of decryption, $\mathcal{D}$'s "integrity" check will always pass on a valid ciphertext. Moreover, $\mathcal{D}$ escapes tracing with probability 1, since the **Trace** algorithm of [KY02c] prepares the invalid ciphertext $\hat{\psi}$ by concatenating invalid ciphertexts $\hat{\psi}_j$ for each of the v components of the scheme. This will result in different decryptions of $\hat{\psi}_1$ under $(\alpha_{1,0}, \beta_{1,0})$ and $(\alpha_{1,1}, \beta_{1,1})$, and thus $\mathcal{D}$ will reply with a plaintext containing no information about the identities of the traitors.

The Fix

The problem with the **Trace** algorithm of [KY02c] is that it implicitly assumed that pirate decoders would decrypt each component of the ciphertext independently from each other, which clearly does not need to be the case. Bearing this in mind, the fix is immediate: it suffices for **Trace** to iteratively query the decoder with v ciphertexts, each constructed to be invalid in just one component, but valid elsewhere. Now, the independence of the v component sub-schemes implies that $\mathcal{D}$ will be unable to tell valid and invalid ciphertexts apart, unless it possesses both user-keys for the single sub-scheme "under testing." As a consequence, **Trace** will end up extracting a traitor codeword from $\mathcal{D}$ with at most t unreadable marks '?', and thus the tracing algorithm $\mathcal{T}(\cdot,\cdot)$ of the collusion-secure code $\mathcal{C}$ will successfully recover the identity of one of the traitor (with probability $1 - \varepsilon$).

Consequences for the Multi-User Scheme of [CPP05a]

Being based on the techniques of [KY02c], the multi-user scheme of [CPP05a] inherits the problem pointed out in Appendix B.7.3. As it turns out, however, in this case the consequences are more severe. In particular, the easy fix that we proposed for the scheme of [KY02c] in Appendix B.7.3 does not apply: interestingly, the higher correlation between the parameters used in the v components of the scheme of [CPP05a], which proved crucial to attain optimal transmission rate, at the same time poses a serious impediment to black-box tracing.

Indeed, ciphertexts in the multi-user scheme of [CPP05a] (*cf.* Appendix B.7.2) have the form $\psi \doteq \langle kP,\ k^2Q,\ (W_1,\ \ldots,\ W_v)\rangle$, in which the same "randomization" values kP, k^2Q are used for all the v two-user sub-schemes. Hence, it is not possible to make the ciphertext invalid in just one component, while preserving its validity in the remaining $(v-1)$ ones (which was the idea behind our fix in Appendix B.7.3). Therefore, it seems that the scheme of [CPP05a], as given, does not support black-box tracing. Since the notion of local public traceability is only meaningful in the black-box setting, this also voids the claimed traceability features of the multi-user scheme of [CPP05a].

To salvage black-box tracing and local public traceability, one could modify the scheme of [CPP05a] and revert to the "parallel" composition of sub-schemes (exactly as in [KY02c]), thus "undoing" the optimization that enabled short ciphertexts. The resulting scheme, however, would just be a variant of [KY02c] with the same parameters, but with the additional need of bilinear maps and reliance on non-standard bilinear-related assumptions.

As a result, it seems appropriate to regard the multi-user scheme of [CPP05a] as a scheme with optimal transmission rate, but with only non-black-box tracing and no public traceability features.

B.7.4 Proofs from Section B.4.5

Proof of Lemma B.4.10

Lemma. If the multi-user scheme without AONT pre-processing is secure w.r.t. indistinguishability under chosen-plaintext attack (*cf.* Theorem B.4.1), then the multi-user scheme with AONT pre-processing is secure w.r.t. the same notion.

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

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

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

Proof of Lemma B.4.11

Lemma. If the two-user scheme in Section B.4.1 is secure w.r.t. indistinguishability under chosen-plaintext attack (*cf.* Theorem B.4.1), then the multi-user scheme in Section B.4.4 is secure w.r.t. the same notion. **Proof:** For the sake of clarity, in the security proof, we will follow the structural approach advocated in [Sho04]. Starting from the actual attack scenario (as defined in Definition B.3.3), we consider a sequence of hypothetical games, all defined over the same underlying probability space. In each game, the adversary's view is obtained in a slightly different way, but its distribution is maintained (computationally) indistinguishable across the games. In the last game, it will be clear that the adversary has (at most) a negligible advantage; by the indistinguishability of any two consecutive games, it will follow that also in the original game the adversary's advantage is negligible.

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

Game $\mathbf{G}_0$. Define $\mathbf{G}_0$ to be the original game as described in Definition B.3.3.

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

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

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

Claim B.7.2 $[1]\ |\Pr[S_0] - \Pr[S_1]| \leq 2\varepsilon^{(1)}$, where $\varepsilon^{(1)}$ is the advantage of some efficient adversary attacking the security of the 2-user scheme from Section B.4.1.

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

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

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

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

Challenge: $\mathcal{A}_1$ outputs two messages $m_0 \doteq (m_0^{(1)} \| \ldots \| m_0^{(v)}), m_1 \doteq (m_1^{(1)} \| \ldots \| m_1^{(v)})$ on which it wishes to be challenged, along with some state τ to be passed to $\mathcal{A}_2$. Now $\mathcal{B}^{(1)}$, in turn,

[12] Notice that the value of ℓ is fixed within this proof, since we fixed the randomness for Encaps across the games.

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

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

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

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

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

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

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

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

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

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

$$\boxed{W_{i-1} \xleftarrow{\$} \{0,1\}^\kappa}$$

Claim B.7.3 $[i]$ $|\Pr[S_{i-1}] - \Pr[S_i]| \leq 2\varepsilon^{(i)}$, where $\varepsilon^{(i)}$ is the advantage of some efficient adversary attacking the security of the 2-user scheme from Section B.4.1.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\boxed{W_i \xleftarrow{\$} \{0,1\}^\kappa}$$

Claim B.7.4 $[i]$ $|\Pr[S_{i-1}] - \Pr[S_i]| \leq 2\varepsilon^{(i)}$, where $\varepsilon^{(i)}$ is the advantage of some efficient adversary attacking the security of the 2-user scheme from Section B.4.1.

The proof of this is by a reduction argument completely analogous to the one used in proving the claims for the cases $2 \leq i \leq \ell$, the only difference being a notational one, since now the

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

reduction will embed the challenge from the 2-user scheme into the component W_i (rather than W_{i-1}).[14]

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

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

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

$$\Pr[S_0] \leq \frac{1}{2} + 2v\varepsilon_{\text{ind}}^{\text{2-user}},$$

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

B.7.5 A Comparison with [BSW06b, BW06b]

Recently, Boneh *et al.* [BSW06b, BW06b] proposed traitor tracing schemes that withstand any number of traitors (*full traceability*), while requiring a sub-linear ciphertext length ($O(\sqrt{n})$). While the schemes of [BSW06b, BW06b] are the most efficient ones supporting full collusion, they are not well suited for the more practical case of small number of traitors (say, logarithmic in the size of the entire user population). Indeed, in this case, the ciphertext in these schemes still contains $O(\sqrt{n})$ elements. In our scheme, assuming the number of traitors t is logarithmic in the number of users n, the ciphertext has poly-logarithmic length $v = O(t^2(\log n + \log \frac{1}{\varepsilon})) = O(\log^3 n)$, which is asymptotically superior to the $O(\sqrt{n})$-ciphertexts of [BSW06b, BW06b].

More importantly, the tracing algorithms of [BSW06b, BW06b] require $O(n^2)$ decryption queries to the pirate decoder, whereas our scheme employs $O(v) = O(\log^3 n)$ decryption queries, and is completely parallelizable.

In brief, the schemes of [BSW06b, BW06b] are preferable in case of full collusions, whereas our scheme has advantages in term of efficiency and of complexity of black-box tracing when the number of traitors is logarithmic.

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

Appendix C

Hardness of k-LWE and Applications in Traitor Tracing

CRYPTO 2014

[LPSS14] with San Ling, Damien Stahlé and Ron Steinfeld

Abstract : *We introduce the* k-LWE problem*, a Learning With Errors variant of the k-SIS problem. The Boneh-Freeman reduction from SIS to k-SIS suffers from an exponential loss in k. We improve and extend it to an LWE to k-LWE reduction with a polynomial loss in k, by relying on a new technique involving trapdoors for random integer kernel lattices. Based on this hardness result, we present the first algebraic construction of a traitor tracing scheme whose security relies on the worst-case hardness of standard lattice problems. The proposed* LWE *traitor tracing is almost as efficient as the* LWE *encryption. Further, it achieves public traceability, i.e., allows the authority to delegate the tracing capability to "untrusted" parties. To this aim, we introduce the notion of* projective sampling family *in which each sampling function is keyed and, with a projection of the key on a well chosen space, one can simulate the sampling function in a computationally indistinguishable way. The construction of a projective sampling family from* k-LWE *allows us to achieve public traceability, by publishing the projected keys of the users. We believe that the new lattice tools and the projective sampling family are quite general that they may have applications in other areas.*

C.1 Introduction

Since the pioneering work of Ajtai [Ajt96a], there have been a number of proposals of cryptographic schemes with security provably relying on the worst-case hardness of standard lattice problems, such as the decision Gap Shortest Vector Problem with polynomial gap (see the surveys [MR09, Reg10]). These schemes enjoy unmatched security guarantees: Security relies on *worst-case* hardness assumptions for problems expected to be *exponentially hard* to solve (with respect to the lattice dimension n), even with quantum computers. At the same time, they often enjoy great asymptotic efficiency, as the basic operations are matrix-vector multiplications in dimension $\widetilde{O}(n)$ over a ring of cardinality $\leq \mathcal{P}oly(n)$. A breakthrough result in that field was the introduction of the Learning With Errors problem (LWE) by Regev [Reg05, Reg09], who showed it to be at least as hard as worst-case lattice problems and exploited it to devise an elementary encryption scheme. Gentry et al. showed in [GPV08] that Regev's scheme may be

adapted so that a master can generate a large number of secret keys for the same public key. As a result, the latter encryption scheme, called dual-Regev, can be naturally extended into a multi-receiver encryption scheme. In the present work, we build traitor tracing schemes from this dual-Regev LWE-based encryption scheme.

Traitor tracing. A traitor tracing scheme is a multi-receiver encryption scheme where malicious receiver coalitions aiming at building pirate decryption devices are deterred by the existence of a tracing algorithm: Using the pirate decryption device, the tracing algorithm can recover at least one member of the malicious coalition. Such schemes are particularly well suited for fighting copyright infringement in the context of commercial content distribution (e.g., Pay-TV, subscription news websites, etc). Since their introduction by Chor et al. [CFN94a], much work has been devoted to devising efficient and secure traitor tracing schemes. The most desirable schemes are fully collusion resistant: they can deal with arbitrarily large malicious coalitions. But, unsurprisingly, the most efficient schemes are in the bounded collusion model where the number of malicious users is limited. The first non-trivial fully collusion resistant scheme was proposed by Boneh et al. [BSW06b]. However, its ciphertext size is still large ($\Omega(\sqrt{N})$, where N is the total number of users) and it relies on pairing groups of composite order. Very recently, Boneh and Zhandry [BZ14] proposed a fully collusion resistant scheme with poly-log size parameters. It relies on indistinguishability obfuscation [GGH$^+$13b], whose security foundation remains to be studied, and whose practicality remains to be exhibited. In this paper, we focus on the bounded collusion model. The Boneh-Franklin scheme [BF99a] is one of the earliest algebraic constructions but it can still be considered as the reference algebraic transformation from the standard ElGamal public key encryption into traitor tracing. This transformation induces a linear loss in efficiency, with respect to the maximum number of traitors. The known transformations from encryption to traitor tracing in the bounded collusion model present at least a linear loss in efficiency, either in the ciphertext size or in the private key size [BF99a, NP00, KY02c, Sir07b, BP08, BN08b]. We refer to [KP10] for a detailed introduction to this rich topic. Also, in Appendix C.7.1, we give a short overview of traitor tracing schemes with their properties, in particular the public traceability.

Our Contributions. We describe the first algebraic construction of a public-key lattice-based traitor tracing scheme. It is semantically secure and enjoys public traceability. The security relies on the hardness of LWE, which is known to be at least as hard as standard worst-case lattice problems [Reg09, Pei09, BLP$^+$13].

The scheme is the extension, described above, of the dual-Regev LWE-based encryption scheme from [GPV08] to a multi-receiver encryption scheme, where each user has a different secret key. In the case of traitor tracing, several keys may be leaked to a traitor coalition. To show that we can trace the traitors, we extend the LWE problem and introduce the k-LWE problem, in which k hint vectors (the leaked keys) are given out.

Intuitively, k-LWE asks to distinguish between a random vector $\vec{t}$ close to a given lattice Λ and a random vector $\vec{t}$ close to the orthogonal subspace of the span of k given short vectors belonging to the dual Λ^* of that lattice. Even if we are given $(\vec{b}_i^*)_{i\leq k}$ small in Λ^*, computing the inner products $\langle \vec{b}_i^*, \vec{t}\rangle$ will not help in solving this problem, since they are small and distributed identically in both cases. The k-LWE problem can be interpreted as a dual of the k-SIS problem introduced by Boneh and Freeman [BF11], which intuitively requests to find a short vector in Λ^* that is linearly independent with the k given short vectors of Λ^*. Their reduction from SIS to k-SIS can be adapted to the LWE setup, but the hardness loss incurred by the reduction is gigantic. We propose a significantly sharper reduction from LWE$_\alpha$ to k-LWE$_\alpha$. This improved reduction requires a new lattice technique: the equivalent for kernel lattices of Ajtai's simultaneous sampling of a random q-ary lattice with a short basis [Ajt99] (see also Lemma C.2.2). We adapt the Micciancio-Peikert framework from [MP12] to sampling a Gaussian $X \in \mathbb{Z}^{m\times n}$

along with a short basis for the lattice $\ker(X) = \{\vec{b} \in \mathbb{Z}^m : \vec{b}^t X = \vec{0}\}$. Kernel lattices also play an important role in the re-randomization analysis of the recent lattice-based multilinear map scheme of Garg et al. [GGH13a], and we believe that our new trapdoor generation tool for such lattices is likely find additional applications in future. We also remark that our technique can be adapted to the SIS to k-SIS reduction. We thus solve the open question left by Boneh and Freeman of improving their reduction [BF11]: from an exponential loss in k to a polynomial loss in k. Consequently, their linearly homomorphic signatures and ordinary signature schemes enjoy much better efficiency/security trade-offs.

Our construction of a traitor tracing scheme from k-LWE can be seen as an additive and noisy variant of the (black-box) Boneh-Franklin traitor tracing scheme [BF99a]. While the Boneh-Franklin scheme is transformed from the ElGamal encryption with a linear loss (in the maximum number of traitors) in efficiency, our scheme is almost as efficient as standard LWE-based encryption, as long as the maximum number of traitors is bounded below $n/(c \log n)$, where n is the LWE dimension determined by the security parameter, and c is a constant. The full functionality of black-box tracing in both the Boneh-Franklin scheme and ours are of high complexity as they both rely on the black-box confirmation: given a superset of the traitors, it is guaranteed to find at least one traitor and no innocent suspect is incriminated. Boneh and Franklin left the improvement of the black-box tracing as an interesting open problem. We show that in lattice setting, the black-box tracing can be accelerated by running the tracing procedure in parallel on untrusted machines. This is a direct consequence of the property of public traceability, i.e., the possibility of running tracing procedure on public information, that our scheme enjoys. We note that almost all traitor tracing systems require that the tracing key must be kept secret. Some schemes [CPP05a, PSNT06a, BW06a, BZ14] achieve public traceability and some others achieve a stronger notion than public traceability, namely the non-repudation, but the setup in these schemes require some interactive protocol between the center and each user such as a secure 2-party computation protocol in [Pfi96], a commitment protocol in [PW97], an oblivious polynomial evaluation in [WHI01, KWHI01, KY02a].

To obtain public traceability and inspired from the notion of projective hash family [CS02], we introduce a new notion of *projective sampling family* in which each sampling function is keyed and, with a projection of the key on a well chosen space, one can simulate the sampling function in a computationally indistinguishable way. The construction of a set of projective sampling families from k-LWE allows us to publicly sample the tracing signals.

Independently, our new lattice tools may have applications in other areas. The k-LWE problem has a similar flavour to the Extended-LWE problem from [OPW11]. It would be interesting to exhibit reductions between these problems. On a closely-related topic, it seems our sampling of a random Gaussian integer matrix X together with a short basis of $\ker(X)$ is compatible with the hardness proof of Extended-LWE from [BLP$^+$13]. In particular, it should be possible to use it as an alternative to [BLP$^+$13, Def 4.5] in the proof of [BLP$^+$13, Le 4.7], to show that Extended-LWE remains hard with many hints independently sampled from discrete Gaussians.

C.2 Preliminaries

If x is a real number, then $\lfloor x \rceil$ is the closest integer to x (with any deterministic rule in case x is half an odd integer). All vectors will be denoted in bold. By default, our vectors are column vectors. We let $\langle \cdot, \cdot \rangle$ denote the canonical inner product. For q prime, we let $\mathbb{Z}_q$ denote the field of integers modulo q. For two matrices A, B of compatible dimensions, we let $(A|B)$ and $(A\|B)$ respectively denote the horizontal and vertical concatenations of A and B. For $A \in \mathbb{Z}_q^{m \times n}$, we define $\mathrm{Im}(A) = \{A\vec{s} : \vec{s} \in \mathbb{Z}_q^n\} \subseteq \mathbb{Z}_q^m$. For $X \subseteq \mathbb{Z}_q^m$, we let $\mathrm{Span}(X)$ denote the set of all

linear combinations of elements of X. We let $X^\perp$ denote the linear subspace $\{\vec{b} \in \mathbb{Z}_q^m : \forall \vec{c} \in X, \langle \vec{b}, \vec{c} \rangle = 0\}$. For a matrix $S \in \mathbb{R}^{m \times n}$, we let $\|S\|$ denote the norm of its longest column. If S is full column-rank, we let $\sigma_1(S) \geq \ldots \geq \sigma_n(S)$ denote its singular values. We let $\mathbb{T}$ denote the additive group $\mathbb{R}/\mathbb{Z}$.

If D_1 and D_2 are distributions over a countable set X, their statistical distance $\frac{1}{2}\sum_{x \in X} |D_1(x) - D_2(x)|$ will be denoted by $\Delta(D_1, D_2)$. The statistical distance is defined similarly if X is measurable. If X is of finite weight, we let $U(X)$ denote the uniform distribution over X. For any invertible $S \in \mathbb{R}^{m \times m}$ and $\vec{c} \in \mathbb{R}^m$, we define the function $\rho_{S,\vec{c}}(\vec{b}) = \exp(-\pi \|S^{-1}(\vec{b} - \vec{c})\|^2)$. For $S = sI_m$, we write $\rho_{s,\vec{c}}$, and we omit the subscripts S and $\vec{c}$ when $S = I_m$ and $\vec{c} = \vec{0}$. We let ν_α denote the one-dimensional Gaussian distribution with standard deviation α.

C.2.1 Euclidean lattices and discrete Gaussian distributions

A lattice is a set of the form $\{\sum_{i \leq n} x_i \vec{b_i} : x_i \in \mathbb{Z}\}$ where the $\vec{b_i}$'s are linearly independent vectors in $\mathbb{R}^m$. In this situation, the $\vec{b_i}$'s are said to form a basis of the n-dimensional lattice. The n-th minimum $\lambda_n(L)$ of an n-dimensional lattice L is defined as the smallest r such that the n-dimensional closed hyperball of radius r centered in $\vec{0}$ contains n linearly independent vectors of L. The smoothing parameter of L is defined as $\eta_\varepsilon(L) = \min\{r > 0 : \rho_{1/r}(\widehat{L} \setminus \vec{0}) \leq \varepsilon\}$ for any $\varepsilon \in (0,1)$, where $\widehat{L} = \{\vec{c} \in \mathrm{Span}(L) : \vec{c}^t \cdot L \subseteq \mathbb{Z}\}$ is the dual lattice of L. It was proved in [MR07, Le. 3.3] that $\eta_\varepsilon(L) \leq \sqrt{\ln(2n(1+1/\varepsilon))/\pi} \cdot \lambda_n(L)$ for all $\varepsilon \in (0,1)$ and n-dimensional lattices L.

For a lattice $L \subseteq \mathbb{R}^m$, a vector $\vec{c} \in \mathbb{R}^m$ and an invertible $S \in \mathbb{R}^{m \times m}$, we define the Gaussian distribution of parameters L, $\vec{c}$ and S by $D_{L,S,\vec{c}}(\vec{b}) \sim \rho_{S,\vec{c}}(\vec{b}) = \exp(-\pi \|S^{-1}(\vec{b} - \vec{c})\|^2)$ for all $\vec{b} \in L$. When $S = \sigma \cdot I_m$, we simply write $D_{L,\sigma,\vec{c}}$. Note that $D_{L,S,\vec{c}} = S^t \cdot D_{S^{-t}L,1,S^{-t}\vec{c}}$. Sometimes, for convenience, we use the notation $D_{L+\vec{c},S}$ as a shorthand for $\vec{c} + D_{L,S,-\vec{c}}$. Gentry et al. [GPV08] gave an algorithm, referred to as GPV algorithm, to sample from $D_{L,S,\vec{c}}$ when given as input a basis $(\vec{b_i})_i$ of L such that $\sqrt{\ln(2n+4)/\pi} \cdot \max_i \|S^{-t}\vec{b_i}\| \leq 1$ (see Lemma C.8.1).

We extensively use q-ary lattices. The q-ary lattice associated to $A \in \mathbb{Z}_q^{m \times n}$ is defined as $\Lambda^\perp(A) = \{\vec{x} \in \mathbb{Z}^m : \vec{x}^t \cdot A = \vec{0} \bmod q\}$. It has dimension m, and a basis can be computed in polynomial-time from A. For $\vec{u} \in \mathbb{Z}_q^m$, we define $\Lambda_{\vec{u}}^\perp(A)$ as the coset $\{\vec{x} \in \mathbb{Z}^m : \vec{x}^t \cdot A = \vec{u}^t \bmod q\}$ of $\Lambda^\perp(A)$.

C.2.2 Random lattices

We consider the following random lattices, called q-ary Ajtai lattices. They are obtained by sampling $A \hookleftarrow U(\mathbb{Z}_q^{m \times n})$ and considering $\Lambda^\perp(A)$. The following lemma provides a probabilistic bound on the smoothing parameter of $\Lambda^\perp(A)$.

Lemma C.2.1 [Adapted from [GPV08, Le. 5.3]] Let q be prime and m, n integers with $m \geq 2n$ and $\varepsilon > 0$, then $\eta_\varepsilon(\Lambda^\perp(A)) \leq 4q^{\frac{n}{m}}\sqrt{\log(2m(1+1/\varepsilon))/\pi}$, for all except a fraction $2^{-\Omega(n)}$ of $A \in \mathbb{Z}_q^{m \times n}$.

It is possible to efficiently sample a close to uniform A along with a short basis of $\Lambda^\perp(A)$ (see [Ajt99, AP11, Pei10, MP12]).

Lemma C.2.2 [Adapted from [AP11, Th. 3.1]] There exists a ppt algorithm that given $n, m, q \geq 2$ as inputs samples two matrices $A \in \mathbb{Z}_q^{m \times n}$ and $T \in \mathbb{Z}^{m \times m}$ such that: the distribution of A is within statistical distance $2^{-\Omega(n)}$ from $U(\mathbb{Z}_q^{m \times n})$; the rows of T form a basis of $\Lambda^\perp(A)$; each row of T has norm $\leq 3mq^{n/m}$.

For $A \in \mathbb{Z}_q^{m\times n}$, $S \in \mathbb{R}^{m\times m}$ invertible, $\vec{c} \in \mathbb{R}^m$ and $\vec{u} \in \mathbb{Z}_q^n$, we define the distribution $D_{\Lambda_{\vec{u}}^{\perp}(A),S,\vec{c}}$ as $\vec{\vec{c}} + D_{\Lambda^{\perp}(A),S,-\vec{c}+\vec{c}}$, where $\vec{\vec{c}}$ is any vector of $\mathbb{Z}^m$ such that $\vec{\vec{c}}^t \cdot A = \vec{u}^t \bmod q$. A sample $\vec{x}$ from $D_{\Lambda_{\vec{u}}^{\perp}(A),S}$ can be obtained using the GPV algorithm along with the short basis of $\Lambda^{\perp}(A)$ provided by Lemma C.2.2. Boneh and Freeman [BF11] showed how to efficiently obtain the residual distribution of $(A, \vec{x})$ without relying on Lemma C.2.2.

Theorem C.2.3 [Adapted from [BF11, Th. 4.3]] Let $n, m, q \geq 2$, $k \geq 0$ and $S \in \mathbb{R}^{m\times m}$ be such that $m \geq 2n$, q is prime with $q > \sigma_1(S)\cdot\sqrt{2\log(4m)}$, and $\sigma_m(S) = q^{\frac{n}{m}}\cdot\max(\Omega(\sqrt{n\log m}), 2\sigma_1(S)^{\frac{k}{m}})$. Let $\vec{u}_1, \ldots, \vec{u}_k \in \mathbb{Z}_q^n$ and $\vec{c}_1, \ldots, \vec{c}_k \in \mathbb{R}^m$ be arbitrary. Then the residual distributions of the tuple $(A, \vec{x}_1, \ldots, \vec{x}_k)$ obtained with the following two experiments are within statistical distance $2^{-\Omega(n)}$.

$$\texttt{Exp}_0 : \qquad A \hookleftarrow U(\mathbb{Z}_q^{m\times n}); \qquad \forall i \leq k : \vec{x}_i \hookleftarrow D_{\Lambda_{\vec{u}_i}^{\perp}(A),S,\vec{c}_i} \ .$$

$$\texttt{Exp}_1 : \quad \forall i \leq k : \vec{x}_i \hookleftarrow D_{\mathbb{Z}^m,S,\vec{c}_i}; \quad A \hookleftarrow U\left(\mathbb{Z}_q^{m\times n} | \forall i \leq k : \vec{x}_i^t \cdot A = \vec{u}_i^t \bmod q\right).$$

This statement generalizes [BF11, Th. 4.3] in three ways. First, the latter corresponds to the special case corresponding to taking all the $\vec{u}_i$'s and $\vec{c}_i$'s equal to $\vec{0}$. This generalization does not add any extra complication in the proof of [BF11, Th. 4.3], but is important for our constructions. Second, the condition on m is less restrictive (the corresponding assumption in [BF11, Th. 4.3] is that $m \geq \max(2n\log q, 2k)$). To allow for such small values of m, we refine the bound on the smoothing parameter of the $\Lambda^{\perp}(A)$ lattice (namely, we use Lemma C.2.1). Third, we allow for a non-spherical Gaussian distribution, which seems needed in our generalized Micciancio-Peikert trapdoor gadget used in the reduction from LWE to k-LWE in Section C.3.2.

We also use the following result on the probability of the Gaussian vectors $\vec{x}_i$ from Theorem C.2.3 being linearly independent over $\mathbb{Z}_q$.

Lemma C.2.4 [Adapted from [BF11, Le. 4.5]] With the notations and assumptions of Theorem C.2.3, the k vectors $\vec{x}_1, \ldots, \vec{x}_k$ sampled in $\texttt{Exp}_0$ and $\texttt{Exp}_1$ are linearly independent over $\mathbb{Z}_q$, except with probability $2^{-\Omega(n)}$.

C.2.3 Rényi Divergence

We use Rényi Divergence (RD) in our analysis, relying on techniques developed in [LPR13, LSS14a, LSS14b]. For any two probability distributions P and Q such that the support of P is a subset of the support of Q over a countable domain X, we define the RD (of order 2) by $R(P\|Q) = \sum_{x\in X}\frac{P(x)^2}{Q(x)}$, with the convention that the fraction is zero when both numerator and denominator are zero. We recall that the RD between two offset discrete Gaussians is bounded as follows.

Lemma C.2.5 [[LSS14a, Le. 4.2]] For any n-dimensional lattice $L \subseteq \mathbb{R}^n$ and invertible matrix S, set $P = D_{L,S,\vec{w}}$ and $Q = D_{L,S,\vec{z}}$ for some fixed $\vec{w}, \vec{z} \in \mathbb{R}^n$. If $\vec{w}, \vec{z} \in L$, let $\varepsilon = 0$. Otherwise, fix $\varepsilon \in (0,1)$ and assume that $\sigma_n(S) \geq \eta_\varepsilon(L)$. Then $R(P\|Q) \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2 \cdot \exp\left(2\pi\|\vec{w}-\vec{z}\|^2/\sigma_n(S)^2\right)$.

We use this bound and the fact that the RD between the parameter distributions of two distinguishing problems can be used to relate their hardness, if they satisfy a certain public samplability property.

Lemma C.2.6 [[LSS14b]] Let Φ, Φ' denote two distributions, and $D_0(r)$ and $D_1(r)$ denote two distributions determined by some parameter r. Let P, P' be two decision problems defined as follows:

- P: Assess whether input x is sampled from distribution X_0 or X_1, where

$$X_0 = \{x : r \leftarrow\!\!\shortmid\, \Phi, x \leftarrow\!\!\shortmid\, D_0(r)\},\ X_1 = \{x : r \leftarrow\!\!\shortmid\, \Phi, x \leftarrow\!\!\shortmid\, D_1(r)\}.$$

- P': Assess whether input x is sampled from distribution X'_0 or X'_1, where

$$X'_0 = \{x : r \leftarrow\!\!\shortmid\, \Phi', x \leftarrow\!\!\shortmid\, D_0(r)\},\ X'_1 = \{x : r \leftarrow\!\!\shortmid\, \Phi', x \leftarrow\!\!\shortmid\, D_1(r)\}.$$

Assume that $D_0(\cdot)$ and $D_1(\cdot)$ have the following *public samplability* property: there exists a sampling algorithm S with run-time T_S such that for all r, b, given any sample x from $D_b(r)$ we have:

- $\mathsf{S}(0, x)$ outputs a sample distributed as $D_0(r)$ over the randomness of S.
- $\mathsf{S}(1, x)$ outputs a sample distributed as $D_1(r)$ over the randomness of S.

If there exists a T-time distinguisher $\mathcal{A}$ for problem P with advantage ε, then, for every $\lambda > 0$, there exists an $O(\lambda\varepsilon^{-2} \cdot (T_S + T))$-time distinguisher $\mathcal{A}'$ for problem P' with advantage $\varepsilon' \geq \frac{\varepsilon^3}{8R(\Phi\|\Phi')} - O(2^{-\lambda})$.

C.2.4 Learning with errors

Let $\vec{s} \in \mathbb{Z}_q^n$ and $\alpha > 0$. We define the distribution $A_{\vec{s},\alpha}$ as follows: Take $\vec{a} \leftarrow\!\!\shortmid\, U(\mathbb{Z}_q^n)$ and $e \leftarrow\!\!\shortmid\, \nu_\alpha$, and return $(\vec{a}, \frac{1}{q}\langle\vec{a}, \vec{s}\rangle + e) \in \mathbb{Z}_q^n \times \mathbb{T}$. The *Learning With Errors problem* LWE_α, introduced by Regev in [Reg05, Reg09], consists in assessing whether an oracle produces samples from $U(\mathbb{Z}_q^n \times \mathbb{T})$ or $A_{\vec{s},\alpha}$ for some constant $\vec{s} \leftarrow\!\!\shortmid\, U(\mathbb{Z}_q^n)$. Regev [Reg09] showed that for $q \leq \mathcal{P}oly(n)$ prime and $\alpha \in (\frac{\sqrt{n}}{2q}, 1)$, LWE is (quantumly) not easier than standard worst-case lattice problems in dimension n with approximation factors $\mathcal{P}oly(n)/\alpha$. This hardness proof was partly dequantized in [Pei09, BLP+13], and the requirements that q should be prime and $\mathcal{P}oly(n)$ were waived.

In this work, we consider a variant LWE where the number of oracle samples that the distinguisher requests is a priori bounded. If m denotes that bound, then we will refer to this restriction as $\mathrm{LWE}_{\alpha,m}$. In this situation, the hardness assumption can be restated in terms of linear algebra over $\mathbb{Z}_q$: Given $A \leftarrow\!\!\shortmid\, U(\mathbb{Z}_q^{m\times n})$, the goal is to distinguish between the distributions (over $\mathbb{T}^m$)

$$\frac{1}{q}U\left(\mathrm{Im}(A)\right) + \nu_\alpha^m \quad \text{and} \quad \frac{1}{q}U\left(\mathbb{Z}_q^m\right) + \nu_\alpha^m.$$

Under the assumption that $\alpha q \geq \Omega(\sqrt{n})$, the right hand side distribution is indeed within statistical distance $2^{-\Omega(n)}$ to $U(\mathbb{T}^m)$ (see, e.g., [MR07, Le. 4.1]). The hardness assumption states that by adding to them a small Gaussian noise, the linear spaces $\mathrm{Im}(A)$ and $\mathbb{Z}_q^m$ become computationally indistinguishable. This rephrasing in terms of linear algebra is helpful in the security proof of the traitor tracing scheme. Note that by a standard hybrid argument, distinguishing between the two distributions given one sample from either, and distinguishing between them given Q samples (from the same distribution), are computationally equivalent problems, up to a loss of a factor Q in the distinguishing advantage.

Finally, we will also use a variant of LWE where the noise distribution ν_α is replaced by $D_{q^{-1}\mathbb{Z},\alpha}$, and where $U(\mathbb{T})$ is replaced by $U(\mathbb{T}_q)$ with $\mathbb{T}_q$ being $q^{-1}\mathbb{Z}$ with addition mod 1. This variant, denoted by LWE', was proved in [Pei10] to be no easier than standard LWE (up to a constant factor increase in α).

C.3 New lattice tools

The security of our constructions relies on the hardness of a new variant of LWE, which may be seen as the dual of the k-SIS problem from [BF11].

Definition C.3.1 Let $k \leq m$, $S \in \mathbb{R}^{m\times m}$ invertible and $C = (\vec{c}_1 \| \cdots \| \vec{c}_k) \in \mathbb{R}^{k\times m}$. The (k, S, C)-LWE$_{\alpha,m}$ problem (or (k, S)-LWE if $C = 0$) is as follows: Given $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$, $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$ and $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{c}_i}$ for $i \leq k$, the goal is to distinguish between the distributions (over $\mathbb{T}^{m+1}$)

$$\frac{1}{q} \cdot U\Big(\mathrm{Im}\Big(\frac{\vec{u}^t}{A}\Big)\Big) + \nu_\alpha^{m+1} \quad \text{and} \quad \frac{1}{q} \cdot U\Big(\mathrm{Span}_{i\leq k}\Big(\frac{1}{\vec{x}_i}\Big)^\perp\Big) + \nu_\alpha^{m+1}.$$

The classical LWE problem consists in distinguishing the left distribution from uniform, without the hint vectors $\vec{x}_i^+ = (1\|\vec{x}_i)$. These hint vectors correspond to the secret keys obtained by the malicious coalition in the traitor tracing scheme. Once these hint vectors are revealed, it becomes easy to distinguish the left distribution from the uniform distribution: take one of the vectors $\vec{x}_i^+$, get a challenge sample $\vec{y}$ and compute $\langle \vec{x}_i^+, \vec{y}\rangle \in \mathbb{T}$; if $\vec{y}$ is a sample from the left distribution, then the centered residue is expected to be of size $\approx \alpha \cdot (\sqrt{m}\sigma_1(S) + \|\vec{c}_i\|)$, which is $\ll 1$ for standard parameter settings; on the other hand, if $\vec{y}$ is sampled from the uniform distribution, then $\langle \vec{x}^+, \vec{y}\rangle$ should be uniform. The definition of (k, S)-LWE handles this issue by replacing $U(\mathbb{Z}_q^{m+1})$ by $U(\mathrm{Span}_{i\leq k}(\vec{x}_i^+)^\perp)$.

Sampling $\vec{x}_i^+$ from $D_{\Lambda^\perp((\vec{u}^t\|A)),S,\vec{c}_i}$ may seem more natural than imposing that the first coordinate of each $\vec{x}_i^+$ is 1. Looking ahead, this constraint will prove convenient to ensure correctness of our cryptographic primitives. Theorem C.3.5 below and its proof can be readily adapted to this hint distribution. They may also be adapted to improve the SIS to k-SIS reduction from [BF11]. Setting $C = 0$ is also more natural, but for technical reasons, our reduction from LWE to (k, S, C)-LWE works with unit vectors $\vec{c}_i$. However, we show that for small $\|\vec{c}_i\|$, there exist polynomial time reductions between (k, S, C)-LWE and (k, S)-LWE.

In the proof of the hardness of (k, S)-LWE problem, we rely on a gadget integral matrix G that has the following properties: its first rows have Gaussian distributions, it is unimodular and its inverse is small. Before going to this proof, we shall build such a gadget matrix by extending Ajtai's simultaneous sampling of a random q-ary lattice with a short basis [Ajt99] (see also Lemma C.2.2) to kernel lattices. More precisely, we adapt the Micciancio-Peikert framework [MP12] to sampling a Gaussian $X \in \mathbb{Z}^{m\times n}$ along with a short basis for the lattice $\ker(X) = \{\vec{b} \in \mathbb{Z}^m : \vec{b}^t X = \vec{0}\}$.

C.3.1 Sampling a Gaussian X with a small basis of $\ker(X)$

The Micciancio-Peikert construction [MP12] relies on a *leftover hash lemma* stating that with overwhelming probability over $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$ and for a sufficiently large σ, the distribution of $A^t \cdot D_{\mathbb{Z}^m,\sigma} \bmod q$ is statistically close to $U(\mathbb{Z}_q^n)$. We use a similar result over the integers, starting from a Gaussian $X \in \mathbb{Z}^{m\times n}$ instead of a uniform $A \in \mathbb{Z}_q^{m\times n}$. The proof of the following lemma relies on [AR13], which improves over a similar result from [AGHS13]. The result would be neater with $\sigma_2 = \sigma_1$, but, unfortunately, we do not know how to achieve it. The impact of this drawback on our results and constructions is mostly cosmetic.

Lemma C.3.2 Let $m \geq n \geq 100$ and $\sigma_1, \sigma_2 > 0$ satisfying $\sigma_1 \geq \Omega(\sqrt{mn\log m})$, $m \geq \Omega(n\log(\sigma_1 n))$ and $\sigma_2 \geq \Omega(n^{5/2}\sqrt{m}\sigma_1^2\log^{3/2}(m\sigma_1))$. Let $X \hookleftarrow D_{\mathbb{Z},\sigma_1}^{m\times n}$. There exists a ppt algorithm that takes $n, m, \sigma_1, \sigma_2, X$ and $\vec{c} \in \mathbb{Z}^n$ as inputs and returns $\vec{x} \in \mathbb{Z}^n, \vec{r} \in \mathbb{Z}^m$ such that $\vec{x} = \vec{c} + X^t\vec{r}$ with $\|\vec{r}\| \leq O(\sigma_2/\sigma_1)$, with probability $1 - 2^{-\Omega(n)}$, and

$$\Delta\big((X, \vec{x}), D_{\mathbb{Z},\sigma_1}^{m\times n} \times D_{\mathbb{Z}^n,\sigma_2,\vec{c}}\big) \leq 2^{-\Omega(n)}.$$

We now adapt the trapdoor construction from [MP12] to kernel lattices.

Theorem C.3.3 Let $n, m_1, \sigma_1, \sigma_2$ be as above, and $m_2 \geq m_1$ bounded as $n^{O(1)}$. There exists a ppt algorithm that given n, m_1, m_2 (in unary), σ_1 and σ_2, returns $X_1 \in \mathbb{Z}^{m_1 \times n}, X_2 \in \mathbb{Z}^{m_2 \times n}$, and $U \in \mathbb{Z}^{m \times m}$ with $m = m_1 + m_2$, such that:

- the distribution of (X_1, X_2) is within statistical distance $2^{-\Omega(n)}$ of the distribution $D_{\mathbb{Z},\sigma_1}^{m_1 \times n} \times (D_{\mathbb{Z}^{m_2},\sigma_2,\vec{\delta}_1} \times \cdots \times D_{\mathbb{Z}^{m_2},\sigma_2,\vec{\delta}_n})$, where $\vec{\delta}_i$ denotes the ith canonical unit vector in $\mathbb{Z}^{m_2}$ whose ith coordinate is 1 and whose remaining coordinates are 0.
- we have $|\det U| = 1$ and $U \cdot X = (I_n \| 0)$ with $X = (X_1 \| X_2)$,
- every row of U has norm $\leq O(\sqrt{nm_1}\sigma_2)$ with probability $\geq 1 - 2^{-\Omega(n)}$.

The second statement implies that the last $m - n$ rows of U form a basis of the random lattice $\ker(X)$.

Proof: We first sample X_1 from $D_{\mathbb{Z},\sigma_1}^{m_1 \times n}$ using the GPV algorithm. We run m_2 times the algorithm from Lemma C.3.2, on the input $n, m_1, \sigma_1, \sigma_2, X_1$ and $\vec{c}$ running through the columns of $C = [I_n | 0_{n \times (m_2 - n)}]$. This gives $X_2 \in \mathbb{Z}^{m_2 \times n}$ and $R \in \mathbb{Z}^{m_1 \times m_2}$ such that $X_2^t = [I_n | \mathbf{0}_{n \times (m_2 - n)}] + X_1^t \cdot R$. One can then see that $U \cdot X = [I_n \| \mathbf{0}]$, where

$$U = \left[\begin{array}{c|c} \vec{0} & I_{m_2} \\ \hline I_{m_1} & -(X_1|\vec{0}) \end{array}\right] \cdot \left[\begin{array}{c|c} I_{m_1} & \vec{0} \\ \hline -R^t & I_{m_2} \end{array}\right] = \left[\begin{array}{c|c} -R^t & I_{m_2} \\ \hline I_{m_1} + (X_1|\vec{0})R^t & -(X_1|\vec{0}) \end{array}\right], X = \left[\begin{array}{c} X_1 \\ \hline X_2 \end{array}\right].$$

The result then follows from Gaussian tail bounds (to bound the norms of the rows of X_1) and elementary computations. ∎

Our gadget matrix G is U^{-t}. In the following corollary, we summarize the properties we will use.

Corollary C.3.4 Let $n, m_1, m_2, m, \sigma_1, \sigma_2$ be as in Theorem C.3.3. There exists a ppt algorithm that given n, m_1, m_2 (in unary), and σ_1, σ_2 as inputs, returns $G \in \mathbb{Z}^{m \times m}$ such that:

- the top $n \times m$ submatrix of G is within statistical distance $2^{-\Omega(n)}$ of $D_{\mathbb{Z},\sigma_1}^{n \times m_1} \times (D_{\mathbb{Z}^{m_2},\sigma_2,\vec{\delta}_1} \times \cdots \times D_{\mathbb{Z}^{m_2},\sigma_2,\vec{\delta}_n})^t$,
- we have $|\det G| = 1$ and $\|G^{-1}\| \leq O(\sqrt{nm_2}\sigma_2)$, with probability $1 - 2^{-\Omega(n)}$.

C.3.2 Hardness of k-LWE

The following result shows that this LWE variant, with S a specific diagonal matrix, is no easier than LWE.

Theorem C.3.5 There exists $c > 0$ such that the following holds for $k = n/(c \log n)$. Let m, q, σ, σ' be such that $\sigma \geq \Omega(n)$, $\sigma' \geq \Omega(n^3\sigma^2/\log n)$, $q \geq \Omega(\sigma'\sqrt{\log m})$ is prime, and $m \geq \Omega(n \log q)$ (e.g., $\sigma = \Theta(n)$, $\sigma' = \Theta(n^5/\log n)$, $q = \Theta(n^5)$ and $m = \Theta(n \log n)$). Then there exists a probabilistic polynomial-time reduction from $\mathrm{LWE}_{m+1,\alpha}$ in dimension n to (k, S)-$\mathrm{LWE}_{m+2n,\alpha'}$ in dimension $4n$, with $\alpha' = \Omega(mn^{3/2}\sigma\sigma'\alpha)$ and $S = \left[\begin{array}{c|c} \sigma \cdot I_{m+n} & 0 \\ \hline 0 & \sigma' \cdot I_n \end{array}\right]$. More concretely, using a (k, S)-$\mathrm{LWE}_{m+2n,\alpha'}$ algorithm with run-time T and advantage ε, the reduction gives an $\mathrm{LWE}_{m+1,\alpha}$ algorithm with run-time $T' = O(\mathcal{P}oly(m) \cdot (\varepsilon - 2^{-\Omega(n/\log n)})^{-2} \cdot (T + \mathcal{P}oly(m))$ and advantage $\varepsilon' = \Omega((\varepsilon - 2^{-\Omega(n/\log n)})^3) - O(2^{-n})$.

The reduction takes an LWE instance and extends it to a related k-LWE instance for which the additional hint vectors $(\vec{x}_i)_{i\leq k}$ are known. The major difficulty in this extension is to restrain the noise increase, as a function of k.

The existing approach for this reduction (that we improve below) is the technique used in the SIS to k-SIS reduction from [BF11]. In the latter approach, the hint vectors are chosen independently from a small discrete Gaussian distribution, and then the LWE matrix A is extended to a larger matrix A' under the constraint that the hint vectors are in the q-ary lattice $\Lambda^{\perp}(A') = \{\vec{b} : \vec{b}^t A' = \vec{0} \bmod q\}$. Unfortunately, with this approach, the transformation from an LWE sample with respect to A, to a k-LWE sample with respect to A', involves a multiplication by the cofactor matrix $\det(G)\cdot G^{-1}$ over $\mathbb{Z}$ of a $k\times k$ full-rank submatrix G of the hint vectors matrix. Although the entries of G are small, the entries of its cofactor matrix are almost as large as $\det G$, which is exponential in k. This leads to an "exponential noise blowup," restraining the applicability range to $k \leq \widetilde{O}(1)$ if one wants to rely on the hardness of LWE with noise rate $1/\alpha \leq \mathcal{P}oly(n)$ (otherwise, LWE is not exponentially hard to solve). To restrain the noise increase for large k, we use the gadget of Corollary C.3.4. Ignoring several technicalities, the core idea underlying our reduction is that the latter gadget allows us to sample a small matrix $\overline{X}_2$ with $\overline{X}_2^{-1}$ also small, which we can then use to transform the given LWE matrix $A^+ = (\vec{u}^t \| A) \in \mathbb{Z}_q^{(m+1)\times n}$ into a taller k-LWE matrix $A'^+ = T\cdot A^+$, using a transformation matrix T of the form

$$T = \left[\begin{array}{c} I_{m+1} \\ \hline -\overline{X}_2^{-1} X_1 \end{array}\right],$$

for some small independently sampled matrix $X_1 = [\vec{1}|\overline{X}_1]$. We can accordingly transform the given LWE sample vector $\vec{b} = A^+\vec{s} + \vec{e}$ for matrix A^+ into an LWE sample $\vec{b}' = T\vec{b} = A'^+\vec{s} + T\vec{e}$ for matrix A'^+ by multiplying the given sample by T. Since $[X_1|\overline{X}_2]\cdot T = 0$, it follows that $[X_1|X_2]\cdot A'^+ = 0$, so we can use k small rows of $[X_1|\overline{X}_2]$ as the k-LWE hints $\vec{x}_i^+$ for the new matrix A'^+, while, at same time, the smallness of T keeps the transformed noise $\vec{e}' = T\vec{e}$ small.

Proof: For a technical reason related to the non-zero centers $\vec{\delta}_i$ in the distribution of the hint vectors produced by our gadget from Corollary C.3.4, we decompose our reduction from $\mathrm{LWE}_{m+1,\alpha}$ to (k,S)-LWE into two subreductions. The first subreduction (outlined above) reduces $\mathrm{LWE}_{m+1,\alpha}$ in dimension n to (k,S,C)-$\mathrm{LWE}_{m+2n,\alpha'}$ in dimension $4n$, where the ith row of C is the unit vector $\vec{c}_i = (0^{m+n}|\vec{\delta}_i) \in \mathbb{R}^{m+2n}$ for $i = 1,\ldots,k$. The second subreduction reduces (k,S,C)-$\mathrm{LWE}_{m+2n,\alpha'}$ in dimension $4n$ to (k,S)-$\mathrm{LWE}_{m+2n,\alpha'}$ in dimension $4n$. We first describe and analyze the first subreduction, and then explain the second subreduction.

Description of the first subreduction. Let $(A^+,\vec{b})$ with $A^+ = (\vec{u}^t\|A)$ denote the given $\mathrm{LWE}_{\alpha,m+1}$ input instance, where $A^+ \hookleftarrow U(\mathbb{Z}_q^{(m+1)\times n})$, and $\vec{b} \in \mathbb{T}^{m+1}$ comes from either the "LWE distribution" $\frac{1}{q}U\left(\mathrm{Im}(A^+)\right) + \nu_\alpha^{m+1}$ or the "Uniform distribution" $\frac{1}{q}U\left(\mathbb{Z}_q^{m+1}\right) + \nu_\alpha^{m+1}$. The reduction maps $(A^+,\vec{b})$ to $(A',\vec{u}',X,\vec{b}')$ with $A' \in \mathbb{Z}_q^{(m+2n)\times 4n}$ and $\vec{u}' \in \mathbb{Z}_q^{4n}$ independent and uniform, $X \in \mathbb{Z}^{k\times(m+2n)}$ with its ith row $\vec{x}_i$ independently sampled from $D_{\Lambda^{\perp}_{-\vec{u}'}(A'),S}$ for $i \leq k$, and $\vec{b}' \in \mathbb{T}^{m+1+2n}$ coming from either the "k-LWE distribution" $\frac{1}{q}U\left(\mathrm{Im}(A'^+)\right) + \nu_\alpha^{m+1+2n}$ if $\vec{b}$ is from the "LWE distribution," or the "k-Uniform distribution" $\frac{1}{q}U\left(\mathrm{Span}_{i\leq k}(\vec{x}_i^+)^{\perp}\right)$ if $\vec{b}$ is from the "Uniform distribution." Here $A'^+ = (\vec{u}'^t\|A')$, and $\vec{x}_i^+$ denotes the vector $(1\|\vec{x}_i)$ for $i \leq k$. The reduction is as follows.

1. Sample gadget $\overline{X}_2 \in \mathbb{Z}^{2n\times 2n}$ using Corollary C.3.4 (with parameters n, m_1, m_2, σ_1 and σ_2 respectively set to k, n, n, σ and σ'), and sample $\overline{X}_1 \hookleftarrow D_{\mathbb{Z},\sigma}^{2n\times m}$. Define $T = \left[\begin{array}{c} I_{m+1} \\ \hline -\overline{X}_2^{-1}\cdot(\mathbf{1}|\overline{X}_1) \end{array}\right] \in$

$\mathbb{Z}^{(m+1+2n)\times(m+1)}$, where $\vec{1}$ is the all-1 vector. Let $X \in \mathbb{Z}^{k\times(m+2n)}$ denote the matrix made of the top k rows of $(\overline{X}_1|\overline{X}_2)$.

2. Sample $C^+ \in \mathbb{Z}_q^{(m+1+2n)\times 3n}$ with independent columns uniform orthogonally to $\mathrm{Im}((\mathbf{1}|X))$ modulo q. Let $\vec{u}_C^t \in \mathbb{Z}_q^{3n}$ be the top row of C^+, and $C \in \mathbb{Z}_q^{(m+2n)\times 3n}$ denote its remaining $m+2n$ rows.

3. Compute $\Sigma = \alpha' \cdot I_{m+1+2n} - T \cdot T^t$ and $\sqrt{\Sigma}$ such that $\sqrt{\Sigma}\cdot\sqrt{\Sigma}^t = \Sigma$; if Σ is not positive definite, abort.

4. Compute $A'^+ = (T \cdot A^+|C^+)$ and $\vec{b}' = T\vec{b} + \frac{1}{q}C^+ \cdot \vec{s}' + \sqrt{\Sigma}\vec{e}'$, with $\vec{s}' \hookleftarrow U(\mathbb{Z}_q^{3n})$ and $\vec{e}' \hookleftarrow \nu_1^{m+1+2n}$. Let $(\vec{u}')^t = (\vec{u}\|\vec{u}_C)^t \in \mathbb{Z}_q^{4n}$ be the top row of A'^+.

5. Return $(A', \vec{u}', X, \vec{b}')$.

Step 1 aims at building a transformation matrix T that sends A^+ to the left n columns of A'^+. Two properties are required from this transformation. First, it must be a linear map with small coefficients, so that when we map the LWE right hand side to the k-LWE right hand side, the noise component does not blow up. Second, it must contain some vectors $(1\|\vec{x}_i)$ in its (left) kernel, with $\vec{x}_i$ normally distributed. These vectors are to be used as k-LWE hints. For this, we use the gadget of the previous subsection. This ensures that the $\vec{x}_i$'s are (almost) distributed as independent Gaussian samples from $D_{\mathbb{Z}^n,\sigma} \times D_{\mathbb{Z}^n,\sigma'}$, and that the matrix T is integral with small coefficients. We define $B \in \mathbb{Z}_q^{2n\times n}$ by $[A^+\|B] = TA^+$, so that we have:

$$\left[\mathbf{1}|\overline{X}_1|\overline{X}_2\right] \cdot \left[\frac{A^+}{B}\right] = \left[\mathbf{1}|\overline{X}_1|\overline{X}_2\right] \cdot \begin{bmatrix} I_{m+1} \\ -\overline{X}_2^{-1}\cdot(\mathbf{1}|\overline{X}_1) \end{bmatrix} \cdot A^+ = \mathbf{0} \bmod q.$$

This means each row of $\left(\overline{X}_1|\overline{X}_2\right)$ belongs to $\Lambda^\perp_{-\vec{u}}(A'')$, where $A'' = [A^t|B^t]^t$.

At this stage, it is tempting to define the k-LWE matrix as A'' and give away the k-LWE hint vectors $\vec{x}_i \in \Lambda^\perp_{-\vec{u}}(A'')$ making up the matrix X. However, this approach does not quite work: we have extended A by $2n$ rows, but we give only k hint vectors (we cannot output them all, as the bottom rows of $\overline{X}_2$ may not be normally distributed). This creates a difficulty for mapping "Uniform" to "k-Uniform" in the reduction. Step 2 circumvents the above difficulty by sampling extra column vectors $C^+ \in \mathbb{Z}_q^{(m+1+2n)\times 3n}$ that are uniform in the subspace orthogonal to the hint vectors $\vec{x}_i^+$ modulo q. When the parameters are properly set, the columns of $[T|C^+]$ span the full subspace orthogonal to the $\vec{x}_i$'s mod q, with overwhelming probability. We finally set $A'^+ = \left[\frac{A^+}{B}\middle|C^+\right]$.

It remains to see how to map "LWE" to "k-LWE." The main problem, when multiplying $\vec{b}$ by T, is that the LWE noise gets skewed. If its covariance matrix was of the form $\alpha^2 \cdot I_{m+1}$, then it becomes $\alpha^2 T \cdot T^t$. To compensate for that, in Step 3, we add to $T \cdot \vec{b}$ an independent Gaussian noise with well-chosen covariance $\Sigma = \alpha'^2 \cdot I_{m+1+2n} - \alpha^2 T \cdot T^t$. We set α' large enough to ensure that this symmetric matrix is positive definite. This noise unskewing technique was adapted to discrete Gaussians and used in cryptography in [Pei10].

Analysis of the first subreduction. All steps of the reduction can be implemented in polynomial time. Its correctness follows from the following three lemmas. The proofs can be found in the appendix.

Lemma C.3.6 The tuple $(A', \vec{u}', X)$ is within statistical distance $2^{-\Omega(n/\log n)}$ of the distribution in which $A' \in \mathbb{Z}_q^{(m+2n)\times 4n}$ and $\vec{u}' \in \mathbb{Z}_q^{4n}$ are independent and uniform, and the rows of $X \in \mathbb{Z}^{k\times(m+2n)}$ are from $D_{\Lambda^\perp_{-\vec{u}'}(A'),S,\vec{c}_i}$, where $\vec{c}_i = (0^{m+n}|\vec{\delta}_i) \in \mathbb{R}^{m+2n}$ and $\vec{\delta}_i$ denotes the ith canonical unit vector in $\mathbb{Z}^n$ for $i = 1, \ldots, k$.

Next, we assume that (A'^+, X) is fixed and consider the distribution of $\vec{b}'$ in the two cases of the distribution of $\vec{b}$. First we consider the "LWE" to "k-LWE" distribution mapping.

Lemma C.3.7 The following holds with probability $1 - 2^{-\Omega(n/\log n)}$ over the choice of $\overline{X}_1$ and $\overline{X}_2$. If $\vec{b} \in \mathbb{T}^{m+1}$ is sampled from $\frac{1}{q}U(\mathrm{Im}A) + \nu_\alpha^{m+1}$, then $\vec{b}' \in \mathbb{T}^{m+1+2n}$ is within statistical distance $2^{-\Omega(n)}$ of $\frac{1}{q}U\left(\mathrm{Im}A'^+\right) + \nu_{\alpha'}^{m+1+2n}$.

Finally, we consider the "Uniform" to "k-Uniform" distribution mapping.

Lemma C.3.8 The following holds with probability $1 - 2^{-\Omega(n/\log n)}$ over the choice of $\overline{X}_1$ and $\overline{X}_2$. If $\vec{b}$ is sampled from $\frac{1}{q}U\left(\mathbb{Z}_q^{m+1}\right) + \nu_\alpha^{m+1}$, then $\vec{b}'$ is within statistical distance $2^{-\Omega(n)}$ of $\frac{1}{q}U\left(\mathrm{Span}_{i\le k}(\vec{x}_i^+)^\perp\right) + \nu_{\alpha'}^{m+1+2n}$.

Overall, we have described a reduction that maps the "LWE distribution" to the "k-LWE distribution," and the "Uniform distribution" to the "k-Uniform distribution," up to statistical distance $2^{-\Omega(n/\log n)}$.

Second subreduction. It remains to reduce the (k, S, C)-LWE with non-zero centers for the hint distribution, to (k, S)-LWE with zero-centered hints. For this, we use Lemma C.2.6 to obtain the following.

Lemma C.3.9 Let $m' = m + 2n$, $n' = 4n$, and assume that $\sigma_{m'}(S) \ge \omega(\sqrt{n})$. If there exists a distinguisher against (k, S)-LWE$_{m',\alpha'}$ in dimension n' with run-time T and advantage ε, then there exists a distinguisher against (k, S, C)-LWE$_{m',\alpha'}$ with run-time $T' = O(\mathcal{P}oly(m') \cdot (\varepsilon - 2^{-\Omega(n)})^{-2} \cdot T)$ and advantage $\varepsilon' = \Omega((\varepsilon - O(2^{-n}))^3/R - O(2^{-n}))$, where $R = \exp(O(k \cdot (2^{-n} + \|C\|^2/\sigma_{m'}(S)^2)))$.

The main idea of the proof of Lemma C.3.9, given in the appendix, is to apply Lemma C.2.6 with P, P' being the (k, S)-LWE and (k, S, C)-LWE problems respectively, which have instances of the form $x = (r, \vec{y})$, where $r = (A, \vec{u}, \{\vec{x}_i\}_{i\le k})$ and the hints $\vec{x}_i$ for $i \le k$ sampled from either the zero-centered distribution $\hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{0}}$ (distribution Φ of r, in (k, S)-LWE) or the non-zero center distribution $\hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{c}_i}$ (distribution Φ' of r, in (k, S, C)-LWE), and $\vec{y} \in \mathbb{T}^{m+1}$ is a sample from either the distribution

$$D_0(r) = \frac{1}{q} \cdot U\Big(\mathrm{Im}\Big(\frac{\vec{u}^t}{A}\Big)\Big) + \nu_\alpha^{m+1}$$

or the distribution

$$D_1(r) = \frac{1}{q} \cdot U\Big(\mathrm{Span}_{i\le k}\Big(\frac{1}{\vec{x}_i}\Big)^\perp\Big) + \nu_\alpha^{m+1}.$$

Given $x = (r, \vec{y})$, is possible to efficiently sample $\vec{y}'$ from either $D_0(r)$ or $D_1(r)$, so the public-samplability property assumed by Lemma C.2.6 is satisfied. This Lemma gives the desired

reduction between (k, S)-LWE and (k, S, C)-LWE, as long as the RD $R(\Phi\|\Phi')$ between the distribution of r in the two problems is polynomially bounded. The latter reduces to obtaining a bound on the RD between a Gaussian distribution and a small offset thereof, which is given by Lemma C.2.5.

In our application of Lemma C.3.9, the (k, S, C)-LWE problem resulting from the first subreduction has $\|C\| = 1$, and $\sigma_{m'}(S) = \sigma$, so that $R = \exp(O(k \cdot (2^{-n} + 1/\sigma^2))) = O(1)$ using $\sigma = \Omega(n)$ and $k \leq n$. This shows that the second subreduction is probabilistic polynomial time. ∎

Our technique can be applied to improve the Boneh-Freeman reduction from SIS to k-SIS, from an exponential loss in k to a polynomial loss in k. In fact, we map A to A'' in the same way (except that we do not use and add $\vec{u}$ on top of the matrix A) and then also use the top k rows of $(\overline{X}_1|\overline{X}_2)$ as the k-SIS hints for the new matrix A''. Then, whenever the adversary can output a short vector $\vec{x_1}\|\vec{x_2}$ that is orthogonal to A'', we can also output a short vector $(\vec{x_1} - \vec{x_2} \cdot \overline{X}_2^{-1}\overline{X}_1)$ which is orthogonal to A. As the rows of $\overline{X}_1$ are distributed as independent Gaussian samples and the adversary is only given its first k rows, it can be shown that, if $\vec{x_1}\|\vec{x_2}$ is linearly independent from the k-SIS hints, then the vector $(\vec{x_1} - \vec{x_2} \cdot \overline{X}_2^{-1}\overline{X}_1)$ is null with a negligible probability. RD may also be used to reduce k-SIS with non-zero-centered hints (with small centers) to k-SIS with zero-centered hints.

C.4 A lattice-based public-key traitor tracing scheme

In this section, we describe and analyze our basic traitor tracing scheme. First, we give the underlying multi-user public-key encryption scheme. We then explain how to implement black-box confirmation tracing.

C.4.1 A multi-user encryption scheme

The scheme is designed for a given security parameter n, a number of users N and a maximum malicious coalition size t. It then involves several parameters q, m, α, S. These are set so that the scheme is correct (decryption works properly on honestly generated ciphertexts) and secure (semantically secure encryption and possibility to trace members of malicious coalitions). In particular, we set $S = \mathrm{Diag}(\sigma, \ldots, \sigma, \sigma', \ldots, \sigma') \in \mathbb{R}^{m\times m}$ where $\sigma' > \sigma$ and their respective numbers of iterations are set so that (t, S)-$\mathrm{LWE}_{m+1,\alpha}$ is hard to solve.

`Setup`. The trusted authority generates a master key pair using the algorithm from Lemma C.2.2. Let $(A, T) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}^{m\times m}$ be the output. We additionally sample $\vec{u}$ uniformly in $\mathbb{Z}_q^n$. Matrix T will be part of the tracing key tk, whereas the public key is $pk = A^+$, with $A^+ = (\vec{u}^t\|A)$.

Each user $\mathcal{U}_i$ for $i \leq N$ obtains a secret key sk_i from the trusted authority, as follows. The authority executes the GPV algorithm using the basis of $\Lambda^\perp(A)$ consisting of the rows of T, and the standard deviation matrix S. The authority obtains a sample $\vec{x}_i$ from $D_{\Lambda^\perp_{-\vec{u}}(A),S}$. The standard deviations $\sigma' > \sigma$ may be chosen as small as $3mq^{n/m}\sqrt{(2m+4)/\pi}$. The user secret key is $\vec{x}_i^+ = (1\|\vec{x}_i) \in \mathbb{Z}^{m+1}$. Using the Gaussian tail bound and the union bound, we have $\|\vec{x}_i\| \leq \sqrt{m}\sigma'$ for all $i \leq N$, with probability $\geq 1 - N \cdot 2^{-\Omega(m)}$.

The tracing key tk consists of the matrix T and all pairs $(\mathcal{U}_i, sk_i)$.

`Encrypt`. The encryption algorithm is exactly the 1-bit encryption scheme from [GPV08, Se. 7.1], which we recall, for readability.[1] The plaintext and ciphertext domains are $\mathcal{P} = \{0, 1\}$ and $\mathcal{C} =$

[1] As usual, the encryption algorithm may be used to encapsulate session keys which are then fed into an efficient data encapsulation mechanism to encrypt the data.

$\mathbb{Z}_q^{m+1}$ respectively, and:

$$\texttt{Enc} : M \mapsto \left[\frac{\vec{u}^t}{A}\right] \cdot \vec{s} + \vec{e} + \left[\frac{M \cdot \lfloor q/2 \rfloor}{\vec{0}}\right], \quad \text{where } \vec{s} \hookleftarrow U(\mathbb{Z}_q^n) \text{ and } \vec{e} \hookleftarrow \lfloor \nu_{\alpha q} \rceil^{m+1}.$$

As explained in [GPV08], this scheme is semantically secure under chosen plaintext attacks (IND-CPA), under the assumption that $\mathrm{LWE}_{m+1,\alpha}$ is hard to solve.

`Decrypt`. To decrypt a ciphertext $\vec{c} \in \mathbb{Z}_q^{m+1}$, user $\mathcal{U}_i$ uses its secret key $\vec{x}_i^+$ and evaluates the following function `Dec` from $\mathbb{Z}_q^{m+1}$ to $\{0,1\}$: Map $\vec{c}$ to 0 if $\langle \vec{x}_i^+, \vec{c} \rangle \bmod q$ is closer to 0 than $\pm\lfloor q/2 \rfloor$.

If $\vec{c}$ is an honestly generated ciphertext of a plaintext $M \in \{0,1\}$, we have $\langle \vec{x}_i^+, \vec{c} \rangle = \langle \vec{x}_i^+, \vec{e} \rangle + M \cdot \lfloor q/2 \rfloor \bmod q$, where $\vec{e} \hookleftarrow \lfloor \nu_{\alpha q} \rceil^{m+1}$. It can be shown that the latter has magnitude $\leq 2\sqrt{m}\alpha q \|\vec{x}_i^+\|$ with probability $1 - 2^{-\Omega(n)}$ over the randomness of $\vec{e}$. This is $\leq 3m\alpha q\sigma'$ for all i, with probability $\geq 1 - N \cdot 2^{-\Omega(n)}$. To ensure the correctness of the scheme, it suffices to set $q \geq 4m\alpha q\sigma'$. Note that other constraints will be added to enable tracing.

Theorem C.4.1 Let m, n, q and N be integers such that q is prime and $N \leq 2^{o(n)}$. Let $\alpha, \sigma, \sigma' > 0$ such that $\sigma' \geq \sigma \geq \Omega(mq^{n/m}\sqrt{\log m})$ and $\alpha \leq 1/(4m\sigma')$. Then the scheme described above is IND-CPA under the assumption that $\mathrm{LWE}_{m+1,\alpha}$ is hard. Further, the decryption algorithm is correct:

$$\forall M \in \{0,1\}, \forall i \leq N : \ \texttt{Dec}\left(\texttt{Enc}(M, pk), sk_i\right) = M$$

holds with probability $\geq 1 - 2^{-\Omega(n)}$ over the randomness used in `Setup` and `Enc`.

C.4.2 Tracing traitors

We now present a black-box confirmation algorithm `Trace`.[2] It is given access to an oracle $\mathcal{O}^{\mathcal{D}}$ that provides black-box access to a decryption device $\mathcal{D}$. It takes as inputs the tracing key $tk = (T, (\mathcal{U}_i, \vec{x}_i^+)_{i \leq N})$ and a set of suspect users $\{\mathcal{U}_{i_1}, \ldots, \mathcal{U}_{i_k}\}$ of cardinality $k \leq t$, where t is the a priori bound on any coalition size. Wlog, we may consider that $k = t$ and $i_j = j$ for all $j \leq k$.

Algorithm `Trace` gathers information about which keys have been used to build decoder $\mathcal{D}$, by feeding different carefully designed distributions to oracle $\mathcal{O}^{\mathcal{D}}$. We consider the following $t+1$ distributions $Tr_0, \ldots, Tr_t$ over $\mathcal{C} = \mathbb{Z}_q^{m+1}$:

$$Tr_i = U\left(\mathrm{Span}(\vec{x}_1^+, \ldots, \vec{x}_i^+)^\perp\right) + \lfloor \nu_{\alpha q} \rceil^{m+1}.$$

The first distribution Tr_0 is the uniform distribution, whereas the last distribution Tr_t is meant to be computationally indistinguishable from $\texttt{Enc}(0)$. We define p_∞ as the probability $\Pr[\mathcal{O}^{\mathcal{D}}(\vec{c}, M) = 1]$ that the decoder can decrypt the ciphertexts, over the randomness of $M \hookleftarrow U(\{0,1\})$ and $\vec{c} \hookleftarrow \texttt{Enc}(M)$. We define p_i as the probability the decoder decrypts the signals in Tr_i, for $i \in [0, t]$:

$$p_i \ = \ \Pr_{\substack{\vec{c} \hookleftarrow Tr_i \\ M \hookleftarrow U(\{0,1\})}} \left[\mathcal{O}^{\mathcal{D}}\left(\vec{c} + \left[\frac{M \cdot \lfloor q/2 \rfloor}{\vec{0}}\right], M\right) = 1\right].$$

A gap between p_{i-1} and p_i is meant to indicate that $\mathcal{U}_i$ is a traitor.

The confirmation and soundness properties are proved in the full version. We now concentrate on a new feature of our scheme: public traceability.

[2] Note that in our context, minimal access is equivalent to standard access: since the plaintext domain is small, plaintext messages can be tested exhaustively.

C.5 Projective sampling and public traceability

We now modify the scheme of the Section C.4 so that the tracing signals can be publicly sampled. For this purpose, we introduce the concept of projective sampling family.

C.5.1 Projective sampling

Inspired from the notion of projective hash family [CS02], we propose the notion of projective sampling family in which each sampling function is keyed and, with a projected key, one can simulate the sampling function in a computationally indistinguishable way. Let X be a finite non-empty set. Let $F = (\mathtt{F}_k)_{k\in K}$ be a collection of sampling functions indexed by K, so that $\mathtt{F}_k$ is a sampling function over X, for every $k \in K$. We call $\mathtt{Sam} = (F, K, X)$ a sampling family. We now introduce the concept of projective sampling.

Definition C.5.1 [Projective Sampling] Let $\mathtt{Sam} = (F, K, X)$ be a sampling family. Let J be a finite, non-empty set, and let $\pi : K \to J$ be a (probabilistic) function. Let also $\mathtt{P} = (\mathtt{P}_j)_{j\in J}$ be a collection of sampling functions over X, and D be a distribution over K. Then $\mathtt{PSam} = (F, K, X, \mathtt{P}, J, \pi, D)$ is called a projective sampling family if, with overwhelming probability over the choice of $k, k' \hookleftarrow D$, and given the secret key k and its projected key $\pi(k)$, 1) the distributions obtained using $\mathtt{F}_k$ and $\mathtt{P}_{\pi(k)}$ are computationally indistinguishable, and 2) the distributions obtained using $\mathtt{F}_k$ and $\mathtt{P}_{\pi(k')}$ can be efficiently distinguished.

The first condition means that for $k \hookleftarrow D$, the value $\pi(k)$ "encodes" the sampling distribution of $\mathtt{F}_k$, so that when $\pi(k)$ is made public, the sampled signal $\mathtt{F}_k$ can be publicly simulated by $\mathtt{P}_{\pi(k)}$. The security requirement is very strong because the adversary is not only given the projected key, as in projective hashing, but also the secret key k. We require that sampling signals from the secret key and from its projected key are indistinguishable for the insiders who know the secret key. This is relevant for traitor tracing, as the traitors are system insiders and they possess secret data. The second condition (that we actually do not directly use in our cryptographic application) allows to prevent the trivial solution consisting in setting $\mathtt{P}_{\pi(k)}$ as an efficient sampling function that is independent of k: the simulation signal $\mathtt{P}_{\pi(k)}$ must be specific to k.[3]

C.5.2 Projective sampling from k-LWE

We construct a set of projective sampling families $(\mathtt{PSam}_i)_{0\le i\le t}$. The parameters are almost identical to the parameters in the $\mathtt{Setup}$ of the multi-user scheme of Section C.4. A further difference, required for simulation purposes in the security proof, is that $\sigma' > \sigma$ must be set $\widetilde{\Omega}(\sqrt{mn} + \pi q)$.

We let $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$ and $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$ be public parameters. For each i, we define $K_i = (\mathbb{Z}_q^m)^i$ and D_i as the distribution on K_i that samples $k = (\vec{x}_j)_{j\le i}$ with $\vec{x}_j \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),\sigma}$ for all $j \le i$. The sampling function $\mathtt{F}_{i,k}$ is defined as $U(\mathrm{Span}_{j\le i}(\vec{x}_j^+)^\perp) + \lfloor \nu_{\alpha q} \rceil^{m+1}$. The projected key $\pi_i(k)$ is defined as follows:

- Sample $H \in \mathbb{Z}_q^{m\times(m-n)}$ uniformly, conditioned on $\mathrm{Im}(A) \subseteq \mathrm{Im}(H)$.
- For each $j \le i$, define $\vec{h}_j^t = -\vec{x}_j^t \cdot H$.
- Finally, set $J = \mathbb{Z}_q^{m\times(m-n)} \times (\mathbb{Z}_q^{m-n})^i$ and set $\pi_i(k) = (H, (\vec{h}_j)_{j\le i})$.

We now define the sampling $\mathtt{P}_{i,\pi_i(k)}$ with projected key $\pi_i(k) = (H, (\vec{h}_j)_{j\le i})$, as follows:

- Set $H_j = (\vec{h}_j^t \| H) \in \mathbb{Z}_q^{(m+1)\times(m-n)}$. We have $\vec{x}_j^{+t} \cdot H_j = \vec{0}$ and $\mathrm{Im}(A^+) \subseteq \mathrm{Im}(H_j)$.

[3] Another trivial situation occurs when $\pi(k) = k$: the projected key leaks the full information about the original key and one cannot safely publish the projected key.

• Set $\mathtt{P}_{i,\pi_i(k)} = U\left(\cap_{j\le i}\mathrm{Im}(H_j)\right) + \lfloor\nu_{\alpha q}\rceil^{m+1}$, with $\cap_{j\le 0}\mathrm{Im}(H_j) = \mathbb{Z}_q^{m+1}$ by convention. Note that $\cap_{j\le i}\mathrm{Im}(H_j) \subseteq \mathrm{Span}_{j\le i}(\vec{x}_j^+)^\perp$.

Theorem C.5.2 For each $i = 0, \ldots, t$, $\mathtt{PSam}_i$ is a projective sampling family. Concretely, under the (i,S)-LWE$_{\alpha,m}$ hardness assumptions, given the uniformly sampled public parameters $(A, \vec{u})$, the secret key $k = (\vec{x}_j)_{j\le i} \hookleftarrow D_i$ and its projected key $\pi_i(k) = (H, (\vec{h}_j)_{j\le i})$, the distributions $\mathtt{F}_{i,k}$ and $\mathtt{P}_{i,\pi_i(k)}$ are indistinguishable. Moreover, they are both indistinguishable from $U(\mathrm{Im}(A^+)) + \lfloor\nu_{\alpha q}\rceil^{m+1}$. Finally, with overwhelming probability, the distributions $\mathtt{F}_{i,k}$ and $\mathtt{P}_{i,\pi_i(k')}$ can be efficiently distinguished, when k' is independently sampled from D_i.

Proof: For the last statement, observe that with overwhelming probability, the secret key k' contains an $\vec{x}'_j \in \mathbb{Z}_q^m$ that does not belong to $\mathrm{Span}_{j\le i}(\vec{x}_j)$ (by Lemma C.2.4). In that case, taking the inner product of all $\vec{x}'_j$'s of k' with a sample from $\mathtt{P}_{i,\pi_i(k')}$ gives small residues modulo q, whereas one of the inner products of the $\vec{x}'_j$'s with a sample from with a sample from $\mathtt{F}_{i,k}$ will be uniform modulo q.

We now consider the first statement. From the hardness of (i,S)-LWE$_{m,\alpha}$, given k, the distributions

$$\mathtt{F}_{i,k} = U(\mathrm{Span}_{j\le i}(\vec{x}_j^+)^\perp) + \lfloor\nu_{\alpha q}\rceil^{m+1} \quad \text{and} \quad U(\mathrm{Im}(A^+)) + \lfloor\nu_{\alpha q}\rceil^{m+1}$$

are indistinguishable. Further, given $k = (\vec{x}_j)_{j\le i}$, the projected key $\pi_i(k) = (H, (\vec{h}_j)_{j\le i})$ can be sampled from D_i. Therefore, given both k and $\pi_i(k)$, the distributions $\mathtt{F}_{i,k}$ and $U(\mathrm{Im}(A^+)) + \lfloor\nu_{\alpha q}\rceil^{m+1}$ remain indistinguishable.

Now, we have $\mathrm{Im}(A^+) \subseteq \cap_{j\le i}\mathrm{Im}(H_j) \subseteq (\mathrm{Span}_{j\le i}(\vec{x}_j^+))^\perp$. Hence:

$$\begin{aligned} U(\mathrm{Im}(A^+)) + U(\cap_{j\le i}\mathrm{Im}(H_j)) &= U(\cap_{j\le i}\mathrm{Im}(H_j)), \\ U(\mathrm{Span}_{j\le i}(\vec{x}_j^+)^\perp) + U(\cap_{j\le i}\mathrm{Im}(H_j)) &= U(\mathrm{Span}_{j\le i}(\vec{x}_j^+)^\perp). \end{aligned}$$

We note that given $\vec{h}_1, \ldots, \vec{h}_i$, one can efficiently sample from $U(\cap_{j\le i}\mathrm{Im}(H_j))$. Therefore, under the hardness of (i,S)-LWE$_{m,\alpha}$, this shows that $\mathtt{F}_{i,k}$, $\mathtt{P}_{i,\pi_i(k)}$ and $U(\mathrm{Im}(A^+)) + \lfloor\nu_{\alpha q}\rceil^{m+1}$ are indistinguishable. ▌

C.5.3 Public traceability from projective sampling

In the scheme of Section C.4, the tracing key $tk = (T, (\mathcal{U}_i, \vec{x}_i)_{i\le N})$ must be kept secret, as it would reveal the secret keys of the users. The tracing signals are samples from $U(\mathrm{Span}_{j\le i}(\vec{x}_j^+)^\perp) + \lfloor\nu_{\alpha q}\rceil^{m+1}$, which exactly matches $\mathtt{F}_{i,k}$. By publishing the projected key $\pi_i(k)$, anyone can use the projective sampling $\mathtt{P}_{i,\pi_i(k)}$: by Theorem C.5.2, given $(k, \pi_i(k))$, $\mathtt{F}_{i,k}$ and $\mathtt{P}_{i,\pi_i(k)}$ are indistinguishable and they are both indistinguishable from the original sampling $U(\mathrm{Im}(A^+)) + \lfloor\nu_{\alpha q}\rceil^{m+1}$. We are thus almost done with public traceability.

However, a subtle point is that we have to use all the projective samplings $(\mathtt{P}_{i,\pi_i(k)})$ for transforming the secret tracing to the public tracing: all the projected keys $(\vec{h}_j)_{j\le N}$ should be published. Because the keys k in $\mathtt{F}_{i,k}$ are not independent, it could occur that the adversary exploits a projected key $\pi_i(k)$ for distinguishing $\mathtt{P}_{i',\pi_{i'}(k')}$ from the original signals. To handle this, we prove that, given $(\vec{x}_j)_{j\le i}$ and all the keys $(\vec{h}_j)_{j\le N}$, the adversary cannot distinguish $\mathtt{P}_{i,\pi_i(k)}$ from the original signals. For this purpose, we exploit a technique from [GKV10] to simulate $(\vec{h}_j)_{i<j\le N}$ from the public information.

Theorem C.5.3 Set $i \leq t$. Under the (i,S)-LWE$_{\alpha,m}$ and the LWE$'_{\alpha,m}$ hardness assumptions, given the secret key $k = (\vec{x}_j)_{j\leq i}$ and the projected keys $(H, (\vec{h}_j)_{j\leq N})$, the following two distributions are indistinguishable

$$\mathtt{P}_{i,\alpha(k)} = U(\cap_{j\leq i}\mathrm{Im}(H_j)) + \lfloor \nu_{\alpha q} \rceil^{m+1} \quad \text{and} \quad U(\mathrm{Im}(A^+)) + \lfloor \nu_{\alpha q} \rceil^{m+1}.$$

Proof:

Assume a ppt attacker is given $(\vec{x}_j)_{j\leq i}$ (with the $\vec{x}_j$'s independently sampled from $D_{\Lambda^\perp_{-\vec{u}}(A),\sigma}$) and all the projected keys $(\vec{h}_j)_{j\leq N})$). We are to prove that, under the (i,S)-LWE$_{\alpha,m}$ and LWE$'_{\alpha,m}$ hardness assumptions, it cannot distinguish between the distributions (over $\mathbb{Z}_q^{m+1}$)

$$U(\mathrm{Im}(A^+)) + \lfloor \nu_{\alpha q} \rceil^{m+1} \quad \text{and} \quad \mathtt{P}_{i,\pi_i(k)} = U(\cap_{j\leq i}\mathrm{Im}(H_j)) + \lfloor \nu_{\alpha q} \rceil^{m+1}.$$

We proceed by a sequence of games.

Game $\mathbf{G}_0$: This is the above distinguishing game. We let ε_0 denote the adversary's distinguishing advantage. The goal is to show that ε_0 is negligible.

Game $\mathbf{G}_1$: In this second game, we sample $\vec{x}_1, \ldots, \vec{x}_i$ from $D_{\Lambda^\perp_{-\vec{u}}(A),\sigma}$ as in **Game**$_0$, but the $\vec{x}_j$'s for $j > i$ are sampled uniformly in $\mathbb{Z}_q^n$, conditioned on $\vec{x}_j^t \cdot A = -\vec{u}^t$. The $\vec{h}_j$'s for $j > i$ are modified accordingly, but the rest is as in **Game**$_0$. We let ε_1 denote the adversary's distinguishing advantage.

The main point is that in **Game**$_1$, no secret information is required for sampling the projected keys $\vec{h}_j$'s for $j > i$. The proof of the following lemma may be found in the full version.

Lemma C.5.4 Under the LWE$'_{\alpha,m}$ hardness assumption, the quantity $|\varepsilon_1 - \varepsilon_0|$ is negligible.

We note that, in **Game**$_1$, the $\vec{h}_j$'s can be sampled publicly from the available data. Therefore, from Theorem C.5.2, under the (i,S)-LWE$_{\alpha,m}$ hardness assumptions, the advantage ε_1 is negligible.∎

Semantic security of the updated scheme. We modify the public information of the scheme of Section C.4, so that we can use the set of projective sampling families described above. For this aim, we simply add the projected key $(H, (\vec{h}_i)_{i\leq N})$ to the public key. The scheme becomes publicly traceable because the tracing signals can be sampled from the projected keys, as explained above. Finally, as the public key has been modified, we should prove that the knowledge of these projected keys provides no significant advantage for an adversary towards breaking the semantic security of the encryption scheme. Fortunately, the semantic security directly follows from Theorem C.5.3, for the particular case of $i = 0$.

C.6 Appendix

C.7 Traitor Tracing

C.7.1 A short overview

Combinatorial schemes versus algebraic schemes. There are two main approaches for devising a traitor tracing encryption scheme. Many constructions are combinatorial in nature

(see [CFN94a, SW98b, CFNP00, SSW01a, PSNT06a, BP08, BN08a], among others): They typically combine an arbitrary encryption scheme with a collusion-resistant fingerprinting code. The most interesting property in combinatorial schemes is the capacity of dealing with black-box tracing. However, the efficiency of these traitor tracing schemes is curbed by the large parameters induced by even the best construction of such codes [Tar08a]: To resist coalitions of up to t malicious users among N users, the code length is $\ell = \Theta(t^2 \log N)$. Lower bounds with the same dependence with respect to t have been given in [PSS03, Tar08a], leaving little hope of significant improvements.

An alternative approach was initiated by Kurosawa and Desmedt in [KD98a] (whose construction was shown insecure in [SW98c]), and by Boneh and Franklin [BF99a]: The tracing functionality directly stems from the algebraic properties of the encryption scheme. As opposed to the combinatorial approach, this algebraic approach is not generic and requires designing ad hoc encryption schemes. We will concentrate on the algebraic approach in this paper. Prior to this work, all known algebraic traitor tracing schemes relied on variants of the Discrete Logarithm Problem: For instance, the earlier constructions (including [KD98a, BF99a, KY02c, KY02d]) rely on the assumed hardness of the Decision Diffie Hellman problem (DDH), whereas others (including [CPP05a, BSW06a, BW06a, ADML$^+$07a, FNP07a]) rely on variants of DDH on groups admitting pairings. The former provide strong security when instantiating with groups for which DDH is expected to be very hard (such as generic elliptic curves over prime fields), whereas the latter achieve improved functionalities while lowering the performance (as a function of the security level).

Public traceability. An important problem on traitor tracing is to handle the case where the tracer is not trusted. In this scenario, the tracing procedure must be run in a way that enables verification of the traitor implication, by a system outsider. The strongest notion for this is non-repudiation: the tracing procedure must produce an undeniable proof of the traitors implication. However, a necessary condition for achieving non-repudiation is that the setup involves some interactive protocol between the center and each user. Indeed, if the center generates all the parameters for the users, then any pirate decoder produced by a collusion of traitors can also be produced by the center and there is no way for the center to trustworthily prove the culpability of the traitors. All the existing schemes enjoying non-repudiation involve complex interactive proofs: a secure 2-party computation protocol in [Pfi96], a commitment protocol in [PW97], an oblivious polynomial evaluation in [WHI01, KWHI01, KY02a].

When considering the standard setting of non-interactive setup, we cannot get the full strength of non-repudiation, but we can still achieve a weaker but very useful property: public traceability. This notion allows anyone to perform the tracing from the public parameters only and hence the traitors implication can be publicly verified. Moreover, public traceability implies the capacity of delegating the tracing procedure: the tracer can run the tracing procedure in parallel on untrusted machines without leaking any secret information. This can prove crucial for the schemes with high tracing complexity. In fact, there are very few (non-interactive) schemes that achieve this property [PSNT06a, BW06a] (some schemes, such as [CPP05a, BP08, BN08a], partially achieve: some parts of the tracing procedure can be run publicly). The scheme [PSNT06a] is generic, based on IPP-codes, and is thus quite impractical. The Boneh-Waters scheme [BW06a] achieves resistance against unbounded coalitions, but has a large ciphertext size of $\Theta(\sqrt{N})$ group elements. All known efficient algebraic schemes are in the bounded collusion model and so far, none of them enjoys public traceability. In this paper, we achieve public traceability without downgrading the efficiency of the proposed sheme.

C.7.2 Public key traitor tracing encryption

A public-key traitor tracing scheme consists of four probabilistic algorithms `Setup`, `Enc`, `Dec` and `Trace`.

- Algorithm `Setup` is run by a trusted authority. It takes as inputs a security parameter λ, a list of users $(\mathcal{U}_i)_{i\leq N}$ and a bound t on the size of traitor coalitions. It computes a public key pk, descriptions of the plaintext and ciphertext domains $\mathcal{P}$ and $\mathcal{C}$, secret keys $(sk_i)_{i\leq N}$, and a tracing key tk (which may contain the sk_i's and additional data). It publishes $pk, \mathcal{P}$ and $\mathcal{C}$, and sends sk_i to user $\mathcal{U}_i$ for all $i \leq N$.

- Algorithm `Enc` can be run by any party. It takes as inputs a public key pk and a plaintext message $M \in \mathcal{P}$. It computes a ciphertext $C \in \mathcal{C}$.

- Algorithm `Dec` can be run by any user. It takes as inputs a secret key sk_i and a ciphertext message $C \in \mathcal{C}$. It computes a plaintext $P \in \mathcal{P}$.

- Algorithm `Trace` is explained below. If the input of `Trace`, i.e., the tracing key tk, is public then we say that the scheme supports public traceability.

We require that `Setup`, `Enc` and `Dec` run in polynomial time, and that with overwhelming probability over the randomness used by the algorithms, we have

$$\forall M \in \mathcal{P}, \forall i \leq N : \texttt{Dec}(sk_i, \texttt{Enc}(pk, M)) = M,$$

where pk and the sk_i's are sampled from `Setup`. We also require the encryption scheme to be IND-CPA.

Algorithm `Trace` aims at deterring coalitions of malicious users (traitors) from building an unauthorized decryption device. It takes as input tk and has access to a decryption device $\mathcal{D}$. `Trace` aims at disclosing the identity of at least one user that participated in building $\mathcal{D}$.

We consider the minimal black-box access model [BF99a]. In this model, the tracing authority has access to an oracle $\mathcal{O}^{\mathcal{D}}$ that itself internally uses $\mathcal{D}$. Oracle $\mathcal{O}^{\mathcal{D}}$ behaves as follows: It takes as input any pair $(C, M) \in \mathcal{C} \times \mathcal{P}$ and returns 1 if $\mathcal{D}(C) = M$ and 0 otherwise; the oracle only tells whether the decoder decrypts C to M or not. We assume that if M is sampled from $U(\mathcal{P})$ and C is the output of algorithm `Enc` given pk and M as inputs, then the decryption device decrypts correctly with probability significantly more than $1/|\mathcal{P}|$:

$$\Pr_{\substack{M \hookleftarrow U(\mathcal{P}) \\ C \hookleftarrow \texttt{Enc}(M)}} \left[\mathcal{O}^{\mathcal{D}}(C, M) = 1\right] \geq \frac{1}{|\mathcal{P}|} + \frac{1}{\lambda^c},$$

for some constant $c > 0$. This assumption is justified by the fact that otherwise the decryption device is not very useful. Alternatively, we may force the correct decryption probability to be non-negligibly close to 1, by using an all-or-nothing transform (see [KY02c]). We also assume that the decoder $\mathcal{D}$ is stateless/resettable, i.e., it cannot see and adapt to it being tested and replies independently to successive queries. Handling stateful pirate boxes has been investigated in [KY01b, KY01a].

In our scheme, algorithm `Trace` will only be a confirmation algorithm. It takes as input a set of (suspect) users $(\mathcal{U}_{i_j})_j$ of cardinality $k \leq t$, and must satisfy the following two properties:

- Confirmation. If the traitors are all in the set of suspects $(\mathcal{U}_{i_j})_{j\leq k}$, then it returns "User $\mathcal{U}_{i_{j_0}}$ is guilty" for some $j_0 \leq k$;

- SOUNDNESS. If it returns "User $\mathcal{U}_{i_{j_0}}$ is guilty" for some $j_0 \leq k$, then user $\mathcal{U}_{i_{j_0}}$ should indeed be a traitor.

The confirmation algorithm should run in polynomial-time. It may be converted into a (costly) full-fledge tracing algorithm by calling it on all subsets of users of cardinality t.

C.7.3 Confirmation and soundness of the proposed traitor tracing

We define the usefulness of the decoder as $\varepsilon := p_\infty - \frac{1}{|\mathcal{P}|} = p_\infty - \frac{1}{2}$. It can be estimated to within a factor 2 with probability $\geq 1 - 2^{-\Omega(n)}$ via the Chernoff bound.

We can now formally describe algorithm `Trace`. It proceeds in three steps, as follows.

1. It computes an estimate $\widetilde{\varepsilon}$ of the usefulness ε of the decoder to within a multiplicative factor of 2, which holds with probability $\geq 1 - 2^{-n}$. This can be obtained via Chernoff's bound, and costs $O(\varepsilon^{-2} n)$.

2. For i from 0 to t, algorithm `Trace` computes an approximation $\widetilde{p}_i$ of p_i to within an absolute error $\leq \frac{\widetilde{\varepsilon}}{16t}$, which holds with probability $\geq 1 - 2^{-n}$ (also using Chernoff's bound).

3. If $\widetilde{p}_i - \widetilde{p}_{i-1} > \frac{\widetilde{\varepsilon}}{8t}$ for some $i \leq t$, then `Trace` returns "User $\mathcal{U}_i$ is guilty." Otherwise, it returns "$\perp$."

Note that we are implicitly using the fact that $\mathcal{D}$ is stateless/resettable. Also, if ε is n^{-c} for some constant c, then `Trace` runs in polynomial time.

We start with the confirmation property.

Theorem C.7.1 Assume that decoder $\mathcal{D}$ was built using $\{sk_{i_j}\}_{j \leq k} \subseteq \{sk_i\}_{i \leq t}$. Under the assumption that (t, S)-LWE$_{m+1,\alpha}$ is hard, algorithm `Trace` returns "User $\mathcal{U}_i$ is guilty" for some $i \leq t$.

Proof: Wlog we may assume that the traitors in the coalition know all the secret keys $sk_1, \ldots, sk_t$. The hardness of (t, S)-LWE$_{m+1,\alpha}$ implies that the distributions `Enc`(0) and Tr_t are computationally indistinguishable. As a consequence, we have that p_t is negligibly close to p_∞ (the rounding to nearest of the samples from $\nu_{\alpha q}$ can be performed directly on the challenge samples, obliviously to any secret data, as in the proof of semantic security of Section C.4.1).

On the other hand, the acceptance probability p_0 is $\leq \frac{1}{2}$. As $p_t - p_0 > \frac{\varepsilon}{2}$ and $|\widetilde{p}_i - p_i| \leq \frac{\varepsilon}{8}$ for all i, we must have $\widetilde{p}_t - \widetilde{p}_0 > \frac{\varepsilon}{4} \geq \frac{\widetilde{\varepsilon}}{8}$, with probability exponentially close to 1. As a consequence, there must exist $i \leq t$ such that $\widetilde{p}_i - \widetilde{p}_{i-1} > \frac{\widetilde{\varepsilon}}{8t}$, and algorithm `Trace` returns "User $\mathcal{U}_i$ is guilty." ∎

Proving the soundness property is more involved. We exploit the hardness of (t, S)-LWE and rely on Theorem C.2.3 several times.

Theorem C.7.2 Assume that decoder $\mathcal{D}$ was built using $\{sk_{i_j}\}_{j \leq k}$. Under the parameter assumptions of Theorem C.2.3 with (k, n) in Theorem C.2.3 set to $(t+1, n+t+1)$, and the computational assumption that $(t+1, S)$-LWE$_{m+1,\alpha}$ is hard: if algorithm `Trace` returns "User $\mathcal{U}_{i_0}$ is guilty", then $i_0 \in \{i_j\}_{j \leq k}$.

Proof: Assume (by contradiction) that the traitors $\{\mathcal{U}_{i_j}\}_{j \leq k}$ with $k \leq t$ succeed in having `Trace` incriminate an innocent user $\mathcal{U}_{i_0}$ (with $i_0 \notin \{i_j\}_{j \leq k}$). We show that the algorithm $\mathcal{T}$ the traitors

use to build the pirate decoder may be exploited for solving $(t+1,S)$-LWE$_{m+1,\alpha}$. First, note that algorithm $\mathcal{T}$ provides an algorithm $\mathcal{A}$ that wins the following game.

Game_0. The game consists of three steps, as follows:

- $\texttt{Initialize}_0$: Sample $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$, $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$ and $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S}$ for $i \le t+1$.
- $\texttt{Input}_0$: Send $A^+ = (\vec{u}^t \| A)$ and $(\vec{x}_i)_{i\le t+1, i\neq i_0}$ to $\mathcal{A}$.
- $\texttt{Challenge}_0$: Sample $b \hookleftarrow U(\{0,1\})$. Send to $\mathcal{A}$ arbitrarily many samples from $U\left(\mathrm{Span}_{i\le i_0-1+b}(\vec{x}_i^+)^\perp\right) + \lfloor \nu_{\alpha q}\rceil^{m+1}$.

We say that $\mathcal{A}$ wins Game_0 if it finds the value of b with non-negligible advantage.

Algorithm $\mathcal{A}$ can be obtained from algorithm $\mathcal{T}$ by sampling plaintext M uniformly in $\{0,1\}$, and giving $(\vec{c} + (M|\vec{0}^t)^t, M)$ as input to $\mathcal{O}^{\mathcal{D}}$, where $\vec{c}$ is any sample from $\texttt{Challenge}_0$. We now introduce two variations of Game_0, which differ in the Initialize and Challenge steps.

Game_1. The game consists of three steps, as follows:

- $\texttt{Initialize}_1$: Sample $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$, $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$, $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),\sigma}$ for $i \le t+1$, and $\vec{b}_j^+ \hookleftarrow U(\mathrm{Span}_{i<i_0}(\vec{x}_i^+)^\perp)$ for $j \le t - i_0 + 2$.
- $\texttt{Input}_1$: Send $A^+ = (\vec{u}^t \| A)$ and $(\vec{x}_i)_{i\le t+1, i\neq i_0}$ to $\mathcal{A}$.
- $\texttt{Challenge}_1$: Sample $b \hookleftarrow U(\{0,1\})$. If $b=0$, then send to $\mathcal{A}$ arbitrarily many samples from $U\left(\mathrm{Span}_{i<i_0}(\vec{x}_i^+)^\perp\right) + \lfloor \nu_{\alpha q}\rceil^{m+1}$. If $b=1$, then send to $\mathcal{A}$ arbitrarily many samples from:

$$U\left(\mathrm{Im}\left[A^+|\vec{b}_1^+|\ldots|\vec{b}_{t-i_0+2}^+\right]\right) + \lfloor \nu_{\alpha q}\rceil^{m+1}.$$

As in Game_0, algorithm $\mathcal{A}$ wins Game_1 if it guesses b with non-negligible advantage.

Game_1' is as Game_1, except that if $b=0$ in the challenge step, then the samples sent to $\mathcal{A}$ are from the distribution $U\left(\mathrm{Span}_{i\le i_0}(\vec{x}_i^+)^\perp\right) + \lfloor \nu_{\alpha q}\rceil^{m+1}$. (The $\vec{b}_j$'s are sampled from $U(\mathrm{Span}_{i<i_0}(\vec{x}_i^+)^\perp)$ in both cases.)

Note that $\mathcal{A}$'s inputs in Game_0, Game_1 and Game_1' are identical (only the distributions of the Challenge steps vary). By the triangle inequality, if $\mathcal{A}$ wins Game_0 with some non-negligible advantage, then it may be used to win either Game_1 or Game_1' with non-negligible advantage. In our use of $\mathcal{A}$ to solve $(t+1,S)$-LWE, we may guess in which situation we are. We now consider the two situations separately.

First situation: Algorithm $\mathcal{A}$ wins Game_1 with non-negligible advantage. Then it may be used to solve $(t+1,S)$-LWE. Indeed, assume we have a $(t+1,S)$-LWE input $(A, \vec{u}, (\vec{x}_i)_{i\le t+1})$, and that we aim at distinguishing between the following distributions over $\mathbb{Z}_q^{m+1}$:

$$U\left(\mathrm{Im}(A^+)\right) + \nu_{\alpha q}^{m+1} \quad \text{and} \quad U\left(\mathrm{Span}_{i\le t+1}(\vec{x}_i^+)^\perp\right) + \nu_{\alpha q}^{m+1}.$$

To solve this problem instance, we sample $\vec{b}_j$ for $j \le t - i_0 + 2$ as in $\texttt{Initialize}_1$. Then we add a uniform $\mathbb{Z}_q$-linear combination of the $\vec{b}_j$'s to the $(t+1,S)$-LWE input samples. Since $m \ge t+n$, these $(t-i_0+2)$ vectors are linearly independent and none of them belongs to $\mathrm{Span}_{i_0\le i\le t+1}(\vec{x}_i^+)^\perp$, with probability $\ge 1-2^{-\Omega(n)}$. In that case, the transformation maps $U\left(\mathrm{Span}_{i\le t+1}(\vec{x}_i^+)^\perp\right)+\nu_{\alpha q}^{m+1}$ to $U\left(\mathrm{Span}_{i<i_0}(\vec{x}_i^+)^\perp\right)+\nu_{\alpha q}^{m+1}$, and maps $U(\mathrm{Im}(A^+))+\nu_{\alpha q}^{m+1}$ to $U(\mathrm{Im}[A^+|\vec{b}_1^+|\ldots|\vec{b}_{t-i_0+2}^+])+\nu_{\alpha q}^{m+1}$. We then round the samples to the nearest integer vectors, and Algorithm $\mathcal{A}$ distinguishes between

the resulting distributions, and its output is forwarded as output to the initial $(t+1, S)$-LWE instance.

Second situation: Algorithm $\mathcal{A}$ wins Game'_1 with non-negligible advantage. It seems quite similar to the first situation, but the following observation hints why its handling is somewhat more complex. In the first situation, the domains of the noiseless variants of the distributions to be distinguished are contained into one another: $\mathrm{Im}([A^+|\vec{b}_1|\dots|\vec{b}_{t-i_0+2}]) \subseteq \mathrm{Span}_{i<i_0}(\vec{x}_i^+)^\perp$. In the second situation, no such inclusion holds. The purpose of the sequence of games below is to map Game'_1 to recover such an inclusion setting.

Let us define Game_2 as being the same as Game'_1, but with the following updated first step:

- $\texttt{Initialize}_2$: Sample $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$, $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$, $\vec{b}_j \hookleftarrow U(\mathbb{Z}_q^m)$ and $v_j \hookleftarrow U(\mathbb{Z}_q)$ for $j \le t - i_0 + 2$, $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S}$ for $i \ge i_0$ and $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}'}(A'),S}$ for $i < i_0$, with
$$A' = \left[A|\vec{b}_1|\dots|\vec{b}_{t-i_0+2}\right] \quad \text{and} \quad \vec{u}' = (\vec{u}\|v_1\|\dots\|v_{t-i_0+2}).$$

We show that the residual distributions at the end of $\texttt{Initialize}_1$ and $\texttt{Initialize}_2$ are essentially the same. For that, we use Theorem C.2.3 twice. First, starting from $\texttt{Initialize}_1$, we swap the samplings of A and $\vec{u}$ with those of $(\vec{x}_i)_{i<i_0}$. This ensures that the residual distribution of $\texttt{Initialize}_1$ is within statistical distance $2^{-\Omega(n)}$ from the residual distribution of the following experiment: Sample $\vec{x}_i \hookleftarrow D_{\mathbb{Z}^m,S}$ for $i < i_0$, $A^+ = (\vec{u}^t\|A) \hookleftarrow U(\mathbb{Z}_q^{(m+1)\times n})$ conditioned on $\vec{x}_i^{+t} \cdot A^+ = \vec{0}$ for all $i < i_0$, $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S}$ for $i \in [i_0, t+1]$, and $\vec{b}_j^+ \hookleftarrow U(\mathrm{Span}_{i<i_0}(\vec{x}_i^+)^\perp)$ for $j \le t - i_0 + 2$. The samplings of the last $\vec{x}_i^+$'s and those of the $\vec{b}_j^+$'s being independent, their order can be exchanged. We can now apply Theorem C.2.3 a second time, to postpone the samplings of $(\vec{x}_i)_{i<i_0}$ after those of the $\vec{b}_j^+$'s. This gives us that the residual distributions of the above experiment and that of $\texttt{Initialize}_2$ are within statistical distance $2^{-\Omega(n)}$. Overall, we have shown that the residual distributions of $(A, \vec{u}, (\vec{b}_j)_j, (v_j)_j, (\vec{x}_i)_i)$ after $\texttt{Initialize}_1$ and $\texttt{Initialize}_2$ are within exponentially small statistical distance. Hence algorithm $\mathcal{A}$ wins Game_2 with non-negligible advantage.

Now, consider Game_3, which differs from Game_2 only in that $\vec{x}_{i_0}$ is also sampled from $D_{\Lambda^\perp_{-\vec{u}'}(A'),S}$.

- $\texttt{Initialize}_3$: Sample $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$, $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$, $\vec{b}_j \hookleftarrow U(\mathbb{Z}_q^m)$ and $v_j \hookleftarrow U(\mathbb{Z}_q)$ for $j \le t - i_0 + 2$, $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S}$ for $i > i_0$ and $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}'}(A'),S}$ for $i \le i_0$

As $\vec{x}_{i_0}$ is not given to $\mathcal{A}$ at step $\texttt{Input}_3$ and as it is not involved in the challenge distributions $U\left(\mathrm{Span}_{i<i_0}(\vec{x}_i^+)^\perp\right) + \lfloor \nu_{\alpha q} \rceil^{m+1}$ and $U(\mathrm{Im}[A^+|\vec{b}_1|\dots|\vec{b}_{t-i_0+2}]) + \lfloor \nu_{\alpha q} \rceil^{m+1}$, this modification does not alter the winning probability of $\mathcal{A}$: algorithm $\mathcal{A}$ also wins Game_3 with non-negligible advantage. Now, we again use Theorem C.2.3 twice, but with $(\vec{x}_i)_{i\le i_0}$: once for swapping the samplings of these $\vec{x}_i$'s with A^+ and the $\vec{b}_j^+$'s, and once for swapping the samplings of A^+ and these $\vec{x}_i$'s. This shows that algorithm $\mathcal{A}$ wins Game_4 with non-negligible advantage, where Game_4 differs from Game_3 only in its first step, as follows.

- $\texttt{Initialize}_4$: Sample $A \hookleftarrow U(\mathbb{Z}_q^{m\times n})$, $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$, $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S}$ for $i \le t$, and $\vec{b}_j^+ \hookleftarrow U(\mathrm{Span}_{i\le i_0}(\vec{x}_i^+)^\perp)$ for $j \le t - i_0 + 2$.

The situation we are in now is very similar to that in the first situation, where $\mathcal{A}$ was supposed to win Game_1. The arguments used in the first situation readily carry over here (up to

replacing $\mathrm{Span}_{i<i_0}\vec{x}_i^+$ and $\mathrm{Span}_{i\geq i_0}\vec{x}_i^+$ by $\mathrm{Span}_{i\leq i_0}\vec{x}_i^+$ and $\mathrm{Span}_{i>i_0}\vec{x}_i^+$, respectively). ▮

C.8 Basic results on lattices

Gentry et al. [GPV08] showed that Klein's algorithm [Kle00] can be used to sample from $D_{L,S,\vec{c}}$. This discrete Gaussian sampler was later refined in [BLP+13].

Lemma C.8.1 [[BLP+13, Le. 2.3]] There exists a ppt algorithm that, given a basis $(\vec{b}_i)_i$ of an n-dimensional lattice L, $\vec{c}\in\mathrm{Span}(L)$ and $S\in\mathbb{R}^{m\times m}$ invertible satisfying $\sqrt{\ln(2n+4)/\pi}\cdot\max_i\|S^{-t}\vec{b}_i\|\leq 1$, returns a sample from $D_{L,S,\vec{c}}$.

The following basic results on lattice Gaussians are usually stated for full-rank lattices. As we consider lattices that are not full-rank, we adapt them. The proofs can be modified readily to handle this more general setup, by relying on an isometry from $\mathrm{Span}(L)$ to $\mathbb{R}^n$ with $n=\dim L$.

Lemma C.8.2 [Adapted from [AGHS13, Le. 3]] For any n-dimensional lattice $L\subseteq\mathbb{R}^m$, $\vec{c}\in\mathrm{Span}(L)$ and $S\in\mathbb{R}^{m\times m}$ invertible satisfying $\sigma_m(S)\geq\eta_\varepsilon(L)$ with $\varepsilon\in(0,1/2)$, we have $\Pr_{\vec{b}\hookleftarrow D_{L,S,\vec{c}}}[\|\vec{b}-\vec{c}\|\geq\sigma_1(S)\cdot\sqrt{n}]\leq 2^{-n+2}$.

Lemma C.8.3 [Adapted from [MR07, Le. 4.4]] For any lattice $L\subseteq\mathbb{R}^m$, $\vec{c}\in\mathrm{Span}(L)$ and $S\in\mathbb{R}^{m\times m}$ invertible satisfying $\sigma_m(S)\geq\eta_\varepsilon(L)$ with $\varepsilon\in(0,1/2)$, we have $\rho_{S,\vec{c}}(L)\in(\frac{1-\varepsilon}{1+\varepsilon},1)\cdot\rho_S(L)$.

Lemma C.8.4 [Special case of [Pei10, Th. 3.1]] Let $S_1,S_2\in\mathbb{R}^{m\times m}$ invertible, $\vec{c}\in\mathbb{R}^m$, and $\Lambda_1,\Lambda_2\subseteq\mathbb{R}^m$ be full-rank lattices. Assume that $1\geq\eta_\varepsilon(S_1^{-1}\Lambda_1)$ and $1\geq\eta_\varepsilon(\sqrt{(S_1S_1^t)^{-1}+(S_2S_2^t)^{-1}}\cdot\Lambda_2)$ for some $\varepsilon\in(0,1/2)$. If $\vec{x}_2\hookleftarrow D_{\Lambda_2,S_2,\vec{0}}$ and $\vec{x}_1\hookleftarrow D_{\Lambda_1,S_1,\vec{c}-\vec{x}_2}$, then the residual distribution of $\vec{x}_1$ is within statistical distance 8ε of $D_{\Lambda_1,S,\vec{c}}$, with $S=\sqrt{S_1S_1^t+S_2S_2^t}$.

Lemma C.8.5 [[AR13, Th. 5.1]] Let $n\geq 100$, $\varepsilon\in(0,1/1000)$, $\sigma\geq 9\sqrt{\ln(2n(1+1/\varepsilon))/\pi}$ and $m\geq 30n\log(\sigma n)$. Let $\vec{c}\in\mathbb{R}^m$ and $X\hookleftarrow D_{\mathbb{Z},\sigma}^{m\times n}$. Let $S\in\mathbb{R}^{m\times m}$ with $\sigma_m(S)\geq 10n\sigma\log^{3/2}(nm\sigma/\varepsilon)$. Then, with probability $\geq 1-2^{-n}$ over the choice of X, we have $X^t\cdot\mathbb{Z}^m=\mathbb{Z}^n$ and $\Delta(X^t\cdot D_{\mathbb{Z}^m,S,\vec{c}},D_{\mathbb{Z}^n,SX,S^t\vec{c}})\leq 2\varepsilon$.

Lemma C.8.6 [[AGHS13, Le. 8]] Let $n\geq 1$, $m\geq 2n$, and $\sigma\geq C\cdot\sqrt{n}$ for some absolute constant C. Let $X\hookleftarrow D_{\mathbb{Z},\sigma}^{m\times n}$. Then, except with probability $2^{-\Omega(m)}$, we have $\sigma_n(X)\geq\Omega(\sigma\sqrt{m})$.

C.9 Missing proofs of Section C.3

Proof of Lemma C.3.2.

We apply Lemma C.8.5 with S invertible chosen so that $SX=\sigma_2 I_n$ for some $\sigma_2>\sigma_1$, thus obtaining an unskewed Gaussian distribution $D_{\mathbb{Z}^n,\sigma_2}$. The scaling σ_2 is chosen sufficiently large so that the assumptions of Lemmas C.8.5 and C.8.6 hold.

We first sample X from $D_{\mathbb{Z},\sigma_1}^{m\times n}$, using Lemma C.8.1. By Lemma C.8.5 (that we use with $\varepsilon=2^{-n}$), its row $\mathbb{Z}$-span is $\mathbb{Z}^n$ with probability $\geq 1-2^{-n}$: we now assume that we are in this situation. Then we sample $\vec{r}$ from $D_{\mathbb{Z}^m,S}$, using Lemma C.8.1 again, for some invertible matrix $S\in\mathbb{R}^{m\times m}$ chosen as described below. Finally, we set $\vec{x}=\vec{c}+X^t\cdot\vec{r}$. If the assumptions of Lemma C.8.5 are satisfied, we know that, except with probability $\leq 2^{-n}$ over X, the distribution of $\vec{x}$ is, conditioned on X, within statistical distance 2ε of $D_{\mathbb{Z}^n,SX,\vec{c}}$.

We build S using the singular value decomposition $X = U_X \cdot \mathrm{Diag}((\sigma_i(X))_{i\le n}) \cdot V_X$, where $U_X \in \mathbb{R}^{m\times n}$ and $V_X \in \mathbb{R}^{n\times n}$ are orthogonal matrices. We define $S = U_S \cdot \mathrm{Diag}((s_i)_{i\le m}) \cdot V_S$ as follows: we set $U_S^t = \begin{bmatrix} V_X & \vec{0} \\ \vec{0} & I_{m-n} \end{bmatrix}$ and $V_S^t = [U_X | U_X^\perp]$, where $U_X^\perp$ is an orthonormal basis for the orthogonal of $U_X \cdot \mathbb{R}^n$; we also set $s_i = \sigma_2/\sigma_i(X)$ for $i \le n$ and $s_i = \sigma_n(S)$ for $i > n$. This leads to $SX = \sigma_2 \cdot I_n$, as required.

To check that the assumptions of Lemma C.8.5 are satisfied, note that the smallest singular value of S is $\sigma_m(S) = s_1 = \sigma_2/\sigma_1(X)$. Hence the assumption $\sigma_m(S) \ge 10n\sigma_1 \log^{3/2}(nm\sigma_1/\varepsilon)$ is satisfied if $\sigma_2 \ge \sigma_1(X) \cdot 10n\sigma_1 \log^{3/2}(nm\sigma_1/\varepsilon)$. The latter holds by the choice of σ_2, using the fact that $\sigma_1(X) \le \|X\| \le \sqrt{m} \cdot \sigma_1$. The second inequality holds with probability $\ge 1 - n2^{-m+2}$, using the union bound and Lemma C.8.2.

Finally, the bound on $\|\vec{r}\|$ follows from Lemma C.8.2 and the facts that $\sigma_1(S) = \sigma_2/\sigma_n(X)$ and $\sigma_n(X) \ge \Omega(\sigma_1\sqrt{m})$ except with probability $2^{-\Omega(m)}$, by Lemma C.8.6.

Proof of Lemma C.3.6.

Let D_0 denote the desired distribution for $(A', \vec{u}', X)$. We first apply Theorem C.2.3 (with the theorem parameters $m, n, k, \sigma_1(S), \sigma_m(S)$ having the values $m + 2n$, $3n$, $n/(c \log n)$, σ' and σ, respectively) to show that D_0 is within statistical distance $2^{-\Omega(n)}$ of the distribution D_1 on tuples $(A', \vec{u}', X)$ defined as follows: $\vec{u}' \in \mathbb{Z}_q^{3n}$ is sampled uniformly, $X \in \mathbb{Z}^{k\times(m+2n)}$ has its ith row $\vec{x}_i$ independently sampled from $D_{\mathbb{Z}^{m+2n}, S, \vec{c}_i}$, and $A' \in \mathbb{Z}_q^{(m+2n)\times 3n}$ is sampled uniformly from the set of solutions to $\vec{x}_i^t \cdot A' = -\vec{u}'^t \bmod q$. Indeed, the assumptions of the theorem are satisfied by our choice of parameters.

Next, let $A' = \left(\frac{A}{B}\middle| C\right)$, where $A \in \mathbb{Z}_q^{m\times n}$, $B \in \mathbb{Z}_q^{2n\times n}$ and $C \in \mathbb{Z}_q^{(m+2n)\times 3n}$. Note that in the distribution D_1, *all* of A' is chosen uniformly from the set of solutions to $X \cdot A' = U' \bmod q$ (where $U' \in \mathbb{Z}_q^{k\times 3n}$ consists of k copies of $\vec{u}'^t$). We now show that D_1 is within statistical distance $2^{-\Omega(n)}$ to the distribution D_2 that is defined as D_1, except that in D_2, the submatrix $A \in \mathbb{Z}_q^{m\times n}$ is chosen independently uniformly at random, and then B, C are chosen uniformly from the set of solutions to $X \cdot A' = U' \bmod q$. The distribution of $(C, \vec{u}', X)$ is the same in D_1 and D_2, by definition. The condition on (A, B) in D_1 is $X_1 \cdot A + X_2 \cdot B = U \bmod q$, where $X_1 \in \mathbb{Z}^{k\times m}$ and $X_2 \in \mathbb{Z}^{k\times 2n}$ are the left and right submatrices of X, respectively, and $U \in \mathbb{Z}_q^{k\times n}$ consists of the n left columns of U'. If X_2 has full rank k over $\mathbb{Z}_q$, then for every choice of $A \in \mathbb{Z}_q^{m\times n}$, the latter equation has the same number of solutions for $B \in \mathbb{Z}_q^{2n\times n}$ (namely $q^{(2n-k)\cdot n}$). Hence, conditioned on X_2 having rank k, the distribution of (A, B) is the same in D_1 and D_2. Therefore, the statistical distance $\Delta(D_1, D_2)$ is $2^{-\Omega(n)}$ if the probability that X_2 has rank k in D_1 is $2^{-\Omega(n)}$. The latter holds by Lemma C.2.4 and our choice of parameters.

Finally, let D_3 denote the distribution of $(A', \vec{u}', X)$ in the reduction. We show below that $\Delta(D_2, D_3) \le 2^{-\Omega(n/\log n)}$, which completes the proof.

First, we consider the distribution of X. By Corollary C.3.4, we have that, in distribution D_3, the last $2n$ columns of X are within statistical distance $\varepsilon_1 = 2^{-\Omega(n/\log n)}$ of $D_{\mathbb{Z},\sigma}^{k\times n} \times D_{\mathbb{Z},\sigma',\vec{\delta}_1} \| \ldots \| D_{\mathbb{Z},\sigma',\vec{\delta}_k}$. Since the first m columns of X are independently distributed as $D_{\mathbb{Z},\sigma}^{k\times m}$ in both D_2 and D_3, it follows that the distribution of X in D_3 is within statistical distance $\varepsilon_1 = 2^{-\Omega(n/\log n)}$ of its distribution $D_{\mathbb{Z}^{m+2n}, S}$ in D_2.

Next, we consider the distribution of A' given some fixed $(\vec{u}', X)$. Observe that the only difference between these conditional distributions in D_2 and D_3 is that in D_3, matrix B is defined as the unique solution to $(\mathbf{1}|\overline{X_1}) \cdot (\vec{u}^t \| A) + \overline{X_2} \cdot B = 0 \bmod q$, whereas in D_2, matrix B is chosen uniformly among the solutions to $(\mathbf{1}|X_1) \cdot (\vec{u}^t \| A) + X_2 \cdot B = 0 \bmod q$, where X_1, X_2 are the top k rows of $\overline{X}_1, \overline{X}_2$, respectively. We show that these conditional distributions are within statistical

distance $\varepsilon_2 = 2^{-\Omega(n)}$, which immediately implies that $\Delta(D_2, D_3) \leq \varepsilon_1 + \varepsilon_2 = 2^{-\Omega(n/\log n)}$, as required.

To see this, let X_1', X_2' denote the bottom $2n-k$ rows of $\overline{X}_1, \overline{X}_2$, respectively. Fix $X_1, X_2, X_2', \vec{u}$, A, with A such that $\eta_{2^{-n}}(\Lambda^{\perp}(A)) = O(\sqrt{n \log m}) \cdot q^{\frac{n}{m}}$. By Lemma C.2.1, this condition holds with probability $1 - 2^{-\Omega(n)}$ over the uniform choice of A. Let B^* denote any solution to $(\mathbf{1}|X_1) \cdot (\vec{u}^t \| A) + X_2 \cdot B = 0 \bmod q$. Let $p(B^*)$ denote the probability that $B = B^*$ in distribution D_3, conditioned on $X_1, X_2, X_2', \vec{u}, A$. We show that $p(B^*)$ is of the form $(1 + \varepsilon_{B^*}) \cdot K$ for any such B^*, for $\varepsilon_{B^*} \leq 2^{-\Omega(n)}$ and some normalization constant K independent of B^*. From this it follows immediately that, in D_3, the conditional distribution of B is within distance $2^{-\Omega(n)}$ of the uniform distribution on the set of solutions to $(\mathbf{1}|X_1) \cdot (\vec{u}^t \| A) + X_2 \cdot B = 0 \bmod q$, which is the conditional distribution of B in D_2, and our claim follows immediately. The probability $p(B^*)$ is the probability that $X_1' \cdot A + X_2' \cdot B = U \bmod q$, conditioned on $X_1, X_2, X_2', \vec{u}, A$. Let $\vec{x}_{1,i}' \in \mathbb{Z}^m$ and $\vec{x}_{2,i}' \in \mathbb{Z}^{2n}$ denote the ith rows of X_1' and X_2', respectively, for $i \leq 2n-k$. Observe that the set of solutions for $\vec{x}_{1,i}' \in \mathbb{Z}^m$ to $\vec{x}_{1,i}'^t \cdot A + \vec{x}_{2,i}'^t \cdot B^* = -\vec{u}^t \bmod q$ is the coset $\Lambda^{\perp}_{-\vec{u}-\vec{x}_{2,i}' \cdot B^*}(A)$ and, since $\vec{x}_{1,i}'$ is independently distributed as $D_{\mathbb{Z}^m,\sigma}$ for each i, it follows that

$$p(B^*) = \prod_{i \leq 2n-k} D_{\mathbb{Z}^m,\sigma}(\Lambda^{\perp}_{-\vec{u}-\vec{x}_{2,i}'^t \cdot B^*}(A)) = \prod_{i \leq 2n-k} \rho_{\sigma,\vec{c}_i}(\Lambda^{\perp}(A))/\rho_{\sigma}(\mathbb{Z}^m),$$

for some $\vec{c}_i \in \mathbb{Z}_q^m$ such that $\vec{c}_i^t \cdot A = \vec{u}^t + \vec{x}_{2,i}'^t \cdot B^* \bmod q$. By Lemma C.8.3, using the choice of $\sigma \geq \eta_{2^{-n}}(\Lambda^{\perp}(A)) = O(\sqrt{n \log m}) \cdot q^{\frac{n}{m}}$, we have $\rho_{\sigma,\vec{c}_i}(\Lambda^{\perp}(A)) = (1 + \varepsilon'_{B^*}) \cdot \rho_{\sigma}(\Lambda^{\perp}(A))$ for some $\varepsilon'_{B^*} \leq 2^{-\Omega(n)}$. It follows that $p(B^*) \sim 1 + \varepsilon_{B^*}$ for some $\varepsilon_{B^*} \leq n 2^{-\Omega(n)}$.

Proof of Lemma C.3.7.

In our proof, we need to use a bound on the probability that a collection of vectors $\vec{t}_1, \ldots, \vec{t}_{d+w}$ uniformly and independently sampled from a linear subspace X of dimension d over $\mathbb{Z}_q$, spans X. This is given by the following proposition.

Proposition C.9.1 Let $d, w, q > 0$ with q prime. Let X denote a d-dimensional $\mathbb{Z}_q$-linear space. Let $\vec{t}_1, \ldots, \vec{t}_{d+w} \in X$ be independently sampled from $U(X)$. Then we have $\mathrm{Span}_{i \leq d+w}(\vec{t}_i) = X$, with probability $\geq 1 - 2^{d+w}/q^{w+1}$.

Proof: For $i \leq d+w$, let χ_i denote the Bernoulli random variable that is 0 if $\vec{t}_i \in \mathrm{Span}_{j<i}(\vec{t}_j)$ and 1 else. Let r_i denote the rank of $\mathrm{Span}_{j \leq i}(\vec{t}_j)$. Since $r_i = r_{i-1} + \chi_i$, we have $r_{d+w} = \sum_{i=1}^{d+w} \chi_i$. Let S denote the set of binary vectors of length $d+w$ and weight $< d$. Then it suffices to bound the probability that $\vec{\chi} = (\chi_1, \ldots, \chi_{d+w}) \in S$. To do so, let $\vec{\chi}' = (\chi_1', \ldots, \chi_{d+w}') \in \{0,1\}^{d+w}$ denote any fixed vector in S. Note that for any $i \leq d+w$, we have $\Pr[\chi_i = 0 | \chi_j = \chi_j' \text{ for } j < i] = q^{\sum_{j<i} \chi_j'}/q^d \leq 1/q$, since $\vec{\chi}' \in S$. It follows that $\Pr[\vec{\chi} = \vec{\chi}'] \leq 1/q^z$, where z denotes the number of zero entries in $\vec{\chi}'$. Since the weight of $\vec{\chi}'$ is $< d$, we have $z > d + w - d = w$, so $\Pr[\vec{\chi} = \vec{\chi}'] \leq 1/q^{w+1}$. Taking a union bound over all $\vec{\chi}' \in S$, and using $|S| \leq 2^{d+w}$ completes the proof. ∎

We now prove the lemma. We have $\vec{b} = \frac{1}{q}A^+ \cdot \vec{s} + \vec{e} \in \mathbb{T}^{m+1}$ with $\vec{e}$ sampled from ν_α^{m+1} and $\vec{s}$ from $U(\mathbb{Z}_q^n)$, so

$$\begin{aligned}\vec{b'} &= T \cdot \vec{b} + \frac{1}{q}C^+ \cdot \vec{s'} + \sqrt{\Sigma} \cdot \vec{e'} \\ &= \frac{1}{q}TA^+ \cdot \vec{s} + \frac{1}{q}C^+ \cdot \vec{s'} + T \cdot \vec{e} + \sqrt{\Sigma} \cdot \vec{e'} \\ &= \frac{1}{q}A'^+ \cdot \begin{bmatrix} \vec{s} \\ \vec{s'} \end{bmatrix} + T \cdot \vec{e} + \sqrt{\Sigma} \cdot \vec{e'}.\end{aligned}$$

Now, since $\vec{s}$ and $\vec{s'}$ are uniform and independent, we have $\frac{1}{q}A'^+ \cdot [\vec{s}\|\vec{s'}]$ is uniformly distributed in $\mathrm{Im}(A'^+)$. Moreover, the vector $T \cdot \vec{e}$ is normally distributed with covariance matrix $\alpha^2 \cdot TT^t$, while $\sqrt{\Sigma}\vec{e'}$ is independent and normally distributed with covariance matrix $\Sigma = \alpha'^2 I_{m+1+2n} - \alpha^2 TT^t$ (we show below that Σ is indeed a valid covariance matrix, i.e., is positive definite, so that $\sqrt{\Sigma}$ exists, except with probability $2^{-\Omega(n/\log n)}$). Therefore, the vector $T \cdot \vec{e} + \sqrt{\Sigma}\vec{e'}$ has distribution $\nu_{\alpha'}^{m+1+2n}$, as required.

It remains to show that $\Sigma = \alpha'^2 I_{m+1+2n} - \alpha^2 TT^t$ is a positive definite matrix, with overwhelming probability over the choice of $\overline{X}_1$ and $\overline{X}_2$. By definition, the singular values of Σ are of the form $\alpha'^2 - \alpha^2\sigma_i(T)^2$, where the $\sigma_i(T)$'s are the singular values of T. It therefore suffices to show that $\alpha'^2 > \alpha^2\sigma_1(T)^2$, where $\sigma_1(T)$ is the largest singular value of T. We have $\sigma_1(T) \leq \sqrt{m+1}\|T\|$ (by Schwarz's inequality). Each column of T has norm $\leq \sqrt{1 + (m+1)\|\overline{X}_2^{-1}\|^2 t^2}$, where t denotes the maximum column norm of the matrix $(\vec{1}|\overline{X}_1)$. Since the columns of $\overline{X}_1$ are sampled from $D_{\mathbb{Z}^{2n},\sigma}$, we have by Lemma C.8.2 that $t \leq \sigma \cdot \sqrt{2n}$, and by Corollary C.3.4 that $\|\overline{X}_2^{-1}\| = O(\sigma' n)$, with both bounds holding with probability $\geq 1 - 2^{-\Omega(n/\log n)}$. It follows that $\sigma_1(T) = O(mn^{3/2}\sigma\sigma')$, and hence the assumption that $\alpha' = \Omega(mn^{3/2}\sigma\sigma'\alpha)$ allows us to complete the proof.

Proof of Lemma C.3.8.

We have $\vec{b} = \frac{1}{q}\vec{y} + \vec{e} \in \mathbb{T}^{m+1}$ with $\vec{e}$ sampled from ν_α^{m+1} and $\vec{y}$ from $U(\mathbb{Z}_q^{m+1})$, so

$$\vec{b'} = \frac{1}{q}T \cdot \vec{y} + \frac{1}{q}C^+ \cdot \vec{s'} + T \cdot \vec{e} + \sqrt{\Sigma}\vec{e'} = \frac{1}{q}[T|C^+] \cdot \begin{bmatrix} \vec{y} \\ \vec{s'} \end{bmatrix} + T \cdot \vec{e} + \sqrt{\Sigma}\vec{e'}.$$

Now, since $\vec{y}$ and $\vec{s'}$ are uniform and independent, we have that $\frac{1}{q}[T|C^+] \cdot [\vec{y}\|\vec{s'}]$ is uniform in $\mathrm{Im}([T|C^+])$.

By construction of T and C, we have that $\mathrm{Im}([T|C^+])$ is a subspace of $X^\perp = \left(\mathrm{Span}_{i \leq k}(\vec{x}_i^+)^\perp\right)$. We claim that in fact $\mathrm{Im}([T|C^+]) = X^\perp$, except with probability $2^{-\Omega(n/\log n)}$ over the choice of the $\vec{x}_i$'s and C^+. Indeed, by Lemmas C.3.6 and C.2.4, with probability $\geq 1 - 2^{-\Omega(n/\log n)}$, the vectors $\vec{x}_1^+, \ldots, \vec{x}_k^+$ are linearly independent over $\mathbb{Z}_q$ and hence the subspace $X^\perp$ has dimension $m+1+2n-k$. Now, the $m+1$ columns of T are linearly independent. Hence, it suffices to show that the $3n$ projections of the columns of C^+ on the orthogonal complement of $\mathrm{Im}(T) \subseteq X^\perp$ span that $(2n-k)$-dimensional space. As these projections are uniform, we can apply Proposition C.9.1, which tells us this is the case with probability $\geq 1 - 2^{3n}/q^{n+k+1} \geq 1 - 2^{-\Omega(n)}$.

We have showed that $\frac{1}{q}[T|C^+] \cdot [\vec{y}\|\vec{s'}]$ is within statistical distance $\leq 2^{-\Omega(n)}$ of $\frac{1}{q}U(X^\perp)$, with probability $\geq 1 - 2^{-\Omega(n/\log n)}$ over the choice of X. As shown in Lemma C.3.7, we also have that the noise term $T \cdot \vec{e} + \sqrt{\Sigma}\vec{e'}$ is within statistical distance $2^{-\Omega(n)}$ of the distribution ν_α^{m+1+2n}, as required.

Proof of Lemma C.3.9.

Consider the following sequence of games.

Let Game_0 denote the original (k, S)-LWE game, in which the distinguisher $\mathcal{B}$ receives an instance of the form $(r, \vec{y})$, where $r = (A, \vec{u}, \{\vec{x}_i\}_{i\leq k})$ with $A \hookleftarrow U(\mathbb{Z}_q^{m'\times n'})$, $\vec{u} \hookleftarrow U(\mathbb{Z}_q^{n'})$ and $\vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{0}}$ for $i \leq k$, and $\vec{y} \in \mathbb{T}^{m'+1}$ is a sample from either the distribution

$$D_0(r) = \frac{1}{q} \cdot U\Big(\mathrm{Im}\Big(\frac{\vec{u}^t}{A}\Big)\Big) + \nu_\alpha^{m'+1}$$

or the distribution

$$D_1(r) = \frac{1}{q} \cdot U\Big(\mathrm{Span}_{i\leq k}\Big(\frac{1}{\vec{x}_i}\Big)^\perp\Big) + \nu_\alpha^{m'+1}.$$

Let $\varepsilon_0(\mathcal{B}) = \varepsilon$ denote the advantage of $\mathcal{B}$ in distinguishing between these distributions in Game_0. Similarly, in the following, we let $\varepsilon_i(\mathcal{B})$ denote the corresponding attacker advantage in Game_i.

Let Game_1 denote a modification of Game_0 in which we change the distribution of A by rejection sampling as follows: we sample A uniformly from $\mathbb{Z}_q^{m'\times n'}$, but reject and resample A if $\eta_{2^{-n}}(A) > 4q^{n'/m'}\sqrt{\log(2m'(1+2^n))/\pi} = O(\sqrt{n})$. By Lemma C.2.1, the rejection probability is $2^{-\Omega(n)}$, and therefore, the distinguishing advantage $\varepsilon_1(\mathcal{B})$ satisfies $\varepsilon_1(\mathcal{B}) \geq \varepsilon_0(\mathcal{B}) - 2^{-\Omega(n)}$.

Let Game_2 denote a modification of Game_1 in which we change the distribution of the hint $\vec{x}_i$'s in r from the zero-centered distribution $D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{0}}$ of (k, S)-LWE to the non-zero centered distribution $D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{c}_i}$ of (k, S, C)-LWE. We observe that, since given $r = (A, \vec{u}, \{\vec{x}_i\}_{i\leq k})$, one can efficiently sample a vector $\vec{y}$ from either distribution $D_0(r)$ or $D_1(r)$, the (k, S, C)-LWE problem has the *public samplability* property needed to apply Lemma C.2.6. It follows that there exists a distinguisher $\mathcal{B}'$ in Game_2 with run-time $T' = O(\mathcal{P}oly(m') \cdot \varepsilon_1(\mathcal{B})^{-2} \cdot T)$ and advantage $\varepsilon_2(\mathcal{B}') \geq \Omega((\varepsilon_1(\mathcal{B}) - O(2^{-n}))^3/R)$, where $R = R(\Phi_1\|\Phi_2)$ denotes the RD between the distributions Φ_1 and Φ_2 of r in Game_1 and Game_2, respectively. Since the $\vec{x}_i$'s are independent, and conditioning on $\vec{u}$ and A, we have, from the multiplicative property of the RD, that

$$\begin{aligned} R &\leq \max_{\vec{u}\in\mathbb{Z}_q^{n'}} \prod_{i\leq k} R\left(D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{0}}\|D_{\Lambda^\perp_{-\vec{u}}(A),S,\vec{c}_i}\right) \\ &\leq \max_{\vec{c}\in\mathbb{R}^{m'}} \prod_{i\leq k} R\left(D_{\Lambda^\perp(A),S,\vec{c}}\|D_{\Lambda^\perp(A),S,\vec{c}+\vec{c}_i}\right). \end{aligned}$$

The latter can be bounded from above by applying Lemma C.2.5. The smoothing condition of the lemma holds since $\sigma_{m'}(S) \geq \omega(\sqrt{n})$, so we have by the rejection step of the previous game that $\sigma_{m'}(S) \geq \eta_{2^{-n}}(A)$. This leads to

$$R \leq \prod_{i\leq k} \exp(2^{-n+3} + 2\pi\|\vec{c}_i\|^2/\sigma_{m'}(S)^2) \leq \exp(k \cdot (2^{-n+3} + 2\pi\|C\|^2/\sigma_{m'}(S)^2)).$$

Finally, let Game_3 denote a modification of Game_2, in which we undo the rejection sampling of A introduced in Game_1, sampling it uniformly instead. By the same argument as in the change from Game_0 to Game_1, the advantage of $\mathcal{B}'$ in Game_3 satisfies $\varepsilon_3(\mathcal{B}') \geq \varepsilon_2(\mathcal{B}') - 2^{-\Omega(n)}$. Note that the instance distribution in Game_3 is identical to that of the (k, S, C)-LWE game, so $\mathcal{B}'$ has advantage $\varepsilon_3(\mathcal{B}')$ against (k, S, C)-LWE, as required.

C.10 Missing proof of Section C.5

Proof of Lemma C.5.4.

Our aim is to reduce $\mathrm{LWE}'_{\alpha,m+1}$ to distinguishing Game_1 and Game_0. Assume we have the following multiple LWE′ input $(B, \vec{y}_{i+1}, \ldots, \vec{y}_N)$ where $B \hookleftarrow U(\mathbb{Z}_q^{m\times n})$, and $\vec{y}_j = B\vec{s}_j + \vec{e}_j$ with $\vec{s}_j \hookleftarrow U(\mathbb{Z}_q^n)$ and either $\vec{e}_j \hookleftarrow U(\mathbb{Z}_q^m)$ for all j, or $\vec{e}_j \hookleftarrow D_{\mathbb{Z}^m,\alpha q}$ for all j. Our goal is to exploit a distinguisher between Game_0 and Game_1 to decide whether the $\vec{e}_j$'s are Gaussian or uniform. We simulate Game_1 and Game_0 as follows (depending on the nature of $\vec{e}_i$):

- Sample $A \in \mathbb{Z}_q^{m\times n}$ and $T \in \mathbb{Z}^{m\times m}$ such that A is uniform conditioned on $B^t \cdot A = 0$ and T is a full-rank basis of $\Lambda^\perp(A)$ satisfying $\|T\| \leq O(\sqrt{mn \log q \log m})$. This can be performed in ppt using [GKV10, Le. 4].
- Define H as a randomized basis of the kernel of B. It is $m\times(m-n)$ with probability $2^{-\Omega(n)}$. The distribution of the pair (A, H) is within statistical distance $2^{-\Omega(n)}$ of its distribution in Game_0 and Game_1.
- Sample $\vec{u} \hookleftarrow U(\mathbb{Z}_q^n)$ and sample the keys $\vec{x}_1, \ldots, \vec{x}_i \hookleftarrow D_{\Lambda^\perp_{-\vec{u}}(A),S}$ by using the trapdoor matrix T (this is why σ' must be set sufficiently large). Compute $\vec{h}_j^t = -\vec{x}_j^t \cdot H$ for $j \leq i$.
- Using linear algebra, find $\vec{c}$ such that $\vec{c}^t \cdot A = \vec{u}^t$. For each $j \in [i+1, N]$:
  - Compute $\vec{u}_j^t = \vec{y}_j^t \cdot A$. Since $\vec{y}_j = B \cdot \vec{s}_j + \vec{e}_j$, we have $\vec{u}_j^t = \vec{e}_j^t \cdot A$ (although we would prefer $\vec{u}^t = \vec{e}_j^t \cdot A$).
  - Sample $\vec{e}'_j \hookleftarrow \vec{c} - \vec{y}_j + D_{\Lambda^\perp(A),S_2,-\vec{c}+\vec{y}_j}$ where $S_2 = \sqrt{SS^t - \alpha^2 q^2 I_m}$ (these are diagonal matrices), using T. Since $\vec{y}_j - \vec{e}_j \in \Lambda^\perp(A)$, we can rewrite the latter as $\vec{e}'_j \hookleftarrow \vec{c} - \vec{e}_j + D_{\Lambda^\perp(A),S_2,-\vec{c}+\vec{e}_j}$.
  - Compute $\vec{z}_j = \vec{y}_j + \vec{e}'_j$. We now have $(\vec{e}_j^t + \vec{e}'^t_j) \cdot A = \vec{z}_j^t \cdot A = \vec{c}^t \cdot A = \vec{u}^t$.
  - Set $\vec{h}_j^t = -\vec{z}_j^t \cdot H$. Note that $\vec{h}_j^t = -(\vec{e}_j^t + \vec{e}'^t_j) \cdot H$.
- Return $A, \vec{u}, H, (\vec{x}_j)_{j\leq i}$ and $(\vec{h}_j)_{j\leq N}$.

We observe that for each $j \in [i+1, N]$, we have $\vec{z}_j = \vec{y}_j + \vec{e}'_j = B \cdot s_j + (\vec{e}_j + \vec{e}'_j)$. We consider two cases.

- When $\vec{e}_j \hookleftarrow D_{\mathbb{Z}^m,\alpha q}$, the residual distribution of $D_{\Lambda^\perp(A),S_2,-\vec{c}+\vec{e}_j}$ is within negligible statistical distance to $D_{\Lambda^\perp(A),S,-\vec{c}}$; this is provided by Lemma C.8.4, whose assumptions are satisfied (thanks to the second lower bound on σ') and to Lemma C.2.1; consequently, the residual distribution of $\vec{e}_j + \vec{e}'_j$ is negligibly close to the distribution $\vec{c} + D_{\Lambda^\perp(A),S,-\vec{c}}$, and hence the distribution of $\vec{z}_j$ is statistically close to $D_{\Lambda^\perp_{\vec{u}}(A),S}$. Overall, the data available to the adversary follows the same distributions as in Game_0, up to negligible statistical distance.
- When $\vec{e}_j \hookleftarrow U(\mathbb{Z}_q^m)$, the residual distribution of $\vec{z}_j$ is uniform (by adapting the argument above). The data available follows the same distributions as in Game_1, up to negligible statistical distance.

This completes the proof of the lemma.

Appendix D

Adaptive CCA Broadcast Encryption with Constant-Size Secret Keys and Ciphertexts

Journal of Information Security 2013
[PPSS13] with David Pointcheval, Siamak F. Shahandashti, and Mario Streflerr

Abstract : *We consider designing public-key broadcast encryption schemes with constant-size secret keys and ciphertexts, achieving chosen-ciphertext security. We first argue that known CPA-to-CCA transforms currently do not yield such schemes. We then propose a scheme, modifying a previous selective CPA secure proposal by Boneh, Gentry, and Waters. Our scheme has constant-size secret keys and ciphertexts and we prove that it is selective chosen-ciphertext secure based on standard assumptions. Our scheme has ciphertexts that are shorter than those of the previous CCA secure proposals. Then we propose a second scheme that provides the functionality of both broadcast encryption and revocation schemes simultaneously using the same set of parameters. Finally we show that it is possible to prove our first scheme adaptive chosen-ciphertext secure under reasonable extensions of the bilinear Diffie-Hellman exponent and the knowledge of exponent assumptions. We prove both of these extended assumptions in the generic group model. Hence, our scheme becomes the first to achieve constant-size secret keys and ciphertexts (both asymptotically optimal) and adaptive chosen-ciphertext security at the same time.*

D.1 Introduction

A *broadcast encryption* is a cryptographic scheme that enables encryption of broadcast content such that only a set of target users, selected at the time of encryption, can decrypt the content. Apparent applications include group communication, pay TV, content protection, file system access control, and geolocation.

A crucial aspect of any cryptographic scheme, which arguably decides its fate of being used in practice, is its efficiency. Since one of the most prominent applications of broadcast encryption is real-time broadcasting, ciphertext size is at the heart of efficiency measures for such schemes, and constructions with constant-size ciphertexts are desirable. Indeed, if one allows the ciphertext

size to grow linearly with the number of target users, construction of secure broadcast encryption becomes trivial. Other important measures of efficiency for broadcast encryption include the secret and public key sizes and the encryption and decryption times.

A broadcast encryption scheme can be *static* or *dynamic*, depending on if the system users need to be fixed once and for all at the setup stage or if it supports new users joining the system at an arbitrary time, incurring only incremental parameter changes. Evidently, dynamic schemes are more flexible and hence more desirable in practical applications.

An important security paradigm for broadcast encryption schemes is that of *adaptive* security. This paradigm captures the fact that an adversary might choose to compromise keys in the system adaptively, based on its acquired knowledge of the system parameters and previously compromised keys and ciphertexts. Such a definition is widely accepted as the proper notion of security for broadcast encryption schemes and there are schemes proposed in the literature that provably achieve security against adaptive adversaries.

On the other hand, security against chosen ciphertext attacks (CCA) is a fundamental notion of security for any encryption scheme, broadcast encryption included. Although there have been a number of proposed broadcast encryption schemes that are secure against chosen plaintext attacks (CPA), the CPA-to-CCA transformations in the literature do not seem to yield CCA secure broadcast encryption schemes with constant-size ciphertexts.

Adaptive and CCA security, and constant-size ciphertexts, have all three been separately achieved for broadcast encryption. However, there has not been any proposal that achieves all three simultaneously. In this paper, we propose a broadcast encryption with constant-size ciphertexts and prove it adaptive CCA secure under assumptions that are reasonable generalizations of previous assumptions in the literature.

The literature on broadcast encryption mainly considers two categories of such schemes and each work usually provides solutions that are efficient only for one of the two cases, depending on whether the content is broadcast to a very small or a very large proportion of registered users. The party who encrypts the content, hence either determines their intended set of target users or that of revoked users, respectively, as an input to the encryption algorithm. Consequently, the latter category of schemes are sometimes called *revocation* schemes.

Consider the pay-TV application in which the content of the broadcast consists of several TV channels. Normally, there are a number of basic channels that are usually bundled together and provided to most of the customers in different packages, and also there are a number of more specialized channels (e.g., pay-per-view) that are of the interest of a small proportion of customers. Hence we face a scenario in which both of the above categories of schemes are simultaneously needed to broadcast the content. Nevertheless, there has been no proposal in the literature that provides both functionalities efficiently, and hence the existing efficient solution to the above scenario is to set up two parallel schemes, each covering part of the broadcast content. In this paper, we propose a scheme that can handle both cases efficiently, providing a solution to the above scenario that does not require maintaining two parallel sets of system parameters.

D.1.1 Related Work

Broadcast encryption was first formalized by Fiat and Naor [FN93]. Their scheme is a private key scheme and proved secure against an upper bounded number of colluders. Fully collusion secure (private-key) broadcast encryption was first proposed in [NNL01], which introduced the subset cover framework that became the basis for many subsequent proposals, including [DF03] which proposed the first public key broadcast encryption.

Boneh, Gentry, and Waters [BGW05] were first to propose a fully collusion-resistant public key broadcast encryption in which the ciphertext size is constant. In all the previous schemes,

the size of the ciphertext is linear in the size of the target set. In this paper we limit our attention to such schemes. They proposed two schemes, respectively CPA and CCA secure, both in the selective model of security. Dynamic broadcast encryption was proposed in [DPP07] where they designed CPA secure schemes that were only partially adaptive secure. Strictly speaking, their scheme is a *revocation* scheme, in which the set of revoked users is selected at the time of encryption, and in turn, any user outside of the revoked set is able to decrypt. [Del07] proposed identity-based broadcast encryption and gave a selective CPA secure scheme.

Adaptive security was first proposed by [GW09] where they gave several schemes achieving adaptive CPA security, including two broadcast encryption schemes and two identity-based broadcast encryption (IBBE) schemes, one of each achieving constant-size ciphertexts in the random oracle model. The schemes proposed in [Wat09] and [LSW10], respectively a broadcast encryption scheme and a revocation scheme, are the only schemes secure under static assumptions (as opposed to the so called q-based ones). The latter work also proposes an identity-based revocation scheme which is proved selective CPA secure. Recently, the first adaptive CCA secure schemes were proposed by [PPS12a], although their schemes do not have constant-size ciphertexts.

D.1.2 Our Contributions

In this paper, we propose an efficient dynamic broadcast encryption scheme (called OurBE) and prove that it is selective CCA secure assuming the widely-used bilinear Diffie-Hellman exponent (BDHE) assumption and a universal one-way hash function (UOWHF). The scheme has constant-size ciphertexts (only two group elements), constant-size secret keys (only one group element), and a public key which grows linearly with the number of users in the system. We construct our scheme by modifying a selective CPA secure scheme (dubbed BGW_1 from now on) by Boneh, Gentry, and Waters [BGW05]. Our modification is minimal in the sense that our scheme has exactly the same ciphertext and secret key sizes as that of BGW_1, and is proved secure under the same assumption, plus the comparatively weak UOWHF assumption. The minor difference is that our scheme has one extra element in the linearly-growing public key. The only other CCA secure scheme with constant-size ciphertexts is a modified version of BGW_1 by the same authors (dubbed BGW_2 from now on), which has ciphertexts that are double the size of our scheme (i.e., four group elements vs. our two). BGW_2 is proved selective CCA secure under BDHE, plus the assumption that a signature scheme used in the construction is strongly unforgeable, which is an assumption of comparable strength as UOWHF.

We also propose an *inclusive-exclusive* broadcast encryption scheme which can act as both a broadcast encryption and a revocation scheme at the same time, as it allows the flexibility to specify either the target set or the revoked set at the time of encryption. The ciphertext and the secret key are still only two and one group elements, respectively, but we need to add one group element per user to the already linearly-growing public key which results in a public key which is 1.5 times that of BGW_1.

Next, we show that it is possible to prove OurBE adaptive CCA secure under generalized versions of existing assumptions. Particularly, we propose generalized versions of the BDHE and the knowledge-of-exponent (KEA) assumptions, and prove that both hold in the generic group model. We argue that both of these are intuitive and reasonable generalizations of accepted assumptions, and in turn, enable achieving the highest level of security with highly-efficient parameters. Namely, OurBE is provably adaptive CCA secure with constant-size ciphertexts and secret keys, and it is the first scheme to achieve such properties.

D.2 Preliminaries

In this section we review the notation we use, the BDHE and GBDHE assumptions, and the notions of security for dynamic broadcast encryption and universal one-way hash function.

Notation We use the following typefaces: Roman X for constants, italic X for variables, sans serif X for algorithms, and calligraphic $\mathcal{X}$ for oracles. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of order p, and $e : \mathbb{G} \times \mathbb{G} \mapsto \mathbb{G}_T$ be a bilinear map. Let g be a generator of $\mathbb{G}$ and $g_T = e(g, g)$.

D.2.1 Dynamic Broadcast Encapsulation

Broadcast encryption is conventionally formalized as *broadcast encapsulation* in which, instead of a ciphertext, a session key is produced, which is required to be indistinguishable from random. Such a scheme can provide public encryption functionality in combination with a symmetric encryption through the hybrid encryption (a.k.a. KEM-DEM) paradigm [CS03]. We hence use the terms encryption and encapsulation interchangeably.

Following [DPP07], we define a (public-key) *dynamic* broadcast encapsulation scheme as a tuple of four algorithms $\mathsf{BE} = (\mathbf{Setup}, \mathsf{Join}, \mathsf{Encaps}, \mathsf{Decaps})$ where:

- $\mathbf{Setup}(1^k)$ outputs $(\mathsf{msk}, \mathsf{ek})$ containing the master secret key and the (initial) encryption key;
- $\mathsf{Join}(\mathsf{msk}, i)$ outputs the key pair (sk_i, pk_i) for user i, and appends pk_i to ek;
- $\mathsf{Encaps}(\mathsf{ek}, S)$ for a set of users S outputs (H, K) containing a ciphertext (a.k.a. header) and a session key; and
- $\mathsf{Decaps}(\mathsf{ek}, sk_i, S, H)$ outputs K if $i \in S$ and $\perp$ otherwise.

Adaptive CCA security for BE is defined via the following experiments for $b \in \{0, 1\}$ between the challenger C and the adversary A:

1. *Setup:* C runs $\mathbf{Setup}(1^k)$ and gives ek to A;
2. *Query:* A arbitrarily issues the following oracle queries:
   - *join* oracle query $\mathcal{J}oin(i)$: C runs $\mathsf{Join}(\mathsf{msk}, i)$ and gives pk_i to A;
   - *corruption* oracle query $\mathcal{C}or(i)$: C gives sk_i to A;
   - *decapsulation* oracle query $\mathcal{D}ec(i, S, H)$: C runs $\mathsf{Decaps}(\mathsf{ek}, sk_i, S, H)$ and gives K to A;
3. *Challenge:* A outputs a set S^* on which it wants to be challenged; C runs $\mathsf{Encaps}(\mathsf{ek}, S^*)$ and gets (H^*, K^*), then sets $K = K^*$ if $b = 0$ or picks a random K if $b = 1$, and finally gives (H^*, K) to A;
4. *Query:* A issues further oracle queries as the previous query phase;
5. *Guess:* A outputs a guess b'. The experiment outputs 1 if $b' = b$ and there is no $i^* \in S^*$ for which there has been a $\mathcal{C}or(i^*)$ or $\mathcal{D}ec(i^*, S^*, H^*)$ query. The experiment outputs 0 otherwise.

For any adversary A, we define its *advantage* against BE in an adaptive CCA attack to be the difference between the probability that the above experiment for $b = 0$ outputs 1 and the probability that the experiment for $b = 1$ outputs 1. The scheme is said to be adaptive CCA secure if for any adversary A its advantage against BE in an adaptive CCA attack is negligible in k.

Selective security is defined via similar games with the difference that A commits to the set S^* before the setup phase. For CPA security, A does not get to query the decryption oracle. We sometimes use SCPA, SCCA, ACPA, and ACCA as shorthands referring to selective CPA, selective CCA, adaptive CPA, and adaptive CCA security.

Note that the above definition (which is based on that of [PPS11][1]) is stronger than that of [BGW05] since they require that the adversary does not make any decryption oracle query with $i \in S^*$ for which $H = H^*$, but we relax the constraint and only require no query with $i \in S^*$ for which $(S, H) = (S^*, H^*)$.

D.2.2 The BDHE and GBDHE Assumptions

Let us define the two sets of polynomials $P = (p_1, \ldots, p_s)$ and $Q = (q_1, \ldots, q_t)$, with $p_1 = q_1 = 1$, and a polynomial f, where $\forall i, k : p_i, q_k, f \in \mathbb{F}_p[X_1, \ldots, X_n]$. Let us also define $g^P = (g^{p_1}, \ldots, g^{p_s})$. We say that f is independent of (P, Q) if it cannot be written as $f = \sum_{i,j=1}^{s} a_{i,j} p_i p_j + \sum_{k=1}^{t} b_k q_k$ for constants $a_{i,j}$ and b_k.

The *generalized decision bilinear Diffie-Hellman exponent (GBDHE)* problem is defined in [BBG05] as follows: given the input $g^{P(x_1,\ldots,x_n)}$ and $g_{\mathrm{T}}^{Q(x_1,\ldots,x_n)}$ for random choices of $x_1, \ldots, x_n \in \mathbb{F}_p$, decide between $g_{\mathrm{T}}^{f(x_1,\ldots,x_n)}$ and a random $T \in \mathbb{G}_{\mathrm{T}}$. The GBDHE assumption says that it is hard to solve the GBDHE problem if f is independent of (P, Q).

The *decision bilinear Diffie-Hellman exponent* assumption (parameterized by n and denoted by n-BDHE), which is an instance of the GBDHE assumption, says that given the input $g, h, \{g^{\alpha^k}\}_{k \in \{1,\ldots,2n\} \setminus \{n+1\}}$ for random $h \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, it is hard to decide between $e(g, h)^{\alpha^{n+1}}$ and a random $T \in \mathbb{G}_{\mathrm{T}}$.

D.2.3 Universal One-Way Hash Function

Consider a keyed hash function H. H is called a universal one-way hash function (UOWHF)[2] if there is no efficient adversary winning the following security game. First, the adversary chooses a message and outputs it. Then, the challenger chooses a random key for H and gives it to the adversary. Finally, the adversary outputs a second message and terminates. The adversary wins if the two messages are different, but their hashes under the chosen key are the same. This notion was first proposed in [NY89], and is shown to be strictly weaker than collision resistance [Sim98, RS04]. In fact, one-way functions are shown to be sufficient for UOWHF [Rom90], whereas collision resistant hash functions are only known to be constructed from claw-free permutations [Dam87] or lattice-based assumptions [GGH96].

D.3 CCA from Generic Transforms?

In this section we consider the two types of general standard model CPA-to-CCA transforms, namely NY-like and CHK-like, and argue that applying these transforms to the proposed broadcast encryption schemes in the literature does not give us CCA security and constant-size ciphertexts.

[1] Note that, in comparison with [PPS11], we ignore the *Reg* parameter here as it can be regarded as part of ek.
[2] UOWHF is also known as target collision resistance (TCR).

NY-like Transforms The Naor-Yung paradigm ([NY90] and [Sah99, DDN00, Lin03]) provides a construction for CCA secure encryption from CPA secure encryption along with non-interactive zero-knowledge proofs. To apply any NY-like transform to a broadcast encryption, one needs to make a NIZK proof of a statement containing the session key K. Such proofs tend to be long and inefficient. Furthermore, all the proposed schemes that have a constant-size ciphertext are pairing-based, and in all these schemes the session key is a member of the target set $\mathbb{G}_{\mathrm{T}}$, but NIZK proofs of statements containing members of $\mathbb{G}_{\mathrm{T}}$ are not known. In particular, Groth-Sahai constructions [GS08] only provide witness indistinguishable proofs for such statements, whereas zero knowledge, and in particular the ability to simulate proofs without knowing a witness, seems to be essential to the security proofs of NY-like constructions.

CHK-like Transforms The Canetti-Halevi-Katz paradigm ([CHK04] and [BK05, Kil06]) provides a construction for CCA secure encryption from CPA secure *identity-based* encryption and an extra authenticating primitive such as signature or message authentication code (MAC). Essential to the paradigm is that any encryption to an identity can be decrypted by the secret key generated for the same identity. However, in the broadcast encryption case, encryptions are made to a set and decryptions are possible by the secret key of any member of that set. Hence, such transforms are not readily applicable to identity-based broadcast encryptions.

D.4 An Efficient Selective CCA Broadcast Encryption

Let $\mathsf{H}_\kappa : \mathbb{G} \mapsto \mathbb{Z}_p$ be a hash family indexed by κ. We define a broadcast encryption scheme OurBE in the following. We describe the system for (at most) $n-1$ users to be notationally consistent with the original scheme of [BGW05], on which the system is based. The system for n users can be defined accordingly.

- $\mathbf{Setup}(1^k, n-1)$ picks a random generator $g \in \mathbb{G}$, two random quantities $\alpha, \gamma \in \mathbb{Z}_p$, and a random index κ for hash function H, computes $v = g^\gamma$, and outputs $\mathsf{msk} = (\alpha, \gamma)$ and $\mathsf{ek} = (g, v, \kappa)$.

- $\mathsf{Join}(\mathsf{msk}, i)$ computes $g_k = g^{(\alpha^k)}$ for $k = i, i+1, n+1-i$, and $n+1+i$, and $d_i = g_i^\gamma$, and outputs $sk_i = d_i$ and $pk_i = (g_i, g_{i+1}, g_{n+1-i}, g_{n+1+i})$. The secret key sk_i is given to the user, and ek is updated by appending pk_i.

- $\mathsf{Encaps}(\mathsf{ek}, S)$ picks a random element $t \in \mathbb{Z}_p$ and sets $K = e(g_{n+1}, g)^t$, which can be equivalently computed as $K = e(g_{n+1-i}, g_i)^t$ for any i, computes H as follows, and outputs (H, K).
$$H = \langle g^t, (v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \prod_{j \in S} g_{n+1-j})^t \rangle.$$

- $\mathsf{Decaps}(\mathsf{ek}, sk_i, S, H)$ parses the header as $H = (C_0, C_1)$, checks if the following equation holds:
$$e(C_1, g) = e(v \cdot g_1^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j \in S} g_{n+1-j}, C_0), \tag{D.1}$$
and if it does, then calculates the session key as follows:
$$K = \frac{e(C_1, g_i)}{e(d_i \cdot g_{1+i}^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j \in S \setminus \{i\}} g_{n+1-j+i},\ C_0)}.$$

In the following we bring a theorem which states that if the hash function H is a universal one-way hash function, then the proposed scheme satisfies selective CCA security under the same assumption as that of the original scheme, namely n-BDHE. Intuitively, the main modification we make in (the encryption algorithm of) the original scheme is the introduction of $g_1^{\mathsf{H}_\kappa(g^t)}$. If this element is not present, as it is in the original scheme, given a header $H = (C_0, C_1)$ corresponding to a key K, one can compute the header (C_0^r, C_1^r) that corresponds to the key K^r, and hence the scheme is malleable. We show that a UOWHF is sufficient to eradicate malleability and get CCA security. This modification is inspired by a similar technique in [BMW05] which, in contrast, was shown to be applicable to an *identity-based* scheme. Here we show that a similar idea is applicable to $\mathsf{BGW_1}$. The proof of the following theorem can be found in Appendix D.8.1. In the proof we use the structure of the keys in the scheme to simulate decryption queries.

Theorem D.4.1 The above scheme is selective CCA secure if the n-BDHE decision problem is hard and H is a universal one-way hash function.

On Dynamicity Note that the bound on the number of users in OurBE does not prevent the system from being able to handle more than $n-1$ users. That is, as long as the system "jumps over" the users number n and $n+1$ (i.e., after user number $n-1$, the next user is numbered $n+2$), the system can handle polynomially many users more than $n-1$ and remains secure. The security of the scheme with more than $n-1$ users can be proved based on the following assumption: given the input h, and $\{g_k = g^{\alpha^k}\}$ for $k \in \{n+1-m, \ldots, n+1+m\} \setminus \{n+1\}$ for random $g, h \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, it is hard to decide between $e(g_{n+1}, h)$ and a random $T \in \mathbb{G}_{\mathrm{T}}$. It is not hard to see that this assumption is equivalent to the following assumption: given the input g, h, and $\{g_k = g^{\alpha^k}\}$ for $k \in \{1, \ldots, 2m\} \setminus \{m\}$ for random $h \in \mathbb{G}$ and $\alpha \in \mathbb{Z}_p$, it is hard to decide between $e(g_m, h)$ and a random $T \in \mathbb{G}_{\mathrm{T}}$. Here $m \geq n+2$ is the last user number to join. This assumption is comparable to the m-BDHE assumption. In fact, like the BDHE assumption, it is an instance of the GBDHE assumption. In view of this observation, OurBE is a *dynamic* broadcast encryption in the sense that: (1) the system setup and the ciphertext size are independent of the upper bound on the number of users; (2) a new user can join anytime without incurring modification of other user secret keys; and (3) the encryption key is incrementally updated by an operation of O(1) complexity.

Comparison The only broadcast encryption scheme in the literature that provides CCA security with constant-size ciphertexts is $\mathsf{BGW_2}$. It has similar secret and public key sizes as our scheme. However, there are differences in terms of security assumptions and ciphertext size. $\mathsf{BGW_2}$ uses a signature or a message authentication code (MAC) and is proved secure under n-BDHE plus the strong unforgeability (SUF) of the signature or the MAC, whereas OurBE needs n-BDHE plus a universal one-way hash function (UOWHF). In theory, SUF and UOWHF are equivalent (both are equivalent to one-wayness), but in practice, hash functions are generally much more efficient than signatures. In terms of ciphertext size, $\mathsf{BGW_2}$ has a ciphertext whose size is (about) double that of $\mathsf{BGW_1}$'s ciphertext: a $\mathsf{BGW_2}$ ciphertext consists of a $\mathsf{BGW_1}$ ciphertext of two $\mathbb{G}$ elements, plus an element in $\mathbb{Z}_p$ and a signature (or a MAC tag). OurBE has the same ciphertext size as that of $\mathsf{BGW_1}$, i.e., only two $\mathbb{G}$ elements. We summarize this comparison in Table D.1. For simplification, we show the total number of elements without the details of the groups to which each element belongs. Note that although pk_i in OurBE includes four group elements, since there are some repeating values the final ek includes the three initial values plus only $2n-1$ extra values of g_i.

Table D.1: Comparison of CCA secure schemes with constant-size ciphertexts

Scheme	$\lvert sk_i\rvert$ / $\lvert\mathsf{ek}\rvert$	$\lvert H\rvert$	Security	Assumption
[BGW05]	1 / 2n+1	4	SCCA	n-BDHE, SUF
OurBE	1 / 2n+2	2	SCCA	n-BDHE, UOWHF

$\lvert\cdot\rvert$: size in number of elements n: number of users plus one.

D.5 Inclusive-Exclusive Broadcast Encryption

In this section we show that OurBE can be slightly modified to provide both the broadcast encryption and the revocation functionality simultaneously; that is, we propose a scheme in which the encrypter may choose to determine either a target set S or a revoked set R of users at the time of encryption, without the need to set up two parallel systems. The decryption naturally goes ahead only if the user is either in S or not in R. In the following we (ab)use the notation "S/R" to indicate "either S or R" as input to the encapsulation and decapsulation algorithms. In practice this can be implemented using the first bit of the input to indicate the inclusive or exclusive mode of operation.

- **Setup**$(1^k, n-1)$ picks random $g \in \mathbb{G}$, $\alpha, \gamma \in \mathbb{Z}_p$, and κ for H, computes $v = g^\gamma$, sets $\pi_0 = g^{\alpha(\alpha^n-1)/(\alpha-1)}$, and outputs $\mathsf{msk} = (\alpha, \gamma)$ and $\mathsf{ek} = (g, v, \pi_0, \kappa)$.
- $\mathsf{Join}(\mathsf{msk}, i)$ computes g_i, g_{i+1}, g_{n+1-i}, g_{n+1+i}, and $d_i = g_i^\gamma$, sets $\pi_i = \pi_0^{\alpha^i}/g_{n+1}$, and outputs $sk_i = d_i$ and $pk_i = (g_i, g_{i+1}, g_{n+1-i}, g_{n+1+i}, \pi_i)$. Now, sk_i is given to the user, and ek is updated by appending pk_i.
- $\mathsf{Encaps}(\mathsf{ek}, S/R)$ picks a random $t \in \mathbb{Z}_p$ and sets $K = e(g_{n+1}, g)^t$, computes H as either of the following accordingly, and outputs (H, K).

$$H = \begin{cases} \langle g^t, (v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \prod_{j\in S} g_{n+1-j})^t \rangle & \text{if } S \text{ given} \\ \langle g^t, (v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \pi_0 / \prod_{j\in R} g_{n+1-j})^t \rangle & \text{if } R \text{ given} \end{cases}$$

- $\mathsf{Decaps}(\mathsf{ek}, sk_i, S/R, H)$ parses $H = (C_0, C_1)$, checks if the either of the following equation accordingly holds:

$$e(C_1, g) = \begin{cases} e(v \cdot g_1^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j\in S} g_{n+1-j}, C_0) & \text{if } S \text{ given} \\ e(v \cdot g_1^{\mathsf{H}_\kappa(C_0)} \cdot \pi_0 / \prod_{j\in R} g_{n+1-j}, C_0) & \text{if } R \text{ given} \end{cases}$$

and if it does, then calculates the session key accordingly as follows:

$$K = \frac{e(C_1, g_i)}{e(d_i \cdot g_{1+i}^{\mathsf{H}_\kappa(C_0)} \cdot \prod_{j\in S\setminus\{i\}} g_{n+1-j+i},\ C_0)}, \quad \text{or}$$

$$K = \frac{e(C_1, g_i)}{e(d_i \cdot g_{1+i}^{\mathsf{H}_\kappa(C_0)} \cdot \pi_i / \prod_{j\in R} g_{n+1-j+i},\ C_0)}.$$

Correctness Let $N = \{1, \ldots, n-1\}$. We have

$$\pi_0 = \prod_{j \in N} g_{n+1-j} \quad \text{and} \quad \pi_i = \prod_{j \in N \setminus \{i\}} g_{n+1-j+i}.$$

Hence in the exclusive mode, for any $i \notin R$ we have:

$$\pi_0 / \prod_{j \in R} g_{n+1-j} = \prod_{j \in N \setminus R} g_{n+1-j} \quad \text{and}$$

$$\pi_i / \prod_{j \in R} g_{n+1-j+i} = \prod_{j \in (N \setminus R) \setminus \{i\}} g_{n+1-j+i}.$$

Hence, if $i \notin R$, the session key the user i calculates in the exclusive mode is effectively the same as the session key it would have calculated if it were decrypting a ciphertext encrypted to $S = N \setminus R$ in the inclusive mode, and therefore the scheme is correct.

Note that the parameters are set in a way that the scheme properly excludes users that join after the time of encryption from inclusive-mode ciphertexts, and includes such users in the exclusive-mode ciphertexts. Unfortunately, the system appears to lose full dynamicity.

Efficiency The scheme enjoys similar desirable efficiency measures as the inclusive-only scheme; that is, the ciphertext and the user secret key sizes are both constant and the public key size is linear in the number of users.

Security A similar security definition to that of broadcast encryption can be defined for such schemes, with the difference that the adversary is now allowed to ask decryption oracle queries for both modes. Naturally exclusive-mode decryption oracle queries $\mathcal{D}ec(i^*, N \setminus S^*, H^*)$ for $i^* \in S^*$ are also not allowed. It is not hard to see that the security of OurBE translates into the above scheme satisfying this security definition.

D.6 Achieving Adaptive CCA Security

Since we have a very efficient scheme with asymptotically optimal size secret keys and ciphertexts which is already proved selective CCA secure based on standard assumptions, in this section we try to see how further we can achieve in terms of security by considering reasonable generalizations of some standard assumptions, while retaining the same optimally efficient secret key and ciphertext sizes. We first propose reasonable generalizations of the GBDHE and prove that they hold in the generic group model; then we prove that OurBE can be proved ACCA secure under these assumptions; and finally we compare our scheme to existing adaptive or CCA secure broadcast encryptions.

D.6.1 The OBDHE Assumption

We consider extending the GBDHE problem assuming that an extra resource is also given: the *Diffie-Hellman computation oracle* $\mathcal{O}^{\mathrm{DH}}_{g,e}$, that takes two inputs $u, v \in \mathbb{G}$ and outputs $w \in \mathbb{G}$ such that $e(u, v) = e(g, w)$. Let us call this the *Oracle BDHE* problem, or OBDHE for short. Formally, we define:

The OBDHE Problem: Given the input $g^{P(x_1,\ldots,x_n)}$ and $g_{\mathrm{T}}^{Q(x_1,\ldots,x_n)}$ for random choices of $x_1, \ldots, x_n \in \mathbb{F}_p$, and access to the $\mathcal{O}^{\mathrm{DH}}_{g,e}$ oracle, decide between $g_{\mathrm{T}}^{f(x_1,\ldots,x_n)}$ and a random $T \in \mathbb{G}_{\mathrm{T}}$.

Note that the GBDHE assumption implies that the only elements (dependent on $x_1, \ldots, x_n$ and) in $\mathbb{G}$ that can be computed are those in the form $g^{\sum a_i p_i}$. Thus, for any $\mathcal{O}^{\mathrm{DH}}_{g,e}$ query

(dependent on $x_1, \ldots, x_n$) we can assume $u = g^{\sigma_u}$ and $v = g^{\sigma_v}$, where σ_u and σ_v are polynomials. Then we will have $w = \mathcal{O}^{\mathrm{DH}}_{g,e}(u, v) = g^{\sigma_u \sigma_v}$. Hence, by providing access to $\mathcal{O}^{\mathrm{DH}}_{g,e}$, basically a number of "free multiplications" in the exponent are given. Let us define $p' = \sigma_u \sigma_v$. If we consider q' queries to $\mathcal{O}^{\mathrm{DH}}_{g,e}$, and the output to the i-th query represented as $w_i = g^{p'_i}$, we can define $P' = (p'_1, \ldots, p'_{q'})$. Our extension of the GBDHE assumption says that it is still hard to solve the GBDHE problem if these "free multiplications" in the exponent do not help breaking the independence property. Formally, letting $\|$ denote concatenation, we define:

Assumption D.6.1 [OBDHE] It is hard to solve the decision (P, Q, f)-OBDHE problem if f is independent of $(P \parallel P', Q)$.

In Appendix D.8.2 we prove that the assumption holds in the generic group model [Sho97, BBG05]. We prove an upper bound on the success of any generic algorithm trying to solve the OBDHE problem which is negligible if p, the order of $\mathbb{F}_p$ is super-polynomial. Leaving technicalities to the appendix, we prove the following theorem:

Theorem D.6.2 The OBDHE Assumption holds in the generic group model.

In fact, our proof is similar to that of [BBG05], suggesting that our assumption is a natural and closely-related extension of GBDHE. It is also worth to note that OBDHE is falsifiable by simply solving the corresponding $(P \parallel P', Q, f)$-GBDHE problem efficiently.

D.6.2 The GKEA Assumption

We propose the generalized knowledge of exponent assumption (GKEA) as follows and prove that it holds in the generic group model.

In the following we use p to denote a polynomial (suppressing the random variables) and $p(x_1, \ldots, x_n)$ to denote the evaluation of p on the input $(x_1, \ldots, x_n)$. Let the tuple $P = (p_1, \ldots, p_s)$ be in $\mathbb{F}_p[X_1, \ldots, X_n]^s$. Let the linear span of P, denoted by $\mathrm{Span}(P)$, be defined as the vector space containing all the polynomials in the form $\sum_{k=1}^{s} a_k p_k$.

Assumption D.6.3 [GKEA] Let the tuple $P = (p_1, \ldots, p_s)$ be in $\mathbb{F}_p[X_1, \ldots, X_n]^s$, where $p_1 = 1$. Let A be an algorithm that given $g^{P(x_1,\ldots,x_n)}$ for a random $(x_1, \ldots, x_n)$, outputs

$$(\ (a_k)_{k=1}^{s}, \ h, \ h^{q(x_1,\ldots,x_n)} \ ), \qquad \text{such that}$$
$$q(x_1, \ldots, x_n) = \sum_{k=1}^{s} a_k p_k(x_1, \ldots, x_n).$$

Consider the subspace of $\mathrm{Span}(P)$ defined as $V_q = \{r \mid r, rq \in \mathrm{Span}(P)\}$ and let $\{r_i\}_{i=1}^{t}$ be a basis for V_q. Then, there exists an extractor E that given the same input as A outputs

$$(b_i)_{i=1}^{t}, \qquad \text{such that} \quad \mathrm{dlog}_g(h) = \sum_{i=1}^{t} b_i r_i(x_1, \ldots, x_n).$$

This assumption basically says that the only way an adversary can produce pairs of the form (h, h^q) is to pick given pairs of the form (h_i, h_i^q) and output $(\prod h_i^{b_i}, \prod (h_i^q)^{b_i})$ for some known values of b_i.

For $P = (1, X)$ and $q(X) = X$, this becomes the original KEA of [Dam91], which basically says that given (g, g^x) the only way an adversary can produce pairs of the form (h, h^x) is to output $(g^b, (g^x)^b)$ for some known value of b. This assumption is referred to KEA1 in [HT98, BP04] and as Diffie-Hellman Knowledge (DHK) in [Den06]. A similar problem is formalized as strong Diffie-Hellman (SDH) in [ABR01].

Table D.2: Comparison of adaptive or CCA secure broadcast encryption schemes

Scheme	O($\|sk_i\|$)	O($\|H\|$)	Security	Assumption
[DF02] BE	$\log n$	$r \log \frac{n}{r}$	ACCA1	(IBE)
BE	$\log^{1+\epsilon} n$	$\frac{r}{\epsilon}$	ACCA1	(HIBE)
[BGW05] BE	**1**	**1**	SCCA	n-BDHE, SUF
[GW09] BE	**1**	**1**	ACPA	n-BDHES, PRF, ROM
BE	**1**	s	ACPA	n-BDHES, PRF
IBBE	**1**	**1**	ACPA	n-BDHES, PRF, ROM
IBBE	**1**	$\sqrt{s}$	ACPA	n-BDHES, PRF
[Wat09] BE	n	**1**	ACPA	dBDH, dLin
[LSW10] R	**1**	r	ACPA	dBDH, dLin
[PPS12a] BE	**1**	$r \log \frac{n}{r}$	**ACCA**	DDH
BE	**1**	r	**ACCA**	DDH
OurBE BE	**1**	**1**	SCCA	n-BDHE, UOWHF
			ACCA	n-OBDHE, GKEA, UOWHF

O($|\cdot|$): order of size, n, s, r: number of total, targeted, revoked users.

For $P = (1, X, Y, YX)$ and $q(X, Y) = X$, this becomes the KEA3 assumption of [BP04], which basically says that given (g, g^x, f, f^x) the only way an adversary can produce pairs of the form (h, h^x) is to output $(g^b f^c, (g^x)^b (f^x)^c)$ for some known values of b and c. This assumption is referred to as Extended KEA (XKEA) in [AF07] and as Extended Diffie-Hellman Knowledge (EDHK) in [DP08].

The above two instances of the assumption have already been proved to hold in the generic group model [Den06, AF07, DP08]. In the following we propose a theorem stating the generic assumption and prove it in Appendix D.8.3.

Theorem D.6.4 The GKEA Assumption holds in the generic group model.

D.6.3 Adaptive CCA Security

In this section we prove OurBE adaptive CCA secure under our generalized versions of the BDHE and knowledge of exponent assumptions. To prove adaptive CCA security, we basically show that a decryption query by the adversary that contains a valid ciphertext does not increase the (cryptographic) 'knowledge' of the adversary. Also note that since ciphertext validity is publicly verifiable, a decryption query that contains an invalid ciphertext does not increase the adversary's knowledge either. Hence we basically show that a CCA attack against the system is equivalent to a CPA attack, under the GKEA assumption and the hash function being a UOWHF. Furthermore, the access to $\mathcal{O}^{\mathrm{DH}}_{g,e}$ enables answering adaptive corruption queries.

Formally, we prove adaptive CCA security assuming that the OBDHE and the GKEA assumptions hold and H is a UOWHF. Intuitively, selective CPA security stems from the BDHE assumption underlying the OBDHE assumption along with the hash function being a UOWHF; the Diffie-Hellman oracle enables adaptive security; and the CCA security is achieved from the

GKEA assumption along with the hash function being a UOWHF. The following theorem is proved in Appendix D.8.4.

Theorem D.6.5 OurBE is adaptive CCA secure if the OBDHE and the GKEA assumptions hold and H is a universal one-way hash function.

We note that we prove CCA security based on the GKEA assumption, an assumption which is much weaker than the generic model itself (and instances of it are shown to be falsifiable [BP04]), and in fact, proving the equivalence of CPA and CCA security is trivial if the generic group model is used directly, since on a decryption query with a first element g^t, we may assume that t is known.

D.6.4 Comparison

Since our scheme is the first to achieve adaptive CCA security with constant-size ciphertexts, we compare our scheme with those from the literature that are adaptive CPA or selective CCA secure. We do not consider schemes that are not fully collusion resistant. The schemes in the literature with constant-size ciphertexts include a selective CCA secure scheme from [BGW05], and three adaptive CPA secure schemes from [GW09] and [Wat09]. The schemes in the literature that do not have constant-size ciphertexts include adaptive CPA secure schemes from [DF02], [GW09] (identity-based) and [LSW10] (revocation scheme), and recent adaptive CCA secure schemes from [PPS12a]. Table D.2 summarizes our comparison. We consider plain and identity-based (IB) broadcast encryption (BE) and revocation (R) schemes. Among these, schemes from [DF02] and [PPS12a] are generic schemes based on (hierarchical) identity-based encryption ((H)IBE), and encryption schemes (implemented under DDH), respectively. Since (H)IBE can be based on various assumption, we simply use it in parentheses in the table. All other schemes are explicit proposals based on various bilinear Diffie-Hellman assumptions, in some cases plus extra assumptions such as strong unforgeability (SUF) of signatures, pseudo-random functions (PRF), and the random oracle model (ROM). To accommodate more information, we omit the O notation and write $\mathrm{O}(f(n,s,r))$ as $f(n,s,r)$. Comparatively more desirable quantities are highlighted in boldface.

D.7 Concluding Remarks

We proposed a very efficient broadcast encryption scheme. The sizes of the secret keys and ciphertexts in the scheme are asymptotically optimal, i.e., constant. We showed that the scheme can be proved selective CCA secure assuming BDHE and a universal one-way hash function. Furthermore, we showed that proving adaptive CCA security is possible if we consider extended versions of the GBDHE and knowledge of exponent assumptions. Considering only the standard assumptions, our scheme provides shorter ciphertexts than the only other known CCA secure scheme. Considering the extended assumptions, our scheme is the first scheme to achieve constant size secret keys and ciphertexts and adaptive CCA security at the same time. The problem of designing schemes that achieve such properties under standard assumptions remains open.

Acknowledgments

This work was supported by the French ANR-09-VERS-016 BEST Project. The authors would like to thank the anonymous reviewers of the ACISP 2012 conference and the International Journal of Information Security.

D.8 Appendix

D.8.1 Proof of Theorem D.4.1

Proof: Suppose there exist a selective CCA adversary A that is able to distinguish the above scheme's keys from random elements. We construct an algorithm B that either outputs a collision for a given key κ or solves the n-BDHE decision problem.

Let B be given an n-BDHE challenge $(g, h, \{g_i\}_{i\in\{1,\ldots,2n\}\setminus\{n+1\}}, T)$ and has to decide whether $T = e(g_{n+1}, h)$ or it is random. B runs A and receives a set S^* of honest users on which it wishes to be challenged. As a UOWHF adversary, B also gives out h as the first message on which it wishes to be challenged and receives a key κ for the hash function. B chooses a random $\beta \in \mathbb{Z}_p$, calculates v as follows, and gives $\mathsf{ek} = (g, v, \kappa)$ to A.

$$v = g^{\beta} \cdot g_1^{-\mathsf{H}_\kappa(h)} \cdot \prod_{j\in S^*} g_{n+1-j}^{-1}. \tag{D.2}$$

On any join query for user i made by the adversary, B gives $pk_i = (g_i, g_{n+1-i}, g_{n+1+i})$ to A.

On any private key query for user i made by A (note that $i \notin S^*$), B calculates the private key as follows and gives it to A.

$$d_i = g_i^{\beta} \cdot g_{1+i}^{-\mathsf{H}_\kappa(h)} \cdot \prod_{j\in S^*} g_{n+1-j+i}^{-1}.$$

Note that d_i is properly simulated since we have $d_i = v^{\alpha^i}$.

On a decryption query $(i, S, (C_0, C_1))$ by A (note that $S \subset S^*$ and $i \in S$), B first checks the validity of the ciphertext using Equation D.1. If the ciphertext is valid then it checks whether $\mathsf{H}_\kappa(h) = \mathsf{H}_\kappa(C_0)$ which in case of validity provides a collision for the hash function H_κ and hence B can output C_0 as the second message and break the UOWHF property.

If Equation D.1 holds and $\mathsf{H}_\kappa(h) \neq \mathsf{H}_\kappa(C_0)$, then let $\delta = \mathsf{H}_\kappa(C_0) - \mathsf{H}_\kappa(h)$. B calculates the key as follows:

$$K = \frac{e(C_1, g \cdot g_n^{1/\delta})}{e(g^{\beta} \cdot g_n^{\beta/\delta} \cdot g_1^{\delta} \cdot \prod_{j\in S^*\setminus S} (g_{n+1-j} \cdot g_{2n+1-j}^{1/\delta})^{-1}, C_0)}.$$

Now since Equation D.1 holds, the ciphertext is in the form

$$(g^t, (v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \prod_{j\in S} g_{n+1-j})^t)$$

for some (unknown) t. Hence, the above calculated K will be as follows:

$$K = \frac{e((v \cdot g_1^{\mathsf{H}_\kappa(g^t)} \cdot \prod_{j\in S} g_{n+1-j})^t, g \cdot g_n^{1/\delta})}{e(g^{\beta} \cdot g_n^{\beta/\delta} \cdot g_1^{\delta} \cdot \prod_{j\in S^*\setminus S} (g_{n+1-j} \cdot g_{2n+1-j}^{1/\delta})^{-1}, g^t)}$$

$$= \left(\frac{e(g^{\beta} \cdot g_1^{\delta} \cdot \prod_{j\in S^*\setminus S} g_{n+1-j}^{-1}, g \cdot g_n^{1/\delta})}{e(g^{\beta} \cdot g_n^{\beta/\delta} \cdot g_1^{\delta} \cdot \prod_{j\in S^*\setminus S} (g_{n+1-j} \cdot g_{2n+1-j}^{1/\delta})^{-1}, g)} \right)^t$$

$$= \left(\frac{e(g^{\beta} \cdot g_1^{\delta} \cdot \prod_{j \in S^* \setminus S} g_{n+1-j}^{-1}, g^{1+\alpha^n/\delta})}{e(g^{\beta(1+\alpha^n/\delta)} \cdot g_1^{\delta} \cdot \prod_{j \in S^* \setminus S} g_{n+1-j}^{-(1+\alpha^n/\delta)}, g)} \right)^t$$

$$= \left(\frac{e(g_1^{\delta}, g^{1+\alpha^n/\delta})}{e(g_1^{\delta}, g)} \right)^t = e(g_1, g)^{\alpha^n t} = e(g_{n+1}, g)^t$$

and hence it is properly simulated. In the above, we have substituted v from Equation D.2 and used the fact that $\forall k : g_{n+k} = g_k^{\alpha^n}$.

At some point, A declares that it is ready to receive the challenge. B calculates the challenge ciphertext as $C = (h, h^{\beta})$ and gives C along with $K = T$ to A. First, note that from Equation D.2 we have

$$v \cdot g_1^{\mathsf{H}_\kappa(h)} \cdot \prod_{j \in S^*} g_{n+1-j} = g^{\beta},$$

and hence C is a valid ciphertext satisfying Equation D.1. Furthermore, assuming that $h = g^t$ for some t, we have

$$h^{\beta} = (g^{\beta})^t = (v \cdot g_1^{\mathsf{H}_\kappa(h)} \cdot \prod_{j \in S^*} g_{n+1-j})^t,$$

which means that if $T = e(g_{n+1}, h) = e(g_{n+1}, g)^t$, then K is the key corresponding to the ciphertext C, and if T is random, then K is a random key.

In the second phase of the attack, B answers A's queries as in the first phase.

At the end, A outputs its guess b. B outputs b as its decision for the n-BDHE challenge. Based on the above discussion, if A is successful in its CCA attack, then either B is able to compute a collision for H_κ and win the UOWHF game, or it is able to solve the n-BDHE decision problem successfully. ∎

D.8.2 Proof of the OBDHE Assumption

In this section, we prove Theorem D.6.2. Let d_P, $d_{P'}$, d_Q, and d_f be respectively the maximum degrees of the polynomials in P, P', Q, and f. We prove the following upper bound in the generic bilinear group model. We consider two random encodings $\xi, \zeta : \mathbb{Z}_p^+ \mapsto \{0,1\}^m$ and write $\mathbb{G} = \{\xi(x) | x \in \mathbb{Z}_p^+\}$ and $\mathbb{G}_{\mathrm{T}} = \{\zeta(x) | x \in \mathbb{Z}_p^+\}$. The following theorem is a sufficient condition for Theorem D.6.2.

Theorem D.8.1 For P, Q, P' f, ξ, ζ, $\mathbb{G}$, $\mathbb{G}_{\mathrm{T}}$ defined above, let $|P| = s$, $|Q| = t$, and $\ell = s + t$. Let $d = \max(2d_P, d_Q, d_f)$. If f is independent of $(P \parallel P', Q)$, then for any A making a total of at most q queries to the oracles computing the group operations and the bilinear pairing, and at most q' queries to the $\mathcal{O}_{g,e}^{\mathrm{DH}}$ oracle, we have:

$$\left| \Pr \left[\mathsf{A} \left(\begin{array}{c} p, \xi(P(x_1, \ldots, x_n)), \\ \zeta(Q(x_1, \ldots, x_n)), \\ \zeta(t_0), \zeta(t_1); \mathcal{O}_{g,e}^{\mathrm{DH}}(\cdot, \cdot) \end{array} \right) = b : \begin{array}{c} x_1, \ldots, x_n, y \xleftarrow{\mathrm{R}} \mathbb{F}_p, \\ b \xleftarrow{\mathrm{R}} \{0,1\}, \\ t_b \leftarrow f(x_1, \ldots, x_n), \\ t_{1-b} \leftarrow y \end{array} \right] - \tfrac{1}{2} \right|$$
$$\leq \frac{(q + q' + \ell + 2)^2 \cdot \max(2d_{P'}, d)}{2p}$$

Proof: Assume that we are given the algorithm A. Consider an algorithm B that interacts with A as follows. B maintains two lists of pairs:

$$L = \{(p_i, \xi_i) : i = 1, \ldots, \tau_0\} \quad \text{and} \quad L_{\mathrm{T}} = \{(q_i, \zeta_i) : i = 1, \ldots, \tau_1\},$$

such that at step τ of its interaction with A: $\tau_0 + \tau_1 = \tau + \ell + 2$. Here, $p_i \in \mathbb{F}_p[X_1, \ldots, X_n]$, $q_i \in \mathbb{F}_p[X_1, \ldots, X_n, Y_0, Y_1]$, and $\xi_i, \zeta_i \in \{0,1\}^m$.

B also maintains a counter τ', initialized at zero, to count the number of $\mathcal{O}^{\mathrm{DH}}_{g,e}$ oracle queries, and a list of polynomials:

$$P' = \{p'_i : i = 1, \ldots, \tau'\}$$

to store the polynomial output of the $\mathcal{O}^{\mathrm{DH}}_{g,e}$ oracle queries.

At step $\tau = 0$, B initializes the lists by setting $p_1, \ldots, p_s$ in L equal to the polynomials in P, $q_1, \ldots, q_t$ in L_{T} equal to the polynomials in Q, $q_{t+1} = Y_0$, and $q_{t+2} = Y_1$. It also chooses $\ell + 2$ random strings in $\{0,1\}^m$ and initializes $\{\xi_i\}_{i=1}^{s}$ and $\{\zeta_i\}_{i=1}^{t+2}$.

B then runs A under the input p, $\{\xi_i\}_{i=1}^{s}$, $\{\zeta_i\}_{i=1}^{t}$, ζ_{t+1}, and ζ_{t+2}. B answers A's oracle queries as follows. We are assuming that A's queries can only be strings obtained from B since B can, by increasing m, make the strings in $\mathbb{G}$ and $\mathbb{G}_{\mathrm{T}}$ arbitrarily hard to guess.

Group operations: For a $\mathbb{G}$ operation query (ξ_i, ξ_j), B calculates $p_{\tau_0+1} \leftarrow p_i \pm p_j$ depending on whether multiplication or division is requested. If $p_{\tau_0+1} = p_l$ for some $l \leq \tau_0$, then B sets $\xi_{\tau_0+1} \leftarrow \xi_l$; otherwise it sets ξ_{τ_0+1} equal to a new random string different from all the previous ξ_i. Then it appends the new pair $(p_{\tau_0+1}, \xi_{\tau_0+1})$ to L, replies to A's query with ξ_{τ_0+1}, and finally increments the counter τ_0. $\mathbb{G}_{\mathrm{T}}$ operation queries are dealt with analogously by updating the list L_{T} and counter τ_1.

Bilinear pairings: For a pairing query of the form (ξ_i, ξ_j), B calculates $q_{\tau_1+1} \leftarrow p_i \cdot p_j$. If $q_{\tau_1+1} = q_l$ for some $l \leq \tau_1$, then B sets $\zeta_{\tau_1+1} \leftarrow \zeta_l$; otherwise it sets ζ_{τ_1+1} equal to a new random string different from all the previous ζ_i. Then it appends the new pair $(q_{\tau_1+1}, \zeta_{\tau_1+1})$ to L_{T}, replies to A's query with ζ_{τ_1+1}, and finally increments the counter τ_1.

$\mathcal{O}^{\mathrm{DH}}_{g,e}$ queries: For a $\mathcal{O}^{\mathrm{DH}}_{g,e}$ query (ξ_i, ξ_j), B calculates $p_{\tau_0+1} \leftarrow p_i \cdot p_j$. If $p_{\tau_0+1} = p_l$ for some $l \leq \tau_0$, then B sets $\xi_{\tau_0+1} \leftarrow \xi_l$; otherwise it sets ξ_{τ_0+1} equal to a new random string different from all the previous ξ_i. B also sets $p'_{\tau'+1} \leftarrow p_{\tau_0+1}$, appends $p'_{\tau'+1}$ to P', and increments the counter τ'. Then it appends the new pair $(p_{\tau_0+1}, \xi_{\tau_0+1})$ to L, replies to A's query with ξ_{τ_0+1}, and finally increments the counter τ_0.

A terminates after at most $q + q'$ queries and returns a guess b'.

Now B chooses $x_1, \ldots, x_n, y \xleftarrow{\mathrm{R}} \mathbb{F}_p$ and $b \xleftarrow{\mathrm{R}} \{0,1\}$, and sets $y_b \leftarrow f(x_1, \ldots, x_n)$ and $y_{1-b} \leftarrow y$. Setting $X_i = x_i$ for all $i = 1, \ldots, n$, $Y_0 = y_0$, and $Y_1 = y_1$, we see that B's interaction provides a perfect simulation for A as long as the chosen random values for the random variables do not result in any equality of the values of the intermediate different polynomials. In other words, the simulation is perfect unless for some i and j we have one of the following:

1. $p_i(x_1, \ldots, x_n) = p_j(x_1, \ldots, x_n)$, yet the polynomials p_i and p_j are not equal, or
2. $q_i(x_1, \ldots, x_n, y_0, y_1) = q_j(x_1, \ldots, x_n, y_0, y_1)$, yet the polynomials q_i and q_j are not equal.

Let Fail be the event that one of the above conditions holds. We bound the probability of this event.

First, if we set $Y_b = f(X_1, \ldots, X_n)$, this does not raise the probability that Fail happens. This is because the above substitution does not create any new equalities between polynomials q_i and q_j. In general, $q_i - q_j$ is in the form

$$\sum_{k=1}^{s}\sum_{l=1}^{s} a_{k,l}p_k p_l + \sum_{k=1}^{s}\sum_{l=1}^{q'} a'_{k,l}p_k p'_l + \sum_{k=1}^{q'}\sum_{l=1}^{q'} a''_{k,l}p'_k p'_l + \sum_{u=1}^{t} b_u q_u + cY_0 + dY_1.$$

Let us define

$$P^* = P \parallel P' = (p^*_1, \ldots, p^*_{s+q'}) = (p_1, \ldots, p_s, p'_1, \ldots, p'_{q'}).$$

Now we can write $q_i - q_j$ in the form

$$\sum_{k=1}^{s+q'}\sum_{l=1}^{s+q'} a_{k,l}p^*_k p^*_l + \sum_{u=1}^{t} b_u q_u + cY_0 + dY_1.$$

Hence assuming that the substitution $Y_b = f(X_1, \ldots, X_n)$, does create a new equality, then $q_i - q_j$, which is in the above form, is a non-zero polynomial, yet setting $Y_b = f(X_1, \ldots, X_n)$ makes it zero. Thus, f must be dependent on $(P \parallel P', Q)$, which is a contradiction.

Now that we made the substitution $Y_b = f(X_1, \ldots, X_n)$, our polynomials are only in $X_1, \ldots, X_n$, and Y_{1-b}. The maximum degree of any polynomial in the form $p_i - p_j$ or $q_i - q_j$ is $\max(2d_P, 2d_{P'}, d_Q, d_f) = \max(2d_{P'}, d)$. Hence, for each pair (i, j), the probability that a random assignment of the random variables is a root of one of the above polynomials is at most $\max(2d_{P'}, d)/p$. Since there are at most $2\binom{q+q'+\ell+2}{2}$ pairs of (p_i, p_j) and (q_i, q_j) in total, we have

$$\begin{aligned}\Pr[\text{Fail}] &\leq \binom{q+q'+\ell+2}{2}\frac{2\max(2d_{P'}, d)}{p} \\ &\leq \frac{(q+q'+\ell+2)^2\max(2d_{P'}, d)}{p}.\end{aligned}$$

Now we would like to bound A's success probability, i.e., $|\Pr[b = b'] - \frac{1}{2}|$. We know that

$$\Pr[b = b'] = \Pr[b = b'|\text{Fail}] \cdot \Pr[\text{Fail}] + \Pr[b = b'|\neg\text{Fail}] \cdot \Pr[\neg\text{Fail}].$$

If Fail does not happen, then B's simulation is perfect. In this case, since b is chosen after the simulation ends, $\Pr[b = b'|\neg\text{Fail}] = \frac{1}{2}$. Substituting this and $\Pr[\neg\text{Fail}] = 1 - \Pr[\text{Fail}]$ in the above equation, we get the following after rearrangement:

$$\Pr[b = b'] - \frac{1}{2} = (\Pr[b = b'|\text{Fail}] - \frac{1}{2}) \cdot \Pr[\text{Fail}].$$

Hence we have

$$|\Pr[b = b'] - \frac{1}{2}| = |\Pr[b = b'|\text{Fail}] - \frac{1}{2}| \cdot \Pr[\text{Fail}] \leq \frac{1}{2}\Pr[\text{Fail}],$$

which gives us the claimed bound and finishes the proof. ∎

D.8.3 Proof of Theorem D.6.4

Proof: Let d_P be the maximum degree of the polynomials in P. We consider a random encoding $\xi : \mathbb{Z}_p^+ \mapsto \{0,1\}^m$ and write $\mathbb{G} = \{\xi(x) | x \in \mathbb{Z}_p^+\}$.

Given an algorithm A we construct the extractor E as follows. E maintains a list L of pairs (p_i, ξ_i), initialized with pairs containing the elements of P and random strings, respectively as the first and second elements.

E runs A on input $(\xi_i)_{i=1}^s$. Any group operation query (ξ_i, ξ_j) is responded by computing $p_i + p_j$ and checking if the resulting polynomial already exists in the list. If it does, E returns the corresponding encoding, and if not, it chooses a new random string as the encoding to be returned, and adds $p_i + p_j$ and the encoding to the list L.

When A terminates and returns (ξ_i, ξ_j) as its output, E finds the corresponding polynomial pair (p_i, p_j). If $p_j \neq p_i q$, E outputs $\perp$. Otherwise, let $\{r_i\}_{i=1}^t$ be defined as above. E decomposes p_i as a linear combination of $\{r_i\}_{i=1}^t$, that is, it finds coefficients $(b_i)_{i=1}^t$ such that $p_i = \sum_{i=1}^t b_i r_i$, and outputs $(b_i)_{i=1}^t$.

Assume that A asks σ queries. E's list contains $s + \sigma$ pairs at the end of the execution of A. All the polynomials in this list are in $\mathrm{Span}(P)$. Since both p_i and p_j are in $\mathrm{Span}(P)$, if $p_j = p_i q$, then $p_i \in V_q$, and hence p_i can be written as a linear combination of $\{r_i\}_{i=1}^t$. Furthermore, the discrete logarithm of A's first input ξ_i is equal to $p_i(x_1, \ldots, x_n)$, which in turn equals $\sum_{i=1}^t b_i r_i(x_1, \ldots, x_n)$. Therefore, E succeeds if its simulation of A's environment is perfect and $p_j = p_i q$.

Note that if A's environment is simulated perfectly, then it outputs a pair for which we have $p_j(x_1, \ldots, x_n) = p_i(x_1, \ldots, x_n) q(x_1, \ldots, x_n)$, but not necessarily $p_j = p_i q$.

Let Fail be the event that E fails. Based on the above discussion, E fails if either it fails to simulate A's environment perfectly or if $p_j \neq p_i q$ but $p_j(x_1, \ldots, x_n) = p_i(x_1, \ldots, x_n) q(x_1, \ldots, x_n)$. E's simulation of the environment for A is perfect unless a set of random values $(x_1, \ldots, x_n)$ result in some equality of the values of the different polynomials in L. Hence, if we add $p_i q$ as the polynomial number $s + \sigma + 1$ to the list L, E's overall probability of failure is bounded by the probability that a set of random values $(x_1, \ldots, x_n)$ result in some equality of the values of the different polynomials in the augmented list of $s + \sigma + 1$ polynomials. Hence we have:

$$\Pr[\text{Fail}] \leq \binom{s+\sigma+1}{2} \frac{d_P}{p} \leq \frac{(s+\sigma+1)^2 d_P}{p},$$

and the proof is complete. ∎

The above proof is in the plain generic group model. It is easy to extend the proof to the bilinear generic group model. Furhtermore, one can see that the proof still works (with some natural modifications) in the model where the adversary is allowed to query the oracles on any encoding, rather than only those it has received before (either as input or as responses to previous oracle queries).

Another point to note is that, in the bilinear group model, any input to the adversary in the target group can be disregarded and hence does not change the assumption.

D.8.4 Proof of Theorem D.6.5

Proof: We make our proof in two stages.

Stage 1: First, we prove that if H is a UOWHF, then the following specific assumption is an instance of the OBDHE assumption as per our definition in Section D.6.1: let $\mathcal{O}^{\mathrm{DH}}_{g,e}$ be an oracle that given (x_1, x_2) outputs y s.t. $e(x_1, x_2) = e(g, y)$. Given the following quantities:

$$g,\ h,\ \{g_k = g^{\alpha^k}\}_{k\in\{1,\ldots,2n\}\setminus\{n+1\}},\ v,$$

and oracle access to $\mathcal{O}^{\mathrm{DH}}_{g,e}$, it is hard to distinguish $e(g^{\alpha^{n+1}}, h)$ from a random value if the queries to $\mathcal{O}^{\mathrm{DH}}_{g,e}$ are restricted to the following, where $C \cap S = \varnothing$:

$$\begin{array}{l}(1)\ |C| \text{ queries } \{\mathcal{O}^{\mathrm{DH}}_{g,e}(g_k, v)\}_{k\in C}, \text{ and}\\ (2) \text{ one query } \mathcal{O}^{\mathrm{DH}}_{g,e}(w, h), \text{ where } w = v g_1^{\mathsf{H}_\kappa(h)} \prod_{j\in S} g_{n+1-j}.\end{array}$$

Consider the hash function $\mathsf{H}_\kappa : \mathbb{G} \mapsto \mathbb{Z}_p$ and define the function $\mu(h) = h^{\mathsf{H}_\kappa(h)}$. In the generic group model, the input to H_κ is an encoding representing h, which is considered to be an encoding that may be chosen *independently* of h. Therefore, we may assume $\mathsf{H}_\kappa(h)$ independent of h. Of course this is true only if the sole way to calculate $\mu(h)$ is through computing $\mathsf{H}_\kappa(h)$ first and then raising h to the power of the hash output. Otherwise, if $\mu(h)$ cannot be computed through group operations, without computing $\mathsf{H}_\kappa(h)$ separately, then the encoding of h cannot be chosen independently of h. For a "good" hash function we may assume that $\mu(h)$ cannot be computed through group operations, without computing $\mathsf{H}_\kappa(h)$ separately.

To be more precise, consider Theorem D.8.1 and its presented proof in Appendix D.8.2. Assume that P also includes an extra element which is a multiplication of a polynomial and the function $\eta(y) = \mathsf{H}_\kappa(g^y)$. Now, if the encoding of $h = g^y$ is chosen independently of h, the proof will still work, i.e., $\Pr[\textsc{Fail}]$ can be shown to be upper-bounded by a negligible bound, unless for some considerable portion of possible y's we have $\rho_1(y)\eta^2(y) + \rho_2(y)\eta(y) + \rho_3(y) = 0$, where ρ_1, ρ_2, and ρ_3 are polynomials of degree at most $\max(2d_{P'}, d)$. This condition implies that $\eta(y)$ can be calculated for some considerable portion of possible y's by solving the above equation.

Formally, let us define a *δ-good* hash family as follows: We say a hash family $\mathsf{H}_\kappa : \mathbb{G} \mapsto \mathbb{Z}_p$ indexed by κ is *δ-good* if for a random κ there does not exist polynomials ρ_1, ρ_2, and ρ_3 of degree at most δ such that for a non-negligible portion of possible y's we have: $\rho_1(y)\ \mathsf{H}^2_\kappa(g^y) + \rho_2(y)\ \mathsf{H}_\kappa(g^y) + \rho_3(y) = 0$. Now since $\max(2d_{P'}, d) = 4n$, we conclude that if H is at least $4n$-good, then its output can be considered independent of the encoding of its input, and hence we may treat it as a constant.

Now assume that for a given random κ and Y, we wish to find a pre-image X, such that $\mathsf{H}_\kappa(X) = Y$. Assume $X = g^x$. If H is not a δ-good hash family, for a random κ there exist polynomials ρ_1, ρ_2, and ρ_3 of degree at most δ such that with a non-negligible probability: $\rho_1(x)\ Y^2 + \rho_2(x)\ Y + \rho_3(x) = 0$. This is a polynomial of order at most δ, and its roots can be found in time which is polynomial in δ and $\log p$ [Ber70, Sho90]. For each root x, one can check whether $\mathsf{H}_\kappa(g^x) = Y$ and find the pre-image X with at most δ checks. Hence, if H is not a δ-good hash family, then it is not a pre-image resistant (a.k.a. one-way) hash function. Since UOWHF implies pre-image resistance, we have the following lemma:

Lemma D.8.2 Let $\mathsf{H}_\kappa : \mathbb{G} \mapsto \mathbb{Z}_p$ be hash function for which p is super-polynomial in k. If H is a universal one-way hash function, then it is δ-good (as per our definition above) for all δ polynomial in k.

Hence, if H is a UOWHF, then the following claim proves that the specific assumption above is an OBDHE assumption as per our definition in Section D.6.1, in which the output of H is

treated as a constant. Note that alternatively one may make the stronger assumption that H is modeled as a non-programmable random oracle [BR93, Nie02]. Also note that since the system is defined for $n-1$ users, S and C are subsets of $\{1,\ldots,n-1\}$.

Claim D.8.3 For the following polynomials and $S, C \subseteq \{1,\ldots,n-1\}$, and for any constant c, f is independent of $(P \parallel P', Q)$ if $C \cap S = \varnothing$.

$$P = (\ 1,\ y,\ \{x^k\}_{k\in\{1,\ldots,2n\}\setminus\{n+1\}},\ z,\ \eta,\ zy + \mathrm{c}xy + y\sum_{j\in S} x^{n+1-j}\ ),$$
$$P' = \{zx^i\}_{i\in C}, \qquad Q = (1), \qquad \text{and} \qquad f = yx^{n+1}.$$

Proof: We have at most one multiplication of polynomials at our disposal. Let us define

$$P_x = \{x^k\}_{k\in\{1,\ldots,2n\}\setminus\{n+1\}}, \quad P_{zx} = \{zx^i\}_{i\in C}, \quad \text{and}$$
$$P_{yx} = P_{yz} = zy + \mathrm{c}xy + y\sum_{j\in S} x^{n+1-j}.$$

To make $f = yx^{n+1}$, since there is a y factor, one of our multiplicands needs to be either y or P_{yx}. Choosing y will not help because we do not have an x^{n+1} to make f, so one of our multiplicands is definitely P_{yx}. The only choice for a second multiplicand that can give us f is one from P_x. Multiplying these terms gives us terms of the form $zyx^i + \mathrm{c}yx^{i+1} + y\sum_{j\in S} x^{n+1-j+i}$, which includes yx^{n+1} if $i \in S$, but then we have to be able to produce the term zyx^i for some $i \in S$ to be able to cancel it out.

To get zyx^i, using only two multiplicands, we have the following four possibilities:

- use y and zx^i to get yzx^i for some $i \in C$, but since $C \cap S = \varnothing$ we can not get yzx^i for any $i \in S$.
- use x^i and P_{yz} again, but this cancels out our desired term yx^{n+1} as well since we have to use the same i.
- use z and P_{yx} to get $z^2y + \mathrm{c}xyz + zy\sum_{j\in S} x^{n+1-j}$, which includes zyx^i if $n+1-i \in S$ or if $i = 1$, but then, in either case, we have to cancel z^2y and the only way to get z^2y is to use the same terms again which cancels our desired term zyx^i as well.
- use P_{zx} and P_{yx} to get $z^2x^ky + \mathrm{c}x^{k+1}yz + zy\sum_{j\in S} x^{n+1-j+k}$, which includes zyx^i if $n+1-i+k \in S$ or if $k+1 = i$, but then, in either case, we have to cancel z^2x^ky and the only way to get z^2x^ky is to use the same terms again with the same k which cancels our desired term zyx^i as well.

Hence f is independent of $(P \parallel P', Q)$ and the proof of Claim D.8.3 is complete. ▌

Stage 2: Now that we have proved our specific assumption is an OBDHE assumption, we prove that under this assumption, the GKEA assumption, and the UOWHF assumption OurBE is adaptive CCA secure.

Let A be an adaptive CCA adversary for OurBE. We construct an adversary B that successfully breaks our specific assumption, if A is successful in its attack against OurBE, the GKEA assumption holds, and H is a UOWHF.

First of all, note that, based on Lemma D.8.2 and a discussion similar to that of Stage 1, as long as H_κ is a UOWHF, it can be indistinguishably simulated *independently* of its input in the generic group model, and hence hashed values can be considered *constant* for this proof. Jumping ahead, we treat $\mathrm{c} = \mathsf{H}_\kappa(C_0)$ and $\mathrm{c}^* = \mathsf{H}_\kappa(C_0^*)$ as constant coefficients for polynomials.

Let B be given the following quantities:

$$g,\ h,\ \{g_k = g^{\alpha^k}\}_{k\in\{1,\ldots,2n\}\setminus\{n+1\}},\ v,\ T,$$

and (restricted) oracle access to $\mathcal{O}^{\mathrm{DH}}_{g,e}$ as specified by the assumption. It is supposed to distinguish whether $T = e(g^{\alpha^{n+1}}, h)$ or T is random. As a UOWHF adversary, B gives out h as the first message on which it wishes to be challenged and receives a key κ for the hash function. B runs A on input $\mathsf{ek} = (g, v, \kappa)$.

On a join query for user i made by the adversary, B gives $pk_i = (g_i, g_{n+1-i}, g_{n+1+i})$ to A.

On any private key query for user i made by A, B queries the oracle $\mathcal{O}^{\mathrm{DH}}_{g,e}(g_i, v)$ and and gives the oracle output to A. Note that if we assume $v = g^\gamma$, then the oracle output is equal to g_i^γ.

On a decryption oracle query (i, S, H), where $H = (C_0, C_1)$, B first checks the ciphertext validity. If the ciphertext is invalid it replies with $\perp$. Let $\mathrm{c} = \mathsf{H}_\kappa(C_0)$. If the ciphertext is valid, then it is in the following form:

$$H = (C_0, C_0^q), \qquad \text{where} \quad q = \gamma + \mathrm{c}\alpha + \sum_{j\in S} \alpha^{n+1-j}. \tag{D.3}$$

Let us assume, without loss of generality, that all the potential $n-1$ users are initiated. Let C denote the set of corrupted users by A and $N^* = \{1,\ldots,2n\}\setminus\{n+1\}$. Now A can be viewed as an algorithm that on input g, v, κ, $\{g_i\}_{i\in N^*}$, and $\{d_i\}_{i\in C}$ outputs $H = (C_0, C_0^q)$ as above. Note that the input to A (excluding $\kappa \notin \mathbb{G}$) can be written as follows:

$$g^P, \qquad \text{where} \quad P = (\ 1,\ \gamma,\ \{\alpha^i\}_{i\in N^*},\ \{\gamma\alpha^i\}_{i\in C}\ ).$$

To apply the GKEA assumption, note that here $\mathrm{Span}(P)$ includes all the elements of the following form:

$$\rho = u + x\gamma + \sum_{i\in N^*} y_i\alpha^i + \gamma\sum_{i\in C} z_i\alpha^i, \qquad \text{for random } u, x, y_i, z_i. \tag{D.4}$$

Consider ρq for some ρ and the q defined above, respectively in Equations D.3 and D.4. For ρq to be in $\mathrm{Span}(P)$, we should have $x = 0$ and $\forall i \in C : z_i = 0$ because otherwise ρq will have either the factor $x\gamma^2$ or $z_i\gamma^2\alpha^i$ for some i and would not fall in $\mathrm{Span}(P)$. Hence any ρ satisfying $\rho q \in \mathrm{Span}(P)$ should be in the form

$$\rho = u + \sum_{i\in N^*} y_i\alpha^i, \qquad \text{for random } u, y_i. \tag{D.5}$$

A basis for such a subspace is the set $\{1, \{\alpha^i\}_{i\in N^*}\}$. Therefore the GKEA assumption guarantees that there exists an extractor that outputs the values $\{\beta, \{b_i\}_{i\in N^*}\}$ such that

$$C_0 = g^{\beta + \sum_{i\in N^*} b_i\alpha^i} = g^\beta \prod_{i\in N^*} g_i^{b_i}.$$

Now note that $K = e(g_{n+1}, C_0)$. Hence the session key can be calculated based on the known representation of C_0 in terms of g and g_i, e.g., as follows:

$$K = e(g_{n+1}, g^{\beta} \prod_{i \in N^*} g_i^{b_i}) = e(g_{n+1}, g)^{\beta} \prod_{i \in N^*} e(g_{n+1}, g_i)^{b_i}$$
$$= e(g_n, g_1)^{\beta} e(g_{2n}, g_1)^{b_n} e(g_{n+2}, g_{2n-1})^{b_{2n}} \prod_{i \in N^* \setminus \{n, 2n\}} e(g_n, g_{i+1})^{b_i}.$$

At some point, the adversary A terminates the first query phase and outputs a set S^* on which it wants to be challenged. B calculates $w = v g_1^{\mathsf{H}_\kappa(h)} \prod_{j \in S^*} g_{n+1-j}$, makes the oracle query $\mathcal{O}^{\mathrm{DH}}_{g,e}(w, h)$, receives the oracle output h', sets the challenge ciphertext as $H^* = (C_0^*, C_1^*) = (h, h')$, and gives H^* along with $K = T$ to A. Let $\mathrm{c}^* = \mathsf{H}_\kappa(C_0^*)$. Note that, Equation D.1 (see page 148) holds, hence C is a valid ciphertext, and C_1^* should be equal to C_0^* raised to a power of the following form:

$$\gamma + \mathrm{c}^* \alpha + \sum_{j \in S^*} \alpha^{n+1-j}.$$

Furthermore, if $T = e(g_{n+1}, h)$, then K is the correct key corresponding to the ciphertext H^*, and if T is random, then K is a random key.

In the second phase of the attack, B answers A's join and corruption oracle queries as in the first phase, and A's decryption oracle queries, in a fashion similar to that of prior to the challenge, as follows.

On a decryption oracle query (i, S, H), where $H = (C_0, C_1)$, B first checks its validity, and if valid, it is in the form of Equation D.3.

Now the input to A can be listed as g, v, κ, $\{g_i\}_{i \in N^*}$, and $\{d_i\}_{i \in C}$, plus $H^* = (C_0^*, C_1^*)$. Let $C_0^* = g^{t^*}$. The input to A can be written as g^P, where P is as follows (κ, K_0, and K_1 can be disregarded as they are not in $\mathbb{G}$):

$$P = (\ 1,\ \gamma,\ \{\alpha^i\}_{i \in N^*},\ \{\gamma \alpha^i\}_{i \in C},\ t^*,\ t^*(\gamma + \mathrm{c}^* \alpha + \sum_{j \in S^*} \alpha^{n+1-j})\ ).$$

$\mathrm{Span}(P)$ includes all the linear combinations ρ of the above terms. Similarly, we argue that ρ cannot include any γ or $\gamma \alpha^i$ terms because they would induce γ^2 or $\gamma^2 \alpha^i$ terms respectively in the product ρq. Furthermore, ρ cannot include the last term because it would induce a non-cancelable $t^* \gamma^2$ term in the product ρq. In addition, note that if ρ includes the term t^*, then ρq would include the term

$$t^*(\gamma + \mathrm{c} \alpha + \sum_{j \in S} \alpha^{n+1-j}).$$

The only way a ρ including this term can be contained in $\mathrm{Span}(P)$ is if $\mathrm{c}^* = \mathrm{c}$ (i.e., $\mathsf{H}_\kappa(C_0^*) = \mathsf{H}_\kappa(C_0)$) and $S = S^*$ (note that $j \leq n-1$, so $n+1-j \geq 2$), which contradicts H being a UOWHF. Therefore, ρ cannot include the term t^*, and again ρ should be in the form of Equation D.5, and hence the session key can be calculated similarly as before.

At the end, A outputs its guess b. B outputs b as its decision for its received challenge. Based on the above discussion, if A is successful in its adaptive CCA attack, then B is able to either contradict H being a UOWHF or distinguish $T = e(g_{n+1}, h)$ from a random element successfully. Hence the proof of Theorem D.6.5 is complete. ∎

Appendix E

Message Tracing with Optimal Ciphertext Rate

Latincrypt '12
[PPS12b] with David Pointcheval, and Mario Strefler

Abstract : *Traitor tracing is an important tool to discourage defrauders from illegally broadcasting multimedia content. However, the main techniques consist in tracing the traitors from the pirate decoders they built from the secret keys of dishonest registered users: with either a black-box or a white-box tracing procedure on the pirate decoder, one hopes to trace back one of the traitors who registered in the system. But new techniques for pirates consist either in sending the ephemeral decryption keys to the decoders for real-time decryption, or in making the full content available on the web for later viewing. This way, the pirate does not send any personal information. In order to be able to trace the traitors, one should embed some information, or watermarks, in the multimedia content itself to make it specific to the registered users.*
This paper addresses this problem of tracing traitors from the decoded multimedia content or rebroadcasted keys, without increasing too much the bandwidth requirements. More precisely, we construct a message-traceable encryption scheme that has an optimal ciphertext rate, i. e. the ratio of global ciphertext length over message length is arbitrarily close to one.

E.0.5 Introduction

Traitor tracing (TT) [CFN94b] is a cryptographic primitive used to broadcast content only to a set of authorized users, with an additional tracing property. The two main goals of such a primitive are

- confidentiality: only the registered users should have access to the broadcast content;
- traceability: if registered users share their secrets to allow unregistered users to access the content, one should be able to trace back at least some of these traitors.

The former property is guaranteed by an encryption procedure, so that only registered users can decrypt and access the content. But an encryption scheme does not prevent users from giving away their secret keys. Even in case several users combine their secret keys in order to make a decryption box (a "pirate decoder"), it should be possible to identify one of the traitors from

the code/secrets in the decoder (white-box tracing) or by simply interacting with the decoder (black-box tracing). The *tracing* property should indeed guarantee that even if several users collude to construct a pirate decoder, at least one of the traitors could be found. It should also guarantee non-frameability: an honest user should not be wrongly declared as a traitor.

To circumvent tracing, pirates might try not to make the decoder available, which excludes both white-box and black-box tracing. Instead, they could make only the decrypted content available, or, in case a hybrid encryption scheme is used, the symmetric keys used to encrypt the content: by *message tracing*, we aim at tracing traitors from this information only, the decoded content.

Message tracing. Fiat and Tassa were the first to consider message tracing; in [FT99], they developed *dynamic traitor tracing* to deal with pirates that rebroadcast decrypted content. They assume that there is a real-time feedback from the broadcast content to the center, so that the watermarks can be adapted to the feedback. Safavi-Naini and Wang [SNW03a] noted that in this setting, dynamic TT can be prevented by delaying the rebroadcasting of the content. To take this counter-measure into account, they proposed *sequential traitor tracing*, where the mark allocation is precomputed, but users are removed according to the feedback received. They construct a sequential TT scheme by combining error-correcting codes and watermarking. Jin and Lotspiech [JL07] claimed that protection should not increase the bandwidth by more than 10 %. To solve this problem, they proposed to extend the tracing procedure over several movies (using "inner" and "outer" codes) and assumed that the pirates will not drop any block. Their sequence key block scheme permits the revocation of users after they have been traced through the rebroadcasted messages. Kiayias and Pehlivanoglu [KP09] showed that the sequence key block scheme allows only to trace and to revoke a limited number of users, and proposed a message-trace-and-revoke scheme without this limitation.

Optimal ciphertext rate. Contrary to the classical tracing where schemes with optimal ciphertext rate exist, the problem of constructing a scheme with optimal cipheretxt size for message tracing is still open. We explain why the solutions for classical tracing fails when applied to message tracing and we then describe our approach.

Boneh and Franklin [BF99b] developed a traitor tracing scheme with a ciphertext size linear in the maximal number of colluding users. Kiayias and Yung [KY02c] further integrated a version of this scheme for two users with a collusion-secure code into the first TT scheme with a constant ciphertext rate. This method can be summarized as follows. The sender essentially encrypts all the blocks twice, so that the recipient can only decrypt one of the two ciphertexts for each block. The tracing procedure consists in using the decrypted ciphertext or the distributed keys to extract a word associated to the pirate decoder. Granted the tracing capability of a collusion-secure code, one can then trace back one of the traitors. Kiayias and Yung's scheme leads to a ciphertext three times bigger than the initial content. Fazio, Nicolosi, and Phan [FNP07b] then achieved a ciphertext rate asymptotically 1. Their method is to encrypt just one particular block twice each time and then apply an all-or-nothing transform (AONT), which guarantees that the pirate cannot drop this particular block because missing just one block makes the pirate unable to get any information on the plaintext. The use of AONT in [KY02c, FNP07b] is interesting but quite impractical because the receiver should wait until he has received n blocks (where n is the code length of the code in use, and thus quite large) to start the decryption procedure. We note that, without aiming to optimize the ciphertext rate, the use of AONT can be avoided by using robust collusion-secure code which allows pirate to drop a fraction of the positions. This is used in [Sir07a, BP08, BN08b] to reduce the ciphertext size. However, in order to get optimal ciphertext rate in [FNP07b], the use of AONT is compulsory, otherwise the pirate could simply

drop the particular block to defeat the tracing procedure.

Focusing now on message tracing, one natural question is why we do not simply apply the above method of optimizing the ciphertext rate. We argue that this method cannot work for message tracing. We first notice that in all the above methods for classical tracing, each user finally gets the same plaintext and if a user redistributes this plaintext, we have no way to trace back the traitor from the distributed message. Therefore, the necessary condition for message tracing is that each user receives a different (marked) version of the plaintext. However, when the plaintext is different for each user, one cannot apply AONT for a whole fixed plaintext, otherwise all but at most one user can decrypt. The use of AONT for message tracing is thus irrelevant. Fortunately, we can still use the method of doubling one particular block by finding out a way to hide this block. Our method consists in using a 2-user anonymous broadcast encryption scheme and then randomly permuting the blocks. With a 2-user anonymous broadcast encryption scheme, the pirate cannot detect any difference between an encryption for both users (which is used for all blocks but the particular block) and an encryption for one of the two users that is used for the particular protected block. Combining with the permutation of the blocks, we can show that the pirate is prevented from detecting the particular protected block. Moreover, beyond the optimization of ciphertext rate, by not using AONT, our scheme also enjoys the property of the sequential decryption via the use of robust collusion secure code as in [BP08, BN08b]: the user can sequentially decrypt the sub-ciphertexts, and does not need to wait to have received the whole ciphertext and to apply the AONT transform to start the decryption procedure.

Our Contribution. Our goal is to improve the technique which consists in distributing two versions of each message block, but without doubling all the blocks. The naive way, presented in section E.2.1, indeed consists, for each message block m_i, to have two equivalent blocks m_i^0 and m_i^1, so that any sequence $\{m_i^{w_i}\}$, whatever $w \in \{0,1\}^n$ is, corresponds to a valid content m. The two versions m_i^0 and m_i^1 can be provided by either adding watermarks to the original message block m_i, or directly when recording with different angles or distances of the shots [BS98]. The blocks, m_i^0 or m_i^1, are both sent over the public channel. However, the user secret keys, usk_i^0 or usk_i^1, have been distributed to the users according to codewords in a traceable code. This means when the authority sees the decoded message m' or the symmetric keys, from each block m_i', it can tell whether it is m_i^0, m_i^1, or the block has been dropped, and then learns which decryption key has been used: usk_i^0, usk_i^1, or none. From this, it can derive one bit of a word: 0, 1, or 'erasure' respectively. Granted the collusion-resistance of the code with erasures, if not too many traitors colluded, at least one of them can be traced back. We can make the number of erasures as low as possible. The naive way thus consists in encrypting each pair of blocks with two keys. Each user owns only one of the two according to the codeword he received from a traceable code. This results in a ciphertext twice the length of the original message, plus the cost for two key encapsulations per block.

To reduce the length of the encrypted payload, the only way is to protect only a few blocks, not all of them (section E.2.2). However, if an adversary can detect which blocks are protected, he can drop some of them without impacting too much the quality of the original message (i. e. a few seconds from a movie). If he knows which blocks are protected, he will simply drop them after decryption, meaning that the output contains no information about the keys that were used. We thus propose a way to achieve this partial protection so that the adversary cannot detect which part is protected or not: even if we protect 1% of the blocks and the adversary drops 20% of the blocks, it will basically drop 20% of the protected blocks only, and not all of them.

A second improvement, presented in section E.2.3 can take advantage of some public-key encryption schemes where we can reuse the randomness in both key encapsulations, as one can

do with ElGamal: given g^r and two different public keys $X_0 = g^{x_0}$ and $X_1 = g^{x_1}$, one can derive two sessions keys $Y_0 = X_0^r = g^{rx_0}$ and $Y_1 = X_1^r = g^{rx_1}$. Knowing either x_0 or x_1 only, a user can extract one session key only and thus either m_i^0 or m_i^1 only. It does not cost two key encapsulations per block, but one only.

This scheme still suffers from long user keys. In section E.3, we use anonymous BE as a primitive to achieve shorter key lengths. We first focus on the two-user case (one message block). A message block either consists of a unique message m_i (not protected) or of two versions m_i^0 and m_i^1: in the former case, m_i should be encrypted for the two users, whereas in the latter case, m_i^0 has to be encrypted for user 0, and m_i^1 for user 1. To this aim, we use a 2-user anonymous broadcast encryption scheme (2ABE). Anonymous broadcast encryption allows the selection of any subset of the user set that should be able to decrypt the ciphertext, while hiding who is able to decrypt [LPQ12]. Suppose we have a 2ABE scheme, and we consider ℓ blocks $(m_1, \ldots, m_\ell)$, among which the k-th block only is protected and thus is provided as a pair (m_k^0, m_k^1). We encrypt all the unique blocks m_i for both users, whereas we encrypt m_k^0 for user 0, and m_k^1 for user 1. The ciphertexts are thereafter randomly permuted (but we assume that the message blocks contain indices to reorder them). User 0 and user 1 will both be able to decrypt ℓ ciphertexts among the $\ell + 1$, and after reordering will be able to get the original message. Due to the anonymity, they have no idea which block the other user cannot decrypt, therefore they have no idea which block is protected.

The encrypted payload is only $(1 + 1/\ell)$-times as long as the original message, plus the cost of 2ABE key encapsulations. Viewing the decrypted message, the authority can extract one bit. To achieve full message tracing, we need to allocate the user secret keys for the 2ABE using a collusion-secure code [BS98] or a even robust collusion-secure code [BKM10] if we consider dropped blocks, and thus erasures.

Organization In section E.1, we define the primitives we are going to use and their security notions. Using these primitives, we then present a black-box construction of a message-tracing schemewith optimal ciphertext rate in section E.2. Section E.3 contains concrete constructions from anonymous BE that has short keys in addition to the optimal ciphertext rate. Section E.4 concludes with some efficiency considerations.

E.1 Definitions

In this section, we define message tracing schemes and the building blocks we will use in their construction. We follow an approach similar to [NSS99] by defining first a two-user primitive which we then extend to the multi-user case using collusion-secure codes.

We first state the *marking assumption*, which provides a way to embed a bit in a message block. This will be applied to blocks we protect, whereas no bit will be embedded in non-protected blocks. Then, from the decoded message, the authority will be able to extract the bits involved in the decryption keys in the pirate decoder, unless the decoder drops the protected blocks. We will thus need the property that nobody can detect which blocks are protected so that if the pirate decoder decides to drop some blocks, the choice will be independent from the protection of the blocks. We will show that we can build such a *message-traceable encryption* from a *2-user anonymous broadcast encryption* scheme. Eventually, from all the bits extracted from the protected blocks (and erasures in case of dropped blocks), using the tracing algorithm of a *collusion-secure code*, we can trace back some of the traitors.

E.1.1 Primitives

A more formal definition of PKE is given in section E.5.1 of the appendix. A more formal definition of ABE is given in section E.5.1 of the appendix.

E.1.2 Marking Content

In order to trace from the message content itself, we need to be able to distribute different versions of a message to different users in an undetectable way. One way is to use watermarks. Another way could exploit different camera shots (angle and distance) of the same scene in a movie[BS98]. We abstract away from the concrete way to create versions and use the *marking assumption* that has been introduced by [BS98] and has become standard since. In the following we assume that, given two blocks m_0 and m_1,

- we can double m_b (for a random $b \in \{0,1\}$) into two *equivalent* messages m_b^0 and m_b^1
- when a user receives m'_0 and m'_1 (such that $m'_b \in \{m_b^0, m_b^1\}$ and $m'_{\bar{b}} = m_{\bar{b}}$, where $\bar{b} = 1 - b$)

one cannot guess b. This essentially means that it is possible to mark a message to protect it, but it is not possible to tell apart protected and unprotected blocks.

In addition, we also assume *robustness* with respect to a symmetric, reflexive relation $\approx_\rho$: for two equivalent blocks $m^0 \approx_\rho m^1$, when the user receives m^b, and tries to alter it (but without changing the meaning or content), he has only a negligible chance to output $m' \approx_\rho m^b$ that is closer to $m^{\bar{b}}$ than to m^b. This reflects that the user cannot change a watermark without completely changing the message.

These two quite practical assumptions guarantee that

- protected blocks are indistinguishable from non-protected blocks;
- when a user has access to one version of the protected block only, we can learn from its output which bit was embedded: the *detected bit*;
- when a user has access to both versions of the protected block, we either detect from its output one explicit bit as above, or we note that both versions have been used: in either case we can output one bit, associated to at least one version of the block available to the user.

Of course, the user can drop some blocks, but this impacts the quality of the message: we will assume that not too many blocks are dropped: at most a fraction η.

E.1.3 Collusion-Secure Codes

Collusion-secure codes [BS98] allow to trace a subset of the users (the traitors) that colluded to produce a word (pirate word) from the codewords they were given. This of course depends on the way traitors can derive words from their codewords: the *feasible set* is the set of the useful words that can be derived from the legitimate codewords. We focus on binary codes, defined over the alphabet $\{0,1\}$. In our context, each bit-value is associated to a decryption key, and a receiver has to decrypt at least one block in each pair of variants to be able to get the global content. If all the codewords in a set agree on a position (i. e. they all have the same bit at this position), then the collusion owns only one decryption key, and thus all the words in the feasible set must have the same bit at this position. However, if some of the words differ at a given position, then the collusion owns both decryption keys, and thus both values are possible at this position. More formally, for any list of t words $w_1, \ldots, w_t \in \{0,1\}^n$, $\mathrm{FS}(w_1, \ldots, w_t) = \{w \in \{0,1\}^n \mid \forall i \in \{1, \ldots, n\}, \exists j \in \{1, \ldots, t\}, w[i] = w_j[i]\}$.

Definition E.1.1 An (N, t, ε, n)-*collusion-secure code* $\mathcal{T}$ for FS is defined by a pair of algorithms $(\mathsf{Gen}, \mathsf{Trace})$, where

- $\mathsf{Gen}(N, \varepsilon)$ takes as input the number N of codewords to output and an error probability ε, it outputs a tracing key tk and a code $\Gamma \subset \{0,1\}^n$ of size N.
- $\mathsf{Trace}(\mathsf{tk}, w)$ takes as input the tracing key tk and a word $w \in \mathrm{FS}(C)$, where C is a collusion of at most t codewords, it outputs a codeword w.

The running time of both algorithms must be polynomial in $N \log(1/\varepsilon)$, and the tracing algorithm should not be wrong too often: with probability less than ε, $w \notin C$.

More precisely, an (N, t, ε, n)-collusion-secure code for FS guarantees that

- given $(\Gamma, \mathsf{tk}) \leftarrow \mathsf{Gen}(N, \varepsilon)$, with $\Gamma \subset \{0,1\}^n$ of size N
- for any collusion $C \subset \Gamma$ of size at most t, for any $w \in \mathrm{FS}(C)$, $\mathsf{Trace}(\mathsf{tk}, w)$ outputs a word in C with probability $1 - \varepsilon$.

Efficient constructions of such codes can be found in [Tar08b]: the resulting code length n for a c-collusion secure code is $O(c^2 \log(N/\varepsilon))$, where ε is the tracing error probability.

When we consider adversaries that can drop some blocks, we need more powerful codes. A word $w^\star \in \{0, 1, \star\}^n$ is in $\mathrm{FS}^\star(C)$ if there is a word $w \in \mathrm{FS}(C)$ such that w equals $w^\star$ at all the non-$\star$ positions. The *extended feasible set* $\mathrm{FS}^\star(w)$ is the set of all words that are $\star$-feasible for w. Such a code that is traceable even in case of erasures is called *robust code*. Efficient constructions of codes for $\mathrm{FS}^\star$ can be found in [BKM10]: the resulting code length n for a c-collusion secure code with a fraction δ of the erasures is $O(c^2 \log(N/\varepsilon)/(1-\delta)^2)$.

E.1.4 Message-Traceable Encryption

A message-traceable encryption scheme Ψ is a multi-cast encryption scheme which allows all the registered users (with a legitimate secret key) to decrypt a ciphertext. In addition, from the decrypted content, it is possible to derive the key (or even the keys) used for the decryption. In the following description, we focus on static schemes (the maximum number of users is set from the beginning):

- $\mathsf{Setup}(1^\kappa, N, t)$, where κ is the security parameter, N the number of users and t the maximal size of a collusion, generates the global parameters param of the system (omitted in the following), N user secret keys $\{\mathsf{USK}_{\mathsf{id}}\}_{\mathsf{id}=1,\ldots,N}$, an encryption key EK, and a tracing key TK.
- $\mathsf{Encrypt}(\mathsf{EK}, m)$ takes as input the encryption key EK and a message m to be sent, it generates a ciphertext c.
- $\mathsf{Decrypt}(\mathsf{USK}, c)$ takes as input a decryption key USK and a ciphertext c, it outputs a message m, or the error symbol $\perp$.
- $\mathsf{Trace}(\mathsf{TK}, c, m)$ takes as input the tracing key TK, with a ciphertext c and the decrypted message m, it returns an index $\mathsf{id} \in [1, N]$ of a user secret key $\mathsf{USK}_{\mathsf{id}}$.

Security Notions. As for any encryption scheme, the first security notion to define is the semantic security, whose security game is presented in figure E.1. Of course, to make tracing possible, the encryption algorithm will possibly derive several equivalent versions of the message m_b to be encrypted, which will decrypt to slightly different messages depending on the key used to decrypt. For this reason, we allow the adversary to choose which decryption key should be used, hence the additional input id.

Definition E.1.2 [Semantic Security] A message-traceable encryption scheme Ψ is said to be $(\tau, N, t, q_D, \varepsilon)$-IND-CCA-secure (semantic security against chosen-ciphertext attacks) if in the security game presented in figure E.1, the advantage, denoted $\mathrm{Adv}_{\Psi}^{\mathsf{ind-cca}}(\kappa, \tau, N, t, q_D)$, of any τ-time adversary $\mathcal{A}$ asking for at most q_D decryption queries ($\mathsf{ODecrypt}$ oracle) is bounded by ε:

$$\mathrm{Adv}_{\Psi}^{\mathsf{ind-cca}}(\kappa, \tau, N, t, q_D) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}_{\Psi,\mathcal{A}}^{\mathsf{ind-cca}-1}(\kappa, N, t) = 1] - \Pr[\mathrm{Exp}_{\Psi,\mathcal{A}}^{\mathsf{ind-cca}-0}(\kappa, N, t) = 1]\} \leq \varepsilon.$$

This definition includes IND-CPA (for Chosen-Plaintext Attacks) when $q_D = 0$, and thus we denote the advantage $\mathrm{Adv}_{\Psi}^{\mathsf{ind-cpa}}(\kappa, \tau, N, t)$.

$\mathrm{Exp}_{\Psi,\mathcal{A}}^{\mathsf{ind-cca}-b}(\kappa, N, t)$
$(\{\mathsf{USK}_{\mathsf{id}}\}, \mathsf{EK}, \mathsf{TK}) \leftarrow \mathsf{Setup}(1^{\kappa}, N, t)$; $\mathcal{Q}_D \leftarrow \emptyset$;
$(state, m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{ODecrypt}(\cdot,\cdot)}(\mathsf{FIND}; \mathsf{EK})$;
$c^* \leftarrow \mathsf{Encrypt}(\mathsf{EK}, m_b)$;
$b' \leftarrow \mathcal{A}^{\mathsf{ODecrypt}(\cdot,\cdot)}(\mathsf{GUESS}; state, c^*)$;
`if` $c^* \in \mathcal{Q}_D$ `then return` 0 `else return` b';

$\mathsf{ODecrypt}(\mathsf{id}, c)$
$\mathcal{Q}_D \leftarrow \mathcal{Q}_D \cup \{c\}$;
$m \leftarrow \mathsf{Decrypt}(\mathsf{USK}_{\mathsf{id}}, c)$; `return` m;

Figure E.1: IND-CCA for message-traceable encryption

We now formalize the additional security notion of traceability: after having received at most t secret keys (the collusion C of traitors), the adversary asks for a ciphertext $c^\star$ of a random message $m^\star$, and outputs a plaintext m that should be equivalent to m^*. The tracing algorithm should then output one of the traitors, otherwise the adversary has won the game. We use the relation $m \approx_\rho m'$ from section E.1.2 to denote that two messages are "similar" in the current context. If the adversary sends a random message (hence $m \not\approx_\rho m^\star$) or alternatively outputs an empty message, we say the adversary lost the game:

Definition E.1.3 [Traceability] A message-traceable encryption scheme Ψ is said to be $(\tau, N, t, \varepsilon)$-traceable if in the security game presented in figure E.2, the success probability, denoted $\mathsf{Succ}_{\Psi,\mathcal{A}}^{\mathbf{Trace}}(\kappa, \tau, N, t)$, of any τ-time adversary asking for at most t secret keys is bounded by ε:

$$\mathsf{Succ}_{\Psi}^{\mathbf{Trace}}(\kappa, \tau, N, t) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}_{\Psi,\mathcal{A}}^{\mathbf{Trace}}(\kappa, N, t) = 1]\} \leq \varepsilon.$$

E.2 A Generic Construction from PKE

E.2.1 Naive Construction

To have a baseline against which to compare our later constructions, we first outline the naive way to construct a message tracing scheme. The construction uses a PKE scheme Π and assigns user keys according to the codewords of the collusion-secure code $\mathcal{T}$. If the codewords have length n, we need $2n$ instances of the PKE scheme.

$\mathrm{Exp}^{\mathbf{Trace}}_{\Psi,\mathcal{A}}(\kappa, N, t)$
$(\{\mathsf{USK}_{\mathsf{id}}\}, \mathsf{EK}, \mathsf{TK}) \leftarrow \mathsf{Setup}(1^\kappa, N, t)$; $\mathcal{Q}_C \leftarrow \emptyset$;
$(state, C) \leftarrow \mathcal{A}(\mathsf{FIND}; \mathsf{EK})$;
$m^\star \xleftarrow{\$} \mathcal{M}$; $c^\star \leftarrow \mathsf{Encrypt}(\mathsf{EK}, m^\star)$;
$m \leftarrow \mathcal{A}(\mathsf{GUESS}; state, c^\star, \{\mathsf{USK}_{\mathsf{id}}\}_{\mathsf{id}\in C})$;
$T \leftarrow \mathsf{Trace}(\mathsf{TK}, c^\star, m)$;
`if` $m = \perp$ `or` $m \not\approx_\rho m^\star$ `then return` 0;
`if` $T \cap C = \emptyset$ `then return` 1 `else return` 0;

Figure E.2: Traceability

- $\mathsf{Setup}(1^\kappa, N, t)$
  1. It first generates a (N, t, ε, n)-traceable code, using $(\Gamma, \mathsf{tk}) \leftarrow \mathcal{T}.\mathsf{Gen}(N, \varepsilon)$, with a low error value ε. We thus denote n the length of a codeword in Γ, and enumerate codewords with indices associated to each users: $\Gamma = \{w_{\mathsf{id}}\}_{\mathsf{id}=1,\ldots,N} \subset \{0,1\}^n$.
  2. it then calls $2n$ times $\Pi.\mathsf{Setup}(1^\kappa)$ to obtain $(\mathsf{dk}_i^b, \mathsf{ek}_i^b)_{b=0,1,i=1\ldots n}$.
  3. it sets $\mathsf{EK} \leftarrow \{\mathsf{ek}_i^b\}_{b=0,1,i=1\ldots n}$, $\mathsf{USK}_{\mathsf{id}} \leftarrow (\mathsf{dk}_i^{w_{\mathsf{id}}[i]})_{i=1\ldots n}$ for all $\mathsf{id} \in [1, N]$, $\mathsf{TK} \leftarrow (\{\mathsf{dk}_i^0, \mathsf{dk}_i^1\}_{i=1,\ldots,n}, \mathsf{tk})$.
- $\mathsf{Encrypt}(\mathsf{EK}, m)$ first splits m into n blocks $m_1, \ldots m_n$. For each block,
  1. it creates two versions m_i^0, m_i^1 of each block m_i
  2. it then encrypts the versions as $c_i^b = \Pi.\mathsf{Encrypt}(\mathsf{ek}_i^b, m_i^b)$
  3. it sets $c = (c_1^0, c_1^1, \ldots, c_n^0, c_n^1)$.
- $\mathsf{Decrypt}(\mathsf{USK}_{\mathsf{id}}, c)$
  1. it parses $c = (c_1^0, c_1^1, \ldots, c_n^0, c_n^1)$
  2. it decrypts $m_i^{w_{\mathsf{id}}[i]} = \Pi.\mathsf{Decrypt}(\mathsf{dk}_i^{w_{\mathsf{id}}[i]}, c_i^{w_{\mathsf{id}}[i]})$ to recover m.
- $\mathsf{Trace}(\mathsf{TK}, c, m)$ extracts the word w' from m and calls $\mathcal{T}.\mathsf{Trace}(\mathsf{tk}, w')$ to get the codeword w_{id} of a colluder.

E.2.2 Improved Construction

We can reduce the ciphertext rate for long messages by watermarking only some blocks. We now describe a generic construction that accomplishes this by encrypting a message consisting of n sequences of ℓ blocks each in such a way that in sequence i, $\ell - 1$ blocks can be decrypted by both users; these blocks are not used for tracing. The other block is duplicated using to different marks and encrypted at two positions $v_0[i], v_1[i]$, each time for one key only: the message at position $v_1[i]$ can only be decrypted by users with key 0, and the message at position $v_0[i]$ only by users with key 1. By doing this, the ciphertext will have a length of $(1+1/l)$-times the length of the message, plus the overhead for encryption.

To reduce the overhead for encryption for long messages, we now model the PKE scheme as a KEM. Given any symmetric cipher $\mathcal{E} = (\mathsf{Enc}, \mathsf{Dec})$, a PKE Π, and a traceable code $\mathcal{T}$, we construct a message-traceable encryption scheme $\hat{\Psi}(\ell, n)$ as follows:

- $\mathsf{Setup}(1^\kappa, N, t, \ell)$:

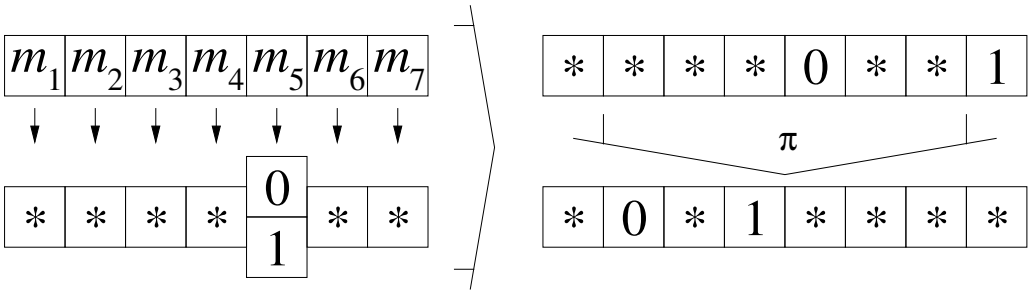


Figure E.3: Hiding a mark at position 5 in a sequence of 7 blocks.

1. It first generates a (N, t, ε, n)-traceable code, using $(\Gamma, \mathsf{tk}) \leftarrow \mathcal{T}.\mathsf{Gen}(N, \varepsilon)$, with a low error value ε. We thus denote n the length of a codeword in Γ, and enumerate codewords with indices associated to each users: $\Gamma = \{w_{\mathsf{id}}\}_{\mathsf{id}=1,\ldots,N} \subset \{0,1\}^n$.
2. It then calls $\Pi.\mathsf{Setup}(1^\kappa)$ $n(\ell+1)$ times to obtain, for $i = 1, \ldots, n$ and $j = 1, \ldots, \ell+1$, $\mathsf{ek}_{i,j}, \mathsf{dk}_{i,j}$. It draws to random vectors $v_0, v_1 \in [1, \ell+1]^n$ with the condition that $v_0[i] \neq v_1[i]$ for all $i = 1, \ldots n$. The position $v_b[i]$ describes the secret key that the users with $w_{\mathsf{id}}[i] = b$ do *not* have. We set

$$\begin{aligned}
\mathsf{USK}_{\mathsf{id}} &\leftarrow \{\mathsf{dk}_{i,j}\}_{\substack{i=1,\ldots,n \\ j \neq v_{w_{\mathsf{id}}[i]}[i]}}, \\
\mathsf{EK} &\leftarrow (\{\mathsf{ek}_{i,j}\}_{\substack{i=1,\ldots,n \\ j=1,\ldots,\ell+1}}, v_0, v_1), \\
\mathsf{TK} &\leftarrow (\{\mathsf{dk}^0_{i,j}, \mathsf{dk}^1_{i,j}\}_{\substack{i=1,\ldots,n \\ j=1,\ldots,\ell+1}}, \mathsf{tk}).
\end{aligned}$$

- $\mathsf{Encrypt}(\mathsf{EK}, m)$ first splits m in $n\ell$ blocks $\{m_{i,j}\}_{\substack{i=1,\ldots,n \\ j=1,\ldots,\ell}}$. For each sequence, at position $i \in \{1, \ldots, n\}$:

  1. it chooses a random position $k \in \{1, \ldots, \ell\}$, to protect block $m_{i,k}$;
  2. it generates two equivalent versions $m^0_{i,k}, m^1_{i,k}$, of this block (see figure E.3), resulting in a list of $\ell + 1$ blocks;
  3. it prepends the position to the block: $M_j = j \| m_{i,j}$, for $j = 1, \ldots, \ell$, $j \neq k$, $M_k = k \| m^0_{i,k}$, $M_{\ell+1} = k \| m^1_{i,k}$;
  4. it chooses a random permutation $\pi \in S_{\ell+1}$ with the restriction that the position of the marked blocks is $v_1[i] = \pi(k)$ and $v_0[i] = \pi(\ell+1)$; and permutes the blocks: $M'_j = M_{\pi(j)}$.
  5. it generates the session keys: $(c_{i,j}, K_{i,j}) \leftarrow \Pi.\mathsf{Encaps}(\mathsf{ek}_{i,j})$,
  6. It then encrypts the blocks under the symmetric keys: $C_{i,j} \leftarrow (c_{i,j}, c'_{i,j} = \mathsf{Enc}_{K_{i,j}}(M'_j))$ for $j = 1, \ldots, \ell+1$.

  The final ciphertext consists of all the pairs $C_{i,j}$ for $i = 1, \ldots, n$ and $j = 1, \ldots, \ell+1$.

- $\mathsf{Decrypt}(\mathsf{USK}_{\mathsf{id}}, C)$ takes as input the key $\mathsf{USK}_{\mathsf{id}} = \{\mathsf{dk}_{i,j}\}_{\substack{i=1,\ldots,n \\ j \neq v_{w_{\mathsf{id}}[i]}[i]}}$ and a ciphertext $C = \{C_{i,j}\}$. For each sequence, at position $i \in \{1, \ldots, n\}$:

  1. it calls $\Pi.\mathsf{Decaps}(\mathsf{usk}^{w_{\mathsf{id}}[i]}_{i,j}, c_{i,j})$, for $j \neq v_{w_{\mathsf{id}}[i]}[i]$, to obtain the session key $K_{i,j}$;
  2. it decrypts the message with $\mathsf{Dec}(K_{i,j}, c'_{i,j})$, which outputs $M'_{i,j}$;

3. it should be able to parse the $M'_{i,j} = p_j \| m_{i,j}$, with $\{p_j\} = \{1, \ldots, \ell\}$, otherwise it stops and outputs $\perp$;
4. it eventually reorders the messages according to p_j, and concatenates the other parts.

It concatenates the ℓ blocks in each sequences, and the n sequence-results to output the full message m.

- $\mathsf{Trace}(\mathsf{TK}, C, m)$ can detect the protected blocks using the decryption keys in TK. From the block that was actually decrypted in each sequence i, it can learn the value of the bit $w[i]$. Then, granted the traceability of the code $\mathcal{T}$, the $\mathcal{T}.\mathsf{Trace}(\mathsf{tk}, w)$ outputs a traitor.

Remark E.2.1 The traceability of the scheme rests on the fact that a user does not know which of the keys are common to all users and which are specific to those with the same bit in the codeword. While a user that shares the information which positions he cannot decrypt with other users is considered to be misbehaving and thus corrupted in our security model, the real-life cost of sharing some of these positions is quite low. The scheme is thus susceptible to a Pirates 2.0-attack as described in [BP09].

E.2.3 Reusing Randomness

To further reduce the ciphertext rate, we can try to reduce the size of the key encapsulation, by reusing the random coins in all the ciphertexts of a sequence or even the complete ciphertext.

We need to make sure that the PKE scheme in the construction remains secure if randomness us reused. If we instantiate the PKE scheme with ElGamal, we can use the results of Bellare, Boldyreva, Kurosawa, and Staddon [BBKS07, sec. 7.1].

The only change is in the encryption:

- $\mathsf{Encrypt}(\mathsf{EK}, m)$: it first splits m in $n\ell$ blocks $\{m_{i,j}\}_{\substack{i=1,\ldots,n \\ j=1,\ldots,\ell}}$. For each sequence, at position $i \in \{1, \ldots, n\}$:

5. it draws the common randomness $R \xleftarrow{\$} \mathcal{R}$, then generates the session keys: $(c_{i,j}, K_{i,j}) \leftarrow \Pi.\mathsf{Encaps}(\mathsf{ek}_{i,j}; R)$,

The final ciphertext consists, as above, of all the pairs $C_{i,j} = (c_{i,j}, c'_{i,j})$, but where all the $c_{i,j}$ in the same sequence use a common part that can be included once only.

Reusing randomness incurs a loss in the security reduction for the PKE scheme that is equal to the number of ciphertexts that share the same randomness [BBKS07, th. 7.3]. This means that e. g. for ElGamal, the group has to be larger. If all $n\ell$ blocks share the same randomness, then for 80-bit security, the group size needs to provide $80 + \log(n\ell)$ bits. To balance the increase in the length of the group elements against the reduced number of group elements necessary, we need to minimize the overall overhead spent for the randomness, which is equal to the number of group elements needed times their bitlength, or $\frac{n\ell}{x}|G_{\kappa+\log(x)}|$.

E.2.4 Security

For reasons of space, and since these constructions are not the main result of this paper, we do not give a security proof for them. We believe that given the security proof of our final construction, the security of the above constructions is an easy corollary.

E.3 A Construction With Shorter Keys

The main disadvantage of the PKE-based construction is the length of the user keys, which must contain a PKE key for each block. To achive shorter keys, we use a primitive that allows encryption to either of two users or to both of them: 2-user broadcast encryption.

Our message-traceable encryption scheme makes use of codes, where the bits of the codewords are embedded in a message by doubling some parts of it, the so-called *protected blocks*. Because we do not want the adversary to learn which parts of the message contain bits of the codeword, we need a broadcast encryption scheme where a user cannot tell whether a block is destined only for his key or for both keys, a 2-user anonymous broadcast encryption (2ABE).

This requires the symmetric cipher used with this construction to be weakly robust[ABN10], since one of the decapsulated keys will be either $\perp$ or an unusable key. The construction uses one instance of the 2ABE scheme Π per bit of the codeword, encrypting $\ell+1$ messages at a time in one sequence, with the target sets determined by the positions v, w where the watermarks are embedded. In this construction, the length of the EK and USK is n times that of Π, and to encrypt a sequence of ℓ blocks, doubling one block, we need $\ell+1$ Π key-encapsulations plus $\ell+1$ symmetrically encrypted message blocks.

E.3.1 Construction of a Message-Traceable Encryption Scheme

Our first construction combines a traceable code $\mathcal{T}$ with a 2ABE scheme Π. If the codewords have length n, we need n instances of the 2ABE scheme. Given any weakly robust[ABN10] symmetric cipher $\mathcal{E} = (\mathsf{Enc}, \mathsf{Dec})$, a 2-user anonymous broadcast encryption Π, and a traceable code $\mathcal{T}$, we construct a message-traceable encryption scheme $\Psi(\ell, n)$ as follows:

- $\mathsf{Setup}(1^\kappa, N, t, \ell)$:
  1. It first generates a (N, t, ε, n)-traceable code, using $(\Gamma, \mathsf{tk}) \leftarrow \mathcal{T}.\mathsf{Gen}(N, \varepsilon)$, with a low error value ε. We thus denote n the length of a codeword in Γ, and enumerate codewords with indices associated to each users: $\Gamma = \{w_{\mathsf{id}}\}_{\mathsf{id}=1,\dots,N} \subset \{0,1\}^n$.
  2. It then calls $\Pi.\mathsf{Setup}(1^\kappa, 2)$ n times to obtain, for $i = 1, \dots, n$, $\mathsf{ek}_i, \mathsf{usk}_i^0, \mathsf{usk}_i^1$. We set

$$\begin{aligned} \mathsf{USK}_{\mathsf{id}} &\leftarrow (\mathsf{usk}_1^{w_{\mathsf{id}}[1]}, \dots, \mathsf{usk}_n^{w_{\mathsf{id}}[n]}), \\ \mathsf{EK} &\leftarrow (\mathsf{ek}_1, \dots, \mathsf{ek}_n), \\ \mathsf{TK} &\leftarrow (\{\mathsf{usk}_i^0, \mathsf{usk}_i^1\}_{i=1,\dots,n}, \mathsf{tk}). \end{aligned}$$

- $\mathsf{Encrypt}(\mathsf{EK}, m)$: it first splits m in $n\ell$ blocks $\{m_{i,j}\}_{\substack{i=1,\dots,n \\ j=1,\dots,\ell}}$. For each sequence, at position $i \in \{1, \dots, n\}$:
  1. it chooses a random position $k \in \{1, \dots, \ell\}$, to protect block $m_{i,k}$;
  2. it generates two equivalent versions $m_{i,k}^0, m_{i,k}^1$, of this block (see figure E.3), resulting in a list of $\ell+1$ blocks;
  3. it prepends the position to the block: $M_j = j \| m_{i,j}$, for $j = 1, \dots, \ell$, $j \neq k$, $M_k = k \| m_{i,k}^0$, $M_{\ell+1} = k \| m_{i,k}^1$;
  4. it chooses a random permutation $\pi \in S_{\ell+1}$ and permutes the blocks: $M'_i = M_{\pi(i)}$. We note $v = \pi(k)$ and $w = \pi(\ell+1)$, the positions of the two equivalent blocks;
  5. it generates session keys for all the blocks, except M'_v and M'_w, with the 2ABE scheme Π, with the full target set $\{0,1\}$, whereas M'_v is targeted to $\{0\}$ only, and M'_w is targeted to $\{1\}$ only. More precisely, it first generates the session keys:

$(c_{i,j}, K_{i,j}) \leftarrow \Pi.\mathsf{Encaps}(\mathsf{ek}_i, \{0,1\})$, for $j \neq v, w$, $(c_{i,v}, K_{i,v}) \leftarrow \Pi.\mathsf{Encaps}(\mathsf{ek}_i, \{0\})$, and $(c_{i,w}, K_{i,w}) \leftarrow \Pi.\mathsf{Encaps}(\mathsf{ek}_i, \{1\})$.

6. It then encrypts the blocks under the symmetric keys: $C_{i,j} \leftarrow (c_{i,j}, c'_{i,j} = \mathsf{Enc}_{K_{i,j}}(M'_j))$ for $j = 1, \ldots, \ell+1$.

The final ciphertext consists of all the pairs $C_{i,j}$ for $i = 1, \ldots, n$ and $j = 1, \ldots, \ell+1$.

- $\mathsf{Decrypt}(\mathsf{USK}_{\mathsf{id}}, C)$ takes as input the key $\mathsf{USK}_{\mathsf{id}} = (\mathsf{usk}_1^{w_{\mathsf{id}}[1]}, \ldots, \mathsf{usk}_n^{w_{\mathsf{id}}[n]})$ and a ciphertext $C = \{C_{i,j}\}$. For each sequence, at position $i \in \{1, \ldots, n\}$:

  1. it calls $\Pi.\mathsf{Decaps}(\mathsf{usk}_i^{w_{\mathsf{id}}[i]}, c_{i,j})$, for $j = 1, \ldots, \ell+1$, to obtain the session key $K_{i,j}$;
  2. it decrypts the message with $\mathsf{Dec}(K_{i,j}, c'_{i,j})$, which outputs either $M'_{i,j}$ or $\perp$ (because of the robustness);
  3. it should be able to parse the $M'_{i,j} = p_j \| m_{i,j}$, with $\{p_j\} = \{1, \ldots, \ell\}$, otherwise it stops and output $\perp$;
  4. it eventually reorders the messages according to p_j, and concatenates the other parts.

  It concatenates the ℓ blocks in each sequences, and the n sequence-results to output the full message m.

- $\mathsf{Trace}(\mathsf{TK}, C, m)$ can detect the protected blocks using the decryption keys in TK. From the block that was actually decrypted in each sequence i, it can learn the value of the bit $w[i]$. Then, granted the traceability of the code $\mathcal{T}$, the $\mathcal{T}.\mathsf{Trace}(\mathsf{tk}, w)$ outputs a traitor.

E.3.2 Security of the Construction

Theorem E.3.1 If the 2ABE scheme Π is IND-CPA and the symmetric encryption scheme $\mathcal{E}$ is IND-CPA, then our construction $\Psi(\ell, n)$ is IND-CPA, and

$$\mathrm{Adv}^{\mathsf{ind-cpa}}_{\Psi(\ell,n)}(\kappa, \tau, N, t) \leq n \cdot (\ell+1) \times \left(2 \cdot \mathrm{Adv}^{\mathsf{ind-cpa}}_{\Pi}(\kappa, \tau_1, 2) + \mathrm{Adv}^{\mathsf{ind-cpa}}_{\mathcal{E}}(\kappa, \tau_2)\right).$$

Proof: Let $\mathcal{A}$ be an adversary against the IND-CPA-security of our construction Ψ. We provide a bound on its advantage using a series of games. The simulator first asks for n public keys ek_i, for $i = 1, \ldots, n$, to $\Pi.\mathsf{Setup}(1^\kappa, 2)$, and thus sends the public key EK to the adversary $\mathcal{A}$. The latter sends back two messages $m^0 = (m^0_{1,1}, \ldots, m^0_{1,\ell}, \ldots, m^0_{n,1}, \ldots, m^0_{n,\ell})$ and $m^1 = (m^1_{1,1}, \ldots, m^1_{1,\ell}, \ldots, m^1_{n,1}, \ldots, m^1_{n,\ell})$:

- In game G_0, the simulator encrypts m^0, with k_i the indices of the protected blocks for sequences $i = 1, \ldots, n$, and π_i the permutations;
- In game G_1, the simulator still encrypts m^0, but with random keys for all the key encapsulations of the 2ABE scheme: with an hybrid sequence of games, we can show that the distance between game G_1 and game G_0 is bounded by $n(\ell+1) \cdot \mathrm{Adv}^{\mathsf{ind-cpa}}_{\Pi}(\kappa, \tau_1, 2)$;
- In game G_2, the simulator encrypts m^1, still with random keys for all the key encapsulations of the 2ABE scheme: with an hybrid sequence of games, we can show that the distance between game G_2 and game G_1 is bounded by $n(\ell+1) \cdot \mathrm{Adv}^{\mathsf{ind-cpa}}_{\mathcal{E}}(\kappa, \tau_2)$;
- In game G_3, the simulator encrypts m^1, with k_i the indices of the protected blocks for sequences $i = 1, \ldots, n$, and π_i the permutations: with an hybrid sequence of games, we can show that the distance between game G_3 and game G_2 is bounded by $n(\ell+1) \cdot \mathrm{Adv}^{\mathsf{ind-cpa}}_{\Pi}(\kappa, \tau_1, 2)$;

This concludes the proof. ▮

Before we prove traceability, we prove a lemma that will help us in the main proof. The lemma states that no adversary can tell which blocks are not encrypted to all users, if he only has one of the two keys, w.l.o.g. usk_i^0.

Lemma E.3.2 If the 2ABE scheme Π is both IND-CPA and ANO-CPA, and the symmetric encryption scheme $\mathcal{E}$ is IND-CPA, then an adversary who only has the usk_i^0 for a sequence i cannot distinguish between the case where the block at position v is encrypted to the target set $\{0\}$ and the block at position w is encrypted to the target set $\{1\}$ and the case where the block at position v is encrypted to the target set $\{0,1\}$ and the block at position w contains a random message. If we denote by $\mathrm{Adv}_{\Psi}^{\mathsf{find}}(\kappa,\tau,N,t)$ the maximal advantage of any adversary within time τ, on any index i, then

$$\mathrm{Adv}_{\Psi}^{\mathsf{find}}(\kappa,\tau,N,t) \leq \mathrm{Adv}_{\Pi}^{\mathsf{ano-cpa}}(\kappa,\tau_1,2) + \mathrm{Adv}_{\Pi}^{\mathsf{ind-cpa}}(\kappa,\tau_2,2) + \mathrm{Adv}_{\mathcal{E}}^{\mathsf{ind-cpa}}(\kappa,\tau_3).$$

Proof: Let $\mathcal{A}$ be an adversary who can distinguish the two cases for our construction Ψ. We provide a bound on its advantage using a series of games. The simulator first asks for n public keys ek_i, for $i = 1,\ldots,n$, to $\Pi.\mathsf{Setup}(1^\kappa, 2)$, and thus sends the public key EK to the adversary $\mathcal{A}$, together with $\mathsf{USK}_{\mathsf{id}}$, that corresponds to a codeword w_{id}. The latter sends back a message $M = (M_{1,1},\ldots,M_{1,\ell},\ldots,M_{n,1},\ldots,M_{n,\ell})$. For all the sequences, except the i-th sequence, for any i, the simulator does as in the real encryption of message M, but we denote m the sequence $M_{i,1}\ldots M_{i,\ell}$.

- In game G_0, the simulator encrypts m, with k the index of the protected block, and (v,w) the positions of the specific ciphertexts to key 0 and key 1 respectively.
- In game G_1, the simulator encrypts m, with k the index of the protected block, and (v,w) the two above positions, but v-th block is encrypted for the two keys, and w-th block is encrypted for the key 1 only. Since the simulator can choose the index v to be the same as the one asked to the anonymity-game, in an indistinguishable way, the distance between game G_1 and game G_0 is bounded by $\mathrm{Adv}_{\Pi}^{\mathsf{ano-cpa}}(\kappa,\tau_1,2)$;
- In game G_2, the simulator encrypts m, with k the index of the protected block, and (v,w) the two above position, but v-th block is encrypted for the two keys, and a random session key is used for the w-th block, while a key encapsulation is sent for key 1 only. Under the semantic security of the 2ABE scheme, since the adversary does not know key 1, the two games are indistinguishable: the distance between game G_2 and game G_1 is bounded by $\mathrm{Adv}_{\Pi}^{\mathsf{ind-cpa}}(\kappa,\tau_2,2)$;
- In game G_3, the simulator encrypts m, with k the index of the protected block, and (v,w) the two above position, but the v-th block is encrypted for the two keys, and at position w, a truly random block (with no position appended) is encrypted under a random session key, while a key encapsulation is sent for key 1 only. Under the semantic security of $\mathcal{E}$, the two games are indistinguishable: the distance between game G_3 and game G_2 is bounded by $\mathrm{Adv}_{\mathcal{E}}^{\mathsf{ind-cpa}}(\kappa,\tau_3)$.

▮

Theorem E.3.3 Even if the adversary can drop a fraction η of the message, if the 2ABE scheme Π is both IND-CPA and ANO-CPA, the symmetric encryption scheme $\mathcal{E}$ is IND-CPA, and the code

$\mathcal{T}$ is (N, t, ε, n)-traceable for FS* for a fraction $\delta > \eta$ of erasures, then our construction Ψ is traceable. More precisely, one needs

$$(\delta - \eta)^2 \geq \frac{1}{2} \times \mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa, \tau, N, t).$$

Proof: We first fix some notation. Let $\eta_{i,j} \stackrel{\mathsf{def}}{=} \Pr[drop(i,j)]$ be the probability that block j in sequence i is dropped. We denote by $\eta \stackrel{\mathsf{def}}{=} E[\eta_{i,j}]$ the global average probability of a block to be dropped. This means that overall, $n\ell\eta = \sum_{i,j} \eta_{i,j}$ blocks will be dropped. We also consider the average in a sequence i: $\eta_i \stackrel{\mathsf{def}}{=} E[\eta_{i,j}]$. Then $\eta = E[\eta_i]$. Now we consider only the protected blocks. Analogously, let $\theta_i \stackrel{\mathsf{def}}{=} \Pr[drop(i,k_i)]$ be the probability for the protected block in sequence i to be dropped. The average probability of a protected block to be dropped is $\theta \stackrel{\mathsf{def}}{=} E[\theta_i]$. This means that the global number of erasures (dropped protected blocks) is $n\theta = \sum_i \theta_i$. We also consider the gap between protected blocks and any block, by first defining $\gamma_i \stackrel{\mathsf{def}}{=} \theta_i - \eta_i$. We additionally define $\gamma \stackrel{\mathsf{def}}{=} E[\gamma_i] = \theta - \eta$, we want to show to be small.

Let us define the subset of the sequences in which the gap γ_i is greater than $\gamma - \alpha$ for some α: $I \stackrel{\mathsf{def}}{=} \{i | \gamma_i \geq \gamma - \alpha\}$. We choose a random sequence index $i \in \{1, \ldots, n\}$, and by the splitting lemma [PS00, lemma 1], we know that $\Pr[i \in I] = \Pr[\gamma_i \geq \gamma - \alpha] \geq \alpha$, so with probability greater than α, the gap $\gamma_i = \theta_i - \eta_i$ between the probabilities for the protected block and any block in sequence i to be dropped is greater than $\gamma - \alpha$.

We now focus on this sequence i and the sub-message $m = m_1 \ldots m_\ell$, where m_k is the protected block. We consider the probability of the adversary to drop block k. If the adversary drops the block k, the simulator outputs 1, otherwise the simulator outputs 0. When interacting with the real scheme, the advantage of the simulator (defined as $\Pr[1 \leftarrow S] - \Pr[0 \leftarrow S]$) in this game is $\theta_i - (1 - \theta_i) = 2\theta_i - 1$. If we change the scheme so that now m_k is encrypted to both parties, and instead of doubling it, a random block is added, then the advantage is $2\eta_i - 1$.

By lemma E.3.2, the difference between these games is bounded by $\mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa, \tau, N, t)$. We conditioned on $i \in I$, or, equivalently, $\theta_i - \eta_i = \gamma_i \geq \gamma - \alpha$, which happens with probability α.

$$\begin{aligned}
2\theta_i - 1 - (2\eta_i - 1) &\leq \frac{1}{\alpha} \times \mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa, \tau, N, t) \\
2(\gamma - \alpha) &\leq \frac{1}{\alpha} \times \mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa, \tau, N, t) \\
\gamma &\leq \alpha + \frac{1}{2\alpha} \times \mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa, \tau, N, t) \\
\theta &\leq (\alpha + \eta) + \frac{1}{2\alpha} \times \mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa, \tau, N, t).
\end{aligned}$$

Then, except when it drops blocks, the adversary outputs messages for the protected blocks, and under the assumptions we did on the marking of contents, we can detect a bit. Furthermore, the detected bit follows the rule for the feasible set FS, and erasures (dropped blocks) extend it to FS*. Since the code $\mathcal{T}$ is (N, t, ε, n)-traceable for FS* for a fraction δ of erasures, then our construction Ψ is traceable with tracing-error probability less than ε if the average fraction θ of

dropped protected blocks is less than δ: this is, if it exists α such that

$$(\alpha+\eta)+\frac{1}{2\alpha}\times \mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa,\tau,N,t)\leq \delta$$
$$\alpha+(\eta-\delta)+\frac{1}{2\alpha}\times \mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa,\tau,N,t)\leq 0$$
$$2\alpha^2-2(\delta-\eta)\alpha+\mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa,\tau,N,t)\leq 0.$$

To find the minimum, we take the derivative:

$$4\alpha-2(\delta-\eta)=0$$
$$\alpha=\frac{\delta-\eta}{2}.$$

This is possible as soon as

$$2\alpha^2-2(\delta-\eta)\alpha+\mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa,\tau,N,t)\leq 0$$
$$\frac{(\delta-\eta)^2}{2}-(\delta-\eta)^2+\mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa,\tau,N,t)\leq 0$$
$$\mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa,\tau,N,t)\leq\frac{(\delta-\eta)^2}{2}.$$

This means that we can trace as long as we choose

$$\delta\geq\eta+\sqrt{2\mathrm{Adv}^{\mathsf{find}}_{\Psi}(\kappa,\tau,N,t)}.$$

∎

E.3.3 A 2-user Anonymous Broadcast Encryption Scheme

We view the 2-key 1-copyrighted public-key encryption scheme of Kiayias and Yung [KY02c], as a 2-user 1-collusion-secure anonymous broadcast encryption scheme (2ABE). For ease of exposition, we model the scheme as a KEM.

Let G be a group of prime order q, with a generator g. The public parameters consist of (G,q,g). Since we consider the 2-user case, we drop the N parameter:

- $\mathsf{Setup}(1^\kappa)$ picks $\alpha,\beta\xleftarrow{\$}\mathbb{Z}_q^\times$. For the two user-keys one chooses $d'_0,d'_1\in\mathbb{Z}_q$, and sets $\mathsf{usk}_u\stackrel{\mathsf{def}}{=}(d_u=\alpha-d'_u\cdot\beta,d'_u)$, for $u=0,1$. The encryption key is $\mathsf{ek}\stackrel{\mathsf{def}}{=}\{(f=g^\alpha,h=g^\beta),\mathsf{upk}_0=h^{d'_0},\mathsf{upk}_1=h^{d'_1}\}$.
- $\mathsf{Encaps}(\mathsf{ek},S;r)$ where $r\in\mathbb{Z}_q^\times$
  - if $S=\{0,1\}$ then $c=(g^r,h^r),K=f^r$
  - else if $S=\{u\}$ then $r'\xleftarrow{\$}\mathbb{Z}_q^\times$, and $c=(g^r,h^{r'}),K=(f/\mathsf{upk}_u)^r\times\mathsf{upk}_u^{r'}$
- $\mathsf{Decaps}(\mathsf{usk}_u,c)$ computes $K=c_0^{d_u}c_1^{d'_u}$. This is equal to $g^{rd_u}\times h^{r'd'_u}=(f/\mathsf{upk}_u)^r\times h^{r'}$. In the latter encryption case, one gets the same session key. In the former encryption case, since $r'=r$, one also gets f^r.

This is a broadcast encryption, because when $S=\{u\}$ user $1-u$ decapsulates differently. Anonymity comes from the fact that a ciphertext is either a Diffie-Hellman pair, when $S=\{0,1\}$, and a random pair in the other case.

E.3.4 Security of the 2ABE

Theorem E.3.4 If solving the DDH problem in the underlying group is hard, then the 2ABE scheme presented in section E.3.3 is ANO-ACPA-secure and

$$\mathrm{Adv}^{\mathsf{ano-acpa}}_{2ABE}(\kappa,\tau) \leq 4 \cdot \mathrm{Adv}^{\mathsf{ddh}}(\kappa,\tau+\tau').$$

Proof: The simulator is given a tuple $(g, A = g^a, B = g^b, C = g^c)$. First the simulator will guess the target set. There are only three possibilities (up to the order of the target sets). The simulator chooses

1. $\{0\},\{1\}$ with probability 1/2.
   In this case, we implicitly set $d'_u = a$ for a user u, and $r = b$ for the challenge random coins. The simulator proceeds as follows. It first flips a bit u to determine where it will embed the challenge. It chooses random $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_q^\times$, and defines $f = g^\alpha, h = g^\beta$. It generates $\mathsf{usk}_{1-u} = (d_{1-u}, d'_{1-u})$ as usual, and thus $\mathsf{upk}_{1-u} = h^{d'_{1-u}}$, but $\mathsf{upk}_u = A$. It thus defines $\mathsf{ek} = ((f,h), \mathsf{upk}_0, \mathsf{upk}_1)$. The simulator then draws a random r' and sets the challenge ciphertext to $(B, h^{r'})$. It also returns $K = B^\alpha A^{r'}/C$.

   If the tuple (g, A, B, C) was a DH tuple, we have a ciphertext $c = (g^b, h^{r'})$ and a key $K = g^{b\alpha + ar' - ab} = (f/\mathsf{upk}_u)^b \times \mathsf{upk}_u^{r'}$, that is for $S = \{u\}$. If the tuple was a random tuple, then the key is random.

2. $\{0\},\{0,1\}$ with probability 1/4.
   In this case, we implicitly set $\beta = a$, and $r = b$ for the challenge random coins. In this case, the simulator knows that only user 0 can be corrupted, so it chooses the secret key for user 0 in advance and uses it to compute a group element that implicitly has a matching α.

   The simulator proceeds as follows. It sets $h = A$, chooses random $d_0, d'_0 \xleftarrow{\$} \mathbb{Z}_q$ and computes $f \stackrel{\mathsf{def}}{=} g^{d_0} A^{d'_0}$. It sets $\mathsf{usk}_0 = (d_0, d'_0)$, $\mathsf{upk}_0 = A^{d'_0}$, and $\mathsf{upk}_1 = X$ for a random group element X. It thus defines $\mathsf{ek} = ((f, A), \mathsf{upk}_0, \mathsf{upk}_1)$. The challenge ciphertext is set to $c = (B, C)$, for the key $K = B^{d_0} C^{d'_0}$.

   If the tuple (g, A, B, C) was a DH tuple, we have a ciphertext $c = (g^b, g^{ab} = h^b)$ and a key $K = g^{bd_0 + abd'_0} = g^{b(d_0 + ad'_0)} = f^b$, that is for $S = \{0,1\}$. If the tuple was a random tuple, then $c = (g^b, g^{ab'} = h^{b'})$ where $b' = c/a \neq b$ and a key $K = g^{bd_0 + ab'd'_0} = g^{b(d_0 + ad'_0)} \times g^{ad'_0(b'-b)} = (f/\mathsf{upk}_0)^b \times \mathsf{upk}_0^{b'}$, that is for $S = \{0\}$.

3. $\{1\},\{0,1\}$ with probability 1/4.
   This case is done analogously to case 2, exchanging user 0 and 1.

∎

Theorem E.3.5 If solving the DDH problem in the underlying group is hard, then the 2ABE scheme presented in section E.3.3 is a 2-user IND-CPA-secure BE scheme and

$$\mathrm{Adv}^{\mathsf{ind-cpa}}_{2ABE}(\kappa,\tau) \leq \mathrm{Adv}^{\mathsf{ddh}}(\kappa,\tau+\tau').$$

Proof: The simulator is given a tuple $(g, A = g^a, B = g^b, C = g^c)$.

If there is no corruption, a ciphertext is $c = (g^r, h^r)$ for a key f^r. We can implicitly set $h = g^x$ for a known x, $r = a$, $f = B$, $c = (A, A^x)$ and $K = C$, which is a real key if C is the actual Diffie-Hellman value, or a random key otherwise.

In case of corruption, we proceed as in the cases 2 and 3 above, where u is the user for which we know the secret key for corruption. ∎

	|EK|	|USK|	|PTXT|	|KeyHeader|	|CTXT|	Ciphertext Rate
Plain ElGamal	$2\ell n\mathsf{G}$	$\ell n\mathsf{G}$	$\ell n\mathsf{B}$	$2\ell n\mathsf{G}$	$2\ell n\mathsf{B}$	$2 + 2\mathsf{G}/\mathsf{B}$
Plain ElGamal + RR	$2\ell n\mathsf{G}$	$\ell n\mathsf{G}$	$\ell n\mathsf{B}$	$\ell n\mathsf{G}$	$2\ell n\mathsf{B}$	$2 + \mathsf{G}/\mathsf{B}$
Imp. ElGamal	$2(\ell+1)n\mathsf{G}$	$\ell n\mathsf{G}$	$\ell n\mathsf{B}$	$2(\ell+1)n\mathsf{G}$	$(\ell+1)n\mathsf{B}$	$1 + 1/\ell + (2 + 2/\ell)\mathsf{G}/\mathsf{B}$
Imp. ElGamal + RR	$2(\ell+1)n\mathsf{G}$	$\ell n\mathsf{G}$	$\ell n\mathsf{B}$	$(\ell+1)n\mathsf{G}$	$(\ell+1)n\mathsf{B}$	$1 + 1/\ell + (1 + 1/\ell)\mathsf{G}/\mathsf{B}$
ABE-Construction	$4n\mathsf{G}$	$2n\mathsf{G}$	$\ell n\mathsf{B}$	$2(\ell+1)n\mathsf{G}$	$(\ell+1)n\mathsf{B}$	$1 + 1/\ell + (2 + 2/\ell)\mathsf{G}/\mathsf{B}$

Table E.1: Comparison

E.4 Conclusion

Table E.1 shows a comparison of several ways to do message-traceable encryption. The simplest way is to use any public-key encryption scheme to encrypt each message block twice: two pairs of keys are generated for each message block. Using ElGamal, we have one group element for the key header (the key encapsulation) per message-version block: PTXT is the plaintext and CTXT is the message-block ciphertext (data encryption). We can reduce the key header to one group element per two versions of a block by reusing randomness. However, using this method it is impossible to reduce the ciphertext rate below 2 without leaving some part of the message unprotected and exposed to untraceable rebroadcasting, since each message block is encrypted twice with the symmetric keys.

Using our construction 1, we immediately cut the number of blocks that must be sent almost in half with only a small constant increase in the key header as compared to plain ElGamal. Our construction 2, which reuses the randomness in the ciphertext, further shrinks the key header that must be transmitted with every message. The cost for this is an increase in the key size by a factor of $\ell + 1$. But asymptotically, with growing message-block size B and constant length G of a group element (or a scalar), the global ciphertext / plaintext ratio is $1 + 1/\ell$, where ℓ is our efficiency parameter, the number of blocks in a sequence.

E.5 Appendix

E.5.1 Definitions

Public-Key Encryption

Definition E.5.1 [Encryption Scheme] A public-key encryption scheme is a 4-tuple of algorithms $\mathcal{PKE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$:

- $\mathsf{Setup}(1^k)$, where k is the security parameter, generates the global parameters param of the system;
- $\mathsf{KeyGen}(\mathsf{param})$ generates a pair of keys, the public (encryption) key ek and the associated private (decryption) key dk;
- $\mathsf{Encrypt}(\mathsf{ek}, m; r)$ produces a ciphertext c on the input message m and the public key ek, using the random coins r (we may omit r when the notation is obvious);
- $\mathsf{Decrypt}(\mathsf{dk}, c)$ decrypts the ciphertext c under the private key dk. It outputs the plaintext, or $\perp$ if the ciphertext is invalid.

The correctness requirement is that $\mathsf{Decrypt}(\mathsf{dk}, \mathsf{Encrypt}(\mathsf{ek}, m)) = m$ if $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ for all parameters.

Such an encryption scheme is said to be (t, q_D, ε)-IND-CCA-secure (semantic security against chosen-ciphertext attacks) if in the security game presented in figure I.3, the advantage, denoted $\mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D)$, of any t-time adversary $\mathcal{A}$ asking at most q_D decryption queries to the ODecrypt oracle is bounded by ε:

$$\mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{ind-cca}-1}_{\mathcal{PKE},\mathcal{A}}(k) = 1] - \Pr[\mathrm{Exp}^{\mathsf{ind-cca}-0}_{\mathcal{PKE},\mathcal{A}}(k) = 1]\}.$$

This definition includes IND-CPA (for Chosen-Plaintext Attacks) when $q_D = 0$.

$\mathrm{Exp}^{\mathsf{ind-cca}-b}_{\mathcal{PKE},\mathcal{A}}(k)$
  $\mathsf{param} \leftarrow \mathsf{Setup}(1^k)$;
  $\mathcal{Q}_D \leftarrow \emptyset$, $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$;
  $(state, m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{ODecrypt}(\cdot)}(\mathsf{FIND}; \mathsf{param}, \mathsf{ek})$;
  $c^* \leftarrow \mathsf{Encrypt}(\mathsf{ek}, m_b)$;
  $b' \leftarrow \mathcal{A}^{\mathsf{ODecrypt}}(\mathsf{GUESS}, state; c^*)$;
  `if` $c^* \in \mathcal{Q}_D$ `then return` 0;
  `else return` b';

$\mathsf{ODecrypt}(c)$
  $\mathcal{Q}_D \leftarrow \mathcal{Q}_D \cup \{c\}$;
  $m \leftarrow \mathsf{Decrypt}(\mathsf{dk}, c)$;
  `return` m;

Figure E.4: $\mathcal{PKE}$: Semantic Security against Chosen-Ciphertext Attacks (IND-CCA)

Anonymous Broadcast Encryption

Anonymous broadcast encryption (ABE) allows to address a message to a subset of the users, without revealing this target set even to users who successfully decrypted the message. We define an ABE as a key encapsulation mechanism (KEM), following the definitions found in [PPS11, LPQ12], and we focus again on the static case:

- $\mathsf{Setup}(1^\kappa, N)$, where k is the security parameter, and N the number of users, generates the global parameters param of the system (omitted in the following), N user secret keys $\{\mathsf{usk}_i\}_{i=1,\ldots,N}$, and an encryption key ek.
- $\mathsf{Encaps}(\mathsf{ek}, S; r)$ takes as input the encryption key ek, the target set $S \subset \{1, \ldots, N\}$, and some random coins r (which are sometimes omitted). It outputs a session key K, and an encapsulation c of K;
- $\mathsf{Decaps}(\mathsf{usk}_i, c)$ takes as input a decryption key and a ciphertext c. It outputs the session key K, or the error symbol $\perp$.

For correctness, we require that for any c that encapsulates a key K for a target set S, if $i \in S$, then $\mathsf{Decaps}(\mathsf{usk}_i, c)$ outputs K. Then, semantic security and anonymity should be satisfied.

Definition E.5.2 [Semantic security] An anonymous broadcast encryption (ABE) scheme Π is said to be $(\tau, N, q_C, q_D, \varepsilon)$-IND-ACCA-secure (semantic security against adaptive corruption and chosen-ciphertext attacks) if in the security game presented in figure E.5, the advantage, denoted $\mathrm{Adv}^{\mathsf{ind-acca}}_{\Pi}(\kappa, \tau, N, q_C, q_D)$, of any τ-time adversary $\mathcal{A}$ corrupting at most q_C users (OCorrupt oracle), and asking for at most q_D decryption queries (ODecrypt oracle), is bounded by ε:

$$\mathrm{Adv}^{\mathsf{ind-acca}}_{\Pi}(\kappa, \tau, N, q_C, q_D) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{ind-acca}-1}_{\Pi,\mathcal{A}}(\kappa, N) = 1] - \Pr[\mathrm{Exp}^{\mathsf{ind-acca}-0}_{\Pi,\mathcal{A}}(\kappa, N) = 1]\}.$$

This definition includes IND-ACPA (for Chosen-Plaintext Attacks) when $q_D = 0$, and thus we denote the advantage $\mathrm{Adv}^{\mathsf{ind-acpa}}_{\Pi}(\kappa, \tau, N, q_C)$. When no corruption is allowed, we denote the advantage $\mathrm{Adv}^{\mathsf{ind-cpa}}_{\Pi}(\kappa, \tau, N)$.

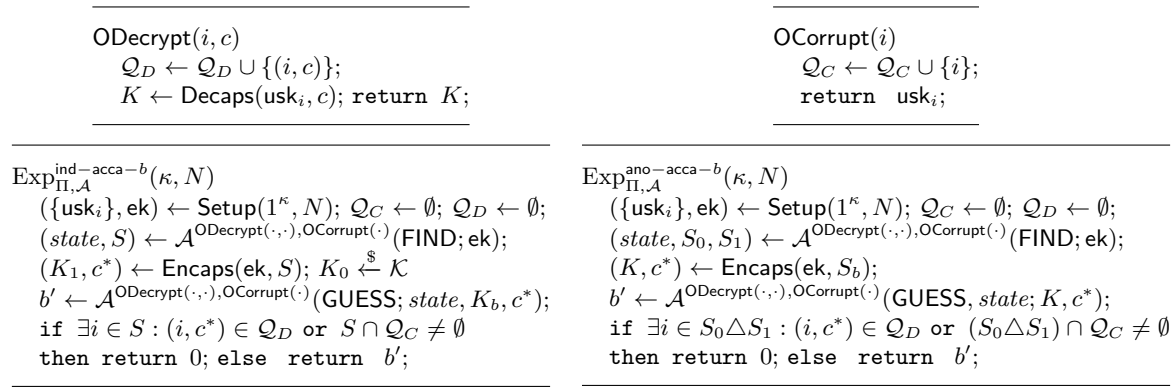


Figure E.5: Security games for ABE

Definition E.5.3 [Anonymity] An anonymous broadcast encryption (ABE) scheme Π is said to be $(\tau, N, q_C, q_D, \varepsilon)$-ANO-ACCA-secure (anonymity against adaptive corruption and chosen-ciphertext attacks) if in the security game presented in figure E.5, the advantage, denoted $\mathrm{Adv}^{\mathsf{ano-acca}}_{\Pi}(\kappa, \tau, N, q_C, q_D)$, of any τ-time adversary $\mathcal{A}$ corrupting at most q_C users (OCorrupt oracle), and asking for at most q_D decryption queries (ODecrypt oracle), is bounded by ε:

$$\mathrm{Adv}^{\mathsf{ano-acca}}_{\Pi}(\kappa, \tau, N, q_C, q_D) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{ano-acca-1}}_{\Pi,\mathcal{A}}(\kappa, N) = 1] - \Pr[\mathrm{Exp}^{\mathsf{ano-acca-0}}_{\Pi,\mathcal{A}}(\kappa, N) = 1]\}.$$

This definition includes ANO-ACPA (for Chosen-Plaintext Attacks) when $q_D = 0$, and thus we denote the advantage $\mathrm{Adv}^{\mathsf{ano-acpa}}_{\Pi}(k, \tau, N, q_C)$. When no corruption is allowed, we denote the advantage $\mathrm{Adv}^{\mathsf{ano-cpa}}_{\Pi}(\kappa, \tau, N)$.

Appendix F

Traitors Collaborating in Public: Pirates 2.0

EUROCRYPT 2009
[BP09] with Olivier Billet

Abstract : *This work introduces a new concept of attack against traitor tracing schemes. We call attacks of this type Pirates 2.0 attacks as they result from traitors collaborating together in a public way. In other words, traitors do not secretly collude but display part of their secret keys in a public place; pirate decoders are then built from this public information. The distinguishing property of Pirates 2.0 attacks is that traitors only contribute partial information about their secret key material which suffices to produce (possibly imperfect) pirate decoders while allowing them to remain anonymous. The side-effect is that traitors can publish their contributed information without the risk of being traced; giving such strong incentives to some of the legitimate users to become traitors allows coalitions to attain very large sizes that were deemed unrealistic in some previously considered models of coalitions.*

This paper proposes a generic model for this new threat, that we use to assess the security of some of the most famous traitor tracing schemes. We exhibit several Pirates 2.0 attacks against these schemes, providing new theoretical insights with respect to their security. We also describe practical attacks against various instances of these schemes. Eventually, we discuss possible variations on the Pirates 2.0 theme.

F.1 Introduction

Traitor tracing is a cryptographic primitive introduced by Chor, Fiat, and Naor in [CFN94b] in the context of secure content distribution. This context covers for instance multimedia content rental, or broadcasting to a very large number of subscribers like in pay-TV systems, mass distribution of high value DVDs, or in web-based distribution of various multimedia contents. In all of these settings, the content is encrypted before its distribution in order to prevent illegal access which helps ensuring the revenues of the distributor. To decrypt the content, every legitimate user is provided with a decryption means, commonly called decoder. The main issue faced by the distributor is the construction and dissemination of unauthorized decoders, possibly creating a parallel market.

Hardware tamper resistant solutions are often too expensive compared to the price of the offered services. Furthermore, it would not prevent an organization from breaking into one box and extracting the necessary information to build and resell unauthorized decoders.

This is where traitor tracing schemes step into the game: the key material embedded in the decoders is diversified on a user basis. Thus, decoders are 'marked' with the identity of the user and traitor tracing allows the authority to trace a user that produced a pirate decoder. Such users, called traitors, are more powerful when they collude to create a pirate decoder. In this case, traitor tracing should allow the tracing of at least one of the traitors that took part in the coalition. A trivial solution to the problem of traitor tracing is to provide every user with a randomly chosen key that identifies him and encrypt the content as many times as there are users in the system. Obviously, such a solution is totally impracticable due to bandwidth restrictions. Hence, bandwidth preservation in traitor tracing schemes is of crucial importance.

Since the seminal work of Chor, Fiat, and Naor, there have been several proposals and improvements in traitor tracing schemes. We give a few landmarks of the work in traitor tracing but this list is of course not exhaustive. Boneh and Franklin exposed an elegant algebraic construction coming with a deterministic tracing procedure in [BF99b]. Fiat and Tassa proposed a way to dynamically remove traitors from the system once they are caught, see [FT99]. Kiayias and Yung gave a powerful method to turn black-box tracing against stateless decoders into black-box tracing against stateful decoders in [KY02b]. In [BSW06b], Boneh, Sahai, and Waters introduced a full collusion traitor tracing scheme with sub-linear ciphertext size and constant size private keys. Traitor tracing schemes based on codes have been much investigated since the seminal work of Boneh and Shaw [BS95]: Kiayias and Yung [KY02c] proposed a scheme with constant rate, [CPP05a, Pfi96] relaxed the assumption that the tracer is a trusted third party, and [BP08, BN08b] recently achieved constant size ciphertexts. Among the most famous traitor tracing schemes are schemes from the NNL framework [NNL01] as they were used as a basis to design the widely spread content protection system for HD-DVDs and Blu-ray disks called AACS [AACa]. These are not exactly traitor tracing schemes, but rather very efficient broadcast encryption schemes with some black-box tracing abilities.

Pirates 2.0 attacks are primarily targeted to code based schemes and schemes from the NNL framework, but might be used against other combinatoric schemes.

F.1.1 Collaborative Traitors: Pirates 2.0

From the point of view of the attack model for traitor tracing schemes, there has been no radical change since the introduction of the concept in [CFN94b]. One remarkable exception is Pirate Evolution from [KP07] which exposes a new threat against trace and revoke schemes such as [NNL01]. In this paper, we introduce another new threat that we call Pirates 2.0 against both traitor tracing schemes and the trace and revoke schemes from [NNL01].

The main characteristics of our new Pirates 2.0 threat are as follows:

Anonymity Guarantee: Traitors that participate in a Pirates 2.0 attack are provided with a guarantee (through the exhibition of a mathematical proof) that *they cannot be traced by the authority.*

Partial Contributions: Traitors never need to reveal their whole secret key.

Public Collusions: Traitors operate in a public environment: they publish secret data from their decoders.

Large Coalitions: Traitors take part in unusually large coalitions.

Dynamic Coalitions: Traitors can come into action only when necessary.

The anonymity guarantee together with considerations on imperfect decoders makes the basis of our attack scenario and everything else heavily relies on it. The anonymity guarantee indeed gives strong incentives to potential traitors to actually take the plunge: With ubiquitous access to the Internet, leaking secret data, say, in a peer-to-peer network without further action can be done very quickly and in a straightforward way. This makes it an appealing scenario among the ever growing [Ele, Def, Sto] set of users hostile to the currently deployed Digital Rights Management systems (DRM). The characteristic that large coalitions can easily be achieved is therefore a direct consequence of the fact that traitors are *guaranteed not to be traced* by an authority.

Considerations on imperfect decoders are the other determinant ingredient: A pirate decoder is considered to be useful if it can decrypt (resp. decrypt with a high probability) valid ciphertexts; such a pirate decoder is called perfect (resp. imperfect) decoder. In previous work, it is assumed that a pirate decoder always decrypts ciphertexts from the tracer when it is not able to detect the presence of the tracing procedure, i.e. it is assumed that the pirate decoder is either perfect or only slightly imperfect. This assumption makes sense in the classical model of coalitions since any coalition, knowing at least one legitimate key, is able to decrypt all valid ciphertexts anyway. However, in the Pirates 2.0 setting, we show that another trade-off is possible for the pirates when the scheme uses variable length ciphertexts: the pirate decoder is only required to decrypt ciphertexts reasonably shaped. As an example of this scenario, consider the NNL scheme where the expansion of valid ciphertexts can vary a lot: A pirate decoder that can decrypt ciphertexts of size lower than, say 1 GB, is *highly* imperfect, but still useful to pirates.

In this paper, we show that some traitor tracing schemes and trace and revoke schemes (including the NNL scheme from [NNL01] and code based schemes) are susceptible to Pirates 2.0 attacks. We give several practical attacks against various instances of such schemes, most notably against the AACS. We then derive the theoretical implications for all these traitor tracing schemes.

F.1.2 Comparing Pirates 2.0 and the Classical Setting

We summarize below the main differences between our new attack model and the classical one:

Motivation: The classical model for coalitions captures the fact that pirates might invest some amount of money in order to sell unauthorized decoder to the black market. In the case of Pirates 2.0, the motivation might be to get rid of a protection system to which a large number of users are hostile. In the history of DVDs for instance, the main motivation to crack the system came from compatibility issues: the protection was thought to be too restrictive.

Static vs. Adaptive: The classical model of pirates is static. The coalitions consist of randomly chosen decoders. Therefore it is not possible to bias the collection process. In a Pirates 2.0 attack, traitors are able to contribute information adaptively, that is, depending on the current state of affair at the moment of the contribution. Therefore, even if during the publication process each traitor operates isolated (i.e. without communication with the other traitors), having access to published information at the time of the contribution makes it a collaborative process.

Anonymity: In the classical model of coalitions, traitors colluding must trust each other, or at least, one third party (say the pirate who collects the secret data). In contrast, the Pirates 2.0 attack only requires that the partial secret information provided by the traitors guarantees their anonymity.

Size of Coalitions: In the classical models, one usually assumes a small number of traitors (especially for combinatorial schemes like those relying on codes or those based on trees). This assumption seems reasonable in the classical model because each traitor must trust a third party and even in the case of an isolated traitor, getting a large number of decoder legally might be very expensive. In Pirates 2.0, this assumption becomes wrong, since the traitors guaranteed to remain anonymous can form a very large coalition.

F.2 Formalization of Pirates 2.0

There are many possible settings for a Pirates 2.0 attack. For instance, the construction of a pirate decoder can be active or passive. In the active case, the contributions made by the traitors are driven by the pirate upon building the pirate decoder. In the passive case, the traitors contribute information at their discretion. In this work, we focus on the last of these scenarios which leaves more freedom to the traitors and makes the attack even more realistic.

Also, there are two possible ways of collecting the information contributed by the traitors: in a centralized way or in a distributed way. Again, the distributed way leads to a stronger attack with less constraints in practice: traitors can easily use peer-to-peer networks to contribute their information, whereas a centralized server is more susceptible to shut down by legal action. We therefore choose to focus on the distributed setting, though in some cases, assuming a centralized entity like a pirate server would render the work of contributing for the traitors and of building a pirate decoder easier than in a peer-to-peer network. In the rest of the paper, we point out where it is relevant to use the facilities that a pirate server would provide.

F.2.1 A Setting for Pirates 2.0

We now describe several concepts that we use in the Pirates 2.0 setting:

Traitors and Pirates. As usual, a traitor is a legitimate user in possession of some secret data that we call his secret key and who leaks part of this secret key. Pirates are not legitimate users: they are not entitled to secret data but are able to collect relevant information from their public environment in order to produce a pirate decoder. We naturally assume that pirates and traitors respectively collect and contribute information in a stateful way: a traitor keeps track of (all) the information he contributed to the public, whereas both pirates and traitors can keep track of all of the information that was contributed to the public.

Contributed Information. The contributed information is the sum of information that was put into the public domain by the traitors at a given point in time, i.e. the secret data leaked from the system. The current contributed information at any point in time is denoted by $\mathcal{C}$. Initially, $\mathcal{C} = \emptyset$.

Traitor's Strategy. A traitor's strategy is a publicly available probabilistic algorithm $\mathsf{Contribute}$ that traitors execute to provide information to pirates. A traitor's strategy comes with a certificate that information leaked following this strategy allows the traitors to preserve some anonymity level. Traitors might in principle use different strategies, but for simplicity we only consider in the following the case where all traitors implement the same strategy.

The strategy $\mathsf{Contribute}$, takes as input the traitor's secret key sk, some information I already contributed by other traitors (for instance the set $\mathcal{C}$ of all contributed information at the time $\mathsf{Contribute}$ is run) as well as the history H of the contributions made by the traitor. The traitor's strategy returns $\mathsf{Contribute}(sk, I, H)$ as the traitor's contribution to the public. (And therefore, the overall information contributed to the public $\mathcal{C}$ is accordingly updated: $\mathcal{C} \leftarrow \mathcal{C} \cup \mathsf{Contribute}(sk, I, H)$.)

Public Information. The public information $\mathcal{P}$ consists of all the public data available

from the broadcaster (such as for instance its public key, the public key of users if any, etc.) together with the contributed information $\mathcal{C}$.

Anonymity Level. The public procedure Anonymity provides the level of anonymity $\mathsf{Anonymity}(sk, S, \mathcal{P})$ of a traitor with the secret key sk who leaked an information S (which corresponds to the sum of all his contributions) following a public strategy (we refine this notion later on by using extraction functions). The anonymity level output by the procedure corresponds to the uncertainty on the traitor's identity from the tracing authority point of view when provided with the sequence of contributed information S. At level 1 the traitor is known, while at level N, the traitor is undistinguishable from another user.

Pirate Decoder. We think of a pirate decoder as the output of an algorithm called Pirate. If the amount of information available from $\mathcal{P}$ is large enough, Pirate produces a pirate decoder $\mathsf{Pirate}(\mathcal{P})$ and simply outputs '`failed`' otherwise.

In the following we assume that the contribution of secret data to the public domain $\mathcal{C}$ by the traitors is a discrete process.

Definition F.2.1 [Security against Pirates 2.0] A traitor tracing scheme is said to be $\alpha-$secure against Pirates 2.0 if it prevents the construction of pirate decoders from information published by traitors with an anonymity level greater than α.

Note that not all traitor tracing (or trace and revoke) schemes are susceptible to Pirates 2.0 attacks. On the other hand, even fully collusion resistant schemes might be at risk as Pirates 2.0 attacks allow highly imperfect decoders: decoder can refuse to decrypt classes of specific ciphertexts—e.g. depending on their size. As we will show in the next sections, some of the most famous schemes, including the one used in the AACS, are susceptible to our new attack strategy.

F.2.2 A Concrete Treatment of Anonymity Estimation

The basic idea behind Pirates 2.0 attacks is that traitors are free to contribute some piece of secret data as long as several users of the system could have contributed exactly the same information *following the same (public) strategy*: this way, they are able to remain somewhat anonymous. The anonymity level is meant to measure exactly how anonymous they remain.

Definition F.2.2 [Extraction Function] An extraction function is an efficiently computable function f that outputs information about the secret key.

Definition F.2.3 [Masked Traitor] A traitor t is said to be masked by a user u for an extraction function f if $f(sk_u) = f(sk_t)$.

This notion of a traitor being masked by another user in the system is the basic undistinguishability notion that allows us to estimate the level of anonymity of a traitor after his contribution:

Definition F.2.4 [Anonymity Level] The level of anonymity of a traitor t after a contribution $\cup_{1\leq i\leq n} f_i(sk_t)$ is defined as the number α of users masking t for each of the n extraction functions f_i simultaneously:

$$\alpha = \#\{u \mid \forall i,\ f_i(sk_t) = f_i(sk_u)\}\ .$$

In the previous definitions, we use the equality between each extraction function f_i to derive the anonymity level. One can wonder why not simply consider equality between the *global* information leaked by a traitor and the global information another user u could extract like

$\cup_i f_i(sk_t) = \cup_j g_j(sk_u)$ with any set of extraction functions $\{g_j\}$. The answer is that we do not want to keep the traitor strategy secret and therefore, the authority can, at least from a theoretical point of view, use its knowledge of the set of extraction functions $\{f_i\}$ used by the traitors to gain additional information and to trace the traitors. (It might well be that there exists another user u such that $\cup_i f_i(sk_t) = \cup_j g_j(sk_u)$ holds, but $\cup_i f_i(sk_t) = \cup_i f_i(sk_v)$ would have been impossible for any user v other than t.)

F.3 Pirates 2.0 and the Subset-Cover Framework

The subset-cover framework proposed by Naor, Naor, and Lotspiech in [NNL01] is a powerful tool to design efficient trace and revoke systems. It captures many previously proposed traitor tracing systems and forms the basis of the so called NNL scheme used in the content protection system for HD-DVDs known as AACS [AACa]. However, as we show in this section, this scheme is susceptible to our attack and we explain how to defeat the AACS system.

F.3.1 Brief Description of the Subset-Cover Framework

The subset-cover framework is a powerful means to capture several trace and revoke designs. It encompasses several traitor tracing schemes proposed to date and maybe even more importantly, serves as the basis for two of the most efficient trace and revoke schemes: the complete subtree scheme and the subset difference scheme.

In the subset-cover framework, the set $\mathtt{N}$ of users in the system is covered by a collection of subsets $\mathtt{S}_i$ such that $\cup_i \mathtt{S}_i \supset \mathtt{N}$ and $\mathtt{S}_i \cap \mathtt{N} \neq \emptyset$. This covering is not a partition of $\mathtt{N}$ and the sets $\mathtt{S}_i$ rather overlap. To every subset $\mathtt{S}_i$ corresponds a long term secret key L_i, and every user that belongs to $\mathtt{S}_i$ is provided with this secret key—or in an equivalent way, with some material that allows him to derive this secret key. Therefore, every user u of the system is given a collection of long term keys $\{L_{i_k}\}$ that together form his secret key which we denote by sk_u.

In order to broadcast some content M, the center uses a standard hybrid scheme: a session key K is first drawn randomly and used to encrypt (with an encryption scheme E') the content, before being encrypted under multiple long term keys (with another encryption scheme E). The long term keys L_{i_k}, $k = 1, \ldots, l$ are chosen so that the corresponding subsets $\mathtt{S}_{i_1}, \ldots, \mathtt{S}_{i_l}$ only cover the set of users entitled to decrypt. Therefore, the center broadcasts ciphertexts of the form:

$$\Big[(i_1, E_{L_{i_1}}(K)),\ (i_2, E_{L_{i_2}}(K)),\ \ldots,\ (i_l, E_{L_{i_l}}(K)) \parallel E'_K(M) \Big]$$

To decrypt, a valid decoder for user u performs the following sequence of operations: It first looks for an index i_j in the first element of each of the l couples $(i_k, E_{i_k}(K))$ in turn such that $\mathtt{S}_{i_j} \subset sk_u$. If no index correspond, the decoder does not decrypt; otherwise, the decoder retrieves the corresponding long term key L_{i_j} and uses it to decrypt the associated encrypted session key $E_{i_j}(K)$ and then decrypts the payload $E'_K(M)$.

Since the system is built to handle revoked users, let us also denote by $\mathtt{R}$ the set of revoked users in the system at any point in time. In order to prevent them (independently, but also working together as a coalition) from accessing the encrypted content $E'_K(M)$, the collection $\mathtt{S}_{i_1}, \ldots, \mathtt{S}_{i_l}$ is specially crafted so that:

$$\bigcup_{k=1}^{l} \mathtt{S}_{i_k} = \mathtt{N} \setminus \mathtt{R} \ .$$

The tracing procedure. Now that we showed how the system deals with revoked users, we have to describe the way it disables pirate decoders. As is usual, the tracing procedure works

with black-box access to the pirate decoder only. The idea is to refine the covering initially used to broadcast ciphertexts so that the pirate decoder cannot decrypt with probability p higher than some threshold. To this end, the authors of [NNL01] suggest to use an hybrid argument: the pirate box is provided with "ciphertexts" with payload $E'_K(M)$ and headers of type j (for $j = 1, \ldots, l$):

$$(i_1, E_{L_{i_1}}(R)), \ldots, (i_j, E_{L_{i_j}}(R)), (i_{j+1}, E_{L_{i_{j+1}}}(K)), \ldots, (i_l, E_{L_{i_l}}(K))$$

where R is some randomly chosen element independent from K. If we denote by p_j the probability that the pirate box correctly decrypts the specially crafted ciphertexts of type j, there must exist an index t such that $|p_t - p_{t-1}| \geq \frac{p}{l}$ and therefore some traitor belongs to $\mathtt{S}_{i_t}$. The tracer then iterates this basic procedure, applying it to an arbitrary covering of $\mathtt{S}_{i_t}$ until either $\mathtt{S}_{i_t}$ contains a single element (which thus matches a traitor) or the pirate box cannot decrypt above the threshold (and no one is accused of being a traitor, but the new partition renders the pirate box useless).

The authors of [NNL01] showed that this tracing procedure is correct as soon as the revocation scheme satisfies a so-called "bifurcation property": every subset can be split into two subsets of roughly the same size. As we will see, this is the case for the two schemes *complete subtree* and *subset difference*.

F.3.2 General Attack Strategy against Subset-Cover Schemes

The generic process for the attack is relatively simple and runs in a few steps:

Elaborating the strategy

The main idea is to select a collection of subsets $\mathtt{S}_{\iota_1}, \ldots, \mathtt{S}_{\iota_w}$ such that:

- The number of users in each subset $\mathtt{S}_{\iota_k}$ is large, so that the anonymity level of the traitors is guaranteed to remain high enough when they contribute the associated long term key L_{ι_k};
- For any set $\mathtt{R}$ of revoked users and any method used by the broadcaster to partition $\mathtt{N}\backslash\mathtt{R}$ into subsets $\mathtt{S}_{i_1}, \ldots, \mathtt{S}_{i_m}$, the probability that one of the subsets $\mathtt{S}_{\iota_k}$ belongs to the partition $\mathtt{S}_{i_1}, \ldots, \mathtt{S}_{i_m}$ is high—say exceeds a given threshold τ—or the broadcaster exceeds its available bandwidth.

Contributing data

Let us define the extraction functions f_i to be $f_i(sk) = L_i$ if $L_i \in sk$ and '`missing`' otherwise. To contribute part of his private key sk_t, a traitor t performs the following sequence of lookups: for each index i from $\{\iota_1, \iota_2, \ldots, \iota_w\}$ (taken in any order) the traitor computes $C = f_i(sk_t)$ and if $C \neq \texttt{failed}$ and $C \notin \mathcal{P}$ returns and outputs C. The information H about sk_t that the traitor already contributed to the public is included in the argument list so that the contribution is $\mathsf{Contribute}(sk_t, \mathcal{P}, H)$.

Building pirate decoders

A pirate decoder simply embeds the public keys $L_{\iota_1}, \ldots, L_{\iota_w}$. Upon reception of a ciphertext

$$\Big[(i_1, E_{L_{i_1}}(K)), (i_2, E_{L_{i_2}}(K)), \ldots, (i_l, E_{L_{i_l}}(K)) \parallel E'_K(M) \Big]$$

from the center, the pirate checks whether $\{\iota_1, \ldots, \iota_w\} \cap \{i_1, \ldots, i_m\} = \emptyset$. If not, that is if there is an index $\iota_k = i_l$ in both sets (which was assumed to occur with high probability), the pirate box recovers the corresponding key L_{ι_k}, uses it to decrypt the session key K from $E_{L_{i_l}}(K)$, and therefore is able to correctly decrypt the payload.

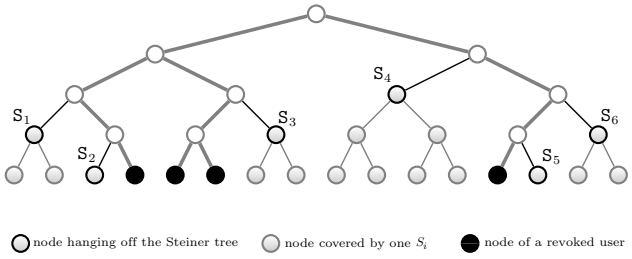


Figure F.1: Complete subtree: leaves correspond to users, $\mathtt{S}_1$, ..., $\mathtt{S}_6$ is the covering that excludes revoked users in black while allowing other users to decrypt derived from the Steiner tree associated to the set of revoked users $\mathtt{R}$.

Anonymity

The level of anonymity of a given traitor t in a subset cover scheme is related to the number of users of the system that know the complete list of subsets $\mathtt{S}_{t_1}$, ..., $\mathtt{S}_{t_l}$ for which the traitor contributed the keys L_{t_1}, ..., L_{t_l} to the public.

F.3.3 Pirates 2.0 against the Complete Subtree Scheme

The complete subtree scheme. In this scheme, the users correspond to the leaves of a complete binary tree whereas the collection of subsets $\mathtt{S}_i$ exactly corresponds to all the possible subtrees in the complete tree. When $|\mathtt{N}| = 2^n$, the complete binary tree is of length n and there are exactly n subtrees that contain a given leaf. Figure F.1 shows a covering using six subsets of twelve users that excludes four revoked users (depicted in black). This subset scheme complies with the bifurcation property since any subset (or equivalently any subtree of the complete binary tree) can be split into two subsets of equal size (the two subtrees rooted at the two children of the root of the original subtree). Regarding key assignment, each user represented by a leaf u in the complete binary tree is provided with the keys L_i associated to the nodes i on the path from the leaf u to the root.

Covering algorithm. In the case of the complete subtree, the covering used to exclude the $r = |\mathtt{R}|$ revoked users from $\mathtt{N}$ is the collection of subsets that hang off the Steiner tree of the revoked leaves. (The Steiner tree of the revoked leaves is the minimal subtree of the complete binary tree that connects all the revoked leaves to the root and it is unique.) Since any user only knows the keys from its leaf to the root and since this path is included in the Steiner tree for revoked users, these users cannot decrypt anymore. This algorithm produces covers of size $O(r \log(N/r))$.

We now give a version of our attack against subset cover schemes in the case of the complete subtree scheme:

Theorem F.3.1 On average, a randomly chosen group of $\rho \log \rho$ traitors (operating isolated) is able to mount a Pirates 2.0 attack against a complete subtree scheme in which the center wants to ensure a ciphertext rate[1] of at most $\rho(N - r)/N$. Moreover, each traitor is guaranteed an anonymity level of N/ρ.

Proof: For simplicity we assume that no collision occurs during the contribution process (the traitors contribute sequentially, although in a completely random way, their share of secret data) and that the contribution of a traitor is readily available to the public. (It is obviously possible

[1] the ciphertext rate is the number of subsets used by the center

to deal with these refinements by considering statistical processes instead and then bounding the loss in efficiency that would occur in such a general case.)

Following the general attack strategy described in the previous section, define $\mathtt{S}_{\iota_1}, \ldots, \mathtt{S}_{\iota_w}$ to be the subsets corresponding to all the subtrees of the complete tree having more than N/ρ leaves so that for each ι_k more than N/ρ users share the corresponding long term keys L_{ι_k}. These subsets also correspond to all the nodes between level 0 (the root) and the level $\lambda = \lfloor \log \rho \rfloor$ and thus, there are $w = 2\lfloor \rho \rfloor$ of them. Then, a traitor contributing one of the L_{ι_k} at level λ together with every L_{ι_j} on the path from node ι_k to the root has a level of anonymity[2] higher than N/ρ. (As mentioned above, more than N/ρ users share the key L_{ι_k} and moreover the same users also know about L_{ι_i} for every node ι_i on the path from node ι_k to the root because of the assignment scheme.) Now, the number of traitors needed to collect the $\lfloor \rho \rfloor$ long term keys (and those above) is given by the answer to the classical coupon collection problem: to collect all the m possible items when one receives a uniformly chosen item at each draw requires $m \log m$ draws on average. This demonstrates the first part of the theorem.

It only remains to show that either a pirate is able to produce a working decoder, or the center uses too much bandwidth (the ciphertext rate is bigger than ρ). Let r be the number of revoked users. Let us assume that the broadcaster only uses subsets rooted at a level $l \geq \lambda$ since otherwise the priate decoder is able to decrypt the ciphertexts. Now every subset can cover at most $N/2^{\lambda}$ users so that $\rho(N-r)/N$ of them are needed to cover the $N-r$ legitimate users.▮

Theoretical and practical impact. From a theoretical point of view, Theorem F.3.1 shows that instead of the $O(r \log(N/r))$ complexity that was first derived, the bandwidth required for the complete subtree scheme to operate securely actually is $O(\rho(N-r)/N + r \log(N/r))$ for a number of $\rho \log \rho$ traitors taking part in a Pirates 2.0 attack.

From a practical point of view, we note that we assumed that every long term key can be leaked by at least one traitor. For a system accommodating 2^{32} users and a long term key at the 12th level, this assumption translates into the fact that among a million of users there is at least one that takes the step of contributing it to the public (with the guarantee of remaining anonymous!); this hypothesis seems reasonable to us.

Also, note that even in the case where one long term key is not contributed by any user, the attack remains valid: the pirate box will not be able to decrypt only with a very small probability.

F.3.4 Pirates 2.0 against the Subset Difference Scheme

The subset difference scheme has been introduced to lower the number of subsets required to partition the set of legitimate users $\mathtt{N} \setminus \mathtt{R}$. It improves on the complete subtree scheme exposed above by a factor of $\log(N/r)$ in terms of bandwidth usage for the headers.

To attain this level of performance, the number of possible subsets has been tremendously increased. Remember that $\mathtt{S}_i$ denotes the full binary subtree of the complete binary tree rooted at node i. Now, for each node j in $\mathtt{S}_i$ different from i, let us denote by $\mathtt{S}_{i,j}$ the binary subtree rooted at node i of which the full binary subtree rooted at node j has been removed. (See examples in Figure F.3.) A user will need to know all the keys $L_{i,j}$ such that he belongs to the subtree rooted at i but not in the subtree rooted at j. However, it would be impossible for each device to store such a huge number of long term keys. This is why a key derivation procedure has been designed to allow the derivation of most of the $O(N)$ long term keys a

[2] having a lot of revoked users in the subtree does not affect the level of anonymity: revoked users know the keys on their path to the root and could have contributed them as well. This, however, affects the decryption threshold of the pirate decoder

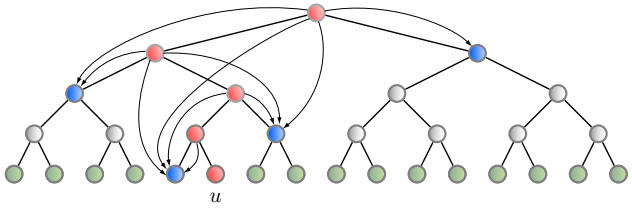


Figure F.2: Key assignment. User u receives all the labels $\mathsf{LABEL}_{i,j}$ such that i is a parent of j and i is on the path from the leaf of u to the root.

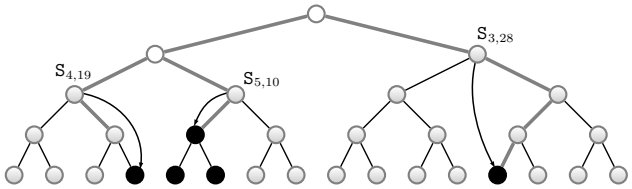


Figure F.3: Subset difference: leaves correspond to users and black nodes are not able to derive the necessary information to decrypt. Therefore $\mathtt{S}_{4,19}$ prevents user 19 from decrypting, $\mathtt{S}_{5,10}$ prevents users 20 and 21 from decrypting, and $\mathtt{S}_{3,28}$ prevents user 28 from decrypting. All other users are able to decrypt.

user is entitled from a much smaller set of keys: a user only needs to store $O(\log^2(N))$ keys. Each node i in the full binary tree is first assigned a random label LABEL_i and labels $\mathsf{LABEL}_{i,j}$ together with their corresponding long term keys $L_{i,j}$ are deduced (in a pseudo-random way) from label LABEL_i. The key derivation procedure then works as follows: from each LABEL_i, a pseudo-random value $\mathsf{LABEL}_{i,j}$ is obtained for each sub-node j using the tree based construction proposed by Goldreich, Goldwasser, and Micali [GGM84]; from this value, a long term key $L_{i,j}$ is eventually deduced (in a pseudo-random way). Each user is then provided with labels $\mathsf{LABEL}_{i,j}$ for all nodes i that are on the path from the leaf that represents the user to the root, and all nodes j hanging off this path as described on Fig. F.2. This key assignment ensures that every user in the subtree rooted at node i but not in the subtree rooted at node j is able to derive $L_{i,j}$ while every user in the subtree rooted at node j is not able to derive $L_{i,j}$.

Covering algorithm. The covering algorithm works by maintaining a subtree T of the Steiner tree of $\mathtt{R}$ and removes nodes from it at each steps:

`init:` Make T the Steiner tree of $\mathtt{R}$.

`select:` If there is only one leaf v_k in T and it is not the root (or node 0), add the subset $\mathtt{S}_{0,k}$ and return. If there is only the root in T, return. Otherwise, select two leaves v_{j_1} and v_{j_2} from T so that their least common ancestor v does not contain any other leave of T than v_{j_1} and v_{j_2}. Call v_{i_1} and v_{i_2} the children of v such that v_{i_1} is the ancestor of v_{j_1} and v_{i_2} the ancestor of v_{j_2}. Then, if $v_{i_1} \neq v_{j_1}$ add $\mathtt{S}_{i_1,j_1}$ to the partition and similarly if $v_{i_2} \neq v_{j_2}$ add $\mathtt{S}_{i_2,j_2}$ to the partition. Remove all the descendants of v from T, which makes v a leaf of T. Reiterate the step 'select'.

An example output of this procedure is shown in Figure F.3.

Theorem F.3.2 On average, a randomly chosen group of $\rho \log \rho$ traitors (operating isolated) is able to mount a Pirates 2.0 attack against a subset difference scheme in which the center wants to ensure a ciphertext rate of at most $\rho(N-r)/N$. Moreover, each traitor is guaranteed a level of anonymity of at least $N/2\rho$.

Proof:

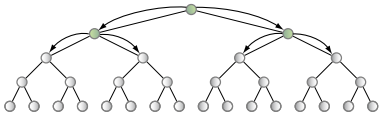


Figure F.4: Direct labels

In the following proof we make use of labels of a special type, that we call *direct labels*. Direct labels are $\mathsf{LABEL}_{i,j}$ such that the node j is a *direct* descendant of the node i. The first six direct labels of the tree are described in the figure F.4.

First, note that a pirate knowing all the keys $L_{i,j}$ where the node i lies in the first $\lambda = \lfloor \log \frac{\rho}{2} \rfloor$ levels, is able to decrypt all the ciphertexts where the rate is lower than $\rho(N-r)/N$ where r is the number of revoked users. Indeed, the broadcaster must use subsets $S_{k,l}$ where the node k does not lie in the first λ levels in order to prevent the pirate from decrypting the ciphertexts. Since each of these subsets covers less than $N/2^{\lambda+1}$ users (those who are in the subtree rooted at node k), the center must use at least $\rho(N-r)/N$ subsets to cover all the legitimate users.

Collecting all the keys rooted at a level $l \leq \lambda$ is however totally unpractical since there are a tremendous number of such keys. The pirate can nevertheless go around this difficulty by collecting labels $\mathsf{LABEL}_{i,j}$ instead of keys $L_{i,j}$ and using the derivation procedure to lower the minimum information to be kept: the labels that users possess allow to derive a large number of keys. Therefore, we claim that it is enough for the pirate to collect all *direct* labels $\mathsf{LABEL}_{i,j}$ where i is located in the first λ levels in order to derive all keys $L_{i,k}$. (Once the pirate knows the two direct labels at node i, he can derive all keys $L_{i,k}$ where k is in the subtree rooted at i.)

To prove the theorem, we show that on average, $\rho \log \rho$ randomly chosen traitors are able to contribute all the direct labels of the first λ levels. Each traitor contributes all his direct labels $\mathsf{LABEL}_{i,j}$ for the nodes i located in the first λ levels. Note that at each level, a traitor has been assigned exactly one of the direct labels. Thus, when all direct labels at level exactly λ have been contributed, so have the direct labels of all the first $\lambda - 1$ levels. As a randomly chosen traitor knows a uniformly chosen direct label out of the $\frac{\rho}{2}$ direct labels of level λ, a randomly chosen group of $\rho \log \rho$ traitors (operating isolated) is able to contribute all direct labels $\mathsf{LABEL}_{i,j}$ where i is located in the first λ levels.

Moreover, such traitors share their contribution with every user in the same subtree rooted at level $\lambda + 1$: each traitor is covered by N/ρ users. ∎

Remark F.3.3 Theorem F.3.2 is proven in the case of static attacks: traitors submit information non-adaptively, such as in a peer-to-peer scenario. However, the number of required traitors to mount a Pirate 2.0 attack can be lowered to ρ in the case of an adaptive attack such as in a server-based scenario.

Impact on AACS. In the case of AACS, the subset difference scheme is used with $N = 2^{31}$ users. The header is written in a so-called Media Key Block or MKB for short which (among other) encodes the indices for the difference subsets as well as the media key encrypted once for each of the corresponding long term keys. These keys are 16 bytes long and the indices are encoded using 5 bytes. According to Section 3.2.5.5 of AACS specifications [AACb]: "*For the purposes of designing performance, a one megabyte buffer is sufficient to process the MKB.*" Although this is not an intrinsic limitation of the system, very large MKBs would decrease the performances of hardware devices and would increase their price. This is why applications like disk replicators often only allocate 1MB space for the MKB. In the case of AACS, this means that only $2^{11.6} = 2^{20}/21$ encrypted keys will be able to fit this space and thus a Pirates 2.0 attack

against the AACS would only require some thousand collaborating traitors which, given the guarantee offered to traitors (a million of other users cover each traitor), seems very practicable.

Also note that once again the attack given here is just an illustration of our general concept of attack. There are several possible improvements and refinements such as taking advantage of the partition algorithm (remember that the scheme is a trace and revoke scheme and not a full traitor tracing scheme, so that it might fail to single out a traitor).

F.4 Pirates 2.0 and Code Based Schemes

Traitor tracing schemes based on codes (be it collusion secure codes [BS95, Tar03] or identifiable parent property codes [HvLLT98, SS01]) have been proposed during more than half a decade [KY02c, CPP05a, PSNT06b, Pha06, Sir07a, BN08b]. Their main advantage is their efficiency in terms of bandwidth requirements, but their main drawback is that their efficiency (in terms of the size of the private key) is highly sensitive to the number of traitors in the coalition.

F.4.1 General Framework of Codes Based Schemes

Traitor tracing schemes built on codes more or less fit in the following framework:

Setup: The scheme generates a code $\mathcal{C}$ of length ℓ which is either a collusion secure code or a q-ary c-IPP code. The alphabet for the code is $\mathtt{A} = \{0, 1\}$ in the case of a collusion secure code and $\mathtt{A} = \{1, \ldots, q\}$ in the case of an IPP code. Then, for each position $i = 1, \ldots, \ell$ in a codeword and for each possible letter a from $\mathtt{A}$, a key $K_{i,a}$ is randomly chosen. Hence, there are 2ℓ possible keys (resp. $q\ell$ possible keys) in the system in the case of collusion secure codes (resp. IPP codes).

Key assignment: Each user u is given a codeword W_u from $\mathcal{C}$. Then, for each position $i = 1, \ldots, \ell$ in this codeword, the user is provided with the key $K_{i,W_u[i]}$ where $W_u[i]$ is the letter at position i in the codeword W_u. Thus, each user gets ℓ keys in its decoder.

Decoder: A ciphertext usually contains a header that specifies the positions of the keys involved in the decryption process. For instance, in the case of the scheme [KY02c] proposed by Kiayias and Yung, all the keys of the user are involved. In the case of the scheme [BN08b] proposed by Boneh and Naor only one key is involved during a decryption process.

F.4.2 Pirates 2.0 against Code Based Schemes

Our goal is to show how our generic attack can be applied to this class of schemes. We do not focus on any concrete construction but rather deal with the underlying codewords. For ease of exposition, we describe an attack when the underlying code is a Tardos' code [Tar03] but this attack might easily translate to other codes.

First, recall that a Tardos' code secure against coalitions of size at most c is built as follows. First, the code length is set to be $\ell = \lceil 100c^2 \log(N/\epsilon) \rceil$. Then, for each integer i in the interval $[1, \ldots, \ell]$ a (secret) value $0 < p_i < 1$ heavily biased towards 0 or 1 is randomly drawn. Then, any of the N codewords is constructed by randomly choosing for each position i in $[1, \ldots, \ell]$ the bit '0' or the bit '1' according to the probability p_i.

Theorem F.4.1 For any traitor tracing scheme that relies on Tardos' code for its set of keys, a set of T traitors collaborating to mount a Pirates 2.0 attack allows to produce a pirate decoder while maintaining a level of anonymity higher than $N \cdot 2^{-\ell/T}$ on the average.

Proof: Since contributing large amounts of a codeword makes your level of anonymity drop a lot, a strategy that handles every traitor with equity is to make them contribute the same amount of secret data. Since there are T traitors, let them each contribute ℓ/T elements of (the secret data associated with) their codeword. Of course, people are then already able to construct pirate decoders with the collected material. The anonymity level α a traitor can expect is easy to assess: if $m = \lceil \ell/T \rceil$,

$$\alpha \quad = N \prod_{i=1}^{m} \left(p_{\sigma(i)}^2 + (1 - p_{\sigma(i)})^2 \right) \quad . \tag{F.1}$$

Indeed, for a randomly chosen traitor, there is a probability p_i that the letter at position i is '0' and for any other codeword randomly chosen a probability p_i that the letter at that position is also '0'. Similarly there is a probability $1 - p_i$ that the letter at position i is '1' and the same probability that another codeword gets the same letter at that position. Therefore, the probability that another codeword gets the same letter as that of the traitor for some position i is $q_i = p_i^2 + (1-p_i)^2$. The probability that a block of size m of the traitor's codeword is the same as that of another user is thus $\prod_{i=1}^{m} q_{\sigma(i)}$, where σ is a permutation of $\{1, \ldots, \ell\}$ that accounts for the particular selection of the block of size m.

The sum from Eq. (F.1) takes into account every possible block of codeword of length m and by multiplying by the total number of users in the system, we get the average number of users masking a randomly chosen traitor, that is its level of anonymity in the system. Now since $p_i^2 + (1 - p_i)^2 \geq \frac{1}{2}$ we get a (very loose) bound on the level of anonymity: $\alpha \geq N \cdot 2^{-\ell/T}$. ▌

Theoretical and practical impact. From a theoretical point of view, the above theorem shows that the number of traitors required to mount a Pirates 2.0 is only *linear in the size of the decoder* and only logarithmic in the number of users in the system. From a practical point of view, it would require about 2^{17} traitors to mount a Pirate 2.0 attack against a traitor tracing scheme that relies on a 30-collusion secure code with 2^{32} users. Each traitor would be masked by about a few thousand users in this case.

F.5 Conclusion

Throughout this paper we presented a novel concept of attack against combinatorial traitor tracing schemes. We focused on the main ideas behind this concept of attack, but some variations could be further investigated. For instance, it is possible to consider the case of dishonest traitors (a common threat to collaborative work is bad contributions which have to be tracked and eliminated). Dishonest traitors capture the fact that the authority could try to perturb the creation of pirate decoders by publishing incorrect information. However, one of the traitors might use its own authorized decoder to verify the contribution of the other traitors: after having sorted out these contributions, he is able to produce a pirate decoder.

Another direction is to consider probabilistic guarantees for the level of anonymity of contributing traitors: the traitors are only certified to have a high level of anonymity with some (possibly very high) probability. This is useful if the authority tries to embed markers specific to a single user. However, there is a trade-off for the authority between the effectiveness of this process against Pirates 2.0 and the efficiency of the scheme.

Eventually, the most interesting direction is probably to provide modified versions of the common traitor tracing schemes that resist Pirates 2.0 attacks without sacrificing the efficiency of the original schemes.

Appendix G

Identity-Based Traitor Tracing

PKC 2007

[ADML+07b] with M. Abdalla, A. W. Dent, J. Malone-Lee, G. Neven, and N. P. Smart

Abstract : *We present the first identity-based traitor tracing scheme. The scheme is shown to be secure in the standard model, assuming the bilinear decision Diffie-Hellman (DBDH) is hard in the asymmetric bilinear pairing setting, and that the DDH assumption holds in the group defining the first coordinate of the asymmetric pairing. Our traitor tracing system allows adaptive pirates to be traced. The scheme makes use of a two level identity-based encryption scheme with wildcards (WIBE) based on Waters' identity-based encryption scheme.*

G.1 Introduction

In 1984 Shamir proposed the concept of identity-based cryptography [Sha84]. However, it took nearly twenty years for the problem of designing an efficient method to implement identity-based encryption (IBE) to be solved. In 2000 and 2001 respectively Sakai, Ohgishi and Kasahara [SOK00] and Boneh and Franklin [BF01] proposed IBE schemes based on elliptic curve pairings. Also, in 2001 Cocks proposed a system based on the quadratic residuosity problem [Coc01].

Identity-based encryption is often justified as a useful technology by its possible use in an e-mail application. However, many people, whilst having a small set of e-mail identities, often belong to a larger set of e-mail groups. An e-mail group, or shared address, is an e-mail address which allows the sender to send a message to a large number of individual e-mail addresses without needing to know the actual individual addresses. Using existing identity-based encryption techniques one can easily implement such a scheme by giving each member of the e-mail group the same ID-private key. Thus all members of the group will share the same private key.

A common business model in PKI world is that the certificate authority charges for each certificate, or block of certificates, issued. In the ID-based world this model corresponds to the trust authority charging for each private key, or block of private keys. However, in our group e-mail example this would mean that the trust authority would only be able to charge for one private key for the whole group, since as soon as one person had the private key they could

share it with the other members of the group. What is needed is a disincentive for the group members to collaborate in this manner.

A similar situation occurs in the traditional symmetric or public key setting in broadcast encryption. Here one solves the associated problem by using a traitor tracing scheme, which allows any person (or set of colluding people) who creates a new decryption device, or key, to be traced. Thus combining the above ideas together we see that there is a possible need for an identity-based traitor tracing scheme.

Surprisingly since the invention of identity-based cryptography by Shamir [Sha84] in 1984, no one seems to have considered this issue. Thus in this paper we present the first identity-based traitor tracing scheme. Our scheme is based on the Waters' WIBE from [ACD+06], which is based on Waters' identity-based encryption scheme [Wat05]. A WIBE is a variant of a hierarchical IBE (HIBE) scheme in that it encrypts to an identity string which is defined on various layers. However, unlike a HIBE, which allows only a single recipient, a WIBE allows one to encrypt to a string which is "wildcarded" on a given set of levels. A WIBE allows one to target a ciphertext at a given group of users by applying the appropriate wildcards.

Our construction is relatively simple: we use a two level WIBE in which the first level represents the name of the group and the second level represents the unique index of a user. This allows e-mails to be addressed to the entire group via the use of a wildcard in the second level. Group membership is 'policed' by the trust authority, which only releases a decryption key to a user if the user is entitled to decrypt messages sent to a particular group. The subtlety of our construction is in the construction of a traitor tracing algorithm.

We prove that our scheme protects the confidentiality of encrypted messages against passive attackers in the standard model, and show that it allows traitor tracing against an adaptive traitor.

Unfortunately, our scheme is not practical due to the combination of Waters' IBE and collusion secure codes [BS95], which results in infeasibly large public key and ciphertext sizes. Thus we leave the construction of a truly efficient identity-based traitor tracing scheme, even in the random oracle model [BR93], as an open problem. In addition we leave as open the problem of creating a scheme which allows a greater number of key extraction queries by the pirate than ours allows. Furthermore, our scheme does not protect against pirate decoder manufacturers mounting chosen-ciphertext attacks, however this later stronger pirate has not been considered in the public-key setting either.

G.2 Preliminaries

G.2.1 Notation

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ be the set of natural numbers and $\{0,1\}^*$ the set of all bit strings. If $k \in \mathbb{N}$ then $\{0,1\}^k$ is the set of bit strings of length k and 1^k is the string of k ones. If $\mathcal{A}$ is a randomized algorithm, then $y \xleftarrow{\$} \mathcal{A}^O(x)$ denotes the assignment to y of the output of $\mathcal{A}$ when run on input x with fresh random coins and with access to oracle O; we write $y \leftarrow \mathcal{A}^O(x)$ if $\mathcal{A}$ is deterministic. If S is a finite set, then $x \xleftarrow{\$} S$ denotes the random generation of an element $x \in S$ using the uniform distribution. A function $\nu : \mathbb{N} \to [0,1]$ is said to be *negligible* if for all $c \in \mathbb{N}$ there exists a $k_c \in \mathbb{N}$ such that $\nu(k) < k^{-c}$ for all $k > k_c$. It is said to be *non-negligible* if there exists a $c \in \mathbb{N}$ such that $\nu(k) > k^{-c}$ for all $k \in \mathbb{N}$.

G.2.2 Computational Assumptions

Our scheme employs asymmetric pairings, which we now recall. Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ denote three finite multiplicative abelian groups of prime order $p > 2^k$. Let g and h be generators of

$\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and let $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ be an efficiently computable isomorphism such that $\psi(h) = g$. We assume that there exists an admissible bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, meaning that for all $a, b \in \mathbb{Z}_p$ (1) $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$, (2) $\hat{e}(g^a, h^b) = 1$ iff $a = 0$ or $b = 0$, and (3) $\hat{e}(g^a, h^b)$ is efficiently computable.

The advantage of an algorithm $\mathcal{A}$ in solving the *computational bilinear Diffie–Hellman (CBDH)* problem in $\mathbb{G}_2$ is defined as

$$\mathrm{Adv}^{\mathrm{cbdh}}_{\mathcal{A},\mathbb{G}_2}(k) \;=\; \Pr Z = \hat{e}(g,h)^{xyz} \;:\; x, y, z \xleftarrow{\$} \mathbb{Z}_p \;;\; Z \xleftarrow{\$} \mathcal{A}(h^x, h^y, h^z) \;.$$

The advantage of $\mathcal{A}$ in solving the decisional variant of this problem, called the *decisional bilinear Diffie–Hellman (DBDH)* problem in $\mathbb{G}_2$, is

$$\begin{aligned}\mathrm{Adv}^{\mathrm{dbdh}}_{\mathcal{A},\mathbb{G}_2}(k) \;=\; \Big| & \Pr\mathcal{A}(h^x, h^y, h^z, Z) = 1 \;:\; x, y, z \xleftarrow{\$} \mathbb{Z}_p \;;\; Z \leftarrow \hat{e}(g,h)^{xyz} \\ & - \Pr\mathcal{A}(h^x, h^y, h^z, Z) = 1 \;:\; x, y, z \xleftarrow{\$} \mathbb{Z}_p \;;\; Z \xleftarrow{\$} \mathbb{G}_T \Big| \;.\end{aligned}$$

We say that the CBDH and DBDH problems in $\mathbb{G}_2$ are *hard* if the respective advantages are negligible functions in k for all algorithms $\mathcal{A}$ with running time polynomial in k.

We also require that the DDH problem in $\mathbb{G}_1$ is hard, namely we require that for all algorithms $\mathcal{A}$, with running time polynomial in k, the following advantage is a negligible function in k,

$$\begin{aligned}\mathrm{Adv}^{\mathrm{xddh}}_{\mathcal{A},\mathbb{G}_1}(k) \;=\; \Big| & \Pr\mathcal{A}(g^x, g^y, Z) = 1 \;:\; x, y \xleftarrow{\$} \mathbb{Z}_p \;;\; Z \leftarrow g^{xy} \\ & - \Pr\mathcal{A}(g^x, g^y, Z) = 1 \;:\; x, y \xleftarrow{\$} \mathbb{Z}_p \;;\; Z \xleftarrow{\$} \mathbb{G}_1 \Big| \;.\end{aligned}$$

Note that if the DDH problem in $\mathbb{G}_1$ is hard, then there cannot exist a computable isomorphism from $\mathbb{G}_1$ to $\mathbb{G}_2$ and thus we must be working in the asymmetric pairing setting. The assumption that the DDH problem is hard in $\mathbb{G}_1$ is referred to as the external DDH problem (XDDH) and has been used before in [BGdMM05, BBS04, Sco02].

G.3 Identity-Based Traitor Tracing

G.3.1 Syntax

In this section we will describe the general model for an identity-based traitor tracing scheme. Broadcast groups are referred to by an identity string $ID \in \{0,1\}^*$, individual users are referred to by an index $i \in \mathbb{N}$. To make user i member of the group ID, the trusted key distribution centre provides it with a personal decryption key $d_{ID,i}$. Anyone can encrypt a message to the general group ID such that all individual users belonging to the group can recover the message.

Formally, an identity-based traitor tracing scheme $\mathcal{IBTT}$ consists of five polynomial-time algorithms:

- A randomised key generation algorithm $\mathcal{G}(1^k)$ taking as input the security parameter k. This algorithm generates a set of domain parameters consisting of a master public key mpk and a master secret key msk.
- A key extraction algorithm $\mathcal{X}(\mathsf{msk}, ID, i)$ which given the master secret key msk, a group identity $ID \in \{0,1\}^*$ and a user index i generates a user secret key $d_{ID,i}$. This algorithm could be probabilistic.

- A probabilistic encryption algorithm $\mathcal{E}(\mathsf{mpk}, ID, \mathcal{M})$ which on input of the master public key mpk, a group identity ID and a message $\mathcal{M}$ outputs a ciphertext C.
- A decryption algorithm $\mathcal{D}(d_{ID,i}, C)$ which on input of a user secret key $d_{ID,i}$ and a ciphertext C outputs a plaintext message $\mathcal{M}$, or $\perp$ to indicate a decryption error.
- A traitor tracing algorithm $\mathcal{T}^{\mathbb{D}}(\mathsf{msk}, ID)$ which has oracle access to a "pirate" decryption box $\mathbb{D}$. The tracing algorithm takes as input the master secret key msk and a group identity ID, and outputs a set of user identifiers (called "traitors") $T \subset \mathbb{N}$.

An identity-based traitor tracing scheme whose tracing algorithm takes as input mpk instead of msk is said to be *publicly-traceable*, since then anyone can execute the tracing algorithm. We shall assume that all "pirate" decryption boxes are resettable [KY02b], meaning that they retain no state between decryptions. In particular, pirate boxes cannot self-destruct.

For correctness we require that $\mathcal{D}(d, \mathcal{E}(\mathsf{mpk}, ID, \mathcal{M})) = \mathcal{M}$ with probability one for all $k \in \mathbb{N}$, $ID, \mathcal{M} \in \{0,1\}^*$, $i \in \mathbb{N}$, $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathcal{G}(1^k)$ and $d \xleftarrow{\$} \mathcal{X}(\mathsf{msk}, ID, i)$.

G.3.2 Secrecy

We require that our ID-based traitor tracing scheme is semantically secure in the presence of adaptive adversaries who have access to a key extraction oracle and, in a chosen-ciphertext attack, a decryption oracle. These are standard notions in ID-based cryptography first introduced in [BF01]. The extension to the setting we have here is immediate, but for completeness we clarify it here.

Secrecy is defined by a two-stage game. The challenger first runs the key generation algorithm to generate a master key pair $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathcal{G}(1^k)$. The master public key mpk is passed to the adversary. In the first stage of the game the adversary has access to a key extraction oracle $\mathcal{X}(\mathsf{msk}, \cdot, \cdot)$, which it can query on arbitrary pairs (ID, i) of group identities ID and user indices i. In a chosen-ciphertext attack, the adversary can also has access to a decryption oracle $\mathcal{D}(\mathcal{X}(\mathsf{msk}, \cdot, \cdot), \cdot)$ from which it can obtain the decryption of any ciphertext C using the key to any pair (ID, i). The first stage ends when the adversary outputs two messages of equal length $\mathcal{M}_0$ and $\mathcal{M}_1$, plus a challenge group identity ID^*.

The challenger then selects a bit b and encrypts $\mathcal{M}_b$ under the group identity ID^* to form the challenge ciphertext $C^* \leftarrow \mathcal{E}(\mathsf{mpk}, ID^*, \mathcal{M}_b)$. The challenge ciphertext is returned to the adversary for the second stage of the game. In this second stage the adversary can perform further queries to its oracles. At the end of the second stage the adversary outputs its guess b' as to the bit b. The adversary wins the game if $b = b'$, if ID^* never appeared in any of the key extraction oracle queries, and, in a chosen-ciphertext attack, if C^* was never submitted to the decryption oracle with group identity ID^*.

The advantage $\mathrm{Adv}^{\text{ind-id-cpa}}_{\mathcal{A},\mathcal{IBTT}}(k)$, respectively $\mathrm{Adv}^{\text{ind-id-cca}}_{\mathcal{A},\mathcal{IBTT}}(k)$, of an adversary $\mathcal{A}$ in breaking the indistinguishability of scheme $\mathcal{IBTT}$ is defined as the probability of $\mathcal{A}$ winning the corresponding game minus one-half. We say that the traitor tracing scheme is IND-ID-CPA, respectively IND-ID-CCA secure, if this advantage is a negligible function in k for any adversary $\mathcal{A}$ with running time polynomial in k.

G.3.3 Traceability

We extend the notion of traceability defined for the public key setting in [BSW06b] to the identity-based setting. We provide definitions for both chosen-plaintext and chosen-ciphertext attack; our scheme however is only proved secure in the chosen-plaintext setting. We note that

to our knowledge there is no public-key traitor tracing system which has been considered in the presence of (the natural analogue of) chosen-ciphertext attacks against the traceability property.

Let $k, c \in \mathbb{N}$ be two security parameters associated to the experiment. The challenger first generates a master key pair $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathcal{G}(1^k)$ and gives mpk to the adversary. The adversary has access to a key extraction oracle $\mathcal{X}(\mathsf{msk}, \cdot, \cdot)$ to which it can submit pairs (ID, i) of its choosing. In a chosen-ciphertext attack, it can also perform queries to a decryption oracle $\mathcal{D}(\mathcal{X}(\mathsf{msk}, \cdot, \cdot), \cdot)$ specifying a group identity ID, a user index i and an arbitrary ciphertext C as in the above secrecy game. The adversary terminates by outputting a group identity ID^* and a pirate decoder $\mathbb{D}$, which is the description of a probabilistic circuit that takes as input ciphertexts and outputs messages. The challenger then runs the tracing algorithm with black-box access to $\mathbb{D}$ to obtain a set of user identifiers $S \xleftarrow{\$} \mathcal{T}^{\mathbb{D}}(\mathsf{msk}, ID^*)$.

By modelling the pirate decoder as a probabilistic circuit, we assume that the decoder is *resettable* or *stateless* [KY02b] in that it does not retain information from previous decryptions, and in particular that it cannot self-destruct. Thus, when being subjected to a series of tracing queries, the pirate decoder responds to each query as if it were the first.

If we let T denote the set of user indices i that the adversary submitted to the key extraction oracle in combination with the group identity ID^*, then we say that the adversary wins the game if the following conditions hold:

- The decryption box decrypts a non-negligible fraction of random ciphertexts encrypted under the group identity ID^*, i.e. for random messages $\mathcal{M}$ we have that $\Pr[\mathbb{D}(\mathcal{E}(\mathsf{mpk}, ID^*, \mathcal{M})) = \mathcal{M}] \geq \delta(k)$ where $\delta(k)$ is a non-negligible function and where the probability is taken over the random choice of $\mathcal{M}$ and over the random coins of the encryption algorithm $\mathcal{E}$ and the pirate box $\mathbb{D}$.
- Either $S = \emptyset$ or $S \not\subseteq T$.
- $\mathcal{A}$ queried the key extraction oracle for at most c different user indices i. We do not restrict the number of different group identities ID for which $\mathcal{A}$ can obtain keys for each of these users (apart from being polynomial in k of course). This reflects that colluding users can use all their decryption keys to construct the pirate box, not just the key corresponding to ID^*. It also means that the number of different groups a single user subscribes to is *not* limited by c.
- In the chosen-ciphertext variant there are no restrictions on $\mathcal{A}$'s queries to the decryption oracle.

The advantage $\mathrm{Adv}^{\text{tra-id-cpa[c]}}_{\mathcal{A},\mathcal{IBTT}}(k)$, respectively $\mathrm{Adv}^{\text{tra-id-cca[c]}}_{\mathcal{A},\mathcal{IBTT}}(k)$, of $\mathcal{A}$ in breaking the traceability of the scheme $\mathcal{IBTT}$ is defined as its probability of winning the above game. We say that $\mathcal{IBTT}$ is c-TRA-ID-CPA, respectively c-TRA-ID-CCA secure, if this advantage is a negligible function in k for all adversaries $\mathcal{A}$ with running time polynomial in k.

The above definition is essentially a *full access* model. One can, following [BF99b] and [BSW06b], define a *minimal access* model in which the oracle available to the tracing algorithm only outputs whether the decoder successfully decrypted the input ciphertext or not, but does not give it the resulting plaintext.

G.4 The Scheme

Our scheme makes use of the two-level WIBE scheme [ACD+06] based on Waters' HIBE scheme [Wat05]. We assume that group identities ID are given by strings of length n_1. As user identifiers we associate to each user an element of a code. The mapping between individual

users, their indices and their codewords is maintained by the trust authority. In practice the code will be a (c, N, ϵ)-collusion secure code [BS95], where N is the maximum number users in the system, c is the maximum number of colluders our tracing algorithm can tolerate, and ϵ is the probability of error that a colluder is not traced. A (c, N, ϵ) collusion secure code can be produced using codewords of size $\ell = O(c^2(\log(N) + \log(1/\epsilon)))$ over an alphabet of size $s = 2$ [Tar03]. Our use of collusion secure codes will result in a scheme which is not publicly traceable, since the tracing algorithm for collusion secure codes requires secret randomness.

Before giving a more precise definition of collusion-secure codes, we need to introduce some additional notation. Let Σ be a symbol alphabet of size $|\Sigma| = s$. If $x = x_1 \dots x_\ell \in \Sigma^\ell$ is a string of ℓ symbols and $I = \{1 \le i_1 < \dots < i_n \le \ell\}$ is a set of indices, then $x|_I$ is the substring $x_{i_1} \dots x_{i_n}$ containing only those symbols of x at positions in I. Let $W = \{w_1, \dots, w_c \in \Sigma^\ell\}$ be a set of symbol strings, and let I be the set of all positions where all strings in W are equal, i.e. I is the maximal set such that $w_1|_I = w_2|_I = \dots = w_c|_I$. Then the *feasible set* of W is defined as the set of all strings that are equal to $w_1, \dots, w_c$ at positions in I, i.e.

$$\mathrm{FS}(W) \;=\; \{x \in \Sigma^\ell \;:\; x|_I = w_1|_I = \dots = w_c|_I\} \;.$$

A (c, N, ϵ) collusion-secure code of length ℓ over alphabet Σ consists of a set $\mathbb{C}$, called the *codebook*, of indexed codewords $w_r^{(i)}$ for $1 \le i \le N$ and $r \in \{0,1\}^\rho$, and a *tracing algorithm* $\mathcal{T}_\mathbb{C}$. These are such that for all collusions $C \subseteq \{1, \dots, N\}$ of size at most c, $W = \{w_r^{(i)} : i \in C\}$, and for all (unbounded) algorithms $\mathcal{A}$ it holds that

$$\Pr \mathcal{T}_\mathbb{C}(x, r) \in C \mid x \in \mathrm{FS}(W) x \xleftarrow{\$} \mathcal{A}(W) r \xleftarrow{\$} \{0,1\}^\rho \;>\; 1 - \epsilon \;,$$

where the probability is taken over the choice of r and the random coins of $\mathcal{T}_\mathbb{C}$ and $\mathcal{A}$. Our scheme uses codewords as "identity strings". This presents a small problem: the definition insists that the set C is chosen *before* $\mathcal{A}$'s execution; whereas, we will allow the adversary to chose the set C adaptively via key extraction queries. We solve this problem by introducing a randomly chosen permutation on $\{1, 2, \dots, N\}$, denoted $\pi \xleftarrow{\$} \mathit{Perm}(N)$ (or if it is desired for efficiency a pseudo-random permutation). We associate the codeword $w_r^{(\pi(i))}$ with the i-th user. It is therefore sufficient that

$$\Pr \mathcal{T}_\mathbb{C}(x, r) \in C \;\Big|\; \begin{array}{c} x \in \mathrm{FS}(W) x \xleftarrow{\$} \mathcal{A}(W) \\ C \xleftarrow{\$} \mathcal{P}(\mathbb{C}, c, r) r \xleftarrow{\$} \{0,1\}^\rho \end{array} \;>\; 1 - \epsilon \;,$$

where $\mathcal{P}(\mathbb{C}, c, r)$ is the set of subsets of $\{w_r \in \mathbb{C}\}$ of size c.

For non-binary alphabets, we use the natural encoding of symbols as bit strings of length $\lceil \log_2 s \rceil$, so that codewords are represented by bit strings of length $n_2 = \lceil \log_2 s \rceil \cdot \ell$.

To set up the scheme we define two sets V_1 and V_2 of random elements in $\mathbb{G}_2$, denoted by $V_i = (v_{i,0}, v_{i,1}, \dots, v_{i,n_i})$. We let $u_{i,j} \leftarrow \psi(v_{i,j})$ and let U_i denote the image of the set V_i under the isomorphism ψ, i.e. $U_i = (u_{i,0}, u_{i,1}, \dots, u_{i,n_i})$. For a bit string B of length n_i we use these sets to define the so-called Waters' hash functions

$$H_i(B) \leftarrow v_{i,0} \prod_{j \in B} v_{i,j},$$

where the product is computed over all values of j for which the j-th bit of B is one. To simplify notation we define

$$G_i(B) \leftarrow u_{i,0} \prod_{j \in B} u_{i,j} = \psi(H_i(B)).$$

Note that $G_i(B)$ can be computed either from the set V_i using the isomorphism ψ, or from the set U_i directly. Also note that $v_{i,j} = h^{\kappa_{i,j}}$ and $u_{i,j} = g^{\kappa_{i,j}}$ for some, unknown, values $\kappa_{i,j} \in \mathbb{Z}_p$.

Our ID-based traitor tracing scheme can now be defined via the following algorithms:

Setup $\mathcal{G}(1^k)$ **:** The key distribution centre generates a set of pairing groups $\mathbb{G}_1, \mathbb{G}_2$ as above at the security level k, along with the sets $V_i \xleftarrow{\$} (\mathbb{G}_2^*)^{n_i}$ for $i = 1, 2$. A random value $\alpha \xleftarrow{\$} \mathbb{Z}_p$ is selected, and one sets $g_1 \leftarrow g^\alpha \in \mathbb{G}_1$ and $h_1 \leftarrow h^\alpha \in \mathbb{G}_2$. We require a second random element $h_2 \xleftarrow{\$} \mathbb{G}_2^*$ and we let $g_2 \leftarrow \psi(h_2)$. Finally, the secret random permutation $\pi \xleftarrow{\$} \mathit{Perm}(N)$ and the secret randomness $r \xleftarrow{\$} \{0,1\}^\rho$ for the code $\mathbb{C}$ is chosen. The master public key is defined to be $\mathsf{mpk} = (g, g_1, h_2, U_1, U_2)$ and the master secret key is $\mathsf{msk} = (h, h_2^\alpha, V_1, V_2, \pi, r)$.

Key Extraction $\mathcal{X}(\mathsf{msk}, ID, i)$ **:** Let id be the codeword corresponding to index i, i.e. the bit string of length $n_2 = \lceil \log_2 s \rceil \cdot \ell$ that is the binary encoding of codeword $w_r^{(\pi(i))}$. The key distribution centre first select random values $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$ and then define the private key as

$$d_{ID,i} \;=\; (\mathsf{id}, a_0, a_1, a_2) \;\leftarrow\; (\mathsf{id}\ ,\ h_2^\alpha H_1(ID)^{r_1} H_2(\mathsf{id})^{r_2}\ ,\ h^{r_1}\ ,\ h^{r_2})$$

Encryption $\mathcal{E}(\mathsf{mpk}, ID, \mathcal{M})$ **:** A message is defined as an element in $\mathbb{G}_\mathrm{T}$. The sender first chooses a $t \xleftarrow{\$} \mathbb{Z}_p$ and then computes the ciphertext $C = (C_1, C_2, C_3, C_4) \in \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_T \times \mathbb{G}_1^{n_2+1}$ as

$$C_1 \leftarrow g^t\ ,\quad C_2 \leftarrow G_1(ID)^t\ ,\quad C_3 \leftarrow \mathcal{M} \cdot \hat{e}(g_1, h_2)^t\ ,\quad C_4 \leftarrow (u_{2,j}^t)_{j=0,\ldots,n_2}\ .$$

Decryption $\mathcal{D}(d_{ID,i}, C)$ **:** Decryption works as follows, on input of C we first compute

$$C_2' \;\leftarrow\; C_4^{(0)} \cdot \prod_{j \in \mathsf{id}} C_4^{(j)} = G_2(\mathsf{id})^t\ ,$$

where the last equality follows since $C_4^{(j)} = u_{2,j}^t$. Then we compute

$$\begin{aligned}
&C_3 \cdot \frac{\hat{e}(C_2, a_1) \cdot \hat{e}(C_2', a_2)}{\hat{e}(C_1, a_0)} \\
&\quad= \quad \mathcal{M} \cdot \hat{e}(g_1, h_2)^t \cdot \frac{\hat{e}(G_1(ID)^t, h^{r_1}) \cdot \hat{e}(G_2(\mathsf{id})^t, h^{r_2})}{\hat{e}(g^t, h_2^\alpha H_1(ID)^{r_1} H_2(\mathsf{id})^{r_2})} \\
&\quad= \quad \mathcal{M} \cdot \hat{e}(g_1, h_2)^t \cdot \frac{\hat{e}(G_1(ID)^{r_1}, h^t) \cdot \hat{e}(G_2(\mathsf{id})^{r_2}, h^t)}{\hat{e}(g^t, h_2^\alpha) \cdot \hat{e}(g^t, H_1(ID)^{r_1} H_2(\mathsf{id})^{r_2})} \\
&\quad= \quad \mathcal{M} \cdot \frac{\hat{e}(g^\alpha, h_2)^t}{\hat{e}(g^t, h_2^\alpha)} \cdot \frac{\hat{e}\,(g^\sigma, h^t)}{\hat{e}\,(g^t, h^\sigma)} \\
&\qquad\qquad \text{where } \sigma = r_1(\kappa_{1,0} + \sum_{j \in ID} \kappa_{1,j}) + r_2(\kappa_{2,0} + \sum_{j \in \mathsf{id}} \kappa_{2,j}) \\
&\quad= \quad \mathcal{M} \cdot \frac{\hat{e}(g^\alpha, h_2)^t}{\hat{e}(g^t, h_2^\alpha)} \\
&\quad= \quad \mathcal{M}.
\end{aligned}$$

Traitor Tracing Algorithm $\mathcal{T}^{\mathbb{D}}(\mathsf{msk}, ID)$ **:** Since we use a collusion-secure code, the tracing step requires the secret randomness r, so tracing can only be done by the key distribution centre. The tracing algorithm has access to a pirate box $\mathbb{D}$ that correctly decrypts ciphertexts for ID with probability $\delta(k)$. For convenience, we let $C_4^{(i,j)}$ denote the $(\lceil \log_2 s \rceil (i-1)+j)$-th element of C_4. For each $1 \le i \le \ell$ and $1 \le j \le \lceil \log_2 s \rceil$, initialise counter $ctr_{i,j} \leftarrow 0$ and run the following test $n = 16k/\delta(k)$ times:

1. Choose a random message $\mathcal{M}$.
2. Encrypt $\mathcal{M}$ under the group identity ID to form a ciphertext
$$C \leftarrow (C_1, C_2, C_3, C_4).$$
3. Replace $C_4^{(i,j)}$ with a random element from $\mathbb{G}_1$.
4. Query the pirate decoder $\mathbb{D}$ on the altered ciphertext C.
5. If the decoder outputs the message $\mathcal{M}$ (or a valid ciphertext in the case of minimal access) then increase $ctr_{i,j}$.

After these iterations, reconstruct the bit string id' of length n_2 as follows. Let $\mathsf{id}'_{i,j}$ denote the bit of id' at position $\lceil \log_2 s \rceil (i-1) + j$. Set $\mathsf{id}'_{i,j} \leftarrow 1$ if $ctr_{i,j} < 4k$, or set $\mathsf{id}'_{i,j} \leftarrow 0$ otherwise. Next, decode the bit string id' as a symbol string x of length ℓ, choosing any symbol if the corresponding bit string is not a valid encoding of a symbol in Σ. Finally, use the tracing algorithm of the code to compute $S \xleftarrow{\$} \mathcal{T}_{\mathbb{C}}(x, r)$ and return the set of traitors $\pi^{-1}(S)$.

G.5 Security Results

The IND-ID-CPA security of our scheme under the DBDH assumption follows from the security of the Waters' HIBE from [Wat05] and an analogue of Theorem 6 of [ACD+06]. As one notices that the scheme is simply the Waters WIBE from [ACD+06] specialised to the 2-level case. In Appendix G.6 we outline the asymmetric version of Waters' HIBE scheme that we are using.

The scheme as it stands is only secure against adversaries who do not make decryption oracle queries. However, extending to chosen-ciphertext security can be done using the techniques described in [ACD+06] based on the techniques of Canetti, Halevi and Katz [CHK04]. This extension will not affect our traitor tracing algorithm given above.

We now turn to showing that our tracing algorithm works. Intuitively, for the $\mathcal{T}_{\mathbb{C}}$ algorithm to work (with error probability ϵ), we need the reconstructed symbol string x to fall within the feasible set of the codewords corresponding to the collusion. This means that on those positions where all the codewords in the collusion are the same, the symbols of x have to be the same as well. We prove that if the ciphertext component $C_4^{(i,j)}$ that is being "tampered" with corresponds to a bit position where all traitors' codewords have a zero, then the pirate box decrypts correctly, unless it can solve the DDH problem in $\mathbb{G}_1$. We also prove that if the tampered component corresponds to an all-one position, then the pirate box is unable to decrypt correctly, unless it can solve the CBDH problem. The $8k/\delta(k)$ iterations are needed because the pirate box only decrypts correctly with probability $\delta(k)$; we use a Chernoff bound to analyse the overall success probability of our tracing algorithm.

Theorem G.5.1 The $\mathcal{IBTT}$ scheme described above is c-TRA-ID-CPA secure under the assumptions that the underlying code is (c, N, ϵ) collusion-secure code of length ℓ over an alphabet of size s, that the DDH problem in $\mathbb{G}_1$ is hard, and that the CBDH problem in $\mathbb{G}_2$ is hard. More specifically, the advantage of any polynomial-time adversary $\mathcal{A}$ in building an untraceable decoder that correctly decrypts a fraction $\delta(k)$ of ciphertexts using the keys of a collusion of at most c users is at most
$$\mathrm{Adv}^{\text{tra-id-cpa[c]}}_{\mathcal{A},\mathcal{IBTT}}(k) \;\leq\; \epsilon + \ell\lceil \log_2 s \rceil \cdot \left(\mathrm{Adv}^{\text{cbdh}}_{\mathcal{B}_2,\mathbb{G}_2}(k) + e^{-k}\right)$$
whenever $\delta(k) \geq 2 \cdot \mathrm{Adv}^{\text{xddh}}_{\mathcal{B}_1,\mathbb{G}_1}(k)$ where $\mathcal{B}_1, \mathcal{B}_2$ are polynomial-time algorithms depending on $\mathcal{A}$ and e is the base of the natural logarithm.

Proof: Let $\mathcal{A}$ be an attacker against the tracing property of the encryption scheme; i.e. $\mathcal{A}$ takes as input mpk and outputs a pirate decryption box $\mathbb{D}$. We use $\mathcal{A}$ to define an attacker $\mathcal{A}'$ against the tracing property of the collusion-secure code; i.e. $\mathcal{A}'$ will take as input a collection of c random codewords $W = \{w_1, \ldots, w_c\}$ and output a value x. We will prove that if $\mathcal{A}$ successfully avoids being traced, then, with high probability, $\mathcal{A}'$ will successfully output a codeword x that cannot be traced. This will provide the required contradiction.

$\mathcal{A}'$ runs as follows. It chooses random unique indices $i_1, \ldots, i_c \in \{1, \ldots, N\}$ and mounts the following attack for the collusion $C = \{i_1, \ldots, i_c\}$. On input codewords $W = \{w_r^{(i_j)} : j = 1, \ldots, c\}$, it first generates a public key $\mathsf{mpk} \leftarrow (g, g_1, h_2, U_1, U_2)$ as described in the setup algorithm $\mathcal{G}$ of the identity-based traitor tracing scheme. $\mathcal{A}'$ then runs $\mathcal{A}$. $\mathcal{A}$ may query a key extraction oracle for identities (ID, i) for at most c values of i. $\mathcal{A}'$ responds to the j-th such query as normal using the codeword $w_r^{(i_j)}$. Since W contains codewords corresponding to a random collusion C, and π is meant to be a random permutation, this response is identically distributed to the response of a correct key extraction algorithm. $\mathcal{A}$ terminates by outputting a pirate decryption box $\mathbb{D}$.

$\mathcal{A}'$ then applies the identity-based traitor tracing scheme's tracing algorithm $\mathcal{T}^{\mathbb{D}}$ to $\mathbb{D}$, halting after $\mathcal{T}^{\mathbb{D}}$ determines the value of the symbol string x. $\mathcal{A}'$ outputs the value x. We prove that the symbol string $x \in \Sigma^\ell$ reconstructed by our tracing algorithm falls outside the feasible set $\mathrm{FS}(W)$ with probability at most

$$\Pr x \notin \mathrm{FS}(W) \;\leq\; \ell\lceil \log_2 s \rceil \cdot (\mathrm{Adv}^{\mathrm{cbdh}}_{\mathcal{B}_2,\mathbb{G}_2}(k) + e^{-k}) \;.$$

The theorem statement then directly follows from the properties of the (c, N, ϵ) collusion-secure code's tracing algorithm $\mathcal{T}_{\mathbb{C}}$.

Let $I \subseteq \{1, \ldots, \ell\}$ be the maximal set of symbol positions such that $w_r^{(i)}|_I = w_r^{(j)}|_I$ for all $i, j \in C$. For positions of x not in I there is nothing to prove, because they do not affect membership of $\mathrm{FS}(W)$. So we focus on the symbols x_i of x at positions $i \in I$. Let $\mathsf{id}_{i,j}$ for $i \in I$ and $1 \leq j \leq \lceil \log_2 s \rceil$ be the bits in the binary representation of codewords corresponding to symbols at positions $i \in I$. Because of the way we defined I, these bits are the same for all users in the coalition. For a single iteration in the tracing algorithm at position (i, j), the following lemmas upper-bound the probability that the decryption box correctly decrypts $\mathcal{M}$ in case $\mathsf{id}_{i,j} = 0$ and that it does not correctly decrypt $\mathcal{M}$ in case $\mathsf{id}_{i,j} = 1$. Hence, we can distinguish between bit positions which are all zeros and all ones. This means we can recover the symbols which are the same in all the codewords for which the attacker has the keys. If the bits in a given bit position are different in the attacker's codewords, then the attacker can detect the tracing attempt and may output whatever they like. However, this does not matter as we only need to recover the symbols which are the same for all codewords in order to apply the code's tracing algorithm. We postpone the proofs of these lemmas until after the proof of the theorem.

Lemma G.5.2 If $\mathsf{id}_{i,j} = 0$ in the codewords of all users in the collusion C, then $\mathbb{D}$ correctly decrypts a random ciphertext that has been tampered with at position (i, j) with probability

$$p_0 \;\geq\; \delta(k) - \mathrm{Adv}^{\mathrm{xddh}}_{\mathcal{B}_1,\mathbb{G}_1}(k) \;.$$

Lemma G.5.3 If $\mathsf{id}_{i,j} = 1$ in the codewords of all users in the collusion C, then $\mathbb{D}$ correctly decrypts a random ciphertext that has been tampered with at position (i, j) with probability

$$p_1 \;\leq\; \mathrm{Adv}^{\mathrm{cbdh}}_{\mathcal{B}_2,\mathbb{G}_2}(k) \;.$$

We also use the following adaptation of the Chernoff bound from [MR95].

Lemma G.5.4 Let $X_1, \ldots, X_n$ be independent, 0/1 valued random variables with expected value p. Let $X = X_1 + \ldots + X_n$, let $\mu = \mathbf{E}[X] = np$ and let $0 \leq \alpha \leq 1$ be a real number. Then we have

$$\Pr X < (1-\alpha)\mu \ < \ e^{-\mu\alpha^2/2} \ .$$

We want to upper-bound the probability that $x_i \neq w_i$. For a position i, j where $\mathsf{id}_{i,j} = 0$, we can see the final value of $ctr_{i,j}$ as the outcome of the sum of $n = 16k/\delta(k)$ independent 0/1 random variables with expected value $p = p_0$. The expected value of $ctr_{i,j}$ is $\mu = np_0$. From Lemma G.5.2 and the assumption that $\mathrm{Adv}^{\mathrm{xddh}}_{\mathcal{B}_1,\mathbb{G}_1}(k) \leq \delta(k)/2$, we know that

$$\mu = np_0 \ \geq \ n(\delta(k) - \mathrm{Adv}^{\mathrm{xddh}}_{\mathcal{B}_1,\mathbb{G}_1}(k)) \ \geq \ \frac{n\delta(k)}{2} = 8k \ .$$

We can then apply the Chernoff bound of Lemma G.5.4 with $\alpha = 1/2$ to upper-bound the probability that the tracing algorithm incorrectly decides that $\mathsf{id}'_{i,j} = 1$ by

$$\begin{aligned} \Pr ctr_{i,j} < 4k \quad &\leq \quad \Pr ctr_{i,j} < \mu/2 \\ &< \quad e^{-\mu/8} \\ &\leq \quad e^{-k} \ . \end{aligned}$$

On the other hand, for a position i, j where $\mathsf{id}_{i,j} = 1$, the probability that the tracing algorithm incorrectly decides that $\mathsf{id}'_{i,j} = 0$ can be upper-bounded by

$$\Pr ctr_{i,j} \geq 4k \ \leq \ \Pr ctr_{i,j} \geq 1 \ = \ p_1 \ \leq \ \mathrm{Adv}^{\mathrm{cbdh}}_{\mathcal{B}_2,\mathbb{G}_2}(k) \ .$$

The probability that $x_i \neq w_i$ is upper-bounded by the probability that the tracing algorithm makes an incorrect decision at any of the bit positions. Since there are $\lceil \log_2 s \rceil$ bits in the encoding of x_i, we have that

$$\Pr x_i \neq w_i \ \leq \ \lceil \log_2 s \rceil \cdot (\mathrm{Adv}^{\mathrm{cbdh}}_{\mathcal{B}_2,\mathbb{G}_2}(k) + e^{-k}) \ ,$$

so that the overall probability that the symbol string x reconstructed by the tracing algorithm is not within the feasible set of W is

$$\Pr x \notin \mathrm{FS}(W) \ \leq \ \ell\lceil \log_2 s \rceil \cdot (\mathrm{Adv}^{\mathrm{cbdh}}_{\mathcal{B}_2,\mathbb{G}_2}(k) + e^{-k}) \ ,$$

from which the theorem follows.▌

We have left to prove the two lemmas that we used above.

Proof of Lemma G.5.2: For the sake of contradiction, let $\mathcal{A}$ denote an adversary against the traitor tracing scheme that produces a decryption box that correctly decrypts random ciphertexts with probability $\delta(k)$, but that correctly decrypts ciphertexts that have been tampered with at position (i', j') with probability $p_0 \leq \delta(k) - \gamma$ for some $\gamma > 0$. We will construct an algorithm $\mathcal{B}_1$ which uses $\mathcal{A}$ to gain an advantage γ in solving the DDH problem in $\mathbb{G}_1$.

Let (g^x, g^y, Z) denote the input to our DDH algorithm $\mathcal{B}_1$ and let $k' = s(i'-1) + j' - 1$. It constructs the master public keys of the ID-based by choosing random exponents $\alpha, \kappa_{i,j} \xleftarrow{\$} \mathbb{Z}_p^*$ for $a = 1, 2$ and $b = 0, \ldots, n_a$ and a random element $h_2 \xleftarrow{\$} \mathbb{G}_2^*$. It sets $g_1 \leftarrow g^\alpha$, $h_1 \leftarrow h^\alpha$, $u_{i,j} \leftarrow g^{\kappa_{i,j}}$, $v_{i,j} \leftarrow h^{\kappa_{i,j}}$, except for $u_{2,k'}$ and $v_{2,k'}$ which it sets to $u_{2,k'} \leftarrow g^x$ and $v_{2,k'} \leftarrow \perp$, respectively. It also chooses secret randomness $r \xleftarrow{\$} \{0,1\}^\rho$ for the collusion-secure code.

$\mathcal{B}_1$ runs $\mathcal{A}$ on input $\mathsf{mpk} = (g, g_1, h_2, U_1 = (u_{1,0}, \ldots, u_{1,n_1}), U_2 = (u_{2,0}, \ldots, u_{2,n_2}))$, responding to its key extraction queries (ID, i) as follows. Let id be the encoding of the codeword $w_r^{(i)}$. We know from the preconditions of the lemma that $\mathsf{id}_{k'} = 0$. $\mathcal{B}_1$ chooses $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$ and computes the secret key $d_{ID,i} = (\mathsf{id}, a_0, a_1, a_2) = (\mathsf{id},\ h_2^{\alpha} H_1(ID)^{r_1} H_2(\mathsf{id})^{r_2},\ h^{r_1},\ h^{r_2})$. Note that because $\mathsf{id}_{k'} = 0$, $\mathcal{B}_1$ can compute $H_2(\mathsf{id})$, even though it does not know $v_{2,k'}$.

At the end of this stage $\mathcal{A}$ will output a pirate decoder $\mathbb{D}$ with respect to a group identity ID of its choice.

All the identities used to create the box $\mathbb{D}$ will have the k'-th bit of their binary code word id set to zero. Algorithm $\mathcal{B}$ then generates a random message $\mathcal{M}$ and forms the ciphertext

$$C_1 \leftarrow g^y\ , \qquad C_2 \leftarrow (g^y)^{\kappa_{1,0}} \cdot \textstyle\prod_{i \in ID} (g^y)^{\kappa_{1,i}}\ ,$$

$$C_3 \leftarrow \mathcal{M} \cdot \hat{e}(g^y, h_2)^{\alpha}\ , \qquad C_4^{(i)} \leftarrow \begin{cases} (g^y)^{\kappa_{2,i}} & \text{for } 0 \le i \le n_2,\ i \ne k'\ , \\ Z & \text{for } i = k'\ . \end{cases}$$

This ciphertext is then passed to the decoder $\mathbb{D}$. Algorithm $\mathcal{B}_1$ outputs 1 if the decoder correctly decrypts $\mathcal{M}$, or outputs 0 otherwise.

If $Z = g^{xy}$, then the ciphertext C is a correctly-formed random ciphertext, so $\mathbb{D}$ will correctly decrypt it with probability $\delta(k)$. If Z is random, then C looks exactly like a ciphertext that has been tampered with at position (i', j'), so $\mathbb{D}$ will correctly decrypt it with probability at most $\delta(k) - \gamma$. The advantage of an algorithm in solving the DDH problem is defined as the difference of the probability that it outputs 1 if $Z = g^{xy}$ and if Z is random, so for our algorithm $\mathcal{B}_1$ we have that

$$\mathrm{Adv}^{\mathrm{xddh}}_{\mathcal{B}_1, \mathbb{G}_1}(k) \ \ge\ \delta(k) - (\delta(k) - \gamma) \ =\ \gamma\ ,$$

from which the lemma follows. ∎

Proof of Lemma G.5.3: For the sake of contradiction, let $\mathcal{A}$ denote an adversary against the traitor tracing scheme that will produce a decryption box $\mathbb{D}$ that correctly decrypts ciphertexts that have been tampered with at position (i', j') with probability p_1. We will construct an algorithm $\mathcal{B}_2$ which uses $\mathcal{A}$ as a subroutine to solve the bilinear computational Diffie–Hellman problem.

Let h^x, h^y, h^z, be $\mathcal{B}_2$'s input for the CBDH problem. Algorithm $\mathcal{B}_2$ chooses random integers $\kappa_{i,j} \xleftarrow{\$} \mathbb{Z}_p$ for $i = 1, 2$ and $0 \le j \le n_i$. Let $k' = s(i'-1) + j' - 1$. It sets

$$g_1 \leftarrow \psi(h^x) \qquad h_2 = h^z$$
$$v_{i,j} \leftarrow h^{\kappa_{i,j}} \text{ and } u_{i,j} \leftarrow g^{\kappa_{i,j}} \text{ for } i = 1, 2 \text{ and } 0 \le j \le n_i$$

except for $u_{2,k'}$ and $v_{2,k'}$ which it sets to

$$v_{2,k'} \leftarrow h^{\kappa_{2,k'}} / h^x = h^{\kappa_{2,k'} - x} \qquad u_{2,k'} \leftarrow \psi(v_{2,k'}) = g^{\kappa_{2,k'} - x}\ .$$

It also chooses secret randomness $r \xleftarrow{\$} \{0,1\}^{\rho}$ for the collusion-secure code. It then runs $\mathcal{A}$ on input $\mathsf{mpk} = (g, g_1, h_2, (u_{1,0}, \ldots, u_{1,n_1}), (u_{2,0}, \ldots, u_{2,n_2}))$.

Algorithm $\mathcal{A}$ will make c key extraction queries (ID, i). Let id be the codeword corresponding to user i; we know from the preconditions of the lemma that $\mathsf{id}_{k'} = 1$ for all users in the collusion. The decryption key $d_{ID,i} = (\mathsf{id}, a_0, a_1, a_2)$ is generated by choosing $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$ at random and

computing

$$\begin{aligned}
a_0 &\leftarrow (h^z)^{\kappa_{2,k'}} \cdot (h^x)^{-r_2} \cdot h^{\kappa_{2,k'} r_2} \cdot H_1(ID)^{r_1} \cdot (h^z \cdot h^{r_2})^{\kappa_{2,0}} \cdot \prod_{i \in \mathsf{id}, i \neq k'} (h^z \cdot h^{r_2})^{\kappa_{2,i}} \\
&= h^{z\kappa_{2,k'} - x r_2 + \kappa_{2,k'} r_2} \cdot H_1(ID)^{r_1} \cdot \left(h^{\kappa_{2,0}} \prod_{i \in \mathsf{id}, i \neq k'} h^{\kappa_{2,i}} \right)^{z + r_2} \\
&= h^{xz - xz + z\kappa_{2,k'} - x r_2 + \kappa_{2,k'} r_2} \cdot H_1(ID)^{r_1} \cdot \left(v_{2,0} \prod_{i \in \mathsf{id}, i \neq k'} v_{2,i} \right)^{z + r_2} \\
&= h^{xz} \cdot H_1(ID)^{r_1} \cdot H_2(\mathsf{id})^{z + r_2} \\
a_1 &\leftarrow h^{r_1}, \\
a_2 &\leftarrow h^z \cdot h^{r_2} \;=\; h^{z + r_2}
\end{aligned}$$

At the end of this stage $\mathcal{A}$ will output a pirate decoder $\mathbb{D}$ with respect to a group identity ID of its choice. Algorithm $\mathcal{B}_2$ then generates the challenge ciphertext with

$$C_1 \leftarrow \psi(h^y)\,, \quad C_2 \leftarrow \psi(h^y)^{\kappa_{1,0}} \textstyle\prod_{i \in ID} \psi(h^y)^{\kappa_{1,i}}\,,$$
$$C_3 \xleftarrow{\$} \mathbb{G}_T\,, \quad C_4^{(i)} \leftarrow \begin{cases} \psi(h^y)^{\kappa_{2,i}} & \text{for } 0 \le i \le n_2,\ i \neq k'\,, \\ Z & \text{where } Z \xleftarrow{\$} \mathbb{G}_1 \text{ for } i = k'\,. \end{cases}$$

By our assumption on the pirate decoder $\mathbb{D}$ with this ciphertext will output, with probability p_1, the corresponding plaintext $\mathcal{M}$ as if $C_4^{(k')}$ were chosen correctly as $u_{2,k'}^y$. In this case $\mathcal{B}_2$ can recover $\hat{e}(g,h)^{xyz}$ by computing $C_3/\mathcal{M}$. Algorithm $\mathcal{B}_2$ then returns this value as its solution to the bilinear computational Diffie–Hellman problem, giving it an advantage

$$\mathrm{Adv}^{\mathrm{cbdh}}_{\mathcal{B}_2, \mathbb{G}_2}(k) \;\ge\; p_1\,,$$

from which the lemma follows. ∎

Acknowledgements.

We would like to thank Yevgeniy Dodis and Aggelos Kiayias for suggesting that a simple method for the converting the q-ary alphabet into binary is sufficient for our purposes.

G.6 Appendix: Waters' HIBE with Asymmetric Pairings

Our scheme is built out of the HIBE suggested by Waters in [Wat05], but in the asymmetric pairing setting and using a scheme of depth 2. In this section we describe the underlying HIBE in full generality.

G.6.1 Scheme Description

Suppose that we want a scheme of depth L. We define L sets $V_1, \ldots, V_L$ of random elements in $\mathbb{G}_2$, with elements denoted $V_i = (v_{i,0}, v_{i,1}, \ldots, v_{i,n_i})$. We let $u_{i,j} = \psi(v_{i,j})$ and let U_i denote the image of the set V_i under the isomorphism ψ, i.e. $U_i = (u_{i,0}, u_{i,1}, \ldots, u_{i,n_i})$.

Just as in our traitor tracing scheme for a bit string B of length n_i we use these sets to define the Waters' hash functions:

$$H_i(B) = v_{i,0} \prod_{j \in B} v_{i,j},$$

where the products are over all the set bits in B. To simplify notation we define

$$G_i(B) = u_{i,0} \prod_{j \in B} u_{i,j} = \psi(H_i(B)).$$

Note that $\psi(H_i(B)) = G_i(B)$ can be computed either from the set V_i using the isomorphism ψ, or from the set U_i directly. Also note that $v_{i,j} = h^{\kappa_{i,j}}$ and $u_{i,j} = g^{\kappa_{i,j}}$ for some, unknown, values $\kappa_{i,j} \in \mathbb{Z}_p$.

Using the entities above, the various algorithms that make up Waters' HIBE scheme are as follows. We assume that id is a tuple $(\mathsf{id}_1, \ldots, \mathsf{id}_l)$ where $l \leq L$ and id_i is a bit string of length n_i, applying a collision resistant hash function if necessary.

Setup $\mathcal{G}(1^k)$ **:** We generate a set of pairing groups as above at the security level k, along with the sets $V_1, \ldots, V_L$ and $U_1, \ldots, U_L$. We require a random element $h \xleftarrow{\$} \mathbb{G}_2$ and let $g \leftarrow \psi(h) \in \mathbb{G}_1$. A random value $\alpha \xleftarrow{\$} \mathbb{Z}_p$ is selected, and we set $g_1 \leftarrow g^\alpha$ and $h_1 \leftarrow h^\alpha$. We require a second random element $h_2 \in \mathbb{G}_2$ and we let $g_2 \leftarrow \psi(h_2)$. The master public key is defined to be $\mathsf{mpk} = \{g, g_1, h_2, U_1, \ldots, U_L\}$ and the master secret key is $\mathsf{msk} = \{h, h_2^\alpha, V_1, \ldots, V_L\}$.

Key Extraction $\mathcal{X}(\mathsf{id}, \mathsf{msk})$ **:** We first select random values $r_1, \ldots, r_l \leftarrow \mathbb{Z}_p$ and then define the private key as

$$d_{\mathsf{id}} = (a_0, a_1, \ldots, a_l) \leftarrow \left(h_2^\alpha \prod_{i=1}^{l} H_i(\mathsf{id}_i)^{r_i}, h^{r_1}, \ldots, h^{r_l} \right) \in \mathbb{G}_2^{l+1}.$$

Encryption $\mathcal{E}(\mathsf{id}, \mathsf{mpk}, \mathcal{M})$ **:** A message is defined as an element in $\mathbb{G}_T$. The sender first choose a $t \leftarrow \mathbb{Z}_p$ and then computes the ciphertext

$$C = (C_1, C_2, C_3) \in \mathbb{G}_1 \times \mathbb{G}_1^l \times \mathbb{G}_\mathrm{T}$$

as

$$C_1 \leftarrow g^t, \quad C_2 \leftarrow \left(C_{2,i} = G_i(\mathsf{id}_i)^t \right)_{i=1}^{l}, \quad C_3 \leftarrow \mathcal{M} \cdot \hat{e}(g_1, h_2)^t.$$

Decryption $\mathcal{D}(C, d_{\mathsf{id}})$ **:** Compute

$$C_3 \cdot \frac{\prod_{i=1}^{l} \hat{e}(C_{2,i}, a_i)}{\hat{e}(C_1, a_0)} = \mathcal{M}.$$

Appendix H

Multi-Channel Broadcast Encryption

ASIA CCS'13
[PPT13] with David Pointcheval, and Viet Cuong Trinh

Abstract : *Broadcast encryption aims at sending a content to a large arbitrary group of users at once. Currently, the most efficient schemes provide constant-size headers, that encapsulate ephemeral session keys under which the payload is encrypted. However, in practice, and namely for pay-TV, providers have to send various contents to different groups of users. Headers are thus specific to each group, one for each channel: as a consequence, the global overhead is linear in the number of channels. Furthermore, when one wants to zap to and watch another channel, one has to get the new header and decrypt it to learn the new session key: either the headers are sent quite frequently or one has to store all the headers, even if one watches one channel only. Otherwise, the zapping time becomes unacceptably long.*

This paper deals with encapsulation of several ephemeral keys, for various groups and thus various channels, in one header only, and we call this new primitive Multi-Channel Broadcast Encryption – MCBE*: one can hope for a much shorter global overhead and a much shorter zapping time since the decoder already has the information to decrypt any available channel at once. Our candidates are private variants of the Boneh-Gentry-Waters scheme, with a constant-size global header, independently of the number of channels.*

H.1 Introduction

Broadcast encryption has been widely and deeply studied as it is a core primitive for many concrete applications. In the following, we focus on the pay-TV scenario, in which users own decoders to decode only the channels they subscribed to. In this context, the broadcaster sends several channels at the same time, to different groups of users or target sets.

Unfortunately, previous broadcast encryption models only dealt with one single content and one single target set at a time. This was a first reasonable goal to get such an efficient broadcast encryption scheme, but not quite relevant to practice. In fact, TV systems contain many channels, with different sets of privileged users. One could argue that this scenario is covered by the usual systems, applying independent broadcast encryption schemes for each channel. However, even with a constant-size ciphertext (header) broadcast encryption, this results in a very inefficient scheme: the bandwidth, or header size, linearly grows in the number

of channels, which could be very large. Of course, one header is enough to decrypt one channel, but in case of zapping from one channel to another channel, one has to start from scratch, and namely to wait for the reception of the new appropriate header, which can take some time, unless the decoder stores all the headers all the time.

These bandwidth and zapping-time problems lead to new efficiency criteria, with a common solution: a broadcast encryption with a short *global header* for multiple channels. The problem of optimizing the bandwidth already appeared in the context of classical (one-channel) broadcast encryption: a broadcast encryption can trivially be constructed from any encryption scheme, by encrypting the session key under each user's key. But this induces a cost that is linear in the number of users. It took more than a decade from the introduction of the primitive [FN93] to come up with an optimal solution: without considering the description of the target set, the header is of constant size [BGW05]. This BGW solution [BGW05] is particularly interesting even if it is still not practical, due to the high decryption complexity: the latter is indeed linear in the size of the target set.

Our new primitive MCBE, for Multi-Channel Broadcast Encryption, addresses the bandwidth and zapping-time problems. In the following, we show that it is possible to solve these problems in an optimal way: a constant-size global header, independently of the number of channels (and of the users too). Unfortunately, this is still an asymptotic result, with room for improvement in practice. Actually, our solutions suffer from the same weakness as the above BGW scheme: the decryption has to take into account all the public keys of the users involved in all the target sets. It is thus not quite efficient. However, it seems that this problem is unavoidable when one compacts the information for all the targeted users into one constant-size ciphertext. We also notice that there is a simple and similar trade-off between the ciphertext size and the decryption time as in [BGW05] by partionning the set of channels into different subsets and then encrypting to each of these subsets. The union of the target sets in a ciphertext is smaller, but there are more ciphertexts. Our objective is therefore to show that the bandwidth and zapping-time problems in the multi-channel setting can be improved from trivial techniques, as BGW did in the one-channel setting.

Finally, we emphasize that the solution requires some new techniques that we will develop in the section H.1.2. In particular, we have to deal with the problem of encapsulating different and independently-looking session keys for the different channels into one constant-size element only. We will then prove the security in the new multi-channel setting.

H.1.1 Broadcast Encryption Schemes

Broadcast encryption was first described by Fiat and Naor in [FN93] but receives much attention since the work of Naor, Naor, and Lotspiech [NNL01] in which they presented a symmetric-key subset-cover framework along with a security model and a security analysis. Dodis and Fazio [DF03] presented the first public-key CCA-secure scheme. Boneh, Gentry, and Waters [BGW05] designed a fully collusion-resistant scheme and proposed a security model where the adversary can corrupt any user, except the users in the challenge target set. With their scheme, the adversary had to precise this challenge target set before knowing the parameters of the system, hence the so-called *selective model*. Delerablée constructed a selectively secure ID-based BE [Del07] in the random oracle model. Thereafter, Gentry and Waters [GW09] defined the *adaptive model*, where the adversary can corrupt users and then adaptively choose the challenge target set, and provided adaptively secure schemes in the standard and the random oracle models. Waters [Wat09] and Lewko *et. al.* [LSW10] used dual system encryption to achieve adaptive security. Recently, a scheme that achieves all desired properties (constant-size ciphertexts, adaptive and CCA security) has been presented in [PPSS12] but it relies on rather non-standard assumptions.

Phan, Pointcheval and Strefler [PPS11] recently gave a global picture of the relations between the security notions for broadcast encryption. However, our setting of multi-channel broadcast encryption goes beyond their considerations, because the adversary could corrupt some users of one channel to break the security of the other channels. The sessions keys of all channels should indeed be compacted into one ciphertext only, there are thus some relations between these keys inside one session and the security model has to take these relations into account.

H.1.2 Contributions

We first propose a formalization of the problem, with the so-called *Multi-Channel Broadcast Encryption* – MCBE. Because of some constraints between the various target sets, we introduce the *dummy-helper technique* that helps to prove the security. We eventually propose two constructions, derived from the Boneh-Gentry-Waters (BGW) [BGW05] scheme. They are private broadcast encryption schemes, with the following properties:

- The first construction is, asymptotically, very competitive with the BGW scheme. In fact, it achieves the constant-size header, independently of the number of channels, while the private decryption key size remains linear in the number of the channels that a user has subscribed to. In addition, it is fully-collusion resistant against basic selective adversaries, *i.e.* adversaries who can only ask corruption queries to get the decryption keys of users in the selective security model (the challenge target set is announced before having seen the global parameters). This is also the security level that the original BGW scheme achieves and our security proof holds under the standard assumption $\mathsf{n}-\mathsf{BDHE}$, as in the original BGW scheme [BGW05].

- The second construction improves on the previous one, to resist to strong selective adversaries who have the power of basic selective adversaries plus unlimited access to encryption and decryption queries, while keeping the parameter sizes and computational assumptions unchanged. To this aim, we introduce the *dummy-helper technique* and make use of a *random oracle* [BR93]. Our scheme is more efficient than the CCA version of the BGW scheme [BGW05] but our dummy-helper technique actually requires the random oracle model.

Dummy-helper technique. In the multi-channel setting, since the session keys of all channels are compacted in only one ciphertext, even if they have to look independent for adversaries, there exists an implicit relation between them, which could be known by the simulator without the whole knowledge of the master key. The *dummy-helper technique* consists in adding a new channel for one additional dummy user. We then get the following interesting properties:

1. For the security analysis: it gives our simulator the possibility to decrypt this channel and get the corresponding session key. This is then sufficient for the simulator to derive the other session keys and successfully answer any decryption query, if the simulator knows the above implicit relation between the encapsulated keys;

2. In practice: by eventually publishing the decryption key of the dummy user, it introduces a channel that can be decoded by all the users registered in the system. It can then be used to send them the program or ads.

We implement this dummy-helper technique in the random oracle model. It is worth noting that, though working in a more complex setting of multi-channel broadcast encryption, the security is achieved under the standard assumption $\mathsf{n}-\mathsf{BDHE}$ as in the BGW scheme.

H.2 Multi-Channel Broadcast Encryption

H.2.1 Syntax

In this section we describe the model for a multi-channel broadcast encryption system. Formally, such a system consists of four probabilistic algorithms:

$\mathsf{Setup}(\lambda)$**:** Takes as input the security parameter λ, it generates the global parameters param of the system, including n the maximal number of users (receivers are implicitly represented by integers in $\{1,\ldots,n\}$), and returns a master key MSK and an encryption key EK. If the scheme allows public encryption, EK is public, otherwise EK is kept private, and can be seen as part of MSK.

$\mathsf{Extract}(i, \mathsf{MSK})$**:** Takes as input the user's index i, together with the master key, and outputs the user's private key d_i.

$\mathsf{Encrypt}(S_1, S_2, \ldots, S_m, \mathsf{EK})$**:** Takes as input m subsets (or target sets) $S_1, S_2, \ldots, S_m$ where, for $i = 1, \ldots, m$, $S_i \subseteq \{1, \ldots, n\}$, and the encryption key EK. It outputs $(\mathsf{Hdr}, K_1, K_2, \ldots, K_m)$ where Hdr encapsulates the ephemeral keys $(K_i)_{i=1,\ldots,m} \in \mathcal{K}$. The key K_i will be associated to the subset S_i. We will refer to Hdr as the broadcast ciphertext, or *header*, whereas this header together with the description of all the target sets is called the *full header*.

$\mathsf{Decrypt}(S_1, S_2, \ldots, S_m, \mathsf{Hdr}, j, d_j, i)$: Takes as input a full header $(S_1, S_2, \ldots, S_m, \mathsf{Hdr})$, a user $j \in \{1, \ldots, n\}$ and its private key d_j, together with a subgroup index $i \in \{1, \ldots, m\}$. If $j \in S_i$, then the algorithm outputs the ephemeral key $K_i \in \mathcal{K}$.

For correctness, we require that for all $S_i \subseteq \{1, \ldots, n\}$ and $j \in S_i$, if $(\mathsf{EK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(\lambda)$, $d_j \leftarrow \mathsf{Extract}(j, \mathsf{MSK})$ and $(\mathsf{Hdr}, K_1, \ldots, K_m) \leftarrow \mathsf{Encrypt}(S_1, S_2, \ldots, S_m, \mathsf{EK})$, one then should get $K_i = \mathsf{Decrypt}(S_1, S_2, \ldots, S_m, \mathsf{Hdr}, j, d_j, i)$.

In practice, the goal of such ephemeral keys is to encrypt the payload, which consists of m messages $M_1, \ldots, M_m$ to be broadcast to the sets $S_1, \ldots, S_m$ respectively. They will thus be encrypted under the symmetric keys $K_1, \ldots, K_m$ into the ciphertexts $\mathsf{CM}_1, \ldots, \mathsf{CM}_m$ respectively. The overall data the broadcaster sends consists of $(S_1, S_2, \ldots, S_m, \mathsf{Hdr}, \mathsf{CM}_1, \mathsf{CM}_2, \ldots, \mathsf{CM}_m)$ where $(S_1, S_2, \ldots, S_m, \mathsf{Hdr})$ is the *full header* and $(\mathsf{CM}_1, \mathsf{CM}_2, \ldots, \mathsf{CM}_m)$ is often called the *encrypted payload*.

H.2.2 Security Model

We define the security of a multi-channel broadcast encryption system by the following game between an attacker $\mathcal{A}$ and a challenger, in the Real-or-Random setting:

Setup. The challenger runs the Setup algorithm to generate the global parameters param of the system, and returns a master key MSK and an encryption key EK. If the scheme is asymmetric, EK is given to $\mathcal{A}$, otherwise it is part of the MSK, and thus kept secret. Corruption and decryption lists Λ_C, Λ_D are set to empty lists.

Query phase 1. The adversary $\mathcal{A}$ adaptively asks queries:

1. Corruption query for the i-th user: the challenger runs $\mathsf{Extract}(i, \mathsf{MSK})$ and forwards the resulting private key to the adversary. The user i is appended to the corruption list Λ_C;
2. Decryption query on the full header $(S_1, \ldots, S_m, \mathsf{Hdr})$ with $u \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. The challenger answers with $\mathsf{Decrypt}(S_1, \ldots, S_m, \mathsf{Hdr}, u, d_u, j)$. The pair (Hdr, S_j) is appended to the decryption list Λ_D;

3. Encryption query (if EK is private) for the target sets $(S_1, S_2, \ldots, S_m)$. The challenger answers with $\mathsf{Encrypt}(S_1, S_2, \ldots, S_m, \mathsf{EK})$.

Challenge. The adversary $\mathcal{A}$ outputs t target sets $S_1^*, \ldots, S_t^* \subseteq \{1, \ldots, n\}$ and an index j, which specifies the attacked target set S_j^*.
The challenger runs $\mathsf{Encrypt}(S_1^*, S_2^*, \ldots, S_t^*, \mathsf{EK})$ and gets $(\mathsf{Hdr}^*, K_1^*, K_2^*, \ldots, K_t^*)$. Next, the challenger picks a random $b \stackrel{\$}{\leftarrow} \{0, 1\}$. If $b = 1$, it picks a random $K_j^* \stackrel{\$}{\leftarrow} \mathcal{K}$. It outputs $(\mathsf{Hdr}^*, K_1^*, \ldots, K_t^*)$ to $\mathcal{A}$.
Note that if $b = 0$, K_j^* is the real key, encapsulated in Hdr^*, and if $b = 1$, K_j^* is random, independent of the header.

Query phase 2. The adversary $\mathcal{A}$ continues to adaptively ask queries as in the first phase.

Guess. The adversary $\mathcal{A}$ eventually outputs its guess $b' \in \{0, 1\}$ for b.

We say the adversary wins the game if $b' = b$, but only if $S_j^* \cap \Lambda_C = \emptyset$ and $(\mathsf{Hdr}^*, S_j^*) \notin \Lambda_D$. We then denote by $\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}) = \Pr[b' = b]$ the probability that $\mathcal{A}$ wins the game, and its advantage is

$$\begin{aligned} \mathrm{Adv}^{\mathsf{ind}}(\mathcal{A}) &= 2 \times \mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}) - 1 \\ &= \Pr[1 \leftarrow \mathcal{A} | b = 1] - \Pr[1 \leftarrow \mathcal{A} | b = 0]. \end{aligned}$$

Definition H.2.1 [Full Security] A multi-channel broadcast encryption scheme is said $(t, \varepsilon, q_C, q_D, q_E)$-secure if for any t-time algorithm $\mathcal{A}$ that makes at most q_C corruption queries, q_D decryption queries, and q_E encryption queries, one has $\mathrm{Adv}^{\mathsf{ind}}(\mathcal{A}) \leq \varepsilon$. We denote by $\mathrm{Adv}^{\mathsf{ind}}(t, q_C, q_D, q_E)$ the advantage of the best t-time adversary.

There are two classical restricted scenarios: a *selective* attacker provides the target sets $S_1^*, S_2^*, \ldots, S_t^* \subseteq \{1, \ldots, n\}$ and index j, which specifies the attacked target set S_j^*, at the beginning of the security game, and one can also restrict the adversary not to ask some queries.

Definition H.2.2 [Basic Selective Security] A multi-channel broadcast encryption scheme is said to be (t, ε, q_C)-selectively secure if it is $(t, \varepsilon, q_C, 0, 0)$-secure against a selective adversary. We denote by $\mathrm{Adv}^{\mathsf{b-ind}}(t, q_C)$ the advantage of the best t-time basic selective adversary.

Note that in the public broadcast setting (where encryption is public), this just excludes decryption queries: we allow CPA adversaries.

Definition H.2.3 [Strong Selective Security] A multi-channel broadcast encryption scheme is said to be $(t, \varepsilon, q_C, q_D, q_E)$-selectively secure if it is $(t, \varepsilon, q_C, q_D, q_E)$-secure against a selective adversary. We denote by $\mathrm{Adv}_s^{\approx}(t, q_C, q_D, q_E)$ the advantages of the best t-time strong selective adversaries.

This definition is much stronger since it not only allows decryption queries in the public setting, but also encryption queries in the private setting.

H.2.3 Disjoint Target Sets

Before presenting our construction in details, we want to stress that our solution requires that all the target sets of the distinct channels are disjoint. Fortunately, this is compatible with our target application of Pay-TV: whenever a user subscribes for a new channel, he is given a new key for decrypting that channel, and it is resonable to consider that the two keys are

independent. More formally, in our systems, we assume there are several channels, which are encrypted to independent target sets of users. The users in the appropriate target sets own decryption keys specific to each channel:

- When a user u registers to the system, he receives a smart card with decryption keys (d_u^i) for every channel i. But at the broadcast time, channel i is encrypted for the target set with the subscribers to this channel only (a subset of the decryption keys);
- Another possibility is to first define U_i the set of all the possible decryption keys for the channel i. When a user u subscribes to a channel i, he receives a key $d_u^i \in U_i$.

In both above cases, the target sets are subsets of predetermined and disjoint sets of keys. As a consequence, the target sets S_i are disjoint too. The drawback is that we have to define many keys in the system.

In a more general setting, in order to limit this number of keys, one could think about sharing keys for several channels. Then, it would make the setting incompatible with our solutions which require disjoint target sets. On the other hand, will reducing the number of keys, it would reduce privacy protection too since would be able to know which channels are registered by similar users, and derive some profiles. Alternatively, in order to limit the global number of keys, one could reassign keys when a user unsubscribes from a channel to another channel.

Anyway, in the following, at a time t, when the broadcaster encapsulates keys for several target sets S_i, we assume them to be disjoint.

H.3 Preliminaries

H.3.1 Computational Assumptions

We first recall the definition of the classical Computational Diffie-Hellman (CDH) assumption:

Definition H.3.1 [CDH Assumption] The $(t, \varepsilon) - \mathsf{CDH}$ assumption says that for any t-time adversary $\mathcal{A}$ that is given $(g, g^r, h) \in \mathbb{G}$, its probability to output h^r is bounded by ε:

$$\mathsf{Succ}^{\mathsf{cdh}}(\mathcal{A}) = \Pr[\mathcal{A}(g, g^r, h) = h^r] \leq \varepsilon.$$

Stronger assumptions have been introduced by Boneh-Gentry-Waters [BGW05]. They both imply the above CDH assumption.

Definition H.3.2 [BDHE Assumption] The $(t, n, \varepsilon) - \mathsf{BDHE}$ assumption says that for any t-time adversary $\mathcal{A}$ that is given $(g, h, g^{\alpha^1}, \ldots, g^{\alpha^n}, g^{\alpha^{n+2}}, \ldots, g^{\alpha^{2n}}) \in \mathbb{G}^{2n+1}$, its probability to output $e(g, h)^{\alpha^{n+1}} \in \mathbb{G}$ is bounded by ε:

$$\mathsf{Succ}^{\mathsf{bdhe}}(\mathcal{A}) = \Pr[\mathcal{A}(g, h, g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}) = e(g_{n+1}, h)] \leq \epsilon.$$

Definition H.3.3 [DBDHE Assumption] The $(t, n, \varepsilon) - \mathsf{DBDHE}$ assumption says that for any t-time adversary $\mathcal{A}$ that is given $(g, h, g^{\alpha^1}, \ldots, g^{\alpha^n}, g^{\alpha^{n+2}}, \ldots, g^{\alpha^{2n}}) \in \mathbb{G}^{2n+1}$, and a candidate to the BDHE problem, that is either $e(g, h)^{\alpha^{n+1}} \in \mathbb{G}$ or a random value T, cannot distinguish the two cases with advantage greater than ε:

$$\mathrm{Adv}^{\mathsf{dbdhe}}(\mathcal{A}) = \left| \begin{array}{l} \Pr[\mathcal{A}(g, h, g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, e(g_{n+1}, h)) = 1] \\ -\Pr[\mathcal{A}(g, h, g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, T)) = 1] \end{array} \right| \leq \epsilon.$$

H.3.2 BGW Overview

To warm up, we first recall the BGW scheme [BGW05], on which our constructions will rely.

$\mathsf{Setup}(\lambda)$**:** Let $\mathbb{G}$ be a bilinear group of prime order p. The algorithm first picks a random generator $g \in \mathbb{G}$ and a random scalar $\alpha \in \mathbb{Z}_p$. It computes $g_i = g^{\alpha^i} \in \mathbb{G}$ for $i = 1, 2, \ldots, n, n+2, \ldots, 2n$. Next, it picks a random scalar $\gamma \in \mathbb{Z}_p$ and sets $v = g^\gamma \in \mathbb{G}$. The public key is $\mathsf{EK} = (g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, v)$, whereas the private decryption key of user $i \in \{1, \ldots, n\}$ is $d_i = v^{\alpha^i}$. These decryption keys are sent by the $\mathsf{Extract}$ algorithm.

$\mathsf{Encrypt}(S, \mathsf{EK})$**:** Pick a random scalar $r \in \mathbb{Z}_p$, and set $K = e(g_{n+1}, g)^r$, where $e(g_{n+1}, g)$ can be computed as $e(g_n, g_1)$ from EK. Next, set: $\mathsf{Hdr} = (g^r, (v \cdot \prod_{j \in S} g_{n+1-j})^r)$, and output (Hdr, K).

$\mathsf{Decrypt}(S, \mathsf{Hdr}, i, d_i, \mathsf{EK})$**:** Parse $\mathsf{Hdr} = (C_1, C_2)$, output $K = e(g_i, C_2)/e(d_i \cdot \prod_{j \in S, j \neq i} g_{n+1-j+i}, C_1)$.

Trivially, when one wants to broadcast m different messages to m different sets $S_1, S_2, \ldots, S_m$, one can combine m independent BGW schemes:

$\mathsf{Setup}(\lambda)$**:** As in the BGW scheme.

$\mathsf{Encrypt}(S_1, S_2, \ldots, S_m, \mathsf{EK})$**:** Pick random scalars $r_1, \ldots, r_m \in \mathbb{Z}_p$, and set

$$K_1 = e(g_{n+1}, g)^{r_1}, \ldots, K_m = e(g_{n+1}, g)^{r_m}$$
$$\mathsf{Hdr} = \left((g^{r_1}, (v \cdot \textstyle\prod_{j \in S_1} g_{n+1-j})^{r_1}), \ldots, (g^{r_m}, (v \cdot \prod_{j \in S_m} g_{n+1-j})^{r_m}) \right).$$

$\mathsf{Decrypt}(S_1, \ldots, S_m, \mathsf{Hdr}, i, (\mathsf{EK}, d_i), j)$**:** Extract $C_1 = g^{r_j}, C_2 = (v \cdot \prod_{j \in S_j} g_{n+1-j})^{r_j}$ from Hdr and decrypt as in BGW.

H.3.3 Intuition

One can note that, in the above "trivial" construction, the number of elements in the header is $2m$, and we want to reduce it. A first attempt is by reusing the same random scalar in all the ciphertexts, which leads to a header of size $m + 1$:

$$\mathsf{Hdr} = \left(g^r, (v \cdot \prod_{j \in S_1} g_{n+1-j})^r, \ldots, (v \cdot \prod_{j \in S_m} g_{n+1-j})^r \right).$$

However, this reuse of random coins suffers from a simple attack: the same random coins result in the same session keys for all channels and a subscriber of a channel can decrypt all channels, since the session key is $e(g_{n+1}, g)^r$. Different r's are thus required in each session keys, but not necessarily totally independent. Our idea is to add an element $X_i \in \mathbb{G}$ corresponding to users $i = 1, \ldots, n$, and to adapt the session key and Hdr using scalars x_i, where $X_i = g^{x_i}$, for $i = 1, \ldots, n$,

$$K_1 = e(g_{n+1}, g)^{r + \sum_{j \in S_1} x_j}, \ldots, K_m = e(g_{n+1}, g)^{r + \sum_{j \in S_m} x_j},$$
$$\mathsf{Hdr} = \left(g^r, (v \cdot \textstyle\prod_{j \in S_1} g_{n+1-j})^{r + \sum_{j \in S_1} x_j}, \ldots, (v \cdot \prod_{j \in S_m} g_{n+1-j})^{r + \sum_{j \in S_m} x_j} \right)$$

The above step shorten the header to $m + 1$ elements, with no more easy attack. But our goal is to have a constant number of elements:

$$\mathsf{Hdr} = \left(g^r, (v \cdot \prod_{j \in S_1} g_{n+1-j})^{r + \sum_{j \in S_1} x_j} \times \cdots \times (v \cdot \prod_{j \in S_m} g_{n+1-j})^{r + \sum_{j \in S_m} x_j} \right)$$

where we essentially multiply all the ciphertexts together. And, magically, it works because a user in a set S_i can cancel out all the terms $(v \cdot \prod_{j \in S_k} g_{n+1-j})^{r+\sum_{j \in S_k} x_j}$ for $k \neq i$ in this product and transform it into his corresponding ciphertext in S_i.

Of course, security has to be proven, this is the goal of the next section to prove the basic selective security. Limitation not to ask decryption nor encryption queries is quite strong, and is the main drawback of the first scheme MCBE_1. And thus, we provide a second construction MCBE_2 that covers strong selective adversaries. For that, we replace $\prod_{j \in S_k} X_j$ by a value outputted by a random oracle on the set S_k and the value g^r at the time of encryption. It will prevent malleability. The *dummy-helper technique* will make the rest.

H.4 Multi-Channel Broadcast Encryption I – MCBE_1

H.4.1 Description

Let us now describe formally our first construction MCBE_1. We will then prove its basic selective security.

$\mathsf{Setup}(\lambda)$**:** The algorithm takes as input the security parameter λ, it generates the global parameters param of the system as follows: Let $\mathbb{G}$ be a bilinear group of prime order p. The algorithm first picks a random generator $g \in \mathbb{G}$ and a random $\alpha \in \mathbb{Z}_p$. It computes $g_i = g^{\alpha^i} \in \mathbb{G}$ for $i = 1, 2, \ldots, n, n+2, \ldots, 2n$. Next, it picks a random $\gamma \in \mathbb{Z}_p$ and sets $v = g^\gamma \in \mathbb{G}$. It also picks additional random scalars $x_1, x_2, \ldots, x_n \in \mathbb{Z}_p$ and sets $X_1 = g^{x_1}, X_2 = g^{x_2}, \ldots, X_n = g^{x_n}$. The master secret key is $\mathsf{MSK} = (g, v, \alpha, \gamma, x_1, x_2, \ldots, x_n)$, while the encryption key (that is private to the broadcaster) is $\mathsf{EK} = (g, v, g_{n+1}, x_1, x_2, \ldots, x_n)$. The public global parameters are $(g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, X_1, X_2, \ldots, X_n)$, whereas the private decryption key is $d_i = v^{\alpha^i}$, for $i \in \{1, \ldots, n\}$. These decryption keys are sent by the $\mathsf{Extract}$ algorithm. We note that if a user registers to t different channels, he will possess t different private decryption keys: n will be the product of the number of users and the number of channels.

$\mathsf{Encrypt}(S_1, S_2, \ldots, S_m, \mathsf{EK})$**:** Pick a random scalar $r \xleftarrow{\$} \mathbb{Z}_p$, set $K_k = e(g_{n+1}, g)^{r+\sum_{j \in S_k} x_j}$ for $k = 1, \ldots, m$. Next, set

$$\mathsf{Hdr} = \left(g^r, \prod_{k=1}^{k=m} (v \cdot \prod_{j \in S_k} g_{n+1-j})^{r+\sum_{j \in S_k} x_j} \right).$$

The broadcaster knows $g_{n+1}, x_1, \ldots, x_n$ from EK. It eventually outputs $(\mathsf{Hdr}, K_1, K_2, \ldots, K_m)$.

$\mathsf{Decrypt}(S_1, \ldots, S_m, \mathsf{Hdr}, i, d_i, k)$**:** Parse $\mathsf{Hdr} = (C_1, C_2)$. If $i \in S_k$ then one computes

$$
\begin{aligned}
K_k &= \frac{e(g_i, C_2)}{e(d_i \cdot \prod_{\substack{j \in S_k \\ j \neq i}} g_{n+1-j+i}, C_1 \cdot \prod_{j \in S_k} X_j) \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} e(d_i \cdot \prod_{j \in S_\ell} g_{n+1-j+i}, C_1 \cdot \prod_{j \in S_\ell} X_j)} \\
&= \frac{e(g_i, C_2)}{e(d_i \cdot \prod_{\substack{j \in S_k \\ j \neq i}} g_{n+1-j+i}, g^{r+\sum_{j \in S_k} x_j}) \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} e(d_i \cdot \prod_{j \in S_\ell} g_{n+1-j+i}, g^{r+\sum_{j \in S_\ell} x_j})} \\
&= \frac{e(g^{\alpha^i}, \prod_{\ell=1}^{\ell=m} (v \cdot \prod_{j \in S_\ell} g_{n+1-j})^{r+\sum_{j \in S_\ell} x_j})}{e(v^{\alpha^i} \cdot (\prod_{\substack{j \in S_k \\ j \neq i}} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_k} x_j}) \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} e(v^{\alpha^i} \cdot (\prod_{j \in S_\ell} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_\ell} x_j})} \\
&= \frac{e(g^{\alpha^i}, (v \cdot \prod_{j \in S_k} g_{n+1-j})^{r+\sum_{j \in S_k} x_j})}{e(v^{\alpha^i} \cdot (\prod_{\substack{j \in S_k \\ j \neq i}} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_k} x_j})} \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} \frac{e(g^{\alpha^i}, (v \cdot \prod_{j \in S_\ell} g_{n+1-j})^{r+\sum_{j \in S_\ell} x_j})}{e(v^{\alpha^i} \cdot (\prod_{j \in S_\ell} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_\ell} x_j})} \\
&= \frac{e((v \cdot \prod_{j \in S_k} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_k} x_j})}{e((v \cdot \prod_{\substack{j \in S_k \\ j \neq i}} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_k} x_j})} \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} \frac{e((v \cdot \prod_{j \in S_\ell} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_\ell} x_j})}{e((v \cdot \prod_{j \in S_\ell} g_{n+1-j})^{\alpha^i}, g^{r+\sum_{j \in S_\ell} x_j})} \\
&= e(g_{n+1-i}^{\alpha^i}, g^{r+\sum_{j \in S_k} x_j}) = e(g_{n+1}, g^{r+\sum_{j \in S_k} x_j}) = e(g_{n+1}, g)^{r+\sum_{j \in S_k} x_j}
\end{aligned}
$$

We used the relations $d_i = v^{\alpha^i}, g_{n+1-j+i} = g_{n+1-j}^{\alpha^i}$, and $g_{n+1-i}^{\alpha^i} = g_{n+1}$.

Remark H.4.1 In MCBE_1, the encryption key EK contains $g_{n+1}, x_1, x_2, \ldots, x_n$ and thus cannot be public: this is a private variant of BGW scheme. Actually, g_{n+1} is not really required, but the $x_i's$ would be enough to break the semantic security, and thus cannot be public either. However, the broadcaster does not need to know α, γ to encrypt, and without them it cannot generate decryption keys for users. We can separate the role of group manager (who generates the decryption keys) and broadcaster (who encrypts and broadcasts the content).

H.4.2 Security Result

We now prove the semantic security of the first scheme.

Theorem H.4.2 The MCBE_1 system achieves the basic selective security under the DBDHE assumption in $\mathbb{G}$. More precisely, if there are n users,

$$\mathrm{Adv}^{\mathsf{b-ind}}(t, q_C) \leq 2 \times \mathrm{Adv}^{\mathsf{dbdhe}}(t', n),$$

for $t' \leq t + (mn + nq_C)T_e$ where T_e is the time complexity for computing an exponentiation and m is the maximum number of channels in the system.

Proof: Let us assume there exists an adversary $\mathcal{A}$ which breaks the semantic security of our first scheme, we build an algorithm $\mathcal{B}$ that has the same advantage in deciding the DBDHE problem in $\mathbb{G}$. This algorithm $\mathcal{B}$ proceeds as follows:

Init. Algorithm $\mathcal{B}$ first takes as input a DBDHE instance $(g, G, g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, T)$, where T is either $e(g_{n+1}, G)$ or a random element of $\mathbb{G}$. It implicitly defines α: $g_i = g^{\alpha^i}$. $\mathcal{B}$ then runs $\mathcal{A}$, and since we are in the selective model, it receives m sets $S_1, \ldots, S_m$ and an index k that $\mathcal{A}$ wishes to be challenged on.

Setup. $\mathcal{B}$ now generates the public global parameters and private keys d_i, for $i \notin S_k$: it first chooses a random scalar $r \in \mathbb{Z}_p$ and sets $h = g^r$, and $h_i = g_i^r$, for $i = 1, \ldots, n$. One chooses a random index η in S_k, and for $i \in \{1, \ldots, n\} \backslash \{\eta\}$, one chooses a random scalar $x_i \in \mathbb{Z}_p$, and computes $X_i = g^{x_i}$. One eventually sets $X_\eta \stackrel{\mathsf{def}}{=} G / \prod_{i \in S_k \backslash \{\eta\}} X_i = g^{x_\eta}$: All the scalars x_i are known, excepted x_η. $\mathcal{B}$ gives $\mathcal{A}$ the public global parameters:

$$(g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, X_1, X_2, \ldots, X_n)$$

$\mathcal{B}$ has to compute all the private decryption keys d_i except for $i \in S_k$: It chooses a random $u \in \mathbb{Z}_p$ and sets

$$v \stackrel{\mathsf{def}}{=} g^u \cdot (\prod_{j \in S_k} g_{n+1-j})^{-1} \qquad d_i \stackrel{\mathsf{def}}{=} g_i^u / (\prod_{j \in S_k} g_{n+1-j+i}) = g^{u \cdot \alpha^i} \cdot (\prod_{j \in S_k} g_{n+1-j})^{-\alpha^i} = v^{\alpha^i}$$

On can remark that $\mathcal{B}$ can compute, without explicitly knowing α, $\prod_{j \in S_k} g_{n+1-j+i}$ for any $i \notin S_k$, and cannot when $i \in S_k$. Moreover, since $d_i = v^{\alpha^i}$, it satisfies the specifications of the schemes.

Challenge. To generate the challenge for $\mathcal{A}$, $\mathcal{B}$ first computes $\mathsf{Hdr} = (C_1, C_2)$ by setting $C_1 = h$, and

$$\begin{aligned}
C_2 &= (h^u \cdot G^u) \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} \left(h^u \cdot \left(\frac{\prod_{j \in S_\ell} h_{n+1-j}}{\prod_{j \in S_k} h_{n+1-j}} \right) \cdot (v \cdot \prod_{j \in S_\ell} g_{n+1-j})^{\sum_{j \in S_\ell} x_j} \right) \\
&= (g^u)^{r + \sum_{i \in S_k} x_i} \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} \left(g^{ur} \cdot \left(\frac{\prod_{j \in S_\ell} g_{n+1-j}}{\prod_{j \in S_k} g_{n+1-j}} \right)^r \cdot \left(v \prod_{j \in S_\ell} g_{n+1-j} \right)^{\sum_{j \in S_\ell} x_j} \right) \\
&= \left(v \prod_{j \in S_k} g_{n+1-j} \right)^{r + \sum_{i \in S_k} x_i} \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} \left(v \prod_{j \in S_\ell} g_{n+1-j} \right)^r \left(v \prod_{j \in S_\ell} g_{n+1-j} \right)^{\sum_{j \in S_\ell} x_j} \\
&= \left(v \prod_{j \in S_k} g_{n+1-j} \right)^{r + \sum_{i \in S_k} x_i} \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m} \left(v \prod_{j \in S_\ell} g_{n+1-j} \right)^{r + \sum_{j \in S_\ell} x_j} \\
&= \prod_{\ell=1}^{\ell=m} \left(v \prod_{j \in S_\ell} g_{n+1-j} \right)^{r + \sum_{j \in S_\ell} x_j}
\end{aligned}$$

We used the following notations and relations $h = g^r$ and $g_{n+1-j}^r = h_{n+1-j}$. Note that $\mathcal{B}$ knows all the values x_i, excepted $x_{i_{k,t}}$, that appears in $h^u \cdot G^u = (v \cdot \prod_{j \in S_k} g_{n+1-j})^{r + \sum_{j \in S_k} x_j}$. To generate session keys, $\mathcal{B}$ first computes, for all $i \neq k$, $K_i = e(g_n, g_1)^{\sum_{j \in S_i} x_j} \cdot e(g_n, h_1)$, and sets $K_k = T \cdot e(g_n, h_1)$. It outputs $(\mathsf{Hdr}, K_1, \ldots, K_m)$ as the challenge to $\mathcal{A}$.

Note that, for $i \neq k$, $K_i = e(g_{n+1}, g)^{r + \sum_{j \in S_i} x_j}$, and, if T is the correct value, $K_k = e(g_{n+1}, G) \cdot e(g_n, h_1) = e(g_{n+1}, g^{\sum_{j \in S_k} x_j}) \cdot e(g_{n+1}, g^r) = e(g_{n+1}, g)^{r + \sum_{j \in S_k} x_j}$. If T is random, the latter is also random.

Guess. $\mathcal{A}$ outputs its guess b' for b. If $b' = b$ the algorithm $\mathcal{B}$ outputs 0 (indicating that $T = e(g_{n+1}, G)$). Otherwise, it outputs 1 (indicating that T is random in $\mathbb{G}_1$). From the above remark, if T is the correct value, $\Pr[\mathcal{B} = 1] = \Pr[b' = b] = (\mathrm{Adv}^{\mathsf{ind}}(\mathcal{A}) + 1)/2$. However, if T is a random value, $\Pr[\mathcal{B} = 1] = 1/2$: $\mathrm{Adv}^{\mathsf{dbdhe}}(\mathcal{B}) = \mathrm{Adv}^{\mathsf{ind}}(\mathcal{A})/2$.

H.5 Multi-Channel Broadcast Encryption II – MCBE_2

We now improve the previous scheme to allow encryption and decryption queries. To this aim, we will need a random oracle.

H.5.1 Dummy-Helper Technique

First, in order to achieve semantic security, we still have to embed the critical element from the $\mathsf{n}-\mathsf{BDHE}$ instance in the challenge header related to the specific target set S_k. In the previous scheme, it was implicitly embedded in the X_η, or at least in one of them. But then, if this element is involved in a decryption query, the simulator cannot answer, hence the limitation for the adversary not to ask decryption queries. For the same reason, it was not possible to simulate encryption queries with this critical value.

Using a random oracle, it is possible to embed this element at the challenge time only, and then, instead of a deterministic $\sum_{i\in S_j} x_i$ one can use a random y_j implicitly defined by Y_j given by a random oracle. With the knowledge of the discrete logarithm y_j (excepted in the challenge ciphertext), the simulator is able to answer all encryption queries, but this is still not enough to answer decryption queries: the simulator has no idea about the random scalar r involved in the ciphertext, whereas it has to compute $e(g_{n+1}, g)^r$. But this can be done by adding a dummy set for which the session key can be computed by the simulator. In this case, we apply the *dummy-helper technique* to prove the security.

H.5.2 Description

$\mathsf{Setup}(\lambda)$**:** it takes as input the security parameter λ, and generates the global parameters param of the system as follows: Let $\mathbb{G}$ be a bilinear group of prime order p; pick a random generator $g \in \mathbb{G}$ and a random scalar $\alpha \in \mathbb{Z}_p$; compute $g_i = g^{\alpha^i} \in \mathbb{G}$ for $i = 1, 2, \ldots, 2n$; pick a random scalar $\gamma \in \mathbb{Z}_p$ and set $v = g^\gamma \in \mathbb{G}$ and $d_n = v^{\alpha^n}$. The algorithm also uses a random oracle $\mathcal{H}$ onto $\mathbb{G}$.

The master key is $\mathsf{MSK} = (g, v, \alpha, \gamma)$, the private encryption key is $\mathsf{EK} = \mathsf{MSK}$ and the public global parameters are $(g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, d_n)$, whereas the private decryption key is $d_i = v^{\alpha^i}$, for $i \in \{1, \ldots, n\}$. These decryption keys are sent by the $\mathsf{Extract}$ algorithm.

$\mathsf{Encrypt}(S_1, \ldots, S_m, \mathsf{EK})$**:** Pick a random scalar $r \in \mathbb{Z}_p$; set $S_{m+1} = \{n\}$, for each set S_i, for $i = 1, \ldots, m+1$ compute $Y_i = \mathcal{H}(i, g^r)$ ($Y_i = g^{y_i}$, for some unknown scalar y_i), and

$$K_i = e(g_{n+1}, Y_i) \cdot e(g_{n+1}, g)^r = e(g_{n+1}, g)^{r+y_i}, \quad i = 1, \ldots, m+1$$

Eventually compute $\mathsf{Hdr} = (C_1, C_2, C_3)$ as follows:

$$\begin{aligned} C_1 &= g^r \\ C_2 &= \prod_{i=1}^{i=m+1} \left(Y_i^{\gamma + \sum_{j\in S_i} \alpha^{n+1-j}} \cdot \left(v \cdot \prod_{j\in S_i} g_{n+1-j} \right)^r \right) = \prod_{i=1}^{i=m+1} \left(v \cdot \prod_{j\in S_i} g_{n+1-j} \right)^{r+y_i} \\ C_3 &= \mathcal{H}(C_1, C_2)^r \end{aligned}$$

Note that the broadcaster knows both α and γ to compute C_2. It outputs $(\mathsf{Hdr}, K_1, \ldots, K_{m+1})$.

$\mathsf{Decrypt}(S_1, \ldots, S_m, \mathsf{Hdr}, i, d_i, k)$**:** Set $S_{m+1} = \{n\}$, parse $\mathsf{Hdr} = (C_1, C_2, C_3)$. If $i \in S_k$ then one first checks whether $e(C_1, \mathcal{H}(C_1, C_2)) = e(g, C_3)$, computes $Y_i = \mathcal{H}(i, g^r)$, for $i = 1, \ldots, m+1$, and computes (as in the previous scheme, where the Y_j's replace some products of the X_i's)

$$\begin{aligned} K_k &= \frac{e(g_i, C_2)}{e(d_i \cdot \prod_{\substack{j \in S_k \\ j \neq i}} g_{n+1-j+i}, C_1 \cdot Y_k) \cdot \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\ell=m+1} e(d_i \cdot \prod_{j \in S_\ell} g_{n+1-j+i}, C_1 \cdot Y_\ell)} \\ &= e(g_{n+1}, g)^{r+y_k} \end{aligned}$$

Note that $d_i = v^{\alpha^i}$, $g_{n+1-j+i} = g_{n+1-j}^{\alpha^i}$, and $g_{n+1-i}^{\alpha^i} = g_{n+1}$.

H.5.3 Security

Theorem H.5.1 The MCBE_2 system achieves the strong selective security under the DBDHE assumption in $\mathbb{G}$. More precisely, if there are n users,

$$\mathrm{Adv}_s^{\approx}(t, q_C, q_D, q_E) \leq 2 \times \mathrm{Adv}^{\mathsf{dbdhe}}(t', n) + 2 \times \mathsf{Succ}^{\mathsf{cdh}}(t'') + 2/p,$$

for $t' \leq t + (nq_C + nmq_D + nmq_E)T_e + (mq_D + mq_E)T_p + mq_D T_{lu}$ and $t'' \leq t + (q_C + q_D + nmq_E)T_e + (q_D + mq_E)T_p + q_D T_{lu}$, where T_e, T_p are the time complexity for computing an exponentiation, a pairings, T_{lu} is the time complexity for a look up in a list, and m is the maximum number of channels in the system.

Proof: We organize our proof in three games:

1. **Game 0:** The real strong selective security game between an adversary and a challenger.
2. **Game 1:** This is similar to Game 0 with a following exception: if we denote $\mathsf{Hdr} = (C_1, C_2, C_3)$ the challenge header, then any decryption query on a different header $\mathsf{Hdr}' = (C_1, C_2', C_3')$, but with the same C_1, we answer $\perp$ (*i.e.* invalid ciphertext). We can show that this exception happens with negligible probability under the CDH assumption.
3. **Game 2:** We can now safely answer all decryption queries $\mathsf{Hdr}' = (C_1, C_2', C_3')$ by $\perp$ and the others using either a valid decryption key or d_n. Using the programmability of the random oracle, and thus the knowledge of the y_i, one can easily simulate the encryption queries. Eventually, the semantic security then relies on the DBDHE assumption.

Game 1: In this game, we know all the secret keys, but answer $\perp$ to a decryption query $\mathsf{Hdr}' = (C_1, C_2', C_3')$, with the same first C_1 as in the challenge header. Our algorithm $\mathcal{B}$ is given a CDH instance $g, A = g^{r^*}, B$, and should answer $C = B^{r^*}$. It runs the adversary $\mathcal{A}$:

- since we consider selective attacks only, the target sets are known from the beginning, and $\mathcal{B}$ can thus first generate the challenge header using r^* as random scalar, without knowing it: $C_1 = A$. Since $\mathcal{B}$ knows MSK, and namely α and γ, it can compute the appropriate C_2: $v^{r^*} = A^\gamma$ and $g_i^r = A^{\alpha^i}$. It then programs $\mathcal{H}(C_1, C_2) = g^u$ for a random scalar u and sets $C_3 = A^u$. The triple (C_1, C_2, C_3) is a perfect header;
- answers all the hash queries $\mathcal{H}(A, X)$, for any X, by B^t for some randomly chosen scalar t;
- answers all the other queries with MSK.

Let now assume that $\mathcal{A}$ asks for a valid decryption query $(S'_1, \ldots, S'_{m'+1}, k', \mathsf{Hdr}')$ in which $C'_1 = A$. Since $C'_3 = \mathcal{H}(C_1, C'_2)^{r^*} = B^{r^* \cdot t}$ for a known value t, one can extract $C = B^{r^*} = (C'_3)^{1/t}$, which breaks the CDH assumption. $\mathsf{Succ}^{\mathsf{ind}}(\mathcal{A}) - \mathsf{Succ}_1(\mathcal{A}) \leq \mathsf{Succ}^{\mathsf{cdh}}(\mathcal{B})$.

Game 2: We now assume there exists a selective adversary $\mathcal{A}$ that breaks the semantic security of our scheme while decryption queries with the same C_1 as in the challenge are answered by $\perp$. We build an algorithm $\mathcal{B}$ that has twice the advantage in deciding the DBDHE in $\mathbb{G}$. As said above, the programmability of the random oracle will help simulating the encryption queries, and the dummy set will help answering the decryption queries. In game 2.1, the algorithm $\mathcal{B}$ is defined as follows:

Init. Algorithm $\mathcal{B}$ first takes as input a DBDHE instance $(g, G, g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, T)$ where $T = e(g_{n+1}, G)$. It implicitly defines α: $g_i = g^{\alpha^i}$. $\mathcal{B}$ then runs $\mathcal{A}$ to receive m^* sets $S^*_1, \ldots, S^*_{m^*}$ and an index k^* that $\mathcal{A}$ wishes to be challenged on. Note that $n \notin S^*_{k^*}$ because the decryption key d_n is public. $\mathcal{B}$ makes use of a random oracle $\mathcal{H}$ which output is a random element in $\mathbb{G}$, and a hash List is initially set empty list, to store all the query-answer, with additional information, when possible. Namely, for a query q, with answer $Y = g^y$, the tuple (q, Y, y) is stored. Sometimes, y will not be known, and thus replaced by $\perp$.

Setup. $\mathcal{B}$ needs to generate the public global parameters and decryption keys d_i, $i \notin S^*_{k^*}$: it chooses a random $u \in \mathbb{Z}_p$ and sets $v \stackrel{\mathsf{def}}{=} g^u / \prod_{j \in S^*_{k^*}} g_{n+1-j}$. It then computes

$$d_i \stackrel{\mathsf{def}}{=} g_i^u / \prod_{j \in S^*_{k^*}} g_{n+1-j+i} = g^{u \cdot \alpha^i} \cdot \left(\prod_{j \in S^*_{k^*}} g_{n+1-j} \right)^{-\alpha^i} = v^{\alpha^i}$$

Eventually, $\mathcal{B}$ gives $\mathcal{A}$ the public global parameters $(g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, d_n)$.

Phase 1. Since we now allow encryption and decryption queries, let show how they can be answered. We first start by the hash queries:

1. There are two kinds of useful hash queries, $(j, u) \in \mathbb{Z}_p \times \mathbb{G}$ or $(u_1, u_2) \in \mathbb{G}^2$. But for any query q, if it has already been asked, the same answer is sent back. Otherwise, $\mathcal{B}$ chooses a random scalar $y \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ and sets $\mathcal{H}(q) = g^y$. It appends the appropriate tuple (q, g^y, y) to the hash List.
2. For an encryption query $(S_1, S_2, \ldots, S_m)$, $\mathcal{B}$ makes the ciphertext as follows: it first chooses a random scalar $r \in \mathbb{Z}_p$ and sets $S_{m+1} = \{n\}$, and $Y_i = \mathcal{H}(i, g^r) = g^{y_i}$ for $i = 1, \ldots, m+1$: y_i is obtained from the hash List. To generate $\mathsf{Hdr} = (C_1, C_2, C_3)$, $\mathcal{B}$ sets $C_1 = g^r$, and computes

$$C_2 = \prod_{i=1}^{m+1} \left(v \cdot \prod_{j \in S_i} g_{n+1-j} \right)^{r+y_i} \qquad C_3 = \mathcal{H}(C_1, C_2)^r$$

   and $K_i = e(g_n, g_1)^{r+y_i}$, for $i = 1, \ldots, m+1$.
3. For a decryption query $(S_1, \ldots, S_{m+1}, \mathsf{Hdr}, i, k)$ in the name of user $i \in S_k$, $\mathcal{B}$ decrypts as follows: it first checks whether $S_k \subseteq S^*_{k^*}$ or not. In the negative case, it finds $j \in S_k \backslash S^*_{k^*}$, and using d_j it can decrypt as the decryption oracle would do; in the positive case
   - $\mathcal{B}$ uses d_n to decrypt, using the decryption oracle, and obtain $K_{m+1} = e(g_{n+1}, g)^{r+y_{m+1}}$;

- $\mathcal{B}$ extracts, from the hash List for $\mathcal{H}(m+1, C_1)$, the value y_{m+1}, and computes

$$L = \frac{K_{m+1}}{e(g_{n+1}, g)^{y_{m+1}}} = e(g_n, g_1)^r$$

- $\mathcal{B}$ extracts, from the hash List for $\mathcal{H}(k, C_1)$, the value y_k, and computes the session key

$$K_k = L \times e(g_n, g_1)^{y_k} = e(g_{n+1}, g)^{r+y_k}$$

Challenge. The challenge has to be generated on the target sets $S_1^*, \ldots, S_{m^*}^*$, with the index k^* for the indistinguishability of the key:

- $\mathcal{B}$ first chooses a random scalar $r^* \in \mathbb{Z}_p$ and sets $h = g^{r^*}$, and $h_i = g_i^{r^*}$ for $i = 1, \ldots, n$;
- it chooses a random scalar $z^* \in \mathbb{Z}_p$ and sets $\mathcal{H}(k^*, h) = Y_{k^*}^* = G/g^{z^*}$, which is the value $Y_{k^*}^* = g^{y_{k^*}^*}$ for an unknown $y_{k^*}^*$. The tuple $((k^*, h), Y_{k^*}^*, \perp)$ is appended to the hash List;
- $\mathcal{B}$ asks for the other values $Y_i^* = \mathcal{H}(i, h) = g^{y_i^*}$, for $i = 1, \ldots, k^*-1, k^*+1, \ldots, m^*+1$

Note that $S_{m^*+1}^* = \{n\}$, then $\mathcal{B}$ generates $\mathsf{Hdr}^* = (C_1^*, C_2^*, C_3^*)$ by setting $C_1^* = h$ and $C_3^* = \mathcal{H}(C_1^*, C_2^*)^{r^*}$, where (as in the previous proof)

$$\begin{aligned} C_2^* &= \left(h^u \cdot (Y_{k^*}^*)^u\right) \prod_{\substack{\ell=1 \\ \ell \neq k^*}}^{\ell=m^*} \left(h^u \cdot \left(\frac{\prod_{j \in S_\ell^*} h_{n+1-j}}{\prod_{j \in S_{k^*}^*} h_{n+1-j}} \right) \left(v \prod_{j \in S_\ell^*} g_{n+1-j} \right)^{y_\ell^*} \right) \\ &= \prod_{\ell=1}^{\ell=m^*} \left(v \prod_{j \in S_\ell^*} g_{n+1-j} \right)^{r^*+y_\ell^*} \end{aligned}$$

To generate the session keys, $\mathcal{B}$ first computes

$$K_i^* = e(g_n, g_1)^{y_i^*} \cdot e(g_n, h_1) = e(g_{n+1}, g)^{r^*+y_i^*}, \qquad i \neq k^*.$$

It then sets

$$K_{k^*}^* = \frac{T \cdot e(g_n, h_1)}{e(g_{n+1}, g^{z^*})}$$

It gives $(\mathsf{Hdr}^*, K_1^*, \ldots, K_{m^*+1}^*)$ as the challenge to $\mathcal{A}$.

Note that when $T = e(g_{n+1}, G)$, with $G = Y_{k^*}^* g^{z^*}$,

$$K_{k^*}^* = \frac{e(g_{n+1}, Y_{k^*}^* g^{z^*}) \cdot e(g_n, h_1)}{e(g_{n+1}, g^{z^*})} = e(g_{n+1}, g)^{y_{k^*}^*} \cdot e(g_{n+1}, g)^{r^*} = e(g_{n+1}, g)^{r^*+y_{k^*}^*}$$

Phase 2. $\mathcal{B}$ responds as in the first phase. Note that, if $\mathcal{A}$ asks a decryption query with $C_1 = C_1^*$, $\mathcal{B}$ simply answers $\perp$.

In this game 2.1, the advantage of $\mathcal{A}$ is unchanged, except in case of problem during the programmation of $\mathcal{H}$, which is required once only, and the query has already been asked with probability $1/p$: $\mathsf{Succ}_1(\mathcal{A}) - \mathsf{Succ}_{2.1}(\mathcal{A}) \leq 1/p$. In a game 2.2, we replace T by a random element in $\mathbb{G}$: $\mathsf{Succ}_{2.2}(\mathcal{A}) = 1/2$, whereas $\mathsf{Succ}_{2.1}(\mathcal{A}) - \mathsf{Succ}_{2.2}(\mathcal{A}) \leq \mathrm{Adv}^{\mathsf{dbdhe}}(\mathcal{B})$.

As a consequence,

$$\mathsf{Succ}_s^{\approx}(\mathcal{A}) \leq \mathsf{Succ}^{\mathsf{cdh}}(\mathcal{B}_1) + \mathrm{Adv}^{\mathsf{dbdhe}}(\mathcal{B}_2) + 1/p + 1/2,$$

where $\mathcal{B}_i$ denotes the simulator $\mathcal{B}$ in Game i. ▌

H.6 Conclusion

We initiate the new research line on multi-channel broadcast encryption. Our objective is to optimize the ciphertext size while maintaining the polynomial-time complexity of all the algorithms. We propose two efficient schemes with constant-size ciphertexts and leave some challenging open problems:

- As already mentioned in the introduction, our schemes share the same weakness as with BGW scheme: the decryption takes into account of all the corresponding public keys of the users in all the target sets. It is thus not quite efficient. A trade-off between the ciphertext size and the decryption time can be done by partitioning the sets for each channel into subsets and then encrypting to each of these subsets. A better solution than this trade-off would definitely be very interesting.
- While privacy concerns imply independent keys for all the channels a user subscribed to, this however also leads to large decryption keys for users (linear in the number of channels). One could prefer to have shorter or even constant-size keys, sacrificing on privacy. This problem is quite related to the above one.
- Our first scheme achieves the basic selective security level in the standard model while our second scheme achieves the strong selective security level, which resists to both CPA and CCA, but in the random oracle model. Ruling out the random oracle seems quite challenging because of the implicit relations between session keys.

Acknowledgments

This work was partially supported by the French ANR-09-VERSO-016 BEST Project, the European Commission through the ICT Program under Contract ICT-2007-216676 ECRYPT II and partially conducted within the context of the International Associated Laboratory Formath Vietnam (LIAFV).

Appendix I

Decentralized Dynamic Broadcast Encryption

SCN 2012

[PPS12a] with David Pointcheval, and Mario Stroller

Abstract : *A broadcast encryption system generally involves three kinds of entities: the group manager that deals with the membership, the encryptor that encrypts the data to the registered users according to a specific policy (the target set), and the users that decrypt the data if they are authorized by the policy. Public-key broadcast encryption can be seen as removing this special role of encryptor, by allowing anybody to send encrypted data. In this paper, we go a step further in the decentralization process, by removing the group manager: the initial setup of the group, as well as the addition of further members to the system, do not require any central authority. Our construction makes black-box use of well-known primitives and can be considered as an extension to the subset-cover framework. It allows for efficient concrete instantiations, with parameter sizes that match those of the subset-cover constructions, while at the same time achieving the highest security level in the standard model under the DDH assumption.*

I.1 Introduction

Broadcast encryption (BE), introduced by Fiat and Naor [FN93] in 1993, allows a sender to securely send private messages to a subset of users, the target set. In 2001, Naor, Naor, and Lotspiech (NNL [NNL01]) introduced the subset-cover framework, where for any target set, the sender can find a partition of the user set, encrypt a session key using the keys associated to each subset in the partition, and finally encrypt the content using the session key. The ciphertext length of the subset-difference (SD) version of NNL depends linearly on the number of users in the revoked set, which was considered to be efficient enough for use in the AACS DRM standard [AAC09]. We generalize the subset-cover framework of NNL to deal with both public-key encryption and dynamic changes of the registered user sets. We furthermore remove the need for trusted authorities by eliminating the group manager, who typically interacts with users to distribute keys at the setup phase or when users join the system. Our approach makes use of group key exchange with subgroup keys [Man09, ACMP10], a primitive that simultaneously distributes different keys to certain subsets of the user group and applies well to the subset-cover framework if one can assign keys for the subgroups involved in the subset cover.

We first instantiate our construction with the Diffie-Hellman key agreement for the key generation and the ElGamal encryption for the public-key encryption, which leads to quite an efficient scheme. The complete-subtree (CS) tree construction resembles the tree-based group key agreement in [KPT04], with the exception that we also create key pairs for internal nodes, and we go beyond their scheme in our construction of SD trees. We then show how our scheme can be extended to achieve the strongest security notion by using Cramer-Shoup encryption, which allows adaptive corruptions and chosen-ciphertext attacks, in the standard model, under the DDH assumption. In addition, we consider various criteria of efficiency: ciphertext size, private part and public part of the decryption keys, number of rounds for the key generation, etc. Thanks to the modularity of our approach, we can use any appropriate group key exchange with subgroup keys: our initial technique iteratively uses the two-party Diffie-Hellman key exchange in a binary tree, which requires a logarithmic number of rounds; we can replace it by logarithmically many parallel executions of the Burmester-Desmedt group key exchange protocol [BD05], which reduces the number of rounds to two. Besides allowing members to join the system, we also sketch how groups could merge at low cost, and how to permanently revoke some users, but we cannot elaborate on this due to space constraints. Our scheme thus achieves a maximum of functionality and security under minimal assumptions, while still being reasonably efficient.

Related Work. Dodis and Fazio [DF02] already constructed a public-key version of the subset-cover framework using IBE for the Complete-Subtree (CS) structure and HIBE of depth $\log N$ for the Subset-Difference (SD) structure. They retain the same efficiency, using (H)IBE keys instead of symmetric keys, and achieve generalized CCA security. In the same year, Dodis and Fazio presented a dynamic, IND-CCA-secure BE scheme [DF03], where the adversary may corrupt users before the challenge phase. IND-CPA-security under adaptive corruption was first achieved by Boneh and Waters [BW06b], who presented a fully-collusion resistant trace-and-revoke scheme. More recently, Gentry and Waters [GW09] described another adaptively IND-CPA-secure scheme. For both schemes, there is no obvious way to make them IND-CCA-secure in the standard model.

Delerablée [Del08] constructed selectively IND-CPA-secure ID-based BE, which allows adding users after the setup. The only existing dynamic BE scheme was developed by Delerablée, Paillier, and Pointcheval[DPP07]. However, their scheme does not provide forward-secrecy, i. e. a new user can decrypt all ciphertexts sent before he joined. Because our scheme provides forward-secrecy, we have to relax their definition of "dynamic". Forward-security has been considered by Yao, Fazio, Dodis, and Lysyanskaya [YFDL04], first for HIBE and then by extension for BE. Their notion of forward-security refers to security of ciphertexts against later corruption of users, which means that user keys must evolve so that previously sent messages remain secure. This is distinct from our notion of forward-secrecy, where we only require that newly joined users cannot decrypt previously sent ciphertexts. However, when a user gets corrupted, messages this user received prior to corruption can be read by the adversary, since the adversary gets the same power as the user. The scheme in [YFDL04] is IND-CCA-secure, but the adversary is more restricted in corrupting users after the challenge phase than in our setting.

Broadcast encryption without a central authority replaces the traditional setup with a group key exchange process that can be an interactive protocol. It was proposed under the name "contributory broadcast encryption" (CBE) in [WQZ+11], along with a semi-adaptively IND-CPA-secure scheme that is not dynamic. A possible application of this could be communication in a social network, where some private information is meant to be read only be a subset of a user's acquaintances, and the network is either peer-to-peer or the service provider is not trusted. The first steps toward subgroup key exchange were done by Manulis [Man09], who

extended a group key exchange (GKE) protocol to allow any two users to compute a common key after the initial phase in which the group key is computed. Following this work, Abdalla et al. [ACMP10] generalized this approach to allow the computation of session keys for arbitrary subsets. We use such a group key exchange protocol with subgroup keys to derive asymmetric encryption keys for subsets. Something similar has been done under the name of "asymmetric group key agreement" (ASGKA) [WMS+09]. In [WMS+09], ASGKA is defined in a way that guarantees only that the keys held by the participants are good for use with a specific encryption scheme. We want to generalize this requirement so that at the end of the protocol run, each user has some randomness, which can thereafter be used for any key generation, and namely to generate key pairs for any key encapsulation mechanism. Since this randomness is shared between various subgroups, we call the scheme we use for the setup "subgroup key exchange" (SKE). Kurnio, Safavi-Naini, and Wang [KSNW03] explicitly consider sponsorship of group candidates by existing members. In our scheme, because of the tree structure, each user can act as a sponsor, and only one sponsor is required for a candidate to join the user set.

Contributions and Organization. In section I.2, we define decentralized dynamic broadcast encryption and subgroup key exchange, a building block we use in our construction that may be of independent interest. We extend the security notions of adaptive IND-CPA and IND-CCA from [PPS11] to our case. We describe a black-box construction of decentralized dynamic broadcast encryption using the subset-cover framework in section I.3 and prove the security of the construction, assuming that the building blocks are secure. In section I.4, we construct a subgroup key exchange protocol based on any secure two-party key exchange protocol. We give two concrete instantiations using our methodology in section I.5, that provide keys for subgroups in the CS and SD structures. Combined with the Cramer-Shoup encryption scheme, this gives us a decentralized dynamic broadcast encryption schemes which additionally achieves the highest security level (fully adaptive IND-CCA-security) in the standard model under the DDH-assumption.

I.2 Definitions

In the following, we describe some generic constructions for broadcast encryption that make use of standard definitions of well-known primitives. We briefly review the notations here, but provide full definitions in the Appendix I.6.1.

A public-key encryption scheme is defined by a tuple of four algorithms $\mathcal{PKE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$. A two-party key exchange protocol is a tuple of two algorithms/protocols $\mathcal{KE} = (\mathsf{Setup}, \mathsf{CommonKey})$. Note that CommonKey is an interactive protocol, but we expect it to be one-round only for the efficiency of our constructions. A message authentication code is a tuple of three algorithms $\mathcal{MAC} = (\mathsf{KeyGen}, \mathsf{GenMac}, \mathsf{VerifMac})$. A pseudo-random generator is a function $\mathcal{F} : X \to Y$ with $|X| \leq |Y|$.

I.2.1 Decentralized Broadcast Encryption

Let us start with the main protocol we want to build: a broadcast encryption scheme, which aims at encrypting a message for a group of users, with a fine-grained selection of the target group. As in [FN93], broadcast encryption generally involves a group manager, that deals with the membership of the users, and an encryptor that specifies the target group (a subgroup of the registered members) for a ciphertext. The encryptor is either a specific person in case of secret-key broadcast encryption, or anybody in case of public-key broadcast encryption. The group manager is either involved once only, at the setup phase, in static schemes, or at any

time a new member wants to join the system, in dynamic schemes [DPP07]. The latter dynamic situation is the most realistic, but makes the group manager quite sensitive, for both security and availability. Our goal is to get rid of such a centralized system.

We thus extend the dynamic broadcast encryption setting [DPP07] so that the membership management can be decentralized. At the same time, we would like to keep everything as small as possible.

1. The ciphertext size should be as small as possible: the ciphertext has to contain the target group structure, and so cannot be smaller than the representation of this structure, which can either be encoded on N bits, where N is the total number of users, and each bit tells whether a user is in the target group or not, or on $r \log N$ bits (resp. $s \log N$ bits), where r (resp. s) is the number of revoked users (resp. included users) among the N registered users. This is sometimes considered independently from the ciphertext, in the header, but anyway both the target set and the encrypted data have to be sent. Our goal is to make the global length as small as possible.

2. When a new user joins the system, it should have minimal impact on other users' secret information and the public information: no impact at all on the keys as in [DPP07] is of course optimal, but when one wants to achieve forward secrecy, it is not possible: some of the keys have to be modified. We will try to keep the impact as small as possible too.

Since we want to avoid any centralized group manager, we will also focus on public-key broadcast encryption, in which a public key is enough to target any subgroup at the encryption time. In addition, instead of encrypting a message, our schemes will generate an encapsulation (or key header) and session keys to be used with any symmetric encryption scheme [Sho00].

Definition I.2.1 [Decentralized Dynamic Broadcast Encapsulation] A decentralized dynamic broadcast encapsulation scheme is a tuple of five algorithms or protocols $\mathcal{DBE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Join}, \mathsf{Encaps}, \mathsf{Decaps})$:

- $\mathsf{Setup}(1^k)$, where k is the security parameter, generates the global parameters param of the system.

- $\mathsf{KeyGen}(\mathsf{param}, U)$ is an interactive protocol between the users in the set U. After the protocol run, it returns the public encryption key EK and a list Reg of the registered users with additional public information. Each user $u \in U$ eventually gets a secret decryption key dk_u.

- $\mathsf{Join}(v, \{u(\mathsf{dk}_u)\}_{u \in U}, Reg, \mathsf{EK})$ is an interactive protocol run between a user v and the set of users U, described in Reg. Each user takes as input his secret key and/or some random coins, the list Reg, and the encryption key EK. After the protocol, Reg and EK are updated, and each user (including v) has a secret decryption key.

- $\mathsf{Encaps}(\mathsf{EK}, Reg, S)$ takes as input the encryption key EK, the user register Reg, and a target set S. It outputs a key header H and a session key $K \in \{0,1\}^k$.

- $\mathsf{Decaps}(\mathsf{dk}_u, S, H)$ takes as input the target set S and a user decryption key dk_u together with a key header H. If dk_u corresponds to a recipient user, it outputs the session key K, else it outputs the error symbol $\perp$.

The correctness requirement is that for all N, any target set $S \subset U_N = [1, N]$ and for any $u \in U_N$, if $u \in S$ then the decapsulation algorithm gives back the key. A decentralized scheme requires that no authority is involved in the KeyGen and Join protocols.

$\mathrm{Exp}^{\mathsf{ind-acca}-b}_{\mathcal{DBE},\mathcal{A}}(k)$
$\mathcal{Q}_C \leftarrow \emptyset$; $\mathcal{Q}_D \leftarrow \emptyset$;
$\mathsf{param} \leftarrow \mathsf{Setup}(1^k)$;
$(state, U) \leftarrow \mathcal{A}(\mathsf{SETUP}; \mathsf{param})$;
$(\mathsf{EK}, Reg, \tau) \leftarrow \mathsf{OExecute}(U)$;
$(state, S) \leftarrow \mathcal{A}^{\mathsf{OJoin}(\cdot),\mathsf{OCorrupt}(\cdot),\mathsf{ODecaps}(\cdot,\cdot,\cdot)}(state; \mathsf{EK}, Reg, \tau)$;
$(H, K) \leftarrow \mathsf{Encaps}(\mathsf{EK}, Reg, S)$;
$K_b \leftarrow K$; $K_{1-b} \xleftarrow{\$} \mathcal{K}$;
$b' \leftarrow \mathcal{A}^{\mathsf{OJoin}(\cdot),\mathsf{OCorrupt}(\cdot),\mathsf{ODecaps}(\cdot,\cdot,\cdot)}(state; H, K_0, K_1)$;
`if` $\exists i \in S, (i, S, H) \in \mathcal{Q}_D$ `or` $i \in \mathcal{Q}_C$;
`then return` 0
`else return` b';

$\mathsf{OExecute}(U)$
$(\mathsf{EK}, Reg) \leftarrow \mathsf{KeyGen}(\mathsf{param}, U)$;
`return` EK, Reg, τ;

$\mathsf{OJoin}(v)$
$(\mathsf{EK}, Reg) \leftarrow \mathsf{Join}(v, U, Reg, \mathsf{EK})$;
`return` EK, Reg, τ;

$\mathsf{OCorrupt}(u)$
$\mathcal{Q}_C \leftarrow \mathcal{Q}_C \cup \{u\}$; `return` dk_u;

$\mathsf{ODecaps}(u, S, H)$
$\mathcal{Q}_D \leftarrow \mathcal{Q}_D \cup \{(u, S, H)\}$;
$K \leftarrow \mathsf{Decaps}(\mathsf{dk}_u, S, H)$;
`return` K;

Figure I.1: $\mathcal{DBE}$: Key Indistinguishability (IND-ACCA)

Security Notions A general overview of the security notions for broadcast encryption has been done in [PPS11]. We extend the strongest one to the decentralized setting. The adversary is still given unlimited access to the Join oracle (dynamic), the Corrupt oracle (adaptive) and Decaps oracle (chosen-ciphertext security). For the group key generation, the definition from [PPS11] models passive adversaries only, since they only receive the public keys. Since in our case this group key generation may be an interactive protocol, we make it more explicit with a Execute-oracle that outputs the public transcript of the full run of this protocol. The security game for DBE is presented in figure I.1: the restriction for the adversary is not to ask for the decapsulation of the challenge ciphertext (which includes the target set S) nor corrupt any user in the target set.

The adversary can ask once the generation of the group structure with a single call to OExecute on a group U of its choice, from which it gets the transcript τ, the encryption key EK and the register Reg. It can thereafter make as many calls it wants to OJoin, to add a user to the structure Reg, which updates EK. The adversary also gets the transcript τ of this interactive protocol. At any time, the adversary can also corrupt a user with a key pair, calling OCorrupt and getting back all the secret information of the user, and decapsulate a ciphertext H, calling ODecaps in the name of a user u.

The main security goal of an encryption scheme (or an encapsulation scheme) is the indis-

tinguishability of a challenge ciphertext: at some point, the adversary thus gets a challenge (H, K_0, K_1), where H encapsulates either K_0 or K_1 for a target set S chosen by the adversary. It has to guess which key is actually encapsulated. Of course, there are the natural restrictions, which are controlled granted the lists $\mathcal{Q}_C$ and $\mathcal{Q}_D$:

- (S, H) has not been asked to the decapsulation oracle for a user u in S
- none of the users in S have been corrupted

A dynamic broadcast encapsulation scheme is $(t, N, q_C, q_D, \varepsilon)$-IND-ACCA-secure (security against adaptive corruption and chosen-ciphertext attacks) if in the security game presented in figure I.1, the advantage $\mathrm{Adv}^{\mathsf{ind-acca}}_{\mathcal{DBE}}(k, t, N, q_C, q_D)$, of any t-time adversary $\mathcal{A}$ creating at most N users (OJoin oracle), corrupting at most q_C of them (OCorrupt oracle), and asking for at most q_D decapsulation queries (ODecaps oracle), is bounded by ε.

$$\mathrm{Adv}^{\mathsf{ind-acca}}_{\mathcal{DBE}}(k, t, N, q_C, q_D) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{ind-acca}-1}_{\mathcal{DBE},\mathcal{A}}(k) = 1] - \Pr[\mathrm{Exp}^{\mathsf{ind-acca}-0}_{\mathcal{DBE},\mathcal{A}}(k) = 1]\}.$$

This definition includes IND-ACPA (for adaptive chosen-plaintext attacks) when $q_D = 0$.

Remark I.2.2 [Forward-secrecy] This definition includes forward-secrecy against new users, i. e. a new user cannot decrypt ciphertexts that were created before he joined. For a definition without forward secrecy, the adversary is prohibited from corrupting users that joined after the challenge phase.

I.2.2 Subgroup Key Exchange

The novelty of our definition is the decentralized key generation procedure, that should also generate keys for certain subgroups in order to be able to broadcast to any target set. This is thus in the same vein as the notion of group key exchange with on-demand computation of subgroup keys (GKE+S) from [ACMP10], that allows some subgroups of users to run a protocol to establish keys between them. But we extend this definition by allowing for keys of some subgroups to be computed during the first protocol run that establishes the global key, without any additional interaction.

Since we want to remain independent of the encryption scheme to be used with the session key, we require that for each subgroup a proto-key is computed, whose entropy can be used as input to a PKE key-pair generation, or to generate a symmetric encryption key.

Definition I.2.3 [Dynamic S-Subgroup Key Exchange Protocol] For a collection $S : \mathbb{N} \to \mathcal{P}(\mathbb{N})$ of subsets of the user set, where for any N, $S(N) \in \mathcal{P}(N)$, a dynamic S-subgroup key exchange protocol $\mathcal{SKE}$ is a tuple of three algorithms and interactive protocols:

- $\mathsf{Setup}(1^k)$, where k is the security parameter, generates the global parameters param of the system;
- $\mathsf{KeyGen}(\mathsf{param}, U)$ is an interactive protocol run between all users in U. It outputs a register Reg that contains a description of U and the subsets for which keys were established according to S, and for each user $u \in U$ a secret usk_u that contains the proto-keys pt_S for all the sub-groups S containing u.
- $\mathsf{Join}(v, U, Reg)$ is an interactive protocol run between user v and the group of users U. It outputs an updated register Reg and for user v and some of the users in U a new secret usk_u that contains the proto-keys pt_S of all the subgroups S they are part of.

$\mathrm{Exp}^{\mathsf{ind}-b}_{\mathcal{SKE},\mathcal{A}}(k)$
$Reg \leftarrow \emptyset$; $\mathcal{Q}_T \leftarrow \emptyset$;
$\mathsf{param} \leftarrow \mathsf{Setup}(1^k)$;
$(state, U) \leftarrow \mathcal{A}(\mathsf{param})$;
$(\mathsf{EK}, Reg, \tau) \leftarrow \mathsf{OExecute}(U)$;
$b' \leftarrow \mathcal{A}^{\mathsf{OJoin}(\cdot),\mathsf{OTest}(\cdot,\cdot)}(state; \mathsf{EK}, Reg, \tau)$;
`return` b';

$\mathsf{OTest}(t, S)$
`if` $\exists (t', K) \wedge t \equiv_S t' \wedge (t', S, K) \in \mathcal{Q}_T$
`then return` K;
`else if` $b = 0$ `then` $K \leftarrow \mathsf{pt}_S(t)$;
`else` $K \xleftarrow{\$} \mathcal{K}$;
$\mathcal{Q}_T \leftarrow \mathcal{Q}_T \cup \{(t, S, K)\}$;
`return` K;

$\mathsf{OExecute}(U)$
$t \leftarrow 0$;
$Reg \leftarrow \mathsf{KeyGen}(\mathsf{param}, U)$;
`return` Reg, τ;

$\mathsf{OJoin}(v)$
$t \leftarrow t + 1$;
$Reg \leftarrow \mathsf{Join}(v, U, Reg)$;
`return` Reg, τ;

Figure I.2: $\mathcal{SKE}$: Key Indistinguishability (IND)

We require that all the users $u \in U$ that run $\mathsf{KeyGen}(\mathsf{param}, U)$ receive the same register Reg and compute matching proto-keys for the subsets they have in common. The same is required of Join.

For the security definition, we extend the definition given in [ACMP10], which seems to be most applicable to our case. Since the protocol is dynamic, the user set can change over time. As in the previous section, we stick to passive adversaries. This is a way of modularizing protocol construction, as passively secure protocols can be made secure against active adversaries using constructions such as [KY07], with additional authentication mechanisms.

The adversary can ask once the generation of the group structure with a unique call to OExecute, at time $t = 0$, on a group U of its choice from which it gets the transcript τ and the register Reg. It can thereafter make as many calls as it wants to OJoin, to add a user to the structure Reg. Each query increases the time index t. The adversary also gets the transcripts τ of these interactive protocols.

The main security goal of key exchange is the indistinguishability of the keys, and their independence. Hence, we use the stronger notion proposed in [AFP05], similar to the Real-or-Random [BDJR97] for encryption. The adversary has access to many $\mathsf{OTest}(t, S)$ queries, that are either answered by the real keys or by truly random and independent keys. Note that according to the protocol, some keys may remain unchanged even when the time period evolves. We even hope to have as many keys as possible that do not evolve, since we want that not too many users are impacted by a new member in the system. We thus say that two pairs (t_1, S) and (t_2, S) are equivalent (denoted by $t_1 \equiv_S t_2$) if S is unchanged between the time periods and therefore they should have the same key. For such equivalent pairs, the same random key is output. We do not provide direct access to a OReveal oracle, which returns the secret key of a user, because as explained in [AFP05], having access to many OTest queries annihilates the advantage provided by OReveal queries.

A subgroup key exchange scheme is said to be (t, N, q_T, ε)-IND-secure if, in the security game presented in figure I.2, the advantage $\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k, t, N, q_T)$ of any t-time adversary $\mathcal{A}$ creating at most N users (the final size of the set U), testing at most q_T keys is bounded by ε.

$$\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k, t, N, q_T) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{ind}-1}_{\mathcal{SKE},\mathcal{A}}(k) = 1] - \Pr[\mathrm{Exp}^{\mathsf{ind}-0}_{\mathcal{SKE},\mathcal{A}}(k) = 1]\}.$$

I.3 Generic Decentralized Broadcast Encryption

As already remarked, in the first definition of dynamic broadcast encryption schemes [DPP07], it is required that the existing users are not affected by a join: their decryption keys should not be modified. Only the encryption key could be modified. This constraint is actually achieved by their scheme, but this is possible because the scheme is not forward-secure: a new user can decrypt all ciphertexts that were sent before he joined (since he cannot be in any revoked set).

To achieve forward-secrecy, we have to relax their definition and allow updates of the user decryption keys. Namely, updates of the decryption keys are necessary for forward-secrecy in the subset-cover framework [NNL01], because some keys are shared by several users. With an appropriate subset-cover structure, it can reach asymptotically optimal overall ciphertext size. On the other hand, the naive scheme, where each user has a single key specific to him, can be made dynamic without decryption key updates, but has ciphertexts whose length is linear in the number of users. As soon as keys are shared between users, forward-secrecy makes it necessary to update these shared keys. Hence our relaxation of the model. However, we require these updates of existing keys to be made via public channels.

I.3.1 Generic Public-Key Subset Cover

A subset-cover structure $\mathcal{SC} = \{S_i\}_{i\in I}$ is a set of subsets S_i of a user set U such that for any subset $S \subset U$ there is a subset $\mathcal{L} \subset I$ such that S can be *partitioned* as $S = \bigcup_{i\in\mathcal{L}} S_i$. In particular, this implies that for all users $u \in U$, $\{u\} \in \mathcal{SC}$. In [NNL01], a secret key is assigned to each set S_i, so a message can be encrypted to any subset $S \subset U$ by finding the cover $\mathcal{L}$ of S. Then a session key is encrypted under all the keys associated to the selected subsets. All the other users are then implicitly revoked, since they cannot decrypt the session key. Because of the partition property, a user in S is in one subset S_i only. Efficiency will thereafter depend on the subset-cover structure.

We extend this framework in three directions:

1. First, we transfer this approach to the public-key world. Each S_i is assigned a key pair of some PKE scheme by some key assignment procedure. This means that the assignment of keys to the subsets depends on the PKE scheme used as well as the assignment procedure. For example, for a subset-cover structure $\mathcal{SC}$ and a PKE $\mathcal{PKE}$, we can use the key assignment that assigns each subset with a key pair drawn independently at random by the trusted center.

2. Second, we replace the trusted center by an interactive protocol, a subgroup key exchange.

3. Third, we allow for the addition of users, hence using a *dynamic subgroup key exchange* to generate the keys for a dynamic subset-cover structure.

We first deal with a dynamic subset-cover structure, assuming a subgroup key exchange as a black box. Thereafter, we will consider concrete structures and efficient subgroup key exchanges.

I.3.2 Dynamic Subset-Cover

We define a dynamic subset-cover as a sequence of subset-covers $\{\mathcal{SC}_i\}$ for $i \geq 0$ users, where each $\mathcal{SC}_i$ contains subsets S_j. These subsets never change, so instead of adding a user to a subset, we remove the old one and add a new one. This also means that the same subset S_j can occur in different time periods (the time period changes each time a new user joins). We start with $\mathcal{SC}_0 = \emptyset$ and an empty user set $U_0 = \emptyset$, and then have $U_{n+1} = U_n \cup \{u_{n+1}\}$. From the definition, it is clear that $|U_n| = n$, and w.l.o.g. $U_n = [1, n]$.

For subset-cover based dynamic broadcast encryption, we will have to generate the keys for all the subsets that are involved in $\mathcal{SC}_n$. The following property will optimize efficiency, in the sense that a minimal number of existing users will be impacted by a new member.

Definition I.3.1 [Splitting Property] We say that a dynamic subset cover $\mathcal{SC}$ has the *splitting property*, if the subset cover at time $n+1$ is composed of subsets that either were part of the subset cover at time n, or contain the new user. $\mathcal{SC}_{n+1} = \mathcal{SC}'_{n+1} \cup \mathcal{SC}''_{n+1}$, where $\mathcal{SC}'_{n+1} \subset \mathcal{SC}_n$ and $S_i \in \mathcal{SC}''_{n+1} \Rightarrow u_{n+1} \in S_i$.

With this property, if a subset changes, it is either removed, or it contains u_{n+1}. Then only sets with the new user need new key generation, which is a minimal requirement anyway.

I.3.3 SC-based Decentralized Dynamic Broadcast Encryption

We first assume we have a dynamic subgroup key exchange $\mathcal{SKE}$ that is compatible with our dynamic subset-cover structure. It means that for any n, the subgroup key exchange provides keys for all the subsets S in $\mathcal{SC}_n$. We will later instantiate such a dynamic subgroup key exchange for some dynamic subset-cover structures.

Let us recall that the SC-based broadcast encryption [NNL01] consists in encrypting the same message under the keys of all the subsets that cover the target set. Since one of our goals is to achieve the highest security level, adaptive chosen-ciphertext security, any modification of the description of the target set or one of the ciphertexts in the list should make the global ciphertext invalid, otherwise the scheme is somewhat malleable, and thus insecure against chosen-ciphertext attacks. We will add a MAC to bind the target header and the ciphertexts together. A similar approach has been used by [BK05, DK05]. Instead of a master secret key, our scheme needs only a public register *Reg* to keep track of the users currently enrolled in the system and their public keys.

We first present in details our construction, and then state the security of the construction. It is important to remember that the subgroup key exchange scheme is only assumed to be passively secure, meaning that the protocol requires authenticated channels. This can be achieved in several ways that we will not discuss here. Because the subset cover is a fixed part of the protocol and defines the subsets for each number of users, and we assume that the number of users in the system is always known, the number of a new user and the subsets he belongs to can be computed deterministically by all users. Meta-issues like trust between users and how they should agree on which users to allow into the group are beyond the scope of this paper.

Definition I.3.2 [dBE] Let $\mathcal{PKE}$ be a PKE, $\mathcal{MAC}$ a MAC, $\mathcal{F} : \mathcal{K} \to \mathcal{R}$ a pseudo-random generator, $\mathcal{SC}$ a dynamic subset-cover, and $\mathcal{SKE}$ a dynamic subgroup key exchange compatible with $\mathcal{SC}$ with keys in $\mathcal{K}$. Our Broadcast Encryption Scheme is defined as follows.

- $\mathsf{Setup}(1^k)$:
  1. Run $\mathcal{PKE}.\mathsf{Setup}(1^k)$ to get $\mathsf{param}_{\mathcal{PKE}}$;
  2. Run $\mathcal{SKE}.\mathsf{Setup}(1^k)$ to get $\mathsf{param}_{\mathcal{SKE}}$;
  3. Publish $\mathsf{param} = (\mathsf{param}_{\mathcal{PKE}}, \mathsf{param}_{\mathcal{SKE}})$.
- $\mathsf{KeyGen}(\mathsf{param}, U_n)$, for some integer n:
  1. Run $\mathcal{SKE}.\mathsf{KeyGen}(\mathsf{param}_{\mathcal{SKE}}, U_n)$ to get *Reg*; Each user $u \in U_n$ gets as output of the protocol the proto-keys pt_S for all subsets S he belongs to according to $\mathcal{SC}$. The decryption key dk_u consists of all these pt_S.

2. He computes $(\mathsf{dk}_S, \mathsf{ek}_S) \leftarrow \mathcal{PKE}.\mathsf{KeyGen}(\mathsf{param}_{\mathcal{PKE}}; \mathcal{F}(\mathsf{pt}_S))$, where we use the PRG to generate from the proto-key the random coins of the key generation algorithm;
3. All the encryption keys ek_S are published as EK;
4. The decryption keys dk_S can be either stored in dk_u for users $u \in S$, or deleted since they can be recomputed;

- $\mathsf{Join}(v, \{u(\mathsf{dk}_u)\}_{u \in U_n}, Reg, \mathsf{EK})$:
  1. Run $\mathcal{SKE}.\mathsf{Join}(v, \{u(\mathsf{dk}_u)\}_{u \in U_n}, Reg)$ to get the new Reg;
  2. Each user u does as above to compute dk_S, ek_S and dk_u. Note that granted the splitting properties, only dk_S, and thus ek_S, for S that contain v are affected;
- $\mathsf{Encaps}(\mathsf{EK}, Reg, S)$:
  1. From the target set S, generate the partition $\mathcal{L}$ with $S = \cup_{\mathcal{L}} S_i$;
  2. Generate a session key $\mathcal{K}_e$ and a MAC key $\mathcal{K}_m$;
  3. For each subset $i \in \mathcal{L}$, generate $c_i = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_{S_i}, \mathcal{K}_e || \mathcal{K}_m)$;
  4. Compute $\sigma = \mathcal{MAC}.\mathsf{GenMac}(\mathcal{K}_m, S || (c_i)_{i \in \mathcal{L}})$;
  5. Output $\mathcal{K}_e$ and $H = ((c_i)_{i \in \mathcal{L}}, \sigma)$.
- $\mathsf{Decaps}(\mathsf{dk}_u, S, H)$:
  1. If $u \in S$, then there is a unique i such that $u \in S_i$, and then dk_u allows to derive $\mathsf{dk} = \mathsf{dk}_{S_i}$;
  2. Extract $\mathcal{K}_e || \mathcal{K}_m = \mathcal{PKE}.\mathsf{Decrypt}(\mathsf{dk}, c_i)$;
  3. Check if σ is a valid MAC under key $\mathcal{K}_m$;
  4. In case of validity, output $\mathcal{K}_e$, otherwise output $\perp$.

The scheme is a correct dynamic broadcast encryption scheme, because of the correctness of the basic primitives $\mathcal{PKE}$, $\mathcal{MAC}$ and $\mathcal{F}$, but also $\mathcal{SKE}$.

Theorem I.3.3 Let us consider the scheme $\mathcal{BE}^{\mathcal{PKE},\mathcal{MAC},\mathcal{F},\mathcal{SKE}}$ from definition I.3.2. We define L_N to be the total number of distinct subsets over all time periods and ℓ_N to be the maximal number of subsets necessary to cover any authorized target set S in $\mathcal{SC}_i$ for any i. If $\mathcal{PKE}$ is an IND-CCA-secure PKE, $\mathcal{MAC}$ is a SUF-CMA-secure MAC, $\mathcal{SKE}$ is a IND-secure SKE, and $\mathcal{F}$ is a pseudo-random generator, then this scheme is a forward-secure IND-ACCA-secure BE scheme:

$$\begin{aligned}\mathrm{Adv}^{\mathsf{ind-acca}}_{\mathcal{DBE}}(k,t,N,q_C,q_D) \leq\ & 2\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k,t,L_N,L_N) + 3\ell_N L_N \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k,t,q_D) \\ & + 2L_N \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{F}}(k,t) + 2\mathsf{Succ}^{\mathsf{suf-cma}}_{\mathcal{MAC}}(k,t,1,q_D).\end{aligned}$$

The variables L_N and ℓ_N depend on the type of subset cover used in the scheme. For CS, L_N is less than $N \log N$ (since at most $\log N$ sets change in each of the at most N steps), and ℓ_N is $r \log \frac{N}{r}$, which is bounded by $N/2$ (the worst-case ciphertext length). For SD, we have $L_N \leq N \log^2 N$ and $\ell_N = 2r-1$. The complete security proof can be found in the Appendix I.6.2.

I.4 Tree-based Subgroup Key Exchange

In this section, we define two subgroup key exchange protocols compatible with the efficient tree-based methods defined in [NNL01]. The tree-based methods are special cases in the subset-cover framework, where the users are organized as leaves in a binary tree, and the subsets S_i can be described in terms of subtrees of this tree.

Complete Subtree. We first review the static complete subtree (CS) structure for N users $\{u_0, \ldots, u_{N-1}\}$. For simplicity, we assume $N = 2^d$, but the description can be generalized to any N. All the users are leaves of the tree, and can be seen as singletons $S_{2^d+i} = \{u_i\}$, for $i = 0, \ldots, 2^d - 1$. Then, for $i = 2^d - 1$ to 1, $S_i = S_{2i} \cup S_{2i+1}$ which contains all the leaves below the node with index i.

Subset Difference. The subset difference (SD) method uses subsets $S_{i,j} = S_i \setminus S_j$, where S_i, S_j are defined as in the CS method, and S_j is a subtree of S_i. All sets S_i from the CS tree are also contained in the SD method, because $S_i = S_{\text{parent}(i),\text{sibling}(i)}$; S_0 is included as a special set.

I.4.1 Static Tree Construction

Let us show how such subset-cover structures naturally give rise to subgroup key exchange protocols. The main tools for our construction of the subgroup key exchange are two primitives: a 2-party key exchange protocol $\mathcal{KE}$ that outputs keys in $\mathcal{K}_{\mathcal{KE}}$ and a pseudo-random generator $\mathcal{G} : \mathcal{K}_{\mathcal{KE}} \to \mathcal{K} \times \mathcal{R}_{\mathcal{KE}}$.

Two users start from random coins in $\mathcal{R}_{\mathcal{KE}}$, and run a key exchange protocol $\mathcal{KE}.\mathsf{CommonKey}$ in order to derive a secret value ck for the subset represented by the node in the tree that is their parent. This common key ck is used as the seed for the PRG $\mathcal{G}$ to derive the two secret keys, the proto-key $\mathsf{pt} \in \mathcal{K}$ and the random coins $r \in R_{\mathcal{KE}}$ for the next key exchange at the level above. Internal nodes thus involve "virtual" users. In summary, the tree is constructed by executing $\mathcal{KE}.\mathsf{CommonKey}$, then computing $\mathcal{G}$, at each level from the bottom up. We derive generic instantiations of the complete subtree (CS) and subset difference (SD) methods on binary trees described in [NNL01].

CS Tree. We define the neighbour of user u with identifier i to be the user u' with identifier $i+1$ if $i \equiv 0 \mod 2$, $i-1$ else and its parent to be the user w with identifier $\lfloor i/2 \rfloor$. At round r, each (virtual) user u created in round $r-1$ has a uniquely defined neighbour u' and a parent w. If he does not, the protocol run is completed: we are either at the root of the tree, or the tree is not complete. The users u and u' have random coins r_u and $r_{u'}$, which they use to run the $\mathcal{KE}$ protocol, resulting in a common key ck_w. From this common key, they derive the proto-key of node w and the randomness for the virtual user w to participate in the next round of key exchanges. The user with the smaller identifier then plays the role of the virtual user w in the next round. As a consequence, for N users, there are $\log N$ rounds. Round r involves $N/2^{r-1}$ (virtual) users.

- $\mathsf{KeyGen}(U_n)$: In round r, for $r = 1, \ldots, \log n$, the users u, u' with parent w at level $(\log n - r)$ proceed as follows:

  1. $\mathsf{ck}_w \leftarrow \mathcal{KE}.\mathsf{CommonKey}(u, u')$;
  2. $(\mathsf{pt}_w, r_w) \leftarrow \mathcal{G}(\mathsf{ck}_w)$;
  3. If $u < u'$, set $u \stackrel{\mathsf{def}}{=} w$;

A similar construction is possible for the more efficient SD scheme. Due to lack of space, we present this construction in the Appendix I.6.4.

I.4.2 Dynamic Tree Construction

Dynamic CS. We define a join procedure for the CS tree described above. We go from $\mathcal{SC}_n$ to $\mathcal{SC}_{n+1}$ by taking the leaf u' with the lowest distance to the root, and if there are several with that property, the one with the lowest index. We then replace it with an inner node w, to which we append both the leaf u' and the new user v. We note that the user identifiers will not be in the same order as the node numbers in the tree.Then we replace the subsets S_j where j is an ancestor of the new user with the new subsets. This ensures that our dynamic CS scheme is forward-secure and has the splitting property of definition I.3.1. The CS key assignment is done as follows.

First the new user v derives a common key c_w with its sibling u'. From this common key, he derives the proto-key of node w and the randomness for the virtual user w to participate in the next round of key exchanges. The user with the smaller identifier then plays the role of w in the next round. This procedure is repeated until the keys of all ancestors of v are recomputed.

- $\mathsf{Join}(v, U_n)$ In the first round, set $u \stackrel{\mathsf{def}}{=} v$. In round r, for $r = 1, \ldots, \log(n+1)$, the user u with neighbour u' and parent w at level $(\log(n+1) - r)$ proceeds as follows:

  1. $\mathsf{ck}_w \leftarrow \mathcal{KE}.\mathsf{CommonKey}(u, u')$;
  2. $(\mathsf{pt}_w, r_w) \leftarrow \mathcal{G}(\mathsf{ck}_w)$;
  3. set $u \stackrel{\mathsf{def}}{=} w$, $u' \stackrel{\mathsf{def}}{=} \mathit{neighbour}(w)$, $w \stackrel{\mathsf{def}}{=} \mathit{parent}(w)$;

A similar construction is possible for the more efficient SD scheme. Due to lack of space, we present this construction in the Appendix I.6.4. We state exactly the security of the dynamic CS construction. Because of the similarities in the construction, a similar result can be obtained for SD.

Theorem I.4.1 Let $\mathcal{KE}$ be an IND-secure KE scheme with session keys in $\mathcal{K}_{\mathcal{KE}}$, and $\mathcal{G} : \mathcal{K}_{\mathcal{KE}} \to \mathcal{K} \times \mathcal{R}_{\mathcal{KE}}$ be a PRG. Then our dynamic CS construction of a $\mathcal{SKE}$ is IND-secure and

$$\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k, t, N, q_T) \leq (N \log N) \left(\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k, t) + \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k, t) \right).$$

The full proof can be found in the Appendix I.6.3.

I.4.3 Efficiency Properties

One of the main advantages of the NNL constructions [NNL01] is the efficient revocation with small ciphertext lengths ($\mathcal{O}(r \log N/r)$ for CS, $\mathcal{O}(2r-1)$ for SD) which is immediately inherited in our public-key scheme. The decryption key is the same length for CS, where each user has to store $\log N$ keys only, and longer ($\mathcal{O}(N \log N)$ for SD), where we cannot use the same key derivation.

In our scheme, for many instantiations of the 2-party key exchange, the private part of the decryption key can even be constant-size: each user keeps his secret random coins r_i, which is enough to iteratively generate all the private information from the public transcript of the key exchange protocols (stored in *Reg* or in the public key). Then, granted the key exchange scheme and $\log N$ public keys, each user can iteratively compute the decryption keys along the path to the root of the tree, and it is in this sense that the user random coins "contain" the keys used to decrypt, as required by the decapsulation algorithm.

Permanent Revocation. Because the length of the ciphertext for SC schemes depends on the number of revoked users, it is desirable to be able to completely remove users from a group. To permanently remove a user at leaf $2i$, we remove it and its sibling leaf $2i+1$ and simply move the user at $2i+1$ to be at node i which becomes a leaf. The keys of the user now at i remain the same as his own key before (at node $2i+1$) and we thus have to update the keys of all subsets in which the revoked user was a member. Concerning the security, it is easy to see that the user $2i$, not having the key of the user $2i+1$, can not learn anything about the updated keys, and this ensures the forward secrecy.

The only problem we face is that we need to keep the tree balanced. Fortunately, our constructions allow a re-organization of the tree in a very efficient manner. Indeed, the tree could be maintained to be an AVL tree at low cost [AVL62]. Whenever a user leaves the system and makes the tree unbalanced, by using $\log N$ rotations, we can re-balance the tree. Note that a rotation needs $\log N$ update operations at worst, so the total cost for a re-balancing is just $\log^2 N$ update operations at worst.

Merging Groups. Instead of joining a single user, we can also efficiently merge two existing groups by executing the key exchange protocol for their root nodes. This will allow every user in the two groups to compute the keys of the new root node.

I.5 Concrete Instantiations

We now give two instantiations of our scheme. The first one is probably the simplest possible case, and achieves IND-ACPA-security under the DDH-assumption. We use the Diffie-Hellman protocol [DH76] as our KE (where the users publish g^x and g^y from their random coins x and y, and get g^{xy} as common key) and ElGamal [ElG85] as the PKE where $\mathsf{ek} = g^{\mathsf{dk}}$, for a random scalar dk). A similar idea can be found in [KPT04], where the authors use a group key exchange protocol on a DH-tree. Because the random coin spaces of both protocols are identical, when we run both in the same group G of order q (scalars in $\mathbb{Z}_q$), if we only want to prove IND-ACPA-security, we can identify dk with the random coins for the key exchange, and thus ek is part of the transcript of the key exchange protocol, leaving us with a single key pair for both schemes. There are several alternatives for the PRG, the simplest one being a hash function modeled by a random oracle, to extract $\mathsf{dk} \in \mathbb{Z}_q$ from the proto-key $\mathsf{pt} \in G$. But we can avoid it, and even any computational assumption, by using a deterministic randomness extractor, as described in [CFGP06, Th. 7], that is a bijection and thus a perfect generator(see definition I.6.4):

Definition I.5.1 If $p = 2q+1$, and G is defined as the sub-group of the squares in $\mathbb{Z}_p^*$, then $\mathrm{ord}(G) = q$ and f is a bijection from G onto $\mathbb{Z}_q$: $f(x) = x$ (`if` $x \leq q$) or $p - x$ (`if` $x > q$).

The second instantiation is more involved. To achieve IND-ACCA-security, we use Cramer-Shoup encryption [CS98] as our PKE. Because the keys in Cramer-Shoup are larger, our KE is a 3-to-8 parallel Diffie-Hellman, where we use public and private keys consisting of three elements each to generate a shared key consisting of eight elements, which allows us to generate additional pseudo-randomness in each step. Our PRG is an embedding function $G^8 \to \mathbb{Z}_q^3 \times \mathbb{Z}_q^5$ that applies the above function f to all components. The first part in $\mathbb{Z}_q^3$ will be used again as random coins for the key exchange, whereas the second part in $\mathbb{Z}_q^5$ leads to the Cramer-Shoup decryption key. To counter malleability of our scheme, we also need a SUF-CMA-secure MAC scheme. As the first scheme, this one relies only on the DDH assumption.

When using the Cramer-Shoup PKE, the decryption key of node i is the tuple $\mathsf{dk}_i = (v_i, w_i, x_i, y_i, z_i)$, the corresponding encryption key ek_i is $(X_i, Y_i, H_i) = (g^{x_i} h^{v_i}, g^{y_i} h^{w_i}, g^{z_i})$. We

need to generate more pseudo-randomness than before, so we define a new key exchange that is essentially a parallel Diffie-Hellman.

Definition I.5.2 [3-8-DHKE] We define a modified Diffie-Hellman key exchange scheme.

- User i draws $a_i, b_i, c_i \xleftarrow{\$} \mathbb{Z}_q$, and sends $(A_i, B_i, C_i) = (g^{a_i}, g^{b_i}, g^{c_i})$;
- User j draws $a_j, b_j, c_j \xleftarrow{\$} \mathbb{Z}_q$, and sends $(A_j, B_j, C_j) = (g^{a_j}, g^{b_j}, g^{c_j})$;
- Then $\mathsf{ck} = (A_i^{a_j}, A_i^{b_j}, A_i^{c_j}, B_i^{a_j}, B_i^{b_j}, B_i^{c_j}, C_i^{a_j}, C_i^{b_j})$.

This easily defines the CommonKey protocol. Its key indistinguishability follows from the following theorem.

Theorem I.5.3 [3-8-DDH] Under the DDH assumption, it is infeasible to distinguish the 14-tuple
$(g^a, g^b, g^c, g^{a'}, g^{b'}, g^{c'}, g^{aa'}, g^{ab'}, g^{ac'}, g^{ba'}, g^{bb'}, g^{bc'}, g^{ca'}, g^{cb'})$ from a random 14-tuple even when given g, and $\mathrm{Adv}^{3-8-\mathsf{ddh}}(k,t) \leq 8 \cdot \mathrm{Adv}^{\mathsf{ddh}}(k, t + 11\tau_{exp})$, where τ_{exp} is the time for an exponentiation.

Proof: We define tuple T_0 to be the tuple as defined above, T_i as the same tuple with all "combined" elements up to the i-th one replaced by a random element. T_8 is therefore a tuple of 14 random elements. Given a distinguisher $\mathcal{A}$ between T_i and T_{i+1}, we construct a solver $\mathcal{B}$ for DDH as follows. Let $(X, Y, Z) = (g^x, g^y, g^z)$ be a DDH challenge tuple. Let $g^{de'}$ be the $i+1$-st combined element. $\mathcal{B}$ chooses a tuple T_i and replaces g^d with X, $g^{e'}$ with Y, and $g^{de'}$ with Y. All other combined elements can be constructed because at least one exponent is known, which takes 11 exponentiations ($11\tau_{exp}$) time. If $z = xy$, $T' = T_i$, else $T' = T_{i+1}$ and the theorem follows. ▌ As a PRG we use the PRG of definition I.5.1 on each component of the common key.

This gives us all the components we need to construct an IND-ACCA-secure BE scheme, whose security is based only on the DDH-assumption. (The DDH-assumption implies the existence of OWF, which is sufficient for MACs.)

Constant-Round Key Generation. While this construction achieves constant-size secrets for the users and requires very little interaction during the Join-procedure, it requires a logarithmic number of rounds for the subgroup key exchange protocol to complete. The Burmester-Desmedt group key exchange protocol[BD05] is, like the above scheme, passively secure in the standard model under the DDH assumption [KY07]. It requires only two rounds, and several instances could be run in parallel to compute keys for all subsets in two rounds. This would however require interaction between all the users each time a new users wants to join.

Acknowledgments

This work was supported by the French ANR-09-VERS-016 BEST Project and the European Commission through the ICT Programme under Contract ICT-2007-216676 ECRYPT II.

I.6 Appendix

I.6.1 Definitions

Definition I.6.1 [Encryption Scheme] A public-key encryption scheme is a 4-tuple of algorithms $\mathcal{PKE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$:

$\mathrm{Exp}^{\mathsf{ind-cca}-b}_{\mathcal{PKE},\mathcal{A}}(k)$
  $\mathsf{param} \leftarrow \mathsf{Setup}(1^k)$;
  $\mathcal{Q}_D \leftarrow \emptyset$, $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$;
  $(state, m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{ODecrypt}(\cdot)}(\mathsf{FIND}; \mathsf{param}, \mathsf{ek})$;
  $c^* \leftarrow \mathsf{Encrypt}(\mathsf{ek}, m_b)$;
  $b' \leftarrow \mathcal{A}^{\mathsf{ODecrypt}}(\mathsf{GUESS}, state; c^*)$;
  `if` $c^* \in \mathcal{Q}_D$ `then return` 0;
  `else return` b';

$\mathsf{ODecrypt}(c)$
  $\mathcal{Q}_D \leftarrow \mathcal{Q}_D \cup \{c\}$;
  $m \leftarrow \mathsf{Decrypt}(\mathsf{dk}, c)$;
  `return` m;

Figure I.3: $\mathcal{PKE}$: Semantic Security against Chosen-Ciphertext Attacks (IND-CCA)

- $\mathsf{Setup}(1^k)$, where k is the security parameter, generates the global parameters param of the system;
- $\mathsf{KeyGen}(\mathsf{param}; r)$ generates a pair of keys, the public (encryption) key ek and the associated private (decryption) key dk, using the random coins r (we may omit r when the notation is obvious);
- $\mathsf{Encrypt}(\mathsf{ek}, m; r)$ produces a ciphertext c on the input message m and the public key ek, using the random coins r (we may omit r when the notation is obvious);
- $\mathsf{Decrypt}(\mathsf{dk}, c)$ decrypts the ciphertext c under the private key dk. It outputs the plaintext, or $\perp$ if the ciphertext is invalid.

We require that $\mathsf{Decrypt}(\mathsf{dk}, \mathsf{Encrypt}(\mathsf{ek}, m)) = m$ if $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ for some parameters.

Such an encryption scheme is said to be (t, q_D, ε)-IND-CCA-secure (semantic security against chosen-ciphertext attacks) if in the security game presented in figure I.3, the advantage, denoted $\mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D)$, of any t-time adversary $\mathcal{A}$ asking at most q_D decryption queries to the ODecrypt oracle is bounded by ε:

$$\mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{ind-cca}-1}_{\mathcal{PKE},\mathcal{A}}(k) = 1] - \Pr[\mathrm{Exp}^{\mathsf{ind-cca}-0}_{\mathcal{PKE},\mathcal{A}}(k) = 1]\}.$$

This definition includes IND-CPA (for Chosen-Plaintext Attacks) when $q_D = 0$.

Definition I.6.2 [Two-Party Key Exchange] A two-party key exchange protocol is a 2-tuple $\mathcal{KE} = (\mathsf{Setup}, \mathsf{CommonKey})$:

- $\mathsf{Setup}(1^k)$, where k is the security parameter, generates the global parameters param of the system;
- $\mathsf{CommonKey}(u, v)$ is an interactive protocol between two users u and v. Both take as private input their random coins, and obtain a common key ck.

We require that users u and v that run $\mathsf{CommonKey}(u, v)$ both get the same ck.

For the sake of clarity, we might omit param in the rest of the paper, but global parameters are always implicit for all the primitives. Such a key exchange scheme is said to be (t, ε)-IND-secure (semantic security or key indistinguishability) if in the security game presented in figure I.4, the

$\mathrm{Exp}^{\mathsf{ind}-b}_{\mathcal{KE},\mathcal{A}}(k)$
  $\mathsf{param} \leftarrow \mathsf{Setup}(1^k)$;
  $(K, \tau) \leftarrow \mathsf{CommonKey}(u, v)$; $K_b \leftarrow K$; $K_{1-b} \xleftarrow{\$} \mathcal{K}$;
  $b' \leftarrow \mathcal{A}(\tau, K_0, K_1)$;
  `return` b';

Figure I.4: $\mathcal{KE}$: Key Indistinguishability (IND)

advantage $\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k, t)$ of any t-time adversary $\mathcal{A}$ is bounded by ε, where the adversary gets the transcript τ of the communications between u and v during the execution of CommonKey:

$$\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k, t) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{ind}-1}_{\mathcal{KE},\mathcal{A}}(k) = 1] - \Pr[\mathrm{Exp}^{\mathsf{ind}-0}_{\mathcal{KE},\mathcal{A}}(k) = 1]\}.$$

In our construction, we will need two additional classical primitives: a message authentication code and pseudo-random functions.

Definition I.6.3 [Message Authentication Code] A message authentication code is a 3-tuple of algorithms $\mathcal{MAC} = (\mathsf{KeyGen}, \mathsf{GenMac}, \mathsf{VerifMac})$:

- $\mathsf{KeyGen}(1^k)$, where k is the security parameter, generates a secret key $\mathsf{sk} \xleftarrow{\$} \mathcal{K}_m$.
- $\mathsf{GenMac}(\mathsf{sk}, m)$ takes as input the secret key and a message, and generates the MAC value σ.
- $\mathsf{VerifMac}(\mathsf{sk}, m, \sigma)$ takes as input the secret key, the message and the alleged signature. It checks the validity of the signature and returns 1 if it is valid, 0 else.

In the following, we will require the strong unforgeability of a one-time MAC: even after one MAC generation query, the adversary cannot generate a new valid pair, even for the already authenticated message. This strong unforgeability is formalized in the security game presented in figure I.5, where the adversary wins if it successfully verifies a pair that has not been generated by the authentication algorithm. Such a message authentication code is said to be $(t, q_M, q_V, \varepsilon)$-SUF-CMA-secure (strong existential unforgeability against chosen-message attacks) if in the security game presented in figure I.5, the success probability $\mathsf{Succ}^{\mathsf{suf}-\mathsf{cma}}_{\mathcal{MAC}}(k, t, q_M, q_V)$ of any t-time adversary $\mathcal{A}$, asking at most q_M MAC values (OGenMac oracle) and q_V verifications (OVerifMac oracle) is bounded by ε:

$$\mathsf{Succ}^{\mathsf{suf}-\mathsf{cma}}_{\mathcal{MAC}}(k, t, q_M, q_V) = \max_{\mathcal{A}}\{\Pr[\mathrm{Exp}^{\mathsf{suf}-\mathsf{cma}}_{\mathcal{MAC},\mathcal{A}}(k) = 1]\}.$$

This definition includes one-time MAC when $q_M = 1$.

Definition I.6.4 [Pseudo-Random Generator] A generator $\mathcal{F} : X \to Y$ is (t, ε)-pseudo-random if the advantage, denoted $\mathrm{Adv}^{\mathsf{prg}}_{\mathcal{F}}(k, t)$, of any t-time adversary $\mathcal{A}$ is bounded by ε:

$$\mathrm{Adv}^{\mathsf{prg}}_{\mathcal{F}}(k, t) = \max_{\mathcal{A}}\{\Pr[\mathcal{A}(y) = 1 \,|\, y \xleftarrow{\$} Y] - \Pr[\mathcal{A}(\mathcal{F}(x)) = 1 \,|\, x \xleftarrow{\$} X]\}.$$

In the following, Y may be the product of two sets $Y_1 \times Y_2$. We will then parse $\mathcal{F}(x) = (f_1(x), f_2(x))$. If $\mathcal{F}$ is a bijection (which implies that the PRG is not expanding), then $\mathcal{F}$ is a perfect generator, with $\varepsilon = 0$ and no computational assumption.

$\mathrm{Exp}^{\mathsf{suf-cma}}_{\mathcal{MAC},\mathcal{A}}(k)$
  $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^k)$;
  $\mathcal{Q}_S \leftarrow \emptyset$; $\mathcal{Q}_V \leftarrow \emptyset$;
  $\mathcal{A}^{\mathsf{OGenMac}(\cdot),\mathsf{OVerifMac}(\cdot,\cdot)}(1^k)$;
  `if` $\exists(m,\sigma) \in \mathcal{Q}_V, (m,\sigma) \notin \mathcal{Q}_S$ `then return` 1;
  `else return` 0;

$\mathsf{OGenMac}(m)$
  $\sigma \leftarrow \mathsf{GenMac}(\mathsf{sk}, m)$;
  $\mathcal{Q}_S \leftarrow \mathcal{Q}_S \cup \{(m,\sigma)\}$
  `return` σ;

$\mathsf{OVerifMac}(m,\sigma)$
  $c = \mathsf{VerifMac}(\mathsf{sk}, m, \sigma)$;
  `if` $c = 1$ `then` $\mathcal{Q}_V \leftarrow \mathcal{Q}_V \cup \{(m,\sigma)\}$;
  `return` c;

Figure I.5: $\mathcal{MAC}$: Unforgeability (SUF-CMA)

I.6.2 Proof of Theorem I.3.3

We assume that $\mathcal{A}$ is an adversary against the IND-ACCA security game. We define a sequence of games, $\mathbf{G}_2, \ldots, \mathbf{G}_{11}$, where $\mathbf{G}_2$ is the IND-ACCA experiment with $b = 0$ and $\mathbf{G}_{11}$ is the IND-ACCA experiment with $b = 1$. Let ℓ be the number of components in a challenge ciphertext (the size of the partition $\mathcal{L}^*$ of the challenge target set S^*). By definition, ℓ is not greater than ℓ_N.

Game $\mathbf{G}_2$: This is the IND-ACCA-Experiment with $b = 0$. We just recall the generation of the challenge ciphertext (the Encaps oracle), and the simulation of the ODecaps oracle:

$\mathsf{Setup}(1^k)$:

1. Run $\mathcal{PKE}.\mathsf{Setup}(1^k)$ to get $\mathsf{param}_{\mathcal{PKE}}$;
2. Run $\mathcal{SKE}.\mathsf{Setup}(1^k)$ to get $\mathsf{param}_{\mathcal{SKE}}$;
3. Publish $\mathsf{param} = (\mathsf{param}_{\mathcal{PKE}}, \mathsf{param}_{\mathcal{SKE}})$.

KeyGen(param, U_n):

1. All the proto-keys pt_S, for all the subsets S in $\mathcal{SC}_n$, are generated using the $\mathcal{SKE}$.KeyGen protocol;
2. Each user $u \in U_n$ gets the proto-keys for all subsets S he belongs to. The decryption key dk_u consists of all these pt_S;
3. He computes $(\mathsf{dk}_S, \mathsf{ek}_S) \leftarrow \mathcal{PKE}.\mathsf{KeyGen}(\mathsf{param}_{\mathcal{PKE}}; \mathcal{F}(\mathsf{pt}_S))$, where we use the PRG to generate the random coins of the key generation algorithm;
4. The adversary receives the transcript of the execution of the $\mathcal{SKE}$.KeyGen protocol.

Join($v, \{u(\mathsf{dk}_u)\}_{u \in U_n}, Reg, \mathsf{EK}$): similar to KeyGen

Encaps(EK, Reg, S^*):

1. From the target set S^*, generate a partition $S^* = \cup_{\mathcal{L}^*} S_i$, we assume of size ℓ;
2. Generate two session keys $\mathcal{K}_e^0$ and $\mathcal{K}_e^1$, as well as a MAC key $\mathcal{K}_m^0$;
3. For each subset $i \in \mathcal{L}^*$, generate $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_{S_i}, \mathcal{K}_e^0 || \mathcal{K}_m^0)$;
4. Then, compute $\sigma^* = \mathcal{MAC}.\mathsf{GenMac}(\mathcal{K}_m^0, S^* || (c_i^*)_{i \in \mathcal{L}^*})$;
5. Outputs $\mathcal{K}_e^0$, $\mathcal{K}_e^1$ and $H^* = ((c_i^*)_{i \in \mathcal{L}^*}, \sigma^*)$.

ODecaps(u, S, H):

1. If u is in S, then there is a unique i such that $u \in S_i$, and then dk_u allows to derive dk_{S_i};
2. Extract $\mathcal{K}_e || \mathcal{K}_m = \mathcal{PKE}.\mathsf{Decrypt}(\mathsf{dk}_{S_i}, c_i)$;
3. If $i \in \mathcal{L}^*$ and $c_i = c_i^*$, check if σ is a valid MAC under key $\mathcal{K}_m$;
4. Else, check if σ is a valid MAC under key $\mathcal{K}_m$;
5. In case of validity, output $\mathcal{K}_e$, otherwise output $\perp$.

Game $\mathbf{G}_3$: We first replace all the proto-keys by random keys: we thus apply the key indistinguishability of the $\mathcal{SKE}$ scheme:

KeyGen(param, U_n):

1. All the proto-keys pt_S are drawn independently at random for all subsets S;

The difference between $\mathbf{G}_3$ and $\mathbf{G}_2$ is bounded by

$$\Pr_3[\mathcal{A} \to 1] - \Pr_2[\mathcal{A} \to 1] \leq \mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k, t, L_N, L_N).$$

Game $\mathbf{G}_4$: We now replace all PKE keys by random keys: we thus apply the pseudo-randomness of the PRG $\mathcal{F}$:

> $\mathsf{KeyGen}(\mathsf{param}, U_n)$:
>
> 3. Each user gets $(\mathsf{dk}_S, \mathsf{ek}_S) \leftarrow \mathcal{PKE}.\mathsf{KeyGen}(\mathsf{param}_{\mathcal{PKE}}; r_S)$, where r_S are random coins, for all subsets S he belongs to;

Using a classical hybrid proof, the difference between $\mathbf{G}_4$ and $\mathbf{G}_3$ is bounded by

$$\Pr_4[\mathcal{A} \to 1] - \Pr_3[\mathcal{A} \to 1] \leq L_N \times \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{F}}(k, t).$$

Game $\mathbf{G}_5$: We introduce an additional MAC key that will be used later in the sub-ciphertexts:

> $\mathsf{Encaps}(\mathsf{EK}, Reg, S^*)$:
>
> 2. Generate two session keys $\mathcal{K}_e^0$ and $\mathcal{K}_e^1$, as well as two MAC keys $\mathcal{K}_m^0$ and $\mathcal{K}_m^1$;

$\mathbf{G}_5$ and $\mathbf{G}_4$ are perfectly indistinguishable:

$$\Pr_5[\mathcal{A} \to 1] = \Pr_4[\mathcal{A} \to 1].$$

Game $\mathbf{G}_6$: We now use the additional MAC key $\mathcal{K}_m^1$ in the challenge sub-ciphertexts, but still use $\mathcal{K}_m^0$ for the MAC computation:

> $\mathsf{Encaps}(\mathsf{EK}, Reg, S^*)$:
>
> 3. For each subset $i \in \mathcal{L}^*$, generate $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_i, \mathcal{K}_e^0 || \mathcal{K}_m^1)$;

Lemma I.6.5 The difference between $\mathbf{G}_6$ and $\mathbf{G}_5$ is bounded by

$$\Pr_6[\mathcal{A} \to 1] - \Pr_5[\mathcal{A} \to 1] \leq \ell \times L_N \times \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D).$$

Game $\mathbf{G}_7$: In this game, we reject decryption queries that should decrypt a sub-ciphertext from the challenge ciphertext.

> $\mathsf{ODecaps}(u, S, H = ((c_i)_{i \in \mathcal{L}}, \sigma))$:
>
> 3. If $i \in \mathcal{L}^*$ and $c_i = c_i^*$, output $\perp$;

Lemma I.6.6 The difference between $\mathbf{G}_7$ and $\mathbf{G}_6$ is bounded by

$$\Pr_7[\mathcal{A} \to 1] - \Pr_6[\mathcal{A} \to 1] \leq \mathsf{Succ}^{\mathsf{suf-cma}}_{\mathcal{MAC}}(k, t, 1, q_D).$$

Game $\mathbf{G}_8$: We define the game $\mathbf{G}_8$ as the game $\mathbf{G}_7$, but we encapsulate $\mathcal{K}_e^1$ instead of $\mathcal{K}_e^0$:

> $\mathsf{Encaps}(\mathsf{EK}, Reg, S^*)$:
>
> 3. For each subset $i \in \mathcal{L}^*$, generate $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_{S_i}, \mathcal{K}_e^1 || \mathcal{K}_m^1)$;

Lemma I.6.7 The difference between $\mathbf{G}_8$ and $\mathbf{G}_7$ is bounded by

$$\Pr_8[\mathcal{A} \to 1] - \Pr_7[\mathcal{A} \to 1] \leq \ell \times L_N \times \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D).$$

Game $\mathbf{G}_9$: Previous game is similar to $\mathbf{G}_7$, but with $\mathcal{K}_e^1$ in the challenge ciphertext. We now go back, as in game $\mathbf{G}_6$: we check MAC values of sub-ciphertexts of the challenge ciphertext under $\mathcal{K}_m^0$:

> $\mathsf{ODecaps}(u, S, H)$:
>
> 3. If $i \in \mathcal{L}^*$ and $c_i = c_i^*$, check if σ is a valid MAC under key $\mathcal{K}_m^0$.

Since we have the same gap as from $\mathbf{G}_6$ to $\mathbf{G}_7$:

$$\Pr_9[\mathcal{A} \to 1] - \Pr_8[\mathcal{A} \to 1] \leq \mathsf{Succ}^{\mathsf{suf-cma}}_{\mathcal{MAC}}(k, t, 1, q_D).$$

Game $\mathbf{G}_{10}$: We eventually change back the use of the MAC key $\mathcal{K}_m^0$ in the challenge sub-ciphertexts:

> $\mathsf{Encaps}(\mathsf{EK}, Reg, S^*)$:
>
> 3. For each subset $i \in \mathcal{L}^*$, generate $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_{S_i}, \mathcal{K}_e^1 || \mathcal{K}_m^0)$;

Since we have the same gap as from $\mathbf{G}_5$ to $\mathbf{G}_6$:

$$\Pr_{10}[\mathcal{A} \to 1] - \Pr_9[\mathcal{A} \to 1] \leq \ell \times L_N \times \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D).$$

We do not use anymore the key $\mathcal{K}_m^1$: this is exactly the IND-ACCA security game with $b = 1$, except for the generation of the encryption keys.

Game $\mathbf{G}_{11}$: We now change back the generation of the encryption keys, using the $\mathcal{SKE}$ protocol and the PRG:

$$\Pr_{11}[\mathcal{A} \to 1] - \Pr_{10}[\mathcal{A} \to 1] \leq \mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k, t, L_N, L_N) + L_N \times \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{F}}(k, t).$$

If we sum up all the gaps, we obtain:

$$\begin{aligned}
\Pr_3[\mathcal{A} \to 1] - \Pr_2[\mathcal{A} \to 1] &\leq \mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k, t, L_N, L_N) \\
\Pr_4[\mathcal{A} \to 1] - \Pr_3[\mathcal{A} \to 1] &\leq L_N \times \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{F}}(k, t) \\
\Pr_6[\mathcal{A} \to 1] - \Pr_4[\mathcal{A} \to 1] &\leq \ell \times L_N \times \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D) \\
\Pr_7[\mathcal{A} \to 1] - \Pr_6[\mathcal{A} \to 1] &\leq \mathsf{Succ}^{\mathsf{suf-cma}}_{\mathcal{MAC}}(k, t, 1, q_D) \\
\Pr_8[\mathcal{A} \to 1] - \Pr_7[\mathcal{A} \to 1] &\leq \ell \times L_N \times \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{E}}(k, t, q_D) \\
\Pr_9[\mathcal{A} \to 1] - \Pr_8[\mathcal{A} \to 1] &\leq \mathsf{Succ}^{\mathsf{suf-cma}}_{\mathcal{MAC}}(k, t, 1, q_D) \\
\Pr_{10}[\mathcal{A} \to 1] - \Pr_9[\mathcal{A} \to 1] &\leq \ell \times L_N \times \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k, t, q_D) \\
\Pr_{11}[\mathcal{A} \to 1] - \Pr_{10}[\mathcal{A} \to 1] &\leq \mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k, t, L_N, L_N) + L_N \times \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{F}}(k, t)
\end{aligned}$$

And this concludes the proof, since $\ell \leq \ell_N$.

Proof of Lemma I.6.5.

Let ℓ be the size of the partition $\mathcal{L}^*$. In order to so show that the adversary cannot detect whether we use the same MAC key that is part of the ciphertext or not, we proceed by another sequence of hybrid games: We define the game G_j (for $j = 0, \ldots, \ell$), in which the j-first sub-ciphertexts c_i^* are defined as in $\mathbf{G}_6$, that is $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_i, \mathcal{K}_e^0||\mathcal{K}_m^1)$, and the next ones are defined as in $\mathbf{G}_5$, that is $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_{S_i}, \mathcal{K}_e^0||\mathcal{K}_m^0)$. It is clear that $G_0 = \mathbf{G}_5$, whereas $G_\ell = \mathbf{G}_6$.

For any $J \in [0, \ell]$, let us play the following game against the IND-CCA challenger of the $\mathcal{PKE}$ encryption scheme:

- Setup/KeyGen:
  1. We receive the challenge public key ek;
  2. We randomly choose one subset $I \in [1, L_N]$ (we bet it will correspond to the J-th ciphertext in the target partition $\mathcal{L}^*$. This guess is correct with probability $1/L_N$, otherwise we abort the game and make $\mathcal{B}$ output 0);
  3. We generate all the pairs $(\mathsf{dk}_{S_i}, \mathsf{ek}_{S_i})$ at random, except for $i = I$, where $\mathsf{ek}_{S_I} = \mathsf{ek}$;
- Encaps(EK, Reg, S^*):
  1. From the target set S^*, generate a partition $S^* = \cup_{\mathcal{L}^*} S_i$, we assume of size ℓ;
  2. If our guess at setup time was correct, the J-th element in $\mathcal{L}^*$ is I;
  3. Generate two session keys $\mathcal{K}_e^0$ and $\mathcal{K}_e^1$, as well as two MAC keys $\mathcal{K}_m^0$ and $\mathcal{K}_m^1$;
  4. For the $J-1$-first elements $i \in \mathcal{L}^*$, generate $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_{S_i}, \mathcal{K}_e^0||\mathcal{K}_m^1)$;
  5. For the J-th element $i \in \mathcal{L}^*$, assumed to be I, ask to the IND-CCA-challenger on the two plaintexts $\mathcal{K}_e^0||\mathcal{K}_m^0$ and $\mathcal{K}_e^0||\mathcal{K}_m^1$, and set c_I^* to be the answer, according to the internal bit b of the IND-CCA challenger;
  6. For the next elements $i \in \mathcal{L}^*$, generate $c_i^* = \mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_{S_i}, \mathcal{K}_e^0||\mathcal{K}_m^0)$;
  7. Then, compute $\sigma^* = \mathcal{MAC}.\mathsf{GenMac}(\mathcal{K}_m^0, S^*||(c_i^*)_{i\in\mathcal{L}^*})$;
  8. Output $\mathcal{K}_e^0$, $\mathcal{K}_e^1$ and $H^* = ((c_i^*)_{i\in\mathcal{L}^*}, \sigma^*)$.
- OCorrupt Queries: Since we condition on the good choice for I, we can answer all the OCorrupt queries by outputting the corresponding decryption keys (they cannot be for I, otherwise the challenge target set would contain corrupted players);
- ODecaps Queries:
  1. If this is for a player that lies in a set $S_i \notin S_I$, we can easily decrypt c_i;
  2. If this is for a player that lies in S_I,
     - either $c_I \neq c_I^*$, and then we can ask the decryption query to the decryption oracle
     - or $c_I = c_I^*$, then check the MAC value with $\mathcal{K}_m^0$. If it is valid, output $\mathcal{K}_e^0$, otherwise output $\perp$.

Our adversary $\mathcal{B}$ against IND-CCA simply forwards the output b' of $\mathcal{A}$ (or outputs zero in case of abort):

$$\begin{aligned}\Pr[\mathcal{B}\to 1|b=0]-\Pr[\mathcal{B}\to 1|b=1] &= \Pr[\mathcal{B}\to 1\wedge I|b=0]-\Pr[\mathcal{B}\to 1\wedge I|b=1]\\ &\quad+\Pr[\mathcal{B}\to 1\wedge \neg I|b=0]-\Pr[\mathcal{B}\to 1\wedge \neg I|b=1]\\ &= \frac{1}{L_N}\times(\Pr[\mathcal{B}\to 1|b=0\wedge I]-\Pr[\mathcal{B}\to 1|b=1\wedge I])\\ &= \frac{1}{L_N}\times(\Pr[\mathcal{A}\to 1|b=0\wedge I]-\Pr[\mathcal{A}\to 1|b=1\wedge I])\,.\end{aligned}$$

In the RHS, the output is independent of the correct guess of I, whereas the LHS is bounded by the best advantage against IND-CCA within time t:

$$\frac{1}{L_N}\times|\Pr[\mathcal{A}\to 1|b=0]-\Pr[\mathcal{A}\to 1|b=1]|\le \mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k,t,q_D).$$

Furthermore, the above game with $b=0$ is exactly G_{J-1}, whereas with $b=1$ this is G_J:

$$\left|\Pr_{G_{J-1}}[\mathcal{A}\to 1]-\Pr_{G_J}[\mathcal{A}\to 1]\right|\le L_N\times\mathrm{Adv}^{\mathsf{ind-cca}}_{\mathcal{PKE}}(k,t,q_D).$$

This concludes the proof.

Proof of Lemma I.6.6.

In order to show that the adversary cannot detect whether we reject a valid MAC under the unknown key $\mathcal{K}_m^0$, we use the following game against the MAC: more precisely, we play the SUF-CMA security game against the MAC, using the challenge MAC key sk as the unknown $\mathcal{K}_m^0$ key. All the other keys are known to the simulator. The MAC generation oracle OGenMac is called for the challenge MAC value by the Encaps oracle, and the MAC verification oracle OVerifMac is called in case of a challenge sub-ciphertext in a decapsulation ODecaps oracle query. A valid MAC value asked to OVerifMac is a forgery, otherwise it should be a reject. Hence, the probability that a valid MAC value is refused is bounded by $\mathsf{Succ}^{\mathsf{suf-cma}}_{\mathcal{MAC}}(k,t,1,q_D)$.

Proof of Lemma I.6.7.

Let ℓ be the size of the partition $\mathcal{L}^*$. In order to so show that the adversary cannot detect whether we encrypt $\mathcal{K}_e^0$ or $\mathcal{K}_e^1$, we proceed as for the proof of Lemma I.6.5, by a sequence of hybrid games: We define the game G_j (for $j=0,\ldots,\ell$), in which the j-first sub-ciphertexts c_i^* are defined as in $\mathbf{G}_8$, that is $c_i^*=\mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_i,\mathcal{K}_e^1||\mathcal{K}_m^1)$, and the next ones are defined as in $\mathbf{G}_7$, that is $c_i^*=\mathcal{PKE}.\mathsf{Encrypt}(\mathsf{ek}_i,\mathcal{K}_e^0||\mathcal{K}_m^1)$. It is clear that $G_0=\mathbf{G}_7$, whereas $G_\ell=\mathbf{G}_8$. Exactly the same analysis as in the proof of Lemma I.6.5 leads to the result. The trick comes from the simulation of ODecaps Queries, in which we output $\perp$ in case a sub-ciphertext of the challenge ciphertext is involved. We do not have to care whether this is $\mathcal{K}_e^0$ or $\mathcal{K}_e^1$.

I.6.3 Proof of Theorem I.4.1

Let $\mathcal{A}$ be an adversary against the IND-security of our CS construction $\mathcal{SKE}$ that invokes at most N users, among them, the user set U_M that runs the KeyGen protocol (where $M=|U_M|$ denotes its size) and T users that join once at a time, in T time steps. Since in each of the T time periods at most $\log N$ nodes are updated, at most $N\log N$ keys will be generated overall (for $M=1$). Game $\mathrm{Exp}^{\mathsf{ind}-0}_{\mathcal{SKE},\mathcal{A}}(k)$ is the experiment where all keys are generated as usual.

This will be our initial game. Game $\mathrm{Exp}^{\mathsf{ind}-1}_{\mathcal{SKE},\mathcal{A}}(k)$ is the experiment where all keys are chosen uniformly at random. This will be our final game. To go from the first game to the final one, we define intermediate games, in which we first replace the session keys that are produced by the two-player key exchange protocols by random keys, and then we replace the proto-keys by random keys.

Game $\mathbf{G}_0$: This is the initial game, that appears in the experiment where $b = 0$.

> $\mathsf{KeyGen}(U_M)$: In round r, for $r = \log M, \ldots, 1$, the simulator executes the following steps for each node w at level $(\log M - r)$ of the tree with children u, u':
>
> 1. $\mathsf{ck}_w \leftarrow \mathcal{KE}.\mathsf{CommonKey}(u, u')$;
> 2. $(\mathsf{pt}_w, r_w) \leftarrow \mathcal{G}(\mathsf{ck}_w)$;
> 3. If $u < u'$, set $u \stackrel{\mathsf{def}}{=} w$.

> $\mathsf{Join}(v, U_n)$ In the first round, set $u = v$. In round r, for $r = \log(n+1), \ldots, 1$, the simulator executes the following steps for user u with neighbour u' and parent w at level $(\log(n+1) - r)$:
>
> 1. $\mathsf{ck}_w \leftarrow \mathcal{KE}.\mathsf{CommonKey}(u, u')$;
> 2. $(\mathsf{pt}_w, r_w) \leftarrow \mathcal{G}(\mathsf{ck}_w)$;
> 3. set $u \stackrel{\mathsf{def}}{=} w$, $u' \stackrel{\mathsf{def}}{=} \mathit{neighbour}(w)$, $w \stackrel{\mathsf{def}}{=} \mathit{parent}(w)$;

Game $\mathbf{G}_1$: We replace all KE session keys on level 1 of the tree with random keys.

> $\mathsf{KeyGen}(U_M)$: In round 1, the simulator executes the following steps for each node w at level 1 of the tree with children u, u':
>
> 1. $\mathsf{ck}_w \leftarrow \mathcal{KE}.\mathsf{CommonKey}(u, u')$; $\mathsf{ck}_w \stackrel{\$}{\leftarrow} \mathcal{K}_{\mathcal{KE}}$;

With a classical hybrid proof, where we successively replace all the real keys by random keys in the $M/2$ two-party key exchanges, we get that the difference between $\mathbf{G}_1$ and $\mathbf{G}_0$ is bounded by

$$\Pr_1[\mathcal{A} \to 1] - \Pr_0[\mathcal{A} \to 1] \leq \frac{M}{2}\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k, t).$$

Game $\mathbf{G}_2$: We replace all proto-keys on level 1 of the tree with random keys.

> $\mathsf{KeyGen}(U_M)$: In round 1, the simulator executes the following steps for each node w at level 1 of the tree with children u, u':
>
> 2. $(\mathsf{pt}_w, r_w) \stackrel{\$}{\leftarrow} \mathcal{K} \times \mathcal{R}_{\mathcal{KE}}$;

With a classical hybrid proof, where we successively replace all the real values by random values in the $M/2$ key derivations, we get that the difference between $\mathbf{G}_2$ and $\mathbf{G}_1$ is bounded by

$$\Pr_2[\mathcal{A} \to 1] - \Pr_1[\mathcal{A} \to 1] \leq \frac{M}{2}\mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k, t).$$

Game $\mathbf{G}_3$: We replace all proto-keys in the initial tree with random keys.

> $\mathsf{KeyGen}(U_M)$: In round r, the simulator executes the following steps for each node w at level r of the tree with children u, u':
>
> 1. $\mathsf{ck}_w \leftarrow \mathcal{KE}.\mathsf{CommonKey}(u, u')$; $\mathsf{ck}_w \xleftarrow{\$} \mathcal{K}_{\mathcal{KE}}$;
> 2. $(\mathsf{pt}_j, r_j) \xleftarrow{\$} \mathcal{K} \times \mathcal{R}_{\mathcal{KE}}$;

By applying iteratively the 2 previous hops at level 2 on $M/2^2$ pairs, and at level 3 on $M/2^3$ pairs, etc, we get that the difference between $\mathbf{G}_3$ and $\mathbf{G}_2$ is bounded by

$$\Pr_3[\mathcal{A} \to 1] - \Pr_2[\mathcal{A} \to 1] \leq (\frac{M}{2} - 1) \left(\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k,t) + \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k,t) \right).$$

Game $\mathbf{G}_4$: We replace all proto-keys created during joins with random keys. The result is a protocol where all proto-keys are drawn independently at random, which describes the experiment with $b = 1$.

> $\mathsf{Join}(v, U_n)$
>
> 1. $\mathsf{ck}_w \leftarrow \mathcal{KE}.\mathsf{CommonKey}(u, u')$; $\mathsf{ck}_w \xleftarrow{\$} \mathcal{K}_{\mathcal{KE}}$;
> 2. $(\mathsf{pt}_j, r_j) \xleftarrow{\$} \mathcal{K} \times \mathcal{R}_{\mathcal{KE}}$;

Lemma I.6.8 The difference between $\mathbf{G}_4$ and $\mathbf{G}_3$ is bounded by

$$\Pr_4[\mathcal{A} \to 1] - \Pr_3[\mathcal{A} \to 1] \leq (T \log N) \left(\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k,t) + \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k,t) \right).$$

In summary, we have

$$\Pr_3[\mathcal{A} \to 1] - \Pr_0[\mathcal{A} \to 1] \leq (M-1) \left(\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k,t) + \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k,t) \right)$$
$$\Pr_4[\mathcal{A} \to 1] - \Pr_3[\mathcal{A} \to 1] \leq (T \log N) \left(\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k,t) + \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k,t) \right).$$

Because $M + T = N$, we obtain

$$\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{SKE}}(k,t,N,q_T) \leq (N \log N) \left(\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k,t) + \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k,t) \right).$$

Note that this is independent of q_T, because we change all the keys.

Proof of Lemma I.6.8.

Let G_0 be the game $\mathbf{G}_3$, G_T be the game $\mathbf{G}_4$. We define $T-1$ intermediate hybrid games G_j ($j = 1 \ldots T-1$), in which we replace all session keys and proto-keys computed during the first j joins with random keys. We proceed as in the previous proofs and obtain

$$\Pr_j[\mathcal{A} \to 1] - \Pr_{j-1}[\mathcal{A} \to 1] \leq \log N \cdot \left(\mathrm{Adv}^{\mathsf{ind}}_{\mathcal{KE}}(k,t) + \mathrm{Adv}^{\mathsf{prg}}_{\mathcal{G}}(k,t) \right).$$

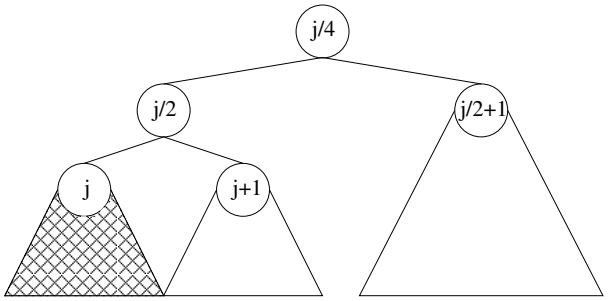


Figure I.6: SD Key Assignment for $S_{j/4,j}$

I.6.4 Constructions for the Subset Difference Method

SD Tree. We can modify the construction of the above CS tree to obtain keys for any subset $S_{i,j} = S_i \setminus S_j$, when $S_j \subset S_i$: To exclude all leaves below node j (w.l.o.g. $j \equiv 0 \mod 4$), we skip the key exchange between j and $j+1$ and directly compute a common key $\mathsf{ck}_{j/2+1,j+1}$ between $j/2+1$ (j's uncle) and $j+1$ (j's sibling). Basically, we identify the public key of $j+1$ with that of $S_{j/2,j}$. After applying $\mathcal{G}$, we have the two key pairs for $S_{j/4,j}$, which we treat as being at node $j/4$. We then continue with $\mathcal{KE}.\mathsf{CommonKey}$ and $\mathcal{G}$ as normal up to node i, to get $S_{i,j}$. This allows us to construct an SD tree in much the same way as a CS tree, except that we "omit" one node in the computation of the key (see figure I.6). Each node i at depth ℓ ($2^\ell \leq i < 2^{\ell+1}$) contains $2^{d-\ell+1} - 4$ blocks of keys that can be computed iteratively, excluding all the possible subtrees, from depth $\ell + 2$ (4 of them) to d ($2^{d-\ell}$ of them).

We define the neighbour and parent of user i as for the CS scheme, and the neighbour of (i, j) to be the neighbour of i. At round r, each user (i, j) created in round $r-1$ has a uniquely defined neighbour i' (if he does not, the protocol run is completed). They both have random coins r_i and $r_{i'}$:

- If $j \neq i'$, it runs $\mathsf{ck}_{i/2,j} \leftarrow \mathcal{KE}.\mathsf{CommonKey}((i,j)(r_i), i'(r_{i'}))$ and sets $(\mathsf{pt}_{i/2}, r_{i/2,j}) \leftarrow \mathcal{G}(c_{i/2,j})$ to derive the information for the node $(i/2, j)$.
- If $j = i'$, it runs $\mathsf{ck}_{i/2} \leftarrow \mathcal{KE}.\mathsf{CommonKey}((i,j)(r_i), i'(r_{i'}))$ and sets $(\mathsf{pt}_{i/2}, r_{i/2}) \leftarrow \mathcal{G}(c_{i/2})$ to derive the information for the node $i/2$.

If $i < i'$, it plays the role of the virtual user $i/2$ in the next round.

Dynamic SD. To join a user, we go from $\mathcal{SC}_n$ to $\mathcal{SC}_{n+1}$ by appending the new user as described for CS. Then we replace those subsets $S_{i,j}$ that contain the new user with the new subsets.

We show that our dynamic SD scheme has the splitting property of definition I.3.1. All $S_{i,j}$ for which i is not an ancestor of the new node are unchanged. All $S_{i,j}$ for which i, but not j is an ancestor of the new node contain the new user. All $S_{i,j}$ for which i is an ancestor of the new node and j is a true ancestor of the new node are unchanged as well. All $S_{i,j}$ for which i is an ancestor and j is the new node correspond to full subtrees $S_{\text{parent}(i),\text{sibling}(i)}$ in the old subset cover. The key assignment for SD is similar to the CS key assignment, but we cannot identify nodes and subsets, and must "jump" the omitted subtree in the computation (figure I.6).

Bibliography

[AA07] Noga Alon and Vera Asodi. Tracing many users with almost no rate penalty. *IEEE Trans. Inform. Theory*, 53(1):437–439, 2007. (Cited on page 83.)

[AACa] AACS LA. AACS Specifications. At `http://www.aacsla.com/specifications/`. (Cited on page 11, 186, 190.)

[AACb] AACS LA. Introduction and Common Cryptographic Elements. At `http://www.aacsla.com/specifications/specs091/AACS`. (Cited on page 195.)

[AAC09] AACS Consortium. Advanced Access Content System (AACS) - introduction and common cryptographic elements book. `http://www.aacsla.com/specifications/`, September 2009. Revision 0.951. (Cited on page 229.)

[ABB10a] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *Proc. of EUROCRYPT*, volume 6110 of *LNCS*, pages 553–572. Springer, 2010. Full version available from the authors upon request. (Cited on page 46.)

[ABB10b] S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *Proc. of CRYPTO*, volume 6223 of *LNCS*, pages 98–115. Springer, 2010. (Cited on page 46.)

[ABC+11] Michel Abdalla, James Birkett, Dario Catalano, Alexander W. Dent, John Malone-Lee, Gregory Neven, Jacob C. N. Schuldt, and Nigel P. Smart. Wildcarded identity-based encryption. *Journal of Cryptology*, 24(1):42–82, January 2011. (Cited on page 44.)

[ABN10] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 480–497. Springer, 2010. (Cited on page 41, 175.)

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158, San Francisco, CA, USA, April 8–12, 2001. Springer, Berlin, Germany. (Cited on page 152.)

[ACD+06] Michel Abdalla, Dario Catalano, Alex Dent, John Malone-Lee, Gregory Neven, and Nigel Smart. Identity-based encryption gone wild. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP 2006: 33rd International Colloquium on Automata, Languages and Programming, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 300–311, Venice, Italy, July 10–14, 2006. Springer, Berlin, Germany. (Cited on page 200, 203, 206.)

[ACMP10] Michel Abdalla, Céline Chevalier, Mark Manulis, and David Pointcheval. Flexible group key exchange with on-demand computation of subgroup keys. In Daniel J. Bernstein and Tanja Lange, editors, *AFRICACRYPT 10: 3rd International Conference on Cryptology in Africa*, volume 6055 of *Lecture Notes in Computer Science*, pages 351–368, Stellenbosch, South Africa, May 3–6, 2010. Springer, Berlin, Germany. (Cited on page 229, 231, 234, 235.)

[ADML+07a] M. Abdalla, A. W. Dent, J. Malone-Lee, G. Neven, D. H. Phan, and N. P. Smart. Identity-based traitor tracing. In *Proceedings of PKC*, volume 4450 of *LNCS*, pages 361–376. Springer, 2007. (Cited on page 131.)

[ADML+07b] Michel Abdalla, Alexander W. Dent, John Malone-Lee, Gregory Neven, Duong Hieu Phan, and Nigel P. Smart. Identity-based traitor tracing. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *PKC 2007: 10th International Conference on Theory and Practice of Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 361–376, Beijing, China, April 16–20, 2007. Springer, Berlin, Germany. (Cited on page 199.)

[AF07] Masayuki Abe and Serge Fehr. Perfect NIZK with adaptive soundness. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 118–136, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Berlin, Germany. (Cited on page 153.)

[AFP05] Michel Abdalla, Pierre-Alain Fouque, and David Pointcheval. Password-based authenticated key exchange in the three-party setting. In Serge Vaudenay, editor, *PKC 2005: 8th International Workshop on Theory and Practice in Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 65–84, Les Diablerets, Switzerland, January 23–26, 2005. Springer, Berlin, Germany. (Cited on page 235.)

[AGHS13] S. Agrawal, C. Gentry, S. Halevi, and A. Sahai. Sampling discrete gaussians efficiently and obliviously. In *Proc. of ASIACRYPT (1)*, volume 8269 of *LNCS*, pages 97–116. Springer, 2013. (Cited on page 121, 136.)

[Ajt96a] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proc. of STOC*, pages 99–108. ACM, 1996. (Cited on page 115.)

[Ajt96b] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th Annual ACM Symposium on Theory of Computing*, pages 99–108, Philadelphia, Pennsylvania, USA, May 22–24, 1996. ACM Press. (Cited on page 33.)

[Ajt99] M. Ajtai. Generating hard instances of the short basis problem. In *Proc. of ICALP*, volume 1644 of *LNCS*, pages 1–9. Springer, 1999. (Cited on page 116, 118, 121.)

[AK05] Tomoyuki Asano and Kazuya Kamio. A tree based one-key broadcast encryption scheme with low computational overhead. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 05: 10th Australasian Conference on Information Security and Privacy*, volume 3574 of *Lecture Notes in Computer Science*, pages 89–100, Brisbane, Queensland, Australia, July 4–6, 2005. Springer, Berlin, Germany. (Cited on page 44.)

[ALO98] William Aiello, Sachin Lodha, and Rafail Ostrovsky. Fast digital identity revocation (extended abstract). In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 137–152, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Berlin, Germany. (Cited on page 18, 20.)

[AP11] J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. *Theor. Comput. Science*, 48(3):535–553, 2011. (Cited on page 34, 118.)

[AR13] D. Aggarwal and O. Regev. A note on discrete gaussian combinations of lattice vectors, 2013. Draft. Available at `http://arxiv.org/pdf/1308.2405v1.pdf`. (Cited on page 121, 136.)

[Asa02] Tomoyuki Asano. A revocation scheme with minimal storage at receivers. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 433–450, Queenstown, New Zealand, December 1–5, 2002. Springer, Berlin, Germany. (Cited on page 44, 64, 83.)

[AVL62] Georgii M. Adelson-Velskii and Evgenii M. Landis. An algorithm for the organization of information. *Proc. USSR Academy of Sciences*, 146:263–266, 1962. (Cited on page 241.)

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany. (Cited on page 147, 152.)

[BBKS07] Mihir Bellare, Alexandra Boldyreva, Kaoru Kurosawa, and Jessica Staddon. Multirecipient encryption schemes: How to save on bandwidth and computation without sacrificing security. *IEEE Trans. Inf. Th.*, 53(11):3927–3943, November 2007. (Cited on page 174.)

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Berlin, Germany. (Cited on page 201.)

[BD05] Mike Burmester and Yvo Desmedt. A secure and scalable group key exchange system. *Inf. Proc. Letters*, 94(3):137–143, May 2005. (Cited on page 230, 242.)

[BDJR97] Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *38th Annual Symposium on Foundations of Computer Science*, pages 394–403, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press. (Cited on page 8, 235.)

[Ber70] Elwyn R. Berlekamp. Factoring polynomials over large finite fields. *Mathematics of Computation*, 24(111):pp. 713–735, 1970. (Cited on page 160.)

[Ber91] Shimshon Berkovits. How to broadcast a secret (rump session). In Donald W. Davies, editor, *Advances in Cryptology – EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 535–541, Brighton, UK, April 8–11, 1991. Springer, Berlin, Germany. (Cited on page 61.)

[BF99a] D. Boneh and M. K. Franklin. An efficient public key traitor tracing scheme. In *Proc. of CRYPTO*, volume 1666 of *LNCS*, pages 338–353. Springer, 1999. (Cited on page 116, 117, 131, 132.)

[BF99b] Dan Boneh and Matthew K. Franklin. An efficient public key traitor tracing scheme. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 338–353, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. (Cited on page vii, 7, 9, 18, 26, 38, 39, 63, 88, 89, 103, 166, 186, 203.)

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany. (Cited on page 199, 202.)

[BF11] D. Boneh and D. M. Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In *Proc. of PKC*, volume 6571 of *LNCS*, pages 1–16. Springer, 2011. Full version available at `http://eprint.iacr.org/2010/453.pdf`. (Cited on page 34, 116, 117, 119, 121, 123.)

[BGdMM05] Lucas Ballard, Matthew Green, Breno de Medeiros, and Fabian Monrose. Correlation-resistant storage via keyword-searchable encryption. Cryptology ePrint Archive, Report 2005/417, 2005. `http://eprint.iacr.org/`. (Cited on page 201.)

[BGhCS04] P. Barreto, S. Galbraith, hEigeartaigh C., and M. Scott. Efficient Pairing Computation on Supersingular Abelian Varieties. Available at `http://eprint.iacr.org/2004/375`, 2004. (Cited on page 102.)

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Berlin, Germany. (Cited on page 8, 25, 31, 32, 144, 145, 147, 148, 150, 153, 154, 214, 215, 218, 219.)

[BK05] Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103, San Francisco, CA, USA, February 14–18, 2005. Springer, Berlin, Germany. (Cited on page 148, 237.)

[BKM10] Dan Boneh, Aggelos Kiayias, and Hart William Montgomery. Robust fingerprinting codes: a near pptimal construction. In *Proceedings of ACM DRM '10*, pages 3–12, New York, NY, USA, 2010. ACM. (Cited on page 168, 170.)

[BLP+13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *STOC*, pages 575–584. ACM, 2013. (Cited on page 36, 116, 117, 120, 136.)

[BLS04] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Efficient implementation of pairing-based cryptosystems. *Journal of Cryptology*, 17(4):321–334, September 2004. (Cited on page 102.)

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984. (Cited on page 5.)

[BM08] J. A. Bondy and U. S. R. Murty. *Graph theory*, volume 244 of *Graduate Texts in Mathematics*. Springer, New York, 2008. (Cited on page 81.)

[BMW05] Xavier Boyen, Qixiang Mei, and Brent Waters. Direct chosen ciphertext security from identity-based techniques. In Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels, editors, *ACM CCS 05: 12th Conference on Computer and Communications Security*, pages 320–329, Alexandria, Virginia, USA, November 7–11, 2005. ACM Press. (Cited on page 149.)

[BN08a] D. Boneh and M. Naor. Traitor tracing with constant size ciphertext. In *Proc. of ACM CCS*, pages 501–510. ACM, 2008. (Cited on page 131.)

[BN08b] Dan Boneh and Moni Naor. Traitor tracing with constant size ciphertext. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 08: 15th Conference on Computer and Communications Security*, pages 501–510, Alexandria, Virginia, USA, October 27–31, 2008. ACM Press. (Cited on page 9, 16, 40, 41, 62, 63, 70, 116, 166, 167, 186, 196.)

[Boy99] Victor Boyko. On the security properties of OAEP as an all-or-nothing transform. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 503–518, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. (Cited on page 40, 97, 99.)

[BP04] Mihir Bellare and Adriana Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 273–289, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Berlin, Germany. (Cited on page 152, 153, 154.)

[BP08] Olivier Billet and Duong Hieu Phan. Efficient traitor tracing from collusion secure codes. In Reihaneh Safavi-Naini, editor, *ICITS 08: 3rd International Conference on Information Theoretic Security*, volume 5155 of *Lecture Notes in Computer Science*, pages 171–182, Calgary, Canada, August 10–13, 2008. Springer, Berlin, Germany. (Cited on page 9, 16, 40, 41, 62, 63, 116, 131, 166, 167, 186.)

[BP09] Olivier Billet and Duong Hieu Phan. Traitors collaborating in public: Pirates 2.0. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 189–205, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany. (Cited on page 9, 43, 174, 185.)

[BPS00] O. Berkman, M. Parnas, and J. Sgall. Efficient Dynamic Traitor Tracing. In *Proceedings of the 11th Symposium on Discrete Algorithms*, pages 586–595, 2000. (Cited on page 9, 88.)

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press. (Cited on page 33, 161, 200, 215.)

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111, Perugia, Italy, May 9–12, 1994. Springer, Berlin, Germany. (Cited on page 99.)

[BS95] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data (extended abstract). In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 452–465, Santa Barbara, CA, USA, August 27–31, 1995. Springer, Berlin, Germany. (Cited on page 9, 14, 62, 186, 196, 200, 204.)

[BS98] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data. *IEEE Trans. Inf. Th.*, 44(5):1897–1905, 1998. A preliminary version appeared in Crypto '95. (Cited on page 9, 39, 88, 90, 102, 167, 168, 169.)

[BSW06a] D. Boneh, A. Sahai, and B. Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Proc. of EUROCRYPT*, volume 4004 of *LNCS*, pages 573–592. Springer, 2006. (Cited on page 131.)

[BSW06b] Dan Boneh, Amit Sahai, and Brent Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 573–592, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany. (Cited on page 9, 25, 29, 88, 113, 116, 186, 202, 203.)

[BW06a] D. Boneh and B. Waters. A fully collusion resistant broadcast, trace, and revoke system. In *Proc. of ACM CCS*, pages 211–220. ACM, 2006. (Cited on page 117, 131.)

[BW06b] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 211–220, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press. (Cited on page 9, 30, 37, 62, 88, 113, 230.)

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 480–499, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany. (Cited on page 9, 25, 37, 45, 116, 117.)

[CDH+00] Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 453–469, Bruges, Belgium, May 14–18, 2000. Springer, Berlin, Germany. (Cited on page 40, 97, 99.)

[CFGP06] Olivier Chevassut, Pierre-Alain Fouque, Pierrick Gaudry, and David Pointcheval. The twist-augmented technique for key exchange. In M. Yung, editor, *PKC 2006*, volume 3958 of *LNCS*, pages 410–426. Springer, 2006. Full version at `http://eprint.iacr.org/2005/061`. (Cited on page 241.)

[CFN94a] B. Chor, A. Fiat, and M. Naor. Tracing traitors. In *Proc. of CRYPTO*, volume 839 of *LNCS*, pages 257–270. Springer, 1994. (Cited on page 116, 131.)

[CFN94b] Benny Chor, Amos Fiat, and Moni Naor. Tracing traitors. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 257–270, Santa Barbara, CA, USA, August 21–25, 1994. Springer, Berlin, Germany. (Cited on page 8, 14, 22, 23, 62, 67, 87, 165, 185, 186.)

[CFNP00] Benny Chor, Amos Fiat, Moni Naor, and Benny Pinkas. Tracing Traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, 2000. (Cited on page 8, 87, 88, 131.)

[CHK04] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222, Interlaken, Switzerland, May 2–6, 2004. Springer, Berlin, Germany. (Cited on page 148, 206.)

[CHKP10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *Proc. of EUROCRYPT*, volume 6110 of *LNCS*, pages 523–552. Springer, 2010. (Cited on page 46.)

[Chv79] V. Chvátal. A greedy heuristic for the set-covering problem. *Math. Oper. Res.*, 4(3):233–235, 1979. (Cited on page 77.)

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *Cryptography and Coding, 8th IMA International Conference*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363, Cirencester, UK, December 17–19, 2001. Springer, Berlin, Germany. (Cited on page 199.)

[CPP05a] Hervé Chabanne, Duong Hieu Phan, and David Pointcheval. Public traceability in traitor tracing schemes. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 542–558, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany. (Cited on page 9, 37, 39, 87, 88, 89, 90, 92, 97, 102, 103, 104, 105, 106, 107, 108, 117, 131, 186, 196.)

[CPP05b] Hervé Chabanne, Duong Hieu Phan, and David Pointcheval. Public Traceability in Traitor Tracing Schemes. In Ronald Cramer, editor, *Advances in Cryptology— EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 542–558. Springer, 2005. (Cited on page 89.)

[CS98] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In Hugo Krawczyk, editor, *Crypto '98*, volume 1462 of *LNCS*, pages 13–25. Springer, 1998. (Cited on page 241.)

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Berlin, Germany. (Cited on page 38, 117, 128.)

[CS03] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003. (Cited on page 6, 146.)

[Dam87] Ivan Damgård. Collision free hash functions and public key signature schemes. In David Chaum and Wyn L. Price, editors, *Advances in Cryptology – EUROCRYPT'87*, volume 304 of *Lecture Notes in Computer Science*, pages 203–216, Amsterdam, The Netherlands, April 13–15, 1987. Springer, Berlin, Germany. (Cited on page 147.)

[Dam91] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456, Santa Barbara, CA, USA, August 11–15, 1991. Springer, Berlin, Germany. (Cited on page 152.)

[Dd11] Paolo D'Arco and Angel L. Pérez del Pozo. Fighting pirates 2.0. In Javier Lopez and Gene Tsudik, editors, *ACNS 11: 9th International Conference on Applied Cryptography and Network Security*, volume 6715 of *Lecture Notes in Computer Science*, pages 359–376, Nerja, Spain, June 7–10, 2011. Springer, Berlin, Germany. (Cited on page 44.)

[DDG13] Renaud Dubois, Margaux Dugardin, and Aurore Guillevic. Golden sequence for the PPSS broadcast encryption scheme with an asymmetric pairing. Cryptology ePrint Archive, Report 2013/477, 2013. `http://eprint.iacr.org/2013/477`. (Cited on page 31.)

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000. (Cited on page 148.)

[Def] DefectiveByDesign. `http://www.defectivebydesign.org/`. (Cited on page 187.)

[Del07] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215, Kuching, Malaysia, December 2–6, 2007. Springer, Berlin, Germany. (Cited on page 8, 145, 214.)

[Del08] CÃĭcile DelerablÃľe. Identity-based broadcast encryption with constant size ciphertexts and private keys. In K. Kurosawa, editor, *Asiacrypt 2007*, volume 4833 of *LNCS*, pages 200–215. Springer, 2008. (Cited on page 230.)

[Den06] Alexander W. Dent. The hardness of the dhk problem in the generic group model. Cryptology ePrint Archive, Report 2006/156, 2006. `http://eprint.iacr.org/2006/156`. (Cited on page 152, 153.)

[DF02] Yevgeniy Dodis and Nelly Fazio. Public-key broadcast encryption for stateless receivers. In *ACM Digital Rights Management—DRM '02*, pages 61–80, Heidelberg, 2002. Springer. LNCS 2696. (Cited on page 9, 31, 32, 62, 88, 153, 154, 230.)

[DF03] Yevgeniy Dodis and Nelly Fazio. Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In Yvo Desmedt, editor, *PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography*,

volume 2567 of *Lecture Notes in Computer Science*, pages 100–115, Miami, USA, January 6–8, 2003. Springer, Berlin, Germany. (Cited on page 8, 9, 62, 88, 144, 214, 230.)

[DFKY05] Y. Dodis, N. Fazio, A. Kiayias, and M. Yung. Scalable Public-Key Tracing and Revoking. *Journal of Distributed Computing*, 17(4):323–347, 2005. (Cited on page 9, 88.)

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE Trans. Inf. Th.*, 22(6):644–654, November 1976. (Cited on page 241.)

[DH00] Ding-Zhu Du and Frank K. Hwang. *Combinatorial group testing and its applications*, volume 12 of *Series on Applied Mathematics*. World Scientific Publishing Co. Inc., River Edge, NJ, second edition, 2000. (Cited on page 63, 69.)

[DK05] Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In J. Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 188–209. Springer, 2005. (Cited on page 237.)

[Dor43] R. Dorfman. The detection of defective members of large populations. *The Annals of Mathematical Statistics*, 14(4):436–440, 1943. (Cited on page 18, 69.)

[DP08] Yvo Desmedt and Duong Hieu Phan. A CCA secure hybrid Damgård's ElGamal encryption. In Joonsang Baek, Feng Bao, Kefei Chen, and Xuejia Lai, editors, *ProvSec 2008: 2nd International Conference on Provable Security*, volume 5324 of *Lecture Notes in Computer Science*, pages 68–82, Shanghai, China, October 31 – November 1, 2008. Springer, Berlin, Germany. (Cited on page 153.)

[DPP07] Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors, *PAIRING 2007: 1st International Conference on Pairing-based Cryptography*, volume 4575 of *Lecture Notes in Computer Science*, pages 39–59, Tokyo, Japan, July 2–4, 2007. Springer, Berlin, Germany. (Cited on page 8, 25, 28, 145, 146, 230, 232, 236.)

[EFF85] P. Erdos, P. Frankl, and Z. Füredi. Families of finite sets in which no set is covered by the union of r others. *Israel J. Math.*, 51(1-2):79–89, 1985. (Cited on page 18, 69.)

[Ele] Electronic Frontier Foundation. `http://www.eff.org/`. (Cited on page 187.)

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985. (Cited on page 241.)

[Fei98] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45(4):634–652, 1998. (Cited on page 81.)

[FN93] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491, Santa Barbara, CA, USA, August 22–26, 1993. Springer, Berlin, Germany. (Cited on page 8, 61, 144, 214, 229, 231.)

[FNP07a] N. Fazio, A. Nicolosi, and D. H. Phan. Traitor tracing with optimal transmission rate. In *Proc. of ISC*, volume 4779 of *LNCS*, pages 71–88. Springer, 2007. (Cited on page 131.)

[FNP07b] Nelly Fazio, Antonio Nicolosi, and Duong Hieu Phan. Traitor tracing with optimal transmission rate. In Juan A. Garay, Arjen K. Lenstra, Masahiro Mambo, and René Peralta, editors, *ISC 2007: 10th International Conference on Information Security*, volume 4779 of *Lecture Notes in Computer Science*, pages 71–88, Valparaíso, Chile, October 9–12, 2007. Springer, Berlin, Germany. (Cited on page viii, 39, 40, 62, 87, 166.)

[FT99] Amos Fiat and Tamir Tassa. Dynamic traitor training. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 354–371, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. (Cited on page 40, 166, 186.)

[FT01] Amos Fiat and Tamir Tassa. Dynamic traitor tracing. *Journal of Cryptology*, 14(3):211–223, 2001. (Cited on page 9, 88.)

[GGH96] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Collision-free hashing from lattice problems. Cryptology ePrint Archive, Report 1996/009, 1996. `http://eprint.iacr.org/1996/009`. (Cited on page 147.)

[GGH13a] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, volume 7881 of *LNCS*, pages 1–17, 2013. (Cited on page 117.)

[GGH+13b] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Proc. of FOCS*, pages 40–49. IEEE Computer Society Press, 2013. (Cited on page 116.)

[GGH+13c] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press. (Cited on page 9, 25, 45.)

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *Symposium on Foundations of Computer Science—FOCS 84*, pages 464–479. IEEE, 1984. (Cited on page 13, 194.)

[GJ79] Michael R. Garey and David S. Johnson. *Computers and intractability.* W. H. Freeman and Co., San Francisco, Calif., 1979. A guide to the theory of NP-completeness, A Series of Books in the Mathematical Sciences. (Cited on page 81.)

[GKV10] S. D. Gordon, J. Katz, and V. Vaikuntanathan. A group signature scheme from lattice assumptions. In *Proc. of ASIACRYPT*, volume 2647 of *LNCS*, pages 395–412. Springer, 2010. (Cited on page 129, 141.)

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. (Cited on page 5.)

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proc. of STOC*, pages 197–206. ACM, 2008. Full version available at `http://eprint.iacr.org/2007/432.pdf`. (Cited on page 33, 35, 115, 116, 118, 126, 127, 136.)

[GRW06] Craig Gentry, Zulfikar Ramzan, and David P. Woodruff. Explicit exclusive set systems with applications to broadcast encryption. In *47th Annual Symposium on Foundations of Computer Science*, pages 27–38, Berkeley, CA, USA, October 21–24, 2006. IEEE Computer Society Press. (Cited on page 20.)

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432, Istanbul, Turkey, April 13–17, 2008. Springer, Berlin, Germany. (Cited on page 148.)

[GST04] Michael T. Goodrich, Jonathan Z. Sun, and Roberto Tamassia. Efficient tree-based revocation in groups of low-state devices. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 511–527, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Berlin, Germany. (Cited on page 20.)

[GSY99] Eli Gafni, Jessica Staddon, and Yiqun Lisa Yin. Efficient methods for integrating traceability and broadcast encryption. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 372–387, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. (Cited on page 9, 18, 20, 88.)

[GW09] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 171–188, Cologne, Germany, April 26–30, 2009. Springer, Berlin, Germany. (Cited on page 8, 31, 32, 145, 153, 154, 214, 230.)

[HS02] Dani Halevy and Adi Shamir. The LSD broadcast encryption scheme. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 47–60, Santa Barbara, CA, USA, August 18–22, 2002. Springer, Berlin, Germany. (Cited on page 20, 62.)

[HT98] Satoshi Hada and Toshiaki Tanaka. On the existence of 3-round zero-knowledge protocols. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 408–423, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Berlin, Germany. (Cited on page 152.)

[HvLLT98] Henk D. L. Hollmann, Jack H. van Lint, Jean-Paul M. G. Linnartz, and Ludo M. G. M. Tolhuizen. On Codes with the Identifiable Parent Property. *J. Comb. Theory, Ser. A*, 82(2):121–133, 1998. (Cited on page 196.)

[JHC+05] Nam-Su Jho, Jung Yeon Hwang, Jung Hee Cheon, Myung-Hwan Kim, Dong Hoon Lee, and Eun Sun Yoo. One-way chain based broadcast encryption schemes. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 559–574, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany. (Cited on page 20.)

[JL07] Hongxia Jin and Jeffery Lotspiech. Renewable traitor tracing: A trace-revoke-trace system for anonymous attack. In *ESORICS*, volume 4734 of *LNCS*, pages 563–577. Springer, 2007. (Cited on page 40, 166.)

[JL09] Hongxia Jin and Jeffrey Lotspiech. Defending against the pirate evolution attack. In *ISPEC '09: Proceedings of the 5th International Conference on Information Security Practice and Experience*, pages 147–158, Berlin, Heidelberg, 2009. Springer-Verlag. (Cited on page 44.)

[KD98a] K. Kurosawa and Y. Desmedt. Optimum traitor tracing and asymmetric schemes. In *Proc. of EUROCRYPT*, LNCS, pages 145–157. Springer, 1998. (Cited on page 131.)

[KD98b] Kaoru Kurosawa and Yvo Desmedt. Optimum traitor tracing and asymmetric schemes. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 145–157, Espoo, Finland, May 31 – June 4, 1998. Springer, Berlin, Germany. (Cited on page 9, 88.)

[KHL03] Chong Hee Kim, Yong Ho Hwang, and Pil Joong Lee. An efficient public key trace and revoke scheme secure against adaptive chosen ciphertext attack. In Chi-Sung Laih, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 359–373, Taipei, Taiwan, November 30 – December 4, 2003. Springer, Berlin, Germany. (Cited on page 9, 88.)

[Kil06] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600, New York, NY, USA, March 4–7, 2006. Springer, Berlin, Germany. (Cited on page 148.)

[Kle00] P. N. Klein. Finding the closest lattice vector when it's unusually close. In *Proc. of SODA*, pages 937–941. ACM, 2000. (Cited on page 136.)

[KMPB05] Tim Kerins, William P. Marnane, Emanuel M. Popovici, and Paulo S. L. M. Barreto. Efficient hardware for the Tate pairing calculation in characteristic three. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 412–426, Edinburgh, UK, August 29 – September 1, 2005. Springer, Berlin, Germany. (Cited on page 102.)

[KP07] Aggelos Kiayias and Serdar Pehlivanoglu. Pirate evolution: How to make the most of your traitor keys. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 448–465, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Berlin, Germany. (Cited on page 43, 186.)

[KP09] Aggelos Kiayias and Serdar Pehlivanoglu. Tracing and revoking pirate rebroadcasts. In *ACNS 2009*, volume 5536 of *LNCS*, pages 253–271. Springer, 2009. (Cited on page 40, 166.)

[KP10] A. Kiayias and S. Pehlivanglu. *Encryption For Digital Content*. Springer, 2010. (Cited on page 7, 116.)

[KPT04] Yongdae Kim, Adrian Perrig, and Gene Tsudik. Tree-based group key agreement. *ACM Trans. Inf. Syst. Sec.*, 7(1):60–96, May 2004. (Cited on page 230, 241.)

[KR03] Ravi Kumar and Alexander Russell. A note on the set systems used for broadcast encryption. In *14th Annual ACM-SIAM Symposium on Discrete Algorithms*,

pages 470–471, Baltimore, Maryland, USA, January 12–14, 2003. ACM-SIAM. (Cited on page 18, 20.)

[KRS99] Ravi Kumar, Sridhar Rajagopalan, and Amit Sahai. Coding constructions for blacklisting problems without computational assumptions. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 609–623, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. (Cited on page 20.)

[KSNW03] Hartono Kurnio, Rei Safavi-Naini, and Huaxiong Wang. A group key distribution scheme with decentralised user join. In *SCN 2003*, volume 2576 of *LNCS*, pages 146–163. Springer, 2003. (Cited on page 231.)

[KWHI01] Hirotaka Komaki, Yuji Watanabe, Goichiro Hanaoka, and Hideki Imai. Efficient asymmetric self-enforcement scheme with public traceability. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 225–239, Cheju Island, South Korea, February 13–15, 2001. Springer, Berlin, Germany. (Cited on page 9, 38, 117, 131.)

[KY01a] A. Kiayias and M. Yung. On crafty pirates and foxy tracers. In *Proc. of DRM Workshop*, volume 2320 of *LNCS*, pages 22–39. Springer, 2001. (Cited on page 132.)

[KY01b] A. Kiayias and M. Yung. Self protecting pirates and black-box traitor tracing. In *Proc. of CRYPTO*, volume 2139 of *LNCS*, pages 63–79. Springer, 2001. (Cited on page 132.)

[KY01c] Aggelos Kiayias and Moti Yung. Self protecting pirates and black-box traitor tracing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 63–79, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany. (Cited on page 9, 38, 88.)

[KY02a] A. Kiayias and M. Yung. Breaking and Repairing Asymmetric Public-Key Traitor Tracing. In *Digital Rights Management—DRM '02*, pages 32–50, Heidelberg, 2002. Springer. LNCS 2696. (Cited on page 9, 38, 88, 117, 131.)

[KY02b] Aggelos Kiayias and Moti Yung. On Crafty Pirates and Foxy Tracers. In Tomas Sander, editor, *Security and Privacy in Digital Rights Management—DRM 2001*, volume 2320 of *Lecture Notes in Computer Science*, pages 22–39. Springer, 2002. (Cited on page 7, 186, 202, 203.)

[KY02c] Aggelos Kiayias and Moti Yung. Traitor tracing with constant transmission rate. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 450–465, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Berlin, Germany. (Cited on page 9, 38, 40, 41, 62, 63, 67, 87, 88, 89, 90, 92, 96, 97, 102, 103, 104, 106, 107, 108, 116, 131, 132, 166, 179, 186, 196, 268.)

[KY02d] Kaoru Kurosawa and Takuya Yoshida. Linear code implies public-key traitor tracing. In David Naccache and Pascal Paillier, editors, *PKC 2002: 5th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2274 of *Lecture Notes in Computer Science*, pages 172–187, Paris, France, February 12–14, 2002. Springer, Berlin, Germany. (Cited on page 9, 88, 131.)

[KY03] Aggelos Kiayias and Moti Yung. Extracting group signatures from traitor tracing schemes. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 630–648, Warsaw, Poland, May 4–8, 2003. Springer, Berlin, Germany. (Cited on page 47.)

[KY06] A. Kiayias and M. Yung. Copyrighting Public-key Functions and Applications to Black-box Traitor Tracing. Full revised version of [KY02c]. Available at: `http://eprint.iacr.org/2006/458/`, 2006. (Cited on page 89, 97.)

[KY07] Jonathan Katz and Moti Yung. Scalable protocols for authenticated group key exchange. *Journal of Cryptology*, 20(1):85–113, January 2007. (Cited on page 235, 242.)

[Lin03] Yehuda Lindell. A simpler construction of cca2-secure public-key encryption under general assumptions. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 241–254, Warsaw, Poland, May 4–8, 2003. Springer, Berlin, Germany. (Cited on page 148.)

[Lov75] L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Math.*, 13(4):383–390, 1975. (Cited on page 77.)

[LPQ12] Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In *PKC 2012*, volume 7293 of *LNCS*, pages 206–224. Springer, 2012. (Cited on page 168, 182.)

[LPR13] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43, 2013. (Cited on page 119.)

[LPSS14] San Ling, Duong Hieu Phan, Damien Stehlé, and Ron Steinfeld. Hardness of k-LWE and applications in traitor tracing. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 315–334, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Berlin, Germany. (Cited on page vii, 9, 33, 35, 37, 115.)

[LSS14a] A. Langlois, D. Stehlé, and R. Steinfeld. GGHLite: More efficient multilinear maps from ideal lattices. In *Proc. of EUROCRYPT*, LNCS, pages 239–256. Springer, 2014. (Cited on page 119.)

[LSS14b] A. Langlois, D. Stehlé, and R. Steinfeld. Improved and simplified security proofs in lattice-based cryptography: using the Rényi divergence rather than the statistical distance, 2014. Available on the webpages of the authors. (Cited on page 119.)

[LSW10] Allison B. Lewko, Amit Sahai, and Brent Waters. Revocation systems with very small private keys. In *2010 IEEE Symposium on Security and Privacy*, pages 273–285, Berkeley/Oakland, California, USA, May 16–19, 2010. IEEE Computer Society Press. (Cited on page 8, 31, 32, 145, 153, 154, 214.)

[Lyn] B. Lynn. PBC Library. Available at `http://crypto.stanford.edu/pbc/`. (Cited on page 102.)

[Man09] Mark Manulis. Group key exchange enabling on-demand derivation of peer-to-peer keys. In *ACNS 2009*, volume 5536 of *LNCS*, pages 1–19. Springer, 2009. Full version at `http://www.manulis.eu/pub.html`. (Cited on page 229, 230.)

[MP12] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Proc. of EUROCRYPT*, volume 7237 of *LNCS*, pages 700–718. Springer, 2012. (Cited on page 116, 118, 121, 122.)

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995. (Cited on page 208.)

[MR07] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput*, 37(1):267–302, 2007. (Cited on page 118, 120, 136.)

[MR09] D. Micciancio and O. Regev. Lattice-based cryptography. In *Post-Quantum Cryptography, D. J. Bernstein, J. Buchmann, E. Dahmen (Eds)*, pages 147–191. Springer, 2009. (Cited on page 33, 115.)

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126, Santa Barbara, CA, USA, August 18–22, 2002. Springer, Berlin, Germany. (Cited on page 161.)

[NNL01] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany. (Cited on page 8, 9, 11, 20, 46, 62, 63, 66, 70, 71, 88, 144, 186, 187, 190, 191, 214, 229, 236, 237, 238, 239, 240.)

[NP98] Moni Naor and Benny Pinkas. Threshold traitor tracing. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 502–517, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Berlin, Germany. (Cited on page 88.)

[NP00] Moni Naor and Benny Pinkas. Efficient trace and revoke schemes. In Yair Frankel, editor, *FC 2000: 4th International Conference on Financial Cryptography*, volume 1962 of *Lecture Notes in Computer Science*, pages 1–20, Anguilla, British West Indies, February 20–24, 2000. Springer, Berlin, Germany. (Cited on page vii, 9, 27, 62, 88, 116.)

[NPP13] Hung Q. Ngo, Duong Hieu Phan, and David Pointcheval. Black-box trace&revoke codes. *Algorithmica*, 67(3):418–448, 2013. (Cited on page vii, 9, 16, 17, 61.)

[NPR12] Hung Q. Ngo, Ely Porat, and Atri Rudra. Efficiently decodable compressed sensing by list-recoverable codes and recursion. In *STACS*, pages 230–241, 2012. (Cited on page 18, 74.)

[NSS99] David Naccache, Adi Shamir, and Julien P. Stern. How to copyright a function? In *PKC '99*, volume 1560 of *LNCS*, pages 188–196. Springer, 1999. (Cited on page 168.)

[Nui09] Koji Nuida. A general conversion method of fingerprint codes to (more) robust fingerprint codes against bit erasure. In Kaoru Kurosawa, editor, *ICITS 09: 4th*

International Conference on Information Theoretic Security, volume 5973 of *Lecture Notes in Computer Science*, pages 194–212, Shizuoka, Japan, December 3–6, 2009. Springer, Berlin, Germany. (Cited on page 9, 16.)

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *21st Annual ACM Symposium on Theory of Computing*, pages 33–43, Seattle, Washington, USA, May 15–17, 1989. ACM Press. (Cited on page 147.)

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd Annual ACM Symposium on Theory of Computing*, pages 427–437, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press. (Cited on page 5, 148.)

[OPW11] A. O'Neill, C. Peikert, and B. Waters. Bi-deniable public-key encryption. In *Proc. of CRYPTO*, volume 6841 of *LNCS*, pages 525–542. Springer, 2011. (Cited on page 117.)

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proc. of STOC*, pages 333–342. ACM, 2009. (Cited on page 36, 46, 116, 120.)

[Pei10] C. Peikert. An efficient and parallel Gaussian sampler for lattices. In *Proc. of CRYPTO*, volume 6223 of *LNCS*, pages 80–97. Springer, 2010. (Cited on page 118, 120, 124, 136.)

[Pfi96] B. Pfitzmann. Trials of traced traitors. In *Information Hiding*, volume 1174 of *LNCS*, pages 49–64. Springer, 1996. (Cited on page 9, 38, 88, 117, 131, 186.)

[Pha06] Duong Hieu Phan. Traitor tracing for stateful pirate decoders with constant ciphertext rate. In Phong Q. Nguyen, editor, *Progress in Cryptology—VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 354–365. Springer, 2006. (Cited on page 196.)

[PPS11] Duong Hieu Phan, David Pointcheval, and Mario Strefler. Security notions for broadcast encryption. In Javier Lopez and Gene Tsudik, editors, *ACNS 11: 9th International Conference on Applied Cryptography and Network Security*, volume 6715 of *Lecture Notes in Computer Science*, pages 377–394, Nerja, Spain, June 7–10, 2011. Springer, Berlin, Germany. (Cited on page 147, 182, 215, 231, 233.)

[PPS12a] Duong Hieu Phan, David Pointcheval, and Mario Strefler. Decentralized dynamic broadcast encryption. In Ivan Visconti and Roberto De Prisco, editors, *SCN 12: 8th International Conference on Security in Communication Networks*, volume 7485 of *Lecture Notes in Computer Science*, pages 166–183, Amalfi, Italy, September 5–7, 2012. Springer, Berlin, Germany. (Cited on page 31, 32, 45, 145, 153, 154, 229.)

[PPS12b] Duong Hieu Phan, David Pointcheval, and Mario Strefler. Message-based traitor tracing with optimal ciphertext rate. In Alejandro Hevia and Gregory Neven, editors, *Progress in Cryptology - LATINCRYPT 2012: 2nd International Conference on Cryptology and Information Security in Latin America*, volume 7533 of *Lecture Notes in Computer Science*, pages 56–77, Santiago, Chile, October 7–10, 2012. Springer, Berlin, Germany. (Cited on page viii, 39, 40, 165.)

[PPSS12] Duong Hieu Phan, David Pointcheval, Siamak Fayyaz Shahandashti, and Mario Strefler. Adaptive CCA broadcast encryption with constant-size secret keys and ciphertexts. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *ACISP 12: 17th Australasian Conference on Information Security and Privacy*, volume 7372 of *Lecture Notes in Computer Science*, pages 308–321, Wollongong, NSW, Australia, July 9–11, 2012. Springer, Berlin, Germany. (Cited on page 214.)

[PPSS13] Duong Hieu Phan, David Pointcheval, Siamak Fayyaz Shahandashti, and Mario Strefler. Adaptive cca broadcast encryption with constant-size secret keys and ciphertexts. *Int. J. Inf. Sec.*, 12(4):251–265, 2013. (Cited on page vii, 8, 30, 143.)

[PPT13] Duong Hieu Phan, David Pointcheval, and Viet Cuong Trinh. Multi-channel broadcast encryption. In Kefei Chen, Qi Xie, Weidong Qiu, Ninghui Li, and Wen-Guey Tzeng, editors, *ASIACCS 13: 8th Conference on Computer and Communications Security*, pages 277–286, Hangzhou, China, May 8–10, 2013. ACM Press. (Cited on page vii, 31, 213.)

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Crypto*, 13(3):361–396, 2000. (Cited on page 178.)

[PSNT06a] D. H. Phan, R. Safavi-Naini, and D. Tonien. Generic construction of hybrid public key traitor tracing with full-public-traceability. In *Proc. of ICALP (2)*, volume 4052 of *LNCS*, pages 264–275. Springer, 2006. (Cited on page 117, 131.)

[PSNT06b] Duong Phan, Reihaneh Safavi-Naini, and Dongvu Tonien. Generic construction of hybrid public key traitor tracing with full-public-traceability. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP 2006: 33rd International Colloquium on Automata, Languages and Programming, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 264–275, Venice, Italy, July 10–14, 2006. Springer, Berlin, Germany. (Cited on page 9, 37, 89, 102, 196.)

[PSNT06c] Duong Hieu Phan, Reihaneh Safavi-Naini, and Dongvu Tonien. Generic Construction of Hybrid Public Key Traitor Tracing with Full-Public-Traceability. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming—ICALP 2006*, volume 4052 of *Lecture Notes in Computer Science*, pages 264–275. Springer, 2006. (Cited on page 89.)

[PSS03] C. Peikert, A. Shelat, and A. Smith. Lower bounds for collusion-secure fingerprinting. In *Proc. of SODA*, pages 472–479, 2003. (Cited on page 131.)

[PT11] Duong Hieu Phan and Viet Cuong Trinh. Identity-based trace and revoke schemes. In Xavier Boyen and Xiaofeng Chen, editors, *ProvSec 2011: 5th International Conference on Provable Security*, volume 6980 of *Lecture Notes in Computer Science*, pages 204–221, Xi'an, China, October 16–18, 2011. Springer, Berlin, Germany. (Cited on page 44.)

[PT13] Duong Hieu Phan and Viet Cuong Trinh. Key-leakage resilient revoke scheme resisting pirates 2.0 in bounded leakage model. In Amr Youssef, Abderrahmane Nitaj, and Aboul Ella Hassanien, editors, *AFRICACRYPT 13: 6th International Conference on Cryptology in Africa*, volume 7918 of *Lecture Notes in Computer Science*, pages 342–358, Cairo, Egypt, June 22–24, 2013. Springer, Berlin, Germany. (Cited on page 44.)

[PW97] Birgit Pfitzmann and Michael Waidner. Asymmetric fingerprinting for larger collusions. In *ACM CCS 97: 4th Conference on Computer and Communications Security*, pages 151–160, Zurich, Switzerland, April 1–4, 1997. ACM Press. (Cited on page 9, 38, 117, 131.)

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *Proc. of STOC*, pages 187–196. ACM, 2008. (Cited on page 46.)

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proc. of STOC*, pages 84–93. ACM, 2005. (Cited on page 33, 115, 120.)

[Reg09] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009. (Cited on page 33, 36, 115, 116, 120.)

[Reg10] O. Regev. The learning with errors problem, 2010. Invited survey in CCC 2010, available at `http://www.cims.nyu.edu/~regev/`. (Cited on page 33, 115.)

[Riv97] Ronald L. Rivest. All-or-nothing encryption and the package transform. In Eli Biham, editor, *Fast Software Encryption – FSE'97*, volume 1267 of *Lecture Notes in Computer Science*, pages 210–218, Haifa, Israel, January 20–22, 1997. Springer, Berlin, Germany. (Cited on page 9, 40, 97.)

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press. (Cited on page 147.)

[RS91] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444, Santa Barbara, CA, USA, August 11–15, 1991. Springer, Berlin, Germany. (Cited on page 5.)

[RS04] Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In Bimal K. Roy and Willi Meier, editors, *Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 371–388, New Delhi, India, February 5–7, 2004. Springer, Berlin, Germany. (Cited on page 147.)

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *40th Annual Symposium on Foundations of Computer Science*, pages 543–553, New York, New York, USA, October 17–19, 1999. IEEE Computer Society Press. (Cited on page 148.)

[Sco02] Mike Scott. Authenticated id-based key exchange and remote log-in with simple token and pin number. Cryptology ePrint Archive, Report 2002/164, 2002. `http://eprint.iacr.org/`. (Cited on page 201.)

[Sha49] Claude E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949. (Cited on page 5.)

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, August 19–23, 1984. Springer, Berlin, Germany. (Cited on page 199, 200.)

[Sho90] Victor Shoup. On the deterministic complexity of factoring polynomials over finite fields. *Information Processing Letters*, 33(5):261 – 267, 1990. (Cited on page 160.)

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266, Konstanz, Germany, May 11–15, 1997. Springer, Berlin, Germany. (Cited on page 152.)

[Sho00] Victor Shoup. Using hash functions as a hedge against chosen ciphertext attack. In B. Preneel, editor, *Eurocrypt 2000*, volume 1807 of *LNCS*, pages 275–288. Springer, 2000. (Cited on page 232.)

[Sho04] V. Shoup. Sequences of Games: A Tool for Taming Complexity in Security Proofs. Manuscript. Available at `shoup.net/papers/games.pdf`, 2004. (Cited on page 109.)

[Sim98] Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345, Espoo, Finland, May 31 – June 4, 1998. Springer, Berlin, Germany. (Cited on page 147.)

[Sir07a] Thomas Sirvent. Traitor tracing scheme with constant ciphertext rate against powerful pirates. In *Proc. of Workshop on Coding and Cryptography (WCC'07)*, pages 379–388, 2007. Full version at `http://eprint.iacr.org/2006/383`. (Cited on page 9, 16, 40, 62, 63, 166, 196.)

[Sir07b] Thomas Sirvent. Traitor tracing scheme with constant ciphertext rate against powerful pirates. In Daniel Augot, Nicolas Sendrier, and Jean-Pierre Tillich, editors, *Workshop on Coding and Cryptography—WCC '07*, pages 379–388, April 2007. (Cited on page 116.)

[SNW03a] Reihaneh Safavi-Naini and Yejing Wang. Sequential traitor tracing. *IEEE Trans. Inf. Th.*, 49(5):1319–1326, May 2003. A preliminary version appeared at Crypto 2000. (Cited on page 40, 166.)

[SNW03b] Reihaneh Safavi-Naini and Yejing Wang. Traitor tracing for shortened and corrupted fingerprints. In *DRM 2003*, volume 2696 of *LNCS*, pages 81–100. Springer, 2003. (Cited on page 9, 16.)

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, Okinawa, Japan, January 2000. (Cited on page 199.)

[SS01] Palash Sarkar and Douglas R. Stinson. Frameproof and IPP Codes. In C. Pandu Rangan and Cunsheng Ding, editors, *Progress in Cryptology—INDOCRYPT 2001*, volume 2247 of *Lecture Notes in Computer Science*, pages 117–126. Springer, 2001. (Cited on page 196.)

[SSW00] J.N. Staddon, D.R. Stinson, and Ruizhong Wei. Combinatorial properties of frameproof and traceability codes. *IEEE Transactions on Information Theory*, 47:1042–1049, 2000. (Cited on page 19, 62.)

[SSW01a] A. Silverberg, J. Staddon, and J. L. Walker. Efficient traitor tracing algorithms using list decoding. In *Proc. of ASIACRYPT*, volume 2248 of *LNCS*, pages 175–192. Springer, 2001. (Cited on page 131.)

[SSW01b] Alice Silverberg, Jessica Staddon, and Judy L. Walker. Efficient traitor tracing algorithms using list decoding. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 175–192, Gold Coast, Australia, December 9–13, 2001. Springer, Berlin, Germany. (Cited on page 14.)

[Sto] Stop DRM Now! `http://stopdrmnow.org/`. (Cited on page 187.)

[SW98a] D. R. Stinson and R. Wei. Combinatorial Properties and Constructions of Traceability Schemes and Frameproof Codes. *SIAM Journal on Discrete Mathematics*, 11(1):41–53, 1998. (Cited on page 9, 87.)

[SW98b] D. R. Stinson and R. Wei. Combinatorial properties and constructions of traceability schemes and frameproof codes. *SIAM J. Discrete Math.*, 11(1):41–53, 1998. (Cited on page 131.)

[SW98c] D. R. Stinson and R. Wei. Key preassigned traceability schemes for broadcast encryption. In *Proc. of SAC*, volume 1556 of *LNCS*, pages 144–156. Springer, 1998. (Cited on page 131.)

[SW98d] Douglas R. Stinson and Ruizhong Wei. Key preassigned traceability schemes for broadcast encryption. In Stafford E. Tavares and Henk Meijer, editors, *SAC 1998: 5th Annual International Workshop on Selected Areas in Cryptography*, volume 1556 of *Lecture Notes in Computer Science*, pages 144–156, Kingston, Ontario, Canada, August 17–18, 1998. Springer, Berlin, Germany. (Cited on page 14.)

[SW03] R. Safavi-Naini and Y. Wang. Sequential Traitor Tracing. *IEEE Transactions on Information Theory*, 49(5):1319–1326, 2003. (Cited on page 9, 88.)

[Tar03] Gábor Tardos. Optimal probabilistic fingerprint codes. In *35th Annual ACM Symposium on Theory of Computing*, pages 116–125, San Diego, California, USA, June 9–11, 2003. ACM Press. (Cited on page 9, 14, 15, 39, 62, 88, 99, 102, 196, 204.)

[Tar08a] G. Tardos. Optimal probabilistic fingerprint codes. *J. ACM*, 55(2), 2008. (Cited on page 131.)

[Tar08b] Gábor Tardos. Optimal probabilistic fingerprint codes. *J. ACM*, 55(2), May 2008. A preliminary version appeared in STOC '03. (Cited on page 170.)

[TM05] Tran Van Trung and Sosina Martirosyan. New Constructions for IPP Codes. *Des. Codes Cryptography*, 35(2):227–239, 2005. (Cited on page 14.)

[TSN06] Dongvu Tonien and Reihaneh Safavi-Naini. An efficient single-key pirates tracing scheme using cover-free families. In Jianying Zhou, Moti Yung, and Feng Bao, editors, *ACNS 06: 4th International Conference on Applied Cryptography and*

Network Security, volume 3989 of *Lecture Notes in Computer Science*, pages 82–97, Singapore, June 6–9, 2006. Springer, Berlin, Germany. (Cited on page 18, 63.)

[TT01] Wen-Guey Tzeng and Zhi-Jia Tzeng. A public-key traitor tracing scheme with revocation using dynamic shares. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 207–224, Cheju Island, South Korea, February 13–15, 2001. Springer, Berlin, Germany. (Cited on page 9, 88.)

[Wat05] Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany. (Cited on page 200, 203, 206, 210.)

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636, Santa Barbara, CA, USA, August 16–20, 2009. Springer, Berlin, Germany. (Cited on page 8, 31, 32, 145, 153, 154, 214.)

[WHI01] Yuji Watanabe, Goichiro Hanaoka, and Hideki Imai. Efficient asymmetric public-key traitor tracing without trusted agents. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 392–407, San Francisco, CA, USA, April 8–12, 2001. Springer, Berlin, Germany. (Cited on page 9, 38, 117, 131.)

[WMS+09] Qianhong Wu, Yi Mu, Willy Susilo, Bo Qin, and Josep Domingo-Ferrer. Asymmetric group key agreement. In Antoine Joux, editor, *Eurocrypt 2009*, volume 5479 of *LNCS*, pages 153–170. Springer, 2009. (Cited on page 231.)

[WQZ+11] Qianhong Wu, Bo Qin, Lei Zhang, Josep Domingo-Ferrer, and Oriol Farras. Bridging broadcast encryption and group key agreement. In D.H. Lee and X. Wang, editors, *Asiacrypt 2011*, volume 7073 of *LNCS*, pages 143–160. Springer, 2011. (Cited on page 230.)

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press. (Cited on page 5.)

[YFDL04] Danfeng Yao, Nelly Fazio, Yevgeniy Dodis, and Anna Lysyanskaya. ID-based encryption for complex hierarchies with applications to forward security and broadcast encryption. In *ACM CCS '04*. ACM Press, 2004. Full version at `http://www.cs.brown.edu/~anna/research.html`. (Cited on page 230.)

[ZZ11] Xingwen Zhao and Fangguo Zhang. Traitor tracing against public collaboration. In *ISPEC '11: The 7th International Conference on Information Security Practice and Experience*, Guangzhou / China, 2011. (Cited on page 44.)

Abstract

In this thesis, we consider a generalization of the encryption from "one-to-one" to "one-to-many" communication. The objective is to allow a center to send secret messages to a large number of receivers. The security notions in "one-to-many" communications need to be extended beyond the notion of confidentiality in "one-to-one" encryption to meet practical requirements. Two main functionalites are studied: traitor tracing which identifies malicious users who leak their secrets to a pirate and broadcast encryption which prevents non-legitimate or revoked users from decrypting broadcasted information.

In the first part of this thesis, we focus on combinatorial schemes. Our objective is to design solutions that support both the functionalities of broadcast encryption and traitor tracing against various pirate strategies. In one direction, we introduce a trace&revoke code and a tracing technique called "shadow group testing" to deal with "smart" pirates. In another direction, we propose a method to integrate revocation into some code-based schemes.

The second part discusses the techniques for constructing algebraic schemes. We first extend some well-known schemes, in particular the pairing-based BGW one, in order to enhance the security and to capture new properties. We then propose the first lattice-based traitor tracing of which the security is based on the hardness of the Learning With Errors problem. We finally consider the combination of algebraic and combinatorial methods and propose an optimal ciphertext rate traitor tracing scheme.

Finally, in the third part of the thesis, we propose an extended attack model, namely Pirates 2.0, that goes beyond the formalism of the conventional attacks. We also propose some generalized primitives for broadcast encryption and traitor tracing to fit new practical requirements such as multi-channel and decentralized broadcast encryption.

Résumé

Nous considérons dans cette thèse une généralisation du chiffrement au cas d'utilisateurs multiples, à savoir la diffusion de données chiffrées. Cette généralisation du chiffrement introduit deux nouveaux problèmes au-delà de la confidentialité : comment le centre peut-il identifier les abonnés malhonnêtes (qui fabriquent des décodeurs pirates et sont appelés traîtres) et comment le centre peut-il révoquer les abonnés malhonnêtes sans avoir besoin de mettre à jour les paramètres du système.

Dans un premier temps, nous prenons l'approche combinatoire dans le but de construire des schémas qui supportent à la fois la traçabilité et la révocation. Nous avons en particulier introduit un nouveau type de code, nommé trace&revoke code, et la technique de "shadow group testing" pour contrer les pirates "intelligents". Nous avons en outre proposé une méthode pour intégrer la révocation à quelques schémas de traçage de traîtres fondés sur les codes.

Dans un deuxième temps, nous suivons l'approche algébrique. Tout d'abord, en considérant les schémas fondés sur les couplages sur des courbes elliptiques, nous renforçons la sécurité du schéma de Boneh-Gentry-Waters et le rendons dynamique. Nous étudions ensuite l'application des réseaux euclidiens et proposons un schéma de traçage de traîtres dont la sécurité est assurée sous l'hypothèse bien connue de LWE (Learning with errors).

La dernière partie de la thèse est consacrée à la présentation d'un nouveau type d'attaque en collaboration publique, appelé attaque Pirates 2.0 et quelques extensions du modèle de diffusion de données qui répondent aux exigences pratiques comme les schémas décentralisés ou les schémas multi-canaux.