PSL UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure de Paris

Multi-User Computation over Encrypted Data

Soutenue par
Ngoc Ky NGUYEN
Le 03 décembre 2024

École doctorale n°386
Sciences Mathématiques de Paris Centre

Spécialité
Informatique

Composition du jury :

Nuttapong ATTRAPADUNG AIST	*Rapporteur*
Carla RÀFOLS SALVADOR Universitat Pompeu Fabra	*Rapportrice*
Fabien LAGUILLAUMIE Université de Montpellier	*Examinateur, Président du jury*
Mark MANULIS Universität der Bundeswehr München	*Examinateur*
Damien VERGNAUD Sorbonne Université	*Examinateur*
Duong Hieu PHAN Télécom Paris	*Directeur de thèse*
David POINTCHEVAL Chercheur, Cosmian	*Directeur de thèse*

ENS | PSL

Résumé

Avec la généralisation de TLS sur le web, la confidentialité des échanges s'est renforcée. Mais cela a du même coup ouvert de nouvelles voies aux acteurs malveillants pour attaquer directement les machines individuelles via leur navigateurs, en contournant tous les dispositifs d'analyse de flux, puisque tout transite dans un tunnel chiffré. Ainsi, pour détecter ou empêcher les attaques, nombre de systèmes opèrent une rupture de flux chiffré pour continuer à analyser les paquets en clair, mettant ainsi à mal la confidentialité. Cette thèse va étudier les mécanismes cryptographiques permettant de garantir la confidentialité des données, tout en permettant des analyses pour garantir la sécurité des usagers et des systèmes. Il s'agira pour cela d'adapter le *chiffrement fonctionnel* ou le *chiffrement à base d'attributs*, pour permettre aux sondes d'extraire les seules informations utiles à des fins de cybersécurité.

Cette thèse se focalise sur le chiffrement fonctionnel avec *plusieurs utilisateurs*, en particulier où des *clients* peuvent individuellement chiffrer leurs données partielles, ou des *senders* peuvent engendrer individuellement leur clé fonctionnelle partielle. Ces chiffrés partiels ou clés partielles peuvent être combinés plus tard, si et seulement s'ils partagent un tag commun, *e.g.* un horodatage. Nous obtenons des résultats par rapport à la notion de sécurité du chiffrement fonctionnel dans ce cadre, avec à la fois de nouvelles définitions et de nouvelles constructions. D'une part, nous proposons un cadre pour définir le chiffrement fonctionnel *multi-client* avec un contrôle d'accès fin sur les clés de déchiffrement, qui est généralisé au cas d'une classe de fonctions ayant des informations publiques lors du chiffrement. D'autre part, nous examinons à nouveau le modèle de sécurité du chiffrement fonctionnel *multi-client décentralisé* et raffinons ses contraintes existantes. Finalement, nour construisons des schémas concrets à l'égard de la classe de fonctions pour calculer les produits scalaires, en exploitant les *espaces vectoriels duaux avec couplages* dans les groupes bilinéaires.

cryptologie ⋆ vie privée ⋆ cybersecurité

Abstract

With the generalisation of TLS over the Web, the confidentiality of communications has been reinforced. However, this also led to new attack vectors for adversarial agents to attack directly the individual machines via their browsers, while bypassing all the tools for data-flow analysis, because everything is transmitted through an encrypted channel. Therefore, in order to detect or prevent the attacks, many systems operate by stopping the encrypted channel and continuing to analyse the data packets in the clear, which thus affects badly the confidentiality. This thesis is going to study the cryptographic mechanisms that allow guaranteeing the confidentiality of data, at the same time permitting the analysis to ensure the security of the users and systems. This will require adapting the techniques of *functional encryption* (FE) or *attribute-based encryption* (ABE), which enable the monitors to extract only the useful information for the cybersecurity purposes.

The main setting of our studies is FE with *multiple users*, in particular where we allow multiple *clients* to independently encrypt their partial data, or multiple *senders* to independently generate their partial functional decryption keys. These partial ciphertexts or partial keys can be later jointly combined, only if they are associated to some identical tag, *e.g.* a timestamp. We obtain various results with respect to the security notions of FE in this setting, both definitionally and constructively. On one hand, we give a definitional framework for *multi-client* FE with fine-grained access control over keys, which is furthermore generalized to function classes that authorize some auxiliary public inputs at the time of encryption. On the other hand, we revisit the widely used security model of *decentralized multi-client* FE and refine existing unnatural constraints of the model. Last but not least, we provide concrete constructions in regards of the particular function class for computing inner products, by leveraging the power of *dual pairing vector spaces* in the bilinear group setting.

cryptography ⋆ privacy ⋆ cybersecurity

Acknowledgments

Paris, a November night, it was one of the darkest I could imagine. As I sank into my chair, I found myself in an idealised world where the oracle had already run its infinite table. Somewhere in its memory lies the story of this thesis - written in Vietnamese, English, and French. Fragments of this story, along with my gratitude, now follow - not in any deliberate order, but as they surface naturally in my thoughts.

I am deeply thankful to my two supervisors, David and Hiệu, for their guidance and invaluable advice throughout our time together. My gratitude extends far beyond my doctoral studies, as both David and Hiệu have accompanied me on my journey from the time I prepared for the ENS diploma to the moment these lines are written—a journey for which I am profoundly grateful. Their extensive knowledge, both in research and the practical aspects of daily academic life, as well as their unwavering dedication to mentoring, are qualities I deeply admire. I am proud to count myself among their students. This thesis would not have been possible without the support of the members of my examining committee. Allow me to use first names to convey a sense of familiarity and gratitude: I wholeheartedly thank Nutts, Carla, Fabien, Mark, and Damien for agreeing to be part of my committee, dedicating their time to reading and evaluating my manuscript, and accompanying me through to the conclusion of my PhD.

My PhD story is also fulfilled with wonderful memories with the CASCADE research team and the Computer Science department at the ENS. I would also like to express my heartfelt gratitude to the administrative and technical teams at the department, including those who have since embarked on their well-deserved retirements or new career paths. From my very first day as a pre-doctoral student to the completion of my journey at ENS, their support has been valuable. For an international student navigating an unfamiliar far-from-home environment, this assistance has meant more than words can convey. At the same time, the three years I spent with the CASCADE team have been a pivotal part of my personal growth. Memories from my journey are interwoven, and following the path they reveal, my gratitude goes to Lenaick, for welcoming me warmly on my first day as a PhD candidate and for sharing precious experience, Paola, for her originality, creativity, and inspiring friendliness, Leonard for enlivening our coffee breaks with engaging and varied discussions, Michael for his positive energy in both research and everyday topics, Paul for his kindness, which I deeply respect, Hugo B. for the unforgettable “bzzz” moments and his great spirit that will stay with me for years, Robert for the enriching discussions from which I’ve learned so much, Huy for our Vietnamese conversations and beyond, which nurtured a shared sense of identity, Antoine for the honest and frank chats about CASCADE, Security, ENS, and more, Guirec for his admirable maturity and disciplined personality, Henry for his perpetual sharpness in thought and insatiable curiosity, Eric for his cutting-edge rationality, a rare and precious quality, Nicolas for his constant good humor and inspiring efforts. I am also grateful to the seniors, including Théo, Baptiste, and Hugo S., as well as the interns and visiting members including Alexandra, Xiaxi, Quentin, and Théophile, with whom the moments shared have been most enjoyable. In more recent memories, my gratitude is for the postdocs who chose CASCADE as a stepping stone in their careers: to Huyền for her positive and energetic attitude, to Florette for her praiseworthy attention to detail, to Wissam for his excellent openness to exploration and discussion. I also send my best wishes to the newer generation of CASCADE,

including Amine, Cédric, Florian, Jules, and Laurent, hoping that their upcoming journeys, whether doctoral or postdoctoral, will be fruitful. My respect and appreciation also go to the permanent members of CASCADE over the years, from my pre-doctoral studies to the present: to Michel Abdalla for his warm friendliness and vast knowledge of cryptography, to Brice Minaud for his essential coffee supply and for keen intellect, to Céline Chevalier for her remarkable dedication to teching and mentoring, to Phong Nguyen for opening doors to new perspectives throughout my studies in such a didactic manner, to Hoeteck Wee for the fascinating discussions and his sublime attention to refinement. I'm also profoundly grateful to David Naccache for his critical thinking and his relentless pursuit of dissecting questions from multiple angles.

I am also deeply thankful to the Maths-Info group of Vietnamese students at ENS, including Linh, Tùng, Quân, Tâm, and Trung, for all our discussions on mathematics and computer science—whether around a blackboard, a dinner table, or a ping-pong table. Speaking of ping-pong, a special shoutout goes to Hiếu, Tài, and our ever-expanding ping-pong group. The time we spent together at ENS and other occasions has been a transformative part of my journey. From ENS, my wholehearted gratitude goes to Ariane Mézard for everything I learned during my Bachelor-Master years in her classes and far beyond. Her recommendations for learning opportunities during my PhD were very helpful, I will always cherish and appreciate them. I also want to give a huge thank-you to Sasha, who has been amazing since the very first hours of our class of 2018 at ENS. The projects we worked on together, and our first paper, have left a lasting impact on my intellectual growth. Moreover, I am grateful for the chance to have met so many incredible individuals, including Dung, Duy, Hiền, Hoa, Khánh, Mahshid, Phương, Pouria, Quang (Quang Dao), Yingfei. The shared moments at conferences and seminars were not only great fun but also helped broaden my learning and perspectives far beyond the ENS environment.

Fragments of memories transcend spatial constraints, and I extend my deepest gratitude to An Khương, whose close guidance, sharing, and support during my time in Vietnam laid the foundation not only for my professional progress but also for my personal growth. I am honored to count myself as one of his students. Thanks from the bottom of my heart to Thành, Hạnh, Phước, Danh Nam with whom I shared intensive seminar sessions, as well as to Quang, who has always been wonderful company in the extended seminar group. I am profoundly grateful to Hà Dương and Bảo Châu (Chau Ngo), who, alongside my supervisors Hiệu and David, initiated the scholarship program that enabled me to study at ENS. Without this opportunity, my academic journey would not have been possible. My warmest cheers go to my former CTF team, Efiens, and to Thái (Thai Duong). The CTF competitions we participated in together, along with Thái's insightful blogs, were my first introductions to security and cryptography - sparking my interest in this fascinating field and guiding me toward the culmination of this thesis. A special shoutout goes to Sang, who has been incredible from our days of racing against deadlines during our bachelor years to now, as we both pursue doctoral studies in French cities thousands of miles away from Vietnam.

Among the most precious memories of my journey lies my most profound gratitude to my family - my father Huynh, my mother Dịu, and my brother Bình - for their unwavering support as I embarked on this path, overcoming the highs and lows while being far from home. *Tôi xin cảm ơn bố tôi, mẹ tôi, em tôi. Tôi luôn tin tấm lòng quý của bậc nghiêm-đường, dùng làm tấm gương mà so sánh, làm mối dẫn-lộ để rèn luyện cái lịch-duyệt, trong cái cơn hồ nghi không biết ăn ở thế nào cho phải của thằng con trẻ. Có duyên giao-kết với mọi người, đến nay đã trải mấy năm giời xa nhà, học xem, học xét, thì tưởng cũng không phải là câu chuyện vô ích. Nay, lúc nào cần mỉm cười tôi xin mỉm cười thành thực, lúc nào cần hiểu rõ lòng của nhau tôi sẽ đi sang câu chuyện cảm tình.* Lastly, I address my gratitude to those I have never met, those I may never know, and those who has quietly contributed to this unfolding story. Whether for better or for worse, they continue to uncover the infinite table, as the oracle writes and the story carries on...

Contents

Part I

Introduction and Preliminaries

Chapter

1 Introduction en Français

Chapter content

1.1 Motivations

Quelques mots sur le Contenu. Si l'on devait situer cette thèse, elle se trouverait probablement dans le déjà vaste, mais toujours en expansion, paysage de la recherche en *cryptographie moderne*. Le traitement rigoureux dont fait l'objet la cryptographie moderne est sans doute l'un des aspects les plus fascinants de l'informatique à ce stade de l'histoire. Contrairement à la cryptographie "classique" où la préoccupation constante est de savoir comment communiquer ce que nous voulons à qui nous voulons, sans avoir le contrôle sur le canal de communication, la cryptographie moderne porte sur ses épaules bien plus de tâches : établir une communication privée et authentifiée sur Internet, organiser des votes électroniques secrets et inviolables, déployer des calculs multi-parties "résistants aux fautes", et bien plus encore. En particulier, des notions telles que l'indistinguabilité calculatoire, la propriété pseudoaléatoire, les preuves à divulgation nulle de connaissance sont introduites, des applications importantes telles que le chiffrement et les signatures numériques sont fondées sur des bases théoriques solides, et bien d'autres directions sont explorées. Cette thèse s'intéresse aux deux principales activités de la cryptographie moderne - l'*activité définitionnelle* et l'*activité constructive* - tout en recourant inexorablement à la *difficulté calculatoire*.

Les problèmes abordés dans cette thèse passent d'abord par une étape définitionnelle, où une fonctionnalité claire est provisoirement donnée dans un modèle clairement défini. Elle est "provisoirement donnée" en raison de l'identification inhérente d'un objectif intuitif sous-jacent, qui mène ensuite à un problème cryptographique adéquat. Il est irréaliste et sujet à l'erreur de lister toutes les situations souhaitées concernant le problème cryptographique en question. Ainsi, la fonctionnalité est décrite dans un modèle bien défini (possiblement idéalisé), et nous exigeons que toute solution candidate émule cette opération dans des modèles plus concrets et clairement spécifiés (qui détaillent les capacités de l'attaquant). Ensuite vient la tâche de construction. Étant donné la fonctionnalité fraîchement définie, l'objectif de toute tentative de construction est non seulement de maintenir la *correction* en "opération normale", mais aussi de viser la *sécurité* contre tout adversaire qui essaierait de manipuler le système pour l'éloigner des états typiques. Cet objectif est notoirement difficile. Les tentatives adversariales peuvent être conçues après la construction terminée, elles peuvent dévier de *toutes* les actions envisagées lors de la conception. On ne peut pas se contenter d'avoir une solution qui fonctionne uniquement pour une liste prédéterminée de *stratégies*, et les heuristiques peuvent seulement fournir des idées sur l'environnement *fonctionnellement envisagé* pour l'opération mais pas sur l'environnement *hostilement choisi*. Pour rendre les choses encore plus difficiles, les constructions candidates pour une utilisation pratique doivent également satisfaire certaines exigences d'*efficacité*.

En remontant à l'aube de la cryptographie moderne, ces deux étapes interdépendantes,

définition-construction, apparaissent de manière omniprésente dans tous les travaux fondateurs sur le chiffrement [GM84], les signatures numériques [GMR88], les générateurs pseudo-aléatoires [BM84, Yao82], les preuves à divulgation nulle de connaissance [GMW86], les protocoles multi-parties pour calculer de manière sécurisée des fonctions générales [Yao86, GMW87], et bien d'autres.

Cette thèse se concentre sur le *chiffrement et ses notions avancées*, qui est l'un des problèmes les plus emblématiques de la cryptographie moderne. Dans tous les chapitres, la structure suit de près le paradigme en deux étapes mentionné ci-dessus : nous identifions d'abord la préoccupation naturelle et le problème cryptographique sous-jacent (via une vue d'ensemble technique), puis nous définissons formellement le problème, et enfin nous montrons que des solutions candidates peuvent exister. Pour cette dernière partie, nous nous limitons à fournir une *preuve de faisabilité* : en principe, les problèmes que nous définissons peuvent être résolus. Nous ne visons pas à fournir des solutions pratiques, bien que certaines discussions sur l'(im)praticabilité des solutions soient données à l'occasion.

Chiffrement fonctionnel (FE). La communication sécurisée utilise fondamentalement des schemas de chiffrement comme bloc de construction : seul un destinataire ayant une clé de déchiffrement peut déchiffrer un chiffré pour obtenir le message sous-jacent, sinon il est garanti par la sécurité sémantique du schéma qu'aucune information sur les données en clair ne fuite. Cette nature de *tout-ou-rien* est restée la norme pendant longtemps. Cependant, l'apparition de systèmes de communication plus complexes et hiérarchiques implique un besoin accru de contrôle plus fin sur les fuites d'information qui viennent des chiffrés. Ainsi, une grande motivation pour les notions avancées de chiffrement est née. Le progrès culmine avec le Chiffrement Fonctionnel (FE) [SW05, BSW11] introduit par Boneh, Sahai et Waters. Le FE permet un traitement plus fin de ce qu'un destinataire peut recevoir : chaque clé de déchiffrement est associée à une fonction et le résultat du déchiffrement selon cette clé est garanti de ne révéler rien de plus que l'évaluation de cette fonction sur le texte en clair sous-jacent. En principe, ces *clés de déchiffrement fonctionnelles* permettent de contrôler la quantité d'information à ce qui correspond au maximum à l'analyse fonctionnelle sur le texte en clair, et rien de plus.

FE reçoit un grand intérêt de la part de la communauté des cryptographes, premièrement en tant qu'une généralisation du chiffrement basé sur l'identité (IBE) [Sha84, Coc01, BF01, BGH07] et du chiffrement basé sur les attributs (ABE) [SW05, GPSW06, OSW07, ALdP11, OT12b], qui malheureusement ne fournissent qu'un contrôle d'accès sur le déchiffrement en conservant un résultat tout-ou-rien. Abdalla *et al.* [ABDP15] sont les premiers à réaliser du FE pour des classes concrètes de fonctions, où une clé de déchiffrement fonctionnelle permet de déchiffrer en obtenant un produit scalaire entre un vecteur-fonction et un vecteur-message. Un tel FE est appelé *inner product* FE (IPFE). Une série d'études sur le IPFE s'étend sur plus d'une décennie, apportant plusieurs résultats intéressants, visant à améliorer les constructions existantes pour les produits scalaires [ALS16, BBL17, CLT18] ou pour atteindre les fonctions quadratiques [BCFG17, Gay20, AS17, Lin17], ou pour concocter de nouvelles notions avancées [GVW15] ainsi que des liens avec d'autres aspects de la cryptographie moderne [AJ15, BV15]. En particulier, un intérêt énorme se manifeste dans le cadre *multi-utilisateurs* [CDG+18b, ABKW19, ABG19, LT19, CDSG+20, AGT21b, SV23, NPP23a, NPS24a]. Ce cadre autorise un grand nombre d'utilisateurs à rejoindre un système de FE et à contribuer soit à un chiffré conjoint soit à une clé conjointe. L'agrégation des clés conjointes permettra de déchiffrer l'agrégation des chiffrés conjoints sous certaines conditions, *e.g.* ces clés partielles partagent un tag fonctionnel identique et ces chiffrés partiels partagent un horodatage identique. Différentes notions de FE peuvent émerger avec différentes variantes de sécurité, avec ou sans la confidentialité des fonctions, par rapport à la corruption des clés des utilisateurs, ainsi que la flexibilité dans le nombre d'utilisateurs. À première vue, ce cadre ressemble aux protocoles multi-parties (MPC), où les joueurs fournissent leurs entrées pour une évaluation fonctionnelle commune tout en maintenant la confidentialité de leurs entrées. Cependant, la principale différence réside dans le fait que le FE est censé être non-interactif à la

fois en termes de chiffrement et de déchiffrement, et sera ainsi préféré dans un environnement décentralisé ou dynamique. FE avec un seul chiffreur pourrait être intéressant d'un point de vue théorique, mais dans la vie réelle, le nombre de fonctions vraiment utiles peut être limité. Quand ce nombre de fonctions est faible, n'importe quel PKE peut être converti en FE en chiffrant de plus les évaluations selon plusieurs fonctions sous une clé spécifique. Cette approche n'est pas possible pour plusieurs utilisateurs, même lorsqu'il n'y a qu'une fonction unique à considérer.

Ainsi, le cœur de cette thèse se concentre sur les schémas de FE dans le cadre multi-utilisateurs pour des classes concrètes de fonctions telles que les produits scalaires.

Plus sur le Cadre Multi-Utilisateurs. En examinant de plus près comment définir raisonnablement le FE dans le cadre multi-utilisateurs, il apparaît que des changements définitionnels sont nécessaires. Naïvement, comme nous l'avons soulevé ci-dessus, quand le nombre de fonctions utiles est polynomialement borné, le Chiffrement à Clé Publique (PKE) peut être utilisé pour chiffrer chaque évaluation de fonction en utilisant une clé publique différente, lors de la participation d'un unique chiffreur. Malheureusement, dans le contexte avec plusieurs utilisateurs où évaluer une fonction requiert des entrées différentes provenant de différents utilisateurs, cette approche basée sur le PKE est impossible puisque le résultat n'est pas connu lors du chiffrement, même s'il n'y a qu'une seule fonction. Une étude systématique du Chiffrement Fonctionnel à Plusieurs Entrées (MIFE) et du Chiffrement Fonctionnel à Plusieurs Clients (MCFE) est conduite dans [GGG+14, GKL+13]. Les deux mécanismes, MIFE et MCFE, permettent de déchiffrer une évaluation de fonction sur une liste d'entrées. Dans le MIFE, un *seul chiffreur* peut chiffrer différentes entrées dans cette liste à des moments distincts, tandis que dans le MCFE il y a plusieurs *clients* qui chiffrent indépendamment leurs entrées respectives. Une autorité fiable est demandée pour délivrer des clés de déchiffrement fonctionnelles afin de déchiffrer les chiffrés. Enfin, à cause des combinaisons potentielles parmi les chiffrés pour le déchiffrement, MIFE et MCFE sont définis comme des primitives à clé secrète pour avoir des propriétés de sécurité non-triviales.

Techniquement, dans le cas du MCFE, un index i pour chaque client et un tag tag (typiquement de type horodatage) sont utilisés pour chaque chiffrement : $(c_1 = \mathsf{Enc}(1, x_1, \mathsf{tag}), \ldots, c_n = \mathsf{Enc}(n, x_n, \mathsf{tag}))$. Une personne qui possède une clé de déchiffrement fonctionnelle dk_f, pour une fonction f n-aire et plusieurs chiffrés (pour le même tag, dans le cas de MCFE) peut calculer $f(x_1, \ldots, x_n)$ mais rien de plus sur les x_i individuels. Implicitement, les clients doivent avoir la possibilité de se coordonner sur les tags, et sur leur usage pratique. En particulier, dans le MCFE, une combinaison des chiffrés qui ont été générés avec des tags différents ne donne pas un chiffré global valide et l'attaquant n'apprendra rien sur cette combinaison. Cela mène à plus de versatilité parce que chiffrer x_i sous tag apporte un sens distinct de chiffrer x_i sous $\mathsf{tag}' \neq \mathsf{tag}$. D'un autre côté, le MIFE n'utilise pas de tags et dès qu'un chiffré est calculé, il peut être utilisé autant de fois pour des combinaisons différentes. En revanche, dans les deux situations de MIFE/MCFE, nous rappelons que le chiffrement doit requérir une clé secrète, sinon n'importe qui pourrait compléter le vecteur de chiffrés initialisé par un utilisateur de plusieurs façons, et ainsi obtenir plusieurs évaluations à partir d'une clé de déchiffrement fonctionnelle. Mais parce que le chiffrement nécessite maintenant une clé secrète dans MCFE, pour chaque composant c_i du chiffré, un sous-ensemble de ces clés peut être corrompu. Cette thèse se focalise sur le MCFE et sa généralisation telle que le MCFE *Décentralisé* (DMCFE). Dans la suite, nous révisons d'abord les propriétés additionnelles qui peuvent accompagner les clés fonctionnelles, puis nous donnons un point de vue plus développé du FE multi-utilisateur au-delà du MCFE.

Propriétés des clés de déchiffrement fonctionnelles. Les deux propriétés naturelles que nous pouvons demander d'une clé de déchiffrement fonctionnelle concernent *un contrôle sur leur usage* et *la confidentialité de la fonction associée.*

Contrôle d'Accès Fin sur les Clés de Déchiffrement Fonctionnelles. Même avant l'introduction

du FE, la gestion des clés de déchiffrement était toujours un problème classique non seulement d'un point de vue théorique mais également dans des scénarios pratiques. Les études sur le chiffrement par diffusion, les systèmes de révocation, et plus généralement, sur l'ABE lui-même, *e.g.* voir [Wee21, Wee22, FWW23] visent tous à aborder cette question dans des situations de plus en plus complexes. Intuitivement, chaque clé de déchiffrement fonctionnelle fait fuiter de l'information sur la clé-maître secrète suite aux évaluations de fonction lors du déchiffrement, *e.g.* des informations sur les secrets qui sont utilisés pour engendrer cette clé de déchiffrement fonctionnelle. Cela signifie que lorsqu'une quantité suffisante de fuites est accumulée, il devient possible pour un attaquant de retrouver la clé-maître de l'autorité. Cette dernière étape peut être aussi simple que résoudre un système d'équations linéaires, par exemple dans le cas du IPFE. Nous aimerions souligner le fait que dans le cas des classes de fonctions générales, il est possible d'imposer un contrôle sur le déchiffrement via la fonction elle-même, au prix de l'efficacité. Abdalla *et al.* [ACGU20] a initié une série de travaux sur l'intégration d'un mécanisme de *contrôle basé sur les attributs* au IPFE. Les travaux qui suivent incluent [NPP22a, NPP25] pour les produits scalaires et [ATY23a] pour la classe des sommes attributs-pondérées. Le défi principal réside dans le fait qu'ajouter un contrôle non trivial basé sur les attributs aux clés de déchiffrement, tout en maintenant l'efficacité, implique nécessairement une amélioration de la classe. Ainsi, un traitement de IPFE avec contrôle d'accès entraîne un traitement d'une classe plus riche avec plus de finesse technique.

La Confidentialité de Fonction. La sécurité sémantique du FE garantit tout d'abord la confidentialité du message, étant donné le chiffré et les clés de déchiffrement fonctionnelles. Dans des cas d'usage différents, les clés de déchiffrement fonctionnelles d'un schéma de FE peuvent révéler des informations sur la fonction sous-jacente. Ainsi, de façon symétrique, une question légitime est de connaître la quantité d'information que de telles clés peuvent révéler sur la fonction, surtout quand cette fonction contient des paramètres sensibles. Des exemples spécifiques incluent des modèles d'apprentissage profonds dont les paramètres constituent la fonction à évaluer, et ces paramètres proviennent de processus d'apprentissage coûteux, qu'il est dans l'intérêt de garder secrets afin d'assurer les services du prestataire. De manière analogue, le développement d'autres notions avancées du PKE témoigne également de certains besoins symétriques si nécessaire, tels que des versions anonymes de l'IBE ou des versions à attributs-cachés/prédicats-cachés de l'ABE. Pour le FE, cette propriété s'appelle *function-hiding* (FH) et les schémas FH-FE sont également un objet théorique important. En utilisant un FH-FE, plusieurs travaux [Lin17, Gay20] explorent des approches différentes pour atteindre du FE par rapport à la classe des fonctions quadratiques, ou plus récemment [AGT21a, AGT22], réussissent à réaliser l'évaluation quadratique dans le cadre MIFE. À l'égard de la classe des produits scalaires, la propriété de function-hiding elle-même inspire certains progrès récents dans le contexte des multi-utilisateurs [SV23, NPS24a, Ngu24].

1.2 Mode d'Emploi: Comment Lire la Suite

Nous venons de survoler les principaux sujets qui ont été abordés dans notre travail de cette thèse : le contexte du FE avec plusieurs utilisateurs et deux propriétés qui accompagnent les clés de déchiffrement fonctionnelles, *i.e. un contrôle sur l'usage des clés* et *la confidentialité de la fonction associée*. Pour conclure cette partie d'introduction en français, nous aimerions présenter la structure de ce manuscrit de thèse, et des parcours de lecture possible selon les intérêts du lecteur. Premièrement, les contenus communs pour tout ordre de lecture sont :

1. *(Une préparation introductive plus élaborée)* Le passage "**Extensions of FE in the Multi-User Setting.**" dans la Section 2.1 enchaîne sur notre sujet du MIFE/MCFE ci-dessus. Il y aura beaucoup plus de détails sur les notions de FE à plusieurs utilisateurs, au-delà du MCFE, *e.g.* MCFE *Décentralisé* (DMCFE) et FE *Dynamique et Décentralisé* (DDFE). Nous

donnons des références sur les avancées récentes dans ces domaines.

2. *(Des questions de recherche & les contributions)* Dans la Section 2.2, nous formulons les questions qui sous-tendent nos résultats au cours de cette thèse. Puis, la Section 2.3 explicite ce que nous avons obtenu vis-à-vis des dites questions.

3. *(Notations & Préliminaires)* Des notions des *espaces vectoriels duaux avec couplages* dans les groupes bilinéaires sont essentielles, et il est fortement conseillé de parcourir la Section 3.3 qui présente les définitions avec des exemples détaillés.

Ensuite, selon l'intérêt, les chapitres de cette thèse peuvent être parcourus comme suit :

(Sur un contrôle sur l'usage des clés) Le Chapitre 4 détaille nos résultats dans le cas du MCFE, puis le Chapitre 5 améliore le modèle de sécurité dans le cas du (D)MCFE, qui s'applique aussi dans le cas avec un contrôle d'accès. Finalement, le Chapitre 6 intègre le modèle plus fort du Chapitre 5 et étend les résultats du Chapitre 4.

(Sur la propriété function-hiding) Le Chapitre 7 présente notre résultat sur le FH-IP-DMCFE.

(L'usage des espaces vectoriels duaux avec couplages - DPVS) Une preuve détaillée qui utilise DPVS est donnée pour le Théorème 6.11, qui démontre à la fois son pouvoir calculatoire et information-théorique. Plus tard, dans le Chapitre 7, un lemme modulaire du type "swapping" des contenus des vecteurs dans DPVS est annoncé et prouvé en détail, qui donne un deuxième exemple sur l'utilisation des techniques de changement de base. Avant toutes ces preuves techniques, nous esquissons les grandes étapes avec des aperçus techniques.

Chapter

2 Introduction

Chapter content

2.1 Context and Motivations

From this Thesis: A Point on Expectation. If to be positioned, this thesis is likely finding itself among the vast and expanding landscape of research in *modern cryptography*. The witnessed rigorous treatment of modern cryptography is possibly one of the fascinating sides of computer science at this stage of history. In contrast to "classical" cryptography in which the everlasting concern is how to communicate what we want to those we want, without having control over the channel of communication, modern cryptography carries on its shoulders many more tasks: establishing private and authenticated communication over the Internet, holding tamper-proof and secret electronic voting, deploying "fault-resilient" multi-party computation, and more. In particular, notions such as computational indistinguishability, pseudorandomness, zero-knowledge proofs are introduced, important applications such as encryption and digital signatures are founded on firm theoretical backgrounds, and much more directions are uncovered. This thesis bears some hints to the two main activities of modern cryptography - the *definitional activity* and the *constructive activity* - while inexorably resorting to *computational difficulty* almost all over.

The problems that are approached in this thesis go first through a definitional stage, where a clear functionality is tentatively given in a clearly defined model. It is "tentatively given" due to the inherent identification of some underlying intuitive objective, which afterwards leads to an adequate cryptographic problem. It is unrealistic and prone to error to list all desired situations regarding the cryptographic problem at hand. Hence, the functionality is described operational in some (possibly idealized) well-defined model and we require any candidate solution emulate this operation in more concrete and clearly specified models (which details the adversary's abilities). Then comes the constructive task. Given the freshly defined functionality, the goal of any construction attempt is not only maintaining *correctness* in "normal operation", but also aiming for *security* against any adversary who will try to manipulate the system away from typical states. This goal is infamously hard. The adversarial attempts can be devised after the construction is finished, they can deviate from *all* envisaged actions at designing time. One cannot be content to have a solution that works only for a fixed predetermined list of *strategies*,

and heuristics can only go so far as giving ideas on the *functionally envisoned* environment for operation but not the *maliciously chosen* environment. To make things even more challenging, candidate constructions for practical use also have to satisfy some *efficiency* requirements.

Tracing back to the dawn of modern cryptography, those two interleaving definitional-constructive steps appear ubiquitously in all seminal works on encryption [GM84], digital signatures [GMR88], pseudorandom generators [BM84, Yao82], zero-knowledge proofs [GMW86], multi-party protocols for securely computing general functions [Yao86, GMW87], and more.

This thesis zooms in on *encryption and its advanced notions*, which is one of the most archetypical problems of modern cryptography. In all chapters the structure closely follows the two-step paradigm that is mentioned above: we first identify the natural concern and underlying cryptographic problem (via some technical overview), then define formally the problem, then show that candidate solutions may exist. For the latter part, we limit ourselves to providing a *claim of feasibility*: in principle the problems we define may be solved. We do not aim to give practical solutions, though some discussions on (im)praticality of solutions are given occasionnally.

Functional Encryption (FE). Secure communication fundamentally uses encryption schemes as its building block: only with a decryption key can a recipient decrypt a ciphertext to obtain the underlying message, otherwise it is guaranteed by the scheme's semantic security that nothing is leaked about the plaintext data. This *all-or-nothing* nature remains the standard for a long time. However, the fact that more complex and hierarchical comunication systems appear means more need of a fine-grained control over the leakage from ciphertexts. Thus a great deal of motivation into advanced notions of encryption is generated. The progress culminates in Functional Encryption (FE) [SW05, BSW11] that is introduced by Boneh, Sahai and Waters. FE allows a finer treatment of what a recipient can obtain: each decryption key is associated with a function and the decryption result *as per* this key is guaranteed to reveal no more than the foregoing function evaluation on the underlying plaintext. In principle, these *functional decryption keys* permit controlling the amount of information to be at most the functional analysis over the plaintext, and not more.

FE received large interest from the cryptographic community, first as a generalization of Identity-Based Encryption (IBE) [Sha84, Coc01, BF01, BGH07] and Attribute-Based Encryption (ABE) [SW05, GPSW06, OSW07, ALdP11, OT12b], which unfortunately provieds only access control over decryption keys while retaining an all-or-nothing result. Realizing FE for concrete function classes is first done by Abdalla *et al.* [ABDP15], where a functional decryption key allows decrypting to an inner product between some function vector and the plaintext vector. A such FE scheme is coined *inner product* FE (IPFE). A long line of works on IPFE spans over almost a decade, with numerous interesting results, to improve existing constructions for inner products [ALS16, BBL17, CLT18] to move to quadratic functions [BCFG17, Gay20, AS17, Lin17], or to concoct new advanced notions [GVW15] as well as relate to other notions in cryptography [AJ15, BV15]. In particular, great enthusiast is manifested in the *multi-user* setting [CDG+18b, ABKW19, ABG19, LT19, CDSG+20, AGT21b, SV23, NPP23a, NPS24a]. This setting allows multiple users to take part in an FE system and contribute either to some joint ciphertext or joint functional key. Aggregation of contributed keys will be able to decrypt aggregation of contributed ciphertexts upon certain conditions, *e.g.* those partial keys share identical function tags and those partial ciphertexts share identical timestamps. A number of different FE notions can stem from different flavors of security, with or without function privacy, regarding corruption of users' keys, as well as flexibility in the number of users. At first sights this setting seems similar to multi-party computation (MPC), where several players provide their inputs for a jointly function evaluation while maintaining their input privacy. However, the principal difference is found in the fact that FE is expected to be non-interactive in both encryption and decryption, and is thus more preferable in decentralized or dynamic environments. While FE with a single encryptor might be of theoretical interest, in real-life, the number of

really useful functions may be limited. When this number of functions is small, any PKE can be converted into FE by additionally encrypting the evaluations by the various functions under specific keys. This approach is impossible for multiple users, even when a unique fixed function is considered.

The center of attention of this thesis is FE schemes in the multi-user setting for concrete function classes such as inner products.

More on Multi-User Settings. Taking a closer look at how to meaningfully define FE in the multi-user setting, it appears that inherent definitional changes are necesary. Naively, as mentioned earlier, when the number of useful functions is polynomially large, Public Key Encryption (PKE) can be used to encrypt each function evaluation using a different public key. Unfortunately in the multi-user setting where evaluating a function requires different inputs from different users, this PKE-based approach is impossible as the result is unknown at encryption time, even when there is only one function. A systematic study into Multi-Input Functional Encryption (MIFE) and Multi-Client Functional Encryption (MCFE) is then conducted in [GGG+14, GKL+13]. Both MIFE and MCFE permit decrypting to a function evaluation over a list of inputs. In MIFE, a *single encryptor* can encrypt different inputs in the list at different time, whereas in MCFE there are multiple *clients* who independently encrypt their respective input. A trusted authority is required to issue functional keys to decrypt jointly the ciphertexts. Last but not least, due to possible combination of ciphertexts for decryption, both MIFE and MCFE are defined as secret-key primitives so as to have non-trivial security guarantees.

Technically, in the case of MCFE, an index i for each client and a (typically time-based) tag tag are used for every encryption: $(c_1 = \mathsf{Enc}(1, x_1, \mathsf{tag}), \ldots, c_n = \mathsf{Enc}(n, x_n, \mathsf{tag}))$. Anyone owning a functional decryption key dk_f, for an n-ary function f and multiple ciphertexts (for the same tag tag, in the case of MCFE) can compute $f(x_1, \ldots, x_n)$ but nothing else about the individual x_i's. Implicitly, clients have to be able to coordinate together on the tags, and different usability in practice. In particular, in MCFE, the combination of ciphertexts generated for different tags does not give a valid global ciphertext and the adversary learns nothing from it. This leads to more versatility since encrypting x_i under tag has a different meaning from encrypting x_i under $\mathsf{tag}' \neq \mathsf{tag}$. On the other hand, MIFE does not use tags and once a ciphertext of x_i is computed, it can be reused for different combinations. However, in both situations of MIFE/MCFE, we recall that encryption must require a private key, otherwise anybody could complete the vector initiated by a user in many ways, and then obtain many various evaluations from a unique functional decryption key. But then, since encryption needs a private key per client in MCFE, for each ciphertext component c_i, some of these keys might get corrupted. The focal point of this thesis is narrowed down to MCFE and its generalization such as the *Decentralized* MCFE (DMCFE). In the following we first review further properties that can accompany the FE functional keys, then give a broader point of view of multi-user FE that even goes beyond MCFE for the sake of a general state of the art.

Properties of Functional Decryption Keys. Two natural and important properties that one can require from the functional keys consist of *controlling their usage* and *hiding the associated function*.

Fine-Grained Control over Functional Decryption Keys. Even before the dawn of FE, managing decryption keys is always a classical problem not only from theoretical points of view but also in real-life scenario. Extensive studies in broadcast encryption, revocation systems, and more generally, of ABE itself, *e.g.* see [Wee21, Wee22, FWW23] all aim at tackling this question in evolvingly more complex situation. Intuitively, each functional decryption key gives away some information on the master secret key following the function evaluation when decrypting, *e.g.* information on the secrets that are used to generate the functional key. This means when accumulating some sufficient amount of leakage, it is feasible for an adversery to recover the

master secret key of the issuance authority. The later recovery step can be as easy as solving linear equations, for example in the case of IPFE. It is worth noting that for general function classes, one can impose decryption control in the function itself at the cost of efficiency. For practical interests, Abdalla *et al.* [ACGU20] started the line of works on integrating a mechanism of *attribute-based access control* into FE for inner products. Follow-up works include [NPP22a, NPP25] for inner products and [ATY23a] for the class of attribute-weighted sums. The main challenge resides in the fact that adding non-trivial access control over decryption keys, while preserving efficiency of the basic IPFE blueprints, is necessarily an enhancement to the function class. Hence, moving from IPFE to IPFE with access control incurs dealing with richer classes and with more technical delicacies.

Function Privacy. Originally an advanced PKE notion, the semantic security of FE guarantees first and foremost the confidentiality of the plaintext, given the ciphertext and the functional keys. When deploying in different use cases, the functional keys of an FE scheme can happen to carry with them information about the underlying function. From a symmetric consideration, a valid question that can be asked is how much information such keys can reveal about the function, especially when the function contains sensitive parameters. Specific examples include models of machine/deep learning whose parameters constitute the function to be evaluated, and these parameters are often the result of expensive training processes, which are kept secret to ensure the business model of the service provider. In an analogous manner, the development of other advanced PKE notions also witness somewhat symmetric privacy requirements when necessary, such as anonymous versions of IBE or attribute/predicate-hiding versions of ABE. For FE schemes, this property is named *function-hiding* (FH) and FH-FE schemes also turn out to be an important theoretical object. From a FH-IPFE, various works [Lin17, Gay20] exploit different approaches to achieve FE schemes for quadratic functions, or more recently [AGT21a, AGT22] succesfully realize quadratic evaluation in the MIFE setting. Sticking to the class of inner products, the property of function-hiding itself inspires recent progress in the *multi-user* setting, *e.g.* see [SV23, NPS24a, Ngu24].

Extensions of FE in the Multi-User Setting. Not long after the seminal works that introduce FE [SW05, BSW11], extensions of FE into the multi-user setting are initiated by Goldwasser *et al.* [GGG+14, GKL+13] with *Multi-Client Functional Encryption* (MCFE) and *Multi-Input Functional Encryption* (MIFE). Speaking of the concrete class to compute inner products, efforts are made over the years for more refinements in definitional frameworks, efficiency, and more [DOT18, CDG+18a, CDG+18b, ACF+18, ABKW19, ABG19, LT19, CDSG+20, ACGU20, NPP22a]. Going beyond inner products, a rising level of interest is directed towards quadratic functions [AGT21a, dPP22, AGT22] or attribute-weighted sums [ATY23a, NPS25, Ngu24].

(Decentralized) Multi-Client Functional Encryption. As a reminder, from [GGG+14, GKL+13, CDG+18a] the MCFE setting allows a fixed number of clients to independently encrypt their data using their private encryption keys, where a functional key can jointly decrypt these ciphertexts only if they share an identical tag. The trust model requires an authority to issue those functional keys. It is of immediate preference to remove this reliance on some trusted party. In the same work, Chotard *et al.* [CDG+18a] questions the need of a central authority for distributing functional keys in MCFE and proposes the notion of *Decentralized* MCFE (DMCFE). In DMCFE, there is no need of an authority anymore and the key generation can be done by multiple *senders* in a collaborative manner. To do so, a fixed number of senders interact during the setup phase, *e.g.* in an MPC-based fashion, so that each sender gets a secret key sk_i. This sk_i allows the sender to contribute to the generation of a functional key DK_f for a function f, and each contribution is associated with a key tag tag-f for later joint combination, where tag-f can contain the description of f itself. As one might expect, liberating DMCFE from the requirement of a trusted authority at the cost of one more interactive phase during setup is a significant change

in terms of the trust model. After [CDG+18a] many follow-up works dive into different angles, ranging from constructions and applications [LT19, ABKW19, ABG19, QLH+24] to security enhancements [NPP23a, NPS24b]. In terms of the supported functions, all aforementioned works focus on the class of inner products, while some progress for more expressive classes to compute attribute-weighted sums is only achieved very recently in [ATY23a, Ngu24]. More interestingly, the two works [ATY23a, Ngu24] focus on a more general version of DMCFE, which is coined *Dynamic Decentralized Functional Encryption* (DDFE) and in the end gives implicit constructions of DMCFE. We will come back to the notion of DDFE in the following, after addressing a technical subtlety of the message/key tags up to the current notion of DMCFE.

Repetitions under One Message/Key Tag. It should be clear from our discussion so far that the functionality of DMCFE requires ciphertexts to share identical tags tag and keys to share identical key tags tag-f so that they can be combiningly decrypted. From a security standpoint, an almost immediate question is what kind of properties we need regarding these tags. Initial works and follow-ups on DMCFE [CDG+18a, LT19] ignore repeated adversarial queries to some slot i under the same tag tag or tag-f. It is argued in [CDG+18a] that it is up to the user's responsibility not to use the same mesage and key tag twice, neither for encryption nor for key-generation, respectively. Nonetheless, we believe that proving security under a repeated usage of tags is still important. First of all, mistakenly or maliciously re-using of tags can happen in practice. Furthermore, given a (D)MCFE that is provably secure even when an adversary can obtain different ciphertexts on the same (i, tag), one can get a MIFE [GGG+14, GKL+13] for the same function class whose semantic security is preserved from the (D)MCFE. In short, the MIFE fixes a public tag and runs the algorithms of the (D)MCFE using that public tag for all encryption/decryption. Any MIFE adversary that tries to combine MIFE ciphertexts are essentially some (D)MCFE adversary who tries to break the semantic security using repeated ciphertexts under the fixed public tag. This connection is studied in preceding works [ABKW19, ABG19] and recently confirmed in [ATY23a, NPP25]. Our final aim for the security of DMCFE and the like will allow repetitions on both message/key tags.

Dynamic Decentralized Functional Encryption. An attentive reader may notice that one shortcoming of DMCFE is the obligation of fixing the numbers of clients and senders at setup time. Despite a great advantage in terms of trust model, resolving this rigidity of DMCFE can enable even more applications in which users can enter the system flexibly at various stages. Chotard *et al.* [CDSG+20] generalize DMCFE and define the notion of *Dynamic Decentralized Functional Encryption* (DDFE). All decentralized properties of DMCFE are preserved in DDFE, whereas the latter allows a *non-interactive global* setup instead of an interactive setup in DMCFE. More specifcally, this global setup provides some public parameters using which any user, clients and senders alike, can run their *local* setup and generate their secret-public keys pair to join the system. At any time, any set of users $\mathcal{U}_M$ can independently encrypt their individual data to contribute to a list of ciphertexts $(\mathsf{CT}_i)_{i\in\mathcal{U}_M}$ under some message tag tag. Similarly, a set of users $\mathcal{U}_K$ can independently contribute to a list of functional keys $(\mathsf{DK}_i)_{i\in\mathcal{U}_K}$ under some key tag tag-f. We recall the usage of tags here is similar to that in (D)MCFE: the ciphertexts and the functional keys can be combined only if they have the same message and key tags, respectively. A DDFE scheme allows jointly decrypting a list of ciphertexts $(\mathsf{CT}_i)_{i\in\mathcal{U}_M}$ using a list of functional keys $(\mathsf{DK}_i)_{i\in\mathcal{U}_K}$, without resorting to any central authority. Chotard *et al.* [CDSG+20] provide a DDFE for the class of inner products, which is then followed by [AGT21b] that revisits and improves by first constructing a FH-MCFE for inner products and then lifting it to a FH-DDFE for inner products. In terms of more expressive function classes, [ATY23a] presents the first DDFE to compute attribute-weighted sums (AWS). All constructions attain only *selective* security under *static* corruption in the ROM, *i.e.* the adversary makes all encryption, key-generation and corruption queries up front in one shot. Very recently, [Ngu24] leverages the state-of-the-art from [AGT21b, ATY23a] to give the first FH-DDFE for inner products and DDFE for AWS without ROM. All mentioned works allow repetitions for message tags. Regarding the security

against repetitions on key tags, [AGT21b] and the FH-IP-DDFE of [Ngu24] explicitly exclude them from their security model, whereas [CDSG+20, ATY23a] and the DDFE for AWS of [Ngu24] consider a simplified functionality that does not consider key generation with respect to tags, thus there is no notion of repetition for key tags.

A Closer Look: How to Define Security of Multi-User FE? The preceding paragraphs elaborate long lines of works on FE in the multi-user setting, from the seminal MIFE/MCFE to the recent DDFE notion. As the notion evolves, or more properties are required such as function-hiding/access control over keys, the security model is necessarily becoming more complicated. It is important to notice that since the introduction of (D)MCFE in [CDG+18a], their security model for (D)MCFE schemes is widely used as a standard. In short, the power of an adversary in the security model of (D)MCFE allows some corruption with respect to the secret keys of the client (to encrypt) or sender (to generate partial functional keys), the security must take into account these corrupted keys, and exclude attacks that trivially breaks the scheme. For instance, in the "vanilla" setting of MCFE, that is, without function-hiding nor key access control, whenever an encryption key ek_i is corrupted, the adversary must query the same challenges $x_i^{(0)} = x_i^{(1)}$ for the corrupted client i following the work of [CDG+18a]. In the discussion after [CDG+18a, Def. 2], it was justified that "*since the encryption might be deterministic, if we allow Left-or-Right encryption queries even for corrupted encryption keys, these queries should be on identical messages: with the encryption key, the adversary could simply re-encrypt and compare in case of deterministic encryption*". However, it is also left open whether one can relax this constraint, which will lead to a stronger security model, as hinted in [CDG+18a] by using probabilistic encryption schemes.

2.2 Research Questions

The above expository discussion leads us to various research questions. The body of works that build up to this thesis focuses on MCFE up to their generalization DMCFE/DDFE for inner products or for attribute-weighted sums, with or without access-control, with or without function-hiding. The following questions may overlap, but we break them down into different aspects for the sake of clarity. It should be clear that resolving multiple of them at the same time is notoriously hard:

1. Following the introduction of (D)MCFE in the seminal paper [CDG+18a], all follow-up studies on (D)MCFE, for instance [CDG+18b, ABKW19, ABG19, LT19, CDSG+20, AGT21b], administered an *admissibility condition* and restricted particularly adversaries to asking the challenge components $\mathbf{x}_i^{(0)} = \mathbf{x}_i^{(1)}$ in case of a corrupted i. *Can we relax this constraint, which will lead to a stronger security model for (D)MCFE, as hinted in [CDG+18a]?*

2. All cited DDFE schemes [CDSG+20, AGT21b, ATY23a, Ngu24] attain only *selective* security under *static* corruption. *How far can we push for adaptive security of DDFE?*

3. Regarding the attribute-based access control over functional keys, existing works can go as far as MIFE, for inner products in [ACGU20] and for attribute-weighted sums in [ATY23a]. *How further can we integrate access control into the multi-user setting,* e.g. *starting from MCFE and potentially all the way up to DDFE?*

4. All cited DDFE schemes [CDSG+20, AGT21b, ATY23a, Ngu24] rely on group-based assumptions and do not provide post-quantum security. The only multi-user FE scheme in the post-quantum regime comes from the DMCFE of [LT19] for inner products and relies on the *Learning with Error* (LWE) assumption. *How far can we push for post-quantum security for DDFE?*

5. The security against repetitions on key tags is either excluded in the FH-IP-DDFE from [AGT21b, Ngu24], or not explicitly considered in [CDSG+20] for IP-DDFE and in [ATY23a, Ngu24] for AWS-DDFE. *How can we achieve security against repetitions for both encryption and key-generation queries in the FH-DDFE and/or DDFE with access control setting?*

2.3 Contributions

We present in this thesis some of our principal contributions that resolve completely or partially the questions from Section 2.2. In each subsection (2.3.1, 2.3.2, 2.3.3, 2.3.4), we detail the main results that will be presented in this thesis, and relate them to the research questions. Other related contributions and other miscellaneous ongoing works that are not presented in this thesis are mentioned in Section 2.3.5.

2.3.1 Contributions: Multi-Client Functional Encryption with Access Control

We have argued above that one important aspect of the functional keys is how to control them, especially given the fact that in standard FE schemes, any functional key can be used unlimitedly after a legitimate issuance. The work by Abdalla *et al.* [ACGU20] is the first to deal with this problem of controlling decryption keys in FE, for the practical function class to compute inner products. This thesis contains follow-up results of [ACGU20], including in paricular [NPP22a] that gives a definitional framework for treating access control in MCFE and in particular gives concrete MCFE scheme for inner products with Linear Secret Sharing access control. This resolves partially question 3 at the level of MCFE. We briefly summarize our results below and refer to Chapter 4 for more extensive motivations behind our work. Other results that advance further this result and can be found in Section 2.3.3, for a later chapter of this thesis, and in Section 2.3.5, for those that are not presented.

Single-client setting. We propose new single-client schemes whose selectively-secure version is almost as efficient as the selectively-secure version in [ACGU20] and the adaptively-secure version is nearly three times as efficient as the adaptively-secure version in [ACGU20]. More importantly, our schemes can be extended to multi-client settings. Our constructions exploit the *Dual Pairing Vector Spaces* that are proposed by Okamoto-Takashima [OT10, OT12b].

Multi-client setting. Our main contribution is an extension from single-client to multi-client without linearly increasing the complexity in the number n of clients. The generic transformation proposed by Abdalla *et al.* [ACGU20, Theorem 6.3] results in a degradation of factor n in both construction and security reduction. As it turns out from [ACGU20, Sect. 6.2], Abdalla *et al.*'s generic transformation can only help to achieve a multi-input scheme and is unlikely to be generalized to a multi-client scheme without further seriously degrading efficiency. On the other hand, because MIFE can be defined as MCFE with a fixed public constant tag, our construction yields a much more efficient MIFE with access control than the Abdalla *et al.*'s scheme (in fact, n times more efficient). More concretely, the total communication among n clients in our MCFE construction is a linear function in n and does not suffer a quadratic blow-up of n^2 group elements as in [ACGU20].

Comparisons. Our concrete constructions focus on the functionality class whose member's description contains inner-product evaluation functions and binary relations to describe access control. In the pairing-based setting, we give comparisons with existing works in Table 2.1. Recall that in MCFE, n can be a large number of clients, while d is the number of attributes, generally small, used in a policy. Concretely, we can consider identity-based functional encryption, as

Scheme	$\mathcal{P}$	$\mathcal{F}$	$\lvert\mathsf{ct}\rvert$	Security
[ACGU20, Sect. 3.1]	MSP; CP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{MSP}}$	$n+2d+2$	sel-sim
[ACGU20, Sect. 3.2]	roMSP; CP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{roMSP}}$	$3nd+3d+2$	ad-ind
Sect. 4.3, Fig. 4.1	LSSS; KP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{LSSS}}$	$nd+2n+7d+3$	ad-ind
[ACGU20, Sect. 6.2] applied to [ACGU20, Sect. 3.1]	MSP; CP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{MSP}}$	$n^2+2nd+2n$	mi-ad-ind
Sect. 4.4.2	LSSS; KP	$\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{LSSS}}$	$8nd+5n$	mc-ad-ind

Table 2.1: We compare our constructions with existing works, in terms of the number of group elements in the ciphertext (column $\lvert\mathsf{ct}\rvert$), the largest predicate class that can be handled (column $\mathcal{P}$), the function class (column $\mathcal{F}$), security (column **Security**). We denote by d the number of attributes needed by the policy in a ciphertext. All our schemes are defined for the functionality class $\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{n,q,\mathsf{LSSS}} = \mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$ constituted by $\mathcal{F}^{\mathsf{IP}} = \{F_{\mathbf{y}} : \mathbb{Z}_q^n \to \mathbb{Z}_q; \mathbf{x} \mapsto \langle \mathbf{x}, \mathbf{y}\rangle \in \mathcal{R}(\mathbb{Z}_q)\}$ and LSSS of Linear Secret Sharing Schemes over attributes in $\mathbb{Z}_q$, where $n, q \in \mathbb{N}$, q is prime and $|\mathcal{R}(\mathbb{Z}_q)| = \mathsf{poly}(\log q)$. The schemes from [ACGU20] are constructed for $\mathcal{F}^{\mathsf{IP}} \times \mathsf{MSP}$ and $\mathcal{F}^{\mathsf{IP}} \times \mathsf{roMSP}$, where MSP, roMSP are classes of *monotone span programs, read-once monotone span programs* over attributes in $\mathbb{Z}_q$. The shorthands (mc, mi, sel, ad, ind, sim) denote multi-client setting, multi-input setting, selective security, adaptive security, indistinguishability-based, simulation-based.

outlined in [ACGU20], where $d = 1$, whatever the size of n: our ciphertext's size is linear instead of quadratic in n as in [ACGU20].

Related Work. Recently, [LLW21] improves upon the single-client construction based on Learning with Errors (LWE) from [ACGU20], for IPFE with access control expressed by bounded depth boolean circuits, achieving better security along with smaller ciphertexts. In another work, [PD21] also studies LWE-based single client constructions for IPFE with access control expressed by general boolean functions but under selective challenge attributes. The single-client LWE-based construction in [PD21] is later lifted to an MIFE using the generic transformation from [ACGU20].

Also in the single-client setting, another line of works attempts to construct FE for a general uniform functionality class such as Turing machines (TMFE), which naturally captures inner-product evaluation under LSSS access control. The work of Agrawal *et al.* [AMVY21] provided a non-adaptively simulation-based secure construction for TMFE in the *dynamic bounded collusion* model under sub-exponential LWE. The construction is later improved in [AKM+22] to achieve adaptive security under polynomial LWE, DDH or bilinear decisional Diffie-Hellman in specific groups, or quadratic residuosity. Towards this goal, both works of [AMVY21, AKM+22] additionally gave constructions of FE for circuits of *unbounded* size and depth, which can also encompass inner-product computation under LSSS access control, based on various standard assumptions such as computational Diffie-Hellman, factoring, or polynomial LWE. All single-client constructions from [AMVY21, AKM+22] use a wide range of cryptographic primitives in a generic manner, which deviates from our goal to give explicit constructions in the multi-user setting.

2.3.2 Contributions: Optimal Security for (Decentralized) Multi-Client Functional Encryption

Being inspired by the possibility of enhancing the security model of (D)MCFE from [CDG+18a], we reexamine the constraints and corruption model in the context of DMCFE. Concerning the corruption model, it is proposed in [CDG+18a] that when an adversary corrupts a client i, they receive both the encryption key ek_i (for encryption) and the secret key sk_i (for key generation). However, in real life these two types of keys might have different levels of protection, or they are independent and non is included in the other. Concerning the constraint of $x_i^{(0)} = x_i^{(1)}$ for the corrupted client i, it is partially justified by that fact that deterministic encryption is used in [CDG+18a] and is left open by [CDG+18a] to lift this constraint, by necessarily relying on probabilistic encrypting mechanism. We summarize our results below, taken from [NPP23a]: we

give a more relaxed version of admissibility for (D)MCFE, along with a definitional framework that argues why our version is *optimal*, *i.e.* it cannot be relaxed further. This resolves question 1 at the level of (D)MCFE. Other results that advance further this result and can be found in Section 2.3.3, for a later chapter of this thesis.

An Improved Security Model for DMCFE. Since previous security notions of DMCFE turn out unstable, the main goal of our results in [NPP23a] is to propose a fair and optimal security model.

Separating Corruptions of ek_i and sk_i. Our first step is thus to *separate* the corruption of sk_j from that of ek_i, *i.e.* the adversary must specify which type of keys it wants to corrupt. This gives more flexibility to the adversary. However, its goal remains the same: distinguish between the encryption of $\mathbf{x}^{(0)}$ and $\mathbf{x}^{(1)}$ in the challenge ciphertext. We notice that this new corruption model captures the previous "both-or-nothing" model in previous works and any scheme that is secure in this new fine-grained model will also be secure in the old one. A very recent work by Agrawal *et al.* [AGT21b] also defined a security model with similar fine-grained corruption, though as mentioned later (see footnote 1 in Chapter 5) their subsequent DMCFE scheme for inner products has $\mathsf{sk}_i = \mathsf{ek}_i$ for every i and by corrupting one an adversary will obtain both keys.

Refining Admissibity for A Stronger Security. Our next objective consists in challenging the belief from previous admissibility conditions and relaxing the restriction $\mathbf{x}_i^{(0)} = \mathbf{x}_i^{(1)}$ in case of a corrupted i. A more relaxed admissiblity means more attacks will be considered, leading to a stronger security notion. To summarize, we revise the security model for (D)MCFE and

> *1.* We provide a new security model for DMCFE under separated corruption of keys and less restrictive admissibility condition. Our security model covers the security model in all previous works, in the sense that being secure in the former implies being secure in the latter.

In Section 5.2.1, we give the intuition of our new formulation for admissibility condition. This new definition will require probabilistic encryption, which excludes the need of private encryption keys. Our security model will thus also consider public-key encryption, as some security still holds when all the encryption keys ek_i are corrupted. Note however this might make sense for limited classes of functions only and becomes completely meaningless when both $(\mathsf{ek}_i, \mathsf{sk}_i)$ can be corrupted at once.

Optimality. At the core of our new security model is a more relaxed admissibility condition. Up to this point one may well wonder if there is still room to relax our condition, in the same way we have done to the admissibility condition put forth since the birth of DMCFE in [CDG+18a]. Our goal is to analyze this question in a rigorous manner. This turns out to be notoriously hard because we aim to settle this infamous problem with satisfactory justifications whenever a new condition is introduced. Intuitively, since all prior works did not elaborate formally whether an admissibility condition must be respected or it is just optional, we have to start from scratch to formalize how "indispensable" a condition is. We thus address this *optimality* question and this leads to our second contribution:

> *2.* We provide a new framework to prove the optimality of our new notion of *admissible attacks*. More formally, this allows us to show that any non-admissible attack would actually break any efficient construction for the functionality. This proves that we only exclude attacks that are at the functionality level and not at the scheme level.

We believe that the conceptual message from our methodology is one main contribution. We refer to Section 5.2.3 for a detailed explanation of our modeling choices as well as the encountered problems.

Impact and Feasibility. While we have shown our security notion to be optimal w.r.t. the functionality for a class of functions, there are two remaining questions, with respect to this new admissibility notion: are the previous constructions secure? Can we construct concrete schemes for non-trivial functionalities?

First, we can show that the class of inner products is a non-trivial class. Furthermore, it has been widely studied, with several candidates: the DDH-based MCFE for inner products from [CDG+18a, ABG19, CDSG+20] cannot be proven secure in our model, due to the following attack, which was artificially excluded in the previous security models. For any corrupted key ek_i, it was required that $\mathbf{x}_i^{(0)} = \mathbf{x}_i^{(1)}$, because of the deterministic encryption: an adversary corrupts client 1 among n clients to get ek_1, then queries the function $\mathbf{y}$ with $\mathbf{y}_1 = 0$ and challenges $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})_{i\in[n]}$ such that the first coordinates $\mathbf{x}_1^{(0)} \neq \mathbf{x}_1^{(1)}$ and $\langle \mathbf{x}^{(0)}, \mathbf{y}\rangle = \langle \mathbf{x}^{(1)}, \mathbf{y}\rangle$. Then, the adversary encrypts $\mathbf{x}_1^{(1)}$ on their own using ek_1. By comparing with the obtained ciphertext on $\mathbf{x}_1^{(b)}$, such an adversary can decide correctly on b. In addition to these DDH-based constructions, in a work by Libert and Titiu [LT19], the authors proposed the first LWE-based MCFE in the standard model. The ciphertext components of this scheme is somewhat randomized by some small Gaussian error, but the above attack still works by choosing $\mathbf{x}_1^{(0)} \neq \mathbf{x}_1^{(1)}$ that are far from each other, then deciding based on the norm of the two ciphertexts' difference[1]. We note that the above attack gives a byproduct that complements our first contribution

> *1-bis.* Our security model is strictly stronger than the security model in almost all previous works, in the sense that prior concrete schemes cannot be proven secure in ours.

Besides the theoretical part introducing and proving our optimal security notion for DMCFE, we also propose new constructions in the DDH setting which meet the proposed level of security. This requires a number of new technical ideas, in particular a technique for achieving admissibility *via revocation* (in a different way than [ABP+17]) and using *dual pairing vector spaces* (DPVS) [OT10, OT12a, OT12b], to build a probabilistic encryption scheme.

Roughly speaking, our new admissibility when translated for the particular cases of inner-products introduces one condition that for all corrupted clients i, for ek_i, for all functional key query $\mathbf{y}$, it must hold that

$$(\mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)}) \cdot \mathbf{y}_i = 0 \ . \tag{2.1}$$

Previous security models required $\mathbf{x}_i^{(0)} = \mathbf{x}_i^{(1)}$, but we now have to deal with the case $\mathbf{x}_i^{(0)} \neq \mathbf{x}_i^{(1)}$ and $\mathbf{y}_i = 0$ additionally. A necessary condition is that our encryption must be probabilistic (otherwise, the attack described in the previous paragraph applies). However, that is *not* enough because we want semantic security for the ciphertext component ct_i of $\mathbf{x}_i^{(b)}$ as well, where $b \xleftarrow{\$} \{0,1\}$ is the challenge bit. When we view this problem under the lens of revocation systems, similarities emerge: as soon as the special value 0 is set for $\mathbf{y}_i$, we want to nullify the ability for recovering information about $\mathbf{x}_i^{(b)}$. The foregoing fits well in the context of revocation. Conveniently, the work by Agrawal *et al.* [ABP+17] solved the "dual" problem, namely using IPFE to construct revocation systems, and along the way, the authors of [ABP+17] presented a DDH-based IPFE that we can embed locally into the vectors in DPVS, components by components. We leverage this idea to concoct DPVS-based DMCFE schemes for inner-product functionality and achieve security under the condition (2.1). In the end, our third contribution is

> *3.* We demonstrate the feasibility of our new security model by presenting DDH-based DMCFE schemes for inner products over polynomially bounded ranges

[1] We use the metrics employed in the context of the LWE-based (D)MCFE in [LT19].

using pairings, the first concrete scheme whose security holds against fine-grained corruptions and a less restrictive admissiblity.

2.3.3 Contributions: Multi-Client Functional Encryption with Public Inputs and Strong Security

Our results on improving the security model of (D)MCFE in Section 2.3.2 serve as an indication of more natural connections between different FE notions. This section summarizes our results from [NPP25] and we refer to Chapter 6 for more thorough discussion. Roughly speaking, throughout our previous mentionning of (D)MCFE, they are *secret-key* primitives. Our improvement from Section 2.3.2 necessitates that the underlying (D)MCFE scheme has probabilistic encryption, which intriguingly raises the question of relation to *public-key* single client FE *à la* [SW05, BSW11]. Moreover, an observation will point out that our improved condition (2.1) in Section 2.3.2 works in the situation where each client encrypts one scalar. The situation where each client can encrypt a vector, if one wants to connect to single client FE of vector encryption, is still left unresolved. In addition, given the results from Section 2.3.1 on controlling decryption keys, another interesting direction is to study our optimal security from Section 2.3.2 in conjunction with access control, which furthermore will provide connection to *public-key* ABE[2].

A Simple Extension of MCFE to also Cover MIFE, public-key FE and ABE. From the above discussion, one crucial question is:

> *How can we extend MCFE in a minimal way to encompass all the settings, from MIFE to public-key FE and ABE?*

Thus appears the motivation behind proposing MCFE with public inputs, wherein we simply augment the ciphertexts of the MCFE with a public inputs. This part of public inputs is taken care by the function evaluation. However, in order to cover public-key FE, we need to consider the stronger notion of admissibility with the possibility to handle sub-vectors.

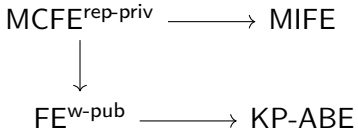
Conceptual Contribution. In essence, we propose a simple extension of MCFE with not only private but also public data to be input to the function evaluation. Combining with the consideration of a stronger admissibility for adversary and sub-vectors in encryption, we cover previous primitives such as MCFE, MIFE, public-key single-input FE and ABE. When only private inputs are considered, if the function involves attributes for access-control, this is necessarily with the attribute-hiding property, while this is not always required. Hence, we will show this is quite relevant for MCFE and MIFE with attribute-based access-control. We also describe, and achieve, a very high security notion that, while considering the multi-client setting with secret-key encryption, also covers public-key attribute-based encryption.

Strong Admissibility and Public-Key Setting. Recently, a stronger and optimal notion of the admissibility of an attack was introduced [NPP23a]. Intuitively, to recall, when there is a unique client, with the initial admissibility from [CDG+18a, ABKW19, ABG19, LT19, CDSG+20, AGT21b], when the encryption key of this single user is corrupted, the only queries that the reduction can forward are the *trivial* one from the FE adversary where $x^{(0)} = x^{(1)}$, hence it is not sufficient to capture the reduction from MCFE to FE with meaningful CPA-security. With strong admissibility from [NPP23a], *i.e.* *without* the requirement that $x_i^{(0)} = x_i^{(1)}$ for corrupted $i \in \mathcal{C}$,

[2]For general function classes it is known that public key single client FE implies public key ABE. Nevertheless, for concrete single client FE with respect to inner products, it is not immediate how to derive public key ABE. Therefore, approaching from single client FE for inner products with access control to derive ABE seems plausible.

we show that the reduction can capture the security of the public-key FE by making public the encryption key. In particular, within the function class to compute inner products, addressing strong admissibility also necessitates the ability to manage sub-vectors in encryption. This is a technically relevant issue because under this stronger admissibility, moving to public-key FE gives the usual functionality of inner products, and not just scalar products.

Our work extends the work from [NPP23a], and we will develop more the conceptual implications in the next paragraph, as well as the concrete case in **Constructive Contributions** below. A discussion on our strong admissiblity is given in paragraph **Discussion on admissibility** after the formal definition in Definition 6.3. The aforementioned implications can be summarized with the following simplified diagram (more details are given in Theorem 6.6), where



- $\mathsf{MCFE}^{\mathsf{rep\text{-}priv}}$ is our new notion of MCFE, with strong admissibility and public inputs, but repetitions are only allowed on the private inputs (multiple encryption queries with the same tag must be with the same public input);
- MIFE is the usual definition, with private inputs only, with repetitions, without tags nor corruptions. The implication comes from the allowed *repetitions on private inputs* in our MCFE;
- $\mathsf{FE}^{\mathsf{w\text{-}pub}}$ is the classical public-key single-input FE definition enhanced with *public inputs*. Implication comes from the *strong-admissibility* that allows to deal with public-key encryption when there is a unique client;
- KP-ABE denotes the usual definition of key-policy ABE. The implication comes from the *public inputs* in $\mathsf{FE}^{\mathsf{w\text{-}pub}}$, that can be used to encode the attributes in a non-hiding way.

It is very interesting that MCFE with the *strong admissibility* from [NPP23a] leads to public-key single-input FE, when there is a unique client, and even to Key-Policy ABE when allowing public inputs (to provide the attributes in the ciphertext).

Constructive Contributions. These implications depend on the actual classes of functions. As a constructive result, we propose an MCFE with the class of functions that combines inner-products (on private inputs) and attribute-based access control (on public inputs) for LSSS policies. It achieves the strong admissibility notion, in the adaptive setting (whereas [ATY23a] only provides selective security), with repetitions on the private inputs and static corruptions. It also deals with sub-vectors (whereas [NPP22a, NPP23a] only consider scalars). As a consequence, removing the tags, the corruptions and the public inputs, we obtain an MIFE for inner products, with strong admissibility and adaptive security; limiting to one client, one gets public-key single-input inner-product FE and KP-ABE for LSSS, with adaptive security. Our construction uses pairings in the ROM, and we note that there exists other approaches to tackle IPFE with access control using lattices, *e.g.* [LLW21, PD21], though they are only single-client to our knowledge.

We would like to emphasize that strong admissibility is not only theoretical (as it allows us to cover public-key single-input inner-product FE) but also more intuitive: the only restriction we impose on the adversary is to prevent them from choosing challenge messages in such a way that, with their corrupted keys and the function evaluation, they cannot trivially win the game by evaluating the function on chosen messages. Requiring the adversary to use the same message for corrupted users as in the previous admissibility now seems somewhat artificial to us. Achieving strong admissibility is also more challenging as it requires the encryption to be

Scheme	$\mathcal{F}$	(priv,pub)-inp	Eval	Security
[ATY23a, Sect. 5]	$\mathcal{F}^{\mathsf{ABP}}_{n,(N_i)_i}$	$((\mathbf{z}_{i,j})_j, (\mathbf{x}_{i,j})_j)$	$\sum_{i\in[n]}\sum_{j\in[N_i]}\langle f_i(\mathbf{x}_{i,j}), \mathbf{z}_{i,j}\rangle$	sel, w-rep, stat, wk-adm
[NPP22a, Sect. 5.4]	$\mathcal{F}^{\mathsf{IP,B}}_{n,q} \times \mathsf{LSSS}$	$(\mathbf{x}[i], \mathsf{S}_i)$	$(\bigwedge_i \mathsf{LSSS}(\mathsf{S}_i)) \cdot \langle \mathbf{x}, \mathbf{y}\rangle$	ad, no-rep, dyn, wk-adm
[NPP23a, Sect. 6]	$\mathcal{F}^{\mathsf{IP,B}}_{n,q}$	$(\mathbf{x}[i], \perp)$	$\langle \mathbf{x}, \mathbf{y}\rangle$	ad, w-rep, stat, s-adm
Corol. 6.13	$\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$	$(\mathbf{x}_i, \mathsf{S}_i)$	$(\bigwedge_i \mathsf{LSSS}(\mathsf{S}_i)) \cdot \sum_{i=1}^{n}\langle \mathbf{x}_i, \mathbf{y}_i\rangle$	ad, rep-priv‡, stat, s-adm

‡ This intermediate notion only provides security against repetitions on private inputs, and not on public inputs.

Table 2.2: We compare our MCFE with existing MCFE, casting the function class into the syntax with *public inputs*. The MCFE in [ATY23a, Sect. 5] is defined for the function class of *attribute-weighted sums* $\mathcal{F}^{\mathsf{ABP}}_{n,(N_i)_i}$ containing $(f_i)_{i\in[n]}$ of *arithmetic branching programs*. that evaluates on public inputs $(\mathbf{x}_{i,j})_j \in (\mathbb{Z}_q^{N_0})^{N_i}$ and private inputs $(\mathbf{z}_{i,j})_j \in (\mathbb{Z}_q^{N_1})^{N_i}$, where N_i is some parameter for slot i, and $N_0, N_1 \in \mathbb{N}$. The MCFE schemes in [NPP22a, Sect. 5.4] are defined for the functionality class $\mathcal{F}^{\mathsf{IP,B}}_{n,q} \times \mathsf{LSSS}$, that evaluates on public attributes S_i and private inputs $\mathbf{x}[i] \in \mathbb{Z}_q$, where $n, q \in \mathbb{N}$, q is prime, $\max_i(|\mathbf{x}[i]|) < B$, and $B = \mathsf{poly}(\lambda) \in \mathbb{N}$ is a polynomial; Meanwhile MCFE from [NPP23a, Sect. 6] is implied by the DMCFE therein and is for $\mathcal{F}^{\mathsf{IP,B}}_{n,q}$ without access control. Sect. 6.4 extends MCFE to the class $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ and $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B}$ that evaluates on public attributes S_i and private inputs $\mathbf{x}_i \in (\mathbb{Z}_q)^{N_i}$ where for all i, $\max(\|\mathbf{x}_i\|_\infty, \|\mathbf{y}_i\|_\infty) < B$, where $B = \mathsf{poly}(\lambda) \in \mathbb{N}$ is a polynomial. We also use the shorthand (no-rep, w-rep) to indicate whether *repetitions* at positions under a challenge tag are allowed, and (sel, ad) to indicate whether *adaptive* challenge ciphertext are allowed. The shorthand (stat, dyn) indicates whether the corruption is *static* or *dynamic*. Finally, (wk-adm, s-adm) indicates whether the *weak* [CDG+18a] or *strong* admissibility [NPP23a] is considered. The preferred properties are underlined.

probabilistic and any deterministic encryption cannot meet strong admissibity as we already explained. Consequently, the only two existing AB-IP-MCFE schemes [NPP22a, ATY23a] are not secure when considering strong admissibility as the encryptions in these schemes are deterministic. Of course, we do not claim to break the schemes [NPP22a, ATY23a], because we consider a stronger security level. We would propose that strong admissibility should be considered in the multi-user setting of FE.

In summary, we propose the first AB-IP-MCFE with strong admissibility and with adaptive security for inner-product functionality while [ATY23a] considers a slightly larger functionality of average weighted sum but with selective security on the challenge messages. In term of efficiency, we have the same asymptotic efficiency as [ATY23a]: each client sends a ciphertext of linear size in the size of its subvector message, independent of the total number of clients. In table 2.3, we compare our construction with existing works.

Relation with Multi-Party Functional Encryption. Our MCFE with Public Input can be seen as a special case of Multi-Party Functional Encryption (MPFE) [AGT21b]. However, our goal is not to define yet another new and more general primitive, but only to add the minimal extension to an existing well-studied primitive to reconcile with other primitives. By simply considering public inputs for MCFE with a stronger admissibility notion, we cover not only attribute-based access control, but also public-key single-input FE. While MPFE is very general, it only considers the secret-key encryption setting and does not cover public-key single-input FE. Up to the notions of MCFE, our results complete the picture of unifying MCFE/MIFE/FE/ABE, by considering the public inputs and the strong admissibility notion. The strong admissibility is necessary (see the paragraph *From Secret-key MCFE to Public-key FE* in Chapter 6), in order to capture the security of public-key FE from the security of MCFE, and is then proven sufficient in our Theorem 6.6. The public inputs are necessary to capture the non-attribute-hiding property of KP-ABE/IBE in the syntax of FE (see the paragraph *Final Syntactical Point: Public Inputs* in Chapter 6), inherits the same spirit of empty-key function in [BSW11], and is cleanly demonstrated in our Theorem 6.6. Finally, our concret final AB-IP-MCFE in Corollary 6.13 is the first to achieve adaptive security for inner-product functionality in the multi-client setting, with public inputs, and with strong admissibility.

2.3.4 Contributions: Function-Hiding DMCFE

To the best of our knowledge, the only candidate of FH-IP-DMCFE comes from [AGT21b], implicitly as a result of their function-hiding FH-IP-DDFE. The implied security of their FH-IP-DMCFE is *selectively* indistinguishability-based in the ROM under *static* corruption, where the adversary makes all encryption, key generation and corruption queries up front in one shot, with repetitions with respect to encryption tags and *no repetitions* with respect to key generation tags. This state-of-the-art leads us to the following question:

> *How far can we raise the security level of pairing-based function-hiding IP-DMCFE in the ROM?*

Our results, taken from [NPS24a], strictly improve on various aspects of security compared with [AGT21b]. We partially resolve question 2 of adaptive security for FH-secure at the level of DMCFE, and partially question 5 of repetitions for FH-security at the level of DMCFE. Other results that advance further this result and can be found in Section 2.3.5, for those that are not presented.

Below and in Table 2.3 are presented a summary of our contributions and a comparison with existing works:

1. *Function-Hiding IP-DMCFE.* We construct the first FH-IP-DMCFE that tolerates *adaptive* encryption queries (with unbounded repetitions) and *adaptive* key generation queries *with a fixed polynomially large number repetitions*, under static corruption. The bounded number of repetitions on key generation queries can be polynomially large and is specified at setup time of the scheme. Our FH-DMCFE thus handles up to an *exponentially large* number of mix-and-match of key repetitions under the same tag tag-f, which is determined by the scheme's parameters. It uses pairings and is provably secure in the ROM. Details about our construction are explained in Section 7.4.

2. *Technical Contribution.* Along the way, we push forward the study of DPVS techniques. We state a novel lemma that shows the indistinguishability of two distributions in a setting where not all input data is known up front. This lemma proves to be the key ingredient for the security proof of our FH-IP-DMCFE scheme in the *adaptive* setting. Due to its oracle-based general formulation, we believe that the lemma can find other applications in the future. The formal statement (Lemma 7.7) and a proof overview can be found in Section 7.4.1. Basic definitions for the DPVS framework are provided in Section 3.3.

Existing Function-Hiding FE Schemes in the Literature. Bishop *et al.* [BJK15] presented the first IPFE scheme that guaranteed a weak variant of the function-hiding property. This construction was lifted to fully function-hiding security by Datta *et al.* [DDM16, DDM17]. This was further improved in terms of efficiency and/or computational hardness assumptions by works of [TAO16, KKS19, KLM+18, Tom19, Tom20]. The constructions of [BJK15, DDM16, TAO16] all leverage the power of *dual pairing vector spaces* (DPVSes) developed by Okamoto and Takashima in [OT10, OT12a, OT12b]. Alternatively, Lin [Lin17] used a different approach to get simpler constructions of FH-IPFE from the ABDP IPFE. Using the same blueprint and exploiting the specific algebraic properties of the underlying inner-product MIFE scheme carefully, Abdalla *et al.* [ACF+18] were able to construct function-hiding MIFE for inner products (FH-IP-MIFE). In [AGT21b], Agrawal *et al.* came up with the first construction of function-hiding MCFE for inner products (FH-IP-MCFE) that is inspired by the FH-IP-MIFE by Datta *et al.* [DOT18]. Very recently, Shi and Vanjani [SV23] presented a generic transformation from single-client to multi-client functional encryption, preserving the function-hiding property and leading to the first FH-IP-MCFE with adaptive security. Remarkably, their security proof does not rely on random oracles. As a follow-up attempt on removing reliance on the ROM,

Scheme	Type	Oracle Queries		Assumptions††	ROM
		Enc	KeyGen		
[AGT21b, Section 6.2]	FH-IP-MCFE	sel, w-rep	sel†	SXDH	✓
[SV23, Section B.3]	FH-IP-MCFE	adap, w-rep	adap†	D-Lin	✗
[AGT21b, Section 6.3]	FH-IP-DMCFE⋆	sel, w-rep	sel, no-rep	SXDH	✓
[Ngu24, Section 4.1]	FH-IP-DMCFE ‡	sel, w-rep	sel, no-rep	SXDH	✗
Corollary 7.9	FH-IP-DMCFE	adap, w-rep	adap, bnd-rep	SXDH	✓

† For MCFE, there is no notion of tags for key generation, hence no notion of repetitions.
‡ This FH-IP-DMCFE is implied by the FH-IP-DDFE of [Ngu24, Section 4.1].
†† All mentioned constructions use pairing groups.
⋆ This FH-IP-DMCFE is implied by the FH-IP-DDFE of [AGT21b, Section 6.3].

Table 2.3: We compare our constructions with existing works, in terms of the type of primitives with function-hiding security (**Type**), whether the encryption oracle (**Enc**) and key generation oracle (**KeyGen**) can be queried adaptively and with repetitions (**Oracle Queries**), which assumptions are used for the security proof (**Assumptions**), and whether the security is proven in the ROM (✓) or not (✗) (**ROM**). The shorthands (sel, adap) denote selective or adaptive oracle queries. The shorthands (w-rep, bnd-rep, no-rep) indicates whether the adversary can demand repetitive queries to the same slot and tag unboundedly, under a fixed bound, or not, in that order. All schemes are defined for the inner-product functionality of their respective type of primitive (see Def. 7.1) and consider only *static* corruption. Preferred properties are underlined.

Nguyen [Ngu24] gives the fisrt FH-IP-DDFE that relies on SXDH while not using ROM, thus extending the results of [SV23] to the more general setting of DDFE for inner products.

In [CDSG+20], Chotard *et al.* generalized DMCFE and defined the notion of *Dynamic Decentralized Functional Encryption* (DDFE) that allows users to join at various stages during the lifetime of a system, while maintaining all decentralized features of DMCFE. Notably, the setup of DDFE is non-interactive and decentralized, while that of DMCFE is *a priori* interactive. In the end, a DDFE scheme allows aggregating data from different sources by decrypting an independent list of ciphertexts using an independent list of functional keys, both of which are fabricated in a completely decentralized manner by users with their sk_i, while requiring no trusted third party. To these extents, DDFE is a primitive strictly stronger than DMCFE, given that the function class of the former contains functions that are well-defined relating to a given list of functional keys and those functions can be expressed by the function class of the latter[3]. In [AGT21b], the authors revisits DDFE for the class of inner products (IP-DDFE) and provide a transformation from FH-IP-MCFE to FH-IP-DDFE, following the approach of Chotard *et al.* [CDSG+20] who presented a similar transformation in the non-function-hiding setting. As a consequence, the FH-IP-DDFE scheme of [AGT21b] entails an FH-IP-DMCFE scheme.

It is worth noting that all known constructions that guarantee function-hiding security rely on pairings. A recent work by Ünal [Üna20] shows that in the manner of most lattice-based approaches, there is little hope to achieve function privacy in IPFE schemes, in the setting of multi-user or not.

2.3.5 Other Contributions

On FE and its extensions. Having at disposal the results on MCFE with fine-grained access control (Section 2.3.1), a provably optimal security notion for the *plain* (D)MCFE (Section 2.3.2), the improvement on the two (Section 2.3.3), as well as the study on function-hiding security (Section 2.3.4) we ask new questions on :

1. *(Dynamic Decentralized Functional Encryption: Generic Constructions with Strong Security)* In a joint work with David Pointcheval and Robert Schädlich, we present new generic

[3]With an appropriate formalization, all function classes in this work, including inner products, satisfy this property.

compilers which, when instantiated with existing schemes from the literature, improve over the state-of-the-art in terms of security, computational assumptions and functionality. Specifically, we obtain the first adaptively secure DDFE schemes for inner products in both the standard and the stronger function-hiding setting which guarantees privacy not only for messages but also for the evaluated functions. Furthermore, we present the first DDFE for inner products whose security can be proven under the LWE assumption in the standard model. Finally, we give the first construction of a DDFE for the attribute-weighted sums functionality with attribute-based access control (with some limitations). All prior constructions guarantee only selective security, rely on group-based assumptions on pairings, and cannot provide access control. Relating to our research questions in Section 2.2, this work give:

(a) *Concrete Instantiations - IP-DDFE.* With respect to the inner-product functionality, we show how to instantiate our compiler with the DMCFE schemes of [CDG+18a, LT19]. In this way, we obtain the first adaptively secure IP-DDFE's, with tag repetition under the SXDH assumption in the ROM, and without tag repetition under the LWE assumption in the standard model. This provides an affirmative answer to questions 2, 4 and 5 in the case of IP-DDFE.

(b) *Concrete Instantiation - FH-IP-DDFE.* In the function-hiding setting of the inner-product functionality, we instantiate our compiler with the FH-DMCFE scheme of [NPS24b]. This gives the first adaptively secure FH-IP-DDFE, with full repetitions on message tags and an a priori polynomially bounded number of repetitions on key tags under the SXDH assumption in the ROM. This provides an affirmative answer to question 2 and partially resolves question 5 in the case of FH-IP-DDFE.

(c) *Concrete Instantiation - AB-AWS-DDFE.* Beyond inner products towards AB-AWS, we instantiate our compilers with a new DMCFE for AWS with access control that is also constructed in this work. We achieve semi-adaptive security (*i.e.* the encryption oracle cannot be called anymore after the first key generation query) under the SXDH and LWE assumptions in the ROM, thereby resolving question 3 and partially question 2 in the case of DDFE for AB-AWS.

This work is currently in submission [NPS25].

2. *(Chosen-Ciphertext Security for MCFE)* In the setting of PKE, it is widely agreed that the *correct and most suitable* security notion for practical use is security against *chosen-ciphertext attacks* (CCA). In the FE regime, the widely known existing work is [BBL17] on single client FE with CCA security. In a joint work with David Pointcheval and Duong Hieu Phan, we embark on studying this question for MCFE, in particular:

   (a) *(Definitions)* We give definitions of the notion for CCA security with respect to MCFE.

   (b) *(Generic Construction)* Given our definitional framework, from

   - an IND-CPA secure MCFE,
   - a perfectly binding and computationally hiding commitment scheme,
   - a secure PRF,
   - a NIZK for a relation that ensures encryption consistency (by messages and by keys) together with commitment opening, satisfying *correctness*, *zero-knowledge*, and *statistical adaptive soundness*,
   - an equivocable-extractable commitment scheme,

   we can obtain an IND-CCA secure MCFE whose security levels are preserved from the underlying IND-CPA MCFE.

(c) *(Concrete Instantiations)* We instantiate our generix transformation for the function class to compute inner products *as per* Section 2.3.3 and Chapter 6. In particular, we achieve efficiency by relying on the famous Groth-Sahai NIZK, and the equivocable-extractable commitment scheme from [ABB+13].

This work is currently in progress [NPP24].

3. *(Traceable Multi-client Functional Encryption)* We consider the problem of *traceability* in MCFE, where functional decryption keys with embedded identities can be traced, given access to a *pirate decoder*. The ongoing project is twofold: we first propose a new security model for MCFE with traceability, and then provide a concrete construction for the class of inner products:

   (a) *(Strong Tracing Definitions)* We define the notion of *traceable* MCFE, whose tracing security is more general than existing works [DPP20, LAKWH22] that apply only for the single client case as well as their naive generalization to the multi-client setting. In a nutshell, our tracing security does *not* impose all functions F that are asked by the adversary to differ on all messages that are asked by the same adversary. This generalizes the notion of [LAKWH22]. Moreover, our notion is *distinguisher*-based, whereas that of [DPP20] is *decoder*-based and inherently needs the constraint where all functions F must differ on all asked messages by the adversary.

   (b) *(Necessity of Stronger Admissibility)* Interestingly, the notion of stronger admissiblity (Section 2.3.2, Chapter 5) plays a central role in identifying the power of our tracing model. We prove two definitional results:

      i. Under the strong admissibility *as per* (Section 2.3.2, Chapter 5), under our *distinguisher*-based tracing security, modulo black-box techniques, there *cannot* exist a secure traitor tracing MCFE with a *public* tracing algorithm that has *only black-box access* to some decoder, *i.e.* a public black-box tracing algorithm, which is resulted from a tracing adversary that queries only functions that do *not* differ on the demanded challenge messages.

      ii. To complement the above result, under the strong admissibility *as per* (Section 2.3.2, Chapter 5), under our *distinguisher*-based tracing security, modulo black-box techniques, there *cannot* exist a secure traitor tracing MCFE with a tracing algorithm that has *only black-box access* to some decoder, traces successfully some function that is asked by the adversary while that function does *not* differ on the demanded challenge messages.

   (c) *(Concrete Instantiations)* We extend the classical embedded-identity private linear broadcast encryption (EIPLBE) introduced in [GKW19] and give constructions from both pairings and lattices assumptions.

   This work is curently in progress [DMN+24].

Other miscellaneous works. In parallel with main works on FE, additional reseach projects on various topics throughout the course of this thesis include:

1. *(Cumulatively All-Lossy-But-One Trapdoor Functions)* Chakraborty, Prabhakaran, and Wichs (PKC'20) recently introduced a new tag-based variant of a theoretical object termed *lossy trapdoor functions*, coined *cumulatively all-lossy-but-one trapdoor functions* (CALBO-TDFs). Informally, CALBO-TDFs allow defining a public tag-based function with a (computationally hidden) special tag, such that the function is lossy for all tags except when the special secret tag is used. In the latter case, the function becomes injective and efficiently invertible using a secret trapdoor. This notion has been used to obtain advanced

constructions of signatures with strong guarantees against leakage and tampering, and also by Dodis, Vaikunthanathan, and Wichs (EUROCRYPT'20) to obtain constructions of randomness extractors with extractor-dependent sources. While these applications are motivated by practical considerations, the only known instantiation of CALBO-TDFs so far relies on the existence of *indistinguishability obfuscation* (iO).

With Benoît Libert and Alain Passelègue, we propose the first two instantiations of CALBO-TDFs based on standard assumptions, circumventing the use of iO by relying on lossy modes and trapdoor mechanisms enabled by these assumptions. This work is published at SCN 2022 - the 13th Conference on Security in Communication Networks [LNP22].

2. *(Pairing-free Blind Signatures from Standard Assumptions in the ROM)* Blind Signatures are a useful primitive for privacy preserving applications such as electronic payments, e-voting, anonymous credentials, and more. However, existing practical blind signature schemes based on standard assumptions require either pairings or lattices. We present the first practical construction of a round-optimal blind signature in the random oracle model based on standard assumptions without resorting to pairings or lattices. In particular, our construction is secure under the strong RSA assumption and DDH (in pairing-free groups). For our construction, we provide a NIZK-friendly signature based on strong RSA, and efficiently instantiate a variant of Fischlin's generic framework (CRYPTO'06). Our Blind Signature scheme has signatures of size 4.28 KB and communication cost 10.98 KB. On the way, we develop techniques that might be of independent interest. In particular, we provide efficient *relaxed* range-proofs for large ranges with subversion zero-knowledge and compact commitments to elements of arbitrary groups. This work is a joint work with Julia Kastner and Michael Reichle, is published at CRYPTO 2024 - the 44th Annual International Cryptology Conference [KNR24].

All the works in this thesis were supported in part by the French ANR Project ANR-19-CE39-0011 PRESTO.

Chapter

3 Preliminaries

Chapter content

3.1 Notations

We write $[n]$ to denote the set $\{1, 2, \ldots, n\}$ for an integer n. For any $q \geq 2$, we let $\mathbb{Z}_q$ denote the ring of integers with addition and multiplication modulo q. For a prime q and an integer N, we denote by $GL_N(\mathbb{Z}_q)$ the general linear group of of degree N over $\mathbb{Z}_q$. We write vectors as row-vectors, unless stated otherwise. For a vector $\mathbf{x}$ of dimension n, the notation $\mathbf{x}[i]$ indicates the i-th coordinate of $\mathbf{x}$, for $i \in [n]$. We will follow the implicit notation in [EHK+13] and use $[\![a]\!]$ to denote g^a in a cyclic group $\mathbb{G}$ of prime order q generated by g, given $a \in \mathbb{Z}_q$. This implicit notation extends to matrices and vectors having entries in $\mathbb{Z}_q$. We use the shorthand ppt for "probabilistic polynomial time", and the symbol $:=$ for assignment/definition when appropriate.

3.2 Hardness Assumptions

We state the assumptions needed for our constructions.

Definition 3.1. *In a cyclic group $\mathbb{G}$ of prime order q, the* **Decisional Diffie-Hellman** (DDH) *problem is to distinguish the distributions*

$$D_0 = \{([\![1]\!], [\![a]\!], [\![b]\!], [\![ab]\!])\} \qquad D_1 = \{([\![1]\!], [\![a]\!], [\![b]\!], [\![c]\!])\}.$$

for $a, b, c \xleftarrow{\$} \mathbb{Z}_q$. The DDH *assumption in $\mathbb{G}$ assumes that no* ppt *adversary can solve the* DDH *problem with non-negligible probability.*

Definition 3.2. *In a cyclic group $\mathbb{G}$ of prime order q, the* **Decisional Separation Diffie-Hellman** (DSDH) *problem is to distinguish the distributions*

$$D_0 = \{(x, y, [\![1]\!], [\![a]\!], [\![b]\!], [\![ab + x]\!])\} \qquad D_1 = \{x, y, ([\![1]\!], [\![a]\!], [\![b]\!], [\![ab + y]\!])\}$$

for any $x, y \in \mathbb{Z}_q$, and $a, b \xleftarrow{\$} \mathbb{Z}_q$. The DSDH *assumption in $\mathbb{G}$ assumes that no* ppt *adversary can solve the* DSDH *problem with non-negligible probability.*

Definition 3.3. *In the bilinear setting* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, \mathbf{e}, q)$*, the* **Symmetric eXternal Diffie-Hellman** (SXDH) *assumption makes the* DDH *assumption in both* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*.*

Definition 3.4. *In the bilinear setting* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, \mathbf{e}, q)$*, the* **Decisional Bilinear Diffie-Hellman** (DBDH) *problem is to distinguish the distributions*

$$D_0 = \{([\![a]\!]_1, [\![b]\!]_1, [\![b]\!]_2, [\![c]\!]_2, [\![abc]\!]_t)\} \qquad D_1 = \{([\![a]\!]_1, [\![b]\!]_1, [\![b]\!]_2, [\![c]\!]_2, [\![r]\!]_t)\}.$$

for $a, b, c, r \xleftarrow{\$} \mathbb{Z}_q$*. The* DBDH *assumption in* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, \mathbf{e}, q)$ *assumes that no* ppt *adversary can decide the* DBDH *problem with non-negligible probability.*

3.3 Dual Pairing Vector Spaces

We use prime-order bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$ are all written additively. Let us fix $N \in \mathbb{N}$ and consider $\mathbb{G}_1^N$ having N copies of $\mathbb{G}_1$. Any $\mathbf{x} = [\![(x_1, \ldots, x_N)]\!]_1 \in \mathbb{G}_1^N$ is identified as the vector $(x_1, \ldots, x_N) \in \mathbb{Z}_q^N$. The $\mathbf{0}$-vector is $\mathbf{0} = [\![(0, \ldots, 0)]\!]_1$. The addition of two vectors, and $\mathbb{Z}_q$-scalar multiplication, in $\mathbb{G}_1^N$ are defined by coordinate-wise addition. Viewing $\mathbb{Z}_q^N$ as a vector space of dimension N over $\mathbb{Z}_q$ with the notions of bases, we can obtain naturally a similar notion of bases for $\mathbb{G}_1^N$. More specifically, any invertible matrix $B \in GL_N(\mathbb{Z}_q)$ identifies a basis $\mathbf{B}$ of $\mathbb{G}_1^N$, whose i-th row $\mathbf{b}_i$ is $[\![B^{(i)}]\!]_1$, where $B^{(i)}$ is the i-th row of B. Naturally we can extend basis changes in $GL_N(\mathbb{Z}_q)$ to changes of bases of $\mathbb{G}_1^N$ by the fact that $\mathbb{G}_1$ is cyclic. Treating $\mathbb{G}_2^N$ similarly, we can furthermore define a product of two vectors $\mathbf{x} = [\![(x_1, \ldots, x_N)]\!]_1 \in \mathbb{G}_1^N, \mathbf{y} = [\![(y_1, \ldots, y_N)]\!]_2 \in \mathbb{G}_2^N$ by $\mathbf{x} \times \mathbf{y} \coloneqq \prod_{i=1}^N \mathbf{e}(\mathbf{x}[i], \mathbf{y}[i]) = [\![\langle (x_1, \ldots, x_N), (y_1, \ldots, y_N) \rangle]\!]_t$. Given a basis $\mathbf{B} = (\mathbf{b}_i)_{i \in [N]}$ of $\mathbb{G}_1^N$, we define $\mathbf{B}^*$ to be a basis of $\mathbb{G}_2^N$ by first defining $B' \coloneqq (B^{-1})^\top$ and the i-th row $\mathbf{b}_i^*$ of $\mathbf{B}^*$ is $[\![B'^{(i)}]\!]_2$. It holds that $B(B')^\top = I_N$ the identity matrix and $\mathbf{b}_i \times \mathbf{b}_j^* = [\![\delta_{i,j}]\!]_t$ for every $i, j \in [N]$, where $\delta_{i,j} = 1$ if and only if $i = j$. We call the pair $(\mathbf{B}, \mathbf{B}^*)$ a *pair of dual orthogonal bases* of $(\mathbb{G}_1^N, \mathbb{G}_2^N)$. If $\mathbf{B}$ is constructed by a random invertible matrix $B \xleftarrow{\$} GL_N(\mathbb{Z}_q)$, we call the resulting $(\mathbf{B}, \mathbf{B}^*)$ a pair of random dual bases. A DPVS is a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, g_t, \mathbf{e}, q, N)$ with dual orthogonal bases.

Basis Changes. In this work, we use extensively *basis changes* over dual orthogonal bases of a DPVS. We again use $\mathbb{G}_1^N$ as a running example. Let $(\mathbf{A}, \mathbf{A}^*)$ be the dual canonical bases of $(\mathbb{G}_1^N, \mathbb{G}_2^N)$. Let $(\mathbf{U} = (\mathbf{u}_i)_i, \mathbf{U}^* = (\mathbf{u}_i^*)_i)$ be a pair of dual bases of $(\mathbb{G}_1^N, \mathbb{G}_2^N)$, corresponding to an invertible matrix $U \in \mathbb{Z}_q^{N \times N}$. Given an invertible matrix $B \in \mathbb{Z}_q^{N \times N}$, the basis change from $\mathbf{U}$ w.r.t B is defined to be $\mathbf{B} \coloneqq B \cdot \mathbf{U}$, which means:

$$\begin{aligned}(x_1, \ldots, x_N)_{\mathbf{B}} &= \sum_{i=1}^N x_i \mathbf{b}_i = (x_1, \ldots, x_N) \cdot \mathbf{B} = (x_1, \ldots, x_N) \cdot B \cdot \mathbf{U} \\ &= (y_1, \ldots, y_N)_{\mathbf{U}} \text{ where } (y_1, \ldots, y_N) \coloneqq (x_1, \ldots, x_N) \cdot B \ .\end{aligned}$$

Under a basis change $\mathbf{B} = B \cdot \mathbf{U}$, we have

$$(x_1, \ldots, x_N)_{\mathbf{B}} = ((x_1, \ldots, x_N) \cdot B)_{\mathbf{U}}\,; \quad (y_1, \ldots, y_N)_{\mathbf{U}} = \left((y_1, \ldots, y_N) \cdot B^{-1}\right)_{\mathbf{B}} \ . \tag{3.1}$$

The computation is extended to the dual basis change $\mathbf{B}^* = B' \cdot \mathbf{U}^*$, where $B' = \left(B^{-1}\right)^\top$:

$$(x_1, \ldots, x_N)_{\mathbf{B}^*} = ((x_1, \ldots, x_N) \cdot B')_{\mathbf{U}^*}\,; \quad (y_1, \ldots, y_N)_{\mathbf{U}^*} = \left((y_1, \ldots, y_N) \cdot B^\top\right)_{\mathbf{B}^*} \ . \tag{3.2}$$

It can be checked that $(\mathbf{B}, \mathbf{B}^*)$ remains a pair of dual orthogonal bases. When we consider a basis change $\mathbf{B} = B \cdot \mathbf{U}$, if $B = (b_{i,j})_{i,j}$ affects only a subset $J \subseteq [N]$ of indices in the representation

w.r.t basis $\mathbf{U}$, we will write B as the square block containing $(b_{i,j})_{i,j}$ for $i,j \in J$ and implicitly the entries of B outside this block are taken from the identity matrix I_N.

The basis changes are particularly useful in our security proofs. Intuitively these changes constitute a transition from a hybrid G having vectors expressed in $(\mathbf{U}, \mathbf{U}^*)$ to the next hybrid $\mathsf{G}_{\mathsf{next}}$ having vectors expressed in $(\mathbf{B}, \mathbf{B}^*)$. We focus on two types of basis changes, which are elaborated below. For simplicity, we consider dimension $N = 2$:

Formal Basis Changes: We change $(\mathbf{U}, \mathbf{U}^*)$ into $(\mathbf{B}, \mathbf{B}^*)$ using

$$B \coloneqq \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}_{1,2} \qquad B' \coloneqq \left(B^{-1}\right)^\top = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}_{1,2}$$
$$\mathbf{B} = B \cdot \mathbf{U} \qquad \mathbf{B}^* = B' \cdot \mathbf{U}^* \ .$$

We use this type in situations such as: in G we have vectors *all* of the form $(x_1, 0)_{\mathbf{U}}, (y_1, 0)_{\mathbf{U}^*}$, and we want to go to $\mathsf{G}_{\mathsf{next}}$ having vectors *all* of the form $(x_1, 0)_{\mathbf{B}}, (y_1, \boxed{y_1})_{\mathbf{B}^*}$. The simulator writes *all* vectors $(x_1, 0)_{\mathbf{U}}, (y_1, 0)_{\mathbf{U}^*}$ in $(\mathbf{U}, \mathbf{U}^*)$ and under this basis change they are written into

$$(x_1,\ 0)_{\mathbf{U}} = (x_1 - 0,\ 0)_{\mathbf{B}} = (x_1,\ 0)_{\mathbf{B}}; \quad (y_1,\ 0)_{\mathbf{U}^*} = (y_1,\ 0 + y_1)_{\mathbf{B}^*} = (y_1,\ y_1)_{\mathbf{B}^*}$$

following the calculations in (3.1) and (3.2). The products between two dual vectors are invariant, *all* vectors are formally written from $(\mathbf{U}, \mathbf{U}^*)$ (corresponding to G) to $(\mathbf{B}, \mathbf{B}^*)$ (corresponding to $\mathsf{G}_{\mathsf{next}}$), the adversary's view over the vectors is thus identical from G to $\mathsf{G}_{\mathsf{next}}$. In particular, this is a kind of *information-theoretic property* of DPVS by basis changing that we exploit to have identical hybrids' hop in the security proof. We list some formal basis changes that are extensively used in this work:

1. *(Duplication)* This is the above example, vectors $\mathbf{b}_2, \mathbf{b}_1^*$ are secret:

$$B \coloneqq \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}_{1,2} \qquad B' \coloneqq \left(B^{-1}\right)^\top = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}_{1,2}$$
$$\mathbf{B} = B \cdot \mathbf{U} \qquad \mathbf{B}^* = B' \cdot \mathbf{U}^* \ .$$

   and $\{(x_1, 0)_{\mathbf{U}}, (y_1, 0)_{\mathbf{U}^*}\} \equiv \left\{(x_1, 0)_{\mathbf{B}}, (y_1, \boxed{y_1})_{\mathbf{B}^*}\right\}$.

2. *(Quotient, by randomness $r \xleftarrow{\$} \mathbb{Z}_q^*$)* The matrices, vector $\mathbf{b}_1$ is secret, are:

$$B \coloneqq \begin{bmatrix} r & 0 \\ 0 & 1 \end{bmatrix}_{1,2} \qquad B' \coloneqq \left(B^{-1}\right)^\top = \begin{bmatrix} 1/r & 0 \\ 0 & 1 \end{bmatrix}_{1,2}$$
$$\mathbf{B} = B \cdot \mathbf{U} \qquad \mathbf{B}^* = B' \cdot \mathbf{U}^* \ .$$

   and $\{(x_1, 0)_{\mathbf{U}}, (y_1, 0)_{\mathbf{U}^*}\} \equiv \left\{(\boxed{x_1 \cdot r}, 0)_{\mathbf{B}}, (\boxed{y_1 \cdot 1/r}, 0)_{\mathbf{B}^*}\right\}$.

3. *(Formal Switch)* this is the same as *(Duplication)*, but the starting coordinates are not 0:

$$B \coloneqq \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}_{1,2} \qquad B' \coloneqq \left(B^{-1}\right)^\top = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}_{1,2}$$
$$\mathbf{B} = B \cdot \mathbf{U} \qquad \mathbf{B}^* = B' \cdot \mathbf{U}^* \ .$$

   and $\{(x_1, x_2)_{\mathbf{U}}, (y_1, y_2)_{\mathbf{U}^*}\} \equiv \left\{(\boxed{x_1 - x_2}, x_2)_{\mathbf{B}}, (y_1, \boxed{y_2 + y_1})_{\mathbf{B}^*}\right\}$.

Computational Basis Change: Given an instance of a computational problem, *e.g.* $[\![(a,b,c)]\!]_1$ of DDH in $\mathbb{G}_1$ where $c-ab=0$ or $\delta \xleftarrow{\$} \mathbb{Z}_q$, we change $(\mathbf{U},\mathbf{U}^*)$ into $(\mathbf{B},\mathbf{B}^*)$ using

$$B \coloneqq \begin{bmatrix} 1 & 0 \\ a & 1 \end{bmatrix}_{1,2} \qquad B' \coloneqq \left(B^{-1}\right)^\top = \begin{bmatrix} 1 & -a \\ 0 & 1 \end{bmatrix}_{1,2}$$
$$\mathbf{B} = B \cdot \mathbf{U} \qquad \mathbf{B}^* = B' \cdot \mathbf{U}^* \ .$$

One situation where this type of basis change can be useful is: in G we have *some* target vectors of the form $(0,\mathsf{rnd})_{\mathbf{U}}$, where $\mathsf{rnd} \xleftarrow{\$} \mathbb{Z}_q$ is a random scalar, together with other $(z_1,z_2)_{\mathbf{U}}$, and *all* the dual is of the form $(0,\ y_2)_{\mathbf{U}^*}$. We want to go to $\mathsf{G}_{\mathsf{next}}$ having $(\boxed{\widetilde{\mathsf{rnd}}},\ \mathsf{rnd})_{\mathbf{B}}$ masked by some randomness $\widetilde{\mathsf{rnd}} \xleftarrow{\$} \mathbb{Z}_q$, while keeping $(0,\ y_2)_{\mathbf{B}^*}$. Because $[\![a]\!]_1$ is given, the simulator can simulate vectors $(z_1,z_2)_{\mathbf{U}}$ directly in $\mathbf{B}$ using $[\![a]\!]_1$ and some known coordinates z_1,z_2. The basis change will be employed for the simulation of target vectors:

$$(c,\ b)_{\mathbf{U}} + (0,\mathsf{rnd})_{\mathbf{B}} = (c - a\cdot b,\ \mathsf{rnd}+b)_{\mathbf{B}};$$
$$(0,\ y_2)_{\mathbf{U}^*} = (0,\ y_2 + a\cdot 0)_{\mathbf{B}^*} = (0,\ y_2)_{\mathbf{B}^*}$$

where *all* vectors in $\mathbf{B}^*$ must be written first in $\mathbf{U}^*$, since we do not have $[\![a]\!]_2$, to see how the basis change affects them. Using the basis change we simulate those target vectors by $(c-a\cdot b,\ \mathsf{rnd}+b)_{\mathbf{B}}$ with rnd implicitly being updated to $\mathsf{rnd}+b$, the uninterested $(z_1,z_2)_{\mathbf{B}}$ are simulated correctly in $\mathbf{B}$, meanwhile the dual vectors $(0,\ y_2)_{\mathbf{B}^*}$ stays the same. Depending on the DDH instance, if $c-ab=0$ the target vectors are in fact $(0,\ \mathsf{rnd})_{\mathbf{B}}$ and we are simulating G, else $c-ab=\delta \xleftarrow{\$} \mathbb{Z}_q$ the target vectors are simulated for $\mathsf{G}_{\mathsf{next}}$ and $\widetilde{\mathsf{rnd}} \coloneqq \delta$. Hence, under the hardness of DDH in $\mathbb{G}_1$, a computationally bounded adversary cannot distinguish its views in the hybrids' hop from G to $\mathsf{G}_{\mathsf{next}}$. Under the SXDH assumption in the DPVS setting, we list some computational basis changes that are extensively used in this work:

1. *(Subspace)* Given the DDH instance $[\![(a,b,c)]\!]$ in the group w.r.t $\mathbf{B}$, this is the above example, the matrices, vectors $\mathbf{b}_2,\mathbf{b}_1^*$ are secret, are:

$$B \coloneqq \begin{bmatrix} 1 & 0 \\ a & 1 \end{bmatrix}_{1,2} \qquad B' \coloneqq \left(B^{-1}\right)^\top = \begin{bmatrix} 1 & -a \\ 0 & 1 \end{bmatrix}_{1,2}$$
$$\mathbf{B} = B \cdot \mathbf{U} \qquad \mathbf{B}^* = B' \cdot \mathbf{U}^* \ .$$

   and $\{(z_1,z_2)_{\mathbf{B}},(0,\mathsf{rnd})_{\mathbf{U}},(0,\ y_2)_{\mathbf{B}^*}\} \approx_c \left\{(z_1,z_2)_{\mathbf{B}},(\boxed{\widetilde{\mathsf{rnd}}},\ \mathsf{rnd})_{\mathbf{B}},(0,\ y_2)_{\mathbf{B}^*}\right\}$.

2. *(Swap)* Given the DDH instance $[\![(a,b,c)]\!]$ in the group w.r.t $\mathbf{B}$, the matrices, vectors $\mathbf{b}_3,\mathbf{b}_1^*,\mathbf{b}_2^*$ are secret, are:

$$B \coloneqq \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -a & a & 1 \end{bmatrix}_{1,2,3} \qquad B' \coloneqq \left(B^{-1}\right)^\top = \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & -a \\ 0 & 0 & 1 \end{bmatrix}_{1,2,3}$$
$$\mathbf{B} = B \cdot \mathbf{U} \qquad \mathbf{B}^* = B' \cdot \mathbf{U}^* \ .$$

   and $\{(z_1,z_2,z_3)_{\mathbf{B}},(x,0,y)_{\mathbf{U}},(r,r,r')_{\mathbf{U}^*}\} \approx_c \left\{(z_1,z_2,z_3)_{\mathbf{B}},(\boxed{0,x},y)_{\mathbf{B}},(r,r,r')_{\mathbf{B}^*}\right\}$.

We remark that the basis changes will modify basis vectors and for the indistinguishability to hold, perfectly in *formal* change and computationally in *computational* changes, all impacted *basis* vectors must not be revealed to the adversary.

Additional Notations. Any $\mathbf{x} = [\![(m_1, \ldots, m_N)]\!]_1 \in \mathbb{G}_1^N$ is identified as $(m_1, \ldots, m_N) \in \mathbb{Z}_q^N$. There is no ambiguity because $\mathbb{G}_1$ is a cyclic group of order q prime. The $\mathbf{0}$-vector is $\mathbf{0} = [\![(0, \ldots, 0)]\!]_1$. The addition of two vectors in $\mathbb{G}_1^N$ is defined by coordinate-wise addition. The scalar multiplication of a vector is defined by $t \cdot \mathbf{x} := [\![t \cdot (m_1, \ldots, m_N)]\!]_1$, where $t \in \mathbb{Z}_q$ and $\mathbf{x} = [\![(m_1, \ldots, m_N)]\!]_1$. The additive inverse of $\mathbf{x} \in \mathbb{G}_1^N$ is defined to be $-\mathbf{x} := [\![(-m_1, \ldots, -m_N)]\!]_1$. The canonical basis $\mathbf{A}$ of $\mathbb{G}_1^N$ consists of $\mathbf{a}_1 := [\![(1, 0 \ldots, 0)]\!]_1, \mathbf{a}_2 := [\![(0, 1, 0 \ldots, 0)]\!]_1, \ldots, \mathbf{a}_N := [\![(0, \ldots, 0, 1)]\!]_1$. By convention the writing $\mathbf{x} = (m_1, \ldots, m_N)$ concerns the canonical basis $\mathbf{A}$.

The Masking Lemma with Repetitions. We state a technical lemma that is employed throughout our proofs. This is a generalized version of [NPP22a, Lemma 4], where the masks can be introduced even when *repetitions* of $\mathbf{c}$-vectors over j and root are allowed. A detailed proof can be found in [NPP25]. How the lemma and the DPVS basis changes technique are used in security proof can be examined in the proof of Theorem 6.11. By starting with getting familiar with the above basis change examples, then going over the proof of Theorem 6.11 (optionally the proof of [NPP25, Lemma 1]), a comprehensive of the DPVS techniques is hopefully conveyed. As another example of how DPVS is employed in our security proofs, though it will in a different context of FH-IP-DMCFE, readers are also encouraged to look at Lemma 7.7 in Section 7.4.1.

Lemma 3.5. *Let $\mathbb{A}$ be an* LSSS*-realizable over a set of attributes $\mathsf{Att} \subseteq \mathbb{Z}_q$. We denote by* $\mathsf{List\text{-}Att}(\mathbb{A})$ *the list of attributes appearing in $\mathbb{A}$ and by P the cardinality of* $\mathsf{List\text{-}Att}(\mathbb{A})$*. Let $\mathsf{S} \subseteq \mathsf{Att}$ be a set of attributes. Let $(\mathbf{H}, \mathbf{H}^*)$ and $(\mathbf{F}, \mathbf{F}^*)$ be two random dual bases of $(\mathbb{G}_1^2, \mathbb{G}_2^2)$ and $(\mathbb{G}_1^8, \mathbb{G}_2^8)$, respectively. The vectors $(\mathbf{h}_1, \mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3)$ are public, while all other vectors are secret. Suppose we have two random labelings $(a_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})} \leftarrow \Lambda_{a_0}(\mathbb{A})$ and $(a'_j)_j \leftarrow \Lambda_{a'_0}(\mathbb{A})$ for $a_0, a'_0 \xleftarrow{\$} \mathbb{Z}_q$. Let J denote the maximum number of* repetitions *at each $j \in \mathsf{S}$ for $\mathbf{c}_j$ or for $\mathbf{c}_{\mathsf{root}}$. Then, under the* SXDH *assumption in $(\mathbb{G}_1, \mathbb{G}_2)$, the following two distributions are computationally indistinguishable:*

$$\left\{ \begin{array}{ll} (x^{(\mathsf{rep})}), y & \\ \mathbf{c}_{j \in \mathsf{S}}^{(\mathsf{rep})} & = (\sigma_j^{(\mathsf{rep})}(1, -j),\ \psi^{(\mathsf{rep})},\ 0^5)_{\mathbf{F}} \\ \mathbf{k}^*_{j \in \mathsf{List\text{-}Att}(\mathbb{A})} & = (\pi_j \cdot (j, 1),\ a_j z,\ 0^5)_{\mathbf{F}^*} \\ \mathbf{c}_{\mathsf{root}}^{(\mathsf{rep})} & = (\psi^{(\mathsf{rep})},\ 0)_{\mathbf{H}} \\ \mathbf{k}^*_{\mathsf{root}} & = (a_0 z,\ 0)_{\mathbf{H}^*} \end{array} \right\} ; \left\{ \begin{array}{ll} (x^{(\mathsf{rep})}), y & \\ \mathbf{c}_{j \in \mathsf{S}}^{(\mathsf{rep})} & = (\sigma_j^{(\mathsf{rep})}(1, -j),\ \psi^{(\mathsf{rep})},\ 0^2,\ \boxed{\tau z_j x^{(\mathsf{rep})}},\ 0^2)_{\mathbf{F}} \\ \mathbf{k}^*_{j \in \mathsf{List\text{-}Att}(\mathbb{A})} & = (\pi_j(j, 1),\ a_j z,\ 0^2,\ \boxed{a'_j y / z_j},\ 0^2)_{\mathbf{F}^*} \\ \mathbf{c}_{\mathsf{root}}^{(\mathsf{rep})} & = (\psi^{(\mathsf{rep})},\ \boxed{\tau x^{(\mathsf{rep})}})_{\mathbf{H}} \\ \mathbf{k}^*_{\mathsf{root}} & = (a_0 z,\ \boxed{a'_0 y})_{\mathbf{H}^*} \end{array} \right\}$$

for any $x^{(\mathsf{rep})}, y \in \mathbb{Z}_q$, where $\mathsf{rep} \in [J]$, and $z_j, \sigma_j, \pi_j, \psi, \tau, z, r'_0 \xleftarrow{\$} \mathbb{Z}_q$.

3.4 Access Structure and Linear Secret Sharing Schemes

We recall below the vocabularies of access structures and linear secret sharing schemes that will be used in this work. Let $\mathsf{Att} = \{\mathsf{att}_1, \mathsf{att}_2, \ldots, \mathsf{att}_m\}$ be a finite universe of attributes. An *access structure* over Att is a family $\mathbb{A} \subseteq 2^{\mathsf{Att}} \setminus \{\varnothing\}$. A set in $\mathbb{A}$ is said to be *authorized*; otherwise it is *unauthorized*. An access structure $\mathbb{A}$ is *monotone* if $\mathsf{S}_1 \subseteq \mathsf{S}_2 \subseteq \mathsf{Att}$ and $\mathsf{S}_1 \in \mathbb{A}$ imply $\mathsf{S}_2 \in \mathbb{A}$. Given a set of attributes $\mathsf{S} \subseteq \mathsf{Att}$, we write $\mathbb{A}(\mathsf{S}) = 1$ if and only if there exists $A \subseteq \mathsf{S}$ such that A is authorized. A secret sharing scheme for an access structure $\mathbb{A}$ over the attributes $\mathsf{Att} = \{\mathsf{att}_1, \mathsf{att}_2, \ldots, \mathsf{att}_m\}$ allows sharing a secret s among the m attributes att_j for $1 \leq j \leq m$, such that: (1) Any authorized set in $\mathbb{A}$ can be used to reconstruct s from the shares of its elements; (2) Given any unauthorized set and its shares, the secret s is statistically identical to a uniform random value. We will use *linear secret sharing schemes* (LSSS), which is recalled below:

Definition 3.6 (LSSS [Bei96])**.** *Let K be a field, $d, f \in \mathbb{N}$, and Att be a finite universe of attributes. A* Linear Secret Sharing Scheme LSSS *over K for an access structure $\mathbb{A}$ over Att is specified by a share-generating matrix $\mathbf{A} \in K^{d \times f}$ such that for any $I \subset [d]$, there exists a vector $\mathbf{c} \in K^d$ with support I and $\mathbf{c} \cdot \mathbf{A} = (1, 0, \ldots, 0)$ if and only if $\{\mathsf{att}_i \mid i \in I\} \in \mathbb{A}$.*

In order to share s using an LSSS over K, one first picks uniformly random values $v_2, v_3, \ldots, v_f \xleftarrow{\$} K$ and the share for an attribute att_i is the i-th coordinate $\mathbf{s}[i]$ of the share vector $\mathbf{s} := (s, v_2, v_3, \ldots, v_f) \cdot \mathbf{A}^\top$. Then, only an authorized set $\{\mathsf{att}_i \mid i \in I\} \in \mathbb{A}$ for some $I \subseteq [d]$ can recover $\mathbf{c}$ to reconstruct s from the shares by: $\mathbf{c} \cdot \mathbf{s}^\top = \mathbf{c} \cdot (\mathbf{A} \cdot (s, v_2, v_3, \ldots, v_f)^\top) = s$. Some canonical examples of LSSS include Shamir's secret sharing scheme for any f-out-of-d threshold gate [Sha79] or Benaloh and Leichter's scheme for any monotone formula [BL90]. An access structure $\mathbb{A}$ is said to be LSSS-*realizable* if there exists a linear secret sharing scheme implementing $\mathbb{A}$.

Let $y \in \mathbb{Z}_q$ where q is prime and for the sake of simplicity, let $\mathsf{Att} \subset \mathbb{Z}_q$ be a set of attributes. Let $\mathbb{A}$ be a monotone access structure over Att realizable by an LSSS over $\mathbb{Z}_q$. A *random labeling* procedure $\Lambda_y(\mathbb{A})$ is a secret sharing of y using LSSS:

$$\Lambda_y(\mathbb{A}) := (y, v_2, v_3, \ldots, v_f) \cdot \mathbf{A}^\top \in \mathbb{Z}_q^d \tag{3.3}$$

where $\mathbf{A} \in \mathbb{Z}_q^{d \times f}$ is the share-generating matrix and $v_2, v_3, \ldots, v_f \xleftarrow{\$} \mathbb{Z}_q$.

3.5 Cryptographic Primitives

3.5.1 Key-policy Attribute-Based Encryption (KP-ABE)

A *key-policy attribute-based encryption* scheme is defined by a tuple of algorithms (Setup, KeyGen, Enc, Dec). The Setup algorithm takes as input a security parameter 1^λ and outputs a public key pk and a master secret key msk. The KeyGen algorithm takes as input a master secret key msk, a policy $\mathbb{A}$, and outputs a secret key $\mathsf{sk}_\mathbb{A}$. The Enc algorithm takes as input a public key pk, a message m in some message space $\mathcal{M}$, and a set of attributes S, and outputs a ciphertext ct_S. The Dec algorithm takes as input a secret key $\mathsf{sk}_\mathbb{A}$ and a ciphertext ct_S, and outputs a message m. A KP-ABE is *correct* if for all $\lambda \in \mathbb{N}$, all $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$, all $\mathbb{A} \in \mathsf{Pol}$, all $\mathsf{S} \subseteq \mathsf{Att}$, all $m \in \mathcal{M}$, and all $\mathsf{sk}_\mathbb{A} \leftarrow \mathsf{Keygen}(\mathsf{msk}, \mathbb{A})$, if Pol accepts S, it holds that $\mathsf{Dec}(\mathsf{sk}_\mathbb{A}, \mathsf{Enc}(\mathsf{pk}, m, \mathsf{S})) = m$.

The *security* of a KP-ABE is defined below.

Definition 3.7. *A KP-ABE scheme $\mathcal{E}$ with respect to a class of policies* Pol *having attribute space* Att *is* CPA-secure *if for every* ppt *adversary $\mathcal{A}$, the following probability is negligible in λ:*

$$\mathsf{Adv}^{\mathsf{kpabe}}_{\mathsf{Pol},\mathsf{Att},\mathcal{A}}(1^\lambda) := \left| \Pr[\mathsf{Expr}^{\mathsf{kpabe}}_{\mathsf{Pol},\mathsf{Att},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2} \right|$$

where the experiment $\mathsf{Expr}^{\mathsf{kpabe}}_{\mathsf{Pol},\mathsf{Att},\mathcal{A}}(1^\lambda)$ is defined as follows:

1. *The challenger runs $\mathsf{Setup}(1^\lambda)$ to obtain $(\mathsf{pk}, \mathsf{msk})$ and outputs pk to $\mathcal{A}$. In the following the adversary $\mathcal{A}$ can make queries adaptively in any order before Finalize.*

2. *(Key queries) The adversary $\mathcal{A}$ adaptively outputs a policy $\mathbb{A}$. The challenger runs $\mathsf{sk}_\mathbb{A} \leftarrow \mathsf{Keygen}(\mathsf{msk}, \mathbb{A})$ and returns $\mathsf{sk}_\mathbb{A}$ to $\mathcal{A}$.*

3. *(Challenge) The adversary $\mathcal{A}$ outputs a pair of messages (m_0, m_1) and a set of attributes S^*. The challenger chooses a bit $b \in \{0, 1\}$ and runs $\mathsf{ct}_{\mathsf{S}^*} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b, \mathsf{S}^*)$.*

4. *(Finalize) The adversary $\mathcal{A}$ outputs a guess $\hat{b}$. If there exists a policy $\mathbb{A}$ such that S^* satisfies $\mathbb{A}$, then the expriment outputs $\hat{b} \xleftarrow{\$} \{0, 1\}$. Otherwise, the experiment outputs $\hat{b} \stackrel{?}{=} b$.*

We can define similar weaker notions of *selective* challenge message and/or *selective* challenges attributes.

3.5.2 Functional Encryption (FE)

Below is a recall of the syntax and security of (public key) single client FE.

Definition 3.8. *A* functional encryption *scheme for a class $\mathcal{F}$ is defined by a tuple of algorithms* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$. *The* Setup *algorithm takes as input a security parameter 1^λ and outputs a public key* pk *and a master secret key* msk. *The* Extract *algorithm takes as input a master secret key* msk *and a function description $F_\lambda : \mathcal{M}_\lambda \to \mathcal{R}_\lambda$, and outputs a secret key sk_F. The* Enc *algorithm takes as input a public key* pk*, a message m in some message space $\mathcal{M}$, outputs a ciphertext* ct. *The* Dec *algorithm takes as input a secret key sk_F and a ciphertext* ct*, and outputs an element in $\mathcal{R}$. An FE for a class $\mathcal{F}$ is* correct *if for all $\lambda \in \mathbb{N}$, all $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$, all $F_\lambda \in \mathcal{F}$, all $m \in \mathcal{M}$, and all $\mathsf{sk}_F \leftarrow \mathsf{Keygen}(F_\lambda, \mathsf{msk})$, it holds that $\mathsf{Dec}(\mathsf{sk}_F, \mathsf{Enc}(\mathsf{pk}, m)) = F_\lambda(m)$.*

The *security* of an FE scheme is defined below.

Definition 3.9. *A FE scheme $\mathcal{E}$ with respect to a class of functions $\mathcal{F}$ is* CPA-secure *if for every* ppt *adversary $\mathcal{A}$, the following probability is negligible in λ:*

$$\mathsf{Adv}^{\mathsf{fe}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) := \left| \Pr[\mathsf{Expr}^{\mathsf{fe}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2} \right|$$

where the experiment $\mathsf{Expr}^{\mathsf{fe}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is defined as follows:

1. *The challenger runs $\mathsf{Setup}(1^\lambda)$ to obtain $(\mathsf{pk}, \mathsf{msk})$ and outputs pk to $\mathcal{A}$. In the following the adversary $\mathcal{A}$ can make queries adaptively in any order before Finalize.*
2. *(Key queries) The adversary $\mathcal{A}$ adaptively outputs a function description F_λ. The challenger runs $\mathsf{sk}_F \leftarrow \mathsf{Extract}(F_\lambda, \mathsf{msk})$ and returns sk_F to $\mathcal{A}$.*
3. *(Challenge) The adversary $\mathcal{A}$ outputs a pair of messages (m_0, m_1). The challenger chooses a bit $b \in \{0,1\}$ and runs $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$.*
4. *(Finalize) The adversary $\mathcal{A}$ outputs a guess $\hat{b}$. If there exists a function description F_λ such that $F(m_0) \neq F(m_1)$, then the expriment outputs $\hat{b} \xleftarrow{\$} \{0,1\}$. Otherwise, the experiment outputs $\hat{b} \stackrel{?}{=} b$.*

We can define similar weaker notions of *selective* challenge message and/or *selective* functional decryption key queries. The notion of FE with *access control* can be captured by considering the class $\mathcal{F}$ that does not only include the calulating function F_λ, but also the access control policies $\mathbb{A}$ given any member $(F_\lambda, \mathbb{A})$ in $\mathcal{F}$ (see Section 4.4.1 for a formal treatment in the case of MCFE). The *correctness* is adapted that the decryption key $\mathsf{sk}_{F,\mathbb{A}}$ can only decrypt the ciphertexts ct to $F(m)$ if the access control policy $\mathbb{A}$ accepts the attributes S of the ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m, \mathsf{S})$. The notion of *security* is defined similarly as Definition 3.9, except that the syntax is adapted to the FE with access control.

3.5.3 Multi-Input Functional Encryption (MIFE)

We recall in the following the syntax and security of multi-input functional encryption, following [GGG+14].

Definition 3.10. *A* multi-input functional encryption *scheme is defined by a tuple of algorithms* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$. *The* Setup *algorithm takes as input a security parameter 1^λ and a number of slots n, and outputs a public parameter* pp*, a master secret key* msk*, and n encryption keys ek_i. The* Extract *algorithm takes as input a function description $F_\lambda : \prod_{i=1}^n \mathcal{D}_{\lambda,i} \to \mathcal{R}_\lambda$ and the master secret key* msk*, and outputs a decryption key dk_F. The* Enc *algorithm takes as input an encryption key ek_i and a message m_i in some message space $\mathcal{D}_{\lambda,i}$, and outputs a ciphertext ct_i.*

The Dec *algorithm takes as input a decryption key* dk_F *and a vector of ciphertexts* ct_i *of length* n*, and outputs an element in* $\mathcal{R}_\lambda$ *or* $\perp$*. An MIFE for a class* $\mathcal{F}$ *is* correct *if for all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{pp}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$*, all* $F_\lambda \in \mathcal{F}$*, all* $m_i \in \mathcal{D}_{\lambda,i}$*, and all* $\mathsf{dk}_F \leftarrow \mathsf{Extract}(F_\lambda, \mathsf{msk})$*, it holds that* $\mathsf{Dec}(\mathsf{dk}_{F_\lambda}, (\mathsf{Enc}(\mathsf{ek}_i, m_i))_{i\in[n]}) = F_\lambda(m_i)_{i\in[n]}$.

The *security* of an MIFE is defined below.

Definition 3.11. *An MIFE scheme* $\mathcal{E}$ *with respect to a class of functions* $\mathcal{F}$ *is* secure *if for every* ppt *adversary* $\mathcal{A}$*, the following probability is negligible in* λ*:*

$$\mathsf{Adv}^{\mathsf{mife}}_{\mathcal{F},\mathcal{A}}(1^\lambda) := \left|\Pr[\mathsf{Expr}^{\mathsf{mife}}_{\mathcal{F},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right|$$

where the experiment $\mathsf{Expr}^{\mathsf{mife}}_{\mathcal{F},\mathcal{A}}(1^\lambda)$ *is defined as follows:*

1. *The challenger runs* $\mathsf{Setup}(1^\lambda, 1^n)$ *to obtain* $(\mathsf{pp}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]})$ *and outputs* pp *to* $\mathcal{A}$*. In the following the adversary* $\mathcal{A}$ *can make queries adaptively in any order before Finalize.*

2. *(Corruption) In the works of [ACGU20, AGT22], the adversary against the MIFE is futhermore allowed to corrupt* ek_i *for some* $i \in [n]$*. This notion of security for MIFE with corruption allows one more oracle for the adversary to corrupt* ek_i *for any slot* $i \in [n]$ *of their choices.*

3. *(Key queries) The adversary* $\mathcal{A}$ *adaptively outputs a function description* F_λ*. The challenger runs* $\mathsf{dk}_F \leftarrow \mathsf{Extract}(F_\lambda, \mathsf{msk})$ *and returns* dk_F *to* $\mathcal{A}$*.*

4. *(Challenge) The adversary* $\mathcal{A}$ *outputs a query* $(i, m_i^{(0)}, m_i^{(1)})$ *for some* $i \in [n]$*. The challenger chooses a bit* $b \in \{0,1\}$ *and encrypts* $m_i^{(b)}$ *to obtain* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{ek}_i, m_i^{(b)})$*. The ciphertext* ct_i *is returned to* $\mathcal{A}$*.*

5. *(Encryption) The adversary* $\mathcal{A}$ *outputs a query* (i, m_i) *for some* $i \in [n]$*. The challenger encrypts* m_i *to obtain* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{ek}_i, m_i)$*. The ciphertext* ct_i *is returned to* $\mathcal{A}$*.*

6. *(Finalize) The adversary* $\mathcal{A}$ *outputs a guess* $\hat{b}$*. If the following conditions is satisfied, the experiment outputs* $\hat{b} \stackrel{?}{=} b$*. Otherwise, the experiment outputs* $\hat{b} \stackrel{\$}{\leftarrow} \{0,1\}$*. Let* $I \subset [n]$ *be the set of corrupted indices, for* $b \in \{0,1\}$ *we define* $\mathbf{X}^{(b)} := \{x_{1,j}^{(b)}, \ldots, x_{n,j}^{(b)}\}_{j=1}^q$ *to be the* q *queried challenges*

   (a) *The pair* $\mathbf{X}^{(0)}, \mathbf{X}^{(1)}$ *satisfies that for all* F *queried by* $\mathcal{A}$*, all* $I' = \{i_1, \ldots, i_t\} \subseteq I$*, all* $\{x'_{i_1}, \ldots, x'_{i_t}\}$*, all* $j_1, \ldots, j_{n-t} \in [q]$ *we have*

$$F\left(\mathsf{order}\left(x^{(0)}_{i_1,j_1}, \ldots, x^{(0)}_{i_{n-t},j_{n-t}}, x'_{i_1}, \ldots, x'_{i_t}\right)\right) = F\left(\mathsf{order}\left(x^{(1)}_{i_1,j_1}, \ldots, x^{(1)}_{i_{n-t},j_{n-t}}, x'_{i_1}, \ldots, x'_{i_t}\right)\right)$$

   (b) *The set* $\{F\}$ *queried by* $\mathcal{A}$ *satisfies that for all* $\mathbf{X}^{(0)}, \mathbf{X}^{(1)}$ *challenges, all* $I' = \{i_1, \ldots, i_t\} \subseteq I$*, all* $\{x'_{i_1}, \ldots, x'_{i_t}\}$*, all* $j_1, \ldots, j_{n-t} \in [q]$ *we have*

$$F\left(\mathsf{order}\left(x^{(0)}_{i_1,j_1}, \ldots, x^{(0)}_{i_{n-t},j_{n-t}}, x'_{i_1}, \ldots, x'_{i_t}\right)\right) = F\left(\mathsf{order}\left(x^{(1)}_{i_1,j_1}, \ldots, x^{(1)}_{i_{n-t},j_{n-t}}, x'_{i_1}, \ldots, x'_{i_t}\right)\right)$$

   such that the ℓ*-input receives its correspond value by the permutation* $\mathsf{order}(\cdot)$*. Intuitively the set of inputs* $\{x'_{i_1}, \ldots, x'_{i_t}\}$ *represents whatever the adversary can put into the (subsets of) corrupted slots, and syntactically we use the permutation* $\mathsf{order}(\cdot)$ *to map values to their correct ordered arguments of the function (*e.g. *input value* x'_{i_1} *to argument* k *if* $i_1 = k \in \mathbb{N}$*).*

We can define similar weaker notions of *selective* challenge message and/or *selective* functional decryption key queries. The notion of MIFE with *access control* can be done in the same manner as we do for FE with access control in the previous paragraph. The *correctness* is adapted that the decryption key $\mathsf{sk}_{F,\mathbb{A}}$ can only decrypt the ciphertexts $(\mathsf{ct}_i)_i$ to $F((m_i)_i)$ if the access control policy $\mathbb{A}$ accepts the attributes S_i of the ciphertext $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, m_i, \mathsf{S}_i)$ for all slots $i \in [n]$.

Part II

Security Models of Multi-Client Functional Encryption: Access Control and Stronger Admissibility

Chapter

4 Multi-Client Functional Encryption with Fine-Grained Access Control

Chapter content

Multi-Client Functional Encryption (MCFE) and Multi-Input Functional Encryption (MIFE) are very interesting extensions of Functional Encryption for practical purpose. They allow to compute joint functions over data from multiple parties. Both primitives are aimed at applications in multi-user settings where decryption can be correctly output for users with appropriate functional decryption keys only.

While the definitions for a single user or multiple users were quite general and can be realized for general classes of functions as expressive as Turing machines or all circuits, efficient schemes have been proposed so far for concrete classes of functions: either only for access control, *i.e.* the identity function under some conditions, or linear/quadratic functions under no condition.

In this chapter, we target classes of functions that explicitly combine some evaluation functions independent of the decrypting user under the condition of some access control. More precisely, we introduce a framework for MCFE with fine-grained access control and propose constructions for both single-client and multi-client settings, for inner-product evaluation and access control via Linear Secret Sharing Schemes (LSSS), with selective and adaptive security. Our starting point was a work by Abdalla *et al.* (Asiacrypt '20) that combines functional encryption in multi-user setting with access control, which relies on a generic transformation from the single-client schemes to obtain MIFE schemes. The scheme by Abdalla *et al.* suffers a quadratic factor of n (where n denotes the number of clients) in the ciphertext size. We follow a different path, via MCFE: we present a *duplicate-and-compress* technique to transform the single-client scheme and obtain a MCFE with fine-grained access control scheme with only a linear factor of n in the ciphertext size. Our final scheme thus outperforms the Abdalla *et al.*'s scheme by a factor n, as one can obtain MIFE from MCFE by making all the labels in MCFE a fixed public constant. The concrete

constructions are secure under the SXDH assumption, in the random oracle model for the MCFE scheme, but in the standard model for the MIFE improvement.

The results on MCFE for inner products with LSSS-based access control also serve as a base for further improvements in our later Chapter 6: though having more compact ciphertexts and full adaptive security comparing to Abdalla *et al.*'s scheme, our final MCFE scheme with access control (accordingly the implied MIFE scheme) allows only encrypting scalar messages and only one-time use of messages/attributes per client per tag, *à la* Chotard *et al.* (Asiacrypt '18) that gives a widely used model of security for MCFE. We refer to Chapter 6 about how to lift some of these restrictions.

4.1 Introduction

One classical issue with encryption is the decryption key, even if legitimately obtained: once delivered, it can be used forever. One may expect revocation, or access control with more fine-grained authentication. This has been extensively studied with broadcast encryption, revocation systems and more generally, with attribute-based encryption (ABE) [Wee21]. Finally, as already explained, FE is a generalization of IBE and ABE, and after having been illustrated with IBE and ABE, linear evaluations [ALS16, ABDP16, BBL17, CLT18] and quadratic evaluations [BCFG17, Gay20, AS17, Lin17] have been proposed. However, there are still very few works that combine function evaluation and access control with concrete schemes. This could provide FE, with concrete function evaluation for some target users, or revocation (of users or functions). Abdalla *et al.* [ACGU20] have been the first to address this problem, for enhancing FE and MIFE with access control. In addition, they informally argue that from an ABE for MIFE one can lift it for free to get MCFE, thus solving both problems at the same time. Precisely, they mentioned "*by resorting for instance, to the notion of multi-client IPFE, where ciphertexts are associated with time-stamps, and only ciphertext with matching time-stamps can be combined (e.g. [CDG+18a]) we believe that our proposed primitive provides a more general and versatile solution to the problem*". Their idea can be interpreted as: tags can be used as specific attributes, and tags can be embedded in policies to automatically obtain multi-client settings. This argument seems formally valid when considering the general form of MIFE and MCFE. However, when considering concrete classes of functions, which is our main focus in this chapter, it is unlikely to be efficiently feasible and we will explain the reason in the technical overview in Section 4.2. We underline that the principal difference between MCFE and MIFE is the presence of tags for producing the ciphertext components, which can be jointly decrypted only if all tags are equal. Thus, we can retrieve an MIFE from MCFE by fixing and publishing one tag, which retains the *same* ciphertext's size from the MCFE scheme to the new MIFE one. Moreover, since the combination of ciphertext components in MCFE is restrained by the tags, its security model is far less restrictive than the security model of MIFE that has to deal with arbitrary combination of ciphertext components. For these reasons, our main objective becomes constructing an MCFE having smaller ciphertext size while permitting access control over decryption keys.

We take a completely different approach than in [ACGU20] to answer this question. Borrowing the terminology from ABE, our work will focus on *key-policy* (KP) constructions, where the policy is defined at the moment of key extraction and a ciphertext associated with certain attributes can be decrypted only if those attributes satisfy the policy. The dual notion of *ciphertext-policy* (CP) constructions is already studied in [ACGU20]. We concentrate solely on particular functionality classes whose description contains two separate parts: a description of functions exclusively for evaluation and a binary relation exclusively for modeling access control. Although this conceptual point of view does *not* take us out of the FE realm and thus can be captured by the general FE notion, it suits perfectly our purpose to compute inner-products along with fine-grained access control provided by *Linear Secret Sharing Schemes* (LSSS) in this work. Then, we start from single-client IPFE schemes with LSSS access control and leverage them to get an MCFE scheme,

where only tags are needed for hashing during encryption, and the hash function is modeled as a random oracle. Removing labels by fixing a public tag for all ciphertexts leads to an MIFE scheme in the standard model that is more efficient than the one from [ACGU20].

Chapter Outline. We start by a technical overview of the definitional framework for access control in FE in Section 4.2; Then, we present our single-client IPFE scheme with LSSS access control in Section 4.3; We extend this scheme to a multi-client setting in Section 4.4.2. It is worth emphasizing that the definitional choices we make is important for a solid modelisation of access control in FE, while taking into account previous informal attempts in previous works such as [ACGU20]. Moreover, in a later significant generalization, this framework turns out to be a case of what we coin *function classes with public inputs* in Chapter 6. In Section 4.4.2 for the MCFE construction, we also elaborate on intermediate steps and on connections to MIFE. All abridged proofs can be found in the full version [NPP22b] of [NPP22a].

4.2 Technical Overview

4.2.1 Formalizing Access Control in Functional Encryption

First of all, we discuss how we formalize access control in the notion of functional encryption, which will affect our formal definitions in both single-client setting and in particular, multi-client setting (Definition 4.2). On the one hand, accompanying an encryption scheme with access control over decryption keys is already expressed by ABE, which in itself is a special case of FE. Thus, FE schemes with fine-grained access control can be described by the general FE notion for any class of functions that can handle the desired access control along with the required computation.

On the other hand, when working with concrete functionality, we usually find ourselves in the context where the evaluation *cannot* express the access control and they cannot be described abstractly using a single functionality. Therefore, in this chapter we consider FE with access control as FE schemes for *particular* functionality class whose description can be separated into two parts $\mathcal{F} \times \mathsf{AC\text{-}K}$: (1) a first part $F \in \mathcal{F}$ for evaluation, (2) and a second part for access control captured by a binary relation $\mathsf{Rel} : \mathsf{AC\text{-}K} \times \mathsf{AC\text{-}Ct} \to \{0,1\}$, for some sets $\mathsf{AC\text{-}K}, \mathsf{AC\text{-}Ct}$. The key extraction is done with respect to $(\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}, F)$, meanwhile the encryption procedure will receive $(\mathsf{ac\text{-}ct} \in \mathsf{AC\text{-}Ct}, x)$. A key $\mathsf{sk}_{\mathsf{ac\text{-}k},F}$ can decrypt a ciphertext $\mathsf{ct}_{\mathsf{ac\text{-}ct}}(x)$ to $F(x)$ if and only if $\mathsf{Rel}(\mathsf{ac\text{-}k}, \mathsf{ac\text{-}ct}) = 1$. We stress that this way of formulation does not take us out of the FE regime, as it is still captured by the general FE notion.

We show how the above fomalization is used in a concrete case. In the following discussion we will distinguish the $\boxed{\text{input}}$ during encryption from the $\ulcorner\text{parameters}\urcorner$ during key extraction. The simplest non-trivial example for access control is identity-based control, *i.e.* $\mathsf{AC\text{-}K} = \mathsf{AC\text{-}Ct} = \mathsf{ID}$ for some identity space ID and $\mathsf{Rel}_{\mathsf{ibe}}(\mathsf{id\text{-}k}, \mathsf{id\text{-}ct}) = \left(\mathsf{id\text{-}k} \stackrel{?}{=} \mathsf{id\text{-}ct}\right)$. In this chapter we focus on $F \in \mathcal{F}^{\mathsf{IP}} = \{F_{\mathbf{y}} : \mathbb{Z}_q^n \to \mathbb{Z}_q\}$ for computing inner products over $\mathbb{Z}_q^n$ for some prime q and $n \in \mathbb{N}$, where $F_{\mathbf{y}}(\mathbf{x}) := \langle \mathbf{x}, \mathbf{y} \rangle$[1]. The functional keys are extracted using $\ulcorner(\mathsf{id\text{-}k}, \mathbf{y})\urcorner$ and the ciphertexts are encrypted using $\boxed{(\mathsf{id\text{-}ct}, \mathbf{x})}$. First of all, it is *not* immediate how $\mathcal{F}^{\mathsf{IP}}$ can be used to implement the check $\tau z \cdot (\ulcorner \mathsf{id\text{-}k} \urcorner - \boxed{\mathsf{id\text{-}ct}})$ for the identity-based control, where τ and z are random values generated for encryption and key extraction, respectively, together acting as a mask of the decryption value. Notably, the value z *cannot* be specified as part of the inner-product evaluation function, because the inner-product evaluation itself must be independent of users at the time of generating functional keys, *nor* as part of the ciphertext. It thus seems indispensable to treat the functionality as $\mathcal{F}^{\mathsf{IP}} \times \mathsf{ID}$: the functional key is generated w.r.t $F_{\ulcorner \mathbf{y} \urcorner} \in \mathcal{F}^{\mathsf{IP}}$ and $\ulcorner \mathsf{id\text{-}k} \urcorner \in \mathsf{ID}$, while the ciphertext is encrypted w.r.t $\boxed{(\mathsf{id\text{-}ct} \in \mathsf{ID}, \mathbf{x} \in \mathbb{Z}_q^n)}$. During decryption for obtaining $\langle \boxed{\mathbf{x}}, \ulcorner \mathbf{y} \urcorner \rangle + \tau z \cdot (\ulcorner \mathsf{id\text{-}k} \urcorner - \boxed{\mathsf{id\text{-}ct}})$, the ID-part of the functional key will implement the control $\tau z \cdot (\ulcorner \mathsf{id\text{-}k} \urcorner - \boxed{\mathsf{id\text{-}ct}})$ whilst the $\mathcal{F}^{\mathsf{IP}}$-part will compute $\langle \boxed{\mathbf{x}}, \ulcorner \mathbf{y} \urcorner \rangle$.

Treatment of Tags in MCFE with Access Control. As mentioned in the introduction, our current objective is constructing MCFE schemes with access control having smaller ciphertexts. We use the functionality $\mathcal{F}^{\mathsf{IP}} \times \mathsf{ID}$ as a running example. The input $\boxed{\mathbf{x}}$ for inner-product calculation is broken down into n components for the entries x_i of $\boxed{\mathbf{x}}$. The encryption procedure takes $\boxed{(x_i, \mathsf{id\text{-}ct}_i, \mathsf{tag}_i)}$ and outputs a ciphertext component ct_i, for some identity $\boxed{\mathsf{id\text{-}ct}_i}$ and a tag $\boxed{\mathsf{tag}_i}$. The decryption procedure receives a functional key, which is derived from $F_{\ulcorner \mathbf{y} \urcorner} \in \mathcal{F}^{\mathsf{IP}}$ and $\ulcorner \mathsf{id\text{-}k} \urcorner \in \mathsf{ID}$, and the n ciphertext components $(\mathsf{ct}_i)_{i=1}^n$. The decrypted result is $\langle \boxed{\mathbf{x}}, \ulcorner \mathbf{y} \urcorner \rangle$ if $\boxed{\mathsf{id\text{-}ct}_i} = \ulcorner \mathsf{id\text{-}k} \urcorner$ for all i and $\boxed{\mathsf{tag}_i} = \boxed{\mathsf{tag}_j}$ for all i, j. In the setting that the identities and tags can

[1]This chapter deals with inner products where each client encrypts a scalar x_i of one vector $\mathbf{x}$ to decrypt to $\langle \mathbf{x}, \mathbf{y} \rangle$ with respect to some function vector $\mathbf{y}$. Chapter 6 extends the treatment to encryption of one vector $\mathbf{x}_i$ per client and the decryption gives sums of inner products $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle$ with respect to an ensemble of vectors $(\mathbf{y}_i)_i$ in a functional key.

be public, if the identity control does not pass or if the tags are not the same, a totally random value is returned by the decryption procedure. We now face the same problem of checking equality among $\boxed{\mathsf{tag}_i}$ in the same manner that has to be done for identities from ID.

First of all, it is unlikely that we want to embed the checks $\boxed{\mathsf{tag}_i} \stackrel{?}{=} \boxed{\mathsf{tag}_j}$ in the $\mathcal{F}^{\mathsf{IP}}$-part. More specifically, we would have to make the decryption compute $(\sum_{i=1}^{n} \boxed{x_i}, \boxed{y_i}) + \tau z \cdot (\boxed{\mathsf{id}\text{-}\mathsf{k}} - \boxed{\mathsf{id}\text{-}\mathsf{ct}_i}) + \sum_{i=1}^{n-1} z_i(\boxed{\mathsf{tag}_i} - \boxed{\mathsf{tag}_{i+1}})$ from n ciphertext components ct_i of $\boxed{(x_i, \mathsf{id}\text{-}\mathsf{ct}_i)}$, for some random values $z, z_i \xleftarrow{\$} \mathbb{Z}_q$ and $\boxed{\mathbf{y} = (y_1, ..., y_n)}$. It is worth noting that the check $z_i(\boxed{\mathsf{tag}_i} - \boxed{\mathsf{tag}_{i+1}})$ needs two values defined at encryption time and not key extraction time. Therefore, in order for the functional key to "perform" the n required checks, all n tags $\boxed{(\mathsf{tag}_1, \ldots, \mathsf{tag}_n)}$ must be encrypted in an IBE-style in ct_i. Roughly speaking, this makes each ct_i of size linear in n, due to the number of group elements required for encrypting the n tags, in addition to a constant number of group elements for encrypting $\boxed{(x_i, \mathsf{id}\text{-}\mathsf{ct}_i)}$. Thus the total communication increases to quadratic in n over all n components ct_i, which is exactly what we are trying to avoid.

Furthermore, it might be tempting to embed the equality checks in the access control but because $\boxed{\mathsf{tag}_i, \mathsf{tag}_j}$ are defined only at encryption time, they are unknown to the key extraction for the ID-part. More generally, in a setting that permits a *different* [2] attribute set $\boxed{\mathsf{S}_i}$ in each individual ciphertext, one can try to regard $\boxed{\mathsf{tag}_i}$ as an attribute in $\boxed{\mathsf{S}_i}$. The correctness insists on the condition $\boxed{\mathbb{A}}\left(\boxed{\mathsf{S}_i}\right) = 1$ for all i and the equality checks $\boxed{\mathsf{tag}_i} \stackrel{?}{=} \boxed{\mathsf{tag}_j}$ must somehow be done by $\boxed{\mathbb{A}}\left(\boxed{\mathsf{S}_i}\right)$, which is not possible due to the fact that $\boxed{\mathsf{tag}_j}$ is independent of both $\boxed{\mathbb{A}}$ and $\boxed{\mathsf{S}_i}$. Consequently, we have to cope with the tags independently from the functionality's description. As a final remark, this also demonstrates the gap between MIFE and MCFE for the concrete functionality to compute inner products under access control by access structures, even though the general notion of MIFE can describe MCFE, provided that the evaluation functions of the underlying functionality class can test equality between $\boxed{\mathsf{tag}_i}$.

4.2.2 Adaptively Secure Single-Client Construction

Our construction for functional encryption schemes with fine-grained access control is using *Dual Pairing Vector Spaces* (DPVSes). We highlight our main ideas to achieve adaptive security. We refer to Section 3.3 for background on DPVSes. Our schemes are key-policy, such that the access structure $\mathbb{A}$ is expressed in the key using vectors $\{(\mathbf{k}^*_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}_{\mathsf{root}}\}$ over $\mathbb{G}_2$ and a set S of attributes are embedded in the ciphertext using vectors $\{(\mathbf{c}_j)_{j \in \mathsf{S}}, \mathbf{c}_{\mathsf{root}}\}$ over $\mathbb{G}_1$, where $\mathsf{List\text{-}Att}(\mathbb{A})$ is the list of attributes appearing in the access structure $\mathbb{A}$. We use a linear secret sharing scheme based on $\mathbb{A}$ to create the shares $(a_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ of $a_0 \xleftarrow{\$} \mathbb{Z}_q$. The shares will then be embedded in the functional secret key components $(\mathbf{k}^*_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$. When all the components corresponding to an authorized set in $\mathbb{A}$ are present, the shares can be combined to reconstruct the secret value a_0, which is now embedded in a key component $\mathbf{k}^*_{\mathsf{root}}$. In all vectors $(\mathbf{c}_j)_j$ and $\mathbf{c}_{\mathsf{root}}$, we put a random value ψ. Intuitively, $[\![\psi a_0]\!]_{\mathsf{t}}$ is masking the IPFE-related ciphertext of Agrawal *et al.*'s type [ALS16]. The vectors $((\mathbf{k}^*_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}^*_{\mathsf{root}})$ and $((\mathbf{c}_j)_{j \in \mathsf{S}}, \mathbf{c}_{\mathsf{root}})$ lie in the dual orthogonal bases. Performing the products $\mathbf{c}_j \times \mathbf{k}^*_j$ and combining over $j \in \mathsf{S}$, where S is an authorized set, will permit recovering $[\![\psi a_0]\!]_{\mathsf{t}}$ that can be used to cancel out $[\![\psi a_0]\!]_{\mathsf{t}}$ in $\mathbf{c}_{\mathsf{root}} \times \mathbf{k}_{\mathsf{root}}$:

$$\begin{array}{cccccccccccccccc} \mathbf{c}_j & (& \cdots & \psi & 0 & \cdots &)_{\mathbf{F}} ; & \mathbf{c}_{\mathsf{root}} & (& \cdots & \psi & 0 & \cdots &)_{\mathbf{H}} \\ \mathbf{k}^*_j & (& \cdots & a_j & 0 & \cdots &)_{\mathbf{F}^*}; & \mathbf{k}^*_{\mathsf{root}} & (& \cdots & a_0 & 0 & \cdots &)_{\mathbf{H}^*} \end{array}$$

We use the techniques for adaptively-secure ABE introduced in the original work of Okamoto and Takashima [OT10, OT12a, OT12b] in the ensuing steps. In vein of the *dual-system*

[2] If all clients must use the *same* set of attributes $\boxed{\mathsf{S}}$, we can treat $\boxed{\mathsf{tag}_i}$ as a virtual attribute in S, while enforcing the same $\boxed{\mathsf{S}}$ for all i. This implies that all $\boxed{\mathsf{tag}_i}$ must be the same. However, this approach requires a consensus among all n clients on S, which general might be more complicated than agreeing on tag.

methodology, there are two modes of operation for keys and ciphertexts: a normal mode and a *semi-functional* mode. A normal key can decrypt any ciphertext, a semi-functional key can decrypt only normal ciphertexts, and decrypting semi-functional ciphertexts using semi-functional keys gives totally random values. The dual-system method proves security by a sequence of indistinguishable changes to make the challenge ciphertext semi-functional, then to make the keys semi-functional and in the end the challenge message will be perfectly hidden from the adversary. Interestingly, there is a twist stemming from the security model when integrating this technique into our security proofs for FE with access control: an adversary can additionally query for keys that work with the challenge ciphertext, i.e. the key's policy is satisfied. So as to achieve adaptive security, we have to be much more careful about which key to turn semi-functional, because the keys whose policies are satisfied should be capable of decrypting the (semi-functional) challenge ciphertext.

Our goal is to mask the value a_0 in $\mathbf{k}^*_{\mathsf{root}}$ by introducing a random mask $a'_0 y$ in the coordinate of *hidden* basis vectors, i.e. those that are not used at all in real life and are defined only for the proof, while the facing coordinate in $\mathbf{c}_{\mathsf{root}}$ is also changed to τx so as to mask ψ:

$$\begin{array}{cccccccccc} \mathbf{c}_j & (& \cdots & \psi & \tau x z_j & \cdots &)_{\mathbf{F}} ; & \mathbf{c}_{\mathsf{root}} & (& \cdots & \psi & \tau x & \cdots &)_{\mathbf{H}} \\ \mathbf{k}^*_j & (& \cdots & a_j & a'_j y / z_j & \cdots &)_{\mathbf{F}^*}; & \mathbf{k}^*_{\mathsf{root}} & (& \cdots & a_0 & a'_0 y & \cdots &)_{\mathbf{H}^*} \end{array} .$$

The values x, y are known constants, $\tau, a'_0, (z_j)_j \xleftarrow{\$} \mathbb{Z}_q$, and $(a'_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ is another ensemble of secret shares for a'_0. Consequently, this will introduce a value $[\![\tau a'_0 xy]\!]_{\mathsf{t}}$ masking $[\![\psi a_0]\!]_{\mathsf{t}}$ when performing the product $\mathbf{c}_{\mathsf{root}} \times \mathbf{k}_{\mathsf{root}}$. We note that the value a'_0 is related to $(a'_j / z_j)_j$ by $a'_0 = \sum_{j \in \mathsf{S}'} z_j \cdot (a'_j / z_j)$ for any S' such that $\mathbb{A}(\mathsf{S}') = 1$. In the end, if $\mathbb{A}(\mathsf{S}) = 1$, from $\mathbf{c}_j$ and $\mathbf{k}_j$ it is possible to reconstruct $[\![\tau a'_0 xy]\!]_{\mathsf{t}}$ and recover $[\![\psi a_0]\!]_{\mathsf{t}}$. Otherwise, the entropy of a'_0 is preserved thanks to the randomness provided by $z_j \xleftarrow{\$} \mathbb{Z}_q$ for randomizing $(a'_j)_j$ to $(a'_j / z_j)_j$ in the components $(\mathbf{c}_j)_j$ of the *unique* challenge ciphertext[3], as well as the fact that $\mathbb{A}(\mathsf{S}) = 0$ means there will be some a'_j / z_j missing in the components $(\mathbf{k}^*_j)_j$ and the value z_j is information-theoretically hidden. Hence, if $\mathbb{A}(\mathsf{S}) = 0$ we will be able to change a'_0 to an independent and uniformly random value $r_0 \xleftarrow{\$} \mathbb{Z}^*_q$. It is obligatory that we apply this argument *key by key*, while considering the key's capability to decrypt the challenge ciphertext, because two different keys might mutually leak information about the same z_j and our statistically argument no longer holds. After a sequence of hybrids on the functional key queries, we can mask all the keys as desired so that the key and the challenge ciphertext will become readily semi-functional for later steps in the proof.

However, only for functional keys whose policy is not satisfied can we perform such a change from a'_0 to r_0, and we can decide the satisfiability only when the adversary adaptively queries for functional keys. Our idea is to introduce r_0 in *all* key components and at the same time use a mechanism to "cancel out" the masks $((a'_j / z_j)_j, r_0)$ in $((\mathbf{k}^*_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}^*_{\mathsf{root}})$ if $\mathbb{A}(\mathsf{S}) = 1$. It is indispensable to have this mechanism because otherwise, as soon as we change a'_0 to r_0, even the reconstruction $\sum_{j \in \mathsf{S}'} z_j \cdot (a'_j / z_j) = a'_0$ is not able to remove r_0 for a correct decryption. In our particular setting for computing inner-products, we observe that if $\mathbb{A}(\mathsf{S}) = 1$, then $\langle \Delta\mathbf{x}, \mathbf{y} \rangle = 0$ for the sake of avoiding trivial attacks, where $\Delta\mathbf{x} \coloneqq \mathbf{x}^*_1 - \mathbf{x}^*_0$ is the difference of the two left-or-right challenge messages and $\mathbf{y}$ is specified the functional key. In the selective setting where $\Delta\mathbf{x}$ is known in advance, the key and ciphertext components can simply be masked using the constants $(x, y) \coloneqq (1, \langle \Delta\mathbf{x}, \mathbf{y} \rangle)$. However, for the goal of adaptive security where $\Delta\mathbf{x}$ is unknown at the time of key extraction, we have to make a trade-off and use DPVSes of dimensions linear in the dimension n of vectors for inner-products and mask the key and ciphertext components as follows:

$$\begin{array}{cccccccccc} \mathbf{c}_j & (& \cdots & \psi & \tau z_j \Delta\mathbf{x}[1] & \cdots & \tau z_j \Delta\mathbf{x}[n] & \cdots &)_{\mathbf{F}} \\ \mathbf{k}^*_j & (& \cdots & a_j & a'_j \mathbf{y}[1] / z_j & \cdots & a'_j \mathbf{y}[n] / z_j & \cdots &)_{\mathbf{F}^*} \\ \hline \mathbf{c}_{\mathsf{root}} & (& \cdots & \psi & \tau \Delta\mathbf{x}[1] & \cdots & \tau \Delta\mathbf{x}[n] & \cdots &)_{\mathbf{H}} \\ \mathbf{k}^*_{\mathsf{root}} & (& \cdots & a_0 & r_0 \mathbf{y}[1] & \cdots & r_0 \mathbf{y}[n] & \cdots &)_{\mathbf{H}^*} \end{array}$$

[3]Since our single-client scheme is public-key, we can obtain multi-challenge security using a standard hybrid argument.

where each i-th pair of constants (x, y) is set to $(\Delta\mathbf{x}[i], \mathbf{y}[i])$ for all $i \in [n]$. Our arguments resort to a slight variant of the technique in [OT10, OT12a, OT12b], stated as a technical lemma (see [NPP22a, Lemma 4], or its more generalized version of Lemma 3.5). The lemma will use some auxiliary hidden vectors (which we do not show here) during the masking process and so as to economize the dimensions of our DPVSes, we apply the lemma n times in a sequence of hybrids to introduce $(\tau\Delta\mathbf{x}[i], r_0\mathbf{y}[i])_i$ while reusing and cleaning those auxiliary hidden vectors after each application. After successfully laying $(r_0\mathbf{y}[i])_i$ in place, the rest of the proof will use r_0 as a source of randomness to completely hide the challenge message. Our single-client constructions are presented in Section 4.3.

4.2.3 The "Duplicate-and-Compress" Technique

We give a glimpse of our main technical method to obtain a multi-client construction from our single-client construction, while maintaining the total ciphertext's size of order linear in n. The intriguing point we observe is as long as each client uses an independent DPVS, the technique we use to take care of the ciphertext/key vectors in the single-client case can be carried out in a *parallel* manner, to some extent. Therefore, in the security proof, we can distribute and accumulate in parallel the necessary information in small-dimension vectors rather than centralizing such information in few vectors of big dimension. Our treatment for the multi-client setting is twofold and we give below the main technical ideas.

Randomisation of secret shares. Along these lines of argument we make use of a technique to randomise secret shares that are integrated in keys for access control. In short, in a computationally indistinguishable manner, Lemma 3.5 from Section 3.3 allow us to introduce first a set of shares $(a'^{(\ell)}_{i,j})_j$ are secret shares of $a'^{(\ell)}_{i,0}$. Then the secret-share relation between $(a'^{(\ell)}_{i,j})_j$ and $a'^{(\ell)}_{i,0}$ is lifted by masking independently $(a'^{(\ell)}_{i,j})_j$ for each attribute j in the key-policy by a uniformly random $z_j \xleftarrow{\$} \mathbb{Z}_q^*$. Importantly, conditioned that each attribute j emits at most one time to the adversary, this masking makes each $a'^{(\ell)}_{i,j}/z_j$ information-theoretically hidden if j is unknown, *i.e.* the attributes from the adversary does not contain j and is not satisfying the policy in the key. These hiding property of $a'^{(\ell)}_{i,j}/z_j$ is used then for later steps of the proof. Similar techniques are encountered again in the proof of Theorem 6.11 in Chapter 6, for which we also give fully formal details.

The more restrictive MCFE. Firstly, Section 4.4.2 presents a construction that enforces the same $\mathsf{S}_1 = \cdots = \mathsf{S}_n = \mathsf{S}$ for all clients, by hashing it using a full-domain hash function modeled as a random oracle (RO), along with the tag at the time of encryption. Indeed, we will use an argument resembling what we do in the single-client construction and perform a masking procedure key by key, where the functional key query for $(\mathbb{A}, \mathbf{y}^{(\ell)})$ is indexed by ℓ. For each $i \in [n]$,we mask $(\mathbf{k}^*_{i,j})_j = (..., a^{(\ell)}_{i,j}, a'^{(\ell)}_{i,j} y/z_j, ...)_j$,$\mathbf{k}^*_{i,\mathsf{root}} = (..., a^{(\ell)}_{i,0}, a'^{(\ell)}_{i,0} y, ...)$ and $(\mathbf{c}_{i,j})_j = (..., \psi_i, \tau x z_j, ...)_j, \mathbf{c}_{i,\mathsf{root}} = (..., \psi_i, \tau x, ...)$, where $(a^{(\ell)}_{i,j})_j, (a'^{(\ell)}_{i,j})_j$ are secret shares of $a^{(\ell)}_{i,0}, a'^{(\ell)}_{i,0}$ respectively. In this more restrictive case of Section 4.4.2 where all n clients use the same S, it entails all clients $i \in [n]$ using the same $a^{(\ell)}_0, a'^{(\ell)}_0$ with their secret shares $(a^{(\ell)}_j)_j, (a'^{(\ell)}_j)_j$ in $(\mathbf{k}^*_{i,j})_j = (..., a^{(\ell)}_j, a'^{(\ell)}_j y/z_j, ...)_j$ and $\mathbf{k}^*_{i,\mathsf{root}} = (..., a^{(\ell)}_0, a'^{(\ell)}_0 y, ...)$. Afterwards, we want to replace $a'^{(\ell)}_0$ by an independent and uniformly random value $r^{(\ell)}_0 \xleftarrow{\$} \mathbb{Z}_q^*$ if $\mathbb{A}(\mathsf{S}_i) = 0$ and clearing the masks otherwise. As our first observation, the reasoning is still based crucially on the fact that in S there will lack some j whose corresponding z_j permits recovering $a'^{(\ell)}_0 = \sum_j z_j(a'^{(\ell)}_j/z_j)$ if $\mathbb{A}(\mathsf{S}) = 0$. It gets clear that as long as $\mathbb{A}(\mathsf{S}) = 0$, for all i independently, the same argument will hold because all i use the same set S of attributes. This observation leads to a *compression* of all $(\mathbf{c}_{i,j})_j, (\mathbf{k}^*_{i,j})_j$ into one pair of dual bases $(\mathbf{F}, \mathbf{F}^*)$ instead of n separate pairs for each $i \in [n]$. As a second observation,when $\mathbb{A}(\mathsf{S}) = 1$, all ciphertext components must be combined together for a correct decryption. As a result, to program the canceling mechanism, instead of naively embedding n pairs of constants $(\Delta\mathbf{x}[k], \mathbf{y}^{(\ell)}[k])_{k=1}^n$ in $(\mathbf{c}_{i,\mathsf{root}}, (\mathbf{c}_{i,j})_j, \mathbf{k}^*_{i,\mathsf{root}}, (\mathbf{k}^*_{i,j})_j)$ for *each* i, we

only need to embed $(\Delta\mathbf{x}[i], \mathbf{y}^{(\ell)}[i])$ in $(\mathbf{c}_{i,\mathsf{root}}, (\mathbf{c}_{i,j})_j, \mathbf{k}^*_{i,\mathsf{root}}, (\mathbf{k}^*_{i,j})_j)$. The grouping by i of the products $\mathbf{c}_{i,\mathsf{root}} \times \mathbf{k}^*_{i,\mathsf{root}}$ as well as $\sum_j \mathbf{c}_{i,j} \times \mathbf{k}^*_{i,j}$ will retrieve $\left[\!\left[\tau r_0^{(\ell)} \langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \right]\!\right]_{\mathsf{t}}$ and we proceed the remaining as in the single-client proof. We point out that in the multi-client setting, it might be the case that some i are corrupted and the retrieval of $\left[\!\left[\tau r_0^{(\ell)} \langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \right]\!\right]_{\mathsf{t}}$ is more complicated when regrouping over i. However, by carefully defining (see Definition 4.3) and considering only *admissible* adversaries, *i.e.* they cannot win by trivial attacks[4], it remains the case. This individual insertion of $(\Delta\mathbf{x}[i], \mathbf{y}^{(\ell)}[i])$ for each i leads to a *duplication* of one pair of dual bases $(\mathbf{H}_i, \mathbf{H}^*_i)$ for each $(\mathbf{c}_{i,\mathsf{root}}, \mathbf{k}^*_{i,\mathsf{root}})$, while all $(\mathbf{c}_{i,j})_j, (\mathbf{k}^*_{i,j})_j$ are readily put in the same basis following our first observation:

(Compressing to same bases) for all $i \in [n]$	$\mathbf{c}_{i,j}$	(	$\cdots$	ψ	$\tau\Delta\mathbf{x}[i]z_j$	$\cdots$	$)_{\mathbf{F}}$
	$\mathbf{k}^*_{i,j}$	(	$\cdots$	$a_j^{(\ell)}$	$a_j'^{(\ell)}\mathbf{y}^{(\ell)}[i]/z_j$	$\cdots$	$)_{\mathbf{F}^*}$
(Duplicating bases) for each $i \in [n]$	$\mathbf{c}_{i,\mathsf{root}}$	(	$\cdots$	ψ	$\tau\Delta\mathbf{x}[i]$	$\cdots$	$)_{\mathbf{H}_i}$
	$\mathbf{k}^*_{i,\mathsf{root}}$	(	$\cdots$	$a_0^{(\ell)}$	$a_0'^{(\ell)}\mathbf{y}^{(\ell)}[i]$	$\cdots$	$)_{\mathbf{H}^*_i}$

We emphasize that this parallel process is feasible thanks to a conveniently smooth control, as low as the level of the vectors' coordinates in DPVSes. This potential of parallelization helps us spread the necessary information for answering adaptive key queries, which accounts for the linearly large dimension, into n collections $\{(\mathbf{k}^*_{i,j})_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}, \mathbf{k}^*_{i,\mathsf{root}}\}_{i \in [n]}$. On the one hand, we change the vectors $(\mathbf{k}^*_{i,j}, \mathbf{c}_{i,j})_{i,j}$ in parallel for all i, while these vectors are written in bases $(\mathbf{F}, \mathbf{F}^*)$. On the other hand, we change the vectors $(\mathbf{k}^*_{i,\mathsf{root}}, \mathbf{c}_{i,\mathsf{root}})_i$ independently for each client i, using the fact that each pair $(\mathbf{k}^*_{i,\mathsf{root}}, \mathbf{c}_{i,\mathsf{root}})$ belong to a separate pair of dual bases $(\mathbf{H}_i, \mathbf{H}^*_i)$. In the end, instead of using n bases of dimension n, we can use n bases of *constant* dimension for $(\mathbf{k}^*_{i,\mathsf{root}})_i$ along with one *constant*-dimension basis for all $\{(\mathbf{k}^*_{i,j})_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}\}_i$, saving a factor n in the ciphertext's size.

The more flexible MCFE. Section 4.4.4 discusses an extension of the above MCFE construction where we do not impose the same set of attributes among n clients. Each client i can now encrypt using a different S_i and the decryption can decrypt the inner-product if and only if $\mathbb{A}(\mathsf{S}_i) = 1$ for all i. Unsurprisingly, our argument as it is from the previous construction, for masking and for replacing $a_{i,0}'^{(\ell)}$ by an independent and uniformly random value, does not hold anymore because there might be two keys corresponding to $\mathbb{A}^{(\ell)}$ and $\mathbb{A}^{(\ell')}$ such that $\mathbb{A}^{(\ell)}(\mathsf{S}_i) \neq \mathbb{A}^{(\ell')}(\mathsf{S}_i)$ and the adversary might try to use key components of the ℓ'-th query to recover $a_{i,0}'^{(\ell)}$ in the ℓ-th query. We thus make use of another layer of random secret shares $(d_{\ell,i})_{i=1}^n$ over n components of each ℓ-th functional key, facing θ_i in the ciphertext components such that $\sum_{i=1}^n \theta_i d_{\ell,i} = 0$. The values $(\theta_i)_i$ are generated as part of the master secret key but $(d_{\ell,i})_{i=1}^n$ are chosen independently for each key. A fully working key can be obtain only if all the n components corresponding to $(d_{\ell,i})_{i=1}^n$ are combined. That will prevent the adversary from trying to mix components between two different keys, i.e. if $\mathbb{A}^{(\ell)}(\mathsf{S}_i) = 0$ we can be sure that $a_{i,0}'^{(\ell)}$ retains its entropy and stays hidden. After a similar masking step using the secret shares $(a_{i,j}'^{(\ell)})_j$ of $a_{i,0}'^{(\ell)}$ independently generated for each i, the randomness provided by $(d_{\ell,i})_{i=1}^n$ allows us to tweak $a_{i,0}'^{(\ell)}$ with a uniformly random value $r_0^{(\ell)}$:

(Compressing) for all $i \in [n]$	$\mathbf{c}_{i,j}$	(	$\cdots$	ψ	$\tau\Delta\mathbf{x}[i]z_j$	$\cdots$	$\cdots)_{\mathbf{F}}$
	$\mathbf{k}^*_{i,j}$	(	$\cdots$	$a_j^{(\ell)}$	$a_{i,j}'^{(\ell)}\mathbf{y}^{(\ell)}[i]/z_j$	$\cdots$	$\cdots)_{\mathbf{F}^*}$
(Duplicating) for each $i \in [n]$	$\mathbf{c}_{i,\mathsf{root}}$	(	$\cdots$	ψ	$\tau\Delta\mathbf{x}[i]$	θ_i	$\cdots)_{\mathbf{H}_i}$
	$\mathbf{k}^*_{i,\mathsf{root}}$	(	$\cdots$	$a_0^{(\ell)}$	$(a_{i,0}'^{(\ell)} + r_0^{(\ell)})\mathbf{y}^{(\ell)}[i]$	$d_{\ell,i}$	$\cdots)_{\mathbf{H}^*_i}$

It is of the utmost importance that we rely on $(d_{\ell,i})_{i=1}^n$, which is particular for each ℓ-th key, to carry out this change from $a_{i,0}'^{(\ell)}$ to $a_{i,0}'^{(\ell)} + r_0^{(\ell)}$. Or else, the adversary can mix and match the

[4]For instance, the adversary might corrupt i^*, query a left-or-right challenge $(\mathbf{x}_0, \mathbf{x}_1)$ where $\Delta\mathbf{x}[i^*] \coloneqq \mathbf{x}_0[i^*] - \mathbf{x}_1[i^*] \neq 0$ and $\Delta\mathbf{x}[i] = 0$ for $i \neq i^*$, then decrypt the challenge ciphertext with a satisfied key for $\mathbf{y}^{(\ell)}$ whose i^*-th entry is non-zero.

ℓ-th and ℓ'-th keys to remove $a_{i,0}'^{(\ell)}$ and distinguish the adding of $r_0^{(\ell)}$, regardless whether S_i is authorized or not. The argument is now computational, in contrast to the information-theoretical indistinguishability when changing from $a_0'^{(\ell)}$ to $r_0^{(\ell)}$ in the more restrictive MCFE. We now perform an unmasking by going backwards to remove the sharing $(a_{i,j}'^{(\ell)})_j$ and $a_{i,0}'^{(\ell)}$ in the key. This transition is completely symmetric. If $\mathbb{A}(\mathsf{S}_i) = 1$ for all i, then the admissibility requires $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle = 0$ and the noise $\tau r_0^{(\ell)}$ can be removed. Otherwise, in case $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \neq 0$, the mask $\tau r_0^{(\ell)}$ persists but the admissbility implies there exists i such that $\mathbb{A}(\mathsf{S}_i) = 0$ and the functional key cannot decrypt the challenge ciphertext. We emphasize that the incapability of the key when $\mathbb{A}(\mathsf{S}_i) = 0$ is ensured by $(d_{\ell,i})_{i=1}^n$. After introducing $r_0^{(\ell)}$, the remaining steps resemble the proof of the less flexible construction in Section 4.4.2. A desirable byproduct of this more flexible construction is that the hash function, which is modeled as a random oracle (RO), is now applied only on the tag. Therefore, we can obtain an MIFE in the standard model that is comparable to the work in [ACGU20] by fixing the hash value of a tag for all ciphertexts and publishing it as a parameter of the scheme.

More on Security of MIFE from MCFE. We emphasize that the obtained MIFE after our more attribute-flexible MCFE with access control in Section 4.4.4 inherits the security from the latter. Notably, for each challenge tag^*, condition 1 in Definition 4.3 prohibits the adversary from querying multiple messages or attributes with the same challenge tag, at the same slot i. This implies that in the MIFE resultant, security does not hold with repetitions at the same slot i either. In general, under access control, the transformation from MCFE to MIFE is highly non-trivial as mentioned in [ATY23a], in order to preserve the security with repetitions on both messages and attributes. In the later Chapter 6 of this thesis, we presents a MCFE with access control that allows repetitions on the messages, but still forbidding repetitions on the attributes.

4.3 Single-Client Functional Encryption For Inner Products with Fine-Grained Access Control via LSSS

We present constructions of FE for the inner-product functionality with attribute-based control expressed using linear secret sharing schemes, starting with the simpler single-client setting. We are in the bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are written additively. The function class of interests is $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$ where $\mathcal{F}^{\mathsf{IP}}$ contains $F_\mathbf{y} : \left(\mathbb{Z}_q^*\right)^n \to \mathbb{Z}_q$ defined as $F_\mathbf{y}(\mathbf{x}) \coloneqq \langle \mathbf{x}, \mathbf{y} \rangle$. The access control is given by $\mathsf{Rel} : \mathsf{LSSS} \times 2^{\mathsf{Att}} \to \{0,1\}$, where $\mathsf{Rel}(\mathbb{A}, \mathsf{S}) = \mathbb{A}(\mathsf{S})$, the class LSSS contains Linear Secret Sharing Schemes over Att, and 2^{Att} denotes the superset of an attribute space $\mathsf{Att} \subseteq \mathbb{Z}_q$. Our constructions are key-policy, where $\mathbb{A}$ is embedded in the key and S is specified in the ciphertext. In order to facilitate the understanding and the motivation of our later multi-client constructions in Section 4.4, we present both selectively-secure and adaptively-secure single-client constructions in Figure 4.1. We leverage the selectively-secure scheme to obtain the adaptively-secure one by replacing certain elements in the former by the corresponding boxed components for the latter.

The main difference between the adaptive version and the selectively-secure version is the increase in the dimension of dual bases, from constant dimensions to dimensions linear in n. The details can be found in Figure 4.1. The computation for encrypting and decrypting stays essentially the same. We refer to the technical overview in Section 4.2 for the main ideas why using bigger DPVSes allows us to achieve the stronger adaptive notion. The *correctness* can be verified in a straightforward manner. Theorem 4.1 proves the adaptive IND-security for the construction corresponding to boxed components in Figure 4.1, where the adversary can query a unique challenge ciphertext and multiple functional keys. Using a standard hybrid argument and recalling that our scheme is public-key provide us with adaptive security against multiple challenge ciphertexts. The easier selective security can be proved using similar techniques.

Theorem 4.1. *Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ be an IPFE scheme with fine-grained access control via LSSS presented in Figure 4.1 in a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$, for the functionality class $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$. Then, $\mathcal{E}$ is secure against chosen-plaintext attacks, adaptively in the attributes and the challenge messages, if the SXDH assumption holds for $\mathbb{G}_1$ and $\mathbb{G}_2$. More precisely, for $\lambda \in \mathbb{N}$ and for any ppt adversary $\mathcal{A}$, let n be the dimension of vectors for inner-product computation, K denote the total number of functional key queries, and P denote the total number of attributes used by the adversary. We have the following bound:*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{E},\mathcal{F}^{\mathsf{IP}},\mathsf{LSSS},\mathcal{A}}(1^\lambda) \leq (2nK \cdot (P(6P+3)+2)+5) \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$$

where $\mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$ denotes the maximum advantage over ppt adversaries against the SXDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ set up with parameter λ.

The full proof can be found in [NPP22a, NPP22b, Appendix B.3]. We give below the main ideas and the sequence of games employed in the proof.

Proof (Main ideas). The sequence of games can be found in Figure 4.2. Using the dual-system methodology, we first change the challenge ciphertext into semi-functional and then we want to change the functional keys into semi-functional as well. This can be done only for the keys corresponding to $(\mathbb{A}, \mathbf{y})$ such that

$$\langle \mathbf{x}_0^*, \mathbf{y} \rangle \neq \langle \mathbf{x}_1^*, \mathbf{y} \rangle \ . \tag{4.1}$$

According to the model of security, condition (4.1) implies that the access structure $\mathbb{A}$ is not satisfied by the attributes S in the challenge ciphertext. Hence, changing the foregoing key into semi-functional does not affect the fact that the ciphertext, which is already semi-functional,

$\underline{\mathsf{Setup}(1^\lambda)}$**:** Choose two pairs of dual orthogonal bases $(\mathbf{F}, \mathbf{F}^*)$ and $(\mathbf{H}, \mathbf{H}^*)$ where $(\mathbf{H}, \mathbf{H}^*)$ is a pair of bases of the dual pairing vector spaces $(\mathbb{G}_1^4, \mathbb{G}_2^4)$ $\boxed{(\mathbb{G}_1^{n+3}, \mathbb{G}_2^{n+3})}$, and $(\mathbf{F}, \mathbf{F}^*)$ are dual bases of $(\mathbb{G}_1^8, \mathbb{G}_2^8)$ $\boxed{(\mathbb{G}_1^{n+7}, \mathbb{G}_2^{n+7})}$. We write

$$\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_4) \qquad \mathbf{H}^* = (\mathbf{h}_1^*, \mathbf{h}_2^*, \mathbf{h}_3^*, \mathbf{h}_4^*)$$
$$\boxed{\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3, \mathbf{h}_4, \ldots, \mathbf{h}_{n+3})} \qquad \boxed{\mathbf{H}^* = (\mathbf{h}_1^*, \mathbf{h}_2^*, \mathbf{h}_3^*, \mathbf{h}_4^*, \ldots, \mathbf{h}_{n+3}^*)}$$
$$\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{f}_4, \mathbf{f}_5, \mathbf{f}_6, \mathbf{f}_7, \mathbf{f}_8) \qquad \mathbf{F}^* = (\mathbf{f}_1^*, \mathbf{f}_2^*, \mathbf{f}_3^*, \mathbf{f}_4^*, \mathbf{f}_5^*, \mathbf{f}_6^*, \mathbf{f}_7^*, \mathbf{f}_8^*)$$
$$\boxed{\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{f}_4, \ldots, \mathbf{f}_{n+5}, \mathbf{f}_{n+6}, \mathbf{f}_{n+7})} \qquad \boxed{\mathbf{F}^* = (\mathbf{f}_1^*, \mathbf{f}_2^*, \mathbf{f}_3^*, \mathbf{f}_4^*, \ldots, \mathbf{f}_{n+5}^*, \mathbf{f}_{n+6}^*, \mathbf{f}_{n+7}^*)}$$

and sample $\mu, z \xleftarrow{\$} \mathbb{Z}_q^*, S, U \xleftarrow{\$} (\mathbb{Z}_q^*)^n$ and write $S = (s_1, \ldots, s_n)$, $U = (u_1, \ldots, u_n)$. Output the public key and the master secret key as

$$\begin{cases} \mathsf{pk} := \left(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ ([\![s_i + \mu \cdot u_i]\!]_1)_{i\in[n]}\right) \\ \mathsf{msk} := (z,\ S,\ U,\ (\mathbf{f}_i^*)_{i\in[3]},\ (\mathbf{h}_i^*)_{i\in[3]})\ . \end{cases}$$

$\underline{\mathsf{Extract}(\mathsf{msk}, \mathbb{A}, \mathbf{y} \in \mathbb{Z}_q^n)}$**:** Let $\mathbb{A}$ be an LSSS-realizable monotone access structure over a set of attributes $\mathsf{Att} \subseteq \mathbb{Z}_q$. First, sample $a_0 \xleftarrow{\$} \mathbb{Z}_q$ and run the labeling algorithm $\Lambda_{a_0}(\mathbb{A})$ (see (3.3)) to obtain the labels $(a_j)_j$ where j runs over the attributes in Att. In the end, it holds that $a_0 = \sum_{j\in A} c_j \cdot a_j$ where j runs over an authorized set $A \in \mathbb{A}$ and $\mathbf{c}_A = (c_j)_{j\in A}$ is the reconstruction vector from LSSS w.r.t A. We denote by $\mathsf{List\text{-}Att}(\mathbb{A})$ the list of attributes appearing in $\mathbb{A}$, with possible repetitions. Parse $\mathsf{msk} = (z,\ S,\ U,\ (\mathbf{f}_i^*)_{i\in[3]},\ (\mathbf{h}_i^*)_{i\in[3]})$. Compute:

$$\mathbf{k}_j^* := (\pi_j \cdot (j,1),\ a_j \cdot z,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})$$
$$\boxed{\mathbf{k}_j^* := (\pi_j \cdot (j,1),\ a_j \cdot z, \overbrace{0, \ldots, 0}^{n \text{ times}},\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})}$$
$$\mathbf{m}_i^* := [\![\mathbf{y}[i]]\!]_2 \text{ for } i \in [n]$$
$$\mathbf{k}_{\mathsf{ipfe}}^* := (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_0 \cdot z,\ 0)_{\mathbf{H}^*} \quad \boxed{\mathbf{k}_{\mathsf{ipfe}}^* := (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_0 \cdot z, \overbrace{0, \ldots, 0}^{n \text{ times}})_{\mathbf{H}^*}}$$

where $\pi_j \xleftarrow{\$} \mathbb{Z}_q$. Output $\mathsf{sk}_{\mathbb{A},\mathbf{y}} := \left(\left(\mathbf{k}_j^*\right)_j, (\mathbf{m}_i^*)_{i\in[n]}, \mathbf{k}_{\mathsf{ipfe}}^*\right)$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathbf{x}, \mathsf{S})}$**:** Parse the public key $\mathsf{pk} = \left(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ ([\![s_i + \mu \cdot u_i]\!]_1)_{i\in[n]}\right)$ and $\mathsf{S} \subseteq \mathsf{Att} \subseteq \mathbb{Z}_q$ as the set of attributes, then sample $\omega, \psi \xleftarrow{\$} \mathbb{Z}_q$. Compute

$$\mathbf{c}_j = \sigma_j \cdot \mathbf{f}_1 - j \cdot \sigma_j \cdot \mathbf{f}_2 + \psi \cdot \mathbf{f}_3 = (\sigma_j \cdot (1, -j),\ \psi,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}} \text{ for each } j \in \mathsf{S}$$
$$\boxed{\mathbf{c}_j = (\sigma_j \cdot (1, -j),\ \psi, \overbrace{0, \ldots, 0}^{n \text{ times}}, 0,\ 0,\ 0,\ 0)_{\mathbf{F}} \text{ for each } j \in \mathsf{S}}$$

where $\sigma_j \xleftarrow{\$} \mathbb{Z}_q$. Finally, compute

$$\mathbf{t}_i = \omega \cdot [\![s_i + \mu \cdot u_i]\!]_1 + [\![\mathbf{x}[i]]\!]_1 = [\![\omega \cdot (s_i + \mu u_i) + \mathbf{x}[i]]\!]_1 \text{ for } i \in [n]$$
$$\mathbf{c}_{\mathsf{ipfe}} = \omega \cdot (\mathbf{h}_1 + \mu\mathbf{h}_2) + \psi \cdot \mathbf{h}_3 = (\omega,\ \mu\omega,\ \psi,\ 0)_{\mathbf{H}} \quad \boxed{\mathbf{c}_{\mathsf{ipfe}} = (\omega,\ \mu\omega,\ \psi,\ \overbrace{0, \ldots, 0}^{n \text{ times}})_{\mathbf{H}}}$$

where $\sigma_i \xleftarrow{\$} \mathbb{Z}_q$ for every $i \in [n]$ and output $\mathsf{ct} := \left((\mathbf{c}_j)_{j\in\mathsf{S}}, (\mathbf{t}_i)_{i\in[n]}, \mathbf{c}_{\mathsf{ipfe}}\right)$.

$\underline{\mathsf{Dec}(\mathsf{sk}_{\mathbb{A},\mathbf{y}}, \mathsf{ct})}$**:** Parse $\mathsf{ct} = \left((\mathbf{c}_j)_{j\in\mathsf{S}},\ (\mathbf{t}_i)_{i\in[n]},\ \mathbf{c}_{\mathsf{ipfe}}\right)$ and $\mathsf{sk}_{\mathbb{A},\mathbf{y}} := \left(\left(\mathbf{k}_j^*\right)_{j\in\mathsf{List\text{-}Att}(\mathbb{A})},\ (\mathbf{m}_i^*)_{i\in[n]},\ \mathbf{k}_{\mathsf{ipfe}}^*\right)$. If there exists $A \subseteq \mathsf{S}$ and $A \in \mathbb{A}$, then compute the reconstruction vector $\mathbf{c} = (c_j)_j$ of the LSSS for A and

$$[\![\mathsf{out}]\!]_{\mathsf{t}} = \sum_{j\in A} \mathbf{c}_j \times (c_j \cdot \mathbf{k}_j^*) + \sum_{i=1}^{n} (\mathbf{e}(\mathbf{t}_i, \mathbf{m}_i^*)) - \left(\mathbf{c}_{\mathsf{ipfe}} \times \mathbf{k}_{\mathsf{ipfe}}^*\right)$$

Finally, compute the discrete logarithm and output $\mathsf{out} \in \mathbb{Z}_q$. Else, output $\perp$.

Figure 4.1: The selectively-secure and $\boxed{\text{adaptively-secure}}$ single-client constructions for IPFE with fine-grained access control via LSSS. The high-level ideas can be found in the technical overview of Section 4.2 and more details are presented in [NPP22a, Section 4].

Game G_0 : $a_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$, $(a_{\ell,j})_{j\in\mathsf{List\text{-}Att}(\mathbb{A})} \leftarrow \Lambda_{a_{\ell,0}}(\mathbb{A})$, $\mathsf{pk} := \Big(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ ([\![s_i + \mu\cdot u_i]\!]_1)_i\Big)$, $\mathbf{F} \in \mathbb{G}_1^{(n+7)\times(n+7)}$, $\mathbf{H} \in \mathbb{G}_1^{(n+3)\times(n+3)}$

$\mathbf{c}_j$	$($	$\sigma_j\cdot(1,-j)$	ψ	0	$\cdots$	0	0	0	0	0	$)_{\mathbf{F}}$
$\mathbf{k}^*_{\ell,j}$	$($	$\pi_{\ell,j}\cdot(j,1)$	$a_{\ell,j}\cdot z$	0	$\cdots$	0	0	0	0	0	$)_{\mathbf{F}^*}$

$\mathbf{t}_i$	$[\![\omega\cdot(s_i+\mu u_i)+\mathbf{x}^*_b[i]]\!]_1$
$\mathbf{m}^*_{\ell,i}$	$[\![\mathbf{y}_\ell[i]]\!]_2$

$\mathbf{c}_{\mathsf{ipfe}}$	$($	ω	$\mu\omega$	ψ	0	$\cdots$	0	$)_{\mathbf{H}}$
$\mathbf{k}^*_{\ell,\mathsf{ipfe}}$	$($	$\langle\mathbf{s},\mathbf{y}_\ell\rangle$	$\langle\mathbf{u},\mathbf{y}_\ell\rangle$	$a_{\ell,0}z$	0	$\cdots$	0	$)_{\mathbf{H}^*}$

Game G_1 : $r'_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$, $\Delta\mathbf{x} := \mathbf{x}^*_b - \mathbf{x}^*_1$, $\mathsf{pk} := \Big(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ ([\![s_i+\mu\cdot u_i]\!]_1)_i\Big)$

$\mathbf{c}_j$	$($	$\sigma_j\cdot(1,-j)$	ψ	0	$\cdots$	0	0	0	0	0	$)_{\mathbf{F}}$
$\mathbf{k}^*_{\ell,j}$	$($	$\pi_{\ell,j}\cdot(j,1)$	$a_{\ell,j}\cdot z$	0	$\cdots$	0	0	0	0	0	$)_{\mathbf{F}^*}$

$\mathbf{c}_{\mathsf{ipfe}}$	$($	ω	$\mu\omega$	ψ	$\tau\Delta\mathbf{x}[1]$	$\cdots$	$\tau\Delta\mathbf{x}[n]$	$)_{\mathbf{H}}$
$\mathbf{k}^*_{\ell,\mathsf{ipfe}}$	$($	$\langle\mathbf{s},\mathbf{y}_\ell\rangle$	$\langle\mathbf{u},\mathbf{y}_\ell\rangle$	$a_{\ell,0}\cdot z$	$r'_{\ell,0}\mathbf{y}_\ell[1]$	$\cdots$	$r'_{\ell,0}\mathbf{y}_\ell[n]$	$)_{\mathbf{H}^*}$

Game G_2 : $\omega' \xleftarrow{\$} \mathbb{Z}_q$, $\mathsf{pk} := \Big(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ ([\![s_i+\mu\cdot u_i]\!]_1)_i\Big)$

$\mathbf{t}_i$	$\Big[\!\!\Big[\omega\cdot s_i + \omega'\cdot u_i + \mathbf{x}^*_b[i]\Big]\!\!\Big]_1$
$\mathbf{m}^*_{\ell,i}$	$[\![\mathbf{y}_\ell[i]]\!]_2$

$\mathbf{c}_{\mathsf{ipfe}}$	$($	ω	ω'	ψ	$\tau\Delta\mathbf{x}[1]$	$\cdots$	$\tau\Delta\mathbf{x}[n]$	$)_{\mathbf{H}}$
$\mathbf{k}^*_{\ell,\mathsf{ipfe}}$	$($	$\langle\mathbf{s},\mathbf{y}_\ell\rangle$	$\langle\mathbf{u},\mathbf{y}_\ell\rangle$	$a_{\ell,0}\cdot z$	$r'_{\ell,0}\mathbf{y}_\ell[1]$	$\cdots$	$r'_{\ell,0}\mathbf{y}_\ell[n]$	$)_{\mathbf{H}^*}$

Game G_3 : $\omega', r''_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$, $\mathbf{s}' = \mathbf{s} + \Delta\mathbf{s}, \mathbf{u}' = \mathbf{u} + \Delta\mathbf{u}$, where $\Delta\mathbf{s}, \Delta\mathbf{u} \in \mathbb{Z}_q^n$ s.t. $\omega\cdot\Delta\mathbf{s} + \omega'\cdot\Delta\mathbf{u} = \mathbf{x}_b - \mathbf{x}_0$ and $\Delta\mathbf{s} + \mu\cdot\Delta\mathbf{u} = 0$, $\mathsf{pk} = \Big(\mathbf{h}_1 + \mu\mathbf{h}_2,\ \mathbf{h}_3,\ (\mathbf{f}_i)_{i\in[3]},\ ([\![s_i+\mu\cdot u_i]\!]_1)_i\Big)$

$\mathbf{t}_i$	$[\![\omega s'_i + \omega' u'_i + \mathbf{x}^*_0[i]]\!]_1$
$\mathbf{m}^*_{\ell,i}$	$[\![\mathbf{y}_\ell[i]]\!]_2$

$\mathbf{c}_{\mathsf{ipfe}}$	$($	ω	ω'	ψ	$\tau\Delta\mathbf{x}[1]$	$\cdots$	$\tau\Delta\mathbf{x}[n]$	$)_{\mathbf{H}}$
$\mathbf{k}^*_{\ell,\mathsf{ipfe}}$	$($	$\langle\mathbf{s}',\mathbf{y}_\ell\rangle$	$\langle\mathbf{u}',\mathbf{y}_\ell\rangle$	$a_{\ell,0}\cdot z$	$r''_{\ell,0}\mathbf{y}_\ell[1]$	$\cdots$	$r''_{\ell,0}\mathbf{y}_\ell[n]$	$)_{\mathbf{H}^*}$

Figure 4.2: Games for Theorem 4.1. The index i runs in $\{1,\dots,n\}$. The index j runs in $\mathsf{List\text{-}Att}(\mathbb{A})$ for key components and in S for ciphertext components. The index ℓ runs in $\{1,\dots,K\}$ for the functional key queries. The transition from G_0 to G_1 can be found in [NPP22a, Lemma 24], which will make use of the auxiliary vectors in $(\mathbf{F},\mathbf{F}^*)$ and $(\mathbf{H},\mathbf{H}^*)$ and contains applications of the "swapping" lemma [NPP22a, NPP22b, Lemma 4]. The generalized version of "swapping" lemma can also be found in Lemma 3.5.

cannot be decrypted using this key. On the other hand, for the functional secret key associated to $(\mathbb{A},\mathbf{y})$ where $\langle\Delta\mathbf{x},\mathbf{y}'\rangle = 0$ and $\Delta\mathbf{x} := \mathbf{x}^*_1 - \mathbf{x}^*_0$, it can remain normal. These keys include those whose policy is satisfied by the attributes in the challenge ciphertext and the decryption will return $\langle\mathbf{x}^*_0,\mathbf{y}\rangle = \langle\mathbf{x}^*_1,\mathbf{y}\rangle$ as expected. To prove the adaptive version, we need a strategy to change the challenge ciphertext and the keys into semi-functional such that the masks in the vectors exist only when condition (4.1) holds. Moreover, because the functional keys might be queried before the challenge messages are declared (we are in the adaptive setting), the keys should still allow correct decryption of normal ciphertexts, which the adversary can compute using pk as well as the later challenge ciphertext if the policy in the key is satisfied. Using the terminology from [OT12b], our main idea is using auxiliary *hidden* vectors $(\mathbf{f}_4,\dots,\mathbf{f}_{n+7})$ over $\mathbf{F}$ and $(\mathbf{h}_4,\dots,\mathbf{h}_{n+3})$ over $\mathbf{H}$, as well as their dual counterparts in $\mathbf{F}^*,\mathbf{H}^*$. These hidden subspace vectors will accommodate $\tau\Delta\mathbf{x}[i]$ in the $(i+3)$-th coordinate of the challenge ciphertext $\mathbf{c}_{\mathsf{ipfe}}$, and $r_0\mathbf{y}[i]$ in the $(i+3)$-th coordinate of functional key $\mathbf{k}_{\mathsf{ipfe}}$ corresponding to $\mathbf{y}$, for each $i\in[n]$ and the random masks $\tau, r_0 \xleftarrow{\$} \mathbb{Z}_q$. Then, when taking the products of vectors in DPVS, there will be a term $\tau r_0\sum_{i\in[n]}\Delta\mathbf{x}[i]\mathbf{y}[i] = \tau r_0\langle\Delta\mathbf{x},\mathbf{y}\rangle$ and it will act as a mask only when $\langle\Delta\mathbf{x},\mathbf{y}\rangle \neq 0$. The masking is done by each index $i\in[n]$, applying Lemma 3.5. For each $i\in[n]$, so as to introduce $r_0\cdot\mathbf{y}[i]$ in $\mathbf{k}_{\mathsf{ipfe}}$ we will have to use 5 auxiliary hidden vectors in $\mathbf{c}_j$

for $(\tau\Delta\mathbf{x}[i],\ 0,\ \tau z_j \cdot \Delta\mathbf{x}[i], 0, 0)_{\mathbf{F}}$ for all $j \in \mathsf{S}$ and $z_j \xleftarrow{\$} \mathbb{Z}_q$. This explains why we need n more coordinates in $(\mathbf{F}, \mathbf{F}^*)$ to accommodate n values $(\tau z_j \cdot \Delta\mathbf{x}[i], a'_j\mathbf{y}[i]/z_j)$ in $(\mathbf{c}_j, \mathbf{k}^*_j) \in \mathbf{F} \times \mathbf{F}^*$, for each j, and 4 more auxiliary hidden vectors, besides the 3 vectors used in real life. The same goes for the need of $n+3$ basis vectors in $(\mathbf{H}, \mathbf{H}^*)$.

We remark that Lemma 3.5 only helps us mask the ℓ-th key components $\mathbf{k}^*_{\ell,\mathsf{ipfe}}$ by another random labeling based on $a'_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$. However, after all the masks $(a'_{\ell,0} \cdot \mathbf{y}[i])_{i\in[n]}$ are in the vector $\mathbf{k}^*_{\ell,\mathsf{ipfe}}$, thanks to the fact that the product in DPVS will give us $\tau a'_{\ell,0}\langle\Delta\mathbf{x}, \mathbf{y}\rangle$, we can change $(a'_{\ell,0} \cdot \mathbf{y}[i])_{i\in[n]}$ to $(r'_{\ell,0} \cdot \mathbf{y}[i])_{i\in[n]}$ *all at once*. If the access structure in the key is *not* satisfied by the challenge attributes, there does not exist any authorized set in S. In other words, there will exist $j \in \mathsf{S}$ such that $\tau z_j \cdot \Delta\mathbf{x}[i]$ appears in the *unique* challenge ciphertext but z_j is totally hidden in the current ℓ-th key. Thanks to the fact that $(a'_{\ell,j}/z_j)_j$ is perfectly randomized by $(z_j)_j$ from the labeling $(a'_{\ell,j})_j$ of $a'_{\ell,0}$, it implies that $a'_{\ell,0}$ is statistically hidden and $(a'_{\ell,0} \cdot \mathbf{y}[i])_{i\in[n]}$ can be replaced by $(r'_{\ell,0} \cdot \mathbf{y}[i])_{i\in[n]}$ for some uniformly independent random value $r'_{\ell,0} \xleftarrow{\$} \mathbb{Z}_q$. Otherwise, if $\mathbb{A}(\mathsf{S}) = 1$, the security model enforces that $\langle\Delta\mathbf{x}, \mathbf{y}\rangle = 0$ and the result does not depend on $a'_{\ell,0}$ anymore. In either case, changing from $(a'_{\ell,0} \cdot \mathbf{y}[i])_{i\in[n]}$ to $(r'_{\ell,0} \cdot \mathbf{y}[i])_{i\in[n]}$ can be justified. We have to perform this masking by $r'_{\ell,0}\mathbf{y}[i]$ for only one key at a time; Or else two different ℓ-th and k-th keys containing the randomized labels $(a'_{\ell,j}/z_j, a'_{k,j}/z_j)_j$ might mutually leak information about z_j for some j embedded in $(\tau z_j)_j$ of the unique **LoR** ciphertext.

The last step is to virtually modify (S, U) in the master secret key msk so that the challenge ciphertext is now encrypting $\mathbf{x}^*_0[i]$ and is no longer depending on b. The new (S', U') will respect the relation dictated in pk, which is known by the adversary. For any functional key corresponding to $\mathbf{y}_\ell$ such that $\langle\Delta\mathbf{x}, \mathbf{y}_\ell\rangle = 0$, simulating the key using (S, U) is identical to doing so using (S', U'). On the other hand, in the case where $\langle\Delta\mathbf{x}, \mathbf{y}_\ell\rangle \neq 0$, simulating the functional key for $\mathbf{y}_\ell$ using (S, U) introduces errors when we update (S, U) to (S', U'). These errors can be corrected using the random mask from previous steps, under the SXDH assumption, to make the keys be in the correct form w.r.t (S', U'). Finally, because the challenge ciphertext no longer depends on b, the advantage becomes 0 and we can conclude.

4.4 Multi-Client Functional Encryption for Inner-Product with Fine-Grained Access Control via LSSS

First of all, we define and give the model of security for *multi-client functional encryption with fine-grained access control* in Section 4.4.1. We then present our main contribution by extending our FE scheme in Section 4.3 from the single-client setting to the multi-client setting in Section 4.4.2, for the functionality class to evaluate inner-products under access control by linear secret-sharing schemes. Theorem 4.8 proves its adaptive security. Finally, in Section 4.4.4 we discuss further our construction and revisit the MIFE regime for comparison with [ACGU20].

4.4.1 Definitions

We extend the notion of functional encryption with fine-grained access control to the multi-client setting. The access control is defined via a relation $\mathsf{Rel} : \mathsf{AC\text{-}K} \times \mathsf{AC\text{-}Ct}_1 \times \cdots \times \mathsf{AC\text{-}Ct}_n \to \{0,1\}$, for some sets $\mathsf{AC\text{-}Ct}_1, \ldots, \mathsf{AC\text{-}Ct}_n$ and $\mathsf{AC\text{-}K}$. A plaintext for client i consists of $(\mathsf{ac\text{-}ct}_i, x_i) \in \mathsf{AC\text{-}Ct}_i \times \mathcal{D}_\lambda$, whose corresponding ciphertext can be decrypted to $F_\lambda(x)$ using the functional key $\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}$ for $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$ if and only if $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$.

Definition 4.2 (Multi-client functional encryption with fine-grained access control). *A* multi-client functional encryption (MCFE) scheme with fine-grained access control *for the functionality class* $\mathcal{F} \times \mathsf{AC\text{-}K}$ *consists of four algorithms* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$*:*

$\mathsf{Setup}(1^\lambda)$**:** *Given as input a security parameter* λ*, output a master secret key* msk *and* $n = n(\lambda)$ *encryption keys* $(\mathsf{ek}_i)_{i\in[n]}$ *where* $n : \mathbb{N} \to \mathbb{N}$ *is a function.*

$\mathsf{Extract}(\mathsf{msk}, F_\lambda, \mathsf{ac\text{-}k})$**:** *Given* $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$*, a function description* $F_\lambda \in \mathcal{F}$*, and the master secret key* msk*, output a decryption key* $\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}$*.*

$\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$**:** *Given as inputs* $\mathsf{ac\text{-}ct}_i \in \mathsf{AC\text{-}Ct}_i$*, an encryption key* ek_i*, a message* $x_i \in \mathcal{D}_\lambda$*, and a tag* tag*, output a ciphertext* $\mathsf{ct}_{\mathsf{tag},i}$*.*

$\mathsf{Dec}(\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}, \mathbf{c})$**:** *Given the decryption key* $\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}$ *and a vector of ciphertexts* $\mathbf{c} := (\mathsf{ct}_{\mathsf{tag},i})_i$ *of length* n*, output an element in* $\mathcal{R}_\lambda$ *or an invalid symbol* $\perp$*.*

Correctness. For sufficiently large $\lambda \in \mathbb{N}$, for all $(\mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda)$, $(F_\lambda, \mathsf{ac\text{-}k}) \in \mathcal{F} \times \mathsf{AC\text{-}K}$ and $\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}} \leftarrow \mathsf{Extract}(\mathsf{msk}, F_\lambda, \mathsf{ac\text{-}k})$, for all tag and $(\mathsf{ac\text{-}ct}_i)_i$ satisfying $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$, for all $(x_i)_{i\in[n]} \in \mathcal{D}_\lambda^n$, if $F_\lambda(x_1, \ldots, x_n) \neq \perp$, the following holds with overwhelming probability:

$$\mathsf{Dec}\left(\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}, (\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{ac\text{-}ct}_i))_{i\in[n]}\right) = F_\lambda(x_1, \ldots, x_n)$$

where $F_\lambda : \mathcal{D}_\lambda^n \to \mathcal{R}_\lambda$ and the probability is taken over the coins of algorithm.

Security. We define an indistinguishability-based security notion taking into account the attribute-based access control as well as the possibility of corruption among multiple clients. We define the *admissibility* of an adversary $\mathcal{A}$ in the security game against $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$. Intuitively, we consider only admissible adversaries who do not win our security game in a trivial manner as well as other meaningful restrictions in the multi-client setting. The admissibility additionally takes into account the satisfiability of the relation for access control, which also complicates the way we model the security notion. In the plain setting, interested readers can refer to [CDG+18a] or [LT19] for more details.

Definition 4.3 (Admissible adversaries). *Let* $\mathcal{A}$ *be a* ppt *adversary and let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ *be an MCFE scheme with fine-grained access control for the functionality class* $\mathcal{F} \times \mathsf{AC\text{-}K}$*.*

In the security game given in Figure 4.3 for $\mathcal{A}$ considering $\mathcal{E}$, let the sets $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ be the sets of corrupted clients, functional key queries, and honest clients, in that order. We say that $\mathcal{A}$ is NOT admissible w.r.t $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ *if any of the following conditions holds:*

1. *There exist two different partial ciphertexts for $x_i^{(b)} \neq x_i^{(b)\prime}$, for some $b \in \{0,1\}$, under one challenge tag* tag *that is queried to* **LoR**.
2. *There exist a tag* tag *and $i, j \in \mathcal{H}$ such that $i \neq j$, there exists a query $(i, x_i^{(0)}, x_i^{(1)}, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ to* **LoR** *but there exist no query $(j, x_j^{(0)}, x_j^{(1)}, \mathsf{tag}, \mathsf{ac\text{-}ct}_j)$ to* **LoR**.
3. *There exists $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ for $i \in [n]$, a function $F \in \mathcal{F}$, and $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$ such that*
   - *We have $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$ and $(F, \mathsf{ac\text{-}k}) \in \mathcal{Q}$.*
   - *For all $i \in \mathcal{H}$, there exists a query $(i, x_i^{(0)}, x_i^{(1)}, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ to* **LoR** *for $(x_i^{(0)}, x_i^{(1)})$.*
   - *For all $i \in \mathcal{C}$, it holds that $x_i^{(0)} = x_i^{(1)}$.*
   - *It holds that $F((x_i^{(0)})_i) \neq F((x_i^{(1)})_i)$.*

Otherwise, we say that $\mathcal{A}$ is admissible w.r.t $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$.

Remark 4.4. As in the plain MCFE with no attribute-based access control in [CDG+18a, LT19], we will consider security with no repetitions (*cf.* condition 1), *i.e.* the adversary cannot query **Enc** nor **LoR** for multiple ciphertexts under the same $(i, \mathsf{tag}, \mathsf{ac\text{-}ct}_i)$. Moreover, the adversary is not allowed to query the encryption oracle **Enc** for ciphertexts under the challenge tag^* that was previously queried to **LoR**. The intuition of this restriction is to prevent trivial attacks where, by querying for ciphertexts under tag^*, the adversary can combine them with the challenge ciphertext under the same tag^* to learn much more information about the challenge bit b and win the game. In addition, for every honest clients i, there must be a ciphertext query to **LoR** under the challenge $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$. That is, we do not take into account the scenario where only partial (in terms of honest clients) challenge ciphertext is queried to **LoR**. We can relax this condition and allow partial challenge ciphertexts by adding a layer of *All-or-Nothing Encapsulation* (AoNE). The AoNE encapsulates the partial components from clients and guarantees that all encapsulated components can be decapsulated if and only if all components are gathered, otherwise the original information remain hidden. The work by Chotard *et al.* [CDSG+20] presents constructions for AoNE in the prime-order (asymmetric) bilinear groups compatible with our current setting. In the MIFE realm, the work of [ACGU20] considers the similar restriction and expects all honest slots $i \in [n]$ are queried to **LoR**.

Remark 4.5. Our syntax and model of MCFE with fine-grained access control require that in order to combine the ciphertext components, they must be encrypted under the same tag and the same set of attributes. One can aim for a more flexible notion in which each client i can encrypt their ciphertext component under a different $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$. However, this creates a much more intricate situation and we have to take into account non-trivial attacks where two different functional keys, whose policies are satisfied by different subsets of clients, may be combined to evaluate the underlying plaintext components of the union of the foregoing subsets. By hashing the tags and attributes during encryption, our concrete constructions enforce the same set of attributes embedded in the ciphertext components. In Section 4.4.4, we discuss how to relax the constraint and achieve the flexible notion where each client i can use a different $(\mathsf{tag}, \mathsf{ac\text{-}ct}_i)$ and hash only tag. As a result, this more flexible MCFE scheme in the RO model can be morphed into an MIFE scheme in the *standard* model by fixing a public tag and publishing its hash (see Section 4.4.4). The security of the MIFE is deduced from that of the MCFE under this fixed and hashed pubic tag.

We are now ready to give the definition for the indistinguishability-based security.

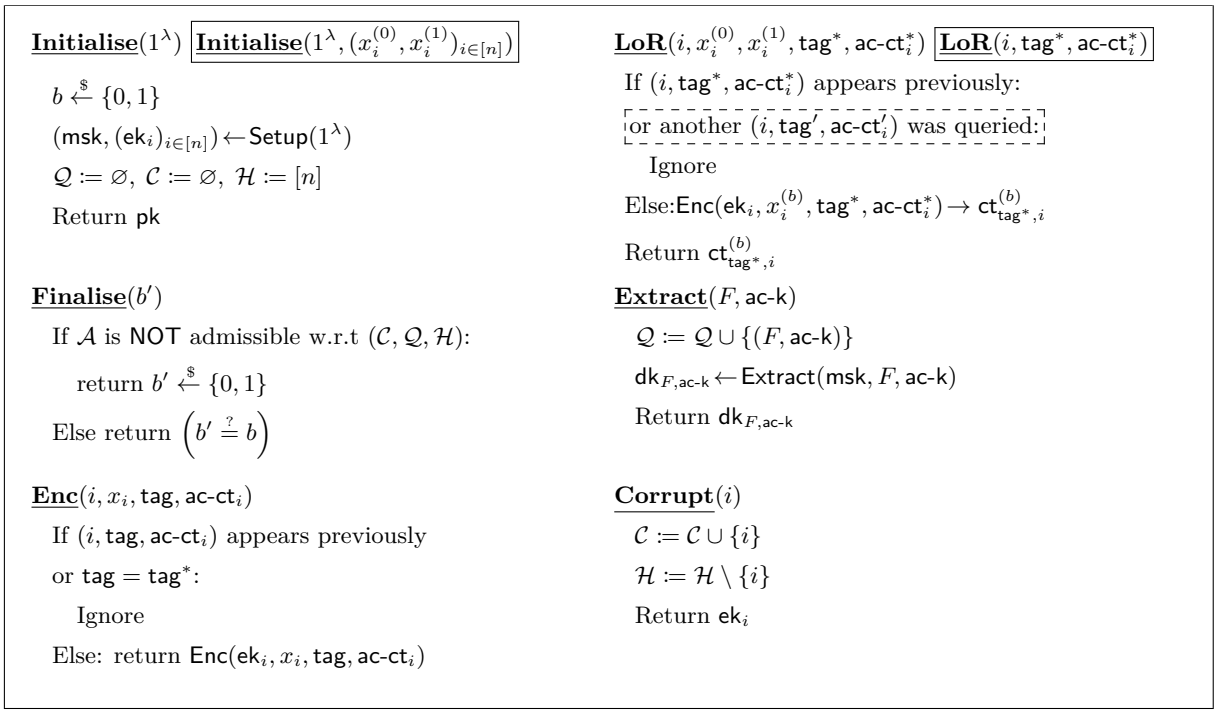


Figure 4.3: The security games $\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)$, $\boxed{\mathsf{Expr}^{\mathsf{mc\text{-}sel\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)}$ and $\boxed{\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}1chal\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)}$ for Definition 4.6

Definition 4.6 (IND-security for MCFE with fine-grained access control). *An MCFE scheme with fine-grained access control* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ *for the functionality class* $\mathcal{F} \times \mathsf{AC\text{-}K}$ *is* IND-secure *if for all* ppt *adversaries* $\mathcal{A}$*, and for all sufficiently large* $\lambda \in \mathbb{N}$*, the following probability is negligible*

$$\mathsf{Adv}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) := \left|\Pr[\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| \ .$$

The game $\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathsf{AC\text{-}K},\mathcal{A}}(1^\lambda)$ *is depicted in Figure 4.3. The probability is taken over the random coins of* $\mathcal{A}$ *and the algorithms. Weaker notions include* selectively IND-secure *and* one-time IND-secure *and are defined using the corresponding games in Figure 4.3.*

Lemma 4.7 allows us to concentrate on the notion of one-time IND-security for our construction.

Lemma 4.7. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ *for the function class* $\mathcal{F} \times \mathsf{AC\text{-}K}$ *be an MCFE scheme with fine-grained access control. If* $\mathcal{E}$ *is one-time IND-secure, then* $\mathcal{E}$ *is IND-secure.*

4.4.2 Construction

This section presents a multi-client FE scheme with fine-grained access control, as defined in Section 4.4.1. We are in the bilinear group $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are written additively. In our concrete construction, the functionality class of interests is $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$ and $\mathcal{F}^{\mathsf{IP}}$ contains $F_\mathbf{y} : \left(\mathbb{Z}_q^*\right)^n \to \mathbb{Z}_q$ that is defined as $F_\mathbf{y}(\mathbf{x}) := \langle \mathbf{x}, \mathbf{y} \rangle$. The access control is given by $\mathsf{Rel} : \mathsf{LSSS} \times \left(\prod_{i=1}^n 2^{\mathsf{Att}}\right) \to \{0,1\}$, where $\mathsf{Rel}(\mathbb{A}, (\mathsf{S}_i)_i) = \prod_i \mathbb{A}(\mathsf{S}_i)$, the class LSSS contains Linear Secret Sharing Schemes over Att, and 2^{Att} denotes the superset of an attribute space $\mathsf{Att} \subseteq \mathbb{Z}_q$. Our constructions are key-policy, where $\mathbb{A}$ is embedded in the key and S is specified in the ciphertext. The tag space Tag contains the tags that accompany plaintext components at the time of encryption.

We also need a full domain hash function $\mathsf{H} : \mathsf{Tag} \times 2^{\mathsf{Att}} \to \mathbb{G}_1^2$, where Tag denotes the set of tags and 2^{Att} contains the subsets of attributes of Att. The details of our construction is given in Figure 4.4. We remark that currently all clients $i \in [n]$ must use the same S for encrypting their inputs x_i, because S is hashed together with tag by H. Section 4.4.4 presents another construction that relaxes the matching condition on S and H then receives only tag as inputs. We note that the *duplicate-and-compress* technique is used by putting the vectors $\{(\mathbf{c}_{i,j}, \mathbf{k}_{i,j})_j\}$ in the same pair of dual bases $(\mathbf{F}, \mathbf{F}^*)$ for all client $i \in [n]$, meanwhile each pair of vectors $(\mathbf{c}_{i,\mathsf{ipfe}}, \mathbf{k}_{i,\mathsf{ipfe}})$ is put in bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for each client $i \in [n]$. In the proof of Theorem 4.8 we detail how the basis changes in [NPP22a, NPP22b, Lemma 4], or its more generalized version of Lemma 3.5, can be done in parallel for $(\mathbf{H}_i, \mathbf{H}_i^*), (\mathbf{F}, \mathbf{F}^*)$ for all $i \in [n]$. The *correctness* of the scheme is verified by:

$$\begin{aligned}
[\![\mathsf{out}]\!]_\mathsf{t} &= \sum_{i=1}^{n} \left(\left(\sum_{j \in A} \mathbf{c}_{i,j} \times (c_j \cdot \mathbf{k}_{i,j}) \right) - (\mathbf{c}_{i,\mathsf{ipfe}} \times \mathbf{k}_{i,\mathsf{ipfe}}) + \mathbf{e}(\mathbf{t}_i, \mathbf{m}_i) \right) \\
&= \sum_{i=1}^{n} \left([\![\psi_i a_0 z]\!]_\mathsf{t} - [\![\omega p_i \cdot \langle S, \mathbf{y} \rangle + \omega' p_i \cdot \langle U, \mathbf{y} \rangle + \psi_i a_0 z]\!]_\mathsf{t} + [\![(\omega s_i + \omega' u_i + x_i) y_i]\!]_\mathsf{t} \right) \\
&= [\![\langle \mathbf{x}, \mathbf{y} \rangle]\!]_\mathsf{t}
\end{aligned}$$

4.4.3 Adaptive Security

We now present the main ideas of the adaptive proof for the multi-client construction described in Section 4.4.2, the detailed proof is presented in the full version [NPP22a, Appendix B.4]. A high-level intuition can be revisited in Section 4.2, as well as in the main ideas of the proof that are given below.

Theorem 4.8. *Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ be a multi-client IPFE scheme with fine-grained access control via LSSS for the functionality class $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$, constructed in Section 4.4.2 in a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$. Then, $\mathcal{E}$ is one-time IND-secure if the SXDH assumption holds for $\mathbb{G}_1$ and $\mathbb{G}_2$. More specifically, for $\lambda \in \mathbb{Z}$ and for any adversary $\mathcal{A}$, let K denote the total number of functional key queries, P denote the total number of attributes used by $\mathcal{A}$, and Q denote the maximum number of random oracle (RO) queries. We have the following bound:*

$$\mathsf{Adv}^{\mathsf{mc\text{-}ind\text{-}1chal\text{-}cpa}}_{\mathcal{E}, \mathcal{F}^{\mathsf{IP}}, \mathsf{LSSS}, \mathcal{A}}(1^\lambda) \leq (2KP \cdot (6P + 3) + 2K + 2Q + 5) \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1, \mathbb{G}_2}(1^\lambda)$$

where $\mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1, \mathbb{G}_2}(1^\lambda)$ denotes the maximum advantage over ppt adversaries against the SXDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ set up with parameter λ.

By combining with Lemma 4.7, we have the following Corollary:

Corollary 4.9. *Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ be a multi-client IPFE scheme with fine-grained access control via LSSS, for the functionality class $\mathcal{F}^{\mathsf{IP}} \times \mathsf{LSSS}$, constructed in Section 4.4.2 in a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$. Then, $\mathcal{E}$ is IND-secure if the SXDH assumption holds for $\mathbb{G}_1$ and $\mathbb{G}_2$.*

Proof (of Theorem 4.8 - Main ideas). Recall that in the security proof for single-client adaptive security (Theorem 4.1) we switch the ℓ-th functional key to semi-functional by augmenting the dimension of the dual bases so that the challenge ciphertext is masked by $\tau \Delta\mathbf{x}[i]$, facing the mask $r_0^{(\ell)} \mathbf{y}^{(\ell)}[i]$ in the corresponding coordinate of the ℓ-th key and $\tau, r_0^{(\ell)} \xleftarrow{\$} \mathbb{Z}_q$ where $\Delta\mathbf{x} := \mathbf{x}^{(1)} - \mathbf{x}^{(0)}$. Afterwards, when doing the product of vectors in the dual bases, there will exist the quantity $\sum_{i=1}^{n} \tau r_0^{(\ell)} \Delta\mathbf{x}[i] \mathbf{y}^{(\ell)}[i] = \tau r_0^{(\ell)} \langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle$, which is non-zero when $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \neq 0$. The dual bases now must have dimension at least n in order to accommodate all the n terms $\Delta\mathbf{x}[i]\mathbf{y}[i]$. However, in the multi-client setting, we are already using n different dual basis pairs $(\mathbf{H}_i, \mathbf{H}_i^*)$ for n clients

$\underline{\mathsf{Setup}(1^\lambda)}$**:** Choose $n+1$ pairs of dual orthogonal bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for $i \in [n]$ and $(\mathbf{F}, \mathbf{F}^*)$where $(\mathbf{H}_i, \mathbf{H}_i^*)$ is a pair of dual bases for $(\mathbb{G}_1^4, \mathbb{G}_2^4)$ and $(\mathbf{F}, \mathbf{F}^*)$ is a pair of dual bases for $(\mathbb{G}_1^8, \mathbb{G}_2^8)$. We denote the basis changing matrices for $(\mathbf{F}, \mathbf{F}^*), (\mathbf{H}, \mathbf{H}_i^*)$ as $(F, F' \coloneqq (F^{-1})^\top), (H_i, H_i' \coloneqq (H_i^{-1})^\top)$ respectively (see Section 3.3 for basis changes in DPVS):

$$(\mathbf{H}_i = H_i \cdot \mathbf{T};\ \mathbf{H}_i^* = H_i' \cdot \mathbf{T}^*)_{i\in[n]} \quad (\mathbf{F} = F \cdot \mathbf{W};\ \mathbf{F}^* = F' \cdot \mathbf{W}^*)$$

where $H_i, H_i' \in \mathbb{Z}_q^{4\times 4}, F, F' \in \mathbb{Z}_q^{8\times 8}$ and $(\mathbf{T} = [\![I_4]\!]_1, \mathbf{T}^* = [\![I_4]\!]_2)$, $(\mathbf{W} = [\![I_8]\!]_1, \mathbf{W}^* = [\![I_8]\!]_2)$ are canonical bases of $(\mathbb{G}_1^4, \mathbb{G}_2^4)$, $(\mathbb{G}_1^8, \mathbb{G}_2^8)$ respectively, for identity matrices I_4 and I_8. We recall that in the multi-client setting the scheme must be a private key encryption scheme. For each $i \in [n]$, we write

$$\mathbf{H}_i = (\mathbf{h}_{i,1}, \mathbf{h}_{i,2}, \mathbf{h}_{i,3}, \mathbf{h}_{i,4}) \qquad \mathbf{H}_i^* = (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*, \mathbf{h}_{i,4}^*)$$
$$\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{f}_4, \mathbf{f}_5, \mathbf{f}_6, \mathbf{f}_7, \mathbf{f}_8) \qquad \mathbf{F}^* = (\mathbf{f}_1^*, \mathbf{f}_2^*, \mathbf{f}_3^*, \mathbf{f}_4^*, \mathbf{f}_5^*, \mathbf{f}_6^*, \mathbf{f}_7^*, \mathbf{f}_8^*)$$

and sample $\mu \xleftarrow{\$} \mathbb{Z}_q^*, S, U, \xleftarrow{\$} (\mathbb{Z}_q^*)^n$ and write $S = (s_1, \ldots, s_n)$, $U = (u_1, \ldots, u_n)$. Perform an n-out-of-n secret sharing on 1, that is, choose $p_i \in \mathbb{Z}_q$ such that $1 = p_1 + \cdots + p_n$. Output the master secret key and the encryption keys as

$$\begin{cases} \mathsf{msk} \coloneqq (S,\ U,\ \mathbf{f}_1^*,\ \mathbf{f}_2^*,\ \mathbf{f}_3^*,\ (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*)_{i\in[n]}) \\ \mathsf{ek}_i \coloneqq (s_i,\ u_i,\ p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ \mathbf{f}_1,\ \mathbf{f}_2,\ \mathbf{f}_3) \text{ for } i \in [n] \end{cases}$$

where $H_i^{(k)}$ denotes the k-th row of H_i.

$\underline{\mathsf{Extract}(\mathsf{msk}, \mathbb{A}, \mathbf{y} \in \mathbb{Z}_q^n)}$**:** Let $\mathbb{A}$ be an LSSS-realizable monotone access structure over a set of attributes $\mathsf{Att} \subseteq \mathbb{Z}_q$. First, sample $a_0 \xleftarrow{\$} \mathbb{Z}_q$ and run the labeling algorithm $\Lambda_{a_0}(\mathbb{A})$ (see Equation (3.3)) to obtain the labels $(a_j)_j$ where j runs over the attributes in Att. In the end, it holds that $a_0 = \sum_{j\in A} c_j \cdot a_j$ where j runs over an authorized set $A \in \mathbb{A}$ and $\mathbf{c} = (c_j)_j$ is the reconstruction vector from LSSS w.r.t A. We denote by $\mathsf{List\text{-}Att}(\mathbb{A})$ the list of attributes appearing in $\mathbb{A}$, with possible repetitions. Parse $\mathsf{msk} = (S,\ U,\ \mathbf{f}_1^*,\ \mathbf{f}_2^*,\ \mathbf{f}_3^*,\ (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*)_{i\in[n]})$ and write $\mathbf{y} = (y_1, \ldots, y_n)$. For each $i \in [n]$, compute $\mathbf{m}_i \coloneqq [\![y_i]\!]_2$ and

$$\mathbf{k}_{i,j} = (\pi_{i,j} \cdot (j, 1),\ a_j \cdot z,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})$$
$$\mathbf{k}_{i,\mathsf{ipfe}} \coloneqq (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_0 \cdot z,\ 0)_{\mathbf{H}_i^*}$$

where $z, \pi_{i,j} \xleftarrow{\$} \mathbb{Z}_q$. Output $\mathsf{dk}_{\mathbb{A},\mathbf{y}} \coloneqq \left((\mathbf{k}_{i,j})_{i,j}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]}\right)$.

$\underline{\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{S})}$**:** Parse $\mathsf{ek}_i \coloneqq (s_i,\ u_i,\ p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ \mathbf{f}_1,\ \mathbf{f}_2,\ \mathbf{f}_3)$ and $\mathsf{S} \subseteq \mathsf{Att} \subseteq \mathbb{Z}_q$ as the set of attributes, compute $\mathsf{H}(\mathsf{tag}, \mathsf{S}) \to ([\![\omega]\!]_1, [\![\omega']\!]_1) \in \mathbb{G}_1^2$ and sample $\psi_i \xleftarrow{\$} \mathbb{Z}_q$. Use $p_i H_i^{(1)}$ and $p_i H_i^{(2)}$ to compute

$$p_i H_i^{(1)} \cdot [\![\omega]\!]_1 + p_i H_i^{(2)} \cdot [\![\omega']\!]_1 = p_i \cdot \left(\omega H_i^{(1)} \cdot g_1 + \omega' H_i^{(2)} \cdot g_1\right) = p_i \cdot (\omega \mathbf{h}_{i,1} + \omega' \mathbf{h}_{i,2}) \ .$$

For each $j \in \mathsf{S}$, compute

$$\mathbf{c}_{i,j} = \sigma_{i,j} \cdot \mathbf{f}_1 - j \cdot \sigma_{i,j} \cdot \mathbf{f}_2 + \psi_i \cdot \mathbf{f}_3 = (\sigma_{i,j} \cdot (1, -j),\ \psi_i,\ 0,\ 0,\ 0,\ 0,\ 0)_{\mathbf{F}}$$

where $\sigma_{i,j} \xleftarrow{\$} \mathbb{Z}_q$. Finally, compute

$$\mathbf{t}_i = s_i \cdot [\![\omega]\!]_1 + u_i \cdot [\![\omega']\!]_1 + [\![x_i]\!]_1 = [\![\omega \cdot s_i + \omega' \cdot u_i + x_i]\!]_1$$
$$\mathbf{c}_{i,\mathsf{ipfe}} = p_i \cdot (\omega \cdot \mathbf{h}_{i,1} + \omega' \cdot \mathbf{h}_{i,2}) + \psi_i \cdot \mathbf{h}_{i,3} = (\omega p_i,\ \omega' p_i,\ \psi_i,\ 0)_{\mathbf{H}_i}$$

and output $\mathsf{ct}_{\mathsf{tag},i} \coloneqq \left((\mathbf{c}_{i,j})_j, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}}\right)$.

$\underline{\mathsf{Dec}(\mathsf{dk}_{\mathbb{A},\mathbf{y}}, \mathbf{c} \coloneqq (\mathsf{ct}_{\mathsf{tag},i}))}$**:** Parse $\mathsf{ct}_{\mathsf{tag},i} = ((\mathbf{c}_{i,j})_{j\in\mathsf{S}}, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}})$ and $\mathsf{dk}_{\mathbb{A},\mathbf{y}} \coloneqq ((\mathbf{k}_{i,j})_{i\in[n], j\in\mathsf{List\text{-}Att}(\mathbb{A})}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]})$. If there exists $A \subseteq \mathsf{S}$ and $A \in \mathbb{A}$, then compute the reconstruction vector $\mathbf{c} = (c_j)_j$ of the LSSS for A and

$$[\![\mathsf{out}]\!]_\mathsf{t} = \sum_{i=1}^{n} \left(\left(\sum_{j\in A} \mathbf{c}_{i,j} \times (c_j \cdot \mathbf{k}_{i,j}) \right) - (\mathbf{c}_{i,\mathsf{ipfe}} \times \mathbf{k}_{i,\mathsf{ipfe}}) + \mathbf{e}(\mathbf{t}_i, \mathbf{m}_i) \right)$$

Finally, compute the discrete logarithm and output the small value out.

Figure 4.4: The construction for multi-client IPFE with fine-grained access control via LSSS from Section 4.4.2.

and the correctness of the construction in Section 4.4.2 makes sure that only when gathering all n ciphertext parts can we decrypt to obtain the inner product. Therefore, it suffices to introduce only $\tau_i \Delta\mathbf{x}[i]$ in the component $\mathbf{c}_{i,\mathsf{ipfe}}$ returned from **LoR** of client i and only $r_{i,0}^{(\ell)}\mathbf{y}^{(\ell)}[i]$ in the corresponding key component $\mathbf{k}^*_{i,\mathsf{ipfe}}$, while duplicating the pair of bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for each $i \in [n]$. Indeed, this is also the best we can do because a client i is not supposed to know other inputs $\mathbf{x}^{(b)}[j]$ of other clients j, where $b \xleftarrow{\$} \{0,1\}$ is the challenge bit. At the same time, we compress the components of the access control part $(\mathbf{c}_{i,j})_j, (\mathbf{k}^*_{i,j})_j$ into the same pair of bases $(\mathbf{F}, \mathbf{F}^*)$ for all clients i. We refer to the introduction for more intuition on this duplicate-and-compress process.

There are some further technical tweaks to be done when applying Lemma 3.5 (only its simpler version as in [NPP22a, NPP22b, Lemma 4] is needed). First of all, we need the factors $\tau_i, r_{i,0}^{(\ell)}$ to be the same, for the grouping later when doing products of vectors in DPVS. This can be done by using the same $\tau_i = \tau$ for all i and during the basis change to mask the ciphertext component there will be a factor $\Delta\mathbf{x}[i]$. Our argument to introduce $r_{i,0}^{(\ell)}$ in fact does not depend on i and therefore we can use the same $r_{i,0}^{(\ell)} = r_0^{(\ell)}$ for all i as well. One might wonder if the dependence of the masks still relies on $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle$ because the adversary is not supposed to query **LoR** for corrupted clients and we can only introduce the masks in the vector components of honest i. As a result, the product of vectors in the dual bases in the end will have $\sum_{i\in\mathcal{H}} \tau r_0^{(\ell)} \Delta\mathbf{x}[i]\mathbf{y}^{(\ell)}[i]$. However, the security model imposes that for all corrupted i, the challenge message satisfies $\mathbf{x}^{(1)}[i] = \mathbf{x}^{(0)}[i]$ and consequently, $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle = 0$ if and only if $\sum_{i\in\mathcal{H}} \Delta\mathbf{x}[i]\mathbf{y}^{(\ell)}[i] = 0$. This implies that the mask $\tau r_0^{(\ell)} \sum_{i\in\mathcal{H}} \Delta\mathbf{x}[i]\mathbf{y}^{(\ell)}[i]$ persists only when $\langle \Delta\mathbf{x}, \mathbf{y}^{(\ell)} \rangle \neq 0$, which is our goal. The masking of ciphertext and key components results from the application of Lemma 3.5 as we are in the adaptive setting and not knowing what policy the ciphertext's attributes will satisfy. The lemma will mask all vectors $\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$ with $a_0'^{(\ell)} \xleftarrow{\$} \mathbb{Z}_q$, using which we perform a random labeling, and under the constraint that all clients i use the same S, the mask $a_0'^{(\ell)}$ will either appear for all i or neither. This enables us to replace it with $r_0^{(\ell)}$. We recall that currently the constraint on using the same S for all i is guaranteed by hashing $(\mathsf{tag}, \mathsf{S})$ together. The more complicated and flexible case with possibly different S_i for each i is discussed in Section 4.4.4. The application of Lemma 3.5 needs some auxiliary vectors in the dual bases $(\mathbf{F}, \mathbf{F}^*)$, which are not needed in the real usage of the scheme. Following the terminology of Okamoto-Takashima [OT12b], those auxiliary vectors form a *hidden* part of the bases.

The final steps are to change (s_i, u_i) in the challenge ciphertext to (s_i', u_i') so that the ciphertext from **LoR** is encrypting $\mathbf{x}^{(0)}$ instead of $\mathbf{x}^{(b)}$ by solving a linear system for $(\Delta S, \Delta U)$ depending on $\mathbf{x}^{(b)} - \mathbf{x}^{(0)}$. We stress that the simulation of corrupted keys can still be done using (s_i, u_i) regardless of the order of **LoR** query, under the admissibility from condition 3 in Definition 4.3 that requires $\Delta\mathbf{x}[i] = \mathbf{x}^{(1)}[i] - \mathbf{x}^{(0)}[i] = 0$ if i is corrupted.

In the case of $\langle \Delta\mathbf{x}, \mathbf{y} \rangle \neq 0$, which then implies $\mathbb{A}(\mathsf{S}) \neq 0$, the functional key queries that are simulated using $(\langle S, \mathbf{y} \rangle, \langle U, \mathbf{y} \rangle)$ are computaionally indistinguishable from the ones in correct forms using $(\langle S', \mathbf{y} \rangle, \langle U', \mathbf{y} \rangle)$, under the SXDH assumption. However, the situation is more complicated than the single-client construction because the oracle **Enc** is using (s_i, u_i) as well. In order to be able to perform the correction step on the functional key, we have to program the full-domain hash function, which is modeled as an RO, such that for all queries $(\mathsf{tag}', \mathsf{S}')$ different from the challenge $(\mathsf{tag}, \mathsf{S})$, the value $\mathsf{H}(\mathsf{tag}', \mathsf{S}')$ belongs to $\mathrm{span}([\![(1,\ \mu)]\!]_1) \subseteq \mathbb{G}_1^2$, for $\mu \xleftarrow{\$} \mathbb{Z}_q$. For the challenge $(\mathsf{tag}, \mathsf{S})$, the value $\mathsf{H}(\mathsf{tag}, \mathsf{S})$ remains a pair of random group elements. The main reason behind this is that our correction step requires $\mathsf{H}(\mathsf{tag}', \mathsf{S}')$ belongs to $\mathrm{span}([\![(1,\ \mu)]\!]_1)$ so that it will not affect the normal ciphertext returned from **Enc**. This implies a linear relation between $\Delta S := S' - S$ and $\Delta U := U' - U$. However, if we put $\mathsf{H}(\mathsf{tag}, \mathsf{S})$ on the line $\mathrm{span}([\![(1,\ \mu)]\!]_1)$ as well, then the intention to switch from $\mathbf{x}^{(0)}$ to $\mathbf{x}^{(b)}$ in the ciphertext from **LoR** will create another linear relation, which reduces significantly the degree of freedom to choose $(\Delta S, \Delta U)$ in order to make the simulation successful. In the end, the challenge ciphertext no longer depends on b and the advantage becomes 0, concluding the proof.

4.4.4 Revisiting MIFE in the Standard Model

We recall that currently our MCFE scheme from Section 4.4.2 enforces the same $(\mathsf{tag}, \mathsf{S})$ when encrypting for all client $i \in [n]$, by hashing them using the full-domain hash function that is modeled as an RO in the security proof. In practice, this could render a significant cost for synchronisation among clients so as to agree on tag *and* the attributes S at the time of encryption. In addition, by fixing one public tag, one can only obtain an MIFE scheme whose security can be proven in the ROM because we still need the random oracle to process S.

If we allow different $(\mathsf{tag}, \mathsf{S}_i)$ for each client i and during encryption the input for hashing depends only on tag, *i.e.* $[\![(\omega_{\mathsf{tag}}, \omega'_{\mathsf{tag}})]\!]_1 \leftarrow \mathsf{H}(\mathsf{tag})$, there is a mix-and-match attack among functional keys that has to be considered. More precisely, suppose for two clients $\{1, 2\}$ encrypting $\mathbf{x} = (x_1, x_2)$ under different sets $(\mathsf{S}_1, \mathsf{S}_2)$ of attributes, the ℓ-th and ℓ'-th key queries have access structures $\mathbb{A}$ and $\mathbb{A}'$ where $\mathbb{A}(\mathsf{S}_1) = \mathbb{A}'(\mathsf{S}_2) = 1$ and $\mathbb{A}'(\mathsf{S}_1) = \mathbb{A}(\mathsf{S}_2) = 0$, for the same inner-product with $\mathbf{y} = \mathbf{y}' = (y_1, y_2)$. Neither of these keys should decrypt $x_1y_1 + x_2y_2$ for the sake of security. However, the construction from Figure 4.4 permits an adversary to use the vectors $\{(\mathbf{c}_{1,j})_j, (\mathbf{k}_{1,j})_j, \mathbf{c}_{1,\mathsf{ipfe}}, \mathbf{k}_{1,\mathsf{ipfe}}\}$ to recover $p_1\omega_{\mathsf{tag}}\langle S, \mathbf{y}\rangle + p_1\omega'_{\mathsf{tag}}\langle U, \mathbf{y}\rangle$. Similar computation allows the same adversary to obtain $p_2\omega_{\mathsf{tag}}\langle S, \mathbf{y}\rangle + p_2\omega'_{\mathsf{tag}}\langle U, \mathbf{y}\rangle$ using $\{(\mathbf{c}_{2,j})_j, (\mathbf{k}_{2,j})_j, \mathbf{c}_{2,\mathsf{ipfe}}, \mathbf{k}_{2,\mathsf{ipfe}}\}$. Finally, observing that $p_1 + p_2 = 1$, exploiting the linear combination $y_1 \cdot [\![\omega_{\mathsf{tag}}s_1 + \omega'_{\mathsf{tag}}u_1 + x_1]\!]_1 + y_2 \cdot [\![\omega_{\mathsf{tag}}s_2 + \omega'_{\mathsf{tag}}u_2 + x_2]\!]_1$ permits finding $\langle \mathbf{x}, \mathbf{y}\rangle$. This demonstrates the main reason why we put S as part of the input to the hash function H in our current scheme. The core of the above problem is the fact that the construction from Section 4.4.2 does not prohibit combining different "root" vectors $\mathbf{k}_{1,\mathsf{ipfe}}$ and $\mathbf{k}_{2,\mathsf{ipfe}}$ w.r.t different access structure $\mathbb{A}$ and $\mathbb{A}'$.

In this section we present a solution, with minimal modifications to the scheme, to overcome the need for hashing S. Suppose now we are in the more flexible setting where $[\![(\omega_{\mathsf{tag}}, \omega'_{\mathsf{tag}})]\!]_1 \leftarrow \mathsf{H}(\mathsf{tag})$ during encryption. During setup phase, the pair $(\mathbf{H}_i, \mathbf{H}_i^*)$ is a pair of dual bases for $(\mathbb{G}_1^5, \mathbb{G}_2^5)$, with one more dimension compared to our less flexible construction. The master secret key msk stays the same, while the encryption key ek_i now contains furthermore $\theta_i\mathbf{h}_{i,5}$ for some $\theta_i \xleftarrow{\$} \mathbb{Z}_q$. Given an LSSS-realizable monotone access structure $\mathbb{A}$, the key extraction $\mathsf{Extract}(\mathsf{msk}, \mathbb{A}, \mathbf{y} \in \mathbb{Z}_q^n)$ returns $\mathsf{dk}_{\mathbb{A},\mathbf{y}} := ((\mathbf{k}_{i,j})_{i,j}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]})$. The encryption $\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}, \mathsf{S}_i)$ returns $\mathsf{ct}_{\mathsf{tag},i} := ((\mathbf{c}_{i,j})_j, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}})$ for each $i \in [n]$. There is a new element $d_{\mathbb{A},i}$ appearing in the extra coordinate in $\mathbf{k}_{i,\mathsf{ipfe}}$ for every $i \in [n]$, where $(d_{\mathbb{A},i})_i$ satisfies $\sum_{i=1}^n \theta_i d_{\mathbb{A},i} = 0$, independently chosen for each functional keys. The vectors are essentially the same as in Figure 4.4, except $(\mathbf{c}_{i,\mathsf{ipfe}}, \mathbf{k}_{i,\mathsf{ipfe}})$ for each i as follows:

$$
\begin{aligned}
\mathsf{ek}_i &:= (s_i,\ u_i,\ p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ \theta_i\mathbf{h}_{i,5},\ \mathbf{f}_1,\ \mathbf{f}_2,\ \mathbf{f}_3)\\
\mathsf{msk} &:= (S,\ U,\ (\theta_i)_i,\ \mathbf{f}_1^*,\ \mathbf{f}_2^*,\ \mathbf{f}_3^*,\ (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \mathbf{h}_{i,3}^*)_{i\in[n]})\\
\mathbf{c}_{i,\mathsf{ipfe}} &:= (\omega_{\mathsf{tag}}p_i,\ \omega'_{\mathsf{tag}}p_i,\ \psi_i,\ 0,\ \theta_i)_{\mathbf{H}_i}\\
\mathbf{k}_{i,\mathsf{ipfe}} &:= (\langle S, \mathbf{y}\rangle, \langle U, \mathbf{y}\rangle,\ a_{i,0} \cdot z,\ 0,\ d_{\mathbb{A},i})_{\mathbf{H}_i^*}
\end{aligned}
$$

The decryption calculation stays invariant because $\sum_{i=1}^n \theta_i d_{\mathbb{A},i} = 0$. In retrospection, the mix-and-match attack we gave at the beginning of this section no longer works, because $\mathbb{A} \neq \mathbb{A}'$ and $\theta_1 d_{\mathbb{A},1} + \theta_2 d_{\mathbb{A}',2} = 0$ only with negligible probability over the choices of $\theta_1, \theta_2, d_{\mathbb{A},1}, d_{\mathbb{A}',2} \xleftarrow{\$} \mathbb{Z}_q$, for two independent random families $(d_{\mathbb{A},i})_{i\in[2]}$ and $(d_{\mathbb{A}',i})_{i\in[2]}$. More formally, the security proof for this modified scheme, where we exploit the one extra 5-th coordinate in $(\mathbf{H}_i, \mathbf{H}_i^*)$, can be obtained with recourse to the proof of Theorem 4.8 in Section 4.4.2 under few changes. We sketch the proof and highlight the main differences compared to the less flexible scheme in [NPP22a, NPP22b, Appendix B.5].

Remark 4.10. Adding this new layer of masking increases the ciphertext's size by only a factor linear in n. Moreover, given this new construction where the set of attributes does not involve in the computation of the full-domain hashing anymore, we can obtain an MIFE in the standard model by fixing one tag for every ciphertext. The random oracle can be removed by publishing a

random fixed value corresponding to $\mathsf{H}(\mathsf{tag})$ for encryption. In the end, we obtain an attribute-based MIFE for inner-products with adaptive security in the standard model, where the adversary can make the challenge query to **LoR** at most once for each slot $i \in [n]$. To achieve security w.r.t multiple queries for same slot, we can apply the technique in [CDG+18b] to enhance our construction with repetitions. Finally, we can apply a layer of All-or-Nothing Encapsulation to the ciphertext components of construction in this Section 4.4.4, so as to remove the tradeoff with respect to partial challenge ciphertexts in case of $(\mathsf{tag}, \mathsf{S}_i)$ for different S_i.

Last but far from the least, we recall that in our MCFE security, for each challenge tag^*, condition 1 in Definition 4.3 prohibits the adversary from querying multiple messages or attributes with the same challenge tag, at the same slot i. Therefore, in the MIFE that is obtained in the end, its security does not hold with repetitions at the same slot i either. Generally, the access control complicates the transformation from MCFE to MIFE as pointed out in [ATY23a], regarding especially repetitions on both messages and attributes. The later Chapter 6 of this thesis presents a MCFE with access control that allows repetitions on the messages, but still forbidding repetitions on the attributes.

Chapter

5 Optimal Security Notion for Decentralized Multi-Client Functional Encryption

Chapter content

In the previous Chapter 4, we see various results on the study of advanced function classes of inner products where access control is integrated for some finer management of decryption keys. Throughout our proof overview of Theorem 4.8, for instance, a vital element for a meaningful provable security is the notion of *admissible adversaries*. This notion of *admissibility* is fundamental for the security notion that underlies the very active research on (Decentralized) Multi-Client Functional Encryption (or (D)MCFE), with interesting constructions, especially for the class of inner products. However, the security notions have been evolving over the time. While the target of the adversary in distinguishing ciphertexts is clear, legitimate scenarios that do not consist of trivial attacks on the functionality are less obvious.

In this chapter, we come back to the security notions of FE and wonder whether only trivial attacks are excluded from previous security games. And, unfortunately, this was not the case.

We then propose a stronger security notion, with a large definition of admissible attacks, and prove it is optimal: any extension of the set of admissible attacks is actually a trivial attack on the functionality, and not against the specific scheme. In addition, we show that all the previous constructions are insecure with respect to this new security notion. Eventually, we propose new DMCFE schemes for the class of inner products that provide the new features and achieve this stronger security notion.

5.1 Introduction

Decentralized Multi-Client Functional Encryption. From the previous Chapter 4, it appears that Multi-Input Functional Encryption (MIFE) and Multi-Client Functional Encryption (MCFE), together with their decentralized variants [GGG+14, GKL+13, CDG+18a], have been receiving a strong interest from the cryptographic community. They generalize the nice functional encryption primitive [SW05, BSW11] where the single input x, in the encryption procedure, is split into an input vector $(x_1, \ldots, x_n)$, and the components can be encrypted independently, possibly by different senders/clients in MCFE. An index i for each component, and a (typically time-based) tag tag for MCFE, are used for every encryption $c_i = \mathsf{Enc}(i, \mathsf{tag}, x_i)$. From the n encrypted components under the same tag tag, anyone owning a functional decryption key dk_f, for the n-ary function f, can compute $f(x_1, \ldots, x_n)$ but nothing else about the individual x_i's. In this chapter, we focus on a standard and optimal security model for the most general form of MCFE, namely *decentralized* MCFE, where the generation of functional decryption keys is also split between multiple clients.

Previous Corruption Model for (D)MCFE. In previous (D)MCFE, encryption was claimed to require a private key ek_i per client, for each component c_i, because of deterministic encryption. Then, some of these keys might get corrupted. In DMCFE, where multiple senders contribute to generate the decryption functional keys and also own secret keys sk_i, and some can get corrupted. Therefore, there exists potential corruption of two categories of keys regarding DMCFE that need to be dealt with: a private encryption key ek_i for encryption and a secret key sk_i for generating functional keys. The proposed corruption model in the work on DMCFE by Chotard *et al.* [CDG+18a] is: when an adversary corrupts a client i, it receives both $(\mathsf{sk}_i, \mathsf{ek}_i)$. However, this does not reflect the real-life situation. In fact, the encryption keys ek_i's and the secret keys sk_i's can have different levels of protection (sk_i looks more critical than ek_i) and can be stored on different devices. This is thus a strong restriction to get both keys in case of corruption. Actually, this corruption model was employed in the previous DMCFE constructions for inner products $f_{\mathbf{y}}(\mathbf{x}) = \langle \mathbf{x}, \mathbf{y} \rangle$, as the numbers of sk_i's and ek_i's are equal, and in most of them particularly, sk_i is either included in ek_i, *e.g.* [CDG+18a, CDG+18b, LT19], or they are the same, *e.g.* [ABKW19, ABG19, AGT21b][1]. But this might not always be the case. Specifically, for quadratic functions computing $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{x}^\top \mathbf{A} \mathbf{x}$ as considered in [AGT21a, AGT22], one could have n^2 secret keys sk_j for the square matrix $\mathbf{A}$ and n encryption keys ek_i only for the input vector $\mathbf{x}$. Hence, the holders of sk_j's and ek_i's might differ.

Previous Notions of Admissible Attacks against (D)MCFE. Generally, studying an advanced cryptographic primitive involves formalizing the ubiquitous perception of trivial attacks when devising its security notion, those that exploit only the *functionality* of the primitive to trivially break any specific constructions. A standard example is the case of *identity-based encryption* [Sha84, Coc01, BF01, BGH07], of which the widely agreed security notion forbids the adversary to obtain the secret key of any identity that could decrypt the challenge ciphertext. In our case of (D)MCFE, everything becomes much more complicated due to the computational aspect of the function class and the corruption in multi-user setting. Following the introduction of (D)MCFE in the seminal paper [CDG+18a], to the best of our knowledge, all follow-up studies on (D)MCFE, for instance [CDG+18b, ABKW19, ABG19, LT19, CDSG+20, AGT21b], administered an *admissibility condition* in order to prohibit trivial attacks, and restricted particularly adversaries to asking the challenge components $\mathbf{x}^{(0)}[i] = \mathbf{x}^{(1)}[i]$ in case of a corrupted

[1] The work of [AGT21b] constructs *function-hiding dynamic decentralized* FE, which directly yields a DMCFE with a stronger property of function secrecy. Even though their proposed security model captures separated corruption of ek_i and sk_i, implying they are different, their dynamic decentralized FE construction uses the same key for both and so does the resulting DMCFE, *i.e.* $\mathsf{sk}_i = \mathsf{ek}_i$ for every i.

i. Attacks that satisfy the admissibility condition are called *admissible attacks*. Nonetheless, there was no satisfactory justification for such a restriction, except that all the constructions used a deterministic encryption, and so the corruption of ek_i could allow to re-encrypt $\mathbf{x}^{(0)}[i]$ and compare with the challenge ciphertext. This was thus also the similar argument to support private encryption keys. Since then, relaxing the foregoing constraint was widely believed to be insurmountable for constructing (D)MCFE schemes and proving their security.

Chapter Outline. This chapter develops further our results for an *improved security model for DMCFE*, which are introduced in Section 2.3.2. We start with the technical overview in Section 5.2 that provides high-level details on the motivation behind our refined security notion for DMCFE: *(i)* allowing an adversary to *separately* corrupt the secret key (sk_j for partial key generation corresponding to some j-th function parameter) or the encryption key (ek_i for encrypting messages of the i-th client), *(ii)* a more relaxed admissiblity condition that implies more attacks are considered in our new security notion, and *(iii)* a framework to argue whether our new notion is *optimal*, *i.e.* cannot be relaxed further. Formal definitions are given in Section 5.3. A concrete study for the class computing inner products, with various intermediate steps, is presented in Sections 5.4.1 and 5.4.3. In following chapters of this thesis, this stronger security notion of DMCFE will be examined again in Chapter 6 with respect to more complex function classes than inner products. All abridged proofs and technical details can be found in the full version [NPP23b] of [NPP23a].

5.2 Technical Overview

5.2.1 Motivations for a Refinement on Admissibility

In the seminal work on DMCFE for a function class $\mathcal{F}$ by Chotard *et al.* [CDG+18a], the authors define the security game with oracles

$$(\textbf{Initialize}, \textbf{Corrupt}, \textbf{LoR}, \textbf{DKeyGenShare}, \textbf{Enc}, \textbf{Finalize})$$

between a challenger and an adversary. The oracle **Initalize** sets up the parameters, including the number of clients n and their secret-encryption key pairs $(\mathsf{sk}_i, \mathsf{ek}_i)$. The oracle **DKeyGenShare** produces functional key components for $F \in \mathcal{F}$ using sk_i under some function tag $\mathsf{tag\text{-}f} \in \mathsf{Tag}$, while **LoR** is the left-or-right oracle, which outputs the challenge ciphertext component of $\mathbf{x}^{(b)}[i]$ under $\mathsf{tag} \in \mathsf{Tag}$ upon receiving $(\mathbf{x}^{(0)}[i], \mathbf{x}^{(1)}[i])$ for $b \xleftarrow{\$} \{0,1\}$. An adversary can corrupt any client i of his choice by querying **Corrupt** so as to receive *both* $(\mathsf{sk}_i, \mathsf{ek}_i)$.

In the end, a set of conditions is specified such that the adversary wins the game only when they conform to these conditions and outputs a correct b' equal to the challenge bit b. The main reason there are such conditions is to focus only on the scenarios where a notion of semantic security really makes sense in this DMCFE setting. We call an attack that satisfies such conditions an *admissible* attack. Checking these conditions is done by a **Finalize** procedure in the security game, according to the sets $\mathcal{C}$ of corrupted clients (asked to **Corrupt**), $\mathcal{H}$ of honest clients, and $\mathcal{Q}$ of key queries (asked to **DKeyGenShare**) during the attack. To recall from [CDG+18a, Def. 2, Def. 5], the adversary's output is ignored and replaced by a random bit, *i.e.* the attack is NOT admissible if one of the following holds:

1. There exists a corrupted client $i \in \mathcal{C}$ such that the i-th components of the challenge messages $(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$ are not the same, *i.e.* $\mathbf{x}^{(0)}[i] \neq \mathbf{x}^{(1)}[i]$.

2. There exists a tag $\mathsf{tag} \in \mathsf{Tag}$ (respectively, $\mathsf{tag\text{-}f} \in \mathsf{Tag}$) and $i \neq j \in \mathcal{H}$ such that the j-th challenge component (respectively, key component) is queried but the i-th challenge component (respectively, key component) is not.

3. None of the two above conditions are satisfied, but there exists a function F queried to **DKeyGenShare** that differs on $(\mathbf{x}^{(0)}, \mathbf{x}^{(1)})$, *i.e.* $F(\mathbf{x}^{(0)}) \neq F(\mathbf{x}^{(1)})$.

Our observation is that only the condition 3 can be justified for the sake of avoiding trivial attacks, while the other conditions 1 and 2 do not have satisfactory explanations. About condition 1, we have seen from the attacks in the paragraph **Impact and Feasibility** of Section 2.3.2 that this condition is artificial and unfortunately excludes also non-trivial attacks. About condition 2, follow-up works [CDG+18b, CDSG+20] and other results on the subject [AGRW17, DOT18, ABKW19, ABG19] show that we can completely remove this constraint. Our objective now becomes devising a less restrictive admissible condition, which should capture and generalize only condition 3. We recall that a less restrictive condition implies *more* attacks will be considered non-trivial and we obtain a *stronger* security model.

5.2.2 Towards a New Admissibility Condition under Separated Corruption of Keys

Our first step is to *separate* the corruption of sk_i from that of ek_i, *i.e.* the adversary has to specify which type of keys with respect to component i he wants to corrupt. All prior works allow the adversary to corrupt both keys *at once*. This separation helps us define in a finer way which information the adversary can deduce using each type of corrupted keys, and thus even deal with public-key encryption. As a result, we have sets of corrupted and honest clients $(\mathcal{C}_{\mathsf{skey}}, \mathcal{H}_{\mathsf{skey}}), (\mathcal{C}_{\mathsf{ekey}}, \mathcal{H}_{\mathsf{ekey}})$, independently for the secret keys $(\mathsf{sk}_j)_j$ and the encryption keys

$(\mathsf{ek}_i)_i$. This even allows to have independent sets of clients owning the secret keys $(\mathsf{sk}_j)_j$ and the encryption keys $(\mathsf{ek}_i)_i$. Our complete security experiment can be found in Figure 5.1. Being already mentioned in **Previous Corruption Model for (D)MCFE** of Section 5.1, to the best of our knowledge, almost all prior proposed contructions of (D)MCFE *cannot* handle separate corruption of ek_i and sk_i, for example, see [CDG+18a, CDG+18b, LT19, ABKW19, ABG19, AGT21b], despite the fact that a such separation is meaningful and is indeed discussed notably in the security model of [AGT21b].

Next, we represent an n-ary function $F : \mathcal{D}_1 \times \cdots \times \mathcal{D}_n \to \mathcal{R}$ of a function class $\mathcal{F}$ by a length-m vector of parameters from $\mathsf{Param}_1 \times \cdots \times \mathsf{Param}_m$, given by a deterministic encoding $\mathsf{p} : \mathcal{F} \to \mathsf{Param}_1 \times \cdots \times \mathsf{Param}_m$ and m can be different from n. Given such representations as vectors for both inputs and functions, we define the notion *deducible inputs and functions* (see Definition 5.1). More specifically, let $\mathcal{H}_{\mathsf{inp}} \subseteq [n], \mathcal{H}_{\mathsf{func}} \subseteq [m]$ and suppose we are given $\mathbf{x}_{\mathsf{inp}} \in (\mathcal{D}_i)_{i \in \mathcal{H}_{\mathsf{inp}}}$ and $\mathbf{y}_{\mathsf{func}} \in (\mathsf{Param}_i)_{i \in \mathcal{H}_{\mathsf{func}}}$ as lists of inputs and parameters that are indexed by $\mathcal{H}_{\mathsf{inp}}, \mathcal{H}_{\mathsf{func}}$ respectively. A vector $\mathbf{z}$ is *deducible* from $\mathbf{x}_{\mathsf{inp}}$ if their coordinates at positions in $\mathcal{H}_{\mathsf{inp}}$ are the same. Similarly, a function G is *deducible* from $\mathbf{y}_{\mathsf{func}}$ if its parameters coincide at positions in $\mathcal{H}_{\mathsf{func}}$ with $\mathbf{y}_{\mathsf{func}}$. Intuitively, the lists $(\mathbf{x}_{\mathsf{inp}}, \mathbf{y}_{\mathsf{func}})$ play the role of "honest" predetermined input's components and function's parameters, whilst the deducible $(\mathbf{z}, G)$ signifies what the adversary can infer by manipulating the remaining "corrupted" parts of the input and function.

Being equipped with this notion of deducible inputs and functions, our admissible condition is formulated as:

> Given the sets $(\mathcal{H}_{\mathsf{skey}}, \mathcal{H}_{\mathsf{ekey}})$, an *attack* is NOT admissible if there exist $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, a function $F \in \mathcal{F}$ with parameters $\mathbf{y} = (y_j)_{j \in [m]}$, two challenges $(\mathbf{x}^{(0)}, \mathbf{x}^{(1)}) := (x_i^{(0)}, x_i^{(1)})_{i \in [n]}$ such that $(F, \mathsf{tag\text{-}f})$ is queried to **DKeyGenShare**, $((x_i^{(0)}, x_i^{(1)})_{i \in [n]}, \mathsf{tag})$ is queried to **LoR** and there exists a pair $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)})$ deducible from $(\mathbf{x}^{(0)}_{\mathsf{ekey}}, \mathbf{x}^{(1)}_{\mathsf{ekey}})$, a function G deducible from $\mathbf{y}_{\mathsf{skey}}$ satisfying $G(\mathbf{z}^{(0)}) \neq G(\mathbf{z}^{(1)})$ where we define $\mathbf{y}_{\mathsf{skey}} := (y_i)_{i \in \mathcal{H}_{\mathsf{skey}}}$ and for $b \in \{0, 1\}$, $\mathbf{x}^{(b)}_{\mathsf{ekey}} := (x_i^{(b)})_{i \in \mathcal{H}_{\mathsf{ekey}}}$.

It can be verified that if an attack satisfies the previous notion of admissibility in the original work [CDG+18a], such an attack will satisfy our notion of admissibility as well. Moreover, we can adapt naturally our admissibility from DMCFE to MCFE[2] and also demonstrate that the prior DDH-based constructions for MCFE with deterministic encryption, for example from [CDG+18a, ABG19, CDSG+20] to name a few, as well as an LWE-based construction for MCFE from [LT19] with slightly randomized encryption by Gaussian errors, cannot be proven secure in our new model by giving a concrete *admissible* attack, as already explained in Section 5.1.

5.2.3 Optimality of the New Admissibility: A Conceptual Challenge

After formulating a new admissibility condition, one natural question arises: *Is this the most suitable condition?* From a conceptual point of view, we want to prove that

> For *certain* function classes, our admissibility cannot be relaxed, *i.e.* one cannot admit *some* non-admissible attack following our definition and still hope to be able to construct *some* specific efficient scheme that is provably secure.

Unsurprisingly, this poses a great definitional problem.

First of all, in all previous studies on (D)MCFE starting from [CDG+18a], the admissibility concerns *adversaries* in the security game. Hence, if we want to prove the above claim, we need to consider *all* possible adversaries that can run non-admissible attacks and argue that they must be excluded. This is hard to argue rigorously, for example, what happens if a "dummy"

[2]The admissibility for MCFE is the particular condition when $\mathcal{H}_{\mathsf{skey}} = [m]$ and thus $\mathbf{y}_{\mathsf{skey}} = \mathbf{y}$, meaning the only deducible function is F itself.

adversary behaves in a non-admissible way but in the end outputs only a random guess for the challenge bit? Therefore, our very first step is to define the admissibility condition differently and take into account general *attacks* instead of adversaries. Afterwards, our optimality notion for an admissibility condition on attacks is stated that:

> An admissibility is optimal for $\mathcal{F}$ if we can construct a *passive* ppt distinguisher $\mathcal{S}$ that receives *some* non-admissible attack coming from the queries of an adversary $\mathcal{A}$ to a challenger Chall in the game for a DMCFE $\mathcal{E}$, uses only properties of $\mathcal{F}$, and devises a *generic* strategy to output the correct challenge bit with significant probability in the security game against *any* arbitrary DMCFE $\mathcal{E}'$.

Intuitively, $\mathcal{S}$ passively observes the non-admissible queries in the attack from some specific interaction between $\mathcal{A}$ against some specific scheme $\mathcal{E}$. Yet, these queries helps $\mathcal{S}$ come up with a general approach to win significantly against any DMCFE scheme in a game that allows such non-admissible behaviors. This means it is impossible to prove security whenever this kind of behaviors is allowed. We formalize all these details in Definition 5.9 and Theorem 5.10, **Remark** 5.12 elaborates more on the proof of our optimality claim. In [NPP23b, Appendix B.1], our admissibility's optimality is verified for concrete most-studied function classes. Informally, we will explain in Section 5.3 the framework we proposed for arguing the optimality of an admissibility. Finally, the detailed admissibility condition for the class of inner products is given in **Remark** 5.14.

5.2.4 DMCFE for Inner Products with Refined Security Model

After introducing a new notion of admissibility in the security model for DMCFE and argue its optimality, we provide concrete constructions of DMCFE for inner products that are secure in this model. Our results are twofold. In Section 5.4.1 we give an intermediate construction where the new admissibility is translated in the case of inner-products together with the *complete* queries restriction (similar to condition 2 in previous works). In Section 5.4.3, we leverage this backbone construction from Section 5.4.1 to remove this complete queries restriction via a generic transformation as well as a concrete scheme with improved security.

In the following we highlight the main ideas of our backbone construction in Section 5.4.1. Our function class of interest is for computing inner products $\mathcal{F}^{\mathsf{IP}} = \{F_{\mathbf{y}}\}$ and $F_{\mathbf{y}} : \left(\mathbb{Z}_q^*\right)^n \to \mathbb{Z}_q$ is defined as $F_{\mathbf{y}}(\mathbf{x}) := \langle \mathbf{x}, \mathbf{y} \rangle$. The parameter vector of $F_{\mathbf{y}}$ is simply $\mathbf{y} = (y_1, \ldots, y_n) \in \mathbb{Z}_q^n$ and thus the number of parameters is the same as the dimension of the $\mathbb{Z}_q$-vector space for a prime q. Our construction relies on the notion of *Dual Pairing Vector Spaces* (DPVSes, see Section 3.3). We use DPVSes in the (additively written) bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, \mathbf{e}, g_1, g_2, g_\mathsf{t})$. We sample n pairs of random dual bases $(\mathbf{H}_i, \mathbf{H}_i^*)_{i=1}^n$. Each client i will use their encryption key ek_i to encrypt the component x_i under some tag to get a ciphertext component $\mathsf{ct}_{\mathsf{tag},i}$, which is a vector of elements in $\mathbb{G}_1$ computed using $\mathbf{H}_i$. Accordingly, each secret key sk_i will be used by the DKeyGenShare in the decentralized key generation so as to generate a key component $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$ for y_i under some tag-f, which is a vector of elements in $\mathbb{G}_2$ computed using $\mathbf{H}_i^*$. During decryption, the product $\mathsf{ct}_{\mathsf{tag},i} \times \mathsf{dk}_{\mathsf{tag\text{-}f},i}$ of vectors lying in dual bases will yield an element in $\mathbb{G}_\mathsf{t}$ of the form in the IPFE scheme by Agrawal, Libert, and Stehlé (ALS) [ALS16]. We denote by $S = (s_1, \ldots, s_n), U = (u_1, \ldots, u_n)$ two vectors of secret scalars, intuitively which will be used in ALS ciphertext components $[\![s_i\omega + u_i\omega' + x_i]\!]$, where $[\![(\omega, \omega')]\!] \leftarrow \mathsf{H}(\mathsf{tag})$ comes from a full-domain hash function. In a centralized setting, such as the single-client scheme in [ALS16] or the MCFE scheme in [CDG+18a], the ALS key extraction provides $\langle S, \mathbf{y} \rangle$ and $\langle U, \mathbf{y} \rangle$ to be used in decryption.

Decentralizing ALS Key Extraction under Separated Corruption. The first technical challenge is how to implement the ALS key extraction in a decentralized manner, because each key generator possessing y_i will not be able to compute $\langle S, \mathbf{y} \rangle$ and $\langle U, \mathbf{y} \rangle$ due to the lack of

$(y_j)_{j\neq i}$. Our idea is to use $(s_i y_i, u_i y_i)$ in the i-th key components, masked by some randomness, then exploit the properties of products in DPVS that multiply facing coordinates together in order to "glue" this randomness to appropriate values in the i-th ciphertext component that enables correct decryption. More specifically, the components have the following form:

$$\begin{array}{lccccccc} \mathsf{ct}_{\mathsf{tag},i} & (& \cdots & \omega p_i & \omega' q_i & \mathsf{ALS\text{-}ciph} & \cdots &)_{\mathbf{H}_i}\ ; \\ \mathsf{dk}_{\mathsf{tag\text{-}f},i} & (& \cdots & s_i y_i \alpha_i + u_i y_i \gamma_i' & s_i y_i \gamma_i + u_i y_i \alpha_i & y_i & \cdots &)_{\mathbf{H}_i^*}; \end{array}$$

where $\mathsf{ALS\text{-}ciph}$ is the scalar in ALS ciphertext and (p_i, q_i) in $\mathsf{ct}_{\mathsf{tag},i}$ together with $(\alpha_i, \gamma_i, \gamma_i')$ in $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$ satisfy $p_i\alpha_i = \zeta_1, q_i\gamma_i = \zeta_2, q_i\alpha_i = \zeta_3, p_i\gamma_i' = \zeta_4$ and $\zeta_1, \zeta_2, \zeta_3, \zeta_4 \xleftarrow{\$} \mathbb{Z}_q^*$ are determined at setup time. However, the aforementioned local gluing technique is not enough, as it alone still reveals information about individual $x_i y_i$ from $\mathsf{ct}_{\mathsf{tag},i} \times \mathsf{dk}_{\mathsf{tag\text{-}f},i}$. A remedy is to put another layer of random secret sharings $\theta_i \xleftarrow{\$} \mathbb{Z}_q$ into the partial key components such that $\sum_{i=1}^n \theta_i = 0$ so that only when n pairs of ciphertext-key components are combined together will we obtain the decrypted result. This new secret shares $(\theta_i)_i$ are randomized by $d_{\mathsf{tag\text{-}f}} \leftarrow \mathsf{H}(\mathsf{tag\text{-}f})$ for each functional key, the newly randomized $(d_{\mathsf{tag\text{-}f}}\theta_i)_i$ will behave indistinguishably from a random independent secret sharing of 0 under DDH. We refer to Section 5.4.1 and the transition $\mathsf{G}_6 \to \mathsf{G}_7$ in in the proof of Theorem 5.15 for more details.

Handling Separated Corruption. Each encryption key ek_i will contain information relevant to the basis $\mathbf{H}_i$ so that client i can compute $\mathsf{ct}_{\mathsf{tag},i}$, meanwhile each key generator can use sk_i related to the *dual* basis $\mathbf{H}_i^*$ for deriving the partial $\mathbf{k}_{i,\mathsf{ipfe}}$. It appears that the contents of ek_i and sk_i belong to dual bases, independent for each i, and we handle their separated corruption by *basis changes* over $(\mathbf{H}_i, \mathbf{H}_i^*)$ in DPVS. We note that as soon as we program the basis change of one basis, we cannot control the change on its dual counterpart (defined by linear relations, see Section 3.3). To this end, our proofs can handle at best the scenario where one key type must be *statically* corrupted whereas the other's corruption can be *dynamic*, otherwise the fact that for some i the keys ek_i and sk_i can be corrupted dynamically, in separate ways, can lead to inconsistency between basis changes. In particular, we use basis changes to program the master secret values $(s_i, u_i)_i$ as well as the secret sharings $(\theta_i)_i$, thus we want to program the changes on the dual bases $\mathbf{H}_i^*$. Consequently, we enforce static corruption on sk_i and perfom those changes on i corresponding to honest sk_i[3].

Achieving New Admissibility by Embedding Revocation Mechanism into Components. The second technical challenge is to handle our new admissibility. In the prior weaker admissible condition introduced in [CDG+18a], where $(\mathsf{ek}_i, \mathsf{sk}_i)$ are corrupted together, if $i \in [n]$ is corrupted then $\mathbf{x}^{(0)}[i] = \mathbf{x}^{(1)}[i]$. Putting forward the translation of our new admissibility in the functionality for inner products, the concrete conditions are: let $\Delta\mathbf{x}[i] := \mathbf{x}^{(0)}[i] - \mathbf{x}^{(1)}[i]$, then

1. For all $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$, $\sum_{i\in(\mathcal{H}_{\mathsf{skey}}\cup\mathcal{H}_{\mathsf{ekey}})} \Delta\mathbf{x}[i]\mathbf{y}[i] = 0$.
2. For all $i \in \mathcal{C}_{\mathsf{ekey}} \setminus \mathcal{C}_{\mathsf{skey}}$, either $\mathbf{x}^{(0)}[i] = \mathbf{x}^{(1)}[i]$ or $\mathbf{y}[i] = 0$.
3. For all $i \in \mathcal{C}_{\mathsf{skey}}$, $\mathbf{x}^{(0)}[i] = \mathbf{x}^{(1)}[i]$.

As the main complication compared to [CDG+18a, CDSG+20], when $i \in \mathcal{C}_{\mathsf{ekey}} \setminus \mathcal{C}_{\mathsf{skey}}$ it can be the case that $\Delta\mathbf{x}[i] \neq 0$ and $\mathbf{y}[i] = 0$. We want to ensure that even in this configuration the adversary cannot distinguish the i-th ciphertext components.

We interpret this situation in the language of *revocation*: if the adversary obtains the i-th key component $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$ for $y_i := \mathbf{y}[i] = 0$, which is honestly generated as $i \in \mathcal{C}_{\mathsf{ekey}} \setminus \mathcal{C}_{\mathsf{skey}}$, then implicitly we are "revoking" the ability to learn information about the i-th challenge component using $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$, even when the adversary can encrypt whatsoever using the corrupted ek_i, whose role now resembles a "public key" as in usual revocation systems. This leads us to the idea of

[3] There are further involved technicalities to ensure that ek_i is constructed consistently, *e.g.* see the transition $\mathsf{G}_7 \to \mathsf{G}_8$ in the proof of Theorem 5.15.

embedding some sort of DDH-based revocation technique into each i-th component. We need to apply some revocation technique that is compatible with the bilinear group setting *and* the ALS ciphertext form, which is current employ at each component i in $\mathsf{ct}_{\mathsf{tag},i}$. We turn our attention to a recent work by Agrawal *et al.* [ABP+17], which builds public trace-and-revoke systems from standard assumptions and is particularly suitable for our objective because their constructions can be generically based on the DDH-based ALS IPFE. At a very high level, the decryption for m of the precedent scheme for revocation can be recasted as:

$$\frac{\text{ALS-IPFE.Dec}(\mathsf{sk}_{id}, \mathsf{ct})}{\langle \mathbf{x}_{id}, \mathbf{v}_{\mathsf{S}} \rangle} = \frac{\langle \mathbf{x}_{id}, \mathbf{v}_{\mathsf{S}} \cdot m \rangle}{\langle \mathbf{x}_{id}, \mathbf{v}_{\mathsf{S}} \rangle} = m \tag{5.1}$$

where sk_{id} is the decryption given for an identity id, $\mathbf{x}_{id}$ is some vector associated to id, and $\mathbf{v}_{\mathsf{S}}$ is derived from the revoked set S. With overwhelming probability, $\langle \mathbf{x}_{id}, \mathbf{v}_{\mathsf{S}} \rangle \neq 0$ conditioned on $id \notin \mathsf{S}$.

To adapt to our situation the division is translated to subtraction of coordinates and our "revoking" test depends only on a scalar y_i and whether $y_i = 0$ or not, the inner product become scalar multiplications in $\mathbb{Z}_q^*$. Consequently, we introduce extra coordinates in the DPVS bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ to implement the aforementioned revocation technique, locally inside the vector's components as follows:

$$\begin{array}{llcccccccl} \mathsf{ct}_{\mathsf{tag},i} & (& \omega p_i & \omega' q_i & \mathsf{ALS\text{-}ciph} - r_i v_i & r_i t_i & \cdots & \mathsf{rand} &)_{\mathbf{H}_i} ; \\ \mathsf{dk}_{\mathsf{tag\text{-}f},i} & (& s_i y_i \alpha_i + u_i y_i \gamma_i' & s_i y_i \gamma_i + u_i y_i \alpha_i & y_i & y_i v_i / t_i & \cdots & d_{\mathsf{tag\text{-}f}} \theta_i &)_{\mathbf{H}_i^*} ; \end{array}$$

Using extra coordinates to contain $[\![(\mathsf{ALS\text{-}ciph} - r_i v_i, r_i t_i)]\!]_1$ in $\mathsf{ct}_{\mathsf{tag},i}$ as well as $[\![(y_i, y_i v_i / t_i)]\!]_2$ in $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$ helps us perform a "local" ALS+revocation decryption for component i, following the idea (5.1), when performing $\mathsf{ct}_{\mathsf{tag},i} \times \mathsf{dk}_{\mathsf{tag\text{-}f},i}$. Intuitively, our uniformly random scalar $r_i \xleftarrow{\$} \mathbb{Z}_q$ plays the role similar to that of $\mathbf{x}_{id}$ in the blueprint from [ABP+17], that helps proving semantic security in the case of "revoked" $y_i = 0$ under random basis changes in DPVS using DDH. This probabilistic layer with $r_i \xleftarrow{\$} \mathbb{Z}_q$ allows to deal with corrupted encryption keys, even when $\mathbf{x}^{(0)}[i] \neq \mathbf{x}^{(1)}[i]$. This somehow covers public-key encryption. Our scheme is secure in the stronger security model under new admissibility and the complete queries restriction, *adaptively* in the challenges, with *dynamic* corruption of ek_i and *static* corruption of sk_i.

5.3 A Stronger Security Model for DMCFE

This section presents a new security model for (D)MCFE, in which we refine the *admissibility* of adversaries in the security game and allow a more fine-grained corruption of keys. Following the line of works by Chotard *et al.* [CDG+18a, CDSG+20], such a notion of admissible adversaries is for excluding the attacks that are inevitable under which we cannot prove security. Our objective is to define the admissible condition in a way that excludes as few attacks as possible, and as soon as such condition is weakened, there is an unconditional generic attack to trivially win the security game against *any* concrete scheme. First of all, Definition 5.1 formalizes the terminologies of parameters of a function and deducible functions/inputs.

Definition 5.1 (Deducible inputs and functions). *Fix $\lambda \in \mathbb{N}$ and denote by $\mathcal{F}_\lambda$ a family of n-ary functions indexed by λ, with domain $\mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n}$ and range $\mathcal{R}_\lambda$, where $n = n(\lambda) \in \mathbb{N}$ is a function. Let $\mathsf{Param}_1, \ldots, \mathsf{Param}_m$ denote m sets of parameters for functions in $\mathcal{F}_\lambda$, where $m = m(\lambda) \in \mathbb{N}$ is a function. Suppose there exists a deterministic encoding $\mathsf{p} : \mathcal{F}_\lambda \to \mathsf{Param}_1 \times \cdots \times \mathsf{Param}_m$, that maps a function $F_{\mathbf{y}} \in \mathcal{F}_\lambda$ to its parameters $\mathbf{y} \in \mathsf{Param}_1 \times \cdots \times \mathsf{Param}_m$. Let $\mathcal{H}_{\mathsf{inp}} \subseteq [n], \mathcal{H}_{\mathsf{func}} \subseteq [m]$ and suppose we are given $\mathbf{x}_{\mathsf{inp}} \in (\mathcal{D}_{\lambda,i})_{i \in \mathcal{H}_{\mathsf{inp}}}$ and $\mathbf{y}_{\mathsf{func}} \in (\mathsf{Param}_j)_{j \in \mathcal{H}_{\mathsf{func}}}$ as lists of inputs and parameters that are indexed by $\mathcal{H}_{\mathsf{inp}}, \mathcal{H}_{\mathsf{func}}$ respectively.*

A vector $\mathbf{z} \in \mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n}$ is said to be deducible from $\mathbf{x}_{\mathsf{inp}}$ *if $\forall\, i \in \mathcal{H}_{\mathsf{inp}} : \mathbf{z}[i] = \mathbf{x}_{\mathsf{inp}}[i]$. A function G is said to be* deducible from $\mathbf{y}_{\mathsf{func}}$ *if for all $i \in \mathcal{H}_{\mathsf{func}}$, we have $\mathsf{p}(G)[i] = \mathbf{y}_{\mathsf{func}}[i]$.*

Remark 5.2. If $\mathbf{y}_{\mathsf{func}} = \mathbf{y}$, for the parameter $\mathbf{y}$ of some function $F_{\mathbf{y}}$, then the only function deducible from $\mathbf{y}_{\mathsf{func}}$ is $F_{\mathbf{y}}$ itself. In the DMCFE setting, there will be situations with non-trivial $\mathbf{y}_{\mathsf{func}}$ where $\mathcal{H}_{\mathsf{func}} \subsetneq [m]$. For concrete function classes in our construction, we focus on the class to compute inner products $\mathcal{F}^{\mathsf{IP}} = \{F_{\mathbf{y}}\}$ and $F_{\mathbf{y}} : \left(\mathbb{Z}_q^*\right)^n \to \mathbb{Z}_q$ that is defined as $F_{\mathbf{y}}(\mathbf{x}) := \langle \mathbf{x}, \mathbf{y} \rangle$. For inner products, the parameters of a function $F_{\mathbf{y}} \in \mathcal{F}^{\mathsf{IP}}$ can be precisely defined to be $\mathsf{p}(F_{\mathbf{y}}) := \mathbf{y} \in \mathbb{Z}_q^n$ and the number of parameters m is equal to the number of arguments n. When $\mathcal{F}_\lambda$ is clear from context, we omit the subscript λ.

We now recall the notion of *decentralized multi-client functional encryption* (DMCFE) whose syntax is given below.

Definition 5.3 (Decentralized Multi-Client Functional Encryption). *A* decentralized multi-client functional encryption (DMCFE) scheme *for a function class $\mathcal{F}$ consists of five algorithms*

$$(\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$$

that are defined below:

$\mathsf{Setup}(1^\lambda)$**:** *Given as input a security parameter λ and $n = n(\lambda), m = m(\lambda)$, generate in a possibly interactive manner n encryption keys $(\mathsf{ek}_i)_{i \in [n]}$ as well as m secret keys $(\mathsf{sk}_j)_{j \in [m]}$ where $m, n : \mathbb{N} \to \mathbb{N}$ are functions.*

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i)$**:** *Given as inputs an encryption key ek_i, a message $x_i \in \mathcal{D}_{\lambda,i}$, and a tag tag, output a ciphertext $\mathsf{ct}_{\mathsf{tag},i}$.*

$\mathsf{DKeyGenShare}(\mathsf{sk}_j, \mathsf{tag\text{-}f}, y_j)$**:** *Given a secret key sk_j, a tag $\mathsf{tag\text{-}f} \in \mathsf{Tag}$, and the j-th parameter y_j, output a partial functional key $\mathsf{dk}_{\mathsf{tag\text{-}f},j}$.*

$\mathsf{DKeyComb}((\mathsf{dk}_{\mathsf{tag\text{-}f},j})_{j \in [m]}, \mathsf{tag\text{-}f}, F)$**:** *Given a tag $\mathsf{tag\text{-}f}$ together with a function F and m partial functional keys $\mathsf{dk}_{\mathsf{tag\text{-}f},j}$ for the parameters $\mathsf{p}(F)$, output the functional key $\mathsf{dk}_{\mathsf{tag\text{-}f},F}$.*

$\mathsf{Dec}(\mathsf{dk}_{\mathsf{tag\text{-}f},F}, \mathbf{c})$**:** *Given the functional decryption key $\mathsf{dk}_{\mathsf{tag\text{-}f},F}$ and a list of ciphertexts $\mathbf{c} := (\mathsf{ct}_{\mathsf{tag},i})_{i=1}^n$ of length n, output an element in $\mathcal{R}_\lambda$ or an invalid symbol $\perp$.*

We make the assumption that all public parameters are included in the secret keys and the encryption keys, as well as the (partial) functional decryption key.

Correctness. We require that for sufficiently large $\lambda \in \mathbb{N}$, for all $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, for all $F \in \mathcal{F}$, $(x_i)_{i\in[n]} \in \mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n}$ and

$$\begin{aligned}&\mathsf{sk}_j, (\mathsf{ek}_i)_{i\in[n]} \leftarrow \mathsf{Setup}(1^\lambda);\ \mathsf{dk}_{\mathsf{tag\text{-}f},j} \leftarrow \mathsf{DKeyGenShare}(\mathsf{sk}_j, \mathsf{tag\text{-}f}, y_j)_{j\in[m]}\ ;\\ &\mathsf{dk}_{\mathsf{tag\text{-}f},F} \leftarrow \mathsf{DKeyComb}((\mathsf{dk}_{\mathsf{tag\text{-}f},j})_j, \mathsf{tag\text{-}f}, F);\ (\mathsf{ct}_{\mathsf{tag},i})_i \leftarrow (\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i))_i\end{aligned}$$

where $i \in [n]$ and $j \in [m]$, the following holds with overwhelming probability:

$$\mathsf{Dec}\left(\mathsf{dk}_{\mathsf{tag\text{-}f},F}, (\mathsf{ct}_{\mathsf{tag},i})_{i=1}^n\right) = F(x_1, \ldots, x_n) \text{ when } F(x_1, \ldots, x_n) \neq \perp^4 \tag{5.2}$$

where $F : \mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n} \to \mathcal{R}_\lambda$ and the probability is taken over the random coins of algorithms.

Security. We follow the approach in the work by Chotard *et al.* [CDG+18a] so as to define the security game with oracles **Initialize**, **Corrupt**, **LoR**, **Enc**, **DKeyGenShare**, and **Finalize**. We recall that the oracle **Enc** is necessary for a simpler notion of one challenge, while retaining an equivalence to the multi-challenge notion using a hybrid argument shown in Lemma 5.6. The adversary is also able to corrupt *separately* the secret key sk_j of any key-generator j as well as the encryption key ek_i of any client i, which is done via requests (i, skey) or (j, ekey) to the oracle **Corrupt**, respectively. We need to exclude trivial attacks that can be mounted in the security experiment. Those restrictions are encompassed in the notion of *admissibility*, which is extended from similar notions in the works of [CDG+18a, CDSG+20].

In a nutshell, Definition 5.4 gives the definition of admissibility, generalizing the admissibility condition that has been consistently used since the seminal work of Chotard *et al.* [CDG+18a]. We refer to Section 5.2 for a high-level discussion. In the subsequent Section 5.3.1, we give the full formal treatment to prove the *optimality* of our admissibility condition in Definition 5.4. The successive security analyses of our DMCFE schemes rely crucially on this definition, translated for the concrete class of inner product in Remark 5.14.

Definition 5.4 (Admissibility condition). *Let $\mathcal{A}$ be a* ppt *adversary and let*

$$\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$$

be a DMCFE scheme for a function class $\mathcal{F}$ set up w.r.t $\lambda \in \mathbb{N}$. In **Finalize**, *considering the queries $(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$, we say that the* attack *corresponding to these queries is* NOT admissible *if the following is satisfied*

> *There exist $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, a function $F \in \mathcal{F}$ having parameters $\mathbf{y} \in \mathsf{Param}_1 \times \cdots \times \mathsf{Param}_m$, two challenges $(x_i^{(0)}, x_i^{(1)})_{i\in[n]}$ such that $(\mathsf{tag\text{-}f}, F) \in \mathcal{Q}$ is queried to* **DKeyGenShare**, *$((x_i^{(0)}, x_i^{(1)})_{i\in[n]}, \mathsf{tag})$ is queried to* **LoR** *and there exists a pair $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)})$ deducible from $(\mathbf{x}_{\mathsf{ekey}}^{(0)}, \mathbf{x}_{\mathsf{ekey}}^{(1)})$, a function G deducible from $\mathbf{y}_{\mathsf{skey}}$ satisfying*
>
> $$G(\mathbf{z}^{(0)}) \neq G(\mathbf{z}^{(1)})\ , \tag{5.3}$$
>
> *where we define $\mathbf{y}_{\mathsf{skey}} := (\mathbf{y}[i])_{i\in\mathcal{H}_{\mathsf{skey}}}$ and for $b \in \{0,1\}$, $\mathbf{x}_{\mathsf{ekey}}^{(b)} := (x_i^{(b)})_{i\in\mathcal{H}_{\mathsf{ekey}}}$.*

Otherwise, we say that the aforementioned attack is admissible.

Definition 5.5 (IND-security for DMCFE). *A DMCFE scheme*

$$\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$$

[4] See [BO13, ABN10] for discussions about this relaxation. The general reason is that some functionality might contain $\perp$ in its range and if $F((x_i)_i) = \perp$ we do not impose $\mathsf{Dec}\left(\mathsf{dk}_{\mathsf{tag\text{-}f},F}, (\mathsf{Enc}(\mathsf{ek}_i, x_i, \mathsf{tag}))_{i=1}^n\right) = F((x_i)_i)$, neither do we disallow it.

for the function class $\mathcal{F} = \{F_\lambda\}_{\lambda\in\mathbb{N}}$ *is* xx-secure *if for all* ppt *adversaries* $\mathcal{A}$*, and for all sufficiently large* $\lambda \in \mathbb{N}$*, the following probability is negligible*

$$\mathsf{Adv}^{\mathsf{xx}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) := \left|\Pr[\mathsf{Expr}^{\mathsf{xx}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| \ .$$

The game $\mathsf{Expr}^{\mathsf{xx}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ *is depicted in Figure 5.1. The security level indicator* xx *can be:* dmc-ind-cpa *to indicate* IND-security *with adaptive challenges, dynamic corruption of* ekey*, and dynamic corruption of* skey*;* dmc-sel-ind-cpa *to indicate* selective IND-security *with selective challenges, dynamic corruption of* ekey*, and dynamic corruption of* skey*;* dmc-stat-ind-cpa *to indicate* static IND-security *with adaptive challenges, static corruption of* ekey*, and static corruption of* skey[5]*;* dmc-ind-cpa-1chal *indicate* one-time IND-security *with one adaptive challenge tag, dynamic corruption of* ekey*, and dynamic corruption of* skey*. The probability is taken over the random coins of* $\mathcal{A}$ *and the algorithms.*

Lemma 5.6 below allows us to concentrate on the notion of one-time IND-security for our DMCFE constructions, whose proof is a standard hybrid argument.

Lemma 5.6. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$ *be a DMCFE scheme for the function class* $\mathcal{F}$*. If* $\mathcal{E}$ *is one-time IND-secure, then* $\mathcal{E}$ *is IND-secure.*

5.3.1 Optimality of Admissibility *as per* Definition 5.4

In this section, we demonstrate that our notion of admissibility in Definition 5.4 is capturing all trivial attacks against DMCFE schemes for non-trivial function classes, which include the class of inner-product and quadratic functionalities. The high-level plan is given below.

Proving Optimality. The general idea on defining the *optimality* of an admissibility condition can be revisited in Section 5.2.3. We now explain briefly how one can show that an admissibility condition is optimal following what we try to define. First and foremost, a notion of optimality makes sense when we consider only certain functionalities and not any arbitrary class of functions. For example, for a general functionality, the adversary's admissibility as defined in Definition 5.4 might not be *efficiently* decidable. Roughly speaking, **Finalize** may have to go through *all* $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)})$ deducible from $(\mathbf{x}^{(0)}_{\mathsf{ekey}}, \mathbf{x}^{(1)}_{\mathsf{ekey}})$ and *all* G deducible from $\mathbf{y}_{\mathsf{skey}}$ so as to check relation (5.3). Therefore, we want to focus on classes for which the admissibility can be decided efficiently by **Finalize**, at least for the sake of having an efficient challenger in the security game.

In addition, we require a further property of the functionality under consideration: in view of the admissibility check (5.3), the deduction of $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)})$ from $(\mathbf{x}^{(0)}_{\mathsf{ekey}}, \mathbf{x}^{(1)}_{\mathsf{ekey}})$ and of a function G from $\mathbf{y}_{\mathsf{skey}}$ can be done efficiently. We coin this property *fixed-component distinguishability*. In summary, we restrain the optimality evaluation to classes that are (1) fixed-component distinguishable and (2) for which the admissibility is efficiently decidable. In [NPP23a, Appendix B.1], we prove that both most studied functionalities for inner products and quadratic evaluations satisfy properties (1) and (2).

The core of our reasoning that an admissibility condition is optimal comprises building a ppt distinguisher, which can exerts a *non-admissible* attack, to trivially win significantly the security game against *any* DMCFE scheme. We recall that Definition 5.4 views attacks as ensembles of queries made by some adversary in its security game. Because the class allows deciding efficiently the admissibility, our distinguisher can efficiently determine which query in the attack will violate the check (5.3), and thanks to the fixed-component distinguishability, the triplet $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)}, G)$ can be concretely reconstructed in an efficient manner.

[5] In addition, we can allow dynamic corruption on one type but static corruption on the other type of keys, such as dmc-stat-sk-ind-cpa to indicate *partially static IND-security* with adaptive challenges, dynamic corruption of ekey, and static corruption of skey.

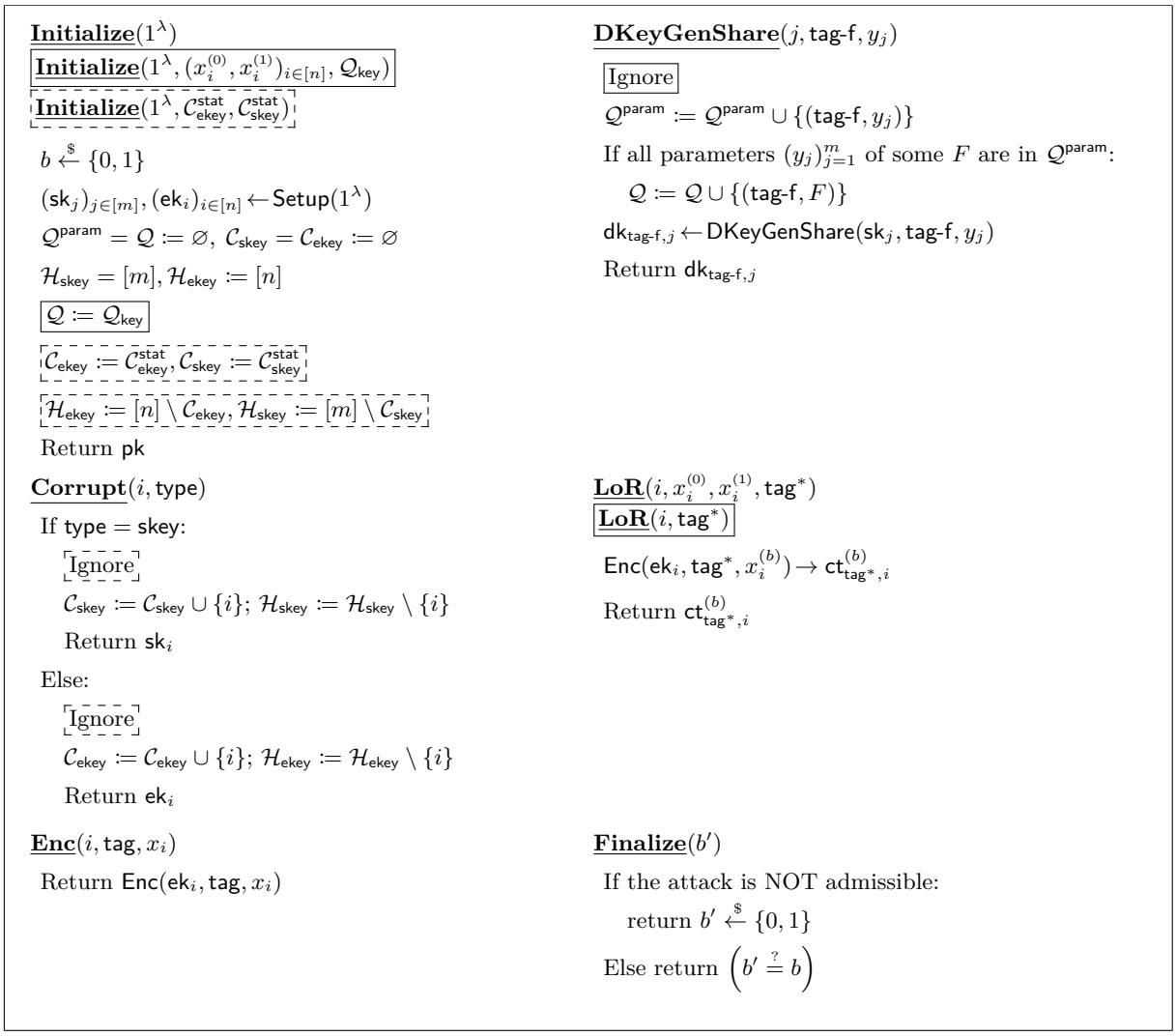


Figure 5.1: The security games $\mathsf{Expr}^{\mathsf{dmc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$, $\boxed{\mathsf{Expr}^{\mathsf{dmc\text{-}sel\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)}$, and $\boxed{\mathsf{Expr}^{\mathsf{dmc\text{-}stat\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)}$ for Definition 5.5. The admissibility condition is defined in Definition 5.4.

In the end, facing any DMCFE challenger that allows the foregoing non-admissible behaviour, our distinguisher can simply use $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)}, G)$ to trivially win the game. This means that whenever we allow a non-admissible attack, or in other words whenever we try to relax Definition 5.4, no DMCFE scheme can be proved secure due to the existence of the above distinguisher.

To begin our formal treatment, we restrain our attention to particular function classes that satisfy certain properties.

Definition 5.7 (Fixed-component distinguishable classes). *Fix $\lambda \in \mathbb{N}$ and denote by $\mathcal{F}_\lambda = \{F : \mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n} \to \mathcal{R}_\lambda\}$ a family of n-ary functions indexed by λ having m parameters, where $m = m(\lambda), n = n(\lambda)$ are functions.*

For $F \in \mathcal{F}_\lambda$, a triple $(\mathbf{x}^{(0)}_{\mathsf{inp}}, \mathbf{x}^{(1)}_{\mathsf{inp}}, \mathcal{H}_{\mathsf{inp}})$, where $\mathbf{x}^{(b)}_{\mathsf{inp}} \in \prod_{i\in\mathcal{H}_{\mathsf{inp}}} \mathcal{D}_{\lambda,i}$ for $b \in \{0,1\}$, is said to be distinguishing F_λ with fixed components *if there exists a deducible pair $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)}) \in \prod_{i\in[n]} \mathcal{D}_{\lambda,i}$ such that $F_\lambda(\mathbf{z}^{(0)}) \neq F_\lambda(\mathbf{z}^{(1)})$ where*

$$\begin{cases} \mathbf{z}^{(b)}[i] = \mathbf{x}^{(b)}[i] \text{ for } b \in \{0,1\}, i \in \mathcal{H}_{\mathsf{inp}} \\ \mathbf{z}^{(0)}[i] = \mathbf{z}^{(1)}[i]\ \forall\, i \in [n] \setminus \mathcal{H}_{\mathsf{inp}} \end{cases}.$$

A function F is said to be fixed-component distinguishable *if there exists a triple distinguishing*

F with fixed components and a ppt *Turing machine that, given as input this triple, outputs the corresponding deducible pair.*

A function class $\mathcal{F}_\lambda$ is fixed-component distinguishable *if for all $F \in \mathcal{F}_\lambda$ with parameters $\mathbf{p} := \mathsf{p}(F)$, there exists a fixed-component distinguishable function G deducible from $(\mathbf{p}[i])_{i\in\mathcal{H}_{\mathsf{func}}}$ for some $\mathcal{H}_{\mathsf{func}} \subseteq [m]$, and a* ppt *Turing machine that, given as inputs $(F, \mathcal{H}_{\mathsf{func}})$, outputs G.*

Remark 5.8. We remark that a function class $\mathcal{F}$ is fixed-component distinguishable does *not* necessarily imply that the admissibility from Definition 5.4 for $\mathcal{F}$ can be efficiently decided. Roughly speaking, given a function among the adversary's queries, the ppt Turing machine from fixed-component distinguishability will output *some* deducible function G for which one can test the admissible condition (5.3) efficiently. But that is not enough, as to decide the admissibility of an attack, we need to check *all* such deducible functions and there is no further guarantee in the case of general functionalities that we can do this check efficiently. In the concrete cases of inner products and quadratic functions, the check over all such deducible functions can be done efficiently by using their linear properties, see [NPP23a, Appendix B.1] for more details.

We now define what means for an admissibility to be *optimal* for a function class $\mathcal{F}$. For simplicity, we can consider the one-challenge setting thanks to Lemma 5.6.

Definition 5.9. *Let $\lambda \in \mathbb{N}$ and denote by $\mathcal{F}$ a family of $n(\lambda)$-ary functions indexed by λ, with $m(\lambda)$ parameters for each member of $\mathcal{F}$. An admissibility condition $\mathsf{adm}(\cdot)$ is* optimal *for $\mathcal{F}$ if there exists a* ppt *distinguisher $\mathcal{S}$ so that for all* non-admissible

$$(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$$

of some ppt *adversary $\mathcal{A}$ and against a DMCFE $\mathcal{E}$ for $\mathcal{F}$ in a security experiment $\mathsf{Expr}_{\mathcal{E},\mathcal{F},\mathcal{A}}$ given in Figure 5.1, we have*

$$\Pr\left[\mathcal{S}((\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})) = b \ : \ b \leftarrow \mathsf{Chall}(\mathsf{rand}_{\mathsf{Chall}})\right] \geq \frac{1}{\mathsf{poly}(\lambda)}$$

where $(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$ is well-defined at the time of **Finalize** *and $b \leftarrow \mathsf{Chall}(\mathsf{rand}_{\mathsf{Chall}})$ means the challenger Chall uses the bit b in $\mathsf{Expr}_{\mathcal{E},\mathcal{F},\mathcal{A}}$.*

We now state our main theorem for the optimality of our admissibility.

Theorem 5.10. *Let $\mathcal{F}$ be a function class that has efficient decidability for admissibility and is fixed-component distinguishable. Then, our admissibility condition as defined in Definition 5.4 is optimal for $\mathcal{F}$.*

Proof (Of Theorem 5.10). Without loss of generality, we consider the one-challenge notion. We need to prove that: there exists a ppt distinguisher $\mathcal{S}$ so that for any non-admissible $(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$ of some DMCFE $\mathcal{E}$ for $\mathcal{F}$ and some ppt adversary $\mathcal{A}$ in a security experiment $\mathsf{Expr}^{\mathsf{xx}}_{\mathcal{E},\mathcal{F},\mathcal{A}}$ given in Figure 5.1, we have

$$\Pr\left[\mathcal{S}((\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})) = b \ : \ b \leftarrow \mathsf{Chall}(\mathsf{rand}_{\mathsf{Chall}})\right] \geq \frac{1}{\mathsf{poly}(\lambda)} \ .$$

Let $\mathcal{E}^{\mathsf{abs}} = (\mathsf{Setup}^{\mathsf{abs}}, \mathsf{DKShare}^{\mathsf{abs}}, \mathsf{DKeyComb}^{\mathsf{abs}}, \mathsf{Enc}^{\mathsf{abs}}, \mathsf{Dec}^{\mathsf{abs}})$ be an abstract DMCFE for $\mathcal{F}$ that satisfies the correctness relation (5.2). We describe the distinguisher $\mathcal{S}$ as follows:

1. The distinguisher $\mathcal{S}$ parses

   $$(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$$

   then use $\mathcal{E}^{\mathsf{abs}}$ and $(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\})$ for abstracting the key components to obtain $\{(\mathsf{dk}^{\mathsf{abs}}_{\mathsf{tag\text{-}f},F,j})_{j\in[m]}\}_{(\mathsf{tag\text{-}f},F)\in\mathcal{Q}}$, the challenge ciphertext components to obtain $(\mathsf{ct}^{\mathsf{abs}}_{\mathsf{tag},i})_{i\in[n]}$ for

each $\{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}$, and the encryption responses to obtain $(\mathsf{ct}_i^{\mathsf{abs},(k)})_{i\in[n]}$. If there are corrupted keys sk_j or ek_i queried by $\mathcal{A}$, they will also be replaced by their abstracted counterparts $\mathsf{sk}_j^{\mathsf{abs}}$ or $\mathsf{ek}_i^{\mathsf{abs}}$. In the following $\mathcal{S}$ only needs the abstract DMCFE $\mathcal{E}^{\mathsf{abs}}$ for $\mathcal{F}$, no matter what the details of the concrete $\mathcal{E}$ are.

2. If there exists $(\mathsf{tag\text{-}f}, F) \in \mathcal{Q}$ such that $F(\mathbf{x}_0^*) \neq F(\mathbf{x}_1^*)$, $\mathcal{S}$ combines the key components of $(\mathsf{tag\text{-}f}, F)$, decrypts the challenge ciphertext components, and outputs 1 if and only if the result is $F(\mathbf{x}_1^*)$. All algorithms come from $\mathcal{E}^{\mathsf{abs}} = (\mathsf{Setup}^{\mathsf{abs}}, \mathsf{DKShare}^{\mathsf{abs}}, \mathsf{DKeyComb}^{\mathsf{abs}}, \mathsf{Enc}^{\mathsf{abs}}, \mathsf{Dec}^{\mathsf{abs}})$. Else, in the following we assume that $F(\mathbf{x}_0^*) = F(\mathbf{x}_1^*)$ for all $(\mathsf{tag\text{-}f}, F) \in \mathcal{Q}$.

3. Because this is a non-admissible attack, $\mathcal{S}$ uses the efficient decidability of $\mathcal{F}$ to find $(\mathsf{tag\text{-}f}, F) \in \mathcal{Q}$, whose parameters is $\mathbf{y} := \mathsf{p}(F)$, so that: there exists a pair $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)})$ deducible from $(\mathbf{x}_{\mathsf{ekey}}^{(0)}, \mathbf{x}_{\mathsf{ekey}}^{(1)})$, a function G deducible from $\mathbf{y}_{\mathsf{skey}}$ satisfying

$$G(\mathbf{z}^{(0)}) \neq G(\mathbf{z}^{(1)}) \ ,$$

where $\mathbf{y}_{\mathsf{skey}} := (\mathbf{y}[i])_{i\in\mathcal{H}_{\mathsf{skey}}}$ and $\mathbf{x}_{\mathsf{ekey}}^{(0)} := (x_i^{(0)})_{i\in\mathcal{H}_{\mathsf{ekey}}}$, $\mathbf{x}_{\mathsf{ekey}}^{(1)} := (x_i^{(1)})_{i\in\mathcal{H}_{\mathsf{ekey}}}$. We remark that finding F can be done efficiently in $\mathcal{Q}$ because the current attack comes from the execution of some ppt adversary $\mathcal{A}$, which implies the size of $\mathcal{Q}$ is polynomially bounded.

4. Because $\mathcal{F}$ is fixed-component distinguishable (see Definition 5.7), using F and $\mathcal{C}_{\mathsf{skey}}$, $\mathcal{S}$ can efficiently find a function G deducible from $\mathbf{y}_{\mathsf{skey}}$ such that G is fixed-component distinguishable.

5. Thanks to the fixed-component distinguishability of G, using $(\mathbf{x}_{\mathsf{ekey}}^{(0)}, \mathbf{x}_{\mathsf{ekey}}^{(1)})$ and $\mathcal{H}_{\mathsf{ekey}}$, the pair $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)})$ can be found efficiently by $\mathcal{S}$.

6. The distinguisher $\mathcal{S}$ then uses the corrupted keys $(\mathsf{ek}_i^{\mathsf{abs}})_{i\in\mathcal{C}_{\mathsf{ekey}}}$ to compute new ciphertext components $(\tilde{\mathsf{ct}}_{\mathsf{tag},i}^{\mathsf{abs}})_{i=1}^n$ of $\mathbf{z}^{(b)}$ implicitly, using $(\mathsf{ct}_{\mathsf{tag},i}^{\mathsf{abs}})_{i\in\mathcal{H}_{\mathsf{ekey}}}$ for the challenge $(\mathbf{x}_b^*[i])_{i\in\mathcal{H}_{\mathsf{ekey}}}$, and using $\mathsf{Enc}^{\mathsf{abs}}$ to encrypt $(\mathbf{z}^{(b)}[i])_{i\in\mathcal{C}_{\mathsf{ekey}}}$ using $(\mathsf{ek}_i^{\mathsf{abs}})_{i\in\mathcal{C}_{\mathsf{ekey}}}$.

7. Next, $\mathcal{S}$ uses the corrupted keys $(\mathsf{sk}_i^{\mathsf{abs}})_{i\in\mathcal{C}_{\mathsf{skey}}}$ to compute new key components $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},G,i}^{\mathsf{abs}})_{i=1}^n$ of G implicitly, using $(\mathsf{dk}_{\mathsf{tag\text{-}f},F,i}^{\mathsf{abs}})_{i\in\mathcal{H}_{\mathsf{skey}}}$, and using $\mathsf{DKShare}^{\mathsf{abs}}$ to derive $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},F,i}^{\mathsf{abs}})_{i\in\mathcal{C}_{\mathsf{skey}}}$ from $(\mathsf{p}(G)[i])_{i\in\mathcal{C}_{\mathsf{skey}}}$.

8. Finally, $\mathcal{S}$ uses $\mathsf{DKeyComb}^{\mathsf{abs}}$ to combine the newly generated key components $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i}^{\mathsf{abs}})_{i=1}^n$. Then $\mathcal{S}$ decrypts the newly generated challenge ciphertext $(\tilde{\mathsf{ct}}_{\mathsf{tag},i}^{\mathsf{abs}})_{i\in[n]}$ using the abstract algorithm $\mathsf{Dec}^{\mathsf{abs}}$, the adversary outputs 1 if and only if the result is equal to $G(\mathbf{z}^{(1)})$.

In the end, $\mathcal{S}$ outputs 1 if and only if $(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$ comes from an execution of any $\mathcal{A}$ against Chall of $\mathcal{E}$ in which Chall picks 1 as the challenge bit. This concludes the proof.

Remark 5.11. The abstract object $\mathcal{E}^{\mathsf{abs}}$ in the proof of Theorem 5.10 are used *only* in our formal proofs of the optimality for our admissible condition. In the concrete constructions of DMCFE, no such abstract objects are needed, as the admissibility are examined via concrete tests over the adversary's queries in the security game. For instance, see [NPP23a, Appendix B.1] for the cases of linear and quadratic functions.

Remark 5.12. The generic distinguisher $\mathcal{S}$ in Theorem 5.10 is *weak* in the sense that all it has is a non-admissible attack with the corresponding

$$(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$$

determined by $\mathcal{A}$'s behaviour during the security game, *not depending on* the concrete implementation of $\mathcal{E}$. However, thanks to the non-admissibility of the given attack and the fixed-component distinguishability of the function class, $\mathcal{S}$ can still output the correct challenge bit, in the security against *any* DMCFE scheme. This means that as soon as we allow some non-admissible behaviour, where the concrete descriptions of $\mathcal{A}$ and $\mathcal{E}$ can be arbitrary as long as this behaviour stays the same, there is no hope in proving security regardless of the specific implementation of $\mathcal{E}$. Equivalently, our Definition 5.4 that excludes exactly these non-admissible attacks cannot be enlarged in any sense and captures the most attacks against which we can prove security. Last but not least, we see clearly the role of the abstract DMCFE $\mathcal{E}^{\mathsf{abs}}$: it abstracts out the concrete details of *some* specific scheme $\mathcal{E}$ from which calculations over the non-admissible queries can be done, and return the "black-boxed" data that obey the correctness of DMCFE schemes for $\mathcal{F}$.

Optimality in the Case of Inner Products in Polynomially Bounded Range. In our concrete DDH-based construction of DMCFE for inner products in Section 5.4.1 and Section 5.4.3, our functionality is not for inner products over $\mathbb{Z}_q^n$ set up w.r.t $\lambda \in \mathbb{N}$, but only for *bounded* vectors such that the inner product evaluation is polynomially large. More specifically, we name this class $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$ in which any function $f_{\mathbf{y}} : \mathbf{x} \mapsto \langle \mathbf{x}, \mathbf{y} \rangle$ receives as inputs $\mathbf{x}$ and has parameters $\mathbf{y}$ such that $\|\mathbf{x}\|_\infty < B$ and $\|\mathbf{y}\|_\infty < B'$, where $B, B' = \mathsf{poly}(\lambda) \in \mathbb{N}$ are polynomials. The concrete admissibility (see Remark 5.14) is still the same because we are still computing inner products. Below wse prove that the admissibility *as per* Definition 5.4 is optimal for $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$.

Theorem 5.13. *Let $\lambda \in \mathbb{N}$ and $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$ be the function class to compute inner products in ranges parametrized by B, B'. Then, our admissibility condition as defined in Definition 5.4 is optimal for $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$.*

Proof. Without loss of generality, we consider the one-challenge notion. We need to prove that: there exists a ppt distinguisher $\mathcal{S}$ so that for any non-admissible attack of an adversary $\mathcal{A}$ against some DMCFE $\mathcal{E}$ for $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$ in a security experiment $\mathsf{Expr}_{\mathcal{E},\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}},\mathcal{A}}$ given in Figure 5.1, we have

$$\Pr\left[\mathcal{S}(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\}) = b \ : \ b \leftarrow \mathsf{Chall}(\mathsf{rand}_{\mathsf{Chall}})\right] \geq \frac{1}{\mathsf{poly}(\lambda)} \ .$$

Any non-admissible attack will make one of the following hold:

1. There exists $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$ s.t $\sum_{i \in \mathcal{H}} \Delta\mathbf{x}[i]\mathbf{y}[i] \neq 0$.
2. There exists $i^* \in \mathcal{C}_{\mathsf{ekey}} \setminus \mathcal{C}_{\mathsf{skey}}$ s.t $(\mathbf{x}^{(0)}[i^*] - \mathbf{x}^{(1)}[i^*])\mathbf{y}[i^*] \neq 0$.
3. There exists $i^* \in \mathcal{C}_{\mathsf{skey}}$ s.t $\mathbf{x}^{(0)}[i^*] \neq \mathbf{x}^{(1)}[i^*]$.

We describe the distinguisher $\mathcal{S}$ as follows and specify the strategy for each case:

1. The distinguisher $\mathcal{S}$ parses

   $$(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$$

   then use $\mathcal{E}^{\mathsf{abs}}$ and $(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\})$ for abstracting the key components to obtain $\{(\mathsf{dk}^{\mathsf{abs}}_{\mathsf{tag\text{-}f},F,j})_{j\in[m]}\}_{(\mathsf{tag\text{-}f},F)\in\mathcal{Q}}$, the challenge ciphertext components to obtain $(\mathsf{ct}^{\mathsf{abs}}_{\mathsf{tag},i})_{i\in[n]}$ for each $\{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}$, and the encryption responses to obtain $(\mathsf{ct}_i^{\mathsf{abs},(k)})_{i\in[n]}$. If there are corrupted keys sk_j or ek_i queried by $\mathcal{A}$, they will also be replaced by their abstracted counterparts $\mathsf{sk}_j^{\mathsf{abs}}$ or $\mathsf{ek}_i^{\mathsf{abs}}$. In the following $\mathcal{S}$ only needs the abstract DMCFE for $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$

   $$\mathcal{E}^{\mathsf{abs}} = (\mathsf{Setup}^{\mathsf{abs}}, \mathsf{DKShare}^{\mathsf{abs}}, \mathsf{DKeyComb}^{\mathsf{abs}}, \mathsf{Enc}^{\mathsf{abs}}, \mathsf{Dec}^{\mathsf{abs}})$$

   that satisfies the correctness requirement, no matter what the details of the concrete $\mathcal{E}$ are.

2. If there exists $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$ such that $\langle \mathbf{x}_0^*, \mathbf{y} \rangle \neq \langle \mathbf{x}_1^*, \mathbf{y} \rangle$, $\mathcal{S}$ combines the key components of $(\mathsf{tag\text{-}f}, \mathbf{y})$, decrypts the challenge ciphertext components, and outputs 1 if and only if the result is $\langle \mathbf{x}_1^*, \mathbf{y} \rangle$. All algorithms come from $\mathcal{E}^{\mathsf{abs}} = (\mathsf{Setup}^{\mathsf{abs}}, \mathsf{DKShare}^{\mathsf{abs}}, \mathsf{DKeyComb}^{\mathsf{abs}}, \mathsf{Enc}^{\mathsf{abs}}, \mathsf{Dec}^{\mathsf{abs}})$. Else, in the following we assume that $\langle \mathbf{x}_0^*, \mathbf{y} \rangle = \langle \mathbf{x}_1^*, \mathbf{y} \rangle$ for all $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$.

3. If case 1 happens:
   - Let $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$ be the query such that $\sum_{i \in \mathcal{H}} \Delta\mathbf{x}[i]\mathbf{y}[i] \neq 0$.
   - $\mathcal{S}$ uses the corrupted secret keys $(\mathsf{sk}_i)_{i \in \mathcal{C}_{\mathsf{skey}}}$ and the honest component $(\mathsf{dk}_{\mathsf{tag\text{-}f},i})_{i \in \mathcal{H}_{\mathsf{skey}}}$ to compute the partial functional keys $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i})_{i=1}^n$ for $\tilde{\mathbf{y}}$ where $(\tilde{\mathbf{y}}[i])_{i \in \mathcal{H}_{\mathsf{skey}}} := (\mathbf{y}[i])_{i \in \mathcal{H}_{\mathsf{skey}}}$ and $(\tilde{\mathbf{y}}[i])_{i \in \mathcal{C}_{\mathsf{skey}}} := \mathbf{0}$. The key derivation algorithm is the abstract algorithm $\mathsf{DKShare}^{\mathsf{abs}}$.
   - $\mathcal{S}$ uses the corrupted encryption keys $(\mathsf{ek}_i)_{i \in \mathcal{C}_{\mathsf{ekey}}}$ and the honest challenge ciphertext components $(\mathsf{ct}_{\mathsf{tag},i})_{i \in \mathcal{H}_{\mathsf{ekey}}}$ to compute the ciphertext components $(\tilde{\mathsf{ct}}_{\mathsf{tag},i})_{i=1}^n$ for $\tilde{\mathbf{x}}$ where implicitly $(\tilde{\mathbf{x}}[i])_{i \in \mathcal{H}_{\mathsf{ekey}}} := (\mathbf{x}_b^*[i])_{i \in \mathcal{H}_{\mathsf{ekey}}}$ and $(\tilde{\mathbf{x}}[i])_{i \in \mathcal{C}_{\mathsf{ekey}}} := \mathbf{0}$. The encryption is done using the abstract algorithm $\mathsf{Enc}^{\mathsf{abs}}$.
   - By using $\mathsf{DKeyComb}^{\mathsf{abs}}$ to combine the newly generated key components $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i})_{i=1}^n$ and $\mathsf{Dec}^{\mathsf{abs}}$ to decrypt the newly generated challenge ciphertext $(\tilde{\mathsf{ct}}_{\mathsf{tag},i})_{i \in [n]}$, $\mathcal{S}$ outputs 1 if and only if the result is equal to $\sum_{i \in \mathcal{H}} \mathbf{x}_1^*[i]\mathbf{y}[i]$.

4. If case 2 happens:
   - If case 1 also happens, $\mathcal{S}$ operates as above.
   - Else, let $i^* \in \mathcal{C}_{\mathsf{ekey}} \setminus \mathcal{C}_{\mathsf{skey}}$ s.t $(\mathbf{x}^{(0)}[i^*] - \mathbf{x}^{(1)}[i^*])\mathbf{y}[i^*] \neq 0$.
   - $\mathcal{S}$ uses the corrupted encryption keys $(\mathsf{ek}_i)_{i \in \mathcal{C}_{\mathsf{ekey}} \setminus \{i^*\}}$ and the honest challenge ciphertext components $(\mathsf{ct}_{\mathsf{tag},i})_{i \in \mathcal{H}_{\mathsf{ekey}}}$ as well as $\mathsf{ct}_{\mathsf{tag},i^*}$ to compute the ciphertext components $(\tilde{\mathsf{ct}}_{\mathsf{tag},i})_{i=1}^n$ for $\tilde{\mathbf{x}}$ where implicitly $(\tilde{\mathbf{x}}[i])_{i \in \mathcal{H}_{\mathsf{ekey}}} := (\mathbf{x}_b^*[i])_{i \in \mathcal{H}_{\mathsf{ekey}}}$, $(\tilde{\mathbf{x}}[i])_{i \in \mathcal{C}_{\mathsf{ekey}} \setminus \{i^*\}} := \mathbf{0}$, and implicitly $\tilde{\mathbf{x}}[i^*] := \mathbf{x}_b^*[i^*]$. The encryption is done using the abstract algorithm $\mathsf{Enc}^{\mathsf{abs}}$.
   - $\mathcal{S}$ uses the corrupted secret keys $(\mathsf{sk}_i)_{i \in \mathcal{C}_{\mathsf{skey}}}$ and the honest component $(\mathsf{dk}_{\mathsf{tag\text{-}f},i})_{i \in \mathcal{H}_{\mathsf{skey}}}$ to compute the partial functional keys $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i})_{i=1}^n$ for $\tilde{\mathbf{y}}$ where $(\tilde{\mathbf{y}}[i])_{i \in \mathcal{H}_{\mathsf{skey}}} := (\mathbf{y}[i])_{i \in \mathcal{H}_{\mathsf{skey}}}$ and $(\tilde{\mathbf{y}}[i])_{i \in \mathcal{C}_{\mathsf{skey}}} := \mathbf{0}$. We notice that implicitly $\tilde{\mathbf{y}}[i^*] := \mathbf{y}[i^*]$ because $i^* \in \mathcal{C}_{\mathsf{ekey}} \setminus \mathcal{C}_{\mathsf{skey}}$. The key derivation algorithm is the abstract algorithm $\mathsf{DKShare}^{\mathsf{abs}}$.
   - By using $\mathsf{DKeyComb}^{\mathsf{abs}}$ to combine the newly generated key components $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i})_{i=1}^n$ and $\mathsf{Dec}^{\mathsf{abs}}$ to decrypt the newly generated challenge ciphertext $(\tilde{\mathsf{ct}}_{\mathsf{tag},i})_{i \in [n]}$, $\mathcal{S}$ outputs 1 if and only if the result is equal to $\mathbf{x}_1^*[i^*]\mathbf{y}[i^*]$.

5. If case 3 happens:
   - If case 1 or case 2 also happens, $\mathcal{S}$ operates as above.
   - Else, let $i^* \in \mathcal{C}_{\mathsf{skey}}$ s.t $\mathbf{x}_0^*[i^*] \neq \mathbf{x}_1^*[i^*]$.
   - $\mathcal{S}$ uses the corrupted secret keys $(\mathsf{sk}_i)_{i \in \mathcal{C}_{\mathsf{skey}} \setminus \{i^*\}}$ and the honest component $(\mathsf{dk}_{\mathsf{tag\text{-}f},i})_{i \in \mathcal{H}_{\mathsf{skey}}}$ to compute the partial functional keys $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i})_{i=1}^n$ for $\tilde{\mathbf{y}}$ where $(\tilde{\mathbf{y}}[i])_{i \in \mathcal{H}_{\mathsf{skey}}} := (\mathbf{y}[i])_{i \in \mathcal{H}_{\mathsf{skey}}}$ and $(\tilde{\mathbf{y}}[i])_{i \in \mathcal{C}_{\mathsf{skey}} \setminus \{i^*\}} := \mathbf{0}$. If $\mathbf{y}[i^*] = 0$ the distinguisher $\mathcal{S}$ sets $\tilde{\mathbf{y}}[i^*] = z$ for some $B' > z \neq 0$ and uses sk_{i^*} to compute $\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i^*}$. The key derivation algorithm is the abstract algorithm $\mathsf{DKShare}^{\mathsf{abs}}$.
   - $\mathcal{S}$ uses the corrupted encryption keys $(\mathsf{ek}_i)_{i \in \mathcal{C}_{\mathsf{ekey}} \setminus \{i^*\}}$ and the honest challenge ciphertext components $(\mathsf{ct}_{\mathsf{tag},i})_{i \in \mathcal{H}_{\mathsf{ekey}}}$ as well as $\mathsf{ct}_{\mathsf{tag},i^*}$ to compute the ciphertext components $(\tilde{\mathsf{ct}}_{\mathsf{tag},i})_{i=1}^n$ for $\tilde{\mathbf{x}}$ where implicitly $(\tilde{\mathbf{x}}[i])_{i \in \mathcal{H}_{\mathsf{ekey}}} := (\mathbf{x}_b^*[i])_{i \in \mathcal{H}_{\mathsf{ekey}}}$, $(\tilde{\mathbf{x}}[i])_{i \in \mathcal{C}_{\mathsf{ekey}} \setminus \{i^*\}} := \mathbf{0}$, and implicitly $\tilde{\mathbf{x}}[i^*] := \mathbf{x}_b^*[i^*]$. The encryption is done using the abstract algorithm $\mathsf{Enc}^{\mathsf{abs}}$.

– By using $\mathsf{DKeyComb}^{\mathsf{abs}}$ to combine the newly generated key components $(\tilde{\mathsf{dk}}_{\mathsf{tag\text{-}f},i})_{i=1}^{n}$ and $\mathsf{Dec}^{\mathsf{abs}}$ to decrypt the newly generated challenge ciphertext $(\tilde{\mathsf{ct}}_{\mathsf{tag},i})_{i\in[n]}$, $\mathcal{S}$ outputs 1 if and only if the result is equal to $\mathbf{x}_1^*[i^*]\mathbf{y}[i^*]$.

The condition in step 2 can be checked efficiently as $\mathcal{A}$ is ppt and $\mathcal{Q}$ must thus be of polynomial size. It can be verified that in all three cases the distinguisher $\mathcal{S}$ outputs 1 if and only if the attack corresponds to an execution in which the challenge picks 1 as the challenge bit, for *any* DMCFE scheme $\mathcal{E}'$. The proof is completed.

Remark 5.14. As a corollary the admissibility's optimality for the class of inner products (including for polynomially bounded ranges that is proved in Theorem 5.13 above), we have specific conditions for admissible attacks in this case:

1. For all vectors $(\mathbf{x}_0^*, \mathbf{x}_1^*)$ that is queried to **LoR**, for all $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$, $\sum_{i\in\mathcal{H}} \Delta\mathbf{x}[i]\mathbf{y}[i] = 0$ where $\Delta\mathbf{x}[i] = \mathbf{x}_0^*[i] - \mathbf{x}_1^*[i]$, where $\mathcal{H} \coloneqq \mathcal{H}_{\mathsf{ekey}} \cap \mathcal{H}_{\mathsf{skey}}$.
2. For all vectors $(\mathbf{x}_0^*, \mathbf{x}_1^*)$ that is queried to **LoR**, for all $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$, for all $i \in \mathcal{C}_{\mathsf{ekey}} \setminus \mathcal{C}_{\mathsf{skey}}$, either $\mathbf{x}_0^*[i] = \mathbf{x}_1^*[i] = 0$ or $\mathbf{y}[i] = 0$.
3. For all vectors $(\mathbf{x}_0^*, \mathbf{x}_1^*)$ that is queried to **LoR**, for all $(\mathsf{tag\text{-}f}, \mathbf{y}) \in \mathcal{Q}$, for all $i \in \mathcal{C}_{\mathsf{skey}}$, $\mathbf{x}_0^*[i] = \mathbf{x}_1^*[i]$.

5.4 DMCFE for Inner Products with Stronger Security

5.4.1 Basic Construction

This section presents a *decentralized* multi-client FE scheme, as defined in Section 5.3, for the function class $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$ and $F_{\mathbf{y}} : \left(\mathbb{Z}_q^*\right)^n \to \mathbb{Z}_q$ is defined as $F_{\mathbf{y}}(\mathbf{x}) \coloneqq \langle \mathbf{x}, \mathbf{y} \rangle$ where $\|\mathbf{x}\|_\infty < B$ and $\|\mathbf{y}\|_\infty < B'$, where $B, B' = \mathsf{poly}(\lambda) \in \mathbb{N}$ are polynomials. The high-level ideas are discussed in Section 5.2.4. As discussed in Remark 5.2, the parameter vector of $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$ is simply $\mathbf{y}$ of size n. Hence the number of secret keys and of encryption keys are equal to n. Our admissibility is also optimal for $\mathcal{F}_{B,B'}^{\mathsf{IP},\mathsf{poly}}$, see Theorem 5.13 in the previous Section 5.3.1. We need a full-domain hash function $\mathsf{H}_1 : \mathsf{Tag} \to \mathbb{G}_1^2$, where Tag denotes the set of tags used for ciphertext components and functional key components. In addition, we also need a hash function $\mathsf{H}_2 : \mathsf{Tag} \times \mathbb{Z}_q^n \to \mathbb{Z}_q$.

We are in the bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are written additively. The details of our DMCFE scheme go as follows:

$\mathsf{Setup}(1^\lambda)$**:** Choose n pairs of dual orthogonal bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for $i \in [n]$, where $(\mathbf{H}_i, \mathbf{H}_i^*)$ is a pair of dual bases for $(\mathbb{G}_1^6, \mathbb{G}_2^6)$. We denote the basis changing matrices for $(\mathbf{H}_i, \mathbf{H}_i^*)$ as (H_i, H_i'):

$$(\mathbf{H}_i = H_i \cdot \mathbf{T};\ \mathbf{H}_i^* = H_i' \cdot \mathbf{T}^*)_{i \in [n]}$$

where $H_i, H_i' \in \mathbb{Z}_q^{6\times 6}$ and $(\mathbf{T} = [\![I_6]\!]_1, \mathbf{T}^* = [\![I_6]\!]_2)$ are canonical bases of $(\mathbb{G}_1^6, \mathbb{G}_2^6)$, for the identity matrix I_6. Sample two full-domain hash functions $\mathsf{H}_1 : \mathsf{Tag} \to \mathbb{G}_1^2$ and $\mathsf{H}_2 : \mathsf{Tag} \times \mathbb{Z}_q^n \to \mathbb{Z}_q$. We recall that interactions are involved only in this Setup phase. For each $i \in [n]$, we write

$$\mathbf{H}_i = (\mathbf{h}_{i,1}, \mathbf{h}_{i,2}, \ldots, \mathbf{h}_{i,6}) \qquad \mathbf{H}_i^* = (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \ldots, \mathbf{h}_{i,6}^*)$$

and sample $S, U, V, T \xleftarrow{\$} (\mathbb{Z}_q^*)^n$ where $S = (s_1, \ldots, s_n)$, $U = (u_1, \ldots, u_n), V = (v_1, \ldots, v_n), T = (t_1, \ldots, t_n)$. Sample $\theta_i \xleftarrow{\$} \mathbb{Z}_q^*$ such that $\sum_{i=1}^n \theta_i = 0$. Then, sample $\zeta_1, \zeta_2, \zeta_3, \zeta_4, p_i, q_i, \alpha_i, \gamma_i, \gamma_i' \xleftarrow{\$} \mathbb{Z}_q$, for $i \in [n]$, satisfying

$$p_i\alpha_i = \zeta_1 \qquad q_i\gamma_i = \zeta_2 \qquad q_i\alpha_i = \zeta_3 \qquad p_i\gamma_i' = \zeta_4 \tag{5.4}$$

and output the *secret keys* sk_i and the *encryption keys* ek_i as follows

$$\mathsf{sk}_i \coloneqq \left(s_i\alpha_i\mathbf{h}_{i,1}^* + s_i\gamma_i\mathbf{h}_{i,2}^*,\ u_i\gamma_i'\mathbf{h}_{i,1}^* + u_i\alpha_i\mathbf{h}_{i,2}^*, \frac{v_i}{t_i}\mathbf{h}_{i,3}^* + \mathbf{h}_{i,4}^*, \theta_i\mathbf{h}_{i,6}^*\right)$$
$$\mathsf{ek}_i \coloneqq (p_iH_i^{(1)} - (\zeta_1 s_i + \zeta_4 u_i)H_i^{(4)},\ q_iH_i^{(2)} - (\zeta_2 s_i + \zeta_3 u_i)H_i^{(4)},\ t_i\mathbf{h}_{i,3} - v_i\mathbf{h}_{i,4},\ \mathbf{h}_{i,4},\ H_i^{(6)})$$

where $H_i^{(k)}$ denotes the k-th row of H_i for $i \in [n]$.

$\mathsf{DKeyGenShare}(\mathsf{sk}_i, (\mathsf{tag\text{-}f}, \mathsf{info}(\mathbf{y})), y_i)$**:** We assume that the function tag contains $\mathsf{tag\text{-}f}$ and public information about $\mathsf{info}(\mathbf{y})$. The i-th parameter is $y_i \coloneqq \mathbf{y}[i]$. Compute $\mathsf{H}_2(\mathsf{tag\text{-}f}, \mathsf{info}(\mathbf{y})) \to d_{\mathsf{tag\text{-}f},\mathbf{y}} \in \mathbb{Z}_q$. Parse

$$\mathsf{sk}_i \coloneqq \left(s_i\alpha_i\mathbf{h}_{i,1}^* + s_i\gamma_i\mathbf{h}_{i,2}^*,\ u_i\gamma_i'\mathbf{h}_{i,1}^* + u_i\alpha_i\mathbf{h}_{i,2}^*, \frac{v_i}{t_i}\mathbf{h}_{i,3}^* + \mathbf{h}_{i,4}^*, \theta_i\mathbf{h}_{i,6}^*\right) .$$

Sample $z \xleftarrow{\$} \mathbb{Z}_q$ then compute

$$\begin{aligned}\mathbf{k}_{i,\mathsf{ipfe}} &= y_i \cdot (s_i\alpha_i\mathbf{h}_{i,1}^* + s_i\gamma_i\mathbf{h}_{i,2}^*) + y_i \cdot (u_i\gamma_i'\mathbf{h}_{i,1}^* + u_i\alpha_i\mathbf{h}_{i,2}^*) + y_i \cdot \left(\frac{v_i}{t_i}\mathbf{h}_{i,3}^* + \mathbf{h}_{i,4}^*\right) + d_{\mathsf{tag\text{-}f},\mathbf{y}} \cdot \theta_i\mathbf{h}_{i,6}^* \\ &= (s_iy_i\alpha_i + u_iy_i\gamma_i',\ s_iy_i\gamma_i + u_iy_i\alpha_i, \frac{v_i}{t_i}y_i,\ y_i,\ 0,\ d_{\mathsf{tag\text{-}f},\mathbf{y}}\theta_i)_{\mathbf{H}_i^*} .\end{aligned}$$

Output $\mathsf{dk}_{\mathsf{tag\text{-}f},i} \coloneqq \mathbf{k}_{i,\mathsf{ipfe}}$.

$\mathsf{DKeyComb}((\mathsf{dk}_{\mathsf{tag\text{-}f},i})_{i\in[n]}, \mathsf{tag\text{-}f}, \mathbf{y})$**:** Output $\perp$ if there is any incoherence of $d_{\mathsf{tag\text{-}f},\mathbf{y}}$ among the $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$. Otherwise, parse $\mathsf{dk}_{\mathsf{tag\text{-}f},i} \coloneqq \mathbf{k}_{i,\mathsf{ipfe}}$ and output $\mathsf{dk}_{\mathsf{tag\text{-}f},\mathbf{y}} \coloneqq (\mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]}$.

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i)$**:** Parse

$$\mathsf{ek}_i \coloneqq (p_i H_i^{(1)} - (\zeta_1 s_i + \zeta_4 u_i) H_i^{(4)},\ q_i H_i^{(2)} - (\zeta_2 s_i + \zeta_3 u_i) H_i^{(4)},\ t_i \mathbf{h}_{i,3} - v_i \mathbf{h}_{i,4},\ \mathbf{h}_{i,4},\ H_i^{(6)})$$

and compute $\mathsf{H}_1(\mathsf{tag}) \to (\llbracket \omega \rrbracket_1, \llbracket \omega' \rrbracket_1) \in \mathbb{G}_1^2$ and sample $r_i \xleftarrow{\$} \mathbb{Z}_q$. Compute

$$\begin{aligned}\mathbf{c}_{i,\mathsf{ipfe}} &= (p_i H_i^{(1)} - (\zeta_1 s_i + \zeta_4 u_i) H_i^{(4)}) \cdot \llbracket \omega \rrbracket_1 + (q_i H_i^{(2)} - (\zeta_2 s_i + \zeta_3 u_i) H_i^{(4)}) \cdot \llbracket \omega' \rrbracket_1 \\ &\quad + r_i \cdot (t_i \mathbf{h}_{i,3} - v_i \mathbf{h}_{i,4}) + x_i \mathbf{h}_{i,4} + H_i^{(6)} \llbracket \omega \rrbracket_1 \\ &= (\omega p_i,\ \omega' q_i,\ r_i t_i, -(\omega\zeta_1 + \omega'\zeta_2) \cdot s_i - (\omega'\zeta_3 + \omega\zeta_4) \cdot u_i + x_i - r_i v_i,\ 0,\ \omega)_{\mathbf{H}_i}\ .\end{aligned}$$

and output $\mathsf{ct}_{\mathsf{tag},i} \coloneqq \mathbf{c}_{i,\mathsf{ipfe}}$.

$\mathsf{Dec}(\mathsf{dk}_{\mathsf{tag\text{-}f},\mathbf{y}}, \mathbf{c})$**:** Parse $\mathsf{dk}_{\mathsf{tag\text{-}f},\mathbf{y}} = (\mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]}$ and $\mathbf{c} \coloneqq (\mathsf{ct}_{\mathsf{tag},i})_i$. Finally, compute the discrete logarithm in base g_t of $\llbracket \mathsf{out} \rrbracket_\mathsf{t} = \sum_{i=1}^n (\mathsf{ct}_{\mathsf{tag},i} \times \mathbf{k}_{i,\mathsf{ipfe}})$ and output the small value out.

The *correctness* of the scheme is verified by:

$$\begin{aligned}&\llbracket \mathsf{out} \rrbracket_\mathsf{t} \\ &= \sum_{i=1}^n (\mathsf{ct}_{\mathsf{tag},i} \times \mathbf{k}_{i,\mathsf{ipfe}}) \\ &= \sum_{i=1}^n \begin{pmatrix} (\omega p_i,\ \omega' q_i,\ r_i t_i, -(\omega\zeta_1 + \omega'\zeta_2) \cdot s_i - (\omega'\zeta_3 + \omega\zeta_4) \cdot u_i + x_i - r_i v_i, \\ 0,\ \omega)_{\mathbf{H}_i} \\ \times \\ (s_i y_i \alpha_i + u_i y_i \gamma_i',\ s_i y_i \gamma_i + u_i y_i \alpha_i, \frac{v_i}{t_i} y_i,\ y_i, \\ 0,\ d_{\mathsf{tag\text{-}f},\mathbf{y}} \theta_i)_{\mathbf{H}_i^*} \end{pmatrix} \\ &\overset{(*)}{=} \sum_{i=1}^n \llbracket \omega\zeta_1 s_i y_i + \omega\zeta_4 u_i y_i + \omega'\zeta_2 s_i y_i + \omega'\zeta_3 u_i y_i + \theta_i d_{\mathsf{tag\text{-}f},\mathbf{y}} \omega \rrbracket_\mathsf{t} \\ &\quad + \sum_{i=1}^n \llbracket (-(\omega\zeta_1 + \omega'\zeta_2) \cdot s_i - (\omega'\zeta_3 + \omega\zeta_4) \cdot u_i + x_i) \cdot y_i \rrbracket_\mathsf{t} \\ &= \llbracket (\omega\zeta_1 + \omega'\zeta_2) \cdot \langle S, \mathbf{y} \rangle + (\omega'\zeta_3 + \omega\zeta_4) \cdot \langle U, \mathbf{y} \rangle \rrbracket_\mathsf{t} + \sum_{i=1}^n \llbracket \theta_i d_{\mathsf{tag\text{-}f},\mathbf{y}} \omega \rrbracket_\mathsf{t} \\ &\quad - \llbracket (\omega\zeta_1 + \omega'\zeta_2) \cdot \langle S, \mathbf{y} \rangle + (\omega'\zeta_3 + \omega\zeta_4) \cdot \langle U, \mathbf{y} \rangle \rrbracket_\mathsf{t} + \llbracket \langle \mathbf{x}, \mathbf{y} \rangle \rrbracket_\mathsf{t} \\ &= \llbracket \langle \mathbf{x}, \mathbf{y} \rangle \rrbracket_\mathsf{t}\ ,\end{aligned}$$

where the equality $(*)$ comes from system (5.4). We recall that $(\theta_i)_{i\in[n]}$ is a secret sharing of 0.

5.4.2 Adaptive Security against Incomplete Queries and Static Corruptions of Secret Keys

We now give the security theorem of one-time IND-security for our construction from Section 5.4.1, *adaptively* in the challenge messages with *dynamic* corruption of encryption keys and *static* corruption of secret keys. We refer to Remark 5.14 for the concrete interpretation of the security model. The full proof can be found in [NPP23b, Theorem 16, Appendix C.2].

Theorem 5.15. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$ *be a DMCFE candidate for* $\mathcal{F}^{\mathsf{IP}}$ *from Section 5.4.1 in a bilinear group* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$*. Then,* $\mathcal{E}$ *is IND-secure with static corruption of secret keys in the ROM if the* SXDH *assumption holds for* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*.*

More specifically, let n denote the dimension for inner-products, K denote the maximum number of key queries, and Q_1, Q_2 denote the maximum number of random oracle (RO) queries to $\mathsf{H}_1, \mathsf{H}_2$ respectively. For any ppt *adversary $\mathcal{A}$ against $\mathcal{E}$ with static secret key corruption and one-time challenge, we have the following bound:*

$$\mathsf{Adv}^{\mathsf{dmc\text{-}stat\text{-}sk\text{-}ind\text{-}cpa\text{-}1chal}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) \leq (3 + 2Q_1 + K) \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda) + \frac{Q_2^2}{2q}$$

and in the reduction there is an additive loss $\mathcal{O}(Q_1 \cdot t_{\mathbb{G}_1} + Q_2 \cdot t_{\mathbb{G}_2})$ in time, where $t_{\mathbb{G}_1}, t_{\mathbb{G}_2}$ is the cost for one addition in $\mathbb{G}_1, \mathbb{G}_2$.

5.4.3 Constructions with Stronger Security against Incomplete Queries

Definitions. In this section, we show how to obtain a DMCFE scheme that is IND-secure against chosen plaintext attacks without *complete* queries restriction (for high level ideas, see condition 2 in Section 5.2.1), under our stronger admissibility notion. The definition of security notion for the new setting is restated so that admissible adversaries can query *incomplete* challenge ciphertexts as well as *incomplete* functional keys.

Definition 5.16 (Admissible attacks with incomplete queries). *Let $\mathcal{A}$ be a* ppt *adversary and let*

$$\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$$

be a DMCFE scheme for a function class $\mathcal{F}$ set up w.r.t $\lambda \in \mathbb{N}$. We denote by $\mathsf{rand}_{\mathsf{Chall}}$ the random coins of the challenger and $\mathsf{rand}_{\mathcal{A}}$ the random coins of the adversary in an experiment given in Figure 5.1. In **Finalize**, *considering the queries $(\mathcal{Q}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{C}_{\mathsf{skey}}, \mathcal{C}_{\mathsf{ekey}}, \{(\mathbf{x}_0^*, \mathbf{x}_1^*, \mathsf{tag})\}, \{(\mathbf{x}, \mathsf{tag}^{(k)})\})$, we say that the attack corresponding to these queries is* NOT admissible *if the following is satisfied*

> *There exist $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, a function $F \in \mathcal{F}$, two challenges $(x_i^{(0)}, x_i^{(1)})_{i\in[n]}$ such that $(F, \mathsf{tag\text{-}f})$ is queried to* **DKeyGenShare** for all *honest components, $((x_i^{(0)}, x_i^{(1)})_{i\in[n]}, \mathsf{tag})$ is queried to* **LoR** for all *honest components, and there exists a pair $(\mathbf{z}^{(0)}, \mathbf{z}^{(1)})$ deducible from $(\mathbf{x}^{(0)}_{\mathsf{ekey}}, \mathbf{x}^{(1)}_{\mathsf{ekey}})$, a function G deducible from $\mathbf{y}_{\mathsf{skey}}$ satisfying*
>
> $$G(\mathbf{z}^{(0)}) \neq G(\mathbf{z}^{(1)}) \ ,$$
>
> *where we define $\mathbf{y}_{\mathsf{skey}} := (y_i)_{i\in\mathcal{H}_{\mathsf{skey}}}$ and for $b \in \{0,1\}$, $\mathbf{x}^{(b)}_{\mathsf{ekey}} := (x_i^{(b)})_{i\in\mathcal{H}_{\mathsf{ekey}}}$.*

Otherwise, we say that the attack is admissible.

Definition 5.17 (IND+-security for DMCFE). *A DMCFE scheme*

$$\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$$

for the function class $\mathcal{F} = \{F_\lambda\}_{\lambda\in\mathbb{N}}$ is IND+-secure *if for all* ppt *adversaries $\mathcal{A}$, and for all sufficiently large $\lambda \in \mathbb{N}$, the following probability is negligible*

$$\mathsf{Adv}^{\mathsf{xx}+}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) := \left|\Pr[\mathsf{Expr}^{\mathsf{xx}+}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2}\right| \ .$$

The probability is taken over coins of $\mathcal{A}$ and the algorithms. The indicator xx *can be among* $\{\mathsf{dmc\text{-}ind\text{-}cpa}, \mathsf{dmc\text{-}sel\text{-}ind\text{-}cpa}, \mathsf{dmc\text{-}stat\text{-}ind\text{-}cpa}, \mathsf{dmc\text{-}ind\text{-}cpa\text{-}1chal}\}$[6]. *The experiment $\mathsf{Expr}^{\mathsf{xx}+}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathsf{Expr}^{\mathsf{xx}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ depicted in Figure 5.1, except that we use Definition 5.16 for the* admissibility *condition in* **Finalize**.

[6] Similarly, we can allow dynamic corruption on one type but static corruption on the other type of keys, such as dmc-stat-sk-ind-cpa+ to indicate *partially static IND-security* with adaptive challenges, dynamic corruption of ekey, and static corruption of skey.

Generic Transformation with Security against Selective Challenges. We follow the same method in [CDSG+20] and apply generically a layer of using a primitive called *All-or-Nothing Encapsulation* (AoNE), so as to make our scheme from Section 5.4.1 secure in our *stronger* security model against incomplete queries. Our AoNE-based transformation uses the generic AoNE from [CDSG+20], which in turn is built on top of a *one-time secure symmetric encryption* (OT-SE). In the security proof, which can be naturally adapted from [CDSG+20, Theorem 26], this OT-SE prevents programing conveniently to achieve adaptive security w.r.t the challenge ciphertexts. We also remark that the static corruption is unavoidable since the security of AoNE makes sense only when being applied on honest components, for the security reduction. This generic transformation is provably secure under *static* corruption and *selective* challenges. The transformation is presented in [NPP23b, Appendix B.2].

Concrete Scheme with Security against Adaptive Challenges. We present a concrete adaptation of our base DMCFE scheme from Section 5.4.1 to satisfy the stronger security notion against *incomplete* challenge ciphertexts as well as *incomplete* functional keys, with minimal modifications being put in boxed components for the ease of comparison. The function class stays the same as in Section 5.4.1, for which our admissibility is optimal (*as per* Theorem 5.13). In contrast to the generic transformation, we build the AoNE concretely by combining one-time pad (OTP) and a random oracle (RO). Then, the programmability of the RO helps us circumvent the problem of adaptive queries. While programming the RO, we indeed exploits in a non-blackbox manner the OTP as a summation in $\mathbb{Z}_q^*$ to accumulate a secret sharing of 0 on the honest parts (known in advance thanks to *static* corruption).

The details of our construction go as follows:

$\mathsf{Setup}(1^\lambda)$**:** Sample two full-domain hash functions $\mathsf{H}_1 : \mathsf{Tag} \to \mathbb{G}_1^2$ and $\mathsf{H}_2 : \mathsf{Tag} \times \mathbb{Z}_q^n \to \mathbb{G}_2$. Choose n pairs of dual orthogonal bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ for $i \in [n]$, where $(\mathbf{H}_i, \mathbf{H}_i^*)$ is a pair of dual bases for $\boxed{(\mathbb{G}_1^8, \mathbb{G}_2^8)}$. We denote the basis changing matrices for $(\mathbf{H}, \mathbf{H}_i^*)$ as (H_i, H_i'):

$$(\mathbf{H}_i = H_i \cdot \mathbf{T};\ \mathbf{H}_i^* = H_i' \cdot \mathbf{T}^*)_{i\in[n]}$$

where $H_i, H_i' \in \mathbb{Z}_q^{8\times 8}$ and $(\mathbf{T} = [\![I_8]\!]_1, \mathbf{T}^* = [\![I_8]\!]_2)$ are canonical bases of $(\mathbb{G}_1^8, \mathbb{G}_2^8)$, for the identity matrix I_8. We recall that interactions are involved only in this Setup phase. For each $i \in [n]$, we write

$$\boxed{\mathbf{H}_i = (\mathbf{h}_{i,1}, \mathbf{h}_{i,2}, \ldots, \mathbf{h}_{i,8}) \quad \mathbf{H}_i^* = (\mathbf{h}_{i,1}^*, \mathbf{h}_{i,2}^*, \ldots, \mathbf{h}_{i,8}^*)}$$

and sample $\zeta_1, \zeta_2, \zeta_3, \zeta_4, S, U, V, T, \boxed{D, E} \xleftarrow{\$} (\mathbb{Z}_q^*)^n$ where $S = (s_1, \ldots, s_n)$, $U = (u_1, \ldots, u_n), V = (v_1, \ldots, v_n), T = (t_1, \ldots, t_n), D = (d_1, \ldots, d_n)$, and $E = (e_1, \ldots, e_n)$. Sample $\theta_1, \ldots, \theta_n \xleftarrow{\$} \mathbb{Z}_q^*$ such that $\sum_{i=1}^n \theta_i = 0$ and for $i \in [n]$ let $p_i, q_i, \alpha_i, \gamma_i, \gamma_i' \xleftarrow{\$} \mathbb{Z}_q$ satisfy

$$p_i\alpha_i = \zeta_1 \qquad q_i\gamma_i = \zeta_2 \qquad q_i\alpha_i = \zeta_3 \qquad p_i\gamma_i' = \zeta_4$$

We set the public parameters to be $\boxed{([\![\langle E, \mathbf{1}\rangle]\!]_1, [\![\langle D, \mathbf{1}\rangle]\!]_2)}$. Sample $\boxed{\epsilon, \delta} \xleftarrow{\$} \mathbb{Z}_q$ and generate random n-out-of-n secret sharings $\boxed{(\epsilon_i)_i, (\delta_i)_i}$ of ϵ, δ so that $\sum_{i=1}^n \epsilon_i = \epsilon, \sum_{i=1}^n \delta_i = \delta$. Output the *secret keys* sk_i and the *encryption keys* ek_i as follows

$$\mathsf{sk}_i := (\boxed{\epsilon_i}, s_i\alpha_i\mathbf{h}_{i,1}^* + s_i\gamma_i\mathbf{h}_{i,2}^*,\ u_i\gamma_i'\mathbf{h}_{i,1}^* + u_i\alpha_i\mathbf{h}_{i,2}^*, \frac{v_i}{t_i}\mathbf{h}_{i,3}^* + \mathbf{h}_{i,4}^*, \boxed{\theta_i H_i'^{(6)} - e_i H_i'^{(8)}, \epsilon\mathbf{h}_{i,8}})$$

$$\mathsf{ek}_i := (\boxed{\delta_i}, p_i H_i^{(1)} - (\zeta_1 s_i + \zeta_4 u_i) H_i^{(4)} - \boxed{d_i H_i^{(7)}},\ q_i H_i^{(2)} - (\zeta_2 s_i + \zeta_3 u_i) H_i^{(4)}, t_i\mathbf{h}_{i,3} - v_i\mathbf{h}_{i,4},\ \mathbf{h}_{i,4},\ H_i^{(6)}, \boxed{\delta\mathbf{h}_{i,7}^*})$$

where $H_i^{(k)}$ denotes the k-th row of H_i for $i \in [n]$ and $\mathbf{1} = (1, \ldots, 1)$.

$\mathsf{DKeyGenShare}(\mathsf{sk}_i, (\mathsf{tag\text{-}f}, \mathsf{info}(\mathbf{y})), y_i)$**:** We assume that the function tag contains $\mathsf{tag\text{-}f}$ and public information about $\mathsf{info}(\mathbf{y})$. The i-th parameter is $y_i \coloneqq \mathbf{y}[i]$. We will use a full-domain hash function $\mathsf{H}_2 : \mathsf{Tag} \times \mathbb{Z}_q^n \to \mathbb{G}_2$. Parse

$$\mathsf{sk}_i \coloneqq (\epsilon_i, s_i\alpha_i\mathbf{h}^*_{i,1} + s_i\gamma_i\mathbf{h}^*_{i,2},\ u_i\gamma'_i\mathbf{h}^*_{i,1} + u_i\alpha_i\mathbf{h}^*_{i,2}, \frac{v_i}{t_i}\mathbf{h}^*_{i,3} + \mathbf{h}^*_{i,4}, \theta_i {H'}_i^{(6)} - e_i {H'}_i^{(8)}, \epsilon\mathbf{h}_{i,8}) \ .$$

Compute $\mathsf{H}_2(\mathsf{tag\text{-}f}, \mathsf{info}(\mathbf{y})) \to [\![\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_2$ and

$$\begin{aligned}
&\mathbf{k}_{i,\mathsf{ipfe}} \\
&= y_i \cdot (s_i\alpha_i\mathbf{h}^*_{i,1} + s_i\gamma_i\mathbf{h}^*_{i,2}) + y_i \cdot (u_i\gamma'_i\mathbf{h}^*_{i,1} + u_i\alpha_i\mathbf{h}^*_{i,2}) \\
&\quad + y_i(\tfrac{v_i}{t_i}\mathbf{h}^*_{i,3} + \mathbf{h}^*_{i,4}) + (\theta_i {H'}_i^{(6)} - e_i {H'}_i^{(8)}) \cdot [\![\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_2 \\
&= (s_iy_i\alpha_i + u_iy_i\gamma'_i,\ s_iy_i\gamma_i + u_iy_i\alpha_i, \tfrac{v_i}{t_i}y_i,\ y_i,\ 0,\ \kappa_{\mathsf{tag\text{-}f},\mathbf{y}}\theta_i, \boxed{0, -e_i\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}})_{\mathbf{H}^*_i} \\
&\boxed{\mathbf{e}(\epsilon_i \cdot [\![\langle E, \mathbf{1}\rangle]\!]_1, [\![\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_2)} = [\![\epsilon_i\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t}
\end{aligned}$$

where $[\![\langle E, \mathbf{1}\rangle]\!]_1$ is public. Output $\mathsf{dk}_{\mathsf{tag\text{-}f},i} \coloneqq (\mathbf{k}_{i,\mathsf{ipfe}}, \boxed{\epsilon \cdot \mathbf{h}_{i,8}, [\![\epsilon_i\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t}})$.

$\mathsf{DKeyComb}(\mathsf{dk}_{\mathsf{tag\text{-}f},i}, \mathsf{tag\text{-}f}, \mathbf{y})$**:** Output $\perp$ if there is any incoherence among the $\mathsf{dk}_{\mathsf{tag\text{-}f},i}$. Else, let $\mathsf{dk}_{\mathsf{tag\text{-}f},i} \coloneqq (\mathbf{k}_{i,\mathsf{ipfe}}, \epsilon \cdot \mathbf{h}_{i,8}, [\![\epsilon_i\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t})$.
Compute $\boxed{[\![\epsilon\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t}} = \sum_{i=1}^n [\![\epsilon_i\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t}$ and output

$$\mathsf{dk}_{\mathsf{tag\text{-}f},\mathbf{y}} \coloneqq ((\mathbf{k}_{i,\mathsf{ipfe}}, \boxed{\epsilon \cdot \mathbf{h}_{i,8}})_{i\in[n]}, \boxed{[\![\epsilon\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t}}) \ .$$

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i)$**:** Parse

$$\mathsf{ek}_i \coloneqq (\delta_i,\ p_iH_i^{(1)} - (\zeta_1s_i + \zeta_4u_i)H_i^{(4)} - d_iH_i^{(7)},\ q_iH_i^{(2)} - (\zeta_2s_i + \zeta_3u_i)H_i^{(4)}, t_i\mathbf{h}_{i,3} - v_i\mathbf{h}_{i,4},\ \mathbf{h}_{i,4}, H_i^{(6)}, \delta \cdot \mathbf{h}^*_{i,7})$$

and compute $\mathsf{H}_1(\mathsf{tag}) \to ([\![\omega]\!]_1, [\![\omega']\!]_1) \in \mathbb{G}_1^2$; and sample $r_i \xleftarrow{\$} \mathbb{Z}_q$. Compute

$$\begin{aligned}
\mathbf{c}_{i,\mathsf{ipfe}} &= (p_iH_i^{(1)} - (\zeta_1s_i + \zeta_4u_i)H_i^{(4)} - d_iH_i^{(7)}) \cdot [\![\omega]\!]_1 + (q_iH_i^{(2)} - (\zeta_2s_i + \zeta_3u_i)H_i^{(4)}) \cdot [\![\omega']\!]_1 \\
&\quad + r_i \cdot (t_i\mathbf{h}_{i,3} - v_i\mathbf{h}_{i,4}) + x_i\mathbf{h}_{i,4} + H_i^{(6)}\,[\![\omega]\!]_1 \\
&= (\omega p_i,\ \omega' q_i,\ r_it_i, -(\omega\zeta_1 + \omega'\zeta_2) \cdot s_i - (\omega'\zeta_3 + \omega\zeta_4) \cdot u_i + x_i - r_iv_i,\ 0,\ \omega, \boxed{-d_i\omega, 0})_{\mathbf{H}_i} \\
\boxed{\mathbf{e}([\![\omega]\!]_2, \delta_i \cdot [\![\langle D, \mathbf{1}\rangle]\!]_2)} &= [\![\delta_i\langle D, \mathbf{1}\rangle\omega]\!]_\mathsf{t}
\end{aligned}$$

where $[\![\langle D, \mathbf{1}\rangle]\!]_2$ comes from the public parameters.
Output $\mathsf{ct}_{\mathsf{tag},i} \coloneqq (\mathbf{c}_{i,\mathsf{ipfe}}, \boxed{\delta \cdot \mathbf{h}^*_{i,7}, [\![\delta_i\langle D, \mathbf{1}\rangle\omega]\!]_\mathsf{t}})$.

$\mathsf{Dec}(\mathsf{dk}_{\mathsf{tag\text{-}f},\mathbf{y}}, \mathbf{c})$**:** Parse

$$\begin{aligned}
\mathsf{dk}_{\mathsf{tag\text{-}f},\mathbf{y}} &= ((\mathbf{k}_{i,\mathsf{ipfe}}, \boxed{\epsilon \cdot \mathbf{h}_{i,8}})_i, \boxed{[\![\epsilon\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t}}); \\
\mathbf{c} &= (\mathbf{c}_{i,\mathsf{ipfe}}, \boxed{\delta \cdot \mathbf{h}^*_{i,7}, [\![\delta_i\langle D, \mathbf{1}\rangle\omega]\!]_\mathsf{t}})_{i=1}^n
\end{aligned}$$

Compute $\boxed{[\![\delta\langle D, \mathbf{1}\rangle\omega]\!]_\mathsf{t}} = \sum_{i=1}^n [\![\delta_i\langle D, \mathbf{1}\rangle\omega]\!]_\mathsf{t}$ and

$$[\![\mathsf{out}]\!]_\mathsf{t} = \sum_{i=1}^n \left((\mathsf{ct}_{\mathsf{tag},i} + \boxed{\epsilon \cdot \mathbf{h}_{i,8}}) \times (\mathbf{k}_{i,\mathsf{ipfe}} + \boxed{\delta \cdot \mathbf{h}^*_{i,7}})\right) + \boxed{[\![\epsilon\langle E, \mathbf{1}\rangle\kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_\mathsf{t} + [\![\delta\langle D, \mathbf{1}\rangle\omega]\!]_\mathsf{t}}.$$

Finally, compute the discrete logarithm and output the small value out.

The *correctness* of the scheme is verified by:

$$
\begin{aligned}
&[\![\mathsf{out}]\!]_{\mathsf{t}} \\
&= \sum_{i=1}^{n} \Big((\mathbf{k}_{i,\mathsf{ipfe}} + \delta \cdot \mathbf{h}^*_{i,7}) \times (\mathsf{ct}_{\mathsf{tag},i} + \epsilon \cdot \mathbf{h}_{i,8}) \Big) + [\![\epsilon \langle E, \mathbf{1} \rangle \kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_{\mathsf{t}} + [\![\delta \langle D, \mathbf{1} \rangle \omega]\!]_{\mathsf{t}} \\
&= \sum_{i=1}^{n} \left(\begin{array}{c} (s_i y_i \alpha_i + u_i y_i \gamma'_i,\ s_i y_i \gamma_i + u_i y_i \alpha_i, \frac{v_i}{t_i} y_i,\ y_i, \\ 0,\ \kappa_{\mathsf{tag\text{-}f},\mathbf{y}} \theta_i, \delta, -e_i \kappa_{\mathsf{tag\text{-}f},\mathbf{y}}) \mathbf{H}^*_i \\ \times \\ (\omega p_i,\ \omega' q_i,\ r_i t_i, -(\omega\zeta_1 + \omega'\zeta_2) \cdot s_i - (\omega'\zeta_3 + \omega\zeta_4) \cdot u_i + x_i - r_i v_i, \\ 0,\ \omega, -d_i \omega, \epsilon)_{\mathbf{H}_i} \end{array} \right) \\
&\quad + [\![\epsilon \langle E, \mathbf{1} \rangle \kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_{\mathsf{t}} + [\![\delta \langle D, \mathbf{1} \rangle \omega]\!]_{\mathsf{t}} \\
&\overset{(*)}{=} \sum_{i=1}^{n} [\![\omega\zeta_1 s_i y_i + \omega\zeta_4 u_i y_i + \omega'\zeta_2 s_i y_i + \omega'\zeta_3 u_i y_i + \theta_i \omega \kappa_{\mathsf{tag\text{-}f},\mathbf{y}}]\!]_{\mathsf{t}} \\
&\quad + \sum_{i=1}^{n} [\![(-(\omega\zeta_1 + \omega'\zeta_2) \cdot s_i - (\omega'\zeta_3 + \omega\zeta_4) \cdot u_i + x_i) \cdot y_i]\!]_{\mathsf{t}} \\
&\quad + \sum_{i=1}^{n} [\![(-(\omega\zeta_1 + \omega'\zeta_2) \cdot s_i - (\omega'\zeta_3 + \omega\zeta_4) \cdot u_i + x_i) \cdot y_i]\!]_{\mathsf{t}} \\
&= [\![\langle \mathbf{x}, \mathbf{y} \rangle]\!]_{\mathsf{t}} \quad .
\end{aligned}
$$

where the equality $(*)$ comes from system (5.4). We recall that $(\theta_i)_{i \in [n]}$ is a secret sharing of 0.

Security. We prove the one-time static security of our DMCFE scheme in the ROM, where the full-domain hash functions are modeled as random oracles, the sets of corrupted clients $\mathcal{C}_{\mathsf{ekey}}$ as well as $\mathcal{C}_{\mathsf{skey}}$ must be sent up front (*static* corruption), while the challenges $(\mathbf{x}^*_0, \mathbf{x}^*_1)$ can be adaptively chosen (*adaptive* challenge). We note that we can achieve a better level of security in our concrete instantiation compared to the generic transformation. On one hand, our transformation follows the same blueprint in the work by Chotard *et al.* [CDSG+20], which is the most relevant to our DMCFE setting. We apply a layer of *All-or-Nothing Encapsulation* (AoNE) to our ciphertext and key components, which ensures that the orignal key/ciphertext components can be recovered only when all parts are gathered. Our concrete DMCFE in Section 5.4.3 builds the AoNE directly by combining one-time pad (OTP) and a random oracle (RO). Then, the programmability of the RO helps us circumvent the problem of adaptive queries. While programming the RO, we indeed exploits in a non-blackbox manner the OTP as a summation in $\mathbb{Z}^*_q$ to accumulate a secret sharing of 0 on the honest parts (known in advance thanks to static corruption).

Theorem 5.18. *Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGenShare}, \mathsf{DKeyComb}, \mathsf{Enc}, \mathsf{Dec})$ be the DMCFE constructed in Section 5.4.3. Then, $\mathcal{E}$ is one-time statically IND+-secure in the ROM following the security model in Definition 5.17 if the SXDH and DBDH assumptions hold for $\mathbb{G}_1$ and $\mathbb{G}_2$. More specifically, let n denote the dimension for inner-products, Q_1, Q_2 denote the maximum number of random oracle (RO) queries to $\mathsf{H}_1, \mathsf{H}_2$ and K denote the total number of functional key queries. For any one-time challenge ppt adversary $\mathcal{A}$ against $\mathcal{E}$ with static corruption of secret keys and encryption keys, we have the following bound:*

$$
\mathsf{Adv}^{\mathsf{dmc\text{-}stat\text{-}1chal+}}_{\mathcal{E},\mathcal{F}^{\mathsf{IP}},\mathcal{A}}(1^\lambda) \leq (K+1)\mathsf{Adv}^{\mathsf{DBDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda) + (3 + 2Q_1 + K)\mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda) + \frac{Q_2^2}{2q} \quad .
$$

Details can be found in [NPP23b, Appendix C.3].

Part III

Further MCFE Security Extension

Chapter

6 Multi-Client Functional Encryption with Public Inputs and Strong Security

Chapter content

In Chapter 4, we observe that the challenge becomes more important when combined with access control, such as attribute-based encryption (ABE), which was actually not covered by the FE and MCFE frameworks. On the other hand, as for complex primitives, many works including some results of this thesis in Chapter 5 have studied the admissibility of adversaries to ensure that the security model encompasses all real threats of attacks. The concrete function class that is studied in Chapter 5 focuses on computing inner products alone. A natural question then arises: *how to study the optimal admissibility for function classes that capture access control?*

In this chapter, adding a public input to FE/MCFE, we cover many previous primitives, notably attribute-based function classes. Furthermore, with the strongest admissibility for inner-product functionality, our framework is quite versatile, as it encrypts multiple sub-vectors, allows repetitions and corruptions, and eventually also encompasses public-key FE and classical ABE, bridging the private setting of MCFE with the public setting of FE and ABE.

Finally, we propose an MCFE with public inputs with the class of functions that combines inner-products (on private inputs) and attribute-based access-control (on public inputs) for LSSS policies. We achieve the first AB-MCFE for inner-products with strong admissibility and with adaptive security. This also leads to MIFE for inner products, public-key single-input inner-product FE with LSSS key-policy and KP-ABE for LSSS, with adaptive security while the previous AB-MCFE construction of Agrawal *et al.* from CRYPTO '23 considers a slightly larger functionality of average weighted sum but with selective security only.

6.1 Introduction and Motivation

On the reconciliation between MCFE and MIFE. At first glance, MIFE appears to be just MCFE with a constant label. However, the distinction is more significant because in MIFE, there

is only one encryptor, while in MCFE, there are multiple encryptors (clients). Therefore, whereas there is no corruption of users in MIFE, dealing with corruptions in MCFE is a main concern. In summary, recent works [NPP22a, NPP23a, AGT22, ATY23a] agree on two principal advantages of MCFE over MIFE:

- with a label associated to each encrypted input, one can limit the combinations of the inputs for each evaluation
- as inputs can be encrypted by different clients, multiple independent secrets are involved, for each client, then one can deal with corruption of individual keys in MCFE, whereas in MIFE there is a unique encryptor and no corruption can be allowed.

At this point, it seems that MCFE is strictly stronger than MIFE. However, again, the situation is more complicated because, as pointed out by [CDG+18b], in the original definition of MCFE [GGG+14, CDG+18a], the clients were assumed not to encrypt two messages under the same label. Under this restriction, one cannot turn a MCFE to a MIFE. In short, MIFE, when augmented with labels, can be seen as an MCFE *with* repetitions but *without* corruption.

But the story is not at the end yet, especially when one wants to combine MCFE and MIFE with other functionalities, such as attribute-based access-control [NPP22a, ATY23a], where the conversion MCFE to MIFE is highly non-trivial as mentioned in [ATY23a]. As a final remark, our context of multi-client/multi-input setting for FE with access control is different from the setting of *multi-authority* ABE, *e.g.* as studied in [DP23], where in our case there is always only *one* authority generating the functional decryption keys.

From Secret-key MCFE to Public-key FE. We now consider a viewpoint that is independent of the multi-user setting. Following the first formalization in [CDG+18a], many follow-up studies on MCFE, for instance [ABKW19, ABG19, LT19, CDSG+20, AGT21b], set down an *admissibility condition* in order to exclude trivial attacks: for any corrupted client i and challenge message-pair $(x_i^{(0)}, x_i^{(1)})$ for i, it requires that $x_i^{(0)} = x_i^{(1)}$. This is indeed the right condition if the secret-key encryption is *deterministic*, which was considered on the first period of development of MCFE, as with the corruption of the encryption key ek_i, the adversary could re-encrypt $x_i^{(0)}$ and compare with the challenge ciphertext. However, we argue in Chapter 5 of this thesis that if the encryption is *probabilistic*, this condition is not well justified and appears too restrictive. We refer to Section 5.2.1 for the motivations and extensive discussion on why there is a need of some less restrictive admissibility when one considers (D)MCFE with probabilistic encryption.

In the particular context of the current chapter, we observe that it is this condition where for all $i \in \mathcal{C}$ all challenge pairs $x_i^{(0)} = x_i^{(1)}$ that prevents to go from the *secret-key* MCFE to the *public-key* FE. To obtain a public-key FE from a secret-key MCFE, the natural approach is to instantiate the MCFE with $n = 1$ client, then to publish the only client's encryption key ek as the public key. Under the early admissibility condition *as per* [CDG+18a] of the underlying MCFE, in order to base the security of the public-key FE on the security of the MCFE, the only queries that the reduction can forward to its MCFE challenger are the *trivial* one from the FE adversary where $x^{(0)} = x^{(1)}$, and this is far weaker than the standard CPA-security of public-key FE. It is now clear that a less restrictive notion of admissibility, equivalently a stronger notion of security that we discuss in Chapter 5, is needed to capture the security of public-key FE from the security of MCFE.

Final Syntactical Point: Public Inputs. When reviewing the existing initial definitions of FE [BSW11], MCFE [CDG+18a], and MIFE [GGG+14], we observe that the syntax of encryption in these definitions themselves *a priori* does not allow parts of the plaintext to be public. When denoting encryption keys ek_i (in the secret-key MCFE/MIFE setting) or public key pk (in the public-key single-client FE), specifically the MIFE syntax in [GGG+14, Section 2.1] is written

$c \leftarrow \mathsf{Enc}(\mathsf{ek}_i, x)$ given the i-th plaintext x, the MCFE syntax in [CDG+18a, Definition 1] is written $c \leftarrow \mathsf{Enc}(\mathsf{ek}_i, x, \mathsf{tag})$ given the i-th plaintext x and the tag tag, and the FE syntax in [BSW11, Definition 2] is written $c \leftarrow \mathsf{Enc}(\mathsf{pk}, x)$ given the plaintext x.

First of all, having the encryption as they are listed above, the IND-CPA security alone implies that no partial information about the plaintext is leaked. This applies to the case $x = (m, \mathsf{S})$ where m is the contents of the message and S is some attribute/index in the context of KP-ABE or IBE. As such, without further specifications, how to derive *non-attribute/index-hiding* KP-ABE/IBE from the existing definitions of FE, MCFE, and MIFE is not clear. It then necessarily requires more properties on the *function class* so as to capture the non-attribute-hiding property. We emphasize that this is also the approach that was taken in [BSW11], where the authors introduced the notion of *empty key* that defines a function such that "anyone can [...] obtain all the information about x that intentionally leaks from c" [BSW11, Page 3]. A similar notion to this empty-key function is indeed what we need to capture the non-attribute-hiding property when expressing KP-ABE/IBE in the syntax of FE. With respect to [BSW11], when describing how to capture KP-ABE or Ciphertext-Policy ABE [BSW11, Page 5], the empty-key function however is not made clear in the key space of all poly-sized boolean formula in the former, nor in the key space of all poly-long bitstrings of variables in the latter. In the multi-user setting of MCFE/MIFE, no such property of empty-key function is mentioned in the introduced definitions [GGG+14, CDG+18a].

Chapter Outline. We first start in Section 6.2 with a high-level overview of our concret construction of MCFE for the inner product function class with access control by LSSS. Various technical challenges are explained and we also highlight the techniques we use to resolve them. In a nutshell, the difficulties come from both aspects at the same time: we integrate access control to the scheme of [NPP23a] (that is presented in Section 5.4.1) while trying to relax the admissibility of the AB-MCFE scheme of [NPP22a] (that is presented in Section 4.4). Techniques of DPVS the are common to both above schemes will be heavily, *e.g.* see the paragraph **Solution to the Third Obstacle**. In particular, in the **Proof Overview** for Theorem 6.11 we make clear principal steps that need DPVS and other techniques for the security proof. We hope that the **Proof Overview** for Theorem 6.11 culminates sufficiently substantial ideas on how to use DPVS in proving security, which has been the case also in Chapters 4 and 5. The formal definitions of function classes with public inputs are given in Section 6.3. Section 6.4 then presents formally our concrete construction, while casting funtion classes with access control in terms of function class with public inputs in Section 6.4.1 before diving into details of the particular case for inner products with LSSS policies in Section 6.4.2. All abridged proofs can be found in the full version [NPP25].

6.2 Technical Overview

Given the above conceptual overview, we now highlight the technical points for our concrete construction of MCFE to compute inner products under access control in Section 6.4. All relevant points that relate to the security proof are given by a dedicated overview in paragraph **Proof Strategy** before the proof of Theorem 6.11. The functionality of interest is $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ and $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B}$ contains $F_{\mathbf{y}_1,\ldots,\mathbf{y}_n} : \prod_{i\in[n]} \left(\mathbb{Z}_q^{N_i}\right) \to \mathbb{Z}_q$ that is defined as $F_{\mathbf{y}_1,\ldots,\mathbf{y}_n}(\mathbf{x}_1,\ldots,\mathbf{x}_n) \coloneqq \sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i\rangle$, where for all i, $\max(\|\mathbf{x}_i\|_\infty, \|\mathbf{y}_i\|_\infty) < B$, where $B = \mathsf{poly}(\lambda) \in \mathbb{N}$ is a polynomial. For the ease of notation, we can assume the subvectors are of length $N = \max_i(N_i)$. The access control is given by $\mathsf{Rel} : \mathsf{LSSS} \times \left(\prod_{i=1}^n 2^{\mathsf{Att}}\right) \to \{0,1\}$, where $\mathsf{Rel}(\mathbb{A}, (\mathsf{S}_i)_i) = \prod_i \mathbb{A}(\mathsf{S}_i)$, the class LSSS contains Linear Secret Sharing Schemes over Att, and 2^{Att} denotes the superset of an attribute space $\mathsf{Att} \subseteq \mathbb{Z}_q$.

First Technical Obstacle: Admissibility with vectors and probabilistic encryption. Our goal is to handle the less restrictive admissibility condition w.r.t the function calculating $F_{\mathbf{y}_1,\ldots,\mathbf{y}_n}(\mathbf{x}_1,\ldots,\mathbf{x}_n) \coloneqq \sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i\rangle$, under access control from Rel. Each of the n clients in our MCFE scheme are encrypting a vector $\mathbf{x}_i$, together with a tag tag and their set of attributes S_i. The fact that we are working with vectors is generalizing first and foremost the construction by Nguyen *et al.* [NPP22a] (Chapter 4) that only supports scalar inputs $x_i \in \mathbb{Z}_q$. Moreover, under the new admissibility that is studied in [NPP23a] (Chapter 5), the conditions for the challenge ciphertexts in terms of corrupted clients i become less restrictive. To recall, the admissibility in [NPP22a] is inherited from the orginal one introduced in [CDG+18a] and will require that for any corrupted $i \in \mathcal{C}$, it holds that $\mathbf{x}_i^{(0)} = \mathbf{x}_i^{(1)}$. Following the motivation that is put forth in [NPP23a] so as to relax the foregoing condition, in the case of scalars where inputs to clients i have dimension 1, the *stronger* admissibility condition is that for any corrupted $i \in \mathcal{C}$, for any key queries with y_i as the i-th parameter for inner products, it must hold $(x_i^{(0)} - x_i^{(1)}) \cdot y_i = 0$. In our case, having the goal of generalizing [NPP22a] (Section 4.4.2) to encrypt vectors under the *stronger* admissibility, the condition becomes: for any corrupted $i \in \mathcal{C}$, for any key queries with $\mathbf{y}_i$ as the i-th parameter for inner products, it must hold $\langle \mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)}, \mathbf{y}_i\rangle = 0$. This opens up much more liberty to the adversary in terms of what they can challenge. That is, as soon as the dimension of the vectors $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)}, \mathbf{y}_i)$ is at least 2, the adversary can choose $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})$ such that $\mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)}$ is orthogonal to $\mathbf{y}_i$, where both $\mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)} \neq \mathbf{0}, \mathbf{y}_i \neq 0$. In retrospective, the scalar version of [NPP23a] (Section 5.4) implies already that either $(x_i^{(0)} - x_i^{(1)}) = 0$ or $y_i = 0$, which is a special case of the vector version. Last but not least, regarding honest $i \in \mathcal{H} \coloneqq [n] \setminus \mathcal{C}$, it must hold that for all key queries with $(\mathbf{y}_i)_{i\in\mathcal{H}}$ as the parameters corresponding to honest slots, $\sum_{i\in\mathcal{H}} \langle \mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)}, \mathbf{y}_i\rangle = 0$. Particularly, the condition $\langle \mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)}, \mathbf{y}_i\rangle = 0$ for any $i \in \mathcal{C}$ and any $(i, \mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)}, \mathsf{tag})$ to **LoR** already implies that encryption of our MCFE must be necessarily probabilistic, because the adversary is allowed to makes challenge queries $\mathbf{x}_i^{(0)} - \mathbf{x}_i^{(1)} \neq \mathbf{0}$. This is highlighted in paragraph *Strong Admissibility and Public-Key Setting* of our introduction (in Section 2.3.3).

Solution to the First Obstacle: Probabilistic Vectorization of the Scheme of [NPP22a]. Our starting point is the scalar construction of [NPP22a] (Section 4.4.2), in the bilinear setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ are all written additively. The crux of our vectorization is to use the *dual pairing vector spaces* (DPVSes) to encode the vectors $\mathbf{x}_i$ and $\mathbf{y}_i$. In particular, each client encrypt their vector $\mathbf{x}_i$ by $\mathbf{c}$-vectors in $\mathbb{G}_1$, and the functional key for $\mathbf{y}_i$ is encoded in $\mathbf{k}^*$-vectors in $\mathbb{G}_2$. The importance is randomness must be added to the $\mathbf{c}$-vectors individually by each i, which cannot be founded on RO or *pseudorandom functions* as in previous works [NPP22a, ATY23a]. To implement such randomness and ensure that *correctness* is preserved, we make use of the concrete fact of DPVS that it provides linear combinations of vectors in $\mathbb{G}_1$ and

$\mathbb{G}_2$. This can be verified when viewing a DPVS as a $\mathbb{Z}_q$-algebra, sastisfying $\mathbb{Z}_q$-linearity and being equipped with an product operation that is provided by the bilinear map $\mathbf{e}$. We refer to Section 6.4 for more details and Algorithm 6.2 to see how the decryption is done. In contrast to the construction of [NPP22a, NPP23a], the decryption becomes much more complicated. The probabilistic vectorization is also used to handle the *repetitions* of challenge ciphertexts, as we will see in the below paragraph.

Second Technical Obstacle: Repetitions and Access Control. We have mentioned in the introduction that tolerating repetitions of challenge messages $\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)}$ is a crucial requirement for MCFE, in order for MCFE to imply MIFE in terms of provably secure cryptographic primitives. In our setting with *both private and public inputs*, the challenge ciphertexts given private $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})$ are encrypted with public parts comprising of a tag and the set of attributes S_i. This means that repetitions are now must be *vis-à-vis* the public parts, in particular S_i. The latter complicates significantly the situation, which is already observed in a very recent work by Agrawal *et al.* [ATY23a]. Indeed on one hand, for a specific slot $i \in [n]$ and tag, *full* repetitions of $(\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)})$ and $\mathsf{S}_i^{(j_i)}$ mean that the MCFE should be resilient against attacks that try combining different attribute set $\mathsf{S}_i^{(j_i)} \neq \mathsf{S}_i^{(\tilde{j}_i)}$ at slot i, where $\mathbb{A}(\mathsf{S}_i^{(j_i)}) \neq \mathbb{A}(\mathsf{S}_i^{(\tilde{j}_i)})$. On the other hand, in terms of the inner product calculation, allowing repetitions on the private inputs $\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)}$ for a fixed (i, tag) needs being taken into account by the admissibility: for all keys who decrypt (the key-policy $\mathbb{A}$ is satisfied), for all repetitions j_i

$$\sum_{i \in \mathcal{H}} \langle \mathbf{x}_i^{(0,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i \rangle = 0 \tag{6.1}$$

This implies for each $i \in \mathcal{H}$, over all repetitions j_i, the term $\langle \mathbf{x}_i^{(0,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i \rangle$ is constant. At the same time, for all $i \in \mathcal{C}$ that are corrupted, under repetitions j_i, it must be

$$\langle \mathbf{x}_i^{(0,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i \rangle = 0 \ . \tag{6.2}$$

This makes sense even in the case of *static corruption*, since we do *not* prohibit such queries even after the set $\mathcal{C}$ is fixed. Finally, condition (6.2) does *not* need to cover private inputs of corrupted $i \in \mathcal{C}$ that are not queried to the oracle **LoR** because there exists no challenge bit b in those self-crafted ciphertexts using $(i \in \mathcal{C}, \mathsf{ek}_i, \mathsf{tag})$ on some $\mathbf{z}_i$, and derypting jointly with others challenge ciphertexts under some key $\mathsf{dk}_{\mathbb{A},(\mathbf{y}_i)_{i\in[n]}}$ always gives the same i-th component $\langle \mathbf{z}_i, \mathbf{y}_i \rangle$ regardless of b. Last but not least, fir keys that do not decrypt (the key-policy $\mathbb{A}$ is satisfied), we do not have the guarantees from the admissibility but the adversary still tries to mix-and-match different repetitive attributes $\mathsf{S}_i^{(j_i)}, \mathsf{S}_i^{(\tilde{j}_i)}$, for instance.

Solution to the Second Obstacle: Masking with (Private-only) Repetitions. In this work we restrain our focus to the case where the repetitions are only allowed for the private inputs $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})$. That is, the adversary is allowed to query multiple $(\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)})$, indexed by j_i, for a fixed $(i, \mathsf{tag}, \mathsf{S}_i)$. Dealing with *private-input* repetitions is handled by our generalization of the masking lemma from [NPP22a]. The formal statement of the lemma can be found in Lemma 3.5. At a high level, the setting of Lemma 3.5 contains a set of $\mathbf{c}$-vectors in which attributes j are encoded, and a set of $\mathbf{k}^*$-vectors that encode a policy $\mathbb{A}$ by secret shares $(a_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ w.r.t the policy $\mathbb{A}$. The lemma proves that for any given repetitive $x^{(\mathsf{rep})}$ and $y \in \mathbb{Z}_q$, where $\mathsf{rep} \in [J]$, we can randomize the $\mathbf{c}$-vectors by random $z_j \xleftarrow{\$} \mathbb{Z}_q^*$, at the same time encoding $(a'_j/z_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ in the $\mathbf{k}$-vectors. Particularly $(a'_j/z_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ is a *decorrelated* set of shares $(a'_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ w.r.t the policy $\mathbb{A}$ to share $a'_0 \xleftarrow{\$} \mathbb{Z}_q$. In the proof of the MCFE, we alllow repetitions of the challenge ciphertexts while fixing $(i, \mathsf{tag}, \mathsf{S}_i)$. After applying Lemma 3.5[1], as soon as $\mathbb{A}(\mathsf{S}_i) = 0$, there is an

[1]The randomness that is needed for the masking also comes from our above probabilistic vectorization, wherever we need individual randomness.

attribute j whose $z_j \xleftarrow{\$} \mathbb{Z}_q^*$ never appears in the $\mathbf{c}$-vectors returned to the adversary, thanks to the fact that $(i, \mathsf{tag}, \mathsf{S}_i)$ is fixed once for all repetitions of private inputs at i. That implies the decorrelated $(a'_j/z_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ cannot be related together, in an *information theoretical* sense, to recover $(a'_j)_{j \in \mathsf{List\text{-}Att}(\mathbb{A})}$ and reconstruct the shared value. We are then allowed to switch a'_0 into a uniformly random value for further steps in the MCFE proof. Finally, as demonstrated in Theorem 6.6, even in this setting of private-only repetitions, our MCFE with public inputs still cover MIFE, and the concrete scheme for inner products with access control in Section 6.4 gives MIFE for inner products.

Third Technical Obstacle: Adaptive Security. Another technical hurdle with which we successfully deal in our MCFE is the adaptive security of the challenge queries $\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)}$ indexed by repetitions j_i along with public inputs $(i, \mathsf{tag}, \mathsf{S}_i)$. Existing comparable schemes either achieves selective security [ATY23a], or considers the simpler scalar case [NPP22a].

Solution to the Third Obstacle: Adaptive Security via Perfect Indistinguishability and Complexity Leveraging. Aiming at adaptive security w.r.t $(\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)})$ with public inputs $(i, \mathsf{tag}, \mathsf{S}_i)$, we employ a *complexity leveraging* technique that is based on *formal* basis changes in the dual pairing vector spaces. More specifically, in order to prove two hybrids $\mathsf{G}_i, \mathsf{G}_{i+K}$ for some fixed K, are indistinguishable in the adaptive security proof, we define an event E that happens with fixed probability and whose probability space depends on the data that can be *adaptively* chosen by the adversary. Then, condition on E we move to the *selective* version $\mathsf{G}_i^*, \mathsf{G}_{i+1}^*, \ldots, \mathsf{G}_{i+K}^*$. If we can prove the sequence of *perfect indistinguishability* involving

$$\{\mathsf{G}_i^* \mid E\} \equiv \{\mathsf{G}_{i+1}^* \mid E\} \equiv \cdots \equiv \{\mathsf{G}_{i+K}^* \mid E\}$$

where E happens with fixed probability and is *independent* of the view of the adversary during the reductions $\{\mathsf{G}_{i+t}^* \; E\} \equiv \{\mathsf{G}_{i+t+1}^* \mid E\}$ in the sequence, for all $t \in [K-1]$, then a probabilistic argument concludes that $\{\mathsf{G}_i\} \equiv \{\mathsf{G}_{i+1}\} \equiv \cdots \equiv \{\mathsf{G}_{i+K}\}$. The formal basis changes are used to achieve perfect indistinguishability between these selective versions $\{\mathsf{G}_{i+t}^* \mid E\}$ of the game. In the MCFE adaptive proof, the adaptive data include $(\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)})$ indexed by multiple repetitions j_i. We extensively use admissibility conditions (6.1) as well as (6.2) to define the basis changes. Details can be found in the proof of Theorem 6.11, the final probabilistic calculation for complexity leveraging can be examined in (6.9), for instance. As a reminder, an integral overview that puts together all above solutions, and more, is given in the paragraph **Proof Strategy** before the proof of Theorem 6.11.

6.3 Multi-Client Functional Encryption with Public Inputs

In this section we refine the definition of *multi-client functional encryption* in which at the time of encryption, each client can specify their own *public* data, while the function class contains functions that evaluate *both* the combined private and public data of clients. In Section 6.3.2 we prove that this general notion covers the original MCFE notion *with* and *without* fine-grained access control, and even more, *e.g.* the notion of *public-attributes* ABE. Interestingly, the syntax of previous formal definitions of FE, either in single-client [BSW11] or multi-client [GGG+14, CDG+18a], allows no public data and let *public-attributes* ABE escape their scope, or has to include an artificial function in the class. More specifically, we discuss in Theorem 6.6 how our formal definition of MCFE with public inputs can be related to other existing primitives.

6.3.1 Definitions

Definition 6.1 (Functions with public inputs). *Let $\lambda, n \in \mathbb{N}$ and let $\mathcal{D}_{\lambda,i}$ and $\mathcal{R}_\lambda$ be domains and ranges indexed by λ in some ensembles $\{\mathcal{D}_{\lambda,i}\}_\lambda$ where $i \in [n]$, $\{\mathcal{R}_\lambda\}_\lambda$, respectively. A* function class *$\mathcal{F} = \{F_{\lambda,n}\}_{\lambda,n}$ with public inputs $(\mathcal{Z}_{\lambda,i})_{i\in[n]}$, where $\mathcal{Z}_{\lambda,i} := \{0,1\}^{poly(\lambda)}$, is defined to contain $F_{\lambda,n} : \prod_{i=1}^{n} (\mathcal{D}_{\lambda,i} \times \mathcal{Z}_{\lambda,i}) \to \mathcal{R}_\lambda$.*

In the following the index n is a function in λ and we omit it for clarity.

Definition 6.2 (Multi-client functional encryption with public inputs). *A* multi-client functional encryption (MCFE) scheme with public inputs*, for the class $\mathcal{F}$ with public inputs $(\mathcal{Z}_{\lambda,i})_{i\in[n]}$ where $\mathcal{Z}_{\lambda,i} := \mathsf{Tag} \times \widetilde{\mathcal{Z}}_{\lambda,i}$ for some set $\mathsf{Tag} = \{0,1\}^{poly(\lambda)}$, consists of four algorithms* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$*:*

$\mathsf{Setup}(1^\lambda, 1^n)$**:** *Given as inputs 1^λ for a security parameter λ, and a number of clients n, output a master secret key msk and n encryption keys $(\mathsf{ek}_i)_{i\in[n]}$.*

$\mathsf{Extract}(\mathsf{msk}, F_\lambda)$**:** *Given a function description $F_\lambda : \prod_{i=1}^{n} (\mathcal{D}_{\lambda,i} \times \mathcal{Z}_{\lambda,i}) \to \mathcal{R}_\lambda$ in $\mathcal{F}$, and the master secret key msk, output a decryption key dk_{F_λ}.*

$\mathsf{Enc}(\mathsf{ek}_i, x_i, z_i)$**:** *Given as inputs* public *data $z_i = (\mathsf{tag}, \tilde{z}_i) \in \mathcal{Z}_{\lambda,i}$ that contains some tag, an encryption key ek_i, a message $x_i \in \mathcal{D}_{\lambda,i}$, output a ciphertext $(\mathsf{ct}_{\mathsf{tag},i}, z_i)$. For a specific client i, the sets $\mathcal{D}_{\lambda,i}$ and $\mathcal{Z}_{\lambda,i}$ are indexed by λ in some ensembles $\{\mathcal{D}_{\lambda,i}\}_\lambda, \{\mathcal{Z}_{\lambda,i}\}_\lambda$.*

$\mathsf{Dec}(\mathsf{dk}_{F_\lambda}, \mathbf{c})$**:** *Given the decryption key dk_{F_λ} and a vector of ciphertexts $\mathbf{c} := (\mathsf{ct}_{\mathsf{tag},i}, z_i)_i$ of length n, output an element in $\mathcal{R}_\lambda$.*

Our syntax can be seen as a particular case of the general primitive *Multi-Party Functional Encryption* (MPFE) [AGT21c] in which we consider the particular case of multi-client while the key generation stays centralized. The main difference is in terms of security where ours is less restrictive (see Definition 6.3), which is sufficient to for establishing connection to other primitives as we will see in Section 6.3.2. Regarding the concrete class calculating *inner products with access control*, we will revisit the connection from MIFE to MCFE in Section 6.4.

Correctness. For sufficiently large $\lambda \in \mathbb{N}$, for all $(\mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda)$, all functions $F_{\lambda,n} : \prod_i (\mathcal{D}_{\lambda,i} \times \mathcal{Z}_{\lambda,i}) \to \mathcal{R}_\lambda$ and $\mathsf{dk}_{F_{\lambda,n}} \leftarrow \mathsf{Extract}(\mathsf{msk}, F_{\lambda,n})$, for all $\mathsf{tag} \in \mathsf{Tag}$ and $(z_i)_{i=1}^n \in \mathcal{Z}_{\lambda,1} \times \cdots \times \mathcal{Z}_{\lambda,n}$, for all $(x_i)_{i\in[n]} \in \mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n}$, if $F_\lambda((x_i, z_i)_i) \neq \perp$ and $z_i = (\mathsf{tag}, \tilde{z}_i) \in \mathcal{Z}_i$ for all i, the following holds with overwhelming probability, over the random coins of the algorithms:

$$\mathsf{Dec}\left(\mathsf{dk}_{F_\lambda}, (\mathsf{Enc}(\mathsf{ek}_i, x_i, z_i))_{i\in[n]}\right) = F_{\lambda,n}((x_i, z_i)_i) \ .$$

Security. First of all we define *admissible* adversaries $\mathcal{A}$ against an MCFE $\mathcal{E}$. We use the recent formulation of admissibility in [NPP23a] (Definition 5.4).

Definition 6.3 (Admissible adversaries with public inputs). *Let $\mathcal{A}$ be a* ppt *adversary and let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ be* an MCFE scheme with public inputs *for the function class $\mathcal{F}$ with public inputs $\mathcal{Z}_{\lambda,i} := \mathsf{Tag} \times \widetilde{\mathcal{Z}}_{\lambda,i}$. In the security game given in Figure 6.1 for $\mathcal{A}$ considering $\mathcal{E}$, let the sets $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ be the sets of corrupted clients, functional key queries, and honest clients, in that order. We say that $\mathcal{A}$* is NOT admissible w.r.t *$(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ if the following condition holds:*

> *There exist $\mathsf{tag} \in \mathsf{Tag}$, a function $F \in \mathcal{F}$ is queried to* **Extract**, *challenges $(x_i^{(0)}, x_i^{(1)}, (\mathsf{tag}, \tilde{z}_i^{(chal)}))_{i\in[n]}$ is queried to* **LoR**, *with public inputs $\tilde{z}_i^{(chal)} \in \widetilde{\mathcal{Z}}_{\lambda,i}$, and there exist vectors $(\mathbf{t}^{(0)}, \mathbf{t}^{(1)}, \mathbf{v}^{(chal)})$ so that $\forall\, i \in \mathcal{H} : \mathbf{t}^{(b)}[i] = x_i^{(b)}$ and $\mathbf{v}^{(chal)}[i] = \tilde{z}_i^{(chal)}$ satisfying*
>
> $$F((\mathbf{t}^{(0)}[i], (\mathsf{tag}, \mathbf{v}[i]))_{i\in[n]}) \neq F((\mathbf{t}^{(1)}[i], (\mathsf{tag}, \mathbf{v}[i]))_{i\in[n]}) \ . \tag{6.3}$$

Otherwise, we say that $\mathcal{A}$ is admissible w.r.t *$(\mathcal{C}, \mathcal{Q}, \mathcal{H})$.*

Discussion on admissibility. We develop below some discussion on the admissibility notion in Definition 6.3:

- *(Repetitions)* In comparison to the original security of MCFE in [CDG+18a], an adversary is still *admissible* if they query multiple times to the challenge oracle for a fixed (i, tag), whereas an admissible adversary *as per* [CDG+18a] is allowed to query at most once for each (i, tag). This aspect of *repetitions* in the admissibility was first studied in [CDG+18b] and later generalized in [CDSG+20]. It is important that when repetitions are allowed for ciphertexts, the security model of MCFE automatically encompasses that of MIFE by replacing tags with a constant value, as confirmed in recent works [ATY23b]. Lastly, in our notion of MCFE with public inputs, we can also consider restricted repetitions only on the private parts $(x_i^{(0)}, x_i^{(1)})$ (see the weaker notion $\mathsf{rep\text{-}priv}$ in the following) and not on the public parts $z_i^{(chall)}$ to the challenge oracle. This form of restricted repetitions gives a weaker notion of security, but it still covers the security of classical MIFE *without* public inputs, as studied in [GGG+14, AJ15, AGRW17, DOT18, ACF+18, Tom19, AGT21a, AGT22].

- *(Weaker constraints)* Regarding the corrupted $i \in \mathcal{C}$ in general, the admissibility check is done in **Finalise** at the end of the security experiment, and Definition 6.3 *per se* allows the adversary to query the challenge oracle **LoR**, whether the corruption is static or not, on

  $$i, x_i^{(0)}, x_i^{(1)}, (\mathsf{tag}^*, \tilde{z}_i^{(chal)})$$

  where $x_i^{(0)} \neq x_i^{(1)}$. The adversary stays admissible as long as the condition (6.3) is not satisfied, *i.e.* the foregoing $x_i^{(0)} \neq x_i^{(1)}$ of corrupted $i \in \mathcal{C}$ does not make F differ with respect to the challenge bit $b \xleftarrow{\$} \{0, 1\}$. The original security of MCFE in [CDG+18a] does not allow attacks where there exists $i \in \mathcal{C}$ such that $x_i^{(0)} \neq x_i^{(1)}$. By allowing a such query, we apparently allow more attacks than the original security model of MCFE in [CDG+18a]. The work [NPP23a] examines the legitimacy of this condition in the plain (Decentralized) MCFE (DMCFE) setting and proposes a stronger security model that does allow $x_i^{(0)} \neq x_i^{(1)}$ of corrupted $i \in \mathcal{C}$ (thus considers more attacks admissible).

- *(Corrupted ciphertexts)* In terms of usage of the corrupted ek_i, for the admissible conditions 6.3 we do *not* put any quantifier on the ciphertexts that can be crafted by the adversary using a corrutped ek_i for $i \in \mathcal{C}$. Because when decrypting jointly a such ciphertext

$\bar{\mathsf{ct}}_i \leftarrow \mathsf{Enc}(\mathsf{ek}_i, \bar{x}_i, \bar{z}_i)$ with other challenge ciphertext components (up to repetitions) vis-à-vis a function F, the evaluation will provide

$$F((\mathbf{t}^{(b)}[j], (\mathsf{tag}, \mathbf{v}[j]))_{j \neq i}, (\bar{x}_i, \bar{z}_i), (\mathbf{t}^{(b)}[j'], (\mathsf{tag}, \mathbf{v}[j']))_{j' \neq i})$$

that always has the same i-th argument and cannot change the output of F. The same reasoning applies when the adversary crafts themselves multiple corrupted ciphertexts.

- *(Checking admissibility)* The admissiblity in Definition 6.3 for general function class may not be efficiently decidable. As we will see later in Section 6.4, within the scope of this paper, the class of functions is restricted to computing inner products with access control by LSSS, and the admissibility can be decided efficiently using concrete conditions 1 and 2 prior to the proof of Theorem 6.11.

In Theorem 6.6 we discuss how an MCFE that is provably secure under the admissibility in Definition 6.3 will imply a provably secure MIFE, and more. For the concrete class of computing *inner products with access control* which is the main subject of Section 6.4, we refer to Remark 6.14.

Definition 6.4 (IND-security with repetitions for MCFE with public inputs). *An MCFE scheme with public inputs* $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ *for the function class* $\mathcal{F}$ *with public inputs is* IND-secure *if for all* ppt *adversaries* $\mathcal{A}$*, and for all sufficiently large* $\lambda \in \mathbb{N}$*, the following probability is negligible*

$$\mathsf{Adv}^{\mathsf{mc\text{-}w\text{-}rep}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) := \left| \Pr[\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2} \right| .$$

The security game $\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ *is depicted in Figure 6.1. The probability is taken over the random coins of* $\mathcal{A}$ *and the algorithms.*

In a more relaxed notion, the scheme $\mathcal{E}$ is *selectively IND-secure* with the security game $\mathsf{Expr}^{\mathsf{mc\text{-}sel\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$, where the challenges are chosen before the setup.

Weaker notions. We can relax the admissibility notion from Definition 6.3, with more exclusions, to obtain weaker security notions considered in literature. They are simpler to achieve, and some generic conversions allow to lift from a weaker to a stronger scheme.

- In previous works, one can consider a weaker notion of security for MCFE in which either all or none of honest components in the challenge are queried. In this case, we say that the MCFE scheme is secure against *complete* queries only and add the following exclusion to the admissibility:

  > There exist a tag tag and $i, j \in \mathcal{H}$ such that $i \neq j$, there exists a query $(i, x_i^{(0)}, x_i^{(1)}, (\mathsf{tag}, *))$ to **LoR** but there exist no query $(j, x_j^{(0)}, x_j^{(1)}, (\mathsf{tag}, *))$ to **LoR**.

  We denote the corresponding experiment with this weaker notion in admissibility, *i.e.* which is called *pos*-security in the literature, with the flag pos in the name of the experiment.

- One can also keep the original security notion from [CDG+18a] by imposing the *same* challenge components for corrupted $i \in \mathcal{C}$. We then add the exclusion to the admissibility:

  > There exists $i \in \mathcal{C}$ such that $x_i^{(0)} \neq x_i^{(1)}$.

  We denote the corresponding experiment with this *weaker* notion in admissibility, with the flag wk.

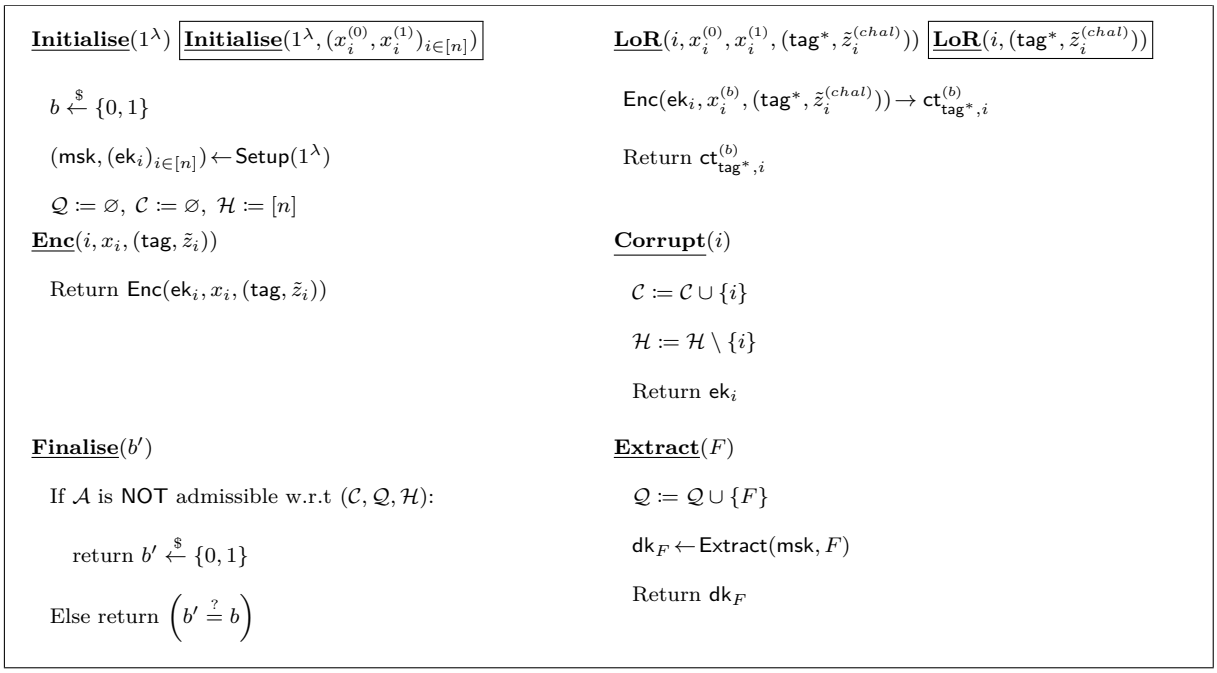


Figure 6.1: The security game $\mathsf{Expr}^{\mathsf{mc\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$, $\boxed{\mathsf{Expr}^{\mathsf{mc\text{-}sel\text{-}ind\text{-}cpa}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)}$ for Definition 6.4

- We also define a notion of security where only one challenge tag tag^* is allowed, with the following exclusion to the admissibility:

  > There exist two tags $\mathsf{tag} \neq \mathsf{tag}'$ and queries $(*, *, *, (\mathsf{tag}, *))$, $(*, *, *, (\mathsf{tag}', *))$ to **LoR**.

  That is, the scheme $\mathcal{E}$ is *one-time IND-secure*, with the flag 1chal in the name of the experiment.

- Finally, if we allow only repetitions on the *private* parts $(x_i^{(0)}, x_i^{(1)})$ and not on the public parts $z_i^{(chal)}$ to **LoR** (or x_i and not on z_i to **Enc**), we denote the corresponding experiment with this weaker notion with the flag rep-priv, with the additional exclusion:

  > There exist a tag tag, an index i and two public values $z \neq z'$, with queries $(i, *, *, (\mathsf{tag}, z))$ to **LoR** or $(i, *, (\mathsf{tag}, z))$ to **Enc**, and $(i, *, *, (\mathsf{tag}, z'))$ to **LoR** or $(i, *, (\mathsf{tag}, z'))$ to **Enc**.

All these flags pos, wk, 1chal, rep-priv can be combined and added to the experiments presented in Figure 6.1.

Lemma 6.5 allows us to concentrate on the notion of one-time IND-security for our construction. The proof is a standard hybrid argument, thanks to the **Enc**-oracle access (in the case of secret-key encryption), in addition to **LoR**.

Lemma 6.5. *Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ for the function class $\mathcal{F}$ be an MCFE scheme with public inputs. If $\mathcal{E}$ is one-time IND-secure, then $\mathcal{E}$ is IND-secure.*

6.3.2 Implications between Notions: MCFE, MIFE, and more

Since its introduction in [GGG+14], a long line of works [AJ15, AGRW17, DOT18, ACF+18, Tom19, AGT21a, AGT22] considers MIFE having only *one* encryptor who can use a master secret key to encrypt independent components of a message. Our definition of MCFE from Definition 6.2

can capture this widely studied (one-encryptor) notion of MIFE, *with* and *without* access control, and in the latter case with *public* attributes. Generally, Theorem 6.6 demonstrates that given a secure MCFE *as per* Definition 6.4 for a strong enough function class with public inputs, we can obtain secure instantiations of standard existing MIFE/MCFE notions in the secret-key setting as well as (single-client) FE/KP-ABE notions in the public-key setting. Relevant notions are recalled in Section 3.5.

Theorem 6.6. *Let $\mathcal{F}$ be a function class with public inputs $(\mathcal{Z}_{\lambda,i})_{i\in[n]}$ where $\mathcal{Z}_{\lambda,i} := \mathsf{Tag} \times \widetilde{\mathcal{Z}}_{\lambda,i}$ for some tag space $\mathsf{Tag} = \{0,1\}^{poly(\lambda)}$. The elements of $\mathcal{F}$ are $F_{\lambda,n} : \prod_{i=1}^{n} (\mathcal{D}_{\lambda,i} \times \mathcal{Z}_{\lambda,i}) \to \mathcal{R}_\lambda$. Suppose that $\mathcal{F}$ contains the identity function $F^{\mathsf{id}}_{\lambda,n}$ where for all $(x_i, z_i)_i$, $F^{\mathsf{id}}_{\lambda,n}((x_i,z_i)_i) = (x_i,z_i)_i$. We suppose further that $\mathcal{F}_{\lambda,n}$ can encode a policy class Pol whose attributes are contained in $\mathsf{Att} \subseteq \widetilde{\mathcal{Z}}_{\lambda,i}$ for all $i \in [n]$. We have the following commutative diagram:*

$$\begin{array}{ccc} \mathit{MCFE}^{\mathsf{xxx}\text{-}\mathsf{rep}\text{-}\mathsf{priv}}[\mathcal{F}, (\mathcal{Z}_{\lambda,i})_{i\in[n]}] & \xrightarrow{\mathsf{rep}\text{-}\mathsf{priv}} & \mathit{MIFE}^{\mathsf{xxx}\text{-}}[\mathcal{F}] \\ \Big\downarrow{\scriptstyle \mathsf{adm}} \ \textit{(Def. 6.3)} & & \\ \mathit{FE}^{\mathsf{xxx}\text{-}}[\mathcal{F}, (\mathcal{Z}_{\lambda,i})_{i\in[n]}] & \xrightarrow[\mathsf{Att}\subseteq\widetilde{\mathcal{Z}}_{\lambda,i}]{\textit{pub. input}} & \mathit{KP\text{-}ABE}^{\mathsf{xxx}\text{-}}_{\mathsf{pub}}[\mathsf{Pol}, \mathsf{Att}] \end{array}$$

where

- *Each arrow "$\to$" preserves the IND-CPA security level* $\mathsf{xxx} \in \{\mathsf{sel}, \mathsf{adp}, \mathsf{stat}\}$ *of challenge-selective, challenge-adaptive, static corruption security respectively. The label of the arrow indicates the necessary property for it to hold, detailed in the proof.*
- $\mathit{MIFE}^{\mathsf{xxx}\text{-}}[\mathcal{F}]$ *denotes an MIFE following Definition 3.11, that can be adapted to capture MIFE for calculations in* $\mathcal{F}$ without *access control as defined in [AJ15, AGRW17, DOT18, ACF+18, Tom19, AGT21a, AGT22].*
- $\mathit{FE}^{\mathsf{xxx}\text{-}}[\mathcal{F}, (\mathcal{Z}_{\lambda,i})_{i\in[n]}]$ *following Definition 3.9, that can be adapted to capture FE* with *access control as in [ACGU20, NPP22a, ATY23a], or* without *access control [BSW11].*
- $\mathit{KP\text{-}ABE}^{\mathsf{xxx}\text{-}}_{\mathsf{pub}}[\mathsf{Pol}, \mathsf{Att}]$ *denotes a KP-ABE for the policy class* Pol *with* public *attributes. The notion follows Definition 3.7.*

Proof. We perform the reductions below. Let $\mathsf{MCFE}^{\mathsf{xxx}\text{-}}[\mathcal{F}, (\mathcal{Z}_{\lambda,i})_{i\in[n]}]$ be a secure MCFE following Definition 6.4. We denote by $(\mathsf{Setup}^{\mathsf{mc}}, \mathsf{Extract}^{\mathsf{mc}}, \mathsf{Enc}^{\mathsf{mc}}, \mathsf{Dec}^{\mathsf{mc}})$ the algorithms of the MCFE. We use ϵ to denote the empty string.

From MCFE to MIFE. Following Definition 3.10, we consider the notion of MIFE having only *one* encryptor who can use a master secret key to encrypt independent components of a message. The function class is $\mathcal{F}$ containing $F_{\lambda,n} : \prod_{i=1}^{n} (\mathcal{D}_{\lambda,i} \times \mathcal{Z}_{\lambda,i}) \to \mathcal{R}_\lambda$. There is *no* public inputs as we are concentrating on the classic MIFE *as per* [GGG+14] that is recalled in Definition 3.10. The obtained MIFE is defined by the algorithms:

$\mathsf{Setup}^{\mathsf{mi}}(1^\lambda, 1^n)$**:** Run $\mathsf{Setup}^{\mathsf{mc}}(1^\lambda, 1^n) \to (\mathsf{msk}^{\mathsf{mc}}, (\mathsf{ek}_i^{\mathsf{mc}})_i)$. Sample a tag $\mathsf{tag} \xleftarrow{\$} \mathsf{Tag}$ and output $\mathsf{msk} := \mathsf{msk}^{\mathsf{mc}}, (\mathsf{ek}_i := (\mathsf{ek}_i^{\mathsf{mc}}, \mathsf{tag}))_i$.

$\mathsf{Extract}^{\mathsf{mi}}(\mathsf{msk}, F_\lambda)$**:** Run $\mathsf{Extract}^{\mathsf{mc}}(\mathsf{msk}^{\mathsf{mc}}, F_\lambda) \to \mathsf{dk}_{F_\lambda}$ and output dk_{F_λ}.

$\mathsf{Enc}^{\mathsf{mi}}(\mathsf{ek}_i, x_i)$**:** Parse $\mathsf{ek}_i := (\mathsf{ek}_i^{\mathsf{mc}}, \mathsf{tag})$. Run $\mathsf{Enc}^{\mathsf{mc}}(\mathsf{ek}_i^{\mathsf{mc}}, x_i, (\mathsf{tag}, \epsilon)) \to \mathsf{ct}_i$ as there is no public inputs in classical MIFE, then output ct_i.

$\mathsf{Dec}^{\mathsf{mi}}(\mathsf{dk}_{F_\lambda}, (\mathsf{ct}_i)_i)$**:** Run and output $\mathsf{Dec}^{\mathsf{mc}}(\mathsf{dk}_{F_\lambda}, (\mathsf{ct}_i)_i)$.

Correctness follows from the correctness of the MCFE. In terms of security, let $\mathcal{A}$ be an adversary against the MIFE *as per* Definition 3.11. We construct an adversary $\mathcal{B}$ breaking $\mathsf{MCFE}^{\mathsf{xxx\text{–}rep\text{-}priv}}[\mathcal{F}, (\mathcal{Z}_{\lambda,i})_{i\in[n]}]$ using $\mathcal{A}$.

The adversary $\mathcal{B}$ simulates the MIFE game by (i) first querying its MCFE challenger on $(1^\lambda, 1^n)$ to obtain the public parameters (if any) then forwards to $\mathcal{A}$; (ii) simulating the MIFE's encryption/challenge queries by fixing a tag tag for all encryption (respectively challenge) ciphertexts and forwarding the encryption (that is, $(i, x_i, (\mathsf{tag}, \epsilon))$) (respectively challenge (that is, $(i, x_i^{(0)}, x_i^{(1)}, (\mathsf{tag}, \epsilon))$)) queries given (i, x_i) or $(i, x_i^{(0)}, x_i^{(1)})$ by $\mathcal{A}$ against the MIFE; (iii) the key-extraction queries are forwarded to the MCFE challenger in a straightforward manner. In the end $\mathcal{B}$ outputs the same as $\mathcal{A}$. If $\mathcal{A}$ wins the MIFE game, then $\mathcal{B}$ wins the MCFE game. We remark that when $\mathcal{A}$ makes *repetitions* over the encryption queries (*i.e.* same i but different messages), the forwarded queries to the MCFE challenger are *repetitions* over the *private* inputs as well, while the public inputs stay (tag, ϵ) for both **Enc** and **LoR**). In particular if $\mathcal{A}$ is admissible following Definition 3.11, all queries by $\mathcal{B}$ to its challenger are admissible *as per* Definition 6.3, in the *private-only repetitions*, because the conditions of MIFE security imposes more restricting conditions, due to the fact that there are more possibilities to combine ciphertexts[2].

From MCFE to (single-client, public-key) FE. The function class is $\mathcal{F}$ containing $F_\lambda : \mathcal{D}_\lambda \times \mathcal{Z}_\lambda \to \mathcal{R}_\lambda$. Following Definition 3.8, the obtained FE is defined by algorithms:

$\mathsf{Setup}^{\mathsf{pk}}(1^\lambda)$**:** Run $\mathsf{Setup}^{\mathsf{mc}}(1^\lambda, 1^1) \to (\mathsf{msk}^{\mathsf{mc}}, \mathsf{ek}^{\mathsf{mc}})$. Output $\mathsf{msk} := \mathsf{msk}^{\mathsf{mc}}, \mathsf{pk} := (\mathsf{ek}^{\mathsf{mc}})$.

$\mathsf{Extract}^{\mathsf{pk}}(\mathsf{msk}, F_\lambda)$**:** Run $\mathsf{Extract}^{\mathsf{mc}}(\mathsf{msk}^{\mathsf{mc}}, F_\lambda) \to \mathsf{dk}_{F_\lambda}$ and output dk_{F_λ}.

$\mathsf{Enc}^{\mathsf{pk}}(\mathsf{pk}, x, z)$**:** Parse $\mathsf{pk} := (\mathsf{ek}^{\mathsf{mc}})$ and $z := (\epsilon, \tilde{z})$ as there is no tag in single client and public key FE. Sample $\mathsf{tag} \xleftarrow{\$} \mathsf{Tag}$ and run $\mathsf{Enc}^{\mathsf{mc}}(\mathsf{ek}^{\mathsf{mc}}, x, (\mathsf{tag}, \tilde{z})) \to \mathsf{ct}$. Finally output ct.

$\mathsf{Dec}^{\mathsf{pk}}(\mathsf{dk}_{F_\lambda}, \mathsf{ct})$**:** Run and output $\mathsf{Dec}^{\mathsf{mc}}(\mathsf{dk}_{F_\lambda}, \mathsf{ct})$.

Correctness follows from the correctness of the MCFE. If the function class captures access control, then the FE is for the same class having access control as well. In terms of security, let $\mathcal{A}$ be an adversary against the FE *as per* Definition 3.9. We construct an adversary $\mathcal{B}$ breaking $\mathsf{MCFE}^{\mathsf{xxx\text{–}rep\text{-}priv}}[\mathcal{F}, (\mathcal{Z}_{\lambda,i})_{i\in[n]}]$, with *static* corruptions, using $\mathcal{A}$. The adversary $\mathcal{B}$ simulates the FE game by (i) first querying its MCFE challenger on $(1^\lambda, 1)$ to obtain the public parameters pp (if any) then *queries* $\mathbf{Corrupt}(1)$, gets ek, and forwards $\mathsf{pk} := \mathsf{ek}$ together with pp to $\mathcal{A}$. We note that the corrupted client is known from the beginning; (ii) simulating the FE's challenge queries by forwarding the challenge queries (*i.e.* sample $\mathsf{tag} \xleftarrow{\$} \mathsf{Tag}$ and define the challenge to be $(1, x^{(0)}, x^{(1)}, (\mathsf{tag}, \tilde{z}^{(chal)}))$) to its MCFE challenger given $\left(\left(x^{(0)}, (\epsilon, \tilde{z}^{(chal)})\right), \left(x^{(1)}, (\epsilon, \tilde{z}^{(chal)})\right)\right)$ by $\mathcal{A}$; (iii) the key extraction queries are forwarded to the MCFE challenger in a straightforward manner. If the FE adversary $\mathcal{A}$ is admissible, *i.e.* $x^{(0)} \neq x^{(1)}$ but $F(x^{(0)}, (\epsilon, \tilde{z}^{(chal)})) = F(x^{(1)}, (\epsilon, \tilde{z}^{(chal)}))$ for all F queried to **Extract**, then the challenge query $(1, x^{(0)}, x^{(1)}, (\mathsf{tag}, \tilde{z}^{(chal)}))$ is on a pair of inputs $(x^{(0)}, (\mathsf{tag}, \tilde{z}^{(chal)})) \neq (x^{(1)}, (\mathsf{tag}, \tilde{z}^{(chal)}))$ conforming to the admissibility. This implies that $\mathcal{B}$ is also admissible following Definition 6.3. Moreover, the fact that every encryption query is defined on a *freshly sampled* tag implies that there is no repetitions for any pair $(1, \mathsf{tag})$ registered to the MCFE challenger. This allows us to allow encrypting different public inputs even though the MCFE is for *private inputs repetitions* only. Therefore, if $\mathcal{A}$ wins the FE game, then $\mathcal{B}$ wins the MCFE game.

[2] In the one-encryptor setting there is no corruption oracle in the MIFE game, *e.g.* see the original in [GGG+14].

Implication to KP-ABE. The implication to KP-ABE follows from the (single-client, public-key) FE case for $\mathcal{F}$ containing $F_\lambda : \mathcal{D}_\lambda \times \mathcal{Z}_\lambda \to \mathcal{R}_\lambda$. Moreover, the identity function is in $\mathcal{F}$ and allows the *all-or-nothing* decryption of KP-ABE, without any evaluation on the plaintext. In particular, thanks to the hypothesis that the function class $\mathcal{F}$ can encode a policy class Pol, and the attribute space Att is contained in $\mathcal{Z}_\lambda$. A reduction from FE to KP-ABE can be obtained with ease. Once again, even though the MCFE is set up for one slot, each time an encryption is created, a fresh tag is sampled therefore not leading to a fully repetitive on *both* private and public inputs. This thus allows encrypting on different attribute sets while there is no full repetitions for any pair $(1, \mathsf{tag})$ registered to the MCFE challenger. Finally, an adversary breaking the KP-ABE allows breaking the MCFE.

Remark 6.7. (From secret key to public key) We emphasize that the crucial point allowing us to go from the *secret key* setting of MCFE to the *public key* setting of FE is the *admissibility* in Definition 6.3 under corruption. More specifically, Definition 6.3 allows the reduction to forward the challenge queries of its (public key) FE to the MCFE challenger, for the only client as $n = 1$,

$$(x^{(0)}, (\mathsf{tag}, \tilde{z}^{(chal)})) \neq (x^{(1)}, (\mathsf{tag}, \tilde{z}^{(chal)}))$$

as long as $F(x^{(0)}, (\epsilon, \tilde{z}^{(chal)})) = F(x^{(1)}, (\epsilon, \tilde{z}^{(chal)}))$ for all F queried to **Extract**. The only *secret encryption key* ek is corrupted up front and known to the FE adversary as a *public key* pk. Comparing to existing admissibility notions in [CDG+18a], which excludes attacks where there exists $i \in \mathcal{C}$ such that $x_i^{(0)} \neq x_i^{(1)}$, the only queries that the reduction can forward are the *trivial* one from the FE adversary where $x^{(0)} = x^{(1)}$. Hence, existing admissibility notions in [CDG+18a] and subsequent works are not sufficient to capture the reduction from MCFE to FE with meaningful CPA-security. Furthermore KP-ABE is made possible (without attribute-hiding) thanks to the public inputs.

Remark 6.8. (Concrete instantiations) Another key observation of Theorem 6.6 is that starting from any provably MCFE, we obtain an MIFE for the same function class by fixing one public tag for all ciphertexts. The security of the resulted MIFE comes from the fact that the security of the underlying MCFE allows *repetitions* at each position i, under the fixed tag, thanks to the admissibility in Definition 6.3. In this chapter, our final construction for MCFE with access-control (see Corollary 6.13) satisfies this security with repetitions along with other favorable properties to be lifted to an MIFE with access-control. We consider the function class $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ as defined in Definition 6.10. Applying Theorem 6.6 to our MCFE in Section 6.4 gives concrete instantiations of the corresponding primitives.

6.4 MCFE with Access Control: Revisited

First of all, we specialize the general notion of MCFE with public inputs so as to define and give the model of security for *multi-client functional encryption with fine-grained access control* in Section 6.4.1. Our main goal is to improve the MCFE construction in [NPP22a, Section 5], which supports only encrypting scalars and does not tolerate *repetitions* of challenge ciphertexts. Section 6.4.2 gives an extension to encrypt subvectors, in a security model where the admissibility allows *repetitions* at positions under a challenge tag. Towards Corollary 6.13, we remove all *one-challenge* and *complete* challenge queries, and the resulted MCFE can be made MIFE by fixing a public tag. This clarifies the conversion from MCFE to MIFE in [NPP22a, Remark 16]. A subtlety is that the fixed public tag is processed by hashing, leading to a MIFE that inherits all security properties of the MCFE but without tags and without corruption. Hence, putting forward the fact that our MCFE does *not* allow repetitions on the attributes per client but only repetitions their private inputs, the obtained MIFE is secure only against *repetitions on private inputs*, *i.e.* potentially repetitive private $\mathbf{x}_i$ and no repetitions on the attributes S_i of each i. We discuss further our construction and revisit the MIFE regime for comparison with [ACGU20, NPP22a] in Remark 6.14.

6.4.1 Definitions

We specialize the notion of MCFE with public inputs in Definition 6.2 to define the notion of multi-client functional encryption with fine-grained access control, *key-policy* and with *public attributes*.

Specialized function class with access control. Let $\lambda \in \mathbb{N}$ be a security parameter and we denote by n the number of clients in the system, which is fixed at set up time. We describe the function class $\mathcal{F} \times \mathsf{AC\text{-}K}$ for the multi-client functional encryption with fine-grained access control below:

- The public attributes of each client i come from $\mathcal{Z}_{\lambda,i} \coloneqq \mathsf{Tag} \times \mathsf{AC\text{-}Ct}_i$ for some set $\mathsf{AC\text{-}Ct}_i$ and a tag space $\mathsf{Tag} = \{0,1\}^{poly(\lambda)}$.
- The access control is defined via a relation $\mathsf{Rel} : \mathsf{AC\text{-}K} \times \mathsf{AC\text{-}Ct}_1 \times \cdots \times \mathsf{AC\text{-}Ct}_n \to \{0,1\}$, for some set $\mathsf{AC\text{-}K}$.
- The function class $\mathcal{F} \times \mathsf{AC\text{-}K}$ contains $(F_\lambda, \mathsf{ac\text{-}k})$ having public inputs $(\mathcal{Z}_{\lambda,i})_{i\in[n]}$.

A plaintext for client i consists of $x_i \in \mathcal{D}_{\lambda,i}$, where $\mathcal{D}_{\lambda,i}$ denotes the domain from which each client i gets their inputs. The corresponding ciphertext can be decrypted to $F_\lambda(x)$ using the functional key $\mathsf{sk}_{F_\lambda,\mathsf{ac\text{-}k}}$ for $\mathsf{ac\text{-}k} \in \mathsf{AC\text{-}K}$ if and only if $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$. Given the above specialization, the syntax of MCFE with access control can be derived from the general syntax of MCFE with public inputs in Definition 6.2. For the sake of analysis of our scheme later on, we give below only the *correctness* and *security* definitions for the specialized function class $\mathcal{F} \times \mathsf{AC\text{-}K}$.

Correctness. For sufficiently large $\lambda \in \mathbb{N}$, for all $(\mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda)$, $(F_\lambda, \mathsf{ac\text{-}k}) \in \mathcal{F} \times \mathsf{AC\text{-}K}$ and $\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}} \leftarrow \mathsf{Extract}(\mathsf{msk}, F_\lambda, \mathsf{ac\text{-}k})$, for all tag and $(\mathsf{ac\text{-}ct}_i)_i$, for all $(x_i)_{i\in[n]} \in \mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n}$, the following holds with overwhelming probability: if $\mathsf{Rel}(\mathsf{ac\text{-}k}, (\mathsf{ac\text{-}ct}_i)_i) = 1$ and $F_\lambda(x_1, \ldots, x_n) \neq \perp$

$$\mathsf{Dec}\left(\mathsf{dk}_{F_\lambda,\mathsf{ac\text{-}k}}, (\mathsf{Enc}(\mathsf{ek}_i, x_i, z_i \coloneqq (\mathsf{tag}, \mathsf{ac\text{-}ct}_i)))_{i\in[n]}\right) = F_\lambda(x_1, \ldots, x_n)$$

where $F_\lambda : \mathcal{D}_{\lambda,1} \times \cdots \times \mathcal{D}_{\lambda,n} \to \mathcal{R}_\lambda$ and the probability is taken over the coins of algorithm.

Security. The security game is depicted in Figure 6.1, where the functionality class is $\mathcal{F} \times \mathsf{AC\text{-}K}$, the set of public data for each client i is $\mathcal{Z}_{\lambda,i} := \mathsf{Tag} \times \mathsf{AC\text{-}Ct}_i$. We recall that our general admissibility in Definition 6.9 allows an adversary to query *multiple* times to the challenge oracle for a fixed (i, tag). In particular, we consider also attacks where multiple $\mathbf{x}_i^{(\mathsf{rep})}$ are queried for the same (i, tag) to the oracle **LoR**, namely with *repetitions* at position i under the challenge tag tag. The formal definition, which is concretely interpreted for the class $\mathcal{F} \times \mathsf{AC\text{-}K}$ based on the general Definition 6.9 of MCFE with public inputs, is given below.

Definition 6.9 (Admissible adversaries with fine-grained access control). *Let $\mathcal{A}$ be a* ppt *adversary and let $\mathcal{E} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ be an* MCFE scheme with fine-grained access control *for the functionality class $\mathcal{F} \times \mathsf{AC\text{-}K}$. In the security game given in Figure 6.1 for $\mathcal{A}$ considering $\mathcal{E}$, let the sets $(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ be the sets of corrupted clients, functional key queries, and honest clients, in that order. We say that $\mathcal{A}$* is NOT admissible w.r.t *$(\mathcal{C}, \mathcal{Q}, \mathcal{H})$ if any of the following conditions holds:*

> *There exist $\mathsf{tag} \in \mathsf{Tag}$, a function $(F, \mathsf{ac\text{-}k}) \in \mathcal{Q}$ is queried to* **Extract***, two challenges $(x_i^{(0)}, x_i^{(1)}, (\mathsf{tag}, \mathsf{ac\text{-}ct}_i))_{i\in[n]}$ are queried to* **LoR***, with public inputs $\mathsf{ac\text{-}ct}_i \in \mathsf{AC\text{-}Ct}_{\lambda,i}$, a pair $(\mathbf{t}^{(0)}, \mathbf{t}^{(1)}, \mathbf{v}^{(chal)})$ so that for $b \in \{0,1\}$, $\forall\, i \in \mathcal{H} : \mathbf{t}^{(b)}[i] = x_i^{(b)}$ and $\mathbf{v}^{(chal)}[i] = \mathsf{ac\text{-}ct}_i$, and*
>
> - *The policy passes[a]: $\mathsf{Rel}(\mathsf{ac\text{-}k}, \mathbf{v}^{(chal)}) = 1$.*
> - *The function evaluation differs:*
>
> $$F\left(\mathbf{t}^{(0)}\right) \neq F\left(\mathbf{t}^{(1)}\right) \quad . \tag{6.4}$$
>
> [a]This is up to attributes replacement in the corrupted slots $i \in \mathcal{C}$, therefore we only required $\mathbf{v}^{(chal)}$ to coincide with only with the *honest* attributes $(\mathsf{ac\text{-}ct}_i)_{i\in\mathcal{H}}$ and leave free the *corrupted* part.

Otherwise, we say that $\mathcal{A}$ is admissible w.r.t *$(\mathcal{C}, \mathcal{Q}, \mathcal{H})$.*

We recall the weaker notion considering only *complete* queries, while facing repetitions, for this concrete $\mathcal{F} \times \mathsf{AC\text{-}K}$.

Weaker notions. We can relax Definition 6.9 to obtain weaker notions, in a similar manner which we use to relax Definition 6.3. The *selective*, *private-input only repetitions*, *complete*, and *one-time* security relaxations are straightforward.

6.4.2 Extension to Sub-vectors

In this section we present an MCFE scheme with fine-graine access control whose i-th ciphertext can encrypt *subvectors* of length N_i. In Remark 6.14 we discuss how to turn our final MCFE for inner products with access control, into an MIFE in the standard model, for computing inner products without access control. The bilinear group is $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$. The function class of interests is $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ as defined in Definition 6.10.

Definition 6.10 (Inner Products with LSSS). *We consider the functionality $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ and $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec}}$ that contains $F_{\mathbf{y}_1,\ldots,\mathbf{y}_n} : \prod_{i\in[n]} \left(\mathbb{Z}_q^{N_i}\right) \to \mathbb{Z}_q$ defined as $F_{\mathbf{y}_1,\ldots,\mathbf{y}_n}(\mathbf{x}_1, \ldots, \mathbf{x}_n) := \sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i \rangle$, which receives as inputs and parameters where for all i, $\max(\|\mathbf{x}_i\|_\infty, \|\mathbf{y}_i\|_\infty) < B$, with $B = \mathsf{poly}(\lambda) \in \mathbb{N}$ being a polynomial. The access control is given by $\mathsf{Rel} : \mathsf{LSSS} \times \left(\prod_{i=1}^n 2^{\mathsf{Att}}\right) \to \{0,1\}$ as $\mathsf{Rel}(\mathbb{A}, (\mathsf{S}_i)_i) = \prod_i \mathbb{A}(\mathsf{S}_i)$. The class LSSS contains Linear Secret Sharing Schemes over Att, and 2^{Att} denotes the superset of an attribute space $\mathsf{Att} \subseteq \mathbb{Z}_q$.*

Construction. The details are given below:

$\underline{\mathsf{Setup}(1^\lambda)}$**:** Choose $n+1$ pairs of dual orthogonal bases $(\mathbf{H}_i, \mathbf{H}_i^*, \mathbf{B}_i, \mathbf{B}_i^*)$ for $i \in [n]$ and $(\mathbf{F}, \mathbf{F}^*, \mathbf{G}, \mathbf{G}^*)$,$(\mathbf{H}_i, \mathbf{H}_i^*)$ is a pair of dual bases for $(\mathbb{G}_1^{2N+4}, \mathbb{G}_2^{2N+4})$, $(\mathbf{B}_i, \mathbf{B}_i^*)$ is a pair of dual bases for $(\mathbb{G}_1^{N+4}, \mathbb{G}_2^{N+4})$, $(\mathbf{F}, \mathbf{F}^*)$ is a pair of dual bases for $(\mathbb{G}_1^{2N+6}, \mathbb{G}_2^{2N+6})$, $(\mathbf{G}, \mathbf{G}^*)$ is a pair of dual bases for $(\mathbb{G}_1^{2N+6}, \mathbb{G}_2^{2N+6})$ [3]. Sample $\mu \xleftarrow{\$} \mathbb{Z}_q^*, \mathbf{S}, \mathbf{U}, \xleftarrow{\$} \prod_{i=1}^n (\mathbb{Z}_q^*)^N$ and write $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_n)$, $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_n)$. Perform an n-out-of-n secret sharing on 1, that is, choose $p_i \in \mathbb{Z}_q$ such that $1 = p_1 + \cdots + p_n$. Then, for each $i \in [n]$, sample N random values $\theta_{i,k} \xleftarrow{\$} \mathbb{Z}_q$. Output the master secret key and the encryption keys as

$$\begin{cases} \mathsf{msk} \coloneqq \Big(\mathbf{S},\ \mathbf{U},\ (\theta_{i,k})_{i\in[N],k\in[N]}, (\mathbf{b}^*_{i,k})_{k\in[N+2]},\ \mathbf{f}_1^*,\ \mathbf{f}_2^*,\ \mathbf{f}_3^*, \\ \qquad \mathbf{g}_1^*,\ \mathbf{g}_2^*,\ \mathbf{g}_3^*,\ (\mathbf{h}^*_{i,1}, \mathbf{h}^*_{i,2}, \mathbf{h}^*_{i,3}, (\mathbf{h}^*_{i,N+3+k})_{k=1}^N)_{i\in[n]}\Big) \\ \mathsf{ek}_i \coloneqq \Big(\mathbf{s}_i,\ \mathbf{u}_i, (B_i^{(k)})_{k\in[N+2]}, \mathbf{b}_{i,N+3},\ \mathbf{f}_1,\ \mathbf{f}_2,\ \mathbf{f}_3, \\ \qquad \mathbf{g}_1,\ \mathbf{g}_2,\ \mathbf{g}_3,\ p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ (\theta_{i,k}\mathbf{h}_{i,N+3+k})_{k=1}^N\Big) \end{cases}$$

where $H_i^{(k)}, B_i^{(k)}$ denotes the k-th row of H_i, B_i respectively.

$\underline{\mathsf{Extract}(\mathsf{msk}, (\mathbf{y}_i)_{i\in[n]} \in \prod_{i=1}^n \mathbb{Z}_q^N, \mathsf{ac\text{-}k} \coloneqq \mathbb{A})}$**:** Let $\mathbb{A}$ be an LSSS-realizable monotone access structure over a set of attributes $\mathsf{Att} \subseteq \mathbb{Z}_q$. First, sample $a_{i,0} \xleftarrow{\$} \mathbb{Z}_q$ and run the labeling algorithm $\Lambda_{a_{i,0}}(\mathbb{A})$ (see Definition 3.3) to obtain the labels $(a_{i,j})_j$ where j runs over the attributes in Att. In the end, it holds that $a_{i,0} = \sum_{j\in A} c_{i,j} \cdot a_{i,j}$ where j runs over some authorized set $A_i \in \mathbb{A}$ and $\mathbf{c}_i = (c_{i,j})_j$ is the reconstruction vector from LSSS w.r.t A_i. We denote by $\mathsf{List\text{-}Att}(\mathbb{A})$ the list of attributes appearing in $\mathbb{A}$, with possible repetitions. For each $i \in [n]$, each $k \in [N]$, sample $d_{\mathbb{A},i,k} \xleftarrow{\$} \mathbb{Z}_q$ such that $\sum_{i=1}^n \sum_{k=1}^N \theta_{i,k} d_{\mathbb{A},i,k} = 0$. For each $i \in [n]$, compute

$$\mathbf{m}_i \coloneqq \left(\mathbf{y}_i, \sum_{i=1}^n a_{i,0}, \mathsf{rnd}_i, 0, 0\right)_{\mathbf{B}_i^*}; \widetilde{\mathbf{m}}_{i,j} \coloneqq (\tilde{\pi}_{i,j} \cdot (j,1),\ a_{i,j},\ 0^N,\ 0,\ 0^N,\ 0,\ 0)_{\mathbf{G}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})$$

$$\mathbf{k}_{i,j} \coloneqq (\pi_{i,j} \cdot (j,1),\ a_{i,j} \cdot z,\ 0^N,\ 0,\ 0^N,\ 0,\ 0)_{\mathbf{F}^*} \text{ for } j \in \mathsf{List\text{-}Att}(\mathbb{A})$$

$$\mathbf{k}_{i,\mathsf{ipfe}} \coloneqq \left(\sum_{i=1}^n \langle \mathbf{s}_i, \mathbf{y}_i\rangle, \sum_{i=1}^n \langle \mathbf{u}_i, \mathbf{y}_i\rangle,\ a_{i,0} \cdot z,\ 0^N,\ (d_{\mathbb{A},i,k})_{k=1}^N, \mathsf{rnd}_{i,\mathsf{ipfe}}\right)_{\mathbf{H}_i^*}$$

where $z, \pi_{i,j}, \mathsf{rnd}_i, \mathsf{rnd}_{i,\mathsf{ipfe}} \xleftarrow{\$} \mathbb{Z}_q$. Output $\mathsf{dk}_{\mathbb{A},\mathbf{y}} \coloneqq \left(\left(\mathbf{k}_{i,j}, \widetilde{\mathbf{m}}_{i,j}\right)_{i,j}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]}\right)$.

$\underline{\mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_i \in \mathbb{Z}_q^N, z_i \coloneqq (\mathsf{tag}, \mathsf{S}_i))}$**:** Parse

$$\mathsf{ek}_i \coloneqq \left(\mathbf{s}_i,\ \mathbf{u}_i, (B_i^{(k)})_{k\in[N+2]}, \mathbf{b}_{i,N+3}, \mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3, p_i \cdot H_i^{(1)},\ p_i \cdot H_i^{(2)},\ \mathbf{h}_{i,3},\ (\theta_{i,k}\mathbf{h}_{i,N+3+k})_{k=1}^N\right)$$

and $\mathsf{S}_i \subseteq \mathsf{Att} \subseteq \mathbb{Z}_q$ as the set of attributes, compute $\mathsf{H}(\mathsf{tag}) \to (\llbracket\omega\rrbracket_1, \llbracket\omega'\rrbracket_1) \in \mathbb{G}_1^2$. Use $p_i H_i^{(1)}$ and $p_i H_i^{(2)}$ to compute

$$p_i H_i^{(1)} \cdot \llbracket\omega\rrbracket_1 + p_i H_i^{(2)} \cdot \llbracket\omega'\rrbracket_1 = p_i \cdot \left(\omega H_i^{(1)} \cdot g_1 + \omega' H_i^{(2)} \cdot g_1\right) = p_i \cdot (\omega \mathbf{h}_{i,1} + \omega' \mathbf{h}_{i,2}) \ .$$

For each $j \in \mathsf{S}_i$, sample $\psi_i, \nu_i \xleftarrow{\$} \mathbb{Z}_q$ and compute

$$\widetilde{\mathbf{t}}_{i,j} = (\widetilde{\sigma}_{i,j} \cdot (1,-j),\ \nu_i,\ 0^N,\ 0,\ 0^N,\ 0,\ 0)_{\mathbf{G}}$$
$$\mathbf{c}_{i,j} = \sigma_{i,j} \cdot \mathbf{f}_1 - j \cdot \sigma_{i,j} \cdot \mathbf{f}_2 + \psi_i \cdot \mathbf{f}_3 = (\sigma_{i,j} \cdot (1,-j),\ \psi_i,\ 0^N,\ 0,\ 0^N,\ 0,\ 0)_{\mathbf{F}}$$

where $\widetilde{\sigma}_{i,j}, \sigma_{i,j} \xleftarrow{\$} \mathbb{Z}_q$. Finally, compute

$$\mathbf{t}_i \coloneqq \sum_{k\in[N]} \left(\llbracket\omega\rrbracket_1 \cdot \mathbf{s}_i[k] \cdot B_i^{(k)} + \llbracket\omega'\rrbracket_1 \cdot \mathbf{u}_i[k] \cdot B_i^{(k)} + \llbracket\mathbf{x}_i[k]\rrbracket_1\right) + \nu_i \cdot \mathbf{b}_{i,N+1} + \rho_i \cdot \mathbf{b}_{i,N+3}$$
$$= (\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i, \nu_i, 0, \rho_i)_{\mathbf{B}_i}$$

$$\mathbf{c}_{i,\mathsf{ipfe}} \coloneqq p_i \cdot (\omega \cdot \mathbf{h}_{i,1} + \omega' \cdot \mathbf{h}_{i,2}) + \psi_i \cdot \mathbf{h}_{i,3} + \sum_{k=1}^N \theta_{i,k}\mathbf{h}_{i,N+3+k} = (\omega p_i,\ \omega' p_i,\ \psi_i,\ 0^N,\ (\theta_{i,k})_{k=1}^N,\ 0)_{\mathbf{H}_i}$$

and output $\mathsf{ct}_{\mathsf{tag},i} \coloneqq \left(\left(\mathbf{c}_{i,j}, \widetilde{\mathbf{t}}_{i,j}\right)_j, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}}\right)$.

---

[3] We denote the basis changing matrices for $(\mathbf{F}, \mathbf{F}^*), (\mathbf{B}_i, \mathbf{B}_i^*), (\mathbf{H}_i, \mathbf{H}_i^*)$ as $(F, F' \coloneqq (F^{-1})^\top), (B_i, B_i' \coloneqq (B_i^{-1})^\top), (H_i, H_i' \coloneqq (H_i^{-1})^\top)$ respectively (see Section 3.3 for basis changes in DPVS).

$\mathsf{Dec}(\mathsf{dk}_{\mathbb{A},\mathbf{y}}, \mathbf{c} \coloneqq (\mathsf{ct}_{\mathsf{tag},i}), \mathsf{aux\text{-}d} \coloneqq \mathsf{tag})$**:** Parse

$$\mathsf{ct}_{\mathsf{tag},i} = \left(\left(\mathbf{c}_{i,j}, \widetilde{\mathbf{t}}_{i,j}\right)_j, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}}\right) \quad \text{and} \quad \mathsf{dk}_{\mathbb{A},\mathbf{y}} \coloneqq \left(\left(\mathbf{k}_{i,j}, \widetilde{\mathbf{m}}_{i,j}\right)_{i,j}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]}\right) .$$

For each $i \in [n]$, if there exists $A_i \subseteq \mathsf{S}_i$ and $A_i \in \mathbb{A}$, then compute the reconstruction vector $(c_{i,j})_j$ of for A_i and perform Algorithm 6.2. Finally, compute the discrete logarithm and output the small value $\mathsf{out} \in [-nNB^2, nNB^2] \subsetneq \mathbb{Z}_q$ [4].

Input: $\mathsf{ct}_{\mathsf{tag},i} = \left(\left(\mathbf{c}_{i,j}, \widetilde{\mathbf{t}}_{i,j}\right)_j, \mathbf{t}_i, \mathbf{c}_{i,\mathsf{ipfe}}\right)$ and $\mathsf{dk}_{\mathbb{A},\mathbf{y}} \coloneqq \left(\left(\mathbf{k}_{i,j}, \widetilde{\mathbf{m}}_{i,j}\right)_{i,j}, (\mathbf{m}_i, \mathbf{k}_{i,\mathsf{ipfe}})_{i\in[n]}\right)$, as well as the reconstruction vector $(c_{i,j})_j$ of the LSSS for a reconstruction set A_i for each i

1. For each j in the reconstruction set A, compute

$$\widetilde{\mathbf{t}}_{0,j} = \sum_i \widetilde{\mathbf{t}}_{i,j} = (\widetilde{\sigma}_{0,j} \cdot (1, -j),\ \sum_i \nu_i,\ 0^N,\ 0,\ 0^N,\ 0,\ 0)_{\mathbf{G}}$$

   where $\widetilde{\sigma}_{0,j} = \sum_i \widetilde{\sigma}_{i,j}$ being a uniformly random value as $\widetilde{\sigma}_{i,j} \xleftarrow{\$} \mathbb{Z}_q$.

2. For each i compute

$$X_i = \sum_{j\in A_i} \widetilde{\mathbf{t}}_{0,j} \times (c_{i,j} \cdot \widetilde{\mathbf{m}}_{i,j}) = \left[\!\!\left[(\sum_i \nu_i) \cdot \left(\sum_{j\in A_i} c_{i,j} \cdot a_{i,j} \right) \right]\!\!\right]_{\mathsf{t}} = \left[\!\!\left[(\sum_i \nu_i) \cdot a_{i,0} \right]\!\!\right]_{\mathsf{t}}$$
$$Y_i = \sum_{j\in A_i} \mathbf{c}_{i,j} \times (c_{i,j} \cdot \mathbf{k}_{i,j}) = [\![\psi_i \cdot a_{i,0} \cdot z]\!]_{\mathsf{t}}$$

   and in the end summing all X_i to obtain $\mathsf{mask} = \sum_i X_i = \left[\!\!\left[(\sum_i \nu_i) \cdot (\sum_i a_{i,0}) \right]\!\!\right]_{\mathsf{t}}$

3. Compute

$$W = \sum_i \mathbf{t}_i \times \mathbf{m}_i = \left[\!\!\left[\sum_i \left(\omega \cdot \langle \mathbf{s}_i, \mathbf{y}_i \rangle + \omega' \cdot \langle \mathbf{u}_i, \mathbf{y}_i \rangle + \langle \mathbf{x}_i, \mathbf{y}_i \rangle \right) + (\sum_i \nu_i) \cdot (\sum_i a_{i,0}) \right]\!\!\right]_{\mathsf{t}}$$

   as well as $Z = \sum_i \left(\mathbf{c}_{i,\mathsf{ipfe}} \times \mathbf{k}_{i,\mathsf{ipfe}} - Y_i \right) = \left[\!\!\left[\omega \cdot \sum_i \langle \mathbf{s}_i, \mathbf{y}_i \rangle + \omega' \cdot \sum_i \langle \mathbf{u}_i, \mathbf{y}_i \rangle \right]\!\!\right]_{\mathsf{t}}$ thanks to $\sum_{i=1}^{n} \sum_{k=1}^{N} \theta_{i,k} d_{\mathbb{A},i,k} = 0$ and $\sum_i p_i = 1$.

4. Finally, compute $\mathsf{out} = W - Z - \mathsf{mask} = \left[\!\!\left[\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \right]\!\!\right]_{\mathsf{t}}$ and then a discrete log of out in base g_{t} to obtain $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle$.

Figure 6.2: The final computation of decryption for the MCFE in Section 6.4, whose *correctness* can be verified according to construction. We note that because of all the probabilistic vectorization during encryption (cf. **Solution to the First Obstacle** in Section 6.2), the decryption algorithm is much more complex than the one in [NPP22a] and [NPP23a].

We now state the security theorem. For simplicity, this theorem proves a weaker notion following Definition 6.3. In the subsequent lemmas we will show how to remove most of the above constraints.

Theorem 6.11. *Let $\mathcal{E}$ be a MCFE scheme with fine-grained access control for the function class $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$, given in Section 6.4 in a bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_{\mathsf{t}}, g_1, g_2, g_{\mathsf{t}}, \mathbf{e}, q)$. Then, in the random oracle model, $\mathcal{E}$ is* one-time statically IND-secure *against* complete *challenge queries with* private-inputs only repetitions *(as per Definition 6.3), under the* SXDH *in $\mathbb{G}_1$ and $\mathbb{G}_2$.*

[4] we represent $\mathbb{Z}_q$ as the ring of integers with addition and multiplication modulo q, containing the representatives in the interval $(-q/2, q/2)$.

Concrete Interpretation of Admissibility. Before going into the proof, we present specific conditions for admissible attacks in the case of *one-challenge, complete, with repetitions on private inputs* with repect to Definition 6.3:

1. For all vectors $(\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)}, (\mathsf{tag}, \mathsf{S}_i))$ that is queried to **LoR**, for all $((\mathbf{y}_i)_{i\in[n]}, \mathbb{A}) \in \mathcal{Q}$, let $\mathcal{H}$ be the set of honest clients and $b \xleftarrow{\$} \{0,1\}$ be the challenge bit. Then fof any $j_i \in [J_i]$, if $\prod_i \mathbb{A}(\mathsf{S}_i) = 1$ then: $\sum_{i\in\mathcal{H}} \langle \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i \rangle = 0$. This implies $\langle \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i \rangle$ is constant for any $j_i \in [J_i]$. We recall that we are in the *private-inputs only repetitions* and therefore there are no repetitions over $(\mathsf{tag}, \mathsf{S}_i)$.

2. For all vectors $(\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)}, (\mathsf{tag}, \mathsf{S}_i))$ that is queried to **LoR**, for all $((\mathbf{y}_i)_{i\in[n]}, \mathbb{A}) \in \mathcal{Q}$. Let $\mathcal{C} := [n] \setminus \mathcal{H}$ be the set of corrupted clients. Then, for all $i \in \mathcal{C}$, all $j_i \in [J]$: $\langle \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i \rangle = 0$.

We recall that these conditions are for the *one-challenge, complete, with repetitions on private inputs* case and are checked in **Finalise** procedure at the end of the security experiment. Particularly, condition 2 is checked for all corrupted clients $i \in \mathcal{C}$ and all $j_i \in [J]$, given any queries that are made to the oracle **LoR** for $i \in \mathcal{C}$ by the adversary. This makes sense even in the case of *static corruption*, since we do *not* prohibit such queries even after the set $\mathcal{C}$ is fixed. Finally, condition 2 does *not* need to cover private inputs of corrupted $i \in \mathcal{C}$ that are not queried to the oracle **LoR** because there exists no challenge bit b in those self-crafted ciphertexts $\mathsf{ct}_{\mathsf{tag},i} \leftarrow \mathsf{Enc}(\mathsf{ek}_i, \mathbf{z}_i, (\mathsf{tag}, \mathsf{S}_i))$. Decrypting $\mathsf{ct}_{\mathsf{tag},i}$ jointly with others challenge ciphertexts $\mathsf{ct}^{(b)}_{\mathsf{tag},j\neq i}$ under some key $\mathsf{dk}_{\mathbb{A},(\mathbf{y}_i)_{i\in[n]}}$ always gives the same i-th component $\langle \mathbf{z}_i, \mathbf{y}_i \rangle$ regardless of b. Last but not least, as a corollary of Theorem 5.13, the above admissibility is *optimal* for the class $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ in Definition 6.10.

Proof Strategy. Before presenting the details of the proof, we give an overview of the strategy. The sequence of games is given in Figure 6.3, 6.4, and 6.5. The high level objective of each step is described:

G_0**:** We start from the first game G_0 which is the security experiment for *one-time statically IND-security* against *complete* challenge queries with *private-inputs only repetitions*. For simplicity, we add a constraint that the challenge tag tag is *not* queried to **Enc**. This incurs a multiplicative loss factor in advantage up to an inverse of polynomial in λ, where we can reduce to the normal $\mathsf{1chal}$ by guessing the challenge tag among the tags for encryption, and responding all of its **Enc** queries $(i, \mathbf{x}_i, (\mathsf{tag}, \mathsf{ac\text{-}ct}_i))$ by $\mathbf{LoR}(i, \mathbf{x}_i, \mathbf{x}_i, (\mathsf{tag}, \mathsf{ac\text{-}ct}_i))$.

$\mathsf{G}_0 \to \mathsf{G}_1$**:** To go to G_1, we perform a sequence of hybrids over the key queries, which are indexed by $\ell \in [K]$. The main goal is to introduce $\Delta\mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ for each client $i \in \mathcal{H}$ (known in advance by static corruption) and repetition $j_i \in [J_i]$ in one (block of) coordinates of the challenge components $\mathbf{c}_{i,j}^{(j_i)}$. The corresponding (block of) coordinates in the key component $\mathbf{k}_{i,j}^{(\ell)}$ will be modified accordingly to contain a random copy of $R \cdot \mathbf{y}_i$ for some random $R \xleftarrow{\$} \mathbb{Z}_q$. The details of the reductions are given in the full proof below, we highlight here the fact that the *correctness* is necessarily preserved thanks to the admissiblity. When the key allows decryption, summing up over all honest clients $i \in \mathcal{H}$ contains

$$R \cdot \sum_{i\in\mathcal{H}} \langle \Delta\mathbf{x}_i, \mathbf{y}_i \rangle = R \cdot \sum_{i\in\mathcal{H}} \langle \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i \rangle = 0 \ . \tag{6.5}$$

Condition 1 ensures first that $\mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ is constant for all $j_i \in [J_i]$[5] and the sum over index $i \in \mathcal{H}$ is well defined. Finally this sum leads to (6.5) which is 0 and does not

[5]This term $\Delta\mathbf{x}_i = \mathbf{0}$ the vector of all 0 when $b = 1$ and can be non-zero when $b = 0$.

intervene the correct decryption[6]. As a final remark, this step exploits the *probabilistic vectorization* and its randomness (see overview in Section 6.2)): first to apply Lemma 3.5 and then follow-ups with the *complexity leveraging* argument, while taking into accounts the private-input repetitions (**Solution to the First/Second Obstacle** in Section 6.2).

$\mathsf{G}_1 \to \mathsf{G}_2$**:** After the hybrids $\mathsf{G}_0 \to \mathsf{G}_1$, we proceeed to G_2 to rewrite the adversary's view of the challenge ciphertext component on the aggregation of the honest i for $\widetilde{\mathbf{t}}_{i,j}^{(j_i)}$: $\widetilde{\mathbf{t}}_{0,j}^{(j_i)} = \sum_{i \in \mathcal{H}} \widetilde{\mathbf{t}}_{i,j}^{(j_i)}$. Thanks to *static* corruption, the set $\mathcal{H}$ is known in advance and $\widetilde{\mathbf{t}}_{0,j}^{(j_i)}$ is well defined. This is a completely formal rewriting that conforms to the calculations in the decryption algorithm (Algorithm 6.2) and hence preserves *correctness*.

$\mathsf{G}_2 \to \mathsf{G}_3$**:** In the next step, we proceed to G_3 by applying the masking lemma (Lemma 3.5), over the each key $\left(\left(\mathbf{k}_{i,j}^{(\ell)}, \widetilde{\mathbf{m}}_{i,j}^{(\ell)}\right)_{i,j}, (\mathbf{m}_i^{(\ell)}, \mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)})_{i \in [n]}\right)$ that is indexed by $\ell \in [K]$. This masking application introduces $\Delta\mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ for each client $i \in \mathcal{H}$ (known in advance by static corruption) and repetition $j_i \in [J_i]$ in one (block of) coordinates of the challenge components $\mathbf{t}_i^{(j_i)}$, while the corresponding (block of) coordinates in the key component $\mathbf{m}_i^{(\ell)}$ will be modified accordingly to contain $R \cdot \mathbf{y}_i$. We remark that this pair of masks are the same as what are introduce in the step $\mathsf{G}_0 \to \mathsf{G}_1$, which is feasible under Lemma 3.5, and are needed for later steps in the proof. The correctness is preserved thanks to a similar argument as in the previous step.

$\mathsf{G}_3 \to \mathsf{G}_4$**:** We move to the *complexity leveraging* argument. As already briefly introduced in paragraph **Solution to the Third Obstacle** of Section 7.1, the complexity leveraging argument is a technique that unfolds as follows:

1. We define an event E that happens with fixed probability and whose probability space depends on the data that can be *adaptively* chosen by the adversary. Then, condition on E we move to the *selective* version $\mathsf{G}_i^*, \mathsf{G}_{i+1}^*, \ldots, \mathsf{G}_{i+K}^*$.
2. Next, we want to prove the sequence of *perfect indistinguishability* involving

$$\{\mathsf{G}_i^* \mid E\} \equiv \{\mathsf{G}_{i+1}^* | E\} \equiv \cdots \equiv \{\mathsf{G}_{i+K}^* \mid E\} \tag{6.6}$$

   where E happens with fixed probability and is *independent* of the view of the adversary during the reductions $\{\mathsf{G}_{i+t}^* \mid E\} \equiv \{\mathsf{G}_{i+t+1}^* | E\}$ in the sequence, for all $t \in [K-1]$.
3. We go back to the orginal *adaptive* games, without resorting to event E, a probabilistic argument concludes that $\{\mathsf{G}_i\} \equiv \{\mathsf{G}_{i+1}\} \equiv \cdots \equiv \{\mathsf{G}_{i+K}\}$. The main idea is given any ppt adaptive adversary, we can construct a simulator of the adaptive games $\{\mathsf{G}_i, \mathsf{G}_{i+1}, \ldots, \mathsf{G}_{i+K}\}$ can *(i)* first guess the adaptively chosen data for event E, *(ii)* interact with its selective challenger, while *(iii)* using the afterwards selective challenger's responses to interact with the adaptive adversary, and *(iv)* in the end, only when E holds, forward the adaptive adversary's final result to the selective challenger.

In the reduction of step 3, the guess at *(i)* is done by the simulator and following the check at *(iv)*, it incurs the simulator's advantage against the selective games being equal to a fixed loss factor $\Pr[E]$ mutiplied to the advantage of the adaptive adversary. However, thanks to the perfect indistinguishability (6.6), between the selective games for all simulators the advantage is 0. Therefore, for the particular above simulator the advantage is also 0 and that implies the arbitrary adaptive adversary's advantage is 0. It remains constructing the selective games $\{\mathsf{G}_i^*, \mathsf{G}_{i+1}^*, \ldots, \mathsf{G}_{i+K}^*\}$ and proving the perfect indistinguishability (6.6). To this end, we exploit formal basis changes in DPVS (cf. examples 1, 2, 3 in Section 3.3).

[6]There is a step in this transition we already use *complexity leveraging*, for a common explanation we refer to $\mathsf{G}_3 \to \mathsf{G}_4$ below.

In a simplified notation the (block of) coordinares in ciphertexts and keys are changed as follows:

$$\text{(Formal quotient)} \quad \begin{cases} \mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)} & = (\cdots, \boxed{r \cdot \mathbf{1}_{\Delta\mathbf{x}_i}}, \boxed{r' \cdot \mathbf{1}}, \cdots)_{\mathbf{H}_i}; \\ \mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)} & = (\cdots, \boxed{R \cdot (\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)})}, \boxed{(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^{N}}, \cdots)_{\mathbf{H}_i^*}; \end{cases}$$

$$\text{(Formal switch, } b \text{ to } 1) \equiv \begin{cases} \mathbf{t}_i^{(j_i)} & = (\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \boxed{\mathbf{x}_i^{(1,j_i)}}, \cdots, \Delta\mathbf{x}_i, \cdots)_{\mathbf{B}_i} \\ \mathbf{m}_i^{(\ell)} & = (\mathbf{y}_i^{(\ell)}, \cdots, \boxed{R' \cdot \mathbf{y}_i^{(\ell)}}, \cdots)_{\mathbf{B}_i^*} \\ \mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)} & = (\cdots, r \cdot \mathbf{1}_{\Delta\mathbf{x}_i}, \boxed{(r'+r)}\mathbf{1}, \cdots)_{\mathbf{H}_i} \\ \mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)} & = (\cdots, \boxed{R' \cdot \left(\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)}\right)}, \boxed{(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^{N}}, \cdots)_{\mathbf{H}_i^*} \end{cases}$$

$$\text{(Redo formal quotient)} \equiv \begin{cases} \mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)} & = (\cdots, \boxed{\Delta\mathbf{x}_i}, \boxed{(\theta_{i,k})_{k=1}^{N}}, 0)_{\mathbf{H}_i} \\ \mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)} & = (\cdots, \boxed{R' \cdot \mathbf{y}_i^{(\ell)}}, \boxed{(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^{N}}, \mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)})_{\mathbf{H}_i^*} \end{cases}$$

where $R, R' \xleftarrow{\$} \mathbb{Z}_q$, $\Delta\mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ is constant for $i \in \mathcal{H}$ over repetitions. We refer to the definition of the event for guesses in (6.10), which ensures that under those formal basis changes correctness is preserved necessarily and we obtain the desired effects on vectors.

$\mathsf{G}_4 \to \mathsf{G}_5$**:** The remaining step is to clean auxiliary coordinates we have modified in the previous steps.

Full Proof. The full proof that develops all details of the above steps is given below.

Proof (Of Theorem 6.11). The sequence of games can be found in Figure 6.3, 6.4, and 6.5. The full-domain hash function $\mathsf{H} : \mathsf{Tag} \times 2^{\mathsf{Att}} \to \mathbb{G}_1^2$ is modeled as a random oracle and we denote by Q the number of random oracle queries by the adversary. The changes that make the transitions between games are highlighted in boxed. The advantage of an adversary $\mathcal{A}$ in a game G_i is denoted by $\mathsf{Adv}(\mathsf{G}_i) := |\Pr[\mathsf{G}_i = 1] - 1/2|$ where the probability is taken over the random choices of $\mathcal{A}$ and coins of G_i.

Game G_0**:** This is the adaptive security game, where the private-input repetitions at each position $i \in [n]$ are indexed by $\mathsf{rep} \in [J_i]$ where J_i is the maximum repetitions queried for position i. We note that for different i, the bound J_i can be different. The challenge ciphertext encrypts subvectors $\mathbf{x}_i^{(b,\mathsf{rep})} \in \mathbb{Z}_q^N$. For simplicity, we add a constraint that the challenge tag tag is *not* queried to **Enc**. This incurs a multiplicative loss factor in advantage up to an inverse of polynomial in λ, where we can reduce to the normal $\mathsf{1chal}$ by guessing the challenge tag among the tags for encryption, and responding all of its **Enc** queries $(i, \mathbf{x}_i, (\mathsf{tag}, \mathsf{ac\text{-}ct}_i))$ by $\mathbf{LoR}(i, \mathbf{x}_i, \mathbf{x}_i, (\mathsf{tag}, \mathsf{ac\text{-}ct}_i))$.

Game G_1**:** We perform a sequence of hybrids over the key queries $(\mathbf{y}_i^{(\ell)})_i$ for $\ell \in [K]$. We denote $\mathsf{G}_{0.\ell}$ the hybrid where all the $\leq (\ell-1)$-th key is programmed as responses from **LoR**

$$\begin{array}{c|c|c|c|c|c} \mathbf{t}_i^{(j_i)} & (\ \omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(b,j_i)} & \nu_i^{(j_i)} & \mathbf{0} & 0 & \rho_i^{(j_i)} \ )_{\mathbf{B}_i} \\ \mathbf{m}_i^{(\leq \ell-1)} & (\ \mathbf{y}_i^{(\leq \ell-1)} & \sum_{i=1}^{n} a_{i,0}^{(\leq \ell-1)} & \mathbf{0} & \mathsf{rnd}_i^{(\leq \ell-1)} & 0 \ )_{\mathbf{B}_i^*} \end{array}$$

$$\begin{array}{c|c|c|c|c|c} \mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)} & (\ p_i\omega_{\mathsf{tag}} & p_i\omega'_{\mathsf{tag}} & \psi_i^{(j_i)} & \Delta\mathbf{x}_i & (\theta_{i,k})_{k=1}^{N} \ )_{\mathbf{H}_i} \\ \mathbf{k}_{i,\mathsf{ipfe}}^{(\leq \ell-1)} & (\ \sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\leq \ell-1)} \rangle & \sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\leq \ell-1)} \rangle & a_{i,0}^{(\leq \ell-1)} z^{(\leq \ell-1)} & \boxed{R \cdot \mathbf{y}_i^{(\leq \ell-1)}} & (d_{\mathbb{A},i,k}^{(\leq \ell-1)})_{k=1}^{N} \ )_{\mathbf{H}_i^*} \end{array}$$

while other ciphertext components from **Enc** are kept in normal form. It holds that $\mathsf{G}_0 = \mathsf{G}_{0.0}$. For $\ell \in [K]$, the transition from $\boxed{\mathsf{G}_{0.\ell-1}}$ to $\boxed{\mathsf{G}_{0.\ell}}$ is as follows: $\boxed{\mathsf{G}_{0.\ell.0}}$ is the same as $\boxed{\mathsf{G}_{0.\ell-1}}$.

$\boxed{\mathsf{G}_{0.\ell.1}}$ is the same as $\boxed{\mathsf{G}_{0.\ell.0}}$ except that we apply Lemma 3.5 to introduce a set of masks in the ciphertexts : $\Delta\mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$. The proof of Lemma 3.5 can be found in [NPP25, Appendix B]. We remark that $\Delta\mathbf{x}_i$ is a vector of differences of the challenge ciphertexts

Game $\mathsf{G}_0 : \mathsf{H}(\mathsf{tag}) \to (\llbracket \omega_{\mathsf{tag}} \rrbracket_1, \llbracket \omega'_{\mathsf{tag}} \rrbracket_1)$, $\mathsf{H}(\mathsf{tag}') \to (\llbracket \chi_{\mathsf{tag}'} \rrbracket_1, \left\llbracket \chi'_{\mathsf{tag}'} \right\rrbracket_1)$, (for **Enc** queries H on tag' are noted by χ and χ') ; $\ell \in [K]$ indexes key queries
$a_{i,0}^{(\ell)} \xleftarrow{\$} \mathbb{Z}_q$, $(a_{i,j}^{(\ell)})_{j \in \mathsf{List\text{-}Att}(\mathbb{A})} \leftarrow \Lambda_{a_{i,0}^{(\ell)}}(\mathbb{A})$, $\sum_{i=1}^{n} \sum_{k=1}^{N} d_{\mathbb{A},i,k}^{(\ell)} \theta_{i,k} = 0$

LoR	$\mathbf{c}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1, -j)$	$\psi_i^{(j_i)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{F}}$
LoR	$\widetilde{\mathbf{t}}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1, -j)$	$\nu_i^{(j_i)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{G}}$
Enc	$\mathbf{c}_{i,j}^{(j_i)}$	(	$\sigma'_{i,j} \cdot (1, -j)$	$\overline{\psi}_i^{(j_i)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{F}}$
	$\mathbf{k}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j, 1)$	$a_{i,j}^{(\ell)} \cdot z^{(\ell)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{F}^*}$
	$\widetilde{\mathbf{m}}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j, 1)$	$a_{i,j}^{(\ell)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{G}^*}$

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(b,j_i)}$	$\nu_i^{(j_i)}$	$\mathbf{0}$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
Enc	$\mathbf{t}_i^{(j_i)}$	(	$\chi \cdot \mathbf{s}_i + \chi' \cdot \mathbf{u}_i + \overline{\mathbf{x}}_i^{(j_i)}$	$\nu_i^{(j_i)}$	$\mathbf{0}$	0	$\overline{\rho}_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^{n} a_{i,0}^{(\ell)}$	$\mathbf{0}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i \omega_{\mathsf{tag}}$	$p_i \omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\mathbf{0}$	$(\theta_{i,k})_{k=1}^{N}$	0	$)_{\mathbf{H}_i}$
Enc	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i \chi_{\mathsf{tag}'}$	$p_i \chi'_{\mathsf{tag}'}$	$\overline{\psi}_i^{(j_i)}$	$\mathbf{0}$	$(\theta_{i,k})_{k=1}^{N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i}$
	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$\mathbf{0}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^{N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game $\mathsf{G}_1 : z_j \xleftarrow{\$} \mathbb{Z}_q^*$, $\Delta \mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ (Masking Application - Lemma 3.5, hybrids over each key query $(\mathbf{y}_i^{(\ell)})_i$, using the DPVS basis changes from Section 3.3, *i.e.* formal ones (1, 2, 3) and computational ones (1, 2))

$\boxed{\mathsf{G}_{0.\ell.1}}$ where $\ell \in [K]$ and K is the maximum number of key queries. We are in the setting of *private-input only* repetitions $a'_{i,0} \xleftarrow{\$} \mathbb{Z}_q$, $(a'_{i,j})_{j \in \mathsf{List\text{-}Att}(\mathbb{A})} \leftarrow \Lambda_{a'_{i,0}}(\mathbb{A})$

LoR	$\mathbf{c}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1, -j)$	$\psi_i^{(j_i)}$	$\mathbf{0}$	0	$\boxed{z_j \cdot \Delta \mathbf{x}_i}$	0	0	$)_{\mathbf{F}}$
	$\mathbf{k}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j, 1)$	$a_{i,j}^{(\ell)} \cdot z^{(\ell)}$	$\mathbf{0}$	0	$\boxed{(a'_{i,j}/z_j) \cdot \mathbf{y}_i^{(\ell)}}$	0	0	$)_{\mathbf{F}^*}$

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(b,j_i)}$	$\nu_i^{(j_i)}$	$\mathbf{0}$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^{n} a_{i,0}^{(\ell)}$	$\mathbf{0}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i \omega_{\mathsf{tag}}$	$p_i \omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\boxed{\Delta \mathbf{x}_i}$	$(\theta_{i,k})_{k=1}^{N}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$\boxed{a'_{i,0} \cdot \mathbf{y}_i^{(\ell)}}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^{N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

$\boxed{\mathsf{G}_{0.\ell.2}}$: $R \xleftarrow{\$} \mathbb{Z}_q$, $\Delta \mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$(Randomization, the honest $\mathcal{H}$ and corrupted $\mathcal{C}$ are known due to *static* corruption, use *formal basis changes*)

LoR	$\mathbf{c}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1, -j)$	$\psi_i^{(j_i)}$	$\mathbf{0}$	0	$\boxed{z_j \cdot \Delta \mathbf{x}_i}$	0	0	$)_{\mathbf{F}}$
	$\mathbf{k}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j, 1)$	$a_{i,j}^{(\ell)} \cdot z^{(\ell)}$	$\mathbf{0}$	0	$\boxed{(a'_{i,j}/z_j) \cdot \mathbf{y}_i^{(\ell)}}$	0	0	$)_{\mathbf{F}^*}$

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i \omega_{\mathsf{tag}}$	$p_i \omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\Delta \mathbf{x}_i$	$(\theta_{i,k})_{k=1}^{N}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$\boxed{(a'_{i,0} + R) \cdot \mathbf{y}_i^{(\ell)}}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^{N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

$\boxed{\mathsf{G}_{0.\ell.3}}$: $R \xleftarrow{\$} \mathbb{Z}_q$, $\Delta \mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$(Reverse Masking Application - Lemma 3.5, only mask $R \cdot \mathbf{y}_i^{(\ell)}$ remains in $\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$ for $i \in \mathcal{H}$)

LoR	$\mathbf{c}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1, -j)$	$\psi_i^{(j_i)}$	$\mathbf{0}$	0	$\boxed{\mathbf{0}}$	0	0	$)_{\mathbf{F}}$
LoR	$\widetilde{\mathbf{t}}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1, -j)$	$\nu_i^{(j_i)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{G}}$
	$\mathbf{k}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j, 1)$	$a_{i,j}^{(\ell)} \cdot z^{(\ell)}$	$\mathbf{0}$	0	$\boxed{\mathbf{0}}$	0	0	$)_{\mathbf{F}^*}$
	$\widetilde{\mathbf{m}}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j, 1)$	$a_{i,j}^{(\ell)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{G}^*}$

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i \omega_{\mathsf{tag}}$	$p_i \omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\Delta \mathbf{x}_i$	$(\theta_{i,k})_{k=1}^{N}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$\boxed{R \cdot \mathbf{y}_i^{(\ell)}}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^{N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

$\mathsf{G}_1 \coloneqq \mathsf{G}_{0.K.3}$, where $\ell \in [K]$ and K is the maximum number of key queries.

Figure 6.3: Games $\mathsf{G}_0, \mathsf{G}_1$ for Theorem 6.11.

at position i, being constants at each i over all repetitions j_i, under the admissibility.

Game G_2 : $R \xleftarrow{\$} \mathbb{Z}_q$, $\Delta\mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ (Rewriting game's description, summing of $\widetilde{\mathbf{t}}_{i,j}^{(j_i)}$ over $i \in \mathcal{H}$ known statically, not affecting correctness)

LoR	$\mathbf{c}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1,-j)$	$\psi_i^{(j_i)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{F}}$
LoR	$\widetilde{\mathbf{t}}_{0,j}^{(j_i)} = \sum_{i\in\mathcal{H}} \widetilde{\mathbf{t}}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1,-j)$	$\sum_{i\in\mathcal{H}} \nu_i^{(j_i)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{G}}$
	$\mathbf{k}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j,1)$	$a_{i,j}^{(\ell)} \cdot z^{(\ell)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{F}^*}$
	$\widetilde{\mathbf{m}}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j,1)$	$a_{i,j}^{(\ell)}$	$\mathbf{0}$	0	$\mathbf{0}$	0	0	$)_{\mathbf{G}^*}$

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(b,j_i)}$	$\nu_i^{(j_i)}$	$\mathbf{0}$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^n a_{i,0}^{(\ell)}$	$\mathbf{0}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i\omega_{\mathsf{tag}}$	$p_i\omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\Delta\mathbf{x}_i$	$(\theta_{i,k})_{k=1}^N$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$R \cdot \mathbf{y}_i^{(\ell)}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game G_3 : $R \xleftarrow{\$} \mathbb{Z}_q$, $\Delta\mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ (Masking Application - Lemma 3.5, hybrids over each key query $(\mathbf{y}_i^{(\ell)})_i$, similar to $\mathsf{G}_1 \to \mathsf{G}_2$)

LoR	$\widetilde{\mathbf{t}}_{0,j}^{(j_i)} = \sum_{i\in\mathcal{H}} \widetilde{\mathbf{t}}_{i,j}^{(j_i)}$	(	$\sigma_{i,j}^{(j_i)} \cdot (1,-j)$	$\sum_{i\in\mathcal{H}} \nu_i^{(j_i)}$	$\mathbf{0}$	0	$\boxed{\mathbf{0}}$	0	0	$)_{\mathbf{G}}$
	$\widetilde{\mathbf{m}}_{i,j}^{(\ell)}$	(	$\pi_{i,j}^{(\ell)} \cdot (j,1)$	$a_{i,j}^{(\ell)}$	$\mathbf{0}$	0	$\boxed{\mathbf{0}}$	0	0	$)_{\mathbf{G}^*}$

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(b,j_i)}$	$\nu_i^{(j_i)}$	$\boxed{\Delta\mathbf{x}_i}$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^n a_{i,0}^{(\ell)}$	$\boxed{R \cdot \mathbf{y}_i^{(\ell)}}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i\omega_{\mathsf{tag}}$	$p_i\omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\Delta\mathbf{x}_i$	$(\theta_{i,k})_{k=1}^N$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$R \cdot \mathbf{y}_i^{(\ell)}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Figure 6.4: Games $\mathsf{G}_2, \mathsf{G}_3$ for Theorem 6.11.

Moreover, the *strong admissibility* also ensures that:

$$\begin{cases} \sum_{i\in\mathcal{H}} \langle \Delta\mathbf{x}_i, \mathbf{y}_i^{(\ell)} \rangle = 0 \\ \langle \Delta\mathbf{x}_i, \mathbf{y}_i^{(\ell)} \rangle = 0 \; \forall \, i \in \mathcal{C} \end{cases}$$

corresponding to any inner product function of $(\mathbf{y}_i)_i$ (together with an LSSS). The ℓ-th key components are programmed to also accomodate newly independent values: $a'_{i,0} \xleftarrow{\$} \mathbb{Z}_q$, $(a'_{i,j})_{j\in\mathsf{List\text{-}Att}(\mathbb{A})} \leftarrow \Lambda_{a'_{i,0}}(\mathbb{A}), z_j \xleftarrow{\$} \mathbb{Z}_q^*$. We emphasize that the random values introduced in key components are randomized secret shares $(a'_{i,j}/z_j) \cdot \mathbf{y}_i$ in which $a'_{i,j}$ are shares of $a'_{i,0}$ by the attributes in $\mathsf{List\text{-}Att}(\mathbb{A})$. Thus, over all honest $i \in \mathcal{H}$, due to the simulated vectors at decryption will cancel the masks, while for corrupted $i \in \mathcal{C}$, the masks are already 0 after performing the product between the i-th ciphertext component and the i-th key component

$$\langle z_j \Delta\mathbf{x}_i, a'_{i,j}/z_j \mathbf{y}_i^{(\ell)} \rangle = \langle \Delta\mathbf{x}_i, a'_{i,j} \mathbf{y}_i^{(\ell)} \rangle = a'_{i,j} \langle \Delta\mathbf{x}_i, \mathbf{y}_i^{(\ell)} \rangle = 0 \ .$$

Moreover, because we are dealing with *vectors* $\Delta\mathbf{x}_i, \mathbf{y}_i^{(\ell)}$, the Lemma 3.5 is applied by components, which is possible due to the appropriate dimension of $\mathbf{c}$-components and $\mathbf{k}$-components, as well as the proof of the Lemma 3.5 itself (see [NPP25, Appendix B]).

$\boxed{\mathsf{G}_{0.\ell.2}}$ We randomize the values $a'_{i,0}$ in the key components by adding a independent fixed random mask R. First of all, we remark that for $i \in \mathcal{H}$ where $\mathbb{A}(\mathsf{S}_i) = 0$, where $\mathbb{A}$ is the LSSS associated to the ℓ-th key query, the change is even perfectly indistinguishable. This is because of the facts that

- the randomized shares $a'_{i,j}/z_j$ are uniformly random and independent thanks to z_j,
- even with *repetitions* at a position i, for a challenge tag tag, the shares z_j are independent for different repetitions given the *private-only* repetitions.

Game G_4 : $R, R' \xleftarrow{\$} \mathbb{Z}_q$, $\Delta\mathbf{x}_i \leftarrow \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}$ (Switching $\mathbf{x}_i^{(b,j_i)}$ to $\mathbf{x}_i^{(1,j_i)}$, using *complexity leveraging*, the invariant coordinates are grouped as "$\cdots$")

$\boxed{\mathsf{G}_{3.1}}$ (Formal Quotient, using $\Delta\mathbf{x}_i$ is constant for $i \in \mathcal{H}$ over repetitions, Hadamard product is denoted "$\circ$", see example 2 on DPVS basis changes)

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(b,j_i)}$	$\nu_i^{(j_i)}$	$\Delta\mathbf{x}_i$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^n a_{i,0}^{(\ell)}$	$R \cdot \mathbf{y}_i^{(\ell)}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$
LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$\cdots$	$\boxed{r\mathbf{1}_{\Delta\mathbf{x}_i}}$	$\boxed{r'\mathbf{1}}$	0		$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\cdots$	$\boxed{R \cdot (\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)})}$	$\boxed{(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$		$)_{\mathbf{H}_i^*}$

$\boxed{\mathsf{G}_{3.2}}$ (Switching, updating secret shares of 0, Hadamard product is denoted "$\circ$", see example 3 on DPVS basis changes)

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \boxed{\mathbf{x}_i^{(1,j_i)}}$	$\nu_i^{(j_i)}$	$\Delta\mathbf{x}_i$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^n a_{i,0}^{(\ell)}$	$\boxed{R' \cdot \mathbf{y}_i^{(\ell)}}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$
LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$\cdots$	$r\mathbf{1}_{\Delta\mathbf{x}_i}$	$(r'-r)\mathbf{1}$	0		$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\cdots$	$\boxed{R' \cdot (\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)})}$	$\boxed{(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$		$)_{\mathbf{H}_i^*}$

$\boxed{\mathsf{G}_4 \coloneqq \mathsf{G}_{3.3}}$ (Formal Quotient, using $\Delta\mathbf{x}_i$ is constant for $i \in \mathcal{H}$ over repetitions, see example 2 on DPVS basis changes)

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(1,j_i)}$	$\nu_i^{(j_i)}$	$\Delta\mathbf{x}_i$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i$	$\sum_{i=1}^n a_{i,0}^{(\ell)}$	$R' \cdot \mathbf{y}_i^{(\ell)}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$
LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i\omega_{\mathsf{tag}}$	$p_i\omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\boxed{\Delta\mathbf{x}_i}$	$\boxed{\theta_{i,k}}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z$	$\boxed{R' \cdot \mathbf{y}_i^{(\ell)}}$	$\boxed{(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game G_5 : (Cleaning)

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \mathbf{x}_i^{(1,j_i)}$	$\nu_i^{(j_i)}$	$\boxed{\mathbf{0}}$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^n a_{i,0}^{(\ell)}$	$\boxed{\mathbf{0}}$	$\mathsf{rnd}^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$
LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i\omega_{\mathsf{tag}}$	$p_i\omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\boxed{\mathbf{0}}$	$\boxed{\theta'_{i,k}}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$\boxed{\mathbf{0}}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Figure 6.5: Games $\mathsf{G}_4, \mathsf{G}_5$ for Theorem 6.11.

- more importantly, when $\mathbb{A}(\mathsf{S}_i) = 0$, it holds that z_j *never* appears in any of the ciphertexts returned to the adversary
- as a consequence, the shares $a'_{i,j}/z_j$ is information theoretically hidden and making $a'_{i,0}$ information theoretically hidden for the adversary.

In the end, in this case for $i \in \mathcal{H}$ where $\mathbb{A}(\mathsf{S}_i) = 0$, what we do is just rewriting an information theoretically hidden value $a'_{i,0}$ to another information theoretically hidden value $a'_{i,0} + R$, and this change goes perfectly indistinguishable. However, there can be the case where some $i \in \mathcal{H}$ it holds $\mathbb{A}(\mathsf{S}_i) = 1$. This case can be treated by formal basis changes together with a *complexity leveraging* argument. We detail below, the details of calculation for DPVS basis changes are recalled and can be revised in Section 3.3.

The main idea is to consider the *selective* version $\mathsf{G}^*_{0.\ell.1.t}$ for $t \in \{1,2,3,4\}$, where the values $(\mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(\ell)})_{i\in[n]}^{j_i \in [J_i]}$ are guessed in advance. We then use formal argument for the transitions $\mathsf{G}^*_{0.\ell.1.1} \to \mathsf{G}^*_{0.\ell.1.4}$ to obtain for $j \in [3]$,

$$\Pr[\mathsf{G}^*_{0.\ell.j} = 1] = \Pr[\mathsf{G}^*_{0.\ell.j+1} = 1] \ . \tag{6.7}$$

In the end, we use a *complexity leveraging* argument to conclude that thanks to (6.7), we have $\Pr[\mathsf{G}_{0.\ell.1} = \mathsf{G}_{0.\ell.1.1} = 1] = \Pr[\mathsf{G}_{0.\ell.2} = \mathsf{G}_{0.\ell.1.4} = 1]$.

For the sequence $\mathsf{G}_{0.\ell.1.1} \to \mathsf{G}_{0.\ell.1.4}$, we make a guess for the values $(\mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(\ell)})_{i\in[n]}^{j_i\in[J_i]}$, choose $R \xleftarrow{\$} \mathbb{Z}_q^*$, random secret sharings $(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$ of 0 where $\theta_{i,k} \neq 0$. We define the event E that the guess is correct on $(\mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(\ell)})_{i\in[n]}^{j_i\in[J_i]}$ and for all $k \in [N]$

$$\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)} = -R \cdot \Delta\mathbf{x}_i[k]\mathbf{y}_i^{(\ell)}[k] \ . \tag{6.8}$$

We describe the *selective* games below, starting from $\mathsf{G}_{0.\ell.1}^* = \mathsf{G}_{0.\ell.1.1}^*$, where event E is assumed true:

Game $\mathsf{G}_{0.\ell.1}^* = \mathsf{G}_{0.\ell.1.1}^*$: The vectors have form:

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i\omega_{\mathsf{tag}}$	$p_i\omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\Delta\mathbf{x}_i$	$(\theta_{i,k})_{k=1}^N$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$a'_{i,0} \cdot \mathbf{y}_i^{(\ell)}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game $\mathsf{G}_{0.\ell.1.2}^*$: We perform a formal basis change to the key components, for $i \in \mathcal{H}$ to change $(\mathbf{H}_i, \mathbf{H}_i^*)$ following matrices: for $r, r' \xleftarrow{\$} \mathbb{Z}_q^*$,

$$H_i[row, col] = \begin{cases} 1 & \text{if } row = col \leq 3 \\ \frac{r}{\Delta\mathbf{x}_i[z]} & \text{if } \exists z \in [N] \text{ s.t. } row = col = 3+z \ \wedge \ \Delta\mathbf{x}_i[z] \neq 0 \\ 1 & \text{if } \exists z \in [N] \text{ s.t. } row = col = 3+z \ \wedge \ \Delta\mathbf{x}_i[z] = 0 \\ \frac{r'}{\theta_{i,z}} & \text{if } \exists z \in [N] \text{ s.t. } row = col = N+3+z \\ 0 & \text{otherwise} \end{cases}; H'_i := \left(H_i^{-1}\right)^\top \ .$$

We remark that the matrix does not have to check non-zeroness of $\theta_{i,z}$, as it is guaranteed by the event E. The vectors have form: we denote the Hadamard product by "$\circ$", and $\mathbf{1}_{\Delta\mathbf{x}_i}$ is the vector of 1's at the positions where $\Delta\mathbf{x}_i$ is non-zero

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$\cdots$	$r \cdot \boxed{\mathbf{1}_{\Delta\mathbf{x}_i}}$	$r' \cdot \boxed{\mathbf{1}}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\cdots$	$a'_{i,0} \cdot \left(\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)}\right)$	$(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game $\mathsf{G}_{0.\ell.1.3}^*$: We perform a formal basis change to the key components, for $i \in \mathcal{H}$ to change $(\mathbf{H}_i, \mathbf{H}_i^*)$ following matrices: for $r, r' \xleftarrow{\$} \mathbb{Z}_q^*$, (for ease of presenting basis changes we write the transposed matrix $H_i^\top$)

$$H_i^\top[row, col] = \begin{cases} 1 & \text{if } row = col \notin \{4+N, \ldots, 3+2N\} \\ 1 & \text{if } row = col \in \{4+N, \ldots, 3+2N\} \ \wedge \ \Delta\mathbf{x}_i[row - N - 3] \neq 0 \\ \frac{r'}{r'+r} & \text{if } row = col \in \{4+N, \ldots, 3+2N\} \ \wedge \ \Delta\mathbf{x}_i[row - N - 3] = 0 \\ -1 & \text{if } \exists z \in [N] \text{ s.t. } row = 3+N+z \ \wedge \ col = 3+z \\ 0 & \text{otherwise} \end{cases}; \quad H'_i := \left(H_i^{-1}\right)^\top \ .$$

We note that on the diagonal $\tilde{z} := row = col \in \{4+N, 3+2N\} \ \wedge \ \Delta\mathbf{x}_i[\tilde{z} - N - 3] = 0$, because coordinate $\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}[\tilde{z} - N] = 0$ as $\Delta\mathbf{x}_i[\tilde{z} - N - 3] = 0$, the moving by $H_i^\top[3+N+\tilde{z}-N-3, 3+\tilde{z}-N-3]$ has no effect on $\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}[\tilde{z}]$. Thus $H_i^{-1}[row, col]$ needs to multiply a factor $(r'+r)/r'$[7] to the coordinate $\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}[\tilde{z}]$ to make sure that after the basis change it becomes $r'+r$. Dually the coordinate $\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}[\tilde{z}] = \theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)}$ stays correctly thanks to the relation (6.8) and we pay attention that $\Delta\mathbf{x}_i[\tilde{z} - N - 3] = 0$. The vectors have form: we denote the Hadamard product by "$\circ$"

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$\cdots$	$r \cdot \mathbf{1}_{\Delta\mathbf{x}_i}$	$\boxed{(r+r') \cdot \mathbf{1}}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\cdots$	$\boxed{(a'_{i,0} + R) \cdot (\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)})}$	$(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

using the hypothesis that event E happens along with the relation (6.8) specifically. Consequently, we just update one secret share of 0 by another. The randomness r' is updated to $r'+r$, indentically distributed.

[7] Therefore the corresponding position on the diagonal of $H_i^\top[\tilde{z}, \tilde{z}] = \frac{r'}{r'+r}$.

Game* $\mathsf{G}^*_{0.\ell.1.4}$: We undo the formal basis changes $\mathsf{G}^*_{0.\ell.1.1} \to \mathsf{G}^*_{0.\ell.1.2}$, where the division by $1/r, 1/(r+r')$ can be done with overwhelming probability since $r, r' \xleftarrow{\$} \mathbb{Z}_q^*$ at the beginning of the game to define the matrices. This gives

$$\begin{array}{llllllll} \mathbf{LoR} & \mathbf{c}^{(j_i)}_{i,\mathsf{ipfe}} & (& p_i\omega_{\mathsf{tag}} & p_i\omega'_{\mathsf{tag}} & \psi_i^{(j_i)} & \Delta\mathbf{x}_i & (\theta_{i,k})_{k=1}^N & 0 &)_{\mathbf{H}_i} \\ i \in \mathcal{H} & \mathbf{k}^{(\ell)}_{i,\mathsf{ipfe}} & (& \sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle & \sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle & a^{(\ell)}_{i,0} z^{(\ell)} & \boxed{(a'_{i,0}+R)\cdot \mathbf{y}_i^{(\ell)}} & (d^{(\ell)}_{\mathbb{A},i,k})_{k=1}^N & \mathsf{rnd}^{(\ell)}_{i,\mathsf{ipfe}} &)_{\mathbf{H}^*_i} \end{array}$$

The above games demonstrate relation (6.7). We now employ the complexity leveraging argument. Let us fix $j \in \{1,2,3\}$. For $u \in \{0.\ell.1.j, 0.\ell.1.j+1\}$ let $\mathsf{Adv}_u(\mathcal{A}) := |\Pr[\mathsf{G}_u(\mathcal{A}) = 1] - 1/2|$ denote the advantage of a ppt adversary $\mathcal{A}$ in game G_u. We build a ppt adversary $\mathcal{B}^*$ playing against G^*_u such that its advantage $\mathsf{Adv}^*_u(\mathcal{B}^*) := |\Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1] - 1/2|$ equals $\gamma \cdot \mathsf{Adv}_u(\mathcal{A})$ for $u \in \{t, t+1\}$, for some constant γ.

The adversary $\mathcal{B}^*$ first guesses the values $(\mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(\ell)})_{i\in[n]}^{j_i\in[J_i]}$, choose $R \xleftarrow{\$} \mathbb{Z}_q^*$, random secret sharings $(\theta_{i,k} \cdot d^{(\ell)}_{\mathbb{A},i,k})_{k=1}^N$ of 0. Then $\mathcal{B}^*$ defines the event E that:

> the guess is correct on $(\mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(\ell)})_{i\in[n]}^{j_i\in[J_i]}$ and for all $k \in [N]$, $\theta_{i,k} \cdot d^{(\ell)}_{\mathbb{A},i,k} = -R \cdot \Delta\mathbf{x}_i[k]\mathbf{y}_i^{(\ell)}[k]$.

When $\mathcal{B}^*$ guesses successfully and E happens, then the simulation of $\mathcal{A}$'s view in G_t is perfect. Otherwise, $\mathcal{B}^*$ aborts the simulation and outputs a random bit b'. Since E happens with some fixed probability γ and is independent from the view of $\mathcal{A}$, we have[8]:

$$\begin{aligned} \mathsf{Adv}^*_u(\mathcal{B}^*) &= \left|\Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1] - \frac{1}{2}\right| \\ &= \left|\Pr[E] \cdot \Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1 \mid E] + \frac{\Pr[\neg E]}{2} - \frac{1}{2}\right| \\ &= \left|\gamma \cdot \Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1 \mid E] + \frac{1-\gamma-1}{2}\right| \\ &\overset{(*)}{=} \gamma \cdot \left|\Pr[\mathsf{G}_u(\mathcal{A}) = 1] - \frac{1}{2}\right| = \gamma \cdot \mathsf{Adv}_u(\mathcal{A}) \end{aligned} \tag{6.9}$$

where $(*)$ comes from the fact that conditioned on E, $\mathcal{B}$ simulates perfectly G_u for $\mathcal{A}$, therefore $\Pr[\mathsf{G}_u(\mathcal{A}) = 1 \mid E] = \Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1 \mid E]$, then we apply the independence between E and $\mathsf{G}_u(\mathcal{A}) = 1$. Together with relation (6.7), this concludes that $\Pr[\mathsf{G}_{0.\ell.1.j} = 1] = \Pr[\mathsf{G}_{0.\ell.1.j+1} = 1]$ for any fixed $j \in \{1,2,3\}$, in particular $\Pr[\mathsf{G}_{0.\ell.1} = \mathsf{G}_{0.\ell.1.1} = 1] = \Pr[\mathsf{G}_{0.\ell.2} = \mathsf{G}_{0.\ell.1.4} = 1]$.

Union bounds on $\mathbb{A}(\mathsf{S}_i) = 0$ (perfect indistinguishability by information-theoretic argument on z_j and $a'_{i,j}/z_j$) and $\mathbb{A}(\mathsf{S}_i) = 1$ (perfect indistinguishability by complexity leveraging) give the conclusion that the game hop is perfectly indistinguishable.

$\boxed{\mathsf{G}_{0.\ell.3}}$ Reverse Masking Application - Lemma 3.5, so that only mask $R\mathbf{y}_i$ remains for $i \in \mathcal{H}$, in $\mathbf{k}^{(\ell)}_{i,\mathsf{ipfe}}$. Once again, the mask R will be canceled by the admissibility condition:

$$\sum_{i\in\mathcal{H}} \langle \mathbf{x}_i^{(b,j_i)} - \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i^{(\ell)} \rangle = 0 \ .$$

We arrive at G_1 after $\mathsf{G}_{0.K.3}$.

Game G_2: We rewrite the game's description to program the vectors $\widetilde{\mathbf{t}}^{(j_i)}_{0,j} = \sum_{i\in\mathcal{H}} \widetilde{\mathbf{t}}^{(j_i)}_{i,j}$. The goal is to consider $\widetilde{\mathbf{t}}^{(j_i)}_{0,j}$ in the subsequent games, *i.e.* we look at the vectors $\widetilde{\mathbf{t}}^{(j_i)}_{0,j}$ instead of the given $\widetilde{\mathbf{t}}^{(j_i)}_{i,j}$ returned to the adversary. The rewriting is totally formal as it follows exactly what is described in Figure 6.2.

[8] This calculation (6.9) to relate $\mathsf{Adv}^*_u(\mathcal{B}^*)$ to $\mathsf{Adv}_u(\mathcal{A})$ is the core of our complexity levaraging argument, being built upon the previous information-theoretic game transtions and the probability of event E.

Game G_3: We apply similarly Lemma 3.5 as in $\mathsf{G}_1 \to \mathsf{G}_2$, by a sequence of hybrids over the ℓ-th functional key, one after another. We remark that the random factor $R \xleftarrow{\$} \mathbb{Z}_q$ is the same as that one introduced in $\mathsf{G}_1 \to \mathsf{G}_2$, this simplifies one guess during the complexity leveraging argument. The formal basis changes resembles those in $\mathsf{G}_1 \to \mathsf{G}_2$ and in the end, the game hop is perfectly indistinguishable.

Game G_4: We use a *complexity leveraging* argument, that depends only on *formal basis changes*. The goal is to switch from $\mathbf{x}_i^{(b,j_i)}$ to $\mathbf{x}_i^{(1,j_i)}$ for $i \in \mathcal{H}$. The details of the *selective* underlying games are given in Figure 6.5. First of all, we make a guess for the values $(\mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(\ell)})_{i\in[n]}^{j_i\in[J_i]}$, choose $R \xleftarrow{\$} \mathbb{Z}_q^*$, random secret sharings $(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$ of 0 where $\theta_{i,k} \neq 0$. We define the event F that

$$\text{the guess is correct on } (\mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(\ell)})_{i\in[n]}^{j_i\in[J_i]} \text{ and for all } k \in [N]$$
$$\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)} = -\Delta\mathbf{x}_i[k]\mathbf{y}_i^{(\ell)}[k] \ , \tag{6.10}$$

so as to make sure $\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)}$ is a secret sharing of 0 conditioned on F. We give the matrices' definitions as follows to demonstrate how the calculation is performed:

Game $\mathsf{G}_{3.1}^* = \mathsf{G}_3^*$: The vectors have form:

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i\omega_{\mathsf{tag}}$	$p_i\omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\Delta\mathbf{x}_i$	$(\theta_{i,k})_{k=1}^N$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$R \cdot \mathbf{y}_i^{(\ell)}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game $\mathsf{G}_{3.2}^*$: We perform a formal basis change to the key components, for $i \in \mathcal{H}$ to change $(\mathbf{H}_i, \mathbf{H}_i^*)$ following matrices: for $r, r' \xleftarrow{\$} \mathbb{Z}_q^*$,

$$H_i[row, col] = \begin{cases} 1 & \text{if } row = col \leq 3 \\ \frac{r}{\Delta\mathbf{x}_i[z]} & \text{if } \exists z \in [N] \text{ s.t. } row = col = 3 + z \ \wedge \ \Delta\mathbf{x}_i[z] \neq 0 \\ 1 & \text{if } \exists z \in [N] \text{ s.t. } row = col = 3 + z \ \wedge \ \Delta\mathbf{x}_i[z] = 0 \\ \frac{r'}{\theta_{i,z}} & \text{if } \exists z \in [N] \text{ s.t. } row = col = N + 3 + z \\ 1 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t. } row = col = N + 3 + z \\ 0 & \text{otherwise} \end{cases} ; H_i' := \left(H_i^{-1}\right)^\top \ .$$

We remark that the matrix does not have to check non-zeroness of $\theta_{i,z}$, as it is guaranteed by the event F. The vectors have form: we denote the Hadamard product by "$\circ$", and $\mathbf{1}_{\Delta\mathbf{x}_i}$ is the vector of 1's at the positions where $\Delta\mathbf{x}_i$ is non-zero

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$\cdots$	$r \cdot \boxed{\mathbf{1}_{\Delta\mathbf{x}_i}}$	$r' \cdot \boxed{1}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\cdots$	$\boxed{R \cdot \left(\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)}\right)}$	$\boxed{(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game $\mathsf{G}_{3.3}^*$: We perform a formal basis change to the key components, for $i \in \mathcal{H}$ to change $(\mathbf{B}_i, \mathbf{B}_i^*)$, $(\mathbf{H}_i, \mathbf{H}_i^*)$ following matrices: for $r, r' \xleftarrow{\$} \mathbb{Z}_q^*$, (for ease of presenting basis changes we write the transposed matrix $H_i^\top$ and B_i^{-1})

$$H_i^\top[row, col] = \begin{cases} 1 & \text{if } row = col \notin \{4 + n, \ldots, 3 + 2N\} \\ 1 & \text{if } row = col \in \{4 + n, \ldots, 3 + 2N\} \ \wedge \ \Delta\mathbf{x}_i[row - N - 3] \neq 0 \\ \frac{r'}{r'+r} & \text{if } row = col \in \{4 + n, \ldots, 3 + 2N\} \ \wedge \ \Delta\mathbf{x}_i[row - N - 3] = 0 \\ -1 & \text{if } \exists z \in [N] \text{ s.t.} \\ & \quad row = 3 + z \ \wedge \ col = 3 + N + z \\ 0 & \text{otherwise} \end{cases} ; \ H_i' := \left(H_i^{-1}\right)^\top$$

$$B_i^{-1}[row, col] = \begin{cases} 1 & \text{if } row = col \\ -1 & \text{if } \exists z \in [N] \text{ s.t.} \\ & \quad row = 1 + N + z \ \wedge \ col = z \\ 0 & \text{otherwise} \end{cases} ; \ B_i' := \left(B_i^{-1}\right)^\top \ .$$

Following the matrices

- The formal changes of $(\mathbf{B}_i, \mathbf{B}_i^*)$ switch $\mathbf{x}_i^{(b,j_i)}$ to $\mathbf{x}_i^{(1,j_i)}$ for $i \in \mathcal{H}$, where for $z \in [N]$ under B_i^{-1}, the coordinate $\mathbf{t}_i^{(j_i)}[z]$ is updated to

$$\mathbf{t}_i^{(j_i)}[z] - \Delta\mathbf{x}_i[z] = \omega \cdot \mathbf{s}_i[z] + \omega' \cdot \mathbf{u}_i[z] + \mathbf{x}_i^{(b,j_i)}[z] + \mathbf{x}_i^{(1,j_i)}[z] - \mathbf{x}_i^{(b,j_i)}[z] = \omega \cdot \mathbf{s}_i[z] + \omega' \cdot \mathbf{u}_i[z] + \mathbf{x}_i^{(1,j_i)}[z] \ .$$

  While dually in $\mathbf{m}_i^{(\ell)}[1+N+z]$ the matrix $B_i^\top$ introduces $R'\mathbf{y}^{(\ell)} := (R+1) \cdot \mathbf{y}_i^{(\ell)}$ staying regroupable with the corresponding $\Delta\mathbf{x}_i$ in $\mathbf{t}_i^{(j_i)}[1+N+z]$.
- The changes of $(\mathbf{H}_i, \mathbf{H}_i^*)$ are also to correct R to R' in the key components, thanks to (6.10) of the games that we recall under this selective sequence, so that the decryption's correctness is preserved. We note that the diagonal of $H_i^\top$ also takes care of the case where $\Delta\mathbf{x}_i[z] = 0$ for $z \in [N]$, in the same manner as we have done for $\mathsf{G}^*_{0.\ell.1.2} \to \mathsf{G}^*_{0.\ell.1.3}$ previously.

The vectors have form: we denote the Hadamard product by "$\circ$"

LoR	$\mathbf{t}_i^{(j_i)}$	(	$\omega \cdot \mathbf{s}_i + \omega' \cdot \mathbf{u}_i + \boxed{\mathbf{x}_i^{(1,j_i)}}$	$\nu_i^{(j_i)}$	$\Delta\mathbf{x}_i$	0	$\rho_i^{(j_i)}$	$)_{\mathbf{B}_i}$
	$\mathbf{m}_i^{(\ell)}$	(	$\mathbf{y}_i^{(\ell)}$	$\sum_{i=1}^n a_{i,0}^{(\ell)}$	$\boxed{R' \cdot \mathbf{y}_i^{(\ell)}}$	$\mathsf{rnd}_i^{(\ell)}$	0	$)_{\mathbf{B}_i^*}$

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$\cdots$	$r \cdot \mathbf{1}_{\Delta\mathbf{x}_i}$	$\boxed{(r'+r) \cdot \mathbf{1}}$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\cdots$	$\boxed{R' \cdot \left(\Delta\mathbf{x}_i \circ \mathbf{y}_i^{(\ell)}\right)}$	$\boxed{(\theta_{i,k} \cdot d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N}$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

using the hypothesis that event F happens along with the relation (6.10) specifically. Consequently, we just update one secret share of 0 by another. The randomness r' is updated to $r' + r$, indentically distributed.

Game* $\mathsf{G}^*_{3.4}$: We undo the formal basis changes $\mathsf{G}^*_{3.1} \to \mathsf{G}^*_{3.2}$ and obtain

LoR	$\mathbf{c}_{i,\mathsf{ipfe}}^{(j_i)}$	(	$p_i\omega_{\mathsf{tag}}$	$p_i\omega'_{\mathsf{tag}}$	$\psi_i^{(j_i)}$	$\Delta\mathbf{x}_i$	$(\theta_{i,k})_{k=1}^N$	0	$)_{\mathbf{H}_i}$
$i \in \mathcal{H}$	$\mathbf{k}_{i,\mathsf{ipfe}}^{(\ell)}$	(	$\sum_i \langle \mathbf{s}_i, \mathbf{y}_i^{(\ell)} \rangle$	$\sum_i \langle \mathbf{u}_i, \mathbf{y}_i^{(\ell)} \rangle$	$a_{i,0}^{(\ell)} z^{(\ell)}$	$\boxed{R' \cdot \mathbf{y}_i^{(\ell)}}$	$(d_{\mathbb{A},i,k}^{(\ell)})_{k=1}^N$	$\mathsf{rnd}_{i,\mathsf{ipfe}}^{(\ell)}$	$)_{\mathbf{H}_i^*}$

Game G_5: We clean the masks so that the adversary's view is independent of the challenge b.

The bit b does not appear in the responses to the adversary anymore, completing the proof.

6.4.3 Upgrading Security

Next, we can apply a layer of All-or-Nothing Encapsulation (AoNE) so as to remove the tradeoff with respect to *incomplete* challenge ciphertexts (*i.e.* remove *pos*-condition in Definition 6.9, under the private-input repetitions). That is, the adversary can now omit some honest slot $i \in \mathcal{H}$ with respect to $(\mathsf{tag}, \mathsf{S}_i)$ for different S_i, up to repetition $(\mathbf{x}_i^{(0,j_i)}, \mathbf{x}_i^{(1,j_i)})$. More specifically, we apply the generic transformation from [NPS22, Lemma 16], to treat the case of MCFE with access control as a special case in the above lemma so as to remove *pos*-condition, given the private-input repetitions. The formal statement is stated below.

Lemma 6.12 (Incomplete Security with Private-Only Repetitions). *Assume there exist (1) a one-challenge MCFE scheme $\mathcal{E}^{\mathsf{pos}}$ for the function class $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ that is secure against* complete *queries,* i.e. *satisfying* pos*-security and (2) an AoNE scheme $\mathcal{E}^{\mathsf{aone}}$ whose message space contains the ciphertext space of $\mathcal{E}^{\mathsf{pos}}$. Then there exists a one-challenge MCFE scheme $\mathcal{E}$ for the same function class $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$ that is even secure against* incomplete *queries. More precisely, for any* ppt *adversary $\mathcal{A}$, there exist* ppt *algorithms $\mathcal{B}_1$ and $\mathcal{B}_2$ such that*

$$\mathsf{Adv}^{\mathsf{mc\text{-}w\text{-}rep\text{-}xxx\text{-}1chal\text{-}cpa}}_{\mathcal{E},\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{(N_i)_{i=1}^n},q,\mathsf{LSSS},\mathcal{A}}(1^\lambda) \leq 12 \cdot \left(\mathsf{Adv}^{\mathsf{mc\text{-}w\text{-}rep\text{-}pos\text{-}xxx\text{-}1chal\text{-}cpa}}_{\mathcal{E}^{\mathsf{pos}},\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{(N_i)_{i=1}^n},q,\mathsf{LSSS},\mathcal{B}_1}(1^\lambda) + \mathsf{Adv}^{\mathsf{mc\text{-}w\text{-}rep\text{-}xxx\text{-}1chal\text{-}cpa}}_{\mathcal{E}^{\mathsf{aone}},\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{(N_i)_{i=1}^n},q,\mathsf{LSSS},\mathcal{B}_2}(1^\lambda) \right)$$

where $\mathsf{xxx} \subseteq \{\mathsf{stat}, \mathsf{sel}\}$.

We refer to the proof of the more general lemma in [NPS22, Lemma 16], with repetitions on the private inputs $\mathbf{x}_i$. Finally, by combining with Lemma 6.5 to allow multiple challenge tags, where the only restriction remains: solely for *private* inputs, and not *public* attributes per client, will repetitions be allowed. We have the following Corollary:

Corollary 6.13. *We consider the bilinear group setting* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ *and the functionality is* $\mathcal{F}^{\mathsf{IP}}_{\mathsf{subvec},B} \times \mathsf{LSSS}$. *Then, there exists a multi-client IPFE scheme with fine-grained access control via LSSS that is* statically IND-secure *in the ROM, against* multiple incomplete *challenge queries with* repetitions on private inputs*, under the* SXDH *assumption in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$.

Remark 6.14. (Towards MIFE for inner products) Corollary 6.13 presents an MCFE for subvectors with fine-grained access control so that its security adapted to the case of subvectors (see Definition 6.4), with *multiple (with possible repetitions on private inputs)* , under a given challenge tag and against *incomplete* queries. We can obtain an MIFE for *inner products* in the standard model by fixing one tag for every ciphertext, *i.e.* the random oracle can be removed by publishing a random fixed value corresponding to $\mathsf{H}(\mathsf{tag})$ for encryption. The security of the resulted MIFE is implied from the security of our MCFE in Corollary 6.13 thanks to the fact that the adversary can make multiple challenge queries to **LoR** for each slot $i \in [n]$, following the admissiblity in Definition 6.3. In particular, security with possible repetitions on private inputs of the MCFE implies security of the obtained MIFE when repetitive private $\mathbf{x}_i$ are used for the same i. In particular, we obtain an MIFE for inner-products with adaptive security in the standard model, whose keys can be control by LSSS restraining no repetitions on attributes per client.

Allowing Repetitions on Attributes. As mentioned at the beginning of this section, the above Lemma 6.12 deals with *incomplete* challenge queries, but only with respect to the *private* input $\mathbf{x}_i$ of each client i. It cannot lift our restriction that we do *not* allow repetitions on the public attributes S_i. This explains why this constraint persists in our final MCFE from Corollary 6.13. The fact that AoNE cannot deal with repetitions on public attributes is also mentioned in recent works [ATY23a] and we leave it as potential extension to remove this constraint.

Chapter

7 Decentralized Multi-Client Functional Encryption with Strong Security

Chapter content

From the previous chapters of this thesis, our main cryptographic object of interests is MCFE. Chapter 5 however makes progress in the domain of a more general notion, namely *Decentralized Multi-Client Functional Encryption* (DMCFE). In particular, the security model of DMCFE, which encompasses the weaker case of MCFE, is vigorously studied in Chapter 5. We revisit DMCFE in this chapter, this time the security model is also the main subject but in a somewhat different flavor - it looks at the secrecy of the functions in functional decryption keys.

In general, DMCFE extends the basic functional encryption to multiple clients that do not trust each other. They can independently encrypt the multiple plaintext-inputs to be given for evaluation to the function embedded in the functional decryption key, defined by multiple parameter-inputs. And they keep control on these functions as they all have to contribute to the generation of the functional decryption keys. Tags can be used in the ciphertexts and the keys to specify which inputs can be combined together. As any encryption scheme, DMCFE provides privacy of the plaintexts. But the functions associated to the functional decryption keys might be sensitive too (*e.g.* a model in machine learning). The function-hiding property has thus been introduced to additionally protect the function evaluated during the decryption process.

The results that are presented in this chapter provide new proof techniques to analyze a new concrete construction of function-hiding DMCFE for inner products, with strong security guarantees in the random oracle model: the adversary can adaptively query multiple challenge ciphertexts and multiple challenge keys, with unbounded repetitions of the same message tags in the ciphertext-queries and a fixed polynomially-large number of repetitions of the same key tags in the key-queries, allowing static corruption of the secret encryption keys. Previous constructions were proven secure in the selective setting only.

7.1 Introduction

Decentralized Multi-Client Functional Encryption. The setup of MCFE requires some authority (a trusted third party) responsible for the setup and generation of functional decryption keys.

The authority possesses a master secret key msk that can be used to handle the distribution of private encryption keys ek_i and deriving functional decryption keys dk_F. When clients do not trust each other, this centralized setting of authority might be a disadvantage. The need for such a central authority is completely eliminated in the so-called *Decentralized Multi-Client Functional Encryption* (DMCFE) introduced by Chotard *et al.* [CDG+18a]. In DMCFE, only during the setup phase do we need interaction for generating parameters that will be needed by the clients later. The key generation is done independently by different *senders*, each has a *secret key* sk_i. Agreeing on a function F, each sender generates their functional key $\mathsf{dk}_{F,i}$ using sk_i, the description of F, and a tag $\mathsf{tag\text{-}f}$. Originally in [CDG+18a], the tag $\mathsf{tag\text{-}f}$ can contain the description of F itself. Using DMCFE, the need of an authority for distributing functional keys is completely removed, with minimal interaction required during setup. The seminal work of [CDG+18a] constructed the first DMCFE for computing inner products (IP-DMCFE), where n clients can independently contribute to the ciphertext vector $(\mathsf{ct}_1 \leftarrow \mathsf{Enc}(\mathsf{ek}_1, \mathsf{tag}, x_1), \ldots, \mathsf{ct}_n \leftarrow \mathsf{Enc}(\mathsf{ek}_n, \mathsf{tag}, x_n))$ and n senders can independently contribute to the functional keys $\mathsf{dk}_{\mathbf{y},1} \leftarrow \mathsf{DKeyGen}(\mathsf{sk}_1, \mathsf{tag\text{-}f}, y_1), \ldots, \mathsf{dk}_{\mathbf{y},n} \leftarrow \mathsf{DKeyGen}(\mathsf{sk}_n, \mathsf{tag\text{-}f}, y_n)$ of some vector $\mathbf{y} = (y_1, \ldots, y_n)$. For the function class to compute inner products, many follow-up works improve upon the work of [CDG+18a] on both aspects of efficiency as well as security, or by giving generic transformation to (D)MCFE from single-client FE [LT19, ABKW19, ABG19].

Repetitions under One Tag. Involving tags at the time of encryption and key generation restricts that only ciphertexts and functional keys having the same tag can be combined in the notion of DMCFE. This raises a natural question: what security can we guarantee when one client uses the same tag on multiple data ? We call such multiple usages of the same tag in a DMCFE system *repetitions*. In the formal security model of (D)MCFE in [CDG+18a] and subsequent works [LT19], once the adversary makes a query for (i, tag), further queries for the same pair (i, tag) will be ignored. This means repetitions are not taken into account. The authors of [CDG+18a] argued that it is the responsibility of the users not to use the same tag twice. However, a security notion for DMCFE that captures a sense of protection even when repetitions mistakenly/maliciously happen will be preferable, *e.g.* this is indeed studied in some other works [ABKW19, ABG19]. In addition, when repetitions are allowed for ciphertexts, the security model of MCFE strictly encompasses MIFE by replacing tags with a constant value, as confirmed in recent works [ATY23a].

Function Privacy in FE. Standard security notions of FE ensure that adversaries do not learn anything about the content of ciphertexts beyond what is revealed by the functions for which they possess decryption keys. However, it is *not* required that functional decryption keys hide the function they decrypt. In practice, this can pose a serious problem because the function itself could contain confidential data. For example, the evaluated function may represent a neural network. Training such networks is often time-consuming and expensive, which is why companies offer their use as a paid service. However, to ensure that customers continue to pay for the use of the product, it is crucial that the concrete parameters of the network (*i.e.* the computed function) remain secret. This additional security requirement for functional encryption schemes is known as the *function-hiding* property. As another example, suppose one wants to perform statistical analysis (*e.g.* weighted averages) of private data from several companies to get a better understanding of the dynamics of a sector. This can be implemented using a DMCFE for inner products. Consulting firms conduct such analyses as a fee-based service. To ensure that clients continue to pay for updated results in the future, the consulting firm may wish to hide the concrete parameters of their calculations. This can be achieved by using a DMCFE with function-hiding security.

Besides practical applications, function-hiding FE schemes for restricted function classes (such as inner products) have also proven to be an important technical building block for the construction of FE schemes for broader function classes: Lin [Lin17] employed a function-hiding

IPFE (FH-IPFE) to obtain an FE scheme for quadratic functions. A different technique was also introduced by Gay in [Gay20] equally aiming at constructing FE for quadratic functions. With several technical novelties, Agrawal *et al.* [AGT21a, AGT22] were able to generalize the aforementioned constructions to obtain MIFE for quadratic functions. Comparisons with existing works can be revisited in Table 2.3 in Section 2.3.4.

Chapter Outline. We start by giving a high-level overview of the technical constructions in Section 7.2. The overview is constituted by a simplified *selective* version of our final *adaptive* FH-DMCFE. At the end of Section 7.2 the main challenges towards adaptive security will be highlighted. More preliminaries are then recalled in Section 7.3. Our principal technical details can be found in Section 7.4, in which a variant of swapping lemma is given in Section 7.4.1. The foregoing lemma plays a crucial role for proving security our basic FH-DMCFE in Section 7.4.2. All abridged proofs and other details can be found in the full version [NPS24b] of [NPS24a].

7.2 Technical Overview: A Simpler Selective Case

In this section of high-level overview, we gives a simpler DMCFE construction with *function-hiding* security for computing inner products *as per* Definition 7.1. The function-hiding security holds against *selective* attacks in which the adversary submits up front: *(i)* all challenge ciphertext queries $(\mathbf{x}_i^{(0,j)}, \mathbf{x}_i^{(1,j)})$, where the index j denotes the j-th repetition at slot i given the challenge tag^* that is queried to the encryption oracle $\mathbf{Enc}(i, \mathsf{tag}^*, \cdot, \cdot)$, and *(ii)* all challenge key queries $(\mathbf{y}_i^{(0,\tilde{j})}, \mathbf{y}_i^{(1,\tilde{j})})$, where the index $\tilde{j}$ dentoes the $\tilde{j}$-th repeition at slot i given the challenge $\mathsf{tag\text{-}f}^*$ that is queried to the key-generation oracle $\mathbf{DKeyGen}(i, \mathsf{tag\text{-}f}^*, \cdot, \cdot)$. Other tags $\mathsf{tag}_\ell \neq \mathsf{tag}^*$ with respect to the j-th query to **Enc** for slot i is denoted by $\mathbf{x}_{\ell,i}^{(j)}$. In the same manner, we denote the $\tilde{j}$-th query to **DKeyGen** for slot i and tag $\mathsf{tag\text{-}f}_k \neq \mathsf{tag\text{-}f}^*$ by $\mathbf{y}_{k,i}^{(\tilde{j})}$. We also restrict the setting to *one-challenge*: there exists a unique challenge tag^* that is queried to the encryption oracle $\mathbf{Enc}(i, \mathsf{tag}^*, \cdot, \cdot)$, and there exists a unique $\mathsf{tag\text{-}f}^*$ that is queried to the key-generation oracle $\mathbf{DKeyGen}(i, \mathsf{tag\text{-}f}^*, \cdot, \cdot)$. Finally, we also restrict the corruption to be *static* and on both $(\mathsf{ek}_i, \mathsf{sk}_i)$ at the time of corruption[1], that is, the set $\mathcal{C}$ of corrupted i is known before the adversary receives answers to all of their queries. The contents of this section is three-fold: we first present the **Selective One-time FH-DMCFE**, then **A One-time Selective Security Proof** for the foregoing FH-DMCFE, and finally **Some Challenges towards Adaptive Security**.

A One-time Selectively Secure Function-Hiding DMCFE. We use a prime-order bilinear group setting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and write $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}$ additively. The number n of senders and of clients is fixed in advance. Given $(\mathsf{tag\text{-}f}, \mathbf{y}_i)$, a sender i generates a partial functional key dk_i. On the client side, given $(\mathsf{tag}, \mathbf{x}_i)$ a client i generates a ciphertext ct_i. An ensemble $(\mathsf{dk}_i)_{i\in[n]}$ with respect to the *same* $\mathsf{tag\text{-}f}$ will be able to decrypt an ensemble $(\mathsf{ct}_i)_{i\in[n]}$ with respect to the *same* tag. The result evaluation is $\sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i \rangle$. In a nutshell, the main ideas for our simpler selective FH-DMCFE are presented below:

- Each client encryption for ct_i is done via *randomizing* a predetermined share $\tilde{t}_i$ of some secret sharing $(\tilde{t}_i)_{i\in[n]}$ of 0. The share $\tilde{t}_i$ is given to the client i in their encryption key ek_i, along with other secret basis information of DPVS. The randomization can be achieved by hashing $\mathsf{H}_1(\mathsf{tag}) \to [\![\mu]\!]_1$.
- Each decentralized key generation for each dk_i can be done simply by hashing $\mathsf{H}_2(\mathsf{tag\text{-}f}) \to [\![\omega]\!]_2$ and masking $\mathbf{y}_i$ in their corresponding dk_i.
- Using the DPVS setting, it is design that when decrypting, the sum

$$\sum_{i=1}^{n} \mathsf{ct}_i \times \mathsf{dk}_i$$

  will contains $\sum_{i=1}^n [\![\mu\omega\tilde{t}_i]\!]_\mathsf{t} = [\![0]\!]_\mathsf{t}$.

We summarize the construction as follows:

$\mathsf{Setup}(1^\lambda)$: Sample $(\tilde{t}_i)_{i\in[n]} \xleftarrow{\$} \mathbb{Z}_q^n$ such that $\sum_i \tilde{t}_i = 0$; generate n pair of DPVS bases $\{(\mathbf{B}_i, \mathbf{B}_i^*)\}_{i\in[n]}$ with respect to $(B_i, \left(B_i^{-1}\right)^\top)$ for some $B_i \in GL_N(\mathbb{Z}_q)$; define $B_i' := \left(B_i^{-1}\right)^\top$ and output $\mathsf{sk}_i = B_i'$ and $\mathsf{ek}_i = (\tilde{t}_i, B_i)$ for $i \in [n]$.

$\mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, \mathbf{y}_i)$: Compute $[\![\mu]\!]_2 = \mathsf{H}_2(\mathsf{tag\text{-}f})$; output $\mathsf{dk}_i = (\mathbf{y}_i, \mu, \mathbf{0})_{\mathbf{B}_i^*}$.

[1]We recall that in Chapter 5 a corruption model where ek_i and sk_i can be corrupted separately is examined. As we elaborate in the concrete admissibility for inner products (**Remark** 5.14, **Theorem** 5.13), for example, this can lead to differences in the admissible condition regarding a corrupted i, *i.e.* it depends on whether corrupting i reveals sk_i or not.

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, \mathbf{x}_i)$: Compute $[\![\omega]\!]_1 = \mathsf{H}_1(\mathsf{tag})$; output $\mathsf{ct}_i = (\mathbf{x}_i, \omega\tilde{t}_i, \mathbf{0})_{\mathbf{B}_i}$.

$\mathsf{Dec}(\{(\mathsf{dk}_i, \mathsf{ct}_i)\}_{i\in[n]})$: Compute the DPVS products $\mathsf{ct}_i \times \mathsf{dk}_i$ for all pairs $(\mathsf{dk}_i, \mathsf{ct}_i)$ to recover $[\![z_i]\!]_\mathsf{t} = [\![\langle \mathbf{x}_i, \mathbf{y}_i\rangle + \mu\omega\tilde{t}_i]\!]_\mathsf{t}$ and find discrete log of $[\![z]\!]_\mathsf{t} = \left[\!\left[\sum_{i\in[n]} z_i\right]\!\right]_\mathsf{t}$.

Proof of One-time Selective Security. As a reminder, we consider *static* corruptions (see Item 1 of Definition 7.2). Additionally, we allow only one challenge tags tag^* for ciphertexts and $\mathsf{tag\text{-}f}^*$ for keys, against complete queries (see Items 3 and 4). We remark that this is sufficient as in Section 7.4.3, we show how to remove both restrictions from the security model via a sequence of generic conversions.

We begin by writing down the view of the adversary, over all received ciphertexts and keys that are indexed by repetitions (*e.g.* the adversary can query multiple times at slot i for ciphertexts under a challenge tag tag^*, and the same is allowed for challenge key queries):

$$\begin{aligned} \mathbf{c}_{\ell,i}^{(j)} &= (\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i} := \omega_\ell \cdot \tilde{t}_i, 0, \mathbf{0}, 0, 0)_{\mathbf{B}_i} & \mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(b,j)}, t_i := \omega \cdot \tilde{t}_i, 0, \mathbf{0}, 0, 0)_{\mathbf{B}_i} \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= (\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 0, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} & \mathbf{d}_i^{(\tilde{j})} &= (\mathbf{y}_i^{(b,\tilde{j})}, \mu, 0, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} \end{aligned} \tag{7.1}$$

Since we are in the *static* corruption setting, it is declared in advance the set $\mathcal{C}$ of corrupted i whose $(\mathsf{ek}_i, \mathsf{sk}_i)$ are known by the adversary. The admissibility condition from Item 1 of Definition 7.2 requires that the challenge ciphertexts $(\mathbf{c}_i^{(j)})_{i\in\mathcal{C}}$ as well as the challenge keys $(\mathbf{d}_i^{(\tilde{j})})_{i\in\mathcal{C}}$ to be independent from b. Therefore, in the following we can concentrate on those honest $i \in \mathcal{H}$.

Introduction of fresh secret shares. We start by randomizing the values t_i and $t_{\ell,i}$ for honest clients $i \in \mathcal{H}$ while relying on the DDH assumption in $\mathbb{G}_1$. Then, fresh secret sharings $(\tau_i)_{i\in\mathcal{H}}$ and $(\tau_{\ell,i})_{i\in\mathcal{H}}$ of 0 are embedded in the ciphertexts for $i \in \mathcal{H}$ as follows:

$$\begin{aligned} \mathbf{c}_{\ell,i}^{(j)} &= (\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \boxed{\tau_{\ell,i}}, \mathbf{0}, 0, 0)_{\mathbf{B}_i} & \mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(b,j)}, t_i, \boxed{\tau_i}, \mathbf{0}, 0, 0)_{\mathbf{B}_i} \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= (\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, \boxed{1}, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} & \mathbf{d}_i^{(\tilde{j})} &= (\mathbf{y}_i^{(b,\tilde{j})}, \mu, \boxed{1}, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} \end{aligned} \tag{7.2}$$

We need the newly introduced secret shares $(\tau_i)_{i\in\mathcal{H}}$ and $(\tau_{\ell,i})_{i\in\mathcal{H}}$ of 0 to be indexed accoridngly by the indices of tags tag^* and tag_ℓ for later steps. Thus, this foregoing change is done by a sequence of hybrids over the ordering of the tags tag^* and tag_ℓ, under the DDH assumption in $\mathbb{G}_2$. In short, each hybrid makes use of the shares $t_{\ell,i}, t_i$ and the random self-reducibility of DDH to insert new random secret shares, then DPVS allows us manipulating these new values and putting the vectors in desired forms. Two additional coordinates that contain 0 at the end of each vector permit performing basis changes in DPVS. Later in the proof for our $\mathsf{FH\text{-}DMCFE}$ based on DPVS, introducing the new random shares $\tau_i, \tau_{\ell,i}$ is taken care by Lemma 7.7, using particularly the DPVS basis changes and DDH. We thus do not write the random share introduction explicitly in the $\mathsf{FH\text{-}DMCFE}$ proof and refer to the transitions $\mathsf{G}_0 \to \mathsf{G}_1$ in the proof of Lemma 7.7 for more details.

Preparations for swapping: Copies of $\mathbf{x}$*-vectors.* We now introduce the vectors $\mathbf{x}_i^{(1)}$ and $\mathbf{x}_{\ell,i}$ in the additional 0-coordinates of the ciphertexts of honest clients $i \in \mathcal{H}$.

$$\begin{aligned} \mathbf{c}_{\ell,i}^{(j)} &= (\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \tau_{\ell,i}, \boxed{\mathbf{x}_{\ell,i}^{(j)}}, 0, 0)_{\mathbf{B}_i} & \mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(b,j)}, t_i, \tau_i, \boxed{\mathbf{x}_i^{(1,j)}}, 0, 0)_{\mathbf{B}_i} \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= (\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 1, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} & \mathbf{d}_i^{(\tilde{j})} &= (\mathbf{y}_i^{(b,\tilde{j})}, \mu, 1, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} \end{aligned} \tag{7.3}$$

The change goes indistinguishable under the DDH assumption in $\mathbb{G}_1$, by using a *Subspace* DPVS basis change (see example 1 in Section 3.3). One delicacy is that we are putting fixed values into the vectors, so we use (multiplicatively up to a constant loss factor) the equivalent variant DSDH of DDH (Definition 3.2).

Preparations for swapping: Modifying secret shares under admissibility. For this step, we lean on the admissibility conditions (items 1 and 2 of Definition 7.2) state for all $j_i, \tilde{j}_i$ that

$$\sum_{i\in[n]} \langle \mathbf{x}_i^{(0,j_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)} \rangle = \sum_{i\in[n]} \langle \mathbf{x}_i^{(1,j_i)}, \mathbf{y}_i^{(1,\tilde{j}_i)} \rangle \qquad \text{and} \qquad \sum_{i\in[n]} \langle \mathbf{x}_{\ell,i}^{(j_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)} \rangle = \sum_{i\in[n]} \langle \mathbf{x}_{\ell,i}^{(j_i)}, \mathbf{y}_i^{(1,\tilde{j}_i)} \rangle$$

as well as $\mathbf{x}_i^{(0,j)} = \mathbf{x}_i^{(1,j)}$ and $\mathbf{y}_i^{(0,\tilde{j})} = \mathbf{y}_i^{(1,\tilde{j})}$ if $i \in \mathcal{C}$. From this, it follows for $b \in \{0,1\}$[2] that

$$\Delta_i^{(b)} := \langle \mathbf{x}_i^{(b,j)}, \mathbf{y}_i^{(b,\tilde{j})} \rangle - \langle \mathbf{x}_i^{(1,j)}, \mathbf{y}_i^{(1,\tilde{j})} \rangle \qquad \text{and} \qquad \Delta_{\ell,i}^{(b)} := \langle \mathbf{x}_{\ell,i}^{(j)}, \mathbf{y}_i^{(b,\tilde{j})} - \mathbf{y}_i^{(1,\tilde{j})} \rangle$$

are constant for all repetitions $j, \tilde{j}$, and $\Delta_i^{(b)} = \Delta_{\ell,i}^{(b)} = 0$ if $i \in \mathcal{C}$. Furthermore, we have that $\sum_{i\in\mathcal{H}} \Delta_i^{(b)} = \sum_{i\in\mathcal{H}} \Delta_{\ell,i}^{(b)} = 0$. Together, these conditions imply that the distributions

$$D_0 = \left\{ (\tau_i)_{i\in\mathcal{H}} : (\tau_i)_{i\in\mathcal{H}} \xleftarrow{\$} \mathbb{Z}_q^{|\mathcal{H}|} \text{ s.t. } \sum_{i\in\mathcal{H}} \tau_i = 0 \right\}$$

$$D_1 = \left\{ (\tau_i)_{i\in\mathcal{H}} : (\tau'_i)_{i\in\mathcal{H}} \xleftarrow{\$} \mathbb{Z}_q^{|\mathcal{H}|} \text{ s.t. } \sum_{i\in\mathcal{H}} \tau_i = 0, \tau_i := \tau'_i - \Delta_i^{(b)} \right\}$$

are identical (and a similar result also holds for all $(\tau_{\ell,i})_{i\in\mathcal{H}}$). Thus, it is an information-theoretic change to provide the adversary with

$$\begin{aligned} \mathbf{c}_{\ell,i}^{(j)} &= (\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \boxed{\tau_{\ell,i} - \Delta_{\ell,i}^{(b)}}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0)_{\mathbf{B}_i} & \mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(b,j)}, t_i, \boxed{\tau_i - \Delta_i^{(b)}}, \mathbf{x}_i^{(1,j)}, 0, 0)_{\mathbf{B}_i} \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= (\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 1, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} & \mathbf{d}_i^{(\tilde{j})} &= (\mathbf{y}_i^{(b,\tilde{j})}, \mu, 1, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} \end{aligned} \tag{7.4}$$

where the values $\Delta_{\ell,i}^{(b)}, \Delta_i^{(b)}$ are furthermore independent from the repetition indices.

Swapping. We recall that we are in a hybrid over each $\mathsf{tag\text{-}f}_k, \mathsf{tag\text{-}f}^*$. Without loss of generality we suppose the current hybrid is changing the challenge key vectors $\mathbf{d}_i^{(\tilde{j})}$. After all above preparations, it is now the Lemma 7.7 coming into play. Specifically, Lemma 7.7 helps us moving $\mathbf{y}_i^{(b,\tilde{j})}$ from the first coordinates to other coordinates on the right, changing to $\mathbf{y}_i^{(1,\tilde{j})}$ while facing $\mathbf{x}_i^{(1,j)}$, in $\mathbf{d}_i^{(\tilde{j})}$:

$$\begin{aligned} \mathbf{c}_{\ell,i}^{(j)} &= (\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \boxed{\tau_{\ell,i}}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0)_{\mathbf{B}_i} & \mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(b,j)}, t_i, \boxed{\tau_i}, \mathbf{x}_i^{(1,j)}, 0, 0)_{\mathbf{B}_i} \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= (\mathbf{y}_{k,i}^{(\tilde{j})}, \mu_k, 1, \mathbf{0}, 0, 0)_{\mathbf{B}_i^*} & \mathbf{d}_i^{(\tilde{j})} &= (\boxed{\mathbf{0}}, \mu, 1, \boxed{\mathbf{y}_i^{(1,\tilde{j})}}, 0, 0)_{\mathbf{B}_i^*} \end{aligned} \tag{7.5}$$

Thanks from the step of *Modifying secret shares under admissibility*, the difference

$$\Delta_i^{(b)} = \langle \mathbf{x}_i^{(b,j)}, \mathbf{y}_i^{(b,\tilde{j})} \rangle - \langle \mathbf{x}_i^{(1,j)}, \mathbf{y}_i^{(1,\tilde{j})} \rangle$$

as well as $\Delta_{\ell,i}^{(b)} = \langle \mathbf{x}_{\ell,i}^{(j)}, \mathbf{y}_i^{(b,\tilde{j})} - \mathbf{y}_i^{(1,\tilde{j})} \rangle$ already exists in $\mathbf{c}_i^{(j)}$. It is of utmost importance that theses differences are in place before the swapping, because moving and changing $\mathbf{y}_i^{(b,\tilde{j})}$ to $\mathbf{y}_i^{(1,\tilde{j})}$ while facing $\mathbf{x}_i^{(1,j)}$, will induce the same differences. These differences have to be canceled to preserve the DPVS product $\mathbf{c}_i^{(j)} \times \mathbf{d}_i^{(\tilde{j})}$ and Lemma 7.7 can be applied[3].

When applying similar arguments as in the steps from (7.3) to (7.5) in a hybrid over $\mathbf{d}_{k,i}^{(\tilde{j})}$ for all k, we finally arrive at:

$$\begin{aligned} \mathbf{c}_{\ell,i}^{(j)} &= (\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \boxed{\tau_{\ell,i}}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0)_{\mathbf{B}_i} & \mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(b,j)}, t_i, \boxed{\tau_i}, \mathbf{x}_i^{(1,j)}, 0, 0)_{\mathbf{B}_i} \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= (\boxed{\mathbf{0}}, \mu_k, 1, \boxed{\mathbf{y}_{k,i}^{(\tilde{j})}}, 0, 0)_{\mathbf{B}_i^*} & \mathbf{d}_i^{(\tilde{j})} &= (\mathbf{0}, \mu, 1, \mathbf{y}_i^{(1,\tilde{j})}, 0, 0)_{\mathbf{B}_i^*} \end{aligned} \tag{7.6}$$

[2] More precisely, the case $b = 0$ follows from the admissibility condition while for $b = 1$, we always have $\Delta_i^{(b)} = \Delta_{\ell,i}^{(b)} = 0$

[3] Some steps of this overview are already taken care in Lemma 7.7, because the lemma is designed in a modular way for treatments of the more complex adaptive security. For instance, the step of **Introduction of fresh secret shares** is one of the first steps in Lemma 7.7. Hence, in this simpler overview of selective security, by "apply Lemma 7.7" we mean an application of the lemma modulo these prepatory steps.

At this point, we can remove the vectors $\mathbf{x}_i^{(b,j)}$ in $\mathbf{c}_i^{(j)}$ which gives us a game that is independent of the bit b. So the adversary's advantage is 0 and the proof is finished.

$$\begin{aligned} \mathbf{c}_{\ell,i}^{(j)} &= (\mathbf{x}_{\ell,i}^{(j)}, t_{\ell,i}, \tau_{\ell,i}, \mathbf{x}_{\ell,i}^{(j)}, 0, 0)_{\mathbf{B}_i} & \mathbf{c}_i^{(j)} &= (\boxed{\mathbf{0}}, t_i, \tau_i, \mathbf{x}_i^{(1,j)}, 0, 0)_{\mathbf{B}_i} \\ \mathbf{d}_{k,i}^{(\tilde{j})} &= (\mathbf{0}, \mu_k, 1, \mathbf{y}_{k,i}^{(\tilde{j})}, 0, 0)_{\mathbf{B}_i^*} & \mathbf{d}_i^{(\tilde{j})} &= (\mathbf{0}, \mu, 1, \mathbf{y}_i^{(1,\tilde{j})}, 0, 0)_{\mathbf{B}_i^*} \end{aligned} \tag{7.7}$$

Problems for Adaptive Security. In the above proof for selective security, we highlight the important role of embedding the differences $\Delta_i^{(b)} = \langle \mathbf{x}_i^{(b,j)}, \mathbf{y}_i^{(b,\tilde{j})} \rangle - \langle \mathbf{x}_i^{(1,j)}, \mathbf{y}_i^{(1,\tilde{j})} \rangle$ as well as $\Delta_{\ell,i}^{(b)} = \langle \mathbf{x}_{\ell,i}^{(j)}, \mathbf{y}_i^{(b,\tilde{j})} - \mathbf{y}_i^{(1,\tilde{j})} \rangle$ into the fresh secret shares of $\mathbf{c}_i^{(j)}$ before the final swapping. The values $\Delta_i^{(b)}$ and $\Delta_{\ell,i}^{(b)}$ depend on both the challenge ciphertext queries, encryption queries, and challenge key queries of the adversary. If we allow the adaversary to make queries *adaptively*, $\Delta_i^{(b)}$ and $\Delta_{\ell,i}^{(b)}$ can be unknown at the time of responding $\mathbf{c}_i^{(j)}$.

In our main FH-DMCFE construction, in order to deal with this situation to achieve adaptive security, the information-theoretic properties of DPVSes is used. This is already crucial in our MCFE construction for inner products of sub-vectors with LSSS access control in Section 6.4.2. As a reminder, under an sequence of *identically distributed* hybrids thanks to DPVS, we can allow the guessing of $\Delta_i^{(b)}, \Delta_{\ell,i}^{(b)}$ and continue the simulation only when the guesses are correct (and the step *Modifying secret shares under admissibility* can be carried out). Because between two successive hybrids the advantage is 0 for any ppt adversary to distinguish, such guesses that incur a multiplicative loss factor still keeps the advantage 0. Last but not least, as we mention in Footnote 3, Lemma 7.7 is designed so that the *Swapping* step can be performed in the adaptive setting. This is notably reflected via the lemma's statement, in which an adversary's views are indistingushable given adaptive access to oracles. Those oracles correspond to the execution of key-generation and ciphertext oracles of the FH-DMCFE security experiment (see Figure 7.1), for challenge and non-challenge queries. Each time Lemma 7.7 is applied, *e.g.* the transition $\mathsf{G}_2 \to \mathsf{G}_3$ in [NPS24b, Appendix A.3] and its details in the proof, we verify the hypothesis of the lemma and list the FH-DMCFE security's oracles outputs in the order of the lemma's oracles to affect the correct vectors. A discussion of our adaptively secure FH-DMCFE is given in Section 7.4.2.

7.3 More Preliminaries

The function family $\mathcal{F}_n^{\mathsf{ip}}$ of bounded-norm inner-product functionalities with n inputs is defined as follows.

Definition 7.1 (Inner Product Functionality). *For $n, \lambda \in \mathbb{N}$, let $\mathcal{D}_\lambda = \mathsf{Param}_\lambda = [-B; B]^N$ and $\mathcal{R}_\lambda = [-nNB^2; nNB^2]$, where $B = B(\lambda)$ and $N = N(\lambda) \colon \mathbb{N} \to \mathbb{N}$ are polynomials. We define the* inner-product functionality *$\mathcal{F}^{\mathsf{ip}} = \{\mathcal{F}^{\mathsf{ip}}_{n,\lambda}\}_{n,\lambda\in\mathbb{N}}$ for $\mathcal{F}^{\mathsf{ip}}_{n,\lambda} = \{f_{n,\lambda,(\mathbf{y}_1,\ldots,\mathbf{y}_n)} \colon \mathcal{D}_\lambda^n \to \mathcal{R}_\lambda\}_{(\mathbf{y}_1,\ldots,\mathbf{y}_n)\in\mathsf{Param}_\lambda^n}$ as the family of functions*

$$f_{n,\lambda,(\mathbf{y}_1,\ldots,\mathbf{y}_n)}(\mathbf{x}_1,\ldots,\mathbf{x}_n) = \textstyle\sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i\rangle \ .$$

Since its first introduction [CDG+18a], and as it is used so far also since Chapter 5, a DMCFE consists of (Setup, DKeyGenShare, DKeyComb, Enc, Dec). The purpose of DKeyComb was motivated for efficiency reasons in [CDG+18a]. Without lost of generality, in this chapter we do not explicate DKeyComb and assume that the key combination is done at the beginning of Dec. That is, for a given function class $\mathcal{F}$, we write $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ as a DMCFE scheme for $\mathcal{F}$.

Function-Hiding Security for DMCFE. We define function-hiding, along with standard security for DMCFE. In the seminal work by Chotard *et al.* [CDG+18a] and its follow-up study [CDSG+20], the security notion does not cover the function-hiding requirement for DMCFE or its more general sibling DDFE. Until recently, the work by Agrawal *et al.* [AGT21b] abstracted out DMCFE into the notion of *Multi-Party Functional Encryption* (MPFE). Or equivalently, [Ngu24] abstracted out DMCFE into the notion of *Dynamic Decentralized Functional Encryption* (DDFE). The authors of [AGT21b] also used MPFE to spell out the function-hiding security for MCFE as well as for DDFE. The latter does capture DMCFE as a particular case but for convenience of the reader, we introduce the detailed function-hiding security for DMCFE, without going through all the abstraction of MPFE nor of DDFE. Our security definition follows the *Game-Playing Framework* in [BR06]: Figure 7.1 defines the experiment $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ with procedures **Initialize**, **DKeyGen**, **Enc**, **Corrupt** and **Finalize**; the adversary $\mathcal{A}$ runs **Initialize**, can call the oracles in any order and any number of times, and finishes the run by calling **Finalize** on input the guess b'.

Definition 7.2 (Function-Hiding Security). *Let $\lambda \in \mathbb{N}$ be a security parameter. For a DMCFE scheme $\mathcal{E}$, a function class $\mathcal{F} = \{\mathcal{F}_{n,\lambda}\}_{n,\lambda}$ and a* ppt *adversary $\mathcal{A}$ we define the experiment $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ as shown in Figure 7.1 and set $\mathcal{H} := [n] \setminus \mathcal{C}$. The oracles* **Enc**, **DKeyGen** *and* **Corrupt** *can be called in any order and any number of times. The adversary $\mathcal{A}$ is* NOT admissible *with respect to $\mathcal{C}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{Q}_{\mathsf{KGen}}$, denoted by $\mathsf{adm}(\mathcal{A}) = 0$, if either one of the following holds:*

1. *There exists a tuple $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ or $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ such that $i \in \mathcal{C}$ and $x_i^{(0)} \neq x_i^{(1)}$[4] or $y_i^{(0)} \neq y_i^{(1)}$.*

2. *There exist $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, two vectors $(x_i^{(0)})_{i\in[n]}, (x_i^{(1)})_{i\in[n]} \in \mathcal{D}_1 \times \cdots \times \mathcal{D}_n$ and functions $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}, f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})} \in \mathcal{F}$ having parameters $(y_i^{(0)}, y_i^{(1)})_{i\in[n]}$ such that*

[4] This admissibility condition on $x_i^{(0)} = x_i^{(1)}$ for all $i \in \mathcal{C}$ was introduced in [CDG+18a] then used in all other works on (D)MCFE [CDG+18a, LT19, ABKW19, ABG19] and later on DDFE [CDSG+20, AGT21b]. A recent work [NPP23a] studies the relaxation that removes this condition for (D)MCFE, *i.e.* allowing $x_i^{(0)} \neq x_i^{(1)}$ for $i \in \mathcal{C}$ and more attacks are considered admissible, and gives a provably secure DMCFE candidate computing inner products. We are not aware of any DMCFE scheme in the literature which is proven secure under the stronger notion from [NPP23a].

Initialize$(1^\lambda, 1^n)$:
$\mathcal{C}, \mathcal{Q}_{\mathsf{Enc}}, \mathcal{Q}_{\mathsf{KGen}} \leftarrow \varnothing$; $b \xleftarrow{\$} \{0,1\}$
$(\mathsf{pp}, (\mathsf{sk}_i)_{i\in[n]}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$
Return pp

DKeyGen$(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)})$:
$\mathcal{Q}_{\mathsf{KGen}} \leftarrow \mathcal{Q}_{\mathsf{KGen}} \cup \{(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)})\}$
Return $\mathsf{dk}_{f,i} \leftarrow \mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, y_i^{(b)})$

Enc$(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)})$:
$\mathcal{Q}_{\mathsf{Enc}} \leftarrow \mathcal{Q}_{\mathsf{Enc}} \cup \{(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)})\}$
Return $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, x_i^{(b)})$

Corrupt(i):
$\mathcal{C} \leftarrow \mathcal{C} \cup \{i\}$; return $(\mathsf{sk}_i, \mathsf{ek}_i)$

Finalize(b'):
If $\mathsf{adm}(\mathcal{A}) = 1$, return $\beta \leftarrow (b' \stackrel{?}{=} b)$
Else, return $\beta \xleftarrow{\$} \{0,1\}$

Figure 7.1: Security game $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ for Definition 7.2

- $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ *and* $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ *for all* $i \in \mathcal{H}$,
- $x_i^{(0)} = x_i^{(1)}$ *and* $y_i^{(0)} = y_i^{(1)}$ *for all* $i \in \mathcal{C}$, *and*
- $f^{(0)}_{n,\lambda,y_1^{(0)},\ldots,y_n^{(0)}}(x_1^{(0)},\ldots,x_n^{(0)}) \neq f^{(1)}_{n,\lambda,y_1^{(1)},\ldots,y_n^{(1)}}(x_1^{(1)},\ldots,x_n^{(1)})$.

Otherwise, we say that $\mathcal{A}$ *is* admissible *w.r.t* $\mathcal{C}$, $\mathcal{Q}_{\mathsf{Enc}}$ *and* $\mathcal{Q}_{\mathsf{KGen}}$ *and write* $\mathsf{adm}(\mathcal{A}) = 1$. *We call* $\mathcal{E}$ *function-hiding if for all* ppt *adversaries* $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) := \left| \Pr\left[\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) = 1\right] - \frac{1}{2} \right|$$

is negligible in λ.

Weaker Notions. We define weaker variants of indistinguishability by restricting the access to the oracles and imposing stronger admissibility conditions. In this chapter we first present our main technical scheme under some weaker notions in Section 7.4.2, then our final scheme under stronger notions is obtained following some general lemmas (see Section 7.4.3).

1. *Security against Static Corruption:* The experiment $\mathbf{Exp}^{\mathsf{statfh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that all queries to the oracle **Corrupt** must be submitted before **Initialize** is called.

2. *Security against Selective Challenges:* The experiment $\mathbf{Exp}^{\mathsf{selfh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that all queries to the oracles **KeyGen** and **Enc** must be submitted before **Initialize** is called.

3. *One-time Security:* The experiment $\mathbf{Exp}^{\mathsf{1chal\text{-}fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that the adversary must declare up front to **Initialize** two additional "challenge" tags $\mathsf{tag}^*, \mathsf{tag\text{-}f}^* \in \mathsf{Tag}$ such that for all $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$:
   - if $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ and $\mathsf{tag} \neq \mathsf{tag}^*$, then $x_i^{(0)} = x_i^{(1)}$,
   - if $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ and $\mathsf{tag\text{-}f} \neq \mathsf{tag\text{-}f}^*$, then $y_i^{(0)} = y_i^{(1)}$.

4. *Security against Complete Challenges:* The experiment $\mathbf{Exp}^{\mathsf{pos\text{-}fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ is the same as $\mathbf{Exp}^{\mathsf{fh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$ except that we add the following condition 3 for $\mathsf{adm}(\mathcal{A}) = 0$ that we call the *complete-query constraint*:

   3. There exists $\mathsf{tag} \in \mathsf{Tag}$ so that a query $\mathbf{Enc}(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)})$ has been asked for some but not all $i \in \mathcal{H}$, or there exists $\mathsf{tag\text{-}f} \in \mathsf{Tag}$ such that a query $\mathbf{KeyGen}(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)})$ has been asked for some but not all $i \in \mathcal{H}$.

In other words, we require for an adversary $\mathcal{A}$ to be *admissible* that, for any tag, either $\mathcal{A}$ makes no encryption (resp. key) query or makes at least one encryption (resp. key) query for each slot $i \in \mathcal{H}$.

5. *Weak Function-Hiding:* We can weaken the function-hiding property by changing condition 2 for $\mathsf{adm}(\mathcal{A}) = 0$. More specifically, we replace it by the following condition 2':

   2'. There exist $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, $(x_i^{(0)})_{i\in[n]}$ *and* $(x_i^{(1)})_{i\in[n]}$ *in* $\mathcal{D}_1 \times \cdots \times \mathcal{D}_n$ *and two functions* $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}, f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})} \in \mathcal{F}$ *having parameters* $(y_i^{(0)}, y_i^{(1)})_{i=1}^n$ *such that*

   - $(i, \mathsf{tag}, x_i^{(0)}, x_i^{(1)}) \in \mathcal{Q}_{\mathsf{Enc}}$ *and* $(i, \mathsf{tag\text{-}f}, y_i^{(0)}, y_i^{(1)}) \in \mathcal{Q}_{\mathsf{KGen}}$ *for all* $i \in \mathcal{H}$,
   - $x_i^{(0)} = x_i^{(1)}$ *and* $y_i^{(0)} = y_i^{(1)}$ *for all* $i \in \mathcal{C}$, *and*
   - $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}(x_1^{(0)},\ldots,x_n^{(0)}) \neq f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(1)},\ldots,x_n^{(1)})$ OR
     $f^{(0)}_{n,\lambda,(y_1^{(0)},\ldots,y_n^{(0)})}(x_1^{(0)},\ldots,x_n^{(0)}) \neq f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(0)},\ldots,x_n^{(0)})$ OR
     $f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(0)},\ldots,x_n^{(0)}) \neq f^{(1)}_{n,\lambda,(y_1^{(1)},\ldots,y_n^{(1)})}(x_1^{(1)},\ldots,x_n^{(1)})$.

   The experiment in this weak function-hiding model is denoted by $\mathbf{Exp}^{\mathsf{wfh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda)$.

We also give a sequence of generic lemmas that can be used to strengthen the security model of our basic FH-DMCFE construction from Section 7.4.2. Specifically, we show how to remove the *complete-query* constraint and the restriction to *one-challenge* security. In this way, we obtain an FH-DMCFE for inner products whose only restrictions on the security model are *static corruptions* and a *polynomially bounded number of repetitions for decryption keys*.

Security against Incomplete Queries. To remove the complete-queries constraint, previous works [CDSG+20, AGT21b] make use of a technique called *all-or-nothing encapsulation* (AoNE). Roughly, AoNE allows all parties of a group to encapsulate individual messages, that can *all* be extracted by everyone if and only if all parties of the group have sent their contribution. Otherwise, *no* message is revealed. In the constructions of [CDSG+20, AGT21b], such an AoNE layer is added on top of both ciphertexts and keys. Intuitively, this approach allows the following reasoning: if an adversary makes encryption queries for all (honest) clients under some tag tag (i.e. the global query is "complete"), then the AoNE scheme allows to obtain all ciphertexts, and we can rely on the security of the DMCFE scheme that is secure against complete challenges. On the other hand, if the adversary queries only some but not all honest clients (i.e. the global query is "incomplete"), then the security of the AoNE scheme guarantees that the adversary does not learn anything about the encapsulated messages. While this construction is well known, previous constructions prove only selective security, even if the employed AoNE scheme is adaptively secure. Therefore, we think it is important to show that this AoNE layer indeed preserves adaptive security if the underlying scheme, which is only secure against complete queries, has this property.

More specifically, the notion of AoNE is a particular functionality of DDFE introduced by Chotard *et al.* [CDSG+20]. In [AGT21b], AoNE also serves as a building block for their FH-DDFE scheme, and it is pointed out that function-hiding and standard security are the same for AoNE, as there is no concept of keys. Since we are focusing on the less general notion DMCFE, we define AoNE in a less general context as a functionality for DMCFE.

Definition 7.3 (All-or-Nothing Encapsulation). *For* $n, \lambda \in \mathbb{N}$, *let* $\mathsf{Tag}_\lambda = \mathcal{R}_\lambda = \{0,1\}^{\mathsf{poly}(\lambda)}$, $\mathcal{K}_\lambda = \varnothing$, $\mathcal{M}_{n,\lambda,\mathrm{pub}} = [n] \times \mathsf{Tag}_\lambda$ *and* $\mathcal{M}_{\lambda,\mathrm{pri}} = \{0,1\}^L$ *for a polynomial* $L = L(\lambda) \colon \mathbb{N} \to \mathbb{N}$. *The* all-or-nothing encapsulation *functionality* $f^{\mathsf{aone}} = \{f^{\mathsf{aone}}_{n,\lambda} \colon \{[n]\} \times (\{[n]\} \times \mathcal{M}_\lambda)^n \to \mathcal{R}_\lambda\}_{n,\lambda\in\mathbb{N}}$ *is defined via*

$$f^{\mathsf{aone}}_{n,\lambda}([n], (i, m_i)_{i\in[n]}) = \begin{cases} (x_i)_{i\in[n]} & \text{if condition } (*) \text{ holds} \\ \bot & \text{otherwise} \end{cases}$$

for all $n, \lambda \in \mathbb{N}$*, where* $\{[n]\}$ *is a singleton consisting of* $[n]$ *as its only member, and condition* $(*)$ *holds if there exists* $\mathsf{tag} \in \mathsf{Tag}_\lambda$ *such that for each* $i \in [n]$*,* m_i *is of the form* $(m_{i,\mathrm{pri}} := x_i \in \{0,1\}^L, m_{i,\mathrm{pub}} := ([n], \mathsf{tag}) \in \mathcal{M}_{n,\lambda,\mathrm{pub}})$*.*

This means in particular that when using DMCFE for the functionality f^{aone}, DKeyGen is not needed and Dec works without taking secret keys as input. The DDFE constructions from [CDSG+20] yield two constructions of DMCFE for the function class AoNE *as per* Definition 7.3. A first generic construction [CDSG+20, Section 4] from identity-based encryption is secure in the standard model. Another concrete construction [CDSG+20, Section 5] from bilinear maps under the Decisional Bilinear Diffie-Hellman (DBDH) assumption is proven secure in the ROM.

We present our result in form of a generic conversion that turns any one-challenge DMCFE scheme secure against complete queries into one that is also secure against incomplete queries.

Lemma 7.4. *Assume there exist (1) a one-challenge (weakly function-hiding) DMCFE scheme* $\mathcal{E}^{\mathsf{pos}}$ *for a function class* $\mathcal{F}$ *that is secure against complete queries, and (2) an AoNE scheme* $\mathcal{E}^{aone}$ *whose message space contains the ciphertext space of* $\mathcal{E}^{\mathsf{pos}}$*. Then there exists a one-challenge (weakly function-hiding) DMCFE scheme* $\mathcal{E}$ *for* $\mathcal{F}$ *that is even secure against incomplete queries. More precisely, for any* ppt *adversary* $\mathcal{A}$*, there exist* ppt *algorithms* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\mathsf{Adv}^{\mathsf{mc\text{-}w\text{-}rep\text{-}xxx\text{-}1chal\text{-}cpa}}_{\mathcal{E},\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{(N_i)_{i=1}^n},q,\mathsf{LSSS},\mathcal{A}}(1^\lambda) \leq 12 \cdot \mathsf{Adv}^{\mathsf{mc\text{-}w\text{-}rep\text{-}pos\text{-}xxx\text{-}1chal\text{-}cpa}}_{\mathcal{E}^{\mathsf{pos}},\mathcal{F}^{\mathsf{IP},\mathsf{poly}}_{(N_i)_{i=1}^n},q,\mathsf{LSSS},\mathcal{B}_1}(1^\lambda) + 12 \cdot \mathsf{Adv}^{\mathsf{1chal\text{-}xxx\text{-}wfh}}_{\mathcal{E}^{aone},f^{aone},\mathcal{B}_2}(1^\lambda) \ ,$$

where $\mathsf{xxx} \subseteq \{\mathsf{stat}, \mathsf{sel}\}$*.*

Our conversion simply adds a layer of DMCFE for AoNE on top of both ciphertexts and keys. On an intuitive level, our simulator initially guesses whether or not the oracle queries for the challenge tag $\mathsf{tag\text{-}f}^*$ (or tag^*) will be complete. If the guess was "complete" and this guess turns out to be correct at the end of the game, then the simulator attacks the underlying DMCFE scheme that is assumed to be secure against complete queries. If the guess was "incomplete" and the guess is correct, then the simulator attacks the security of the AoNE scheme. If the guess was incorrect (which happens with probability 1/2), then the simulator aborts with a random bit. In this way, we can upper bound the advantage of a distinguisher between two successive hybrids in terms of the advantages that efficient adversaries can achieve against the underlying AoNE and DMCFE schemes. We point out that this argument crucially relies on the *one-challenge* setting. Due to the guess on the (in)completeness of the oracle queries, we lose a factor 1/2 in the security proof. Thus, a hybrid argument over a polynomial number of incomplete queries would incur an exponential security loss. Therefore, it is important to add security against incomplete queries in the one-challenge model.

Details about the conversion as well as the proof are given in [NPS24b, NPS24a, Appendix B.1]. We mention that a concurrent work by Shi and Vanjani [SV23] presents a similar conversion in the MCFE setting.

Security against Multiple Challenges. It remains to discuss how a one-challenge FH-DMCFE scheme for inner products can be made resistant against multiple challenge queries. First, observe that the equivalence of one-challenge and multi-challenge security in the standard setting (without function privacy) is trivial. Indeed, the proof can be done by a sequence of hybrids over the different tags queried to the encryption oracle. This approach, however, does not directly generalize to the function-hiding setting. The problem is that now both encryption and key-generation queries depend on the challenge bit $b \in \{0,1\}$. Since ciphertexts and keys can be arbitrarily combined in general, such a sequence of hybrids leads to a situation where an adversary is able to mix ciphertexts that encrypt the left message with keys generated for the right function or vice versa. However, the function-hiding admissibility does not provide any security guarantees in the case of such a mixed decryption. Therefore, we cannot change

ciphertexts and keys one by one anymore. We solve this problem by first proving security against multiple challenges in the *weakly function-hiding* setting. This model provides us exactly with the necessary guarantee for mixed decryptions, which allows a hybrid argument over all function and message tags to subsequently swap keys and ciphertexts. Afterwards, we apply another standard transformation that turns weakly function-hiding DMCFE schemes for inner products back into full-fledged function-hiding DMCFE (see Lemma 7.6). Previous works [LV16, ACF+18] presented that transformation for single-input and multi-input FE schemes.

We state the formal lemmas below. The proofs are standard and the latter is very similar to [LV16, ACF+18].The full proofs can be referred in [NPS24b, Appendix B.2, B.3] for completeness.

Lemma 7.5. *Let* $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a DMCFE scheme for the function class* $\mathcal{F}$*. If* $\mathcal{E}$ *is one-challenge weakly function-hiding, then it is also weakly function-hiding. More specifically, for any* ppt *adversary* $\mathcal{A}$*, there exists a* ppt *algorithm* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{xxx\text{-}wfh}}_{\mathcal{E},\mathcal{F},\mathcal{A}}(1^\lambda) \leq (q_e + q_k) \cdot \mathsf{Adv}^{\mathsf{1chal\text{-}xxx\text{-}wfh}}_{\mathcal{E},\mathcal{F},\mathcal{B}}(1^\lambda) \ ,$$

where q_e *and* q_k *denote the maximum numbers of different tags* tag *and* tag-f *that* $\mathcal{A}$ *can query to* **Enc** *and* **DKeyGen** *respectively, and* $\mathsf{xxx} \subseteq \{\mathsf{stat}, \mathsf{sel}, \mathsf{pos}\}$*.*

Lemma 7.6. *If there exists a weakly function-hiding DMCFE scheme* $\mathcal{E}$ *for* $\mathcal{F}^{\mathsf{ip}}$*, then there exists a (fully) function-hiding DMCFE scheme* $\mathcal{E}'$ *for* $\mathcal{F}^{\mathsf{ip}}$*. More precisely, for any* ppt *adversary* $\mathcal{A}$*, there exists a* ppt *algorithm* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{xxx\text{-}fh}}_{\mathcal{E}',\mathcal{F}^{\mathsf{ip}},\mathcal{A}}(1^\lambda) \leq 3 \cdot \mathsf{Adv}^{\mathsf{xxx\text{-}wfh}}_{\mathcal{E},\mathcal{F}^{\mathsf{ip}},\mathcal{B}}(1^\lambda) \ ,$$

where $\mathsf{xxx} \subseteq \{\mathsf{stat}, \mathsf{sel}, \mathsf{1chal}, \mathsf{pos}\}$*.*

7.4 A FH-DMCFE for Inner Products

7.4.1 Swapping Lemma

In this section we state a technical lemma that will be the basis of the security analysis of our function-hiding IP-DMCFE. This lemma plays an important role in the proof of Theorem 7.8 and is revisited in Section 7.4.2. As a reminder, we refer to the paragraph **Problems for Adaptive Security** in the technical overview of Section 7.2 for a discussion on why the oracles in the following statement of Lemma 7.7 are relevant afterwards in the FH-DMCFE proof.

Lemma 7.7 (Swapping). *Let $\lambda \in \mathbb{N}$ and $H = H(\lambda), K = K(\lambda), L = L(\lambda), J_i = J_i(\lambda), \widetilde{J}_i = \widetilde{J}_i(\lambda), N = N(\lambda) \in \mathbb{N}$ where $i \in [H]$ and $H, K, L, J_i, \widetilde{J}_i, N : \mathbb{N} \to \mathbb{N}$ are polynomials. Let $\tilde{J} := \max_{i\in[H]}\{\tilde{J}_i\}$, where the maximum is over polynomial evaluations $\tilde{J}_i(\lambda) \in \mathbb{N}$. Let $(\mathbf{B}_i, \mathbf{B}_i^*)$, for each $i \in [H]$, be a pair of random dual bases of dimension $2N+2N\cdot\tilde{J}+4$ in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$. All basis vectors are kept secret. Let $R, R_1, \ldots, R_K \in \mathbb{Z}_q$ be some public scalars. For $i \in [H]$, $\ell \in [L]$ and $k \in [K]$, sample $\sigma_i, \sigma_{i,k}, r, r_\ell \xleftarrow{\$} \mathbb{Z}_q$ conditioned on $\sum_{i\in[H]} \sigma_i = R$ and $\sum_{i\in[H]} \sigma_{k,i} = R_k$.*

We consider the following oracles:

$\underline{\tilde{\mathcal{O}}_\mathbf{d}}$**:** *On input $(\ell, i, \mathbf{y}_{\ell,i}^{(\mathsf{rep})}, \mathbf{y}_{\ell,i}^{(\mathsf{rep})'}) \in [L] \times [H] \times \mathbb{Z}_q^N \times \mathbb{Z}_q^N$, where $\mathsf{rep} \in [J_i]$ is a counter for the number of queries of the form $(\ell, i, \star, \star)$, sample $\rho_{\ell,i}^{(\mathsf{rep})} \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (\mathbf{y}_{\ell,i}^{(\mathsf{rep})},\ \mathbf{y}_{\ell,i}^{(\mathsf{rep})'},\ r_\ell,\ 0,\ \rho_{\ell,i}^{(\mathsf{rep})},\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i}\ .$$

$\underline{\boxed{\mathcal{O}_\mathbf{d}^b}}$**:** *For $b \in \{0,1\}$, on input $(i, \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)}) \in [H] \times \mathbb{Z}_q^N$, where $\tilde{j}_i \in [\tilde{J}_i]$ is a counter for the number of queries of the form $(i, \star, \star)$, sample $\rho_i^{(\tilde{j}_i)} \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\text{If } \boxed{b=0}: \quad \mathbf{d}_i^{(\tilde{j}_i)} = (\boxed{\mathbf{y}_i^{(1,\tilde{j}_i)}},\ \boxed{0^N},\ r,\ 0,\ \rho_i^{(\tilde{j}_i)},\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i}$$
$$\text{If } \boxed{b=1}: \quad \mathbf{d}_i^{(\tilde{j}_i)} = (\boxed{0^N},\ \boxed{\mathbf{y}_i^{(0,\tilde{j}_i)}},\ r,\ 0,\ \rho_i^{(\tilde{j}_i)},\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i}\ .$$

$\underline{\mathcal{O}_\mathbf{c}}$**:** *On input $(i, \mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)}) \in [H] \times \mathbb{Z}_q^N \times \mathbb{Z}_q^N$, where $j_i \in [J_i]$ is a counter for the number of queries of the form $(i, \star, \star)$, sample $\pi_i^{(j_i)} \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\mathbf{c}_i^{(j_i)} = (\mathbf{x}_i^{(1,j_i)},\ \mathbf{x}_i^{(0,j_i)},\ \sigma_i,\ \pi_i^{(j_i)},\ 0,\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i^*}\ .$$

$\underline{\tilde{\mathcal{O}}_\mathbf{c}}$**:** *On inputs $(k, i, \mathbf{x}_{k,i}^{(\mathsf{rep})}) \in [K] \times [H] \times \mathbb{Z}_q$, where $\mathsf{rep} \in [J_i]$ is a counter for the number of queries of the form $(k, i, \star)$, sample $\pi_{k,i}^{(\mathsf{rep})} \xleftarrow{\$} \mathbb{Z}_q$ and output*

$$\mathbf{c}_{k,i}^{(\mathsf{rep})} = (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i}^{(\mathsf{rep})},\ 0,\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i^*}\ .$$

If $\sum_{i=1}^H \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle = 0$ and $\sum_{i=1}^H \langle \mathbf{y}_i^{(1,\tilde{j}_i)} - \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(\mathsf{rep})} \rangle = 0$ for all $\tilde{j}_i \in [\tilde{J}_i], \mathsf{rep}, j_i \in [J_i]$, then the following advantage is negligible under the SXDH assumption:

$$\Big| \Pr[\mathcal{A}^{\tilde{\mathcal{O}}_\mathbf{d}, \boxed{\mathcal{O}_\mathbf{d}^0}}_{\tilde{\mathcal{O}}_\mathbf{c}, \mathcal{O}_\mathbf{c}}\Big(1^\lambda, N, H, K, L, (J_i, \widetilde{J}_i)_{i\in[H]}, R, (R_k)_{k\in[K]}\Big) \to 1]$$
$$- \Pr[\mathcal{A}^{\tilde{\mathcal{O}}_\mathbf{d}, \boxed{\mathcal{O}_\mathbf{d}^1}}_{\tilde{\mathcal{O}}_\mathbf{c}, \mathcal{O}_\mathbf{c}}\Big(1^\lambda, N, H, K, L, (J_i, \widetilde{J}_i)_{i\in[H]}, R, (R_k)_{k\in[K]}\Big) \to 1] \Big|$$
$$\leq (4n\tilde{J}N + 4) \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$$

where $\mathcal{A}$ can query the oracles $\tilde{\mathcal{O}}_\mathbf{d}$,$\boxed{\mathcal{O}_\mathbf{d}^b}$,$\mathcal{O}_\mathbf{c}$,$\tilde{\mathcal{O}}_\mathbf{c}$ adaptively, i.e. *the queries can be made in any order and any number of times respecting the (polynomial) upper bounds $K, L, (J_i, \widetilde{J}_i)_{i\in[H]}$.*

We give an informal proof sketch of the main ideas. The complete proof is given subsequently. The DPVS properties are once more used extensively, and the lemma itself is presented in a fairly modular way for independent interests.

Outline of the Proof. We explain the main steps in our proof as follows, where details about *formal* and *computational* basis changes can be revised from the examples in **Basis Changes** of Section 3.3. The proof is done so that for *all* the repetitions $\tilde{j}_i \in [\tilde{J}_i]$, we perform the change from the repetition $\mathbf{y}_i^{(0,\tilde{j}_i)}$ into $\mathbf{y}_i^{(1,\tilde{j}_i)}$ by the $\tilde{j}_i$-th block of isolated coordinates in the vectors $\mathbf{d}_i^{(\tilde{j}_i)}$. It is crucial that the polynomially large bound $\tilde{J} \geq \max_{i\in[n],\mathsf{tag\text{-}f}\in\mathsf{Tag}} \tilde{J}_{i,\mathsf{tag\text{-}f}}$ is known in advance, so as to well define the dimension of DPVS bases.

We start from the game where the sample given to the adversary $\mathcal{A}$ follows D_0 and the changes on vectors throughout the games are put in $\boxed{\text{boxes}}$. We use the notation $\mathbf{0} \coloneqq 0^N$ and write $\mathbf{0}^{\tilde{J}} \coloneqq \mathbf{0} \parallel \ldots \parallel \mathbf{0}$, for $\tilde{J}$ times. Our first step is to exploit the fact that $r \xleftarrow{\$} \mathbb{Z}_q$ is a uniformly random value and for each $j_i \in [J_i]$ all the secret shares σ_i in $\mathbf{c}_i^{(j_i)}$ sum to a known constant R. This helps us perform a *computational* basis change on $(\mathbf{B}_i, \mathbf{B}_i^*)$ and introduce a value $r' \xleftarrow{\$} \mathbb{Z}_q^*$ in $\mathbf{d}_i[2N + 2N \cdot \tilde{J} + 4]$ as well as random secret sharings of 0, common for $j_i \in [J_i]$, namely $(\tau_i)_{i=1}^H, (\tau'_{k,i})_{i=1}^H$, in $(\mathbf{c}_i^{(j_i)}[2N + 2N \cdot \tilde{j}_i + 4])_{i=1}^H$, $(\mathbf{c}_{k,i}^{(\mathsf{rep})}[2N + 2N \cdot \tilde{j}_i + 4])_{i=1}^H$. We use the hypothesis that all basis vectors are kept secret so that the computational basis change using DDH cannot be detected by the adversary. More details can be found in the transition $\mathsf{G}_0 \to \mathsf{G}_1$.

After G_1, we perform a *formal* duplication to go to G_2 in which we duplicate coordinates $[1, N], [N + 1, 2N]$ to the $\tilde{J}$ blocks $[2N \cdot \tilde{j} + 4, N + 2N \cdot \tilde{j} + 3], [N + 2N \cdot \tilde{j} + 4, 2N + 2N \cdot \tilde{j} + 3]$, where $\tilde{j}$ runs in $[\tilde{J}]$, in vectors $\mathbf{c}_i^{(j_i)}, \mathbf{c}_{k,i}^{(\mathsf{rep})}$ for all $i \in [H], k \in [K], j_i \in [J_i]$.

$$
\begin{array}{lccccccccccl}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \left(\mathbf{0}\right. & \left.\mathbf{0}\right)^{\tilde{J}} & 0 &)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{y}_i^{(1,\tilde{j}_i)} & \mathbf{0} & r & 0 & \rho_i^{(\tilde{j}_i)} & \left(\mathbf{0}\right. & \left.\mathbf{0}\right)^{\tilde{J}} & r' &)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \left(\boxed{\mathbf{x}_i^{(1,j_i)}}\right. & \left.\boxed{\mathbf{x}_i^{(0,j_i)}}\right)^{\tilde{J}} & \tau_i &)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \left(\boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}}\right. & \left.\boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}}\right)^{\tilde{J}} & \tau'_{k,i} &)_{\mathbf{B}_i^*}
\end{array}
$$

The duplication is done for *all* vectors $\mathbf{c}_i^{(j_i)}, \mathbf{c}_{k,i}^{(\mathsf{rep})}$ also across all repetitions $\mathsf{rep} \in [J]$. At a more technical level, this formal basis change will affect *all* vectors $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}, \mathbf{d}_i$ as well, also across all repetitions $\tilde{j}_i, \mathsf{rep} \in [\tilde{J}_i]$. Roughly speaking, by the duality of $(\mathbf{B}_i, \mathbf{B}_i^*)$, this basis change will incur "moving" coordinates $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{J} + 3], [N + 2N \cdot \tilde{J} + 4, 2N + 2N \cdot \tilde{J} + 3]$, for each $\tilde{j}_i \in [\tilde{J}]$ to $[1, N], [N + 1, 2N]$ in the $\mathbf{d}$-vectors. In this simple $\mathsf{G}_1 \to \mathsf{G}_2$, the moved coordinates contain 0, so they do not pose any problems.

After G_2, in all $\mathbf{c}$-vectors, each of the $\tilde{J}$ blocks $[2N \cdot \tilde{j} + 4, N + 2N \cdot \tilde{j} + 3], [N + 2N \cdot \tilde{j} + 4, 2N + 2N \cdot \tilde{J} + 3]$ contains a copy of the coordinates $[1, N], [N + 1, 2N]$. This allows us to perform a *computational* basis change under SXDH in order to swap between $[1, N]$ and $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3]$ in $\mathbf{d}_i^{(\tilde{j}_i)}$, for each $\tilde{j}_i \in [\tilde{J}_i]$ and $\tilde{J}_i \leq \tilde{J}$ by definition. We stress that for different $\tilde{j}_i$, the swap will move contents of $[1, N]$ to separated coordinates in different $\mathbf{d}_i^{(\tilde{j}_i)}$. In other words, for every $\tilde{j}_i, \tilde{j}'_i$, the coordinates $[2N \cdot \tilde{j}'_i + 4, N + 2N \cdot \tilde{j}'_i + 3]$ is well defined for $\mathbf{d}_i^{(\tilde{j}_i)}$ because $\tilde{j}_i \leq \tilde{J}_i \leq \tilde{J}$ and we have

$$
\mathbf{d}_i^{(\tilde{j}_i)}[2N \cdot \tilde{j}'_i + 4, N + 2N \cdot \tilde{j}'_i + 3] = \begin{cases} \mathbf{y}_i^{(1,\tilde{j}'_i)} & \text{if } \tilde{j}_i = \tilde{j}'_i \\ \mathbf{0} & \text{if } \tilde{j}_i \neq \tilde{j}'_i \end{cases} . \tag{7.8}
$$

The randomness is taken from ρ_i at coordinate $2N + 3$ in $\mathbf{d}_i^{(\tilde{j}_i)}$.

$$
\begin{array}{lccccccccccl}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 &)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \boxed{\mathbf{0}} & \mathbf{0} & r & 0 & \boxed{\rho_i^{(\tilde{j}_i)}} & \cdots & \boxed{\mathbf{y}_i^{(1,\tilde{j}_i)}} & \mathbf{0} & \cdots & r' &)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \cdots & \tau_i &)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \cdots & \tau'_{k,i} &)_{\mathbf{B}_i^*}
\end{array}
$$

As a sanity check, we observe that this change preserves the products $\mathbf{d}_i^{(\tilde{j}_i)} \times \mathbf{c}_i^{(j_i)}$ and $\mathbf{d}_i^{(\tilde{j}_i)} \times \mathbf{c}_{k,i}^{(\mathsf{rep})}$ for all $k \in [K], \tilde{j}_i \in [\tilde{J}_i]$. Moreover, the computational basis change allows us to target only the vectors $(\mathbf{d}_i^{(\tilde{j}_i)})_{i\in[H]}$ while letting $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}$ for $\ell \in [L], i \in [H]$ unchanged.

Upon reaching G_3, we are ready to approach the centerpiece of our proof. A *formal* basis change maintains perfectly identical views for the adversary in two games, resulting in a 0 difference in winning advantages under efficient simulation. We combine such formal basis changes with a *complexity leveraging* argument. In general, these kinds of arguments degrade the probability of a succesful simulation by an exponential factor. In our case, however, an exponential multiple of 0 is still 0. This implies that, as long as we restrict ourselves to formal bases changes that do not rely on any computational assumption, the simulator can initially guess all queries submitted by the adversary throughout the game, thus considering the selective game. We recall that the same strategy is also used previously in the proof of Theorem 6.11 for our MCFE for the class of inner products with LSSS access control in Section 6.4 of Chapter 6. The **Proof Strategy** of Theorem 6.11 can be revisited for the sake of revising the complexity leveraging technique.

Formal basis changes highlight the information-theoretic properties of DPVS. However, they are often much harder to use than computational changes. The reason is that a formal basis change affects *all* vectors, including all repetitions, in the same manner. In contrast to computational changes, it is not possible to apply changes only to *some* vectors. Intuitively, this is why in G_2 and G_3 we had to move all repetitions $\mathbf{d}_i^{(\tilde{j}_i)}$ into separate coordinates to prepare for the formal basis changes.

We now explain the sequence of games on which the complexity leveraging is applied. We want to perform some sort of swapping between coordinates $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3]$ and $[N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$ of $\mathbf{d}_i^{(\tilde{j}_i)}$ and reach G_6 whose vectors are:

$$
\begin{array}{lccccccccccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 &)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{0} & \mathbf{0} & r & 0 & \rho_i^{(\tilde{j}_i)} & \cdots & \boxed{\mathbf{0}} & \boxed{\mathbf{y}_i^{(0,\tilde{j}_i)}} & \cdots & r' &)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \boxed{\mathbf{x}_i^{(1,j_i)}} & \boxed{\mathbf{x}_i^{(0,j_i)}} & \cdots & \tilde{\tau}_i &)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \cdots & \tilde{\tau}'_{k,i} &)_{\mathbf{B}_i^*}
\end{array}
$$

The complexity leveraging will be applied to the *selective* versions $\mathsf{G}_3^* \to \mathsf{G}_4^* \to \mathsf{G}_5^* \to \mathsf{G}_6^*$ and only *formal* basis changes will be used in between. In these selective versions the simulator guesses the values $(\mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)})_{i\in[H]}^{\tilde{j}_i\in[\tilde{J}_i], j_i\in[J_i]}$ and the hybrids are conditioned on a "good" event that these guesses are correct. The "good" event happens with fixed probability. This leads to an identical adversary's view:

$$\Pr[\mathsf{G}_3^* = 1] = \Pr[\mathsf{G}_4^* = 1] = \Pr[\mathsf{G}_5^* = 1] = \Pr[\mathsf{G}_6^* = 1] \ . \tag{7.9}$$

We briefly highlight the selective games' ideas below:

- In $\mathsf{G}_3^* \to \mathsf{G}_4^*$ a formal basis change is applied to do a quotient by $\mathbf{y}_i^{(1,\tilde{j}_i)}[z]$ for $z \in [N]$ over all $\tilde{J}$ blocks $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3], [N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$, where $\tilde{j}_i$ runs in $[\tilde{J}_i]$, of $\mathbf{c}$-vectors. We refer to matrices (7.12) in the proof for more details. We note that thanks to (7.8), for $\tilde{j}_i \neq \tilde{j}'_i \in [\tilde{J}]$, this change makes $\mathbf{d}_i^{(\tilde{j}_i)}[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3] = \mathbf{1}$ while $\mathbf{d}_i^{(\tilde{j}_i)}[2N \cdot \tilde{j}'_i + 4, N + 2N \cdot \tilde{j}'_i + 3] = \mathbf{0}$ for $\tilde{j}'_i \neq \tilde{j}_i$.

- In $\mathsf{G}_4^* \to \mathsf{G}_5^*$, we define a formal basis change that uses the *fixed* randomness $r' \in \mathbb{Z}_q^*$ in $\mathbf{d}_i^{(\tilde{j}_i)}[2N + 2N \cdot \tilde{j}_i + 4]$ (introduced from G_1) to switch 1 to 0 at coordinates $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 4]$ while marking 1 at coordinates $[N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$ of $\mathbf{d}_i^{(\tilde{j}_i)}$, for *all* $\tilde{j}_i$. The specific matrix definition is given in equation (7.13). Thanks to the observation at the end of G_3^*, for each repetition $\mathbf{d}_i^{(\tilde{j}_i)}$ only the $\tilde{j}_i$-th blocks $[2N \cdot \tilde{j}_i + 4, N + 2N \cdot \tilde{j}_i + 3], [N + 2N \cdot \tilde{j}_i + 4, 2N + 2N \cdot \tilde{j}_i + 3]$ is affected, while other blocks stay $\mathbf{0}$. We note that unlike $\mathbf{d}_i^{(\tilde{j}_i)}$, the vectors $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}$ stay invariant because $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}[2N + 2N \cdot \tilde{J} + 4] = 0$.

Dually, because of the formal duplication in G_2 to all $\tilde{J} \geq \tilde{J}_i$ blocks, all $\mathbf{c}$-vectors will be altered such that the *accumulated* differences

$$\sum_{\substack{\tilde{j}_i \in [\tilde{J}_i] \\ z \in [N]}} \mathbf{c}_i^{(j_i)}[2N \cdot \tilde{j}_i + 3 + z] - \mathbf{c}_i^{(j_i)}[N + 2N \cdot \tilde{j}_i + 3 + z] = \frac{1}{r'} \sum_{\tilde{j}_i \in [\tilde{J}_i]} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle$$

will be added to τ_i in $\mathbf{c}_i^{(j_i)}[2N + 2N \cdot \tilde{J} + 4]$ (see (7.18) in the proof). For $\mathbf{c}_{k,i}^{(\mathsf{rep})}$, similarly, we have the accumulated differences added to $\tau'_{k,i}$ is

$$\sum_{\substack{\tilde{j}_i \in [\tilde{J}_i] \\ z \in [N]}} \mathbf{c}_{k,i}^{(\mathsf{rep})}[2N \cdot \tilde{j}_i + 3 + z] - \mathbf{c}_{k,i}^{(\mathsf{rep})}[N + 2N \cdot \tilde{j}_i + 3 + z] = \frac{1}{r'} \sum_{\tilde{j}_i \in [\tilde{J}_i]} \langle \mathbf{y}_i^{(1,\tilde{j}_i)} - \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_{k,i}^{(\mathsf{rep})} \rangle \ .$$

To show that this compensation for the accumulated differences in the τ_i and $\tau'_{k,i}$ cannot be noticed by the adversary, we exploit the conditions on the oracle queries in the statement of the lemma. Specifically, the condition $\sum_{i=1}^{H} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle = 0$ implies that $\frac{1}{r'}(\langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle)$ is constant for all $\tilde{j}_i \in [\tilde{J}], j_i \in [J_i]$ (see (7.11) for a formal argument) and $\sum_{i \in [H]} \frac{1}{r'}(\langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle) = 0$. From this observation, it follows that after adding the value $1/r' \cdot \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle$ to τ_i for all $i \in [H]$, $(\tau_i)_{i \in [H]}$ is still a secret sharing of 0. The same reasoning applies for $1/r' \cdot \langle \mathbf{y}_i^{(1,\tilde{j}_i)} - \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_{k,i}^{(\mathsf{rep})} \rangle$ which is added to the secret sharing $(\tau'_{k,i})_{i=1}^{H}$ in $(\mathbf{c}_{k,i}^{(\mathsf{rep})}[2N + 2N \cdot \tilde{J} + 4])_{i=1}^{H}$.

- In $\mathsf{G}_5^* \to \mathsf{G}_6^*$ we redo the quotient, still being in the selective variants conditioned on the "good" event.
- Finally, we also emphasize that all above DPVS formal basis changes do *not* depend on the *exponentially large* number of combinations $(\mathbf{d}_i^{(\tilde{j}_i)})_{i \in [H]}$, up to repetitions $\tilde{j}_i \in [\tilde{J}_i]$. We use the fact that each $i \in [H]$ has its vectors written in an *independent* pair of bases $(\mathbf{B}_i, \mathbf{B}_i^*)$, along with the crucial property (7.8) that allows treating each $\tilde{j}_i$-th repetition in an isolated block of the $\mathbf{d}_i^{(\tilde{j}_i)}$ vector, all $(\mathbf{d}_i^{(\tilde{j}_i)})^{\tilde{j}_i \in [\tilde{J}_i]}$ at the same time. To summarize, the specific information theoretic property of DPVS formal basis changes makes sure that *all* vectors in $(\mathbf{B}_i, \mathbf{B}_i^*)$ will be modified according to the basis matrices. The matrices (7.12) and (7.13) change consistently the $\tilde{j}_i$-th block in all pairs $(\mathbf{d}_i^{(\tilde{j}_i)}, \mathbf{c}_i^{(j_i)})$. For different $\tilde{j}_i \neq \tilde{j}'_i$ property (7.8) makes sure those matrices' change are trivial, *i.e.* $\mathbf{0}$ stays $\mathbf{0}$, in $\tilde{j}'_i$-th block of $\mathbf{d}_i^{(\tilde{j}_i)}$. Furthermore, even though all $\tilde{J}_i \leq \tilde{J}$ blocks of $\mathbf{c}_i^{(j_i)}$ are changed consistently by the matrices, in terms of the contents of all $\mathbf{d}_i^{(\tilde{j}_i)}$, different $\mathbf{c}_i^{(j_i)}$ from different i cannot be combined because they are in different bases. The only constraint is a fixed polynomially large upper bound $\tilde{J} \geq \max_{i \in [n], \mathsf{tag\text{-}f} \in \mathsf{Tag}} \tilde{J}_{i,\mathsf{tag\text{-}f}}$ so that the dimensions are well defined.

The probability calculation (see footnote 6) of the complexity leveraging makes use of the fact that the "good" event happens with a fixed probability in conjunction with property (7.9), leading to $\Pr[\mathsf{G}_3 = 1] = \Pr[\mathsf{G}_4 = 1] = \Pr[\mathsf{G}_5 = 1] = \Pr[\mathsf{G}_6 = 1]$. Coming out of the complexity-leveraging argument, the very last step consists in swapping $\mathbf{x}_i$ from coordinates $[N+2N\cdot\tilde{j}_i+3, 2N+2N\cdot\tilde{j}_i+3]$ back to $[1, N]$ (see $\mathsf{G}_6 \to \mathsf{G}_7$) and some cleaning in order to make the vectors follow D_1 (see $\mathsf{G}_7 \to \mathsf{G}_8$).

The Full Proof. The full proof is presented below.

Proof (Of Lemma 7.7). The proof is done via a sequence of hybrid games. The games are depicted in Figure 7.2. Unless stated otherwise, for simpler notations in the following we omit the index i from $j_i \in [J_i], \tilde{j}_i \in [\tilde{J}_i], \mathbf{d}^{(\tilde{j}_i)}, \mathbf{c}^{(j_i)}$ and write $j \in [J], \tilde{j} \in [\tilde{J}], \mathbf{d}^{(\tilde{j})}, \mathbf{c}^{(j)}$. For each $i \in [H]$, the value j denotes the maximum number of possible repetitions $(\mathbf{d}_{\ell,i}^{(\mathsf{rep})})_{\mathsf{rep}}$, $(\mathbf{c}_i^{(j)})_j$, and $(\mathbf{c}_{k,i}^{(\mathsf{rep})})_{\mathsf{rep}}$, indexed

Game G_0: The vectors are sampled according to D_0. The poly-bound $\tilde{J} \geq \max_{i\in[H]}\{\tilde{J}_i\}$ is fixed. Indices are running $j_i \in [J_i], \tilde{j}_i \in [\tilde{J}_i]$.

Game G_1: (Random 0-Secret Sharing) $\sum_{i=1}^{H} \tau_i = \sum_{i=1}^{H} \tau'_{k,i} = 0$, $\mathbf{0} := 0^N$

$$\begin{array}{lcccccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \mathbf{0}^{\tilde{J}} & \mathbf{0}^{\tilde{J}} & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{y}_i^{(1,\tilde{j}_i)} & \mathbf{0} & \boxed{r} & 0 & \rho_i^{(\tilde{j}_i)} & \mathbf{0}^{\tilde{J}} & \mathbf{0}^{\tilde{J}} & \boxed{r'} \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \boxed{\sigma_i} & \pi_i^{(j_i)} & 0 & \mathbf{0}^{\tilde{J}} & \mathbf{0}^{\tilde{J}} & \boxed{\tau_i} \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \mathbf{0}^{\tilde{J}} & \mathbf{0}^{\tilde{J}} & \boxed{\tau'_{k,i}} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Game G_2: (Formal Duplication from coordinates $[1,N],[N+1,2N]$ in $\mathbf{B}_i^*$, for $\mathbf{c}_i^{(j_i)}$ the coordinates $[2N\cdot\tilde{j}_i+4, 2N\cdot\tilde{j}_i+N+3], [3N\cdot\tilde{j}_i+4, 4N\cdot\tilde{j}_i+N+3]$ change)

$$\begin{array}{lcccccccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{y}_i^{(1,\tilde{j}_i)} & \mathbf{0} & r & 0 & \rho_i^{(\tilde{j}_i)} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & r' \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \boxed{\mathbf{x}_i^{(1,j_i)}} & \boxed{\mathbf{x}_i^{(0,j_i)}} & \cdots & \tau_i \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \cdots & \tau'_{k,i} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Game G_3: (Computational Swapping between $[1,N]$ and $[2N\cdot\tilde{j}_i+4, 3N\cdot\tilde{j}_i+N+3]$ in $\mathbf{d}_i^{(\tilde{j}_i)}$ using $(2N+3)$-randomness in $\mathbf{B}_i$, by $n\cdot\tilde{J}\cdot N$ DSDH instances)

$$\begin{array}{lcccccccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \boxed{\mathbf{0}} & \mathbf{0} & r & 0 & \boxed{\rho_i^{(\tilde{j}_i)}} & \cdots & \boxed{\mathbf{y}_i^{(1,\tilde{j}_i)}} & \mathbf{0} & \cdots & r' \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \cdots & \tau_i \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \cdots & \tau'_{k,i} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Inside a *complexity leveraging* argument, at the same time for all repetitions $\tilde{j}_i \in [\tilde{J}_i]$ of $\mathbf{d}_i^{(\tilde{j}_i)}$, m runs in $[N]$:

Game G_4^*: (Formal Quotient on coordinates $[2N\cdot\tilde{j}_i+4, 2N\cdot\tilde{j}_i+2N+3]$ in $\mathbf{B}_i$)

$$\begin{array}{lcccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \cdots & 1^N & \mathbf{0} & \cdots & r' \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \cdots & (\mathbf{y}_i^{(1,\tilde{j}_i)}[m]\mathbf{x}_i^{(1,j_i)}[m])_m & (\mathbf{y}_i^{(0,\tilde{j}_i)}[m]\mathbf{x}_i^{(0,j_i)}[m])_m & \cdots & \tau_i \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \cdots & (\mathbf{y}_i^{(1,\tilde{j}_i)}[m]\mathbf{x}_{k,i}^{(\mathsf{rep})}[m])_m & (\mathbf{y}_i^{(0,\tilde{j}_i)}[m]\mathbf{x}_{k,i}^{(\mathsf{rep})}[m])_m & \cdots & \tau'_{k,i} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Game G_5^*: $\tilde{\tau}_i := \tau_i + \frac{1}{r'}\sum_{\tilde{j}\in[\tilde{J}]}\left(\langle \mathbf{y}^{(\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)}\rangle - \langle \bar{\mathbf{y}}^{(\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)}\rangle\right)$, $\tilde{\tau}'_{k,i} := \tau'_{k,i} + \frac{1}{r'}\sum_{\tilde{j}\in[\tilde{J}]}\left(\langle \mathbf{y}^{(\tilde{j}_i)}, \mathbf{x}_{k,i}^{(\mathsf{rep})}\rangle - \langle \bar{\mathbf{y}}^{(\tilde{j}_i)}, \mathbf{x}_{k,i}^{(\mathsf{rep})}\rangle\right)$ (Formal Swapping)

$$\begin{array}{lcccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \cdots & \boxed{\mathbf{0}} & \boxed{1^N} & \cdots & r' \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \cdots & (\mathbf{y}_i^{(1,\tilde{j}_i)}[m]\mathbf{x}_i^{(1,j_i)}[m])_m & (\mathbf{y}_i^{(0,\tilde{j}_i)}[m]\mathbf{x}_i^{(0,j_i)}[m])_m & \cdots & \boxed{\tilde{\tau}_i} \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \cdots & (\mathbf{y}_i^{(1,\tilde{j}_i)}[m]\mathbf{x}_{k,i}^{(\mathsf{rep})}[m])_m & (\mathbf{y}_i^{(0,\tilde{j}_i)}[m]\mathbf{x}_{k,i}^{(\mathsf{rep})}[m])_m & \cdots & \boxed{\tilde{\tau}'_{k,i}} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Game G_6^*: (Formal Quotient on coordinates $[2N\cdot\tilde{j}_i+4, 2N\cdot\tilde{j}_i+2N+3]$ in $\mathbf{B}_i$)

$$\begin{array}{lcccccccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{0} & \mathbf{0} & r & 0 & \rho_i^{(\tilde{j}_i)} & \cdots & \boxed{\mathbf{0}} & \boxed{\mathbf{y}_i^{(0,\tilde{j}_i)}} & \cdots & r' \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \boxed{\mathbf{x}_i^{(1,j_i)}} & \boxed{\mathbf{x}_i^{(0,j_i)}} & \cdots & \tilde{\tau}_i \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}} & \cdots & \tilde{\tau}'_{k,i} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Game G_7: (Computational Swapping between $[1,N]$ and $[2N\cdot\tilde{j}_i+4, 3N\cdot\tilde{j}_i+N+3]$ in $\mathbf{d}_i^{(\tilde{j}_i)}$ using $(2N+3)$-randomness in $\mathbf{B}_i$, by $n\cdot J\cdot N$ DSDH instances)

$$\begin{array}{lcccccccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{0} & \boxed{\mathbf{y}_i^{(0,\tilde{j}_i)}} & r & 0 & \rho_i^{(\tilde{j}_i)} & \cdots & \mathbf{0} & \boxed{\mathbf{0}} & \cdots & r' \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \cdots & \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \cdots & \tilde{\tau}_i \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \cdots & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \cdots & \tilde{\tau}'_{k,i} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Game G_8: Undo G_2, G_1 (Cleaning) – Vectors sampled according to D_1.

$$\begin{array}{lcccccccc}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (& \mathbf{y}_{\ell,i}^{(\mathsf{rep})} & \mathbf{y}_{\ell,i}^{(\mathsf{rep})'} & r_\ell & 0 & \rho_{\ell,i}^{(\mathsf{rep})} & \mathbf{0}^{\tilde{J}} & \mathbf{0}^{\tilde{J}} & 0 \quad)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j}_i)} = (& \mathbf{0} & \mathbf{y}_i^{(0,\tilde{j}_i)} & r & 0 & \boxed{\rho_i^{(\tilde{j}_i)}} & \mathbf{0}^{\tilde{J}} & \boxed{\mathbf{0}^{\tilde{J}}} & \boxed{0} \quad)_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j_i)} = (& \mathbf{x}_i^{(1,j_i)} & \mathbf{x}_i^{(0,j_i)} & \sigma_i & \pi_i^{(j_i)} & 0 & \boxed{\mathbf{0}^{\tilde{J}}} & \boxed{\mathbf{0}^{\tilde{J}}} & \boxed{0} \quad)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} = (& \mathbf{x}_{k,i}^{(\mathsf{rep})} & \mathbf{x}_{k,i}^{(\mathsf{rep})} & \sigma_{k,i} & \pi_{k,i}^{(\mathsf{rep})} & 0 & \boxed{\mathbf{0}^{\tilde{J}}} & \boxed{\mathbf{0}^{\tilde{J}}} & \boxed{0} \quad)_{\mathbf{B}_i^*}
\end{array}$$

Figure 7.2: Games for proving Lemma 7.7.

by rep and j over all ℓ, k. The bound $\tilde{J} \geq \max_{i\in[H]}\{\tilde{J}_i\}$ of repetitions queried by the adversary for $\mathbf{d}$-vectors is fixed in advance. For the ease of notation, we define $J \coloneqq \max_{i\in[H]}\{J_i\}$ as the number of repetitions queried by the adversary for $\mathbf{c}$-vectors, not fixed in advance. We note that the dimensions of the DPVS bases depends on the $\tilde{J}$, *i.e.* the maximum number of repetitions we allow the adversary to make on $\mathbf{d}_i^{(\tilde{j})}$. We use notation $\mathbf{0} \coloneqq 0^N$ and write $\mathbf{0}^{\tilde{J}} \coloneqq \mathbf{0} \parallel \ldots \parallel \mathbf{0}$, for $\tilde{J}$ times. We describe the sequence of hybrids below.

Game G_0: The vectors are computed according to the interaction:

$$\mathcal{A}^{\tilde{\mathcal{O}}_{\mathbf{d}},\boxed{\mathcal{O}^0_{\mathbf{d}}}}_{\tilde{\mathcal{O}}_{\mathbf{c}},\mathcal{O}_{\mathbf{c}}}\Big(1^\lambda, N, H, K, L, (J_i, \widetilde{J}_i)_{i\in[H]}, R, (R_k)_{k\in[K]}\Big) \ .$$

Game G_1: The transition is completely computational and we can target solely all the $\tilde{j}$-th repetitions $\mathbf{d}_i^{(\tilde{j})}$, while leaving all $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}$ unchanged:

$$\begin{aligned}
&\mathbf{d}_{\ell,i}^{(\mathsf{rep})} = (\mathbf{y}_{\ell,i}^{(\mathsf{rep})},\ \mathbf{y}_{\ell,i}^{(\mathsf{rep})'},\ r_\ell,\ 0,\ \rho_{\ell,i}^{(\mathsf{rep})},\ 0^{2N\cdot\tilde{J}+1})_{\mathbf{B}_i}\\
&(\mathbf{d}_i^{(\tilde{j})} = (\mathbf{y}_i^{(1,\tilde{j})},\ \mathbf{0},\ r,\ 0,\ \rho_i^{(\tilde{j})},\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ \boxed{r'})_{\mathbf{B}_i})_{i\in[H]}\\
&(\mathbf{c}_i^{(j)} = (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ \boxed{\tau_i})_{\mathbf{B}_i^*})^{j\in[J]}_{i\in[H]}\\
&(\mathbf{c}_{k,i}^{(\mathsf{rep})} = (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i}^{(\mathsf{rep})},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ \boxed{\tau'_{k,i}})_{\mathbf{B}_i^*})^{\mathsf{rep}\in[J]}_{i\in[H],k\in[K]} \ .
\end{aligned}$$

where $r' \xleftarrow{\$} \mathbb{Z}_q^*$ and for all $j \in [J]$, the secret sharings $(\tau_i)_i$ and $(\tau'_{k,i})_i$ in $\mathbf{c}_i$-vectors satisfies: $\sum_{i=1}^H \tau_i = \sum_{i=1}^H \tau'_{k,i} = 0$, for $k \in [K]$. We emphasize that the same share τ_i is used across all repetitions $\mathbf{c}_i^{(j)}$ for a given i. Moreover, for the $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}$-vectors we do not introduce additional randomness such as r', which is enabled by the current computational change where we can compute vectors in $\mathbf{B}_i$ (*i.e.* $\mathbf{d}_{\ell,i}^{(\mathsf{rep})}$-vectors versus $\mathbf{d}_i^{(j')}$ vectors) differently.

We proceed in two steps:

Game $\mathsf{G}_{0.1}$: We first use the subspace-indistinguishability to introduce $r' \xleftarrow{\$} \mathbb{Z}_q^*$ at coordinate $2N + 2N\cdot\tilde{J} + 4$ of $\mathbf{d}_i^{(\tilde{j})}$, while keeping $\mathbf{c}_i^{(j)}[2N + 2N\cdot\tilde{J} + 4] = \mathbf{d}_{\ell,i}^{(\mathsf{rep})}[2N + 2N\cdot\tilde{J} + 4] = \mathbf{c}_{k,i}^{(\mathsf{rep})}[2N + 2N\cdot\tilde{J} + 4] = 0$. Given a DSDH instance $([\![a]\!]_1, [\![b]\!]_1, [\![c]\!]_1)$ in $\mathbb{G}_1$ where $\delta \coloneqq c - ab$ is either 0 or 1, the basis changing matrices are:

$$\mathbf{B}_i = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}_{2N+1,2N+2N\cdot\tilde{J}+4} \cdot \mathbf{H}_i; \quad \mathbf{B}_i^* = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix}_{2N+1,2N+2N\cdot\tilde{J}+4} \cdot \mathbf{H}_i^* \ .$$

All vectors changed under these bases are secret. We compute $\mathbf{B}_i$ using $[\![a]\!]_1$ and write the $\mathbf{d}$-vectors as follows:

$$\begin{aligned}
\mathbf{d}_i^{(\tilde{j})} &= (\mathbf{y}_i^{(\tilde{j})},\ \mathbf{0},\ r,\ 0,\ \rho_i,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}_i} + (\mathbf{0},\ \mathbf{0},\ br',\ 0,\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ cr')_{\mathbf{H}_i}\\
&= (\mathbf{y}_i^{(\tilde{j})},\ \mathbf{0}, \boxed{r + br'},\ 0,\ \rho_i,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}}, \boxed{\delta r'})_{\mathbf{B}_i}\\
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} &= (\mathbf{y}_{\ell,i}^{(\mathsf{rep})},\ \mathbf{y}_{\ell,i}^{(\mathsf{rep})'},\ r_\ell,\ 0,\ \rho_{\ell,i},\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}_i} \ .
\end{aligned}$$

We cannot compute $\mathbf{b}^*_{i,2N+1}$ but can write the $\mathbf{c}$-vectors in $\mathbf{H}^*$ and observe that they stay invariant in $\mathbf{B}_i^*$ as the $(2N + 2N\cdot\tilde{J} + 4)$-th coordinate is 0:

$$\begin{aligned}
\mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{H}_i^*}\\
&= (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}_i^*}\\
\mathbf{c}_{k,i}^{(\mathsf{rep})} &= (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{H}_i^*}\\
&= (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}_i^*} \ .
\end{aligned}$$

If $\delta = 0$ we are in G_0 else we are in $\mathsf{G}_{0.1}$, while updating r to $r + br'$[5]. The difference in advantages is $|\Pr[\mathsf{G}_{0.1} = 1] - \Pr[\mathsf{G}_0 = 1]| \leq 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}_1}(1^\lambda)$.

Game $\mathsf{G}_{0.2}$: We use DSDH in $\mathbb{G}_2$ to introduce any chosen secret sharings $(\tau_i)_{i\in[H]}$ and $(\tau'_{k,i})_{i\in[H]}$ of 0, *i.e.* $\sum_{i=1}^H \tau_i = \sum_{i=1}^H \tau'_{k,i} = 0$, such that $\tau_i, \tau'_{k,i} \neq 0$ for all i, for every $k \in [K]$. The secret sharings do not depend on the repetitions. Given a DSDH instance $([\![a]\!]_2, [\![b]\!]_2, [\![c]\!]_2)$ in $\mathbb{G}_2$ where $\delta \coloneqq c - ab$ is either 0 or 1, the bases $(\mathbf{B}_i, \mathbf{B}^*_i)$ are changed following:

$$\mathbf{B}_i = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix}_{2N+1,2N+2N\cdot\tilde{J}+4} \cdot \mathbf{H}_i; \quad \mathbf{B}^*_i = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}_{2N+1,2N+2N\cdot\tilde{J}+4} \cdot \mathbf{H}^*_i \ .$$

All vectors changed under these bases are secret. We compute $\mathbf{B}^*_i$ using $[\![a]\!]_2$ and write the $\mathbf{c}$-vectors as follows:

$$\begin{aligned}
\mathbf{c}^{(j)}_i &= (\mathbf{x}^{(1,j)}_i,\ \mathbf{x}^{(0,j)}_i,\ \sigma_i,\ \pi^{(j)}_i,\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}^*_i} \\
&\quad + (\mathbf{0},\ \mathbf{0},\ b\tau_i,\ 0,\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ c\tau_i)_{\mathbf{H}^*_i} \\
&= (\mathbf{x}^{(1,j)}_i,\ \mathbf{x}^{(0,j)}_i, \boxed{\sigma_i + b\tau_i},\ \pi^{(j)}_i,\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}}, \boxed{\delta\tau_i})_{\mathbf{B}^*_i} \\
\mathbf{c}^{(\mathsf{rep})}_{k,i} &= (\mathbf{x}^{(\mathsf{rep})}_{k,i},\ \mathbf{x}^{(\mathsf{rep})}_{k,i},\ \sigma_{k,i},\ \pi^{(\mathsf{rep})}_{k,i},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}^*_i} \\
&\quad + (\mathbf{0},\ \mathbf{0},\ b\tau'_{k,i},\ 0,\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ c\tau'_{k,i})_{\mathbf{H}^*_i} \\
&= (\mathbf{x}^{(\mathsf{rep})}_{k,i},\ \mathbf{x}^{(\mathsf{rep})}_{k,i},\ \boxed{\sigma_{k,i} + b\tau'_{k,i}},\ \pi^{(\mathsf{rep})}_{k,i},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ \boxed{\delta\tau'_{k,i}})_{\mathbf{B}^*_i} \ .
\end{aligned}$$

For each $j \in [J]$, the secret shares $(\sigma_i)_{i=1}^H$ are updated to $(\sigma_i + b\tau_i)_{i=1}^H$ and still satisfy:

$$\sum_{i=1}^H (\sigma_i + b\tau_i) = \left(\sum_{i=1}^H \sigma_i\right) + b\left(\sum_{i=1}^H \tau_i\right) = R$$

because $(\tau_i)_{i=1}^H$ is a secret sharing of 0. Similarly, $(\sigma_{k,i})_{i=1}^H$ are updated to $(\sigma_{k,i} + b\tau'_{k,i})_{i=1}^H$ and stay shares of R_k. We cannot compute $\mathbf{b}_{i,2N+2N\cdot\tilde{J}+4}$ but can write the $\mathbf{d}$-vectors in $\mathbf{H}_i$, for $r'', r_\ell \xleftarrow{\$} \mathbb{Z}_q$, $r' \xleftarrow{\$} \mathbb{Z}^*_q$:

$$\begin{aligned}
\mathbf{d}^{(\tilde{j})}_i &= (\mathbf{y}^{(1,\tilde{j})}_i,\ \mathbf{0},\ r'',\ 0,\ \rho_i,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ r')_{\mathbf{H}_i} \\
&= (\mathbf{y}^{(1,\tilde{j})}_i,\ \mathbf{0},\ r'' + ar',\ 0,\ \rho_i,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ r')_{\mathbf{B}_i} \\
\mathbf{d}^{(\mathsf{rep})}_{\ell,i} &= (\mathbf{y}^{(\mathsf{rep})}_{\ell,i},\ \mathbf{y}^{(\mathsf{rep})'}_{\ell,i},\ r_\ell,\ 0,\ \rho_{\ell,i},\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{H}_i} \\
&= (\mathbf{y}^{(\mathsf{rep})}_{\ell,i},\ \mathbf{y}^{(\mathsf{rep})'}_{\ell,i},\ r_\ell,\ 0,\ \rho_{\ell,i},\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}_i} \ ,
\end{aligned}$$

while simulating $r \coloneqq r'' + ar'$ perfectly uniformly at random in $\mathbb{Z}_q$. If $\delta = 0$ we are in $\mathsf{G}_{0.1}$, else we are in $\mathsf{G}_{0.2} = \mathsf{G}_1$. The difference in advantages is $|\Pr[\mathsf{G}_{0.2} = 1] - \Pr[\mathsf{G}_{0.1} = 1]| \leq 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}_2}(1^\lambda)$.

After $\mathsf{G}_{0.2} = \mathsf{G}_1$, the vectors are now:

$$\begin{aligned}
\mathbf{d}^{(\mathsf{rep})}_{\ell,i} &= (\mathbf{y}^{(\mathsf{rep})}_{\ell,i},\ \mathbf{x}^{(\mathsf{rep})}_{\ell,i},\ r_\ell,\ 0,\ \rho_{\ell,i},\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}_i} \\
\mathbf{d}^{(\tilde{j})}_i &= (\mathbf{y}^{(1,\tilde{j})}_i,\ \mathbf{0},\ \boxed{r},\ 0,\ \rho_i,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ r')_{\mathbf{B}_i} \\
\mathbf{c}^{(j)}_i &= (\mathbf{x}^{(1,j)}_i,\ \mathbf{x}^{(0,j)}_i,\ \boxed{\sigma_i},\ \pi^{(j)}_i,\ 0,\ 0^{2N\cdot\tilde{J}},\ \boxed{\tau_i})_{\mathbf{B}^*_i} \\
\mathbf{c}^{(\mathsf{rep})}_{k,i} &= (\mathbf{x}^{(\mathsf{rep})}_{k,i},\ \mathbf{x}^{(\mathsf{rep})}_{k,i},\ \boxed{\sigma_{k,i}},\ \pi_{k,i},\ 0,\ 0^{2N\cdot\tilde{J}}, \boxed{\tau'_{k,i}})_{\mathbf{B}^*_i}
\end{aligned}$$

and in total $|\Pr[\mathsf{G}_1 = 1] - \Pr[\mathsf{G}_0 = 1]| \leq 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}_2}(1^\lambda) + 2 \cdot \mathsf{Adv}^{\mathsf{DDH}}_{\mathbb{G}_1}(1^\lambda)$.

[5]It is thanks to the randomness of $r \xleftarrow{\$} \mathbb{Z}_q$ that allows us to update br' without changing the distribution. When applying this swapping lemma for our FH-DMCFE scheme, this random r is provided by the RO while hashing the tags.

Game G_2: We perform a formal duplication on *all* $\mathbf{c}$-vectors:

$$\left(\mathbf{c}_i^{(j)} = (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\ \left(\boxed{\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)}}\right)^{\tilde{J}},\ \tau_i)_{\mathbf{B}_i^*}\right)_{i\in[H]}^{j\in[J]}$$
$$\left(\mathbf{c}_{k,i}^{(\mathsf{rep})} = (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i}^{(\mathsf{rep})},\ 0,\ \left(\boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})}}\right)^{\tilde{J}},\ \tau'_{k,i})_{\mathbf{B}_i^*}\right)_{i\in[H],k\in[K]}^{\mathsf{rep}\in[J]}\ .$$

For $\mathbf{c}_i^{(j)}$ the coordinates $[2N\cdot\tilde{j}+4, 2N\cdot\tilde{j}+N+3], [2N\cdot\tilde{j}+N+4, 2N\cdot\tilde{j}+2N+3]$ change. We perform a formal basis change to duplicate $(\mathbf{x}_i^{(1,j)}, \mathbf{x}_i^{(0,j)})$ (respectively $(\mathbf{x}_{k,i}^{(\mathsf{rep})}, \mathbf{x}_{k,i}^{(\mathsf{rep})})$) from coordinates $[1,N], [N+1, 2N]$ to $[2N\cdot\tilde{j}+4, 2N\cdot\tilde{j}+N+3], [2N\cdot\tilde{j}+N+4, 2N\cdot\tilde{j}+2N+3]$ of $\mathbf{c}_i^{(j)}$ (respectively of $\mathbf{c}_{k,i}^{(\mathsf{rep})}$), *for all* $\tilde{j}\in[\tilde{J}]$. We emphasize that the pair $(\mathbf{x}_i^{(1,j)}, \mathbf{x}_i^{(0,j)})$ (respectively $(\mathbf{x}_{k,i}^{(\mathsf{rep})}, \mathbf{x}_{k,i}^{(\mathsf{rep})})$) are duplicated $\tilde{J}$ times into $\tilde{J}$ separated blocks $[2N\cdot\tilde{j}+4, 2N\cdot\tilde{j}+N+3], [2N\cdot\tilde{j}+N+4, 2N\cdot\tilde{j}+2N+3]$ in $\mathbf{c}_i^{(j)}$ (respectively in $\mathbf{c}_{k,i}^{(\mathsf{rep})}$). The bases are changed following using the following matrices (we denote $B_i[row, col]$ the entry at row row and column col of B_i)

$$B_i = \begin{cases} B_i[row,col] & = 1 \text{ if } row = col \\ B_i[row,col] & = 1 \text{ if } (row,col)\in\{(2N\tilde{j}+4+d, 1+d) : d\in[0,N-1], j\in[\tilde{J}]\} \\ B_i[row,col] & = 1 \text{ if } (row,col)\in\{(2N\tilde{j}+N+4+d, N+1+d) : d\in[0,N-1], j\in[\tilde{J}]\} \\ B_i[row,col] & = 0 \text{ otherwise} \end{cases}$$
$$B_i' \coloneqq \left(B_i^{-1}\right)^\top$$
$$\mathbf{B}_i = B_i\cdot\mathbf{H}_i;\quad \mathbf{B}_i^* = B_i'\cdot\mathbf{H}_i^*\ .$$

We write the vectors as follows, observing that the $\mathbf{d}$-vectors stay invariant because for all $\tilde{j}, \mathsf{rep}\in[\tilde{J}]$, their coordinates $[2N\cdot\tilde{j}+4, 2N\cdot\tilde{j}+N+3], [2N\cdot\tilde{j}+N+4, 2N\cdot\tilde{j}+2N+3]$ are all 0 and the duplication is done correctly for the $\mathbf{c}$-vectors:

$$\begin{aligned}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} &= (\mathbf{y}_{\ell,i}^{(\mathsf{rep})},\ \mathbf{y}_{\ell,i}^{(\mathsf{rep})'},\ r_\ell,\ 0,\ \rho_{\ell,i},\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{H}_i} \\
&= (\mathbf{y}_{\ell,i}^{(\mathsf{rep})},\ \mathbf{y}_{\ell,i}^{(\mathsf{rep})'},\ r_\ell,\ 0,\ \rho_{\ell,i},\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ 0)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j})} &= (\mathbf{y}_i^{(1,\tilde{j})},\ 0,\ r,\ 0,\ \rho_i,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ r')_{\mathbf{H}_i} \\
&= (\mathbf{y}_i^{(1,\tilde{j})},\ 0,\ r,\ 0,\ \rho_i,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ r')_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ \tau_i)_{\mathbf{H}_i^*} \\
&= (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\cdots,\ \boxed{\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)}},\cdots,\ \tau_i)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} &= (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i},\ 0,\ \mathbf{0}^{\tilde{J}},\ \mathbf{0}^{\tilde{J}},\ \tau'_{k,i})_{\mathbf{H}_i^*} \\
&= (\mathbf{x}_{k,i}^{(\mathsf{rep})}, \mathbf{x}_{k,i}^{(\mathsf{rep})}, \sigma_{k,i}, \pi_{k,i},\ 0,\cdots, \boxed{\mathbf{x}_{k,i}^{(\mathsf{rep})}, \mathbf{x}_{k,i}^{(\mathsf{rep})}}\cdots, \tau'_{k,i})_{\mathbf{B}_i^*}\ .
\end{aligned}$$

We are in G_1 in bases $(\mathbf{H}_i, \mathbf{H}_i^*)$ and in G_2 in bases $(\mathbf{B}_i, \mathbf{B}_i^*)$. The change is formal and we have $\Pr[\mathsf{G}_2 = 1] = \Pr[\mathsf{G}_1 = 1]$. Dually, the destination coordinates in the $\mathbf{d}$-vectors are all 0 hence they stay unchanged.

Game G_3: For each $\tilde{j}\in[\tilde{J}]$, we perform a computational swap between $[1,N]$ and $[2N\cdot\tilde{j}+4, 2N\cdot\tilde{j}+N+3]$ in $\mathbf{d}_i^{(\tilde{j})}$ using $(2N+3)$-randomness. We need $n\cdot\tilde{J}\cdot N$ DSDH instances $(\left[\!\left[a_{i,z}^{(\tilde{j})}\right]\!\right]_1, \left[\!\left[b_{i,z}^{(\tilde{j})}\right]\!\right]_1, \left[\!\left[c_{i,z}^{(\tilde{j})}\right]\!\right]_1)$ in $\mathbb{G}_1$ where $\delta_{i,z}^{(\tilde{j})} \coloneqq c_{i,z}^{(\tilde{j})} - a_{i,z}^{(\tilde{j})} b_{i,z}^{(\tilde{j})}$ is either 0 or $\mathbf{y}_i^{(1,\tilde{j})}[z]$, for $z\in[N], \tilde{j}\in[J], i\in[n]$. The basis changes for $(\mathbf{B}_i, \mathbf{B}_i^*)$ will use $(\left[\!\left[a_{i,z}^{(\tilde{j})}\right]\!\right]_1, \left[\!\left[b_{i,z}^{(\tilde{j})}\right]\!\right]_1, \left[\!\left[c_{i,z}^{(\tilde{j})}\right]\!\right]_1)$ for $z\in[N], \tilde{j}\in[J]$.

For a fixed i, focusing on the basis change of $(\mathbf{B}_i, \mathbf{B}_i^*)$, to incorporate the separate $\tilde{J}\cdot N$ DSDH instances and change the N coordinates over the $\tilde{J}$ repetitions, the computation

on the $\mathbf{c}$-vectors can be done thanks to the fact that we have enough $\tilde{J}$ separate $N \times N$ blocks in the matrix, these blocks are defined over the diagonal for the dual effects after our basis changes (we encourage revisiting Section 3.3 for the dual compuation of DPVS basis changes). More specifically, each of those distinct $N \times N$ block will change the N coordinates of each of the $\tilde{J}$ repetitions, where the matrices' entries are defined *as per* the DSDH instances.

The swaps are possible thanks to the pairs $(\mathbf{x}_i^{(1,j)}, \mathbf{x}_i^{(0,j)})$ (respectively $(\mathbf{x}_{k,i}^{(\mathsf{rep})}, \mathbf{x}_{k,i}^{(\mathsf{rep})})$) in $\mathbf{c}_i^{(j)}$ (respectively in $\mathbf{c}_{k,i}^{(\mathsf{rep})}$) that are resulted from previous game G_2.

It is important that the change is computational using DDH in $\mathbb{G}_1$, therefore we can write the vectors in appropriate bases using DSDH while targeting all the repetitions of $(\mathbf{d}_i^{(\tilde{j})})_i$. For instance, for a given $\tilde{j}$, we compute $\mathbf{B}_i$ using $[\![a]\!]_1$ and write the $\mathbf{d}$-vectors as follows:

$$\begin{aligned}
\mathbf{d}_i^{(\tilde{j})} &= (\underbrace{0,..,0,\mathbf{y}_i^{(1,\tilde{j})}[z],..,\mathbf{y}_i^{(1,\tilde{j})}[N]}_{\text{first } (z-1)\text{-th coords are 0}},\mathbf{0},r,0,\rho_i,\mathbf{0},..,\mathbf{0},\underbrace{\mathbf{y}_i^{(1,\tilde{j})}[1],..,\mathbf{y}_i^{(1,\tilde{j})}[z-1],0,..,0}_{\text{last } (N-z+1)\text{-th coords are 0}},\mathbf{0},..,\mathbf{0},r')_{\mathbf{B}_i} \\
&\quad + (\underbrace{0,..,0,-c_{i,z}^{(\tilde{j})},0,..,0}_{z\text{-th coord among } N},\mathbf{0},0,0,b_{i,z}^{(\tilde{j})},\mathbf{0},..,\mathbf{0},\underbrace{0,..,0,c_{i,z}^{(\tilde{j})},0,..,0}_{z\text{-th coord among } N},\mathbf{0},..,\mathbf{0},0)_{\mathbf{H}_i} \\
&= (\underbrace{0,..,0,\boxed{\mathbf{y}_i^{(1,\tilde{j})}[z]-\delta_{i,z}^{(\tilde{j})}},..,\mathbf{y}_i^{(1,\tilde{j})}[N]}_{\text{first } (z-1)\text{-th coords are 0}}, \\
&\qquad\qquad \mathbf{0},r,0,\boxed{\rho_i+b_{i,z}^{(\tilde{j})}},\mathbf{0},..,\mathbf{0},\underbrace{\mathbf{y}_i^{(1,\tilde{j})}[1],..,\mathbf{y}_i^{(1,\tilde{j})}[z-1],\boxed{\delta_{i,z}^{(\tilde{j})}},..,0}_{\text{last } (N-z)\text{-th coords are 0}},\mathbf{0},..,\mathbf{0},r')_{\mathbf{B}_i} \\
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} &= (\mathbf{y}_{\ell,i}^{(\mathsf{rep})},\mathbf{y}_{\ell,i}^{(\mathsf{rep})'},r_\ell,0,\rho_{\ell,i},\mathbf{0}^{\tilde{J}},\mathbf{0}^{\tilde{J}},0)_{\mathbf{B}_i}
\end{aligned}$$

where the coordinates $(\mathbf{0},\cdots,\mathbf{0})$ put in $\mathbf{H}_i$ will not be affected by the corresponding blocks in the basis matrix.

For all other $\mathbf{c}$-vectors their coordinates remain intact and are 0.

$$\begin{aligned}
\mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\ \cdots,\ \mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \cdots,\ \tau_i)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} &= (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i}^{(\mathsf{rep})},\ 0,\ \cdots,\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \cdots,\ \tau'_{k,i})_{\mathbf{B}_i^*}\ .
\end{aligned}$$

The security loss is $2 \cdot n \cdot \tilde{J} \cdot N \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$.

The vectors, when we arrive at G_3, are of the form:

$$\begin{aligned}
\mathbf{d}_{\ell,i}^{(\mathsf{rep})} &= (\mathbf{y}_{\ell,i}^{(\mathsf{rep})},\mathbf{y}_{\ell,i}^{(\mathsf{rep})'},r_\ell,0,\rho_{\ell,i}^{(\mathsf{rep})},\cdots,\mathbf{0},\mathbf{0},\cdots,0)_{\mathbf{B}_i} \\
\mathbf{d}_i^{(\tilde{j})} &= (\boxed{\mathbf{0}},\mathbf{0},r,0,\boxed{\rho_i},\cdots,\boxed{\mathbf{y}_i^{(1,\tilde{j})}},\mathbf{0},\cdots,r')_{\mathbf{B}_i} \\
\mathbf{c}_i^{(j)} &= (\mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \sigma_i,\ \pi_i^{(j)},\ 0,\ \cdots,\ \mathbf{x}_i^{(1,j)},\ \mathbf{x}_i^{(0,j)},\ \cdots,\ \tau_i)_{\mathbf{B}_i^*} \\
\mathbf{c}_{k,i}^{(\mathsf{rep})} &= (\mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \sigma_{k,i},\ \pi_{k,i}^{(\mathsf{rep})},\ 0,\ \cdots,\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \mathbf{x}_{k,i}^{(\mathsf{rep})},\ \cdots,\ \tau'_{k,i})_{\mathbf{B}_i^*}\ ,
\end{aligned}$$

where for each $j \in [J]$, $(\tau_i)_{i=1}^H$,$(\tau'_{k,i})_{i=1}^H$ are random secret sharings of 0, with $\tau_i, \tau'_{k,i} \neq 0$ for all i, and $r' \xleftarrow{\$} \mathbb{Z}_q^*$. Our goal in the next three games $\mathsf{G}_4, \mathsf{G}_5, \mathsf{G}_6$ is to swap $\mathbf{y}_i^{(1,\tilde{j})}$ from coordinates $[2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + N + 3]$ to $\mathbf{y}_i^{(0,\tilde{j})}$ coordinates $[2N \cdot \tilde{j} + N + 4, 2N \cdot \tilde{j} + 2N + 3]$ of $\mathbf{d}_i^{(\tilde{j})}$, for all $i \in [H]$. The main idea is to consider the *selective* version G_t^* for $t \in \{4,5,6\}$, where the values $(\mathbf{y}_i^{(1,\tilde{j})}[k], \mathbf{y}_i^{(0,\tilde{j})}[k], \mathbf{x}_i^{(1,j)}, \mathbf{x}_i^{(0,j)})_{i\in[H],k\in[N]}^{j\in[J]}$ are guessed in advance. We then use formal argument for the transitions $\mathsf{G}_t^* \to \mathsf{G}_{t+1}^*$ for $j \in \{3,4,5\}$ to obtain

$$\Pr[\mathsf{G}_3^* = 1] = \Pr[\mathsf{G}_4^* = 1] = \Pr[\mathsf{G}_5^* = 1] = \Pr[\mathsf{G}_6^* = 1]\ . \tag{7.10}$$

In the end, we use a *complexity leveraging* argument to conclude that thanks to (7.10), we have $\Pr[\mathsf{G}_3 = 1] = \Pr[\mathsf{G}_4 = 1] = \Pr[\mathsf{G}_5 = 1] = \Pr[\mathsf{G}_6 = 1]$.

Temporarily, the index i is put back to emphasize that $j_i, \tilde{j}_i$ might differ among different i. For the sequence $\mathsf{G}_3 \to \mathsf{G}_6$, we make a guess for the values $(\mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)})$ with $j_i \in [J_i]$, $\tilde{j}_i \in [\tilde{J}_i]$, $i \in [H]$, choose $r' \xleftarrow{\$} \mathbb{Z}_q^*$, random secret sharings $(\tau_i, \tau'_{k,i}, \tilde{\tau}_i, \tilde{\tau}'_{k,i})_{i=1}^H$ of 0 for each $j_i \in [J_i]$, with $\tau_i, \tau'_{k,i} \neq 0$ for all i. We define the event E that the guess is correct on $(\mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)})_{i\in[H]}^{j_i\in[J_i],\tilde{j}_i\in[\tilde{J}_i]}$ and for any $j_i \in [J_i]$

$$\tilde{\tau}_i - \tau_i = \frac{1}{r'} \sum_{\tilde{j}_i \in [\tilde{J}_i]} \left(\langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle \right)$$
$$\tilde{\tau}'_{k,i} - \tau'_{k,i} = \frac{1}{r'} \sum_{\tilde{j}_i \in [\tilde{J}_i]} \left(\langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(\mathsf{rep})} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(\mathsf{rep})} \rangle \right) \ .$$

Before elaborating the games, we start by showing an important property. The admissibility condition 2 in Definition 7.2 gives $\sum_{i=1}^H \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle = \sum_{i=1}^H \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle$ for any $\tilde{j}_i \in [\tilde{J}_i], j_i \in [J_i]$. Suppose that there exists $\varnothing \neq I' \subseteq [H]$ and $j'_i, j''_i \in [J]$ so that

$$\sum_{i\in I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j'_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j'_i)} \rangle \neq \sum_{i\in I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j''_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j''_i)} \rangle \ ,$$

while

$$\sum_{i\in[H]\setminus I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j'_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j'_i)} \rangle = \sum_{i\in[H]\setminus I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j''_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j''_i)} \rangle \ .$$

Summing both sides give an inequality

$$\begin{aligned} &\sum_{i\in I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j'_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j'_i)} \rangle + \sum_{i\in[H]\setminus I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j'_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j'_i)} \rangle \\ &\neq \sum_{i\in I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j''_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j''_i)} \rangle + \sum_{i\in[H]\setminus I'} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j''_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j''_i)} \rangle \end{aligned}$$

that contradicts condition 2 in Definition 7.2. The same can be argued w.r.t $\mathbf{x}_i^{(\mathsf{rep})}$. Therefore, for each $i \in [H]$, for all $\tilde{j}_i \in [\tilde{J}], j_i \in [J]$, the terms

$$\begin{cases} \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(0,j_i)} \rangle \\ \langle \mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{x}_i^{(\mathsf{rep})} \rangle - \langle \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(\mathsf{rep})} \rangle \end{cases} \tag{7.11}$$

are constants.

From this point, in order to ease the presentation, we omit the index i from $j_i \in [J_i], \tilde{j}_i \in [\tilde{J}_i], \mathbf{d}^{(\tilde{j}_i)}, \mathbf{c}^{(j_i)}$ and write $j \in [J], \tilde{j} \in [\tilde{J}], \mathbf{d}^{(\tilde{j})}, \mathbf{c}^{(j)}$. We describe the selective games below, starting from G_3^*, where event E is assumed true:

Game G_3^*: This is the selective version of G_3, assuming event E is true. For the basis matrices, thanks to the large enough dimension we can define the entries based on all repetitions $(\mathbf{y}_i^{(1,\tilde{j})}[m], \mathbf{y}_i^{(0,\tilde{j})}[m])_m$ where $\tilde{j} \in [\tilde{J}]$. For different $\tilde{j}_1 \neq \tilde{j}_2 \in [\tilde{J}]$, the entries in the matrix that depend on $\mathbf{d}_i^{(\tilde{j}_1)}$ will affect only the $\mathbf{0}$-entries in $\mathbf{d}_i^{(\tilde{j}_2)}$, thus not creating errors, and *vice versa*.

Game G_4^*: Knowing $(\mathbf{y}_i^{(1,\tilde{j})}, \mathbf{y}_i^{(0,\tilde{j})}, \mathbf{x}_i^{(1,j)}, \mathbf{x}_i^{(0,j)})_{i\in[H]}^{j\in[J],\tilde{j}\in[\tilde{J}]}$, we do quotients by B_i defined via

$$B_i := (B_i[row, col]) \tag{7.12}$$

where

$$B_i[row, col] = \begin{cases} 1 & \text{if } row = col \leq 2N+3 \\ \frac{1}{\mathbf{y}_i^{(1,\tilde{j})}[z]} & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + 3 + z \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0 \\ 1 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + 3 + z \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] = 0 \\ \frac{1}{\mathbf{y}_i^{(0,\tilde{j})}[z]} & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + N + 3 + z \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] \neq 0 \\ 1 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + N + 3 + z \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$B_i' \coloneqq \left(B_i^{-1}\right)^\top; \qquad \mathbf{B}_i = B_i \cdot \mathbf{H}_i; \qquad \mathbf{B}_i^* = B_i' \cdot \mathbf{H}_i^* \ .$$

The entries of the matrix take into account several conditions:

- The entries $B_i[row, col] = 1$, if $row = col \leq 2N + 3$, fix the vectors coordinates that we do not want to change in $\mathbf{d}_i^{(\tilde{j})}$.
- For $\tilde{j} \in [J], z \in [N]$, the entries

$$B_i[row, col] = \begin{cases} B_i[row, col] & = \frac{1}{\mathbf{y}_i^{(1,\tilde{j})}[z]} \text{ if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + 3 + z \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0 \\ B_i[row, col] & = 1 \text{ if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + 3 + z \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] = 0 \end{cases}$$

  will change the coordinate $\mathbf{d}_i^{(\tilde{j})}[2N \cdot \tilde{j} + 3 + z]$ into 1 iff $\mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0$. Dually for all $\mathbf{c}$-vectors their coordinate $2N \cdot \tilde{j} + 3 + z$ will be multiply by $\mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0$. An analogous computation is performed by

$$B_i[row, col] = \begin{cases} \frac{1}{\mathbf{y}_i^{(0,\tilde{j})}[z]} & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + N + 3 + z \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] \neq 0 \\ 1 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t.} \\ & \quad row = col = 2N \cdot \tilde{j} + N + 3 + z \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] = 0 \end{cases}$$

  but with respect to $\mathbf{y}_i^{(0,\tilde{j})}$ and for coordinates $2N \cdot \tilde{j} + N + 3 + z$, for $z \in [n]$.

Game G_5^* : In this game we perform a formal basis change to move all the values 1 from coordinates $[2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + N + 3]$ for all to coordinates $[2N \cdot \tilde{j} + N + 4, 2N \cdot \tilde{j} + 2N + 3]$ of $\mathbf{d}_i$. The basis changing matrices B_i from G_4^* to G_5^* is defined below

$$B_i \coloneqq (B_i[row, col]) \tag{7.13}$$

where

$$B_i[row, col] = \begin{cases} 1 & \text{if } row = col \\ \frac{1}{r'} & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + 3 + z) \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0 \\ 0 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + 3 + z) \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] = 0 \\ \frac{-1}{r'} & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + N + 3 + z) \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] \neq 0 \\ 0 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + N + 3 + z) \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$B_i' \coloneqq \left(B_i^{-1}\right)^\top; \qquad \mathbf{B}_i = B_i \cdot \mathbf{H}_i; \qquad \mathbf{B}_i^* = B_i' \cdot \mathbf{H}_i^* \ .$$

From the previous game it holds that, for $\tilde{j} \in [\tilde{J}], z \in [N]$,

$$\begin{aligned} \mathbf{d}_i^{(\tilde{j})}[2N \cdot \tilde{j} + 3 + z] = 1 \text{ iff } \mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0 \\ \mathbf{d}_i^{(\tilde{j})}[2N \cdot \tilde{j} + N + 3 + z] = 0 \quad \forall z \end{aligned} \tag{7.14}$$

while for any $j \in [J]$

$$\begin{aligned} \mathbf{c}_i^{(j)}[2N \cdot \tilde{j} + 3 + z] = \mathbf{y}_i^{(1,\tilde{j})}[z] \cdot \mathbf{x}_i^{(1,j)}[m] \text{ iff } \mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0 \\ \mathbf{c}_i^{(j)}[2N \cdot \tilde{j} + 3 + z] = \mathbf{x}_i^{(1,j)}[m] \text{ iff } \mathbf{y}_i^{(1,\tilde{j})}[z] = 0 \\ \mathbf{c}_i^{(j)}[2N \cdot \tilde{j} + N + 3 + z] = \mathbf{y}_i^{(0,\tilde{j})}[z] \cdot \mathbf{x}_i^{(0,j)}[m] \text{ iff } \mathbf{y}_i^{(0,\tilde{j})}[z] \neq 0 \\ \mathbf{c}_i^{(j)}[2N \cdot \tilde{j} + N + 3 + z] = \mathbf{x}_i^{(0,j)}[m] \text{ iff } \mathbf{y}_i^{(0,\tilde{j})}[z] = 0 \ . \end{aligned}$$

We again recall that the pairs $(\mathbf{x}_i^{(1,j)}, \mathbf{x}_i^{(0,j)})$ (similarly $(\mathbf{x}_{k,i}^{(\mathsf{rep})}, \mathbf{x}_{k,i}^{(\mathsf{rep})})$ in $\mathbf{c}_{k,i}^{(\mathsf{rep})}$ for the same argument) in $\mathbf{c}_i^{(j)}$ are already in position thanks to G_2. The above formal basis change will modify these $\mathbf{d}$-vectors such that: for $\tilde{j} \in [\tilde{J}], z \in [N]$

$$\mathbf{d}_i^{(\tilde{j})}[2N \cdot \tilde{j} + 3 + z] = 0 \, \forall z \in [N] \tag{7.15}$$

$$\mathbf{d}_i^{(\tilde{j})}[2N \cdot \tilde{j} + N + 3 + z] = 1 \text{ iff } \mathbf{y}_i^{(0,\tilde{j})}[z] \neq 0 \tag{7.16}$$

$$\mathbf{d}_i^{(\tilde{j})}[2N \cdot \tilde{j} + N + 3 + z] = 0 \text{ iff } \mathbf{y}_i^{(0,\tilde{j})}[z] = 0 \tag{7.17}$$

where (7.15) comes from

$$B_i[row, col] = \begin{cases} \frac{1}{r'} & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + 3 + z) \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] \neq 0 \\ 0 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + 3 + z) \ \wedge \ \mathbf{y}_i^{(1,\tilde{j})}[z] = 0 \end{cases},$$

(7.16) comes from (7.14) and

$$B_i[row, col] = \begin{cases} \frac{-1}{r'} & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + N + 3 + z) \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] \neq 0 \end{cases},$$

and (7.17) comes again from (7.14) together with

$$B_i[row, col] = \begin{cases} 0 & \text{if } \exists \tilde{j} \in [J], z \in [N] \text{ s.t} \\ & \quad (row, col) = (2N + 2N \cdot \tilde{j} + 4, 2N \cdot \tilde{j} + N + 3 + z) \ \wedge \ \mathbf{y}_i^{(0,\tilde{j})}[z] = 0 \end{cases}.$$

Temporarily until the end of this G_6, the index i is put back to emphasize that $j_i, \tilde{j}_i$ might differ among different i. Accordingly, the $\mathbf{c}$-vectors are changed as follows, for $j \in [J]$,

$$\begin{aligned}\mathbf{c}_i^{(j_i)}[2N\cdot\tilde{j}_i + 2N + 4] = \tau_i + \frac{1}{r'}\sum_{\tilde{j}_i\in[\tilde{J}_i]}\Bigg(&\sum_{\substack{z\in[N]\\ \text{cond E3}}} \mathbf{y}_i^{(1,\tilde{j}_i)}[z]\cdot\mathbf{x}_i^{(1,j_i)}[z] - \mathbf{y}_i^{(0,\tilde{j}_i)}[z]\cdot\mathbf{x}_i^{(0,j_i)}[z]\\ &+ \sum_{\substack{z\in[N]\\ \text{cond E2}}} \mathbf{y}_i^{(1,\tilde{j}_i)}[z]\cdot\mathbf{x}_i^{(1,j_i)}[z] - \sum_{\substack{z\in[N]\\ \text{cond E1}}} \mathbf{y}_i^{(0,\tilde{j}_i)}[z]\cdot\mathbf{x}_i^{(0,j_i)}[z]\Bigg)\\ = \tau_i + \frac{1}{r'}\sum_{\tilde{j}_i\in[\tilde{J}_i]}&\left(\langle\mathbf{y}_i^{(1,\tilde{j}_i)},\mathbf{x}_i^{(1,j_i)}\rangle - \langle\mathbf{y}_i^{(0,\tilde{j}_i)},\mathbf{x}_i^{(0,j_i)}\rangle\right)\ ,\end{aligned} \tag{7.18}$$

where the conditions are

(E3) $\mathbf{y}_i^{(1,\tilde{j}_i)}[z] \neq 0 \ \wedge\ \mathbf{y}_i^{(0,\tilde{j}_i)}[z] \neq 0$,
(E2) $\mathbf{y}_i^{(1,\tilde{j}_i)}[z] \neq 0 \ \wedge\ \mathbf{y}_i^{(0,\tilde{j}_i)}[z] = 0$ and
(E1) $\mathbf{y}_i^{(1,\tilde{j}_i)}[z] = 0 \ \wedge\ \mathbf{y}_i^{(0,\tilde{j}_i)}[z] \neq 0$.

This formal swapping will modify the secret sharings τ_i, in a given $\mathbf{c}_i^{(j_i)}$, into another secret sharing of 0 for any fixed $\tilde{j}_i, j_i$ thanks to condition

$$\sum_{i=1}^{H}\langle\mathbf{y}_i^{(1,\tilde{j}_i)},\mathbf{x}_i^{(1,j_i)}\rangle = \sum_{i=1}^{H}\langle\mathbf{y}_i^{(0,\tilde{j}_i)},\mathbf{x}_i^{(0,j_i)}\rangle\ .$$

Moreover the updated τ_i does not depend on $\tilde{j}_i, j_i$ because $\langle\mathbf{y}_i^{(1,\tilde{j}_i)},\mathbf{x}_i^{(1,j_i)}\rangle - \langle\mathbf{y}_i^{(0,\tilde{j}_i)},\mathbf{x}_i^{(0,j_i)}\rangle$ is constant for all $\tilde{j}_i \in [\tilde{J}_i], j_i \in [J_i]$, for any fixed $i \in [H]$, thanks to the observation (7.11). In the vectors $\mathbf{c}_{k,i}^{(\mathsf{rep})}$ a similar argument can be done, because the difference being added to $\tau'_{k,i}$ is $\frac{1}{r'}\sum_{\tilde{j}_i\in[\tilde{J}_i]}\left(\langle\mathbf{y}_i^{(1,\tilde{j}_i)},\mathbf{x}_i^{(\mathsf{rep})}\rangle - \langle\mathbf{y}_i^{(0,\tilde{j}_i)},\mathbf{x}_i^{(\mathsf{rep})}\rangle\right)$ together with the hypothesis

$$\sum_{i=1}^{H}\langle\mathbf{y}_i^{(1,\tilde{j}_i)},\mathbf{x}_i^{(\mathsf{rep})}\rangle = \sum_{i=1}^{H}\langle\mathbf{y}_i^{(0,\tilde{j}_i)},\mathbf{x}_i^{(\mathsf{rep})}\rangle\ .$$

We will use again the observation (7.11) to conclude that the difference does not depend on repetitions.

Game G_6^* : The game hop from G_5^* to G_6^* to undo these quotients can be defined similarly as we have done from $\mathsf{G}_3^* \to \mathsf{G}_4^*$, in order to multiply back $\mathbf{y}_i^{(0,\tilde{j}_i)}$ into coordinates $[2N\cdot\tilde{j}_i + N + 4, 2N\cdot\tilde{j}_i + 2N + 3]$.

The above games demonstrate relation (7.10). We now employ the complexity leveraging argument. Let us fix $t \in \{3,4,5\}$. For $u \in \{t, t+1\}$ let $\mathsf{Adv}_u(\mathcal{A}) := |\Pr[\mathsf{G}_u(\mathcal{A}) = 1] - 1/2|$ denote the advantage of a ppt adversary $\mathcal{A}$ in game G_u. We build a ppt adversary $\mathcal{B}^*$ playing against G_u^* such that its advantage $\mathsf{Adv}_u^*(\mathcal{B}^*) := |\Pr[\mathsf{G}_u^*(\mathcal{B}^*) = 1] - 1/2|$ equals $\gamma\cdot\mathsf{Adv}_u(\mathcal{A})$ for $u \in \{t, t+1\}$, for some constant γ.

The adversary $\mathcal{B}^*$ first guesses the values $(\mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)})$ with $j_i \in [J_i]$, $\tilde{j}_i \in [\tilde{J}_i]$, $i \in [H]$, choose $r' \xleftarrow{\$} \mathbb{Z}_q^*$, random secret sharings $(\tau_i, \tau'_{k,i}, \tilde{\tau}_i, \tilde{\tau}'_{k,i})_{i=1}^H$ of 0 for each $j_i \in [J_i]$, with $\tau_i, \tau'_{k,i} \neq 0$ for all i. Then $\mathcal{B}^*$ defines the event E that the guess is correct on $(\mathbf{y}_i^{(1,\tilde{j}_i)}, \mathbf{y}_i^{(0,\tilde{j}_i)}, \mathbf{x}_i^{(1,j_i)}, \mathbf{x}_i^{(0,j_i)})_{i\in[H]}^{j_i\in[J_i],\tilde{j}_i\in[\tilde{J}_i]}$ and for any $j_i \in [J_i]$

$$\begin{aligned}\tilde{\tau}_i - \tau_i &= \frac{1}{r'}\sum_{\tilde{j}_i\in[\tilde{J}_i]}\left(\langle\mathbf{y}_i^{(1,\tilde{j}_i)},\mathbf{x}_i^{(1,j_i)}\rangle - \langle\mathbf{y}_i^{(0,\tilde{j}_i)},\mathbf{x}_i^{(0,j_i)}\rangle\right)\\ \tilde{\tau}'_{k,i} - \tau'_{k,i} &= \frac{1}{r'}\sum_{\tilde{j}_i\in[\tilde{J}_i]}\left(\langle\mathbf{y}_i^{(1,\tilde{j}_i)},\mathbf{x}_i^{(\mathsf{rep})}\rangle - \langle\mathbf{y}_i^{(0,\tilde{j}_i)},\mathbf{x}_i^{(\mathsf{rep})}\rangle\right)\ .\end{aligned}$$

When $\mathcal{B}^*$ guesses successfully and E happens, then the simulation of $\mathcal{A}$'s view in G_t is perfect. Otherwise, $\mathcal{B}^*$ aborts the simulation and outputs a random bit b'. Since E happens with some fixed probability γ and is independent from the view of $\mathcal{A}$, we have[6]:

$$\begin{aligned}
\mathsf{Adv}^*_u(\mathcal{B}^*) &= \left|\Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1] - \frac{1}{2}\right| \\
&= \left|\Pr[E] \cdot \Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1 \mid E] + \frac{\Pr[\neg E]}{2} - \frac{1}{2}\right| \\
&= \left|\gamma \cdot \Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1 \mid E] + \frac{1-\gamma-1}{2}\right| \\
&\overset{(*)}{=} \gamma \cdot \left|\Pr[\mathsf{G}_u(\mathcal{A}) = 1] - \frac{1}{2}\right| = \gamma \cdot \mathsf{Adv}_u(\mathcal{A})
\end{aligned} \tag{7.19}$$

where $(*)$ comes from the fact that conditioned on E, $\mathcal{B}$ simulates perfectly G_u for $\mathcal{A}$, therefore $\Pr[\mathsf{G}_u(\mathcal{A}) = 1 \mid E] = \Pr[\mathsf{G}^*_u(\mathcal{B}^*) = 1 \mid E]$, then we apply the independence between E and $\mathsf{G}_u(\mathcal{A}) = 1$. Together with relation (7.10), this concludes that $\Pr[\mathsf{G}_t = 1] = \Pr[\mathsf{G}_{t+1} = 1]$ for any fixed $t \in \{3, 4, 5\}$, in particular $\Pr[\mathsf{G}_6 = 1] = \Pr[\mathsf{G}_3 = 1]$.

Game G_7: Similar to the transition G_2 to G_3, we use a computational swap between $[N+1, 2N]$ and $[2N \cdot \tilde{j} + N + 4, 2N \cdot \tilde{j} + 2N + 3]$ in $\mathbf{d}_i^{(\tilde{j})}$ using $(2N+3)$-randomness. We need $n \cdot \tilde{J} \cdot N$ DSDH instances $(\llbracket a_{i,z}^{(\tilde{j})} \rrbracket_1, \llbracket b_{i,z}^{(\tilde{j})} \rrbracket_1, \llbracket c_{i,z}^{(\tilde{j})} \rrbracket_1)$ in $\mathbb{G}_1$ where $\delta_{i,z}^{(\tilde{j})} := c_{i,z}^{(\tilde{j})} - a_{i,z}^{(\tilde{j})} b_{i,z}^{(\tilde{j})}$ is either 0 or $\mathbf{x}_i^{(\tilde{j})}[z]$, for $z \in [N], \tilde{j} \in [\tilde{J}], i \in [n]$. The security loss is $2 \cdot n \cdot \tilde{J} \cdot N \cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$.

Game G_8: We redo the transitions from G_1 to G_2 to clean the vectors.

After arriving at G_8, the vectors are computed following the interaction

$$\mathcal{A}^{\tilde{\mathcal{O}}_{\mathbf{u}}, \boxed{\mathcal{O}^1_{\mathbf{u}}}}_{\tilde{\mathcal{O}}_{\mathbf{v}}, \mathcal{O}_{\mathbf{v}}}\Big(1^\lambda, N, H, K, L, (J_i, \widetilde{J}_i)_{i\in[H]}, R, (R_k)_{k\in[K]}\Big) \ ,$$

the transitions are indistinguishable under SXDH, and the proof is finished.

7.4.2 Basic Construction

This section presents our basic adaptively secure $\mathsf{FH\text{-}DMCFE}$ construction $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for the function class $\mathcal{F}^{\mathsf{ip}}$, where each client encrypts a vector of length $N \in \mathbb{N}$. As a reminder, we refer to the beginning of Sections 7.3 as well as Chapter 3 for the notations, inlcuding those of implicit represention for group elements and the bilinear group setting. The notations of DPVS and the writing of their vectors with respect to the dual bases are recalled in Section 3.3. Our adaptively secure scheme's details are given in Figure 7.3.

Correctness. The correctness property is demonstrated as follows:

$$[\![\mathsf{out}]\!]_{\mathsf{t}} = \sum_{i=1}^{n} \mathbf{c}_i \times \mathbf{d}_i = \sum_{i=1}^{n} [\![\langle \mathbf{x}_i, \mathbf{y}_i \rangle + \mu\omega \cdot \tilde{t}_i]\!]_{\mathsf{t}} = \left[\!\!\left[\sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle + \mu\omega \cdot \sum_{i=1}^{n} \tilde{t}_i \right]\!\!\right]_{\mathsf{t}} = \left[\!\!\left[\sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle \right]\!\!\right]_{\mathsf{t}} ,$$

and we are using the fact that $\sum_{i=1}^{n} \tilde{t}_i = 0$.

[6] This calculation (6.9) to relate $\mathsf{Adv}^*_u(\mathcal{B}^*)$ to $\mathsf{Adv}_u(\mathcal{A})$ is the core of our complexity levaraging argument, being built upon the previous information-theoretic game transtions and the probability of event E.

$\mathsf{Setup}(1^\lambda, 1^n)$: Sample matrices $(B_i, B_i^*) \leftarrow \mathsf{DPVSGen}(\mathbf{G}, 1^{2N\cdot(\tilde{J}+1)+4})$, for $i \in [n]$, of dimensions $2N \cdot (\tilde{J}+1)+4$ that specify dual orthogonal bases $(\mathbf{B}_i, \mathbf{B}_i^*)$[a]. Sample $(\tilde{t}_i)_i \xleftarrow{\$} \mathbb{Z}_q^n$ conditioned on $\sum_{i=1}^n \tilde{t}_i = 0$. Output the public parameters $\mathsf{pp} := \mathbf{G}$, secret keys sk_i and the encryption keys ek_i for all $i \in [n]$ as follows:

$$\mathsf{sk}_i := (\mathbf{b}^*_{i,1}, \ldots, \mathbf{b}^*_{i,N},\ B^*_{i,N+1},\ \mathbf{b}^*_{i,N+2}), \qquad \mathsf{ek}_i := \big(\tilde{t}_i, (\mathbf{b}_{i,1}, \ldots, \mathbf{b}_{i,N},\ B_{i,N+1},\ \mathbf{b}_{i,N+3})\big)$$

$\mathsf{DKeyGen}(\mathsf{sk}_i, \mathsf{tag\text{-}f}, \mathbf{y}_i)$: Parse $\mathsf{sk}_i = (\mathbf{b}^*_{i,1}, \ldots, \mathbf{b}^*_{i,N},\ B^*_{i,N+1},\ \mathbf{b}^*_{i,N+2})$, compute $\mathsf{H}_2(\mathsf{tag\text{-}f}) \to [\![\mu]\!]_2$ and sample $\pi_i \xleftarrow{\$} \mathbb{Z}_q$. Then output[b]

$$\mathbf{d}_i = \sum_{k=1}^{N} \mathbf{y}_i[k]\mathbf{b}^*_{i,k} + [\![\mu]\!]_2 \cdot B^*_{i,N+1} + \pi_i \mathbf{b}^*_{i,N+2} = (\mathbf{y}_i,\ \mu,\ \pi_i,\ 0,\ 0^{N+2N\cdot\tilde{J}+1})_{\mathbf{B}_i^*} \ .$$

$\mathsf{Enc}(\mathsf{ek}_i, \mathsf{tag}, \mathbf{x}_i)$: Parse $\mathsf{ek}_i = (\tilde{t}_i, (\mathbf{b}_{i,1}, \ldots, \mathbf{b}_{i,N},\ B_{i,N+1},\ \mathbf{b}_{i,N+3}))$, compute $\mathsf{H}_1(\mathsf{tag}) \to [\![\omega]\!]_1$ and sample a random scalar $\rho_i \xleftarrow{\$} \mathbb{Z}_q$. Then output[c]

$$\mathbf{c}_i = \sum_{k=1}^{N} \mathbf{x}_i[k]\mathbf{b}_{i,1} + \tilde{t}_i\,[\![\omega]\!]_1 \cdot B_{i,N+1} + \rho_i \mathbf{b}_{i,N+3} = (\mathbf{x}_i,\ \tilde{t}_i\omega,\ 0,\ \rho_i,\ 0^{N+2N\cdot\tilde{J}+1})_{\mathbf{B}_i} \ .$$

$\mathsf{Dec}(\mathbf{d}, \mathbf{c})$: Parse $\mathbf{d} := (\mathbf{d}_i)_{i\in[n]}$ and $\mathbf{c} := (\mathbf{c}_i)_i$. Compute $[\![\mathsf{out}]\!]_\mathsf{t} = \sum_{i=1}^n \mathbf{c}_i \times \mathbf{d}_i$, then find and output the discrete log out.

[a] For each $i \in [n]$, we denote j-th row of $\mathbf{B}_i$ (resp. $\mathbf{B}_i^*$) by $\mathbf{b}_{i,j}$ (resp. $\mathbf{b}^*_{i,j}$). Similarly, $B_{i,k}$ (respectively $B^*_{i,k}$) denotes the k-th row of the basis changing matrix B_i (respectively B_i^*).

[b] Throughout the computation of $\mathbf{d}_i$, only the hash value $[\![\mu]\!]_2 \in \mathbb{G}_2$ is used, never μ in the clear.

[c] Throughout the computation of $\mathbf{c}_i$, only the hash value $[\![\omega]\!]_1 \in \mathbb{G}_1$ is used, never ω in the clear.

Figure 7.3: FH-DMCFE scheme $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for inner products. We work in the prime-order bilinear group setting $\mathbf{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, g_\mathsf{t}, \mathbf{e}, q)$ and use two full-domain hash functions $\mathsf{H}_1 : \mathsf{Tag} \to \mathbb{G}_1$ and $\mathsf{H}_2 : \mathsf{Tag} \to \mathbb{G}_2$. Let $\tilde{J} = \mathsf{poly}(\lambda)$.

Security. Theorem 7.8 states that the scheme given in Fig. 7.3 is *function-hiding*, *one-challenge* secure against *complete queries* under *static corruption*. An unbounded number of ciphertext repetitions is allowed, while the number of key repetitions is fixed as a parameter of the scheme. In Section 7.4.3, we argue that most restrictions on the security model can be removed by applying a sequence of generic lemmas.

Theorem 7.8. *The DMCFE scheme* $\mathcal{E} = (\mathsf{Setup}, \mathsf{DKeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *in Fig. 7.3 for the function class* $\mathcal{F}^{\mathsf{ip}}$ *is* one-challenge, function-hiding *secure against* complete *queries under* static *corruption in the ROM, if the* SXDH *assumption holds for* $(\mathbb{G}_1, \mathbb{G}_2)$.

More specifically, we let q_e *and* q_k *denote the maximum number of distinct tags queried to* **Enc** *and* **KeyGen**, *respectively. Furthermore, for* $i \in [n]$ *and* $\mathsf{tag}, \mathsf{tag\text{-}f} \in \mathsf{Tag}$, *we define* $\tilde{J}_{i,\mathsf{tag\text{-}f}}$ *to be the numbers of queries of the form* $\mathbf{KeyGen}(i, \mathsf{tag\text{-}f}, \star, \star)$. *We require that* $\max_{i\in[n],\mathsf{tag\text{-}f}\in\mathsf{Tag}} \tilde{J}_{i,\mathsf{tag\text{-}f}} \le \tilde{J}$, *where* $\tilde{J}$ *is specified by the DMCFE scheme at* Setup *time. Then, for any* ppt *adversary* $\mathcal{A}$ *against* $\mathcal{E}$, *we have the following bound:*

$$\mathsf{Adv}^{\mathsf{1chal\text{-}pos\text{-}statfh}}_{\mathcal{E},\mathcal{F}^{\mathsf{ip}},\mathcal{A}}(1^\lambda) \le \Big((q_k+1)\cdot(4n\tilde{J}N+4)+4N+q_e+1\Big)\cdot \mathsf{Adv}^{\mathsf{SXDH}}_{\mathbb{G}_1,\mathbb{G}_2}(1^\lambda)$$

The proof of Theorem 7.8 follows closely the proof sketch of the selective scheme in Section 7.2. As explained in the paragraph **Problems for Adaptive Security**, the main difficulty towards adaptive security lies in enabling the steps (7.3) to (7.5) in a sequence of hybrids without knowing $\Delta_i^{(b)}$ and $\Delta_{\ell,i}^{(b)}$ in advance. In the DPVS setting, the transition from one hybrid to the next corresponds exactly to an application of Lemma 7.7. Even though $\tilde{J}$ is fixed, it can

be polynomially large leading to an exponentially number of combinations of key repetitions, this is also handled by Lemma 7.7. We refer to the high level in section 7.4.1. The full proof of theorem 7.8 can be found in [NPS24b, Appendix A.3].

7.4.3 Upgrading Security

Given Lemmas 7.4, 7.5, and 7.6, we now generically transform our FH-DMCFE from Section 7.4.2 to upgrade its security. Specifically, we first apply Lemma 7.4 and follow the generic IBE-based AoNE from [CDSG+20, Section 4]. We use any adaptively secure pairing-based IBE [CLL+13, JR17] under SXDH[7] to obtain generically a DMCFE for AoNE, in order to allow *incomplete queries*. We then use Lemma 7.5 to allow *multiple challenges*, while downgrading from function-hiding to weak function-hiding. Finally, we apply Lemma 7.6 to re-establish full-fledged *function-hiding*. The final scheme is summarized in the below corollary, with newly accomplished properties being *emphasized*.

Corollary 7.9. *There exists an FH-DMCFE scheme for the function class $\mathcal{F}^{\mathsf{ip}}$ that is* adaptively *function-hiding secure against static corruption, while allowing unbounded repetitions for ciphertext queries and* a fixed polynomially large number of repetitions for key-generation queries*, under the* SXDH *assumption in the ROM.*

[7]The seminal adaptively secure group-based (H)IBE is [Wat09] but it relies on both DDH and D-Lin.

Part IV

Conclusion and Future Works

Chapter

8 Conclusion

A Recap. As the thesis draws its conclusion, we recall here the research questions that emerge during the preparation of this thesis:

1. Following the introduction of (D)MCFE in the seminal paper [CDG+18a], all follow-up studies on (D)MCFE, for instance [CDG+18b, ABKW19, ABG19, LT19, CDSG+20, AGT21b], administered an *admissibility condition* and restricted particularly adversaries to asking the challenge components $\mathbf{x}_i^{(0)} = \mathbf{x}_i^{(1)}$ in case of a corrupted i. *Can we relax this constraint, which will lead to a stronger security model for (D)MCFE, as hinted in [CDG+18a]?*

2. All cited DDFE schemes [CDSG+20, AGT21b, ATY23a, Ngu24] attain only *selective* security under *static* corruption. *How far can we push for adaptive security of DDFE?*

3. Regarding the attribute-based access control over functional keys, existing works can go as far as MIFE, for inner products in [ACGU20] and for attribute-weighted sums in [ATY23a]. *How further can we integrate access control into the multi-user setting,* e.g. *starting from MCFE and potentially to DDFE?*

4. All cited DDFE schemes [CDSG+20, AGT21b, ATY23a, Ngu24] rely on group-based assumptions and do not provide post-quantum security. The only multi-user FE scheme in the post-quantum regime comes from the DMCFE of [LT19] for inner products and relies on the *Learning with Error* (LWE) assumption. *How far can we push for post-quantum security for DDFE?*

5. The security against repetitions on key tags is either excluded in the FH-IP-DDFE from [AGT21b, Ngu24], or not explicitly considered in [CDSG+20] for IP-DDFE and in [ATY23a, Ngu24] for AWS-DDFE. *How can we achieve security against repetitions for both encryption and key-generation queries in the FH-DDFE and/or DDFE with access control setting?*

The main first two chapters of this thesis include Chapter 4 that resolves partially question 3 at the level of MCFE, meanwhile Chapter 5 resolves question 1 at the level of (D)MCFE. Combiningly, Chapter 6 improves the resolution of question 3, while integrating the resolution of question 1 in the access control context, at the level of MCFE. These current results fall short of giving full answers for question 5 at the level of MCFE, with strong admissibility and access control.

The later chapter of the thesis includes Chapter 7 that resolves question 2 of adaptive security for FH-secure at the level of DMCFE, and partially question 5 of repetitions for FH-security at the level of DMCFE. In a submission that is not presented in this thesis [NPS25], our results provide an affirmative answer to questions 2, 4 and 5 in the case of IP-DDFE. Moving on to FH-IP-DDFE, the work of [NPS25] also provides an affirmative answer to question 2 and partially resolves question 5. Finally, regarding DDFE for AB-AWS, the foregoing work thereby resolves question 3 and partially question 2.

Independently, other research directions of FE in the multi-user setting are explored in ongoing works, including CCA security for MCFE [NPP24] or traceability in MCFE [DMN+24].

Future Works. The almost obvious remaining work to continue concerns the above questions, those that are partially resolved, those that are only up to (D)MCFE and not yet DDFE, those that are not yet FH, or those that are only to compute inner products and not yet AWS. Needless to say, more unexpected questions may arise, which do not relate to any of the above. At the time of writing this thesis, there are a few examples of such unexpected questions:

1. *(From MIFE to MCFE)* The cornerstone of this thesis is MCFE and its generalization. It is known that under a strong security guarantee, *i.e.* semantic security holds even when an adversary can obtain different ciphertexts on the same (i, tag), one can obtain a MIFE for the same function class from a MCFE. About the other direction, it is worth recalling that the work [ACGU20] hints at going to MCFE from MIFE with access control: tags can be used as specific attributes, and tags can be embedded in policies to automatically obtain multi-client settings. As we have elaborated in Chapter 4, this argument seems formally valid when considering the general form of MIFE and MCFE. However, when considering concrete classes of functions, it is unlikely to be efficiently feasible.

   For instance, considering the class to compute inner products, an attempt to go all the way from a single client IPFE to MIFE for inner products resembles the below:

$$\mathsf{IPFE} \xrightarrow{(1)} \text{secret-key } \mathsf{IPFE} \xrightarrow{(2)} \mathsf{IP\text{-}MIFE} \xrightarrow{(3)} \mathsf{IP\text{-}MIFE} \text{ w/ } \mathsf{Tag} \to ?? \to \mathsf{IP\text{-}MCFE}$$

   where step (1) privatizes the public key of IPFE into some msk for encryption, assuming the keys can be decomposed for encrypting slots i independently[1] step (2) consists of decomposing the msk into multiple encryption keys ek_i, step (3) allows treating tags as simple as relying on the ROM during encryption (for deriving the encryption randomness, for example). Unfortunately, we find ourselves stuck at ?? where the crux is to handle *corruption*, which is an important aspect of MCFE. After step (2), towards MCFE each ek_i will be assigned to a client i. However, the decomposing step (2) seems unlikely to make all these ek_i independent, as they are still linked implicitly as msk and intially share some secret randomness as part of the starting IPFE. Corrupting ek_i leaks the shared secret randomness that connects them all, and breaks the security. There might be a totally different approach to solve the matter, but it stays elusive at the end of this thesis.

2. *(Basing MCFE on Assumptions other than* DDH *and* LWE*)* The main chapters of this thesis do not detail our LWE-based DMCFE/DDFE for inner products from [NPS25]. We recall that our final LWE-based DDFE for inner products is the first IP-DDFE that relies on an assumption other than DDH. One main advantage of LWE is its widely-believed post-quantum security, and preferably one will look also into other widely-believed post-quatum assumptions for the sake of varieties and avoiding "putting all eggs in one basket". In the following, we pay attention to the particular family of code-based assumptions, which includes the *Learning Parity with Noise* (LPN) that recently receives lots of attention. At the time of writing this thesis other candidates are not yet discovered. We speculate one reason may lie in the technique that is used for almost all existing LWE-based (D)MCFE/DDFE, which is introduced in the celebrated work by Libert and Titiu [LT19]. More specifically, it is linked to the way tags are handled in these LWE-based (D)MCFE schemes. In a simplified manner, the public parameters in the MCFE of [LT19] contains Gentry-Sahai-Waters ciphertexts [GSW13] of 0 and 1 that are $(\mathbf{A}_{0,i}, \mathbf{A}_{1,i})_i$. Using a secret encryption key $\mathbf{s}_i$, the encryption of input $\mathbf{x}_i$ under some tag tag results in

$$\mathsf{ct}_i \coloneqq \mathbf{G}_0 \cdot \mathbf{x}_i + \mathbf{A}_{\mathsf{tag}} \cdot \mathbf{s}_i + \mathsf{error}$$

[1] This holds for many IPFE schemes, *e.g.* the famous DDH-based IPFE from [ALS16] has this property.

where $\mathbf{G}_0$ is some gadget matrix that allows decrypting with a trapdoor, and $\mathbf{A}_{\mathsf{tag}}$ is the homomorphic product of GSW ciphertexts $\mathbf{A}_{\mathsf{tag}[i],i}$ that are indexed by the bits $\mathsf{tag}[i]$[2]. Decryption by a key $\sum_i \mathbf{s}_i \cdot y_i$ with respect to a function of parameters $(y_i)_i$ proceeds by computing back $\mathbf{A}_{\mathsf{tag}}$ from tag, then getting

$$\sum_i \mathsf{ct}_i \cdot y_i - \mathbf{A}_{\mathsf{tag}} \cdot \left(\sum_i \mathbf{s}_i \cdot y_i \right) = \mathbf{G}_0 \cdot (\sum_i \mathbf{x}_i \cdot y_i) + \widetilde{\mathsf{error}}$$

and the trapdoor of $\mathbf{G}_0$ allows decrypting to $\sum_i \mathbf{x}_i \cdot y_i$. Since to our knowledge we do not know analogues of homomorphic techniques for code-based cryptography, the above blueprint from [LT19] does not seem to generalize immediately to the code-based setting. It might be worth mentionning that in the code-based cryptography setting, coming up with new techniques for handling tags is not only beneficial to the domain of (D)MCFE but also to other domains whose primitives may need to deal with tags, *e.g.* some advanced tag-based version of *lossy trapdoor functions*[3] for which the recent groundbreaking work on code-based lossy cryptography [DJ24] also leaves as future works.

[2]More precisely, the tag is processed by an admissible hash function before being used for the homomorphic GSW evaluation.

[3]These tag-based versions can be found in [LSSS17, BL17, LNP22] and they also rely on GSW-style homomorphic evaluation, or if not on techniques from preimage sampling [GPV08, MP12] that is also not known in code-based cryptography.

Bibliography

[ABB+13] Michel Abdalla, Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, and David Pointcheval. SPHF-friendly non-interactive commitments. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 214–234. Springer, Heidelberg, December 2013. 25

[ABDP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 733–751. Springer, Heidelberg, March / April 2015. 4, 10

[ABDP16] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Better security for functional encryption for inner product evaluations. Cryptology ePrint Archive, Report 2016/011, 2016. `https://eprint.iacr.org/2016/011`. 40

[ABG19] Michel Abdalla, Fabrice Benhamouda, and Romain Gay. From single-input to multi-client inner-product functional encryption. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 552–582. Springer, Heidelberg, December 2019. 4, 10, 12, 13, 14, 18, 19, 62, 64, 65, 88, 116, 122, 145

[ABKW19] Michel Abdalla, Fabrice Benhamouda, Markulf Kohlweiss, and Hendrik Waldner. Decentralizing inner-product functional encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 128–157. Springer, Heidelberg, April 2019. 4, 10, 12, 13, 14, 19, 62, 64, 65, 88, 116, 122, 145

[ABN10] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 480–497. Springer, Heidelberg, February 2010. 70

[ABP+17] Shweta Agrawal, Sanjay Bhattacherjee, Duong Hieu Phan, Damien Stehlé, and Shota Yamada. Efficient public trace and revoke from standard assumptions: Extended abstract. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 2277–2293. ACM Press, October / November 2017. 18, 68

[ACF+18] Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 597–627. Springer, Heidelberg, August 2018. 12, 22, 94, 96, 97, 126

[ACGU20] Michel Abdalla, Dario Catalano, Romain Gay, and Bogdan Ursu. Inner-product functional encryption with fine-grained access control. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*,

pages 467–497. Springer, Heidelberg, December 2020. 6, 12, 14, 15, 16, 34, 40, 41, 47, 52, 53, 97, 100, 145, 146

[AGRW17] Michel Abdalla, Romain Gay, Mariana Raykova, and Hoeteck Wee. Multi-input inner-product functional encryption from pairings. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 601–626. Springer, Heidelberg, April / May 2017. 64, 94, 96, 97

[AGT21a] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-input quadratic functional encryption from pairings. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 208–238, Virtual Event, August 2021. Springer, Heidelberg. 6, 12, 62, 94, 96, 97, 117

[AGT21b] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-party functional encryption. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part II*, volume 13043 of *LNCS*, pages 224–255. Springer, Heidelberg, November 2021. 4, 10, 13, 14, 15, 17, 19, 21, 22, 23, 62, 65, 88, 122, 124, 145

[AGT21c] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-party functional encryption. In *Theory of Cryptography*. Springer International Publishing, 2021. 93

[AGT22] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. Multi-input quadratic functional encryption: Stronger security, broader functionality. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 711–740. Springer, Heidelberg, November 2022. 6, 12, 34, 62, 88, 94, 96, 97, 117

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 308–326. Springer, Heidelberg, August 2015. 4, 10, 94, 96, 97

[AKM+22] Shweta Agrawal, Fuyuki Kitagawa, Anuja Modi, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Bounded functional encryption for turing machines: Adaptive security from general assumptions. Cryptology ePrint Archive, Report 2022/316, 2022. `https://ia.cr/2022/316`. 16

[ALdP11] Nuttapong Attrapadung, Benoît Libert, and Elie de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 90–108. Springer, Heidelberg, March 2011. 4, 10

[ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, Heidelberg, August 2016. 4, 10, 40, 43, 66, 146

[AMVY21] Shweta Agrawal, Monosij Maitra, Narasimha Sai Vempati, and Shota Yamada. Functional encryption for Turing machines with dynamic bounded collusion from LWE. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 239–269, Virtual Event, August 2021. Springer, Heidelberg. 16

[AS17] Prabhanjan Ananth and Amit Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*,

volume 10210 of *LNCS*, pages 152–181. Springer, Heidelberg, April / May 2017. 4, 10, 40

[ATY23a] Shweta Agrawal, Junichi Tomida, and Anshu Yadav. Attribute-based multi-input FE (and more) for attribute-weighted sums. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 464–497. Springer, Heidelberg, August 2023. 6, 12, 13, 14, 15, 20, 21, 47, 59, 88, 90, 91, 92, 97, 114, 116, 145

[ATY23b] Shweta Agrawal, Junichi Tomida, and Anshu Yadav. Attribute-based multi-input fe (and more) for attribute-weighted sums. In *Advances in Cryptology - IACR CRYPTO 2023*. Springer-Verlag, 2023. `https://eprint.iacr.org/2023/1191`. 94

[BBL17] Fabrice Benhamouda, Florian Bourse, and Helger Lipmaa. CCA-secure inner-product functional encryption from projective hash functions. In Serge Fehr, editor, *PKC 2017, Part II*, volume 10175 of *LNCS*, pages 36–66. Springer, Heidelberg, March 2017. 4, 10, 24, 40

[BCFG17] Carmen Elisabetta Zaira Baltico, Dario Catalano, Dario Fiore, and Romain Gay. Practical functional encryption for quadratic functions with applications to predicate encryption. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 67–98. Springer, Heidelberg, August 2017. 4, 10, 40

[Bei96] Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Technion - Israel Institute of Technology, Haifa, Israel, June 1996. `https://www.cs.bgu.ac.il/~beimel/Papers/thesis.pdf`. 31

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001. 4, 10, 62

[BGH07] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryption without pairings. In *48th FOCS*, pages 647–657. IEEE Computer Society Press, October 2007. 4, 10, 62

[BJK15] Allison Bishop, Abhishek Jain, and Lucas Kowalczyk. Function-hiding inner product encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 470–491. Springer, Heidelberg, November / December 2015. 22

[BL90] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 27–35. Springer, Heidelberg, August 1990. 32

[BL17] Xavier Boyen and Qinyi Li. All-but-many lossy trapdoor functions from lattices and applications. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 298–331. Springer, Heidelberg, August 2017. 147

[BM84] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984. 4, 10

[BO13] Mihir Bellare and Adam O'Neill. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. In Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab, editors, *CANS 13*, volume 8257 of *LNCS*, pages 218–234. Springer, Heidelberg, November 2013. 70

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006. 122

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011. 4, 10, 12, 19, 21, 62, 88, 89, 93, 97

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In Venkatesan Guruswami, editor, *56th FOCS*, pages 171–190. IEEE Computer Society Press, October 2015. 4, 10

[CDG+18a] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Decentralized multi-client functional encryption for inner product. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 703–732. Springer, Heidelberg, December 2018. 12, 13, 14, 16, 17, 18, 19, 21, 24, 40, 52, 53, 62, 64, 65, 66, 67, 69, 70, 88, 89, 90, 93, 94, 95, 99, 116, 122, 145

[CDG+18b] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Multi-client functional encryption with repetition for inner product. Cryptology ePrint Archive, Report 2018/1021, 2018. `https://eprint.iacr.org/2018/1021`. 4, 10, 12, 14, 59, 62, 64, 65, 88, 94, 145

[CDSG+20] Jérémy Chotard, Edouard Dufour-Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Dynamic decentralized functional encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 747–775. Springer, Heidelberg, August 2020. 4, 10, 12, 13, 14, 15, 18, 19, 23, 53, 62, 64, 65, 67, 69, 70, 81, 83, 88, 94, 122, 124, 125, 142, 145

[CLL+13] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter IBE and signatures via asymmetric pairings. In Michel Abdalla and Tanja Lange, editors, *PAIRING 2012*, volume 7708 of *LNCS*, pages 122–140. Springer, Heidelberg, May 2013. 142

[CLT18] Guilhem Castagnos, Fabien Laguillaumie, and Ida Tucker. Practical fully secure unrestricted inner product functional encryption modulo p. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 733–764. Springer, Heidelberg, December 2018. 4, 10, 40

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *8th IMA International Conference on Cryptography and Coding*, volume 2260 of *LNCS*, pages 360–363. Springer, Heidelberg, December 2001. 4, 10, 62

[DDM16] Pratish Datta, Ratna Dutta, and Sourav Mukhopadhyay. Functional encryption for inner product with full function privacy. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 164–195. Springer, Heidelberg, March 2016. 22

[DDM17] Pratish Datta, Ratna Dutta, and Sourav Mukhopadhyay. Strongly full-hiding inner product encryption. *Theoretical Computer Science*, 2017. 22

[DJ24] Quang Dao and Aayush Jain. Lossy cryptography from code-based assumptions. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024*, Cham, 2024. Springer Nature Switzerland. 147

[DMN+24] Thanh Do, Truong Mac, Ky Nguyen, Duong Hieu Phan, and Huy Vu. Traceable multi-client functional encryption, 2024. *Work in wrogress.* 25, 145

[DOT18] Pratish Datta, Tatsuaki Okamoto, and Junichi Tomida. Full-hiding (unbounded) multi-input inner product functional encryption from the k-Linear assumption. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 245–277. Springer, Heidelberg, March 2018. 12, 22, 64, 94, 96, 97

[DP23] Pratish Datta and Tapas Pal. Decentralized multi-authority attribute-based inner-product FE: Large universe and unbounded. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 587–621. Springer, Heidelberg, May 2023. 88

[DPP20] Xuan Thanh Do, Duong Hieu Phan, and David Pointcheval. Traceable inner product functional encryption. In Stanislaw Jarecki, editor, *CT-RSA 2020*, volume 12006 of *LNCS*, pages 564–585. Springer, Heidelberg, February 2020. 25

[dPP22] Paola de Perthuis and David Pointcheval. Two-client inner-product functional encryption with an application to money-laundering detection. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 725–737. ACM Press, November 2022. 12

[EHK+13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, Heidelberg, August 2013. 27

[FWW23] Cody Freitag, Brent Waters, and David J. Wu. How to use (plain) witness encryption: Registered ABE, flexible broadcast, and more. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 498–531. Springer, Heidelberg, August 2023. 6, 11

[Gay20] Romain Gay. A new paradigm for public-key functional encryption for degree-2 polynomials. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 95–120. Springer, Heidelberg, May 2020. 4, 6, 10, 12, 40, 117

[GGG+14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 578–602. Springer, Heidelberg, May 2014. 5, 11, 12, 13, 33, 62, 88, 89, 93, 94, 96, 97, 98

[GKL+13] S. Dov Gordon, Jonathan Katz, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/774, 2013. `https://eprint.iacr.org/2013/774`. 5, 11, 12, 13, 62

[GKW19] Rishab Goyal, Venkata Koppula, and Brent Waters. New approaches to traitor tracing with embedded identities. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 149–179. Springer, Heidelberg, December 2019. 25

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. 4, 10

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988. 4, 10

[GMW86] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract). In *27th FOCS*, pages 174–187. IEEE Computer Society Press, October 1986. 4, 10

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987. 4, 10

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309. 4, 10

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 147

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013. 146

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 503–523. Springer, Heidelberg, August 2015. 4, 10

[JR17] Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. *Journal of Cryptology*, 30(4):1116–1156, October 2017. 142

[KKS19] Sungwook Kim, Jinsu Kim, and Jae Hong Seo. A new approach to practical function-private inner product encryption. *Theoretical Computer Science*, 783:22–40, 2019. 22

[KLM+18] Sam Kim, Kevin Lewi, Avradip Mandal, Hart Montgomery, Arnab Roy, and David J. Wu. Function-hiding inner product encryption is practical. In Dario Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 544–562. Springer, Heidelberg, September 2018. 22

[KNR24] Julia Kastner, Ky Nguyen, and Michael Reichle. Pairing-Free Blind Signatures from Standard Assumptions in the ROM. In *Advances in Cryptology - IACR CRYPTO 2024*, 2024. `https://ia.cr/2023/1810`. 26

[LAKWH22] Fucai Luo, Saif Al-Kuwari, Haiyan Wang, and Weihong Han. Generic construction of trace-and-revoke inner product functional encryption. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *ESORICS 2022, Part I*, volume 13554 of *LNCS*, pages 259–282. Springer, Heidelberg, September 2022. 25

[Lin17] Huijia Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 599–629. Springer, Heidelberg, August 2017. 4, 6, 10, 12, 22, 40, 116

[LLW21] Qiqi Lai, Feng-Hao Liu, and Zhedong Wang. New lattice two-stage sampling technique and its applications to functional encryption - stronger security and smaller ciphertexts. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 498–527. Springer, Heidelberg, October 2021. 16, 20

[LNP22] Benoît Libert, Ky Nguyen, and Alain Passelègue. Cumulatively All-Lossy-But-One Trapdoor Functions from Standard Assumptions. In *SCN 2022 - Proceedings of the 13th Conference on Security in Communication Networks*, September 2022. 26, 147

[LSSS17] Benoît Libert, Amin Sakzad, Damien Stehlé, and Ron Steinfeld. All-but-many lossy trapdoor functions and selective opening chosen-ciphertext security from LWE. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 332–364. Springer, Heidelberg, August 2017. 147

[LT19] Benoît Libert and Radu Titiu. Multi-client functional encryption for linear functions in the standard model from LWE. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 520–551. Springer, Heidelberg, December 2019. 4, 10, 12, 13, 14, 18, 19, 24, 52, 53, 62, 65, 88, 116, 122, 145, 146, 147

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In Irit Dinur, editor, *57th FOCS*, pages 11–20. IEEE Computer Society Press, October 2016. 126

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012. 147

[Ngu24] Duy Nguyen. Dynamic decentralized functional encryptions from pairings in the standard model. Cryptology ePrint Archive, Paper 2024/580, 2024. `https://eprint.iacr.org/2024/580`. 6, 12, 13, 14, 15, 23, 122, 145

[NPP22a] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Multi-client functional encryption with fine-grained access control. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 95–125. Springer, Heidelberg, December 2022. 6, 12, 15, 20, 21, 31, 41, 45, 48, 49, 50, 55, 57, 58, 88, 89, 90, 91, 92, 97, 100, 103

[NPP22b] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Multi-client functional encryption with fine-grained access control. Cryptology ePrint Archive, Report 2022/215, 2022. `https://eprint.iacr.org/2022/215`. 41, 48, 50, 55, 57, 58

[NPP23a] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Optimal security notion for decentralized multi-client functional encryption. In Mehdi Tibouchi and Xiaofeng Wang, editors, *ACNS 23, Part II*, volume 13906 of *LNCS*, pages 336–365. Springer, Heidelberg, June 2023. 4, 10, 13, 16, 17, 19, 20, 21, 63, 71, 73, 74, 88, 89, 90, 91, 94, 103, 122, 156

[NPP23b] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Optimal security notion for decentralized multi-client functional encryption. Cryptology ePrint Archive, Paper 2023/435, 2023. `https://eprint.iacr.org/2023/435`. Full version of [NPP23a]. 63, 66, 79, 81, 83

[NPP24] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Chosen-ciphertext security for multi-client functional encryption: Definitions and constructions, 2024. *Work in wrogress.* 25, 145

[NPP25] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. Multi-client functional encryption with public inputs and strong security. In *Public-Key Cryptography - IACR PKC 2025*. Springer, 2025. `https://eprint.iacr.org/2024/740`. 6, 12, 13, 19, 31, 89, 106, 108

[NPS22] Ky Nguyen, David Pointcheval, and Robert Schädlich. Dynamic decentralized functional encryption with strong security. Cryptology ePrint Archive, Paper 2022/1532, 2022. `https://eprint.iacr.org/2022/1532`. Full version of [NPS25]. 113, 114

[NPS24a] Ky Nguyen, David Pointcheval, and Robert Schädlich. Decentralized multi-client functional encryption with strong security. *IACR Communications in Cryptology*, 1(2), 2024. 4, 6, 10, 12, 22, 117, 125, 156

[NPS24b] Ky Nguyen, David Pointcheval, and Robert Schädlich. Decentralized multi-client functional encryption with strong security. Cryptology ePrint Archive, Paper 2024/764, 2024. `https://eprint.iacr.org/2024/764.` Full version of [NPS24a]. 13, 24, 117, 121, 125, 126, 142

[NPS25] Ky Nguyen, David Pointcheval, and Robert Schädlich. Dynamic decentralized functional encryption:generic constructions with strong security. In *Public-Key Cryptography - IACR PKC 2025*. Springer, 2025. 12, 24, 145, 146, 156

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM CCS 2007*, pages 195–203. ACM Press, October 2007. 4, 10

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 191–208. Springer, Heidelberg, August 2010. 15, 18, 22, 43, 45

[OT12a] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 591–608. Springer, Heidelberg, April 2012. 18, 22, 43, 45

[OT12b] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In Xiaoyun Wang and Kazue Sako, editors,

ASIACRYPT 2012, volume 7658 of *LNCS*, pages 349–366. Springer, Heidelberg, December 2012. 4, 10, 15, 18, 22, 43, 45, 50, 57

[PD21] Tapas Pal and Ratna Dutta. Attribute-based access control for inner product functional encryption from LWE. In Patrick Longa and Carla Ràfols, editors, *LATINCRYPT 2021*, volume 12912 of *LNCS*, pages 127–148. Springer, Heidelberg, October 2021. 16, 20

[QLH+24] Xinyuan Qian, Hongwei Li, Meng Hao, Guowen Xu, Haoyong Wang, and Yuguang Fang. Decentralized multi-client functional encryption for inner product with applications to federated learning. *IEEE Transactions on Dependable and Secure Computing*, 2024. 13

[Sha79] Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979. 32

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, Heidelberg, August 1984. 4, 10, 62

[SV23] Elaine Shi and Nikhil Vanjani. Multi-client inner product encryption: Function-hiding instantiations without random oracles. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 622–651. Springer, Heidelberg, May 2023. 4, 6, 10, 12, 22, 23, 125

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005. 4, 10, 12, 19, 62

[TAO16] Junichi Tomida, Masayuki Abe, and Tatsuaki Okamoto. Efficient functional encryption for inner-product values with full-hiding security. In Matt Bishop and Anderson C. A. Nascimento, editors, *ISC 2016*, volume 9866 of *LNCS*, pages 408–425. Springer, Heidelberg, September 2016. 22

[Tom19] Junichi Tomida. Tightly secure inner product functional encryption: Multi-input and function-hiding constructions. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 459–488. Springer, Heidelberg, December 2019. 22, 94, 96, 97

[Tom20] Junichi Tomida. Tightly secure inner product functional encryption: Multi-input and function-hiding constructions. *Theoretical Computer Science*, 833:56–86, 2020. 22

[Ünal20] Akin Ünal. Impossibility results for lattice-based functional encryption schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 169–199. Springer, Heidelberg, May 2020. 23

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Heidelberg, August 2009. 142

[Wee21] Hoeteck Wee. Broadcast encryption with size $N^{1/3}$ and more from k-lin. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 155–178, Virtual Event, August 2021. Springer, Heidelberg. 6, 11, 40

[Wee22] Hoeteck Wee. Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 217–241. Springer, Heidelberg, May / June 2022. 6, 11

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, November 1982. 4, 10

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986. 4, 10

RÉSUMÉ

Avec la généralisation de TLS sur le web, la confidentialité des échanges s'est renforcée. Mais cela a du même coup ouvert de nouvelles voies aux acteurs malveillants pour attaquer directement les machines individuelles via leur navigateurs, en contournant tous les dispositifs d'analyse de flux, puisque tout transite dans un tunnel chiffré. Ainsi, pour détecter ou empêcher les attaques, nombre de systèmes opèrent une rupture de flux chiffré pour continuer à analyser les paquets en clair, mettant ainsi à mal la confidentialité. Cette thèse va étudier les mécanismes cryptographiques permettant de garantir la confidentialité des données, tout en permettant des analyses pour garantir la sécurité des usagers et des systèmes. Il s'agira pour cela d'adapter le *chiffrement fonctionnel* ou le *chiffrement à base d'attributs*, pour permettre aux sondes d'extraire les seules informations utiles à des fins de cybersécurité.

Cette thèse se focalise sur le chiffrement fonctionnel avec *plusieurs utilisateurs*, en particulier où des *clients* peuvent individuellement chiffrer leurs données partielles, ou des *senders* peuvent engendrer individuellement leur clé fonctionnelle partielle. Ces chiffrés partiels ou clés partielles peuvent être combinés plus tard, si et seulement s'ils partagent un tag commun, *e.g.* un horodatage. Nous obtenons des résultats par rapport à la notion de sécurité du chiffrement fonctionnel dans ce cadre, avec à la fois de nouvelles définitions et de nouvelles constructions. D'une part, nous proposons un cadre pour définir le chiffrement fonctionnel *multi-client* avec un contrôle d'accès fin sur les clés de déchiffrement, qui est généralisé au cas d'une classe de fonctions ayant des informations publiques lors du chiffrement. D'autre part, nous examinons à nouveau le modèle de sécurité du chiffrement fonctionnel *multi-client décentralisé* et raffinons ses contraintes existantes. Finalement, nour construisons des schémas concrets à l'égard de la classe de fonctions pour calculer les produits scalaires, en exploitant les *espaces vectoriels duaux avec couplages* dans les groupes bilinéaires.

MOTS CLÉS

cryptologie ⋆ vie privée ⋆ cybersecurité

ABSTRACT

With the generalisation of TLS over the Web, the confidentiality of communications has been reinforced. However, this also led to new attack vectors for adversarial agents to attack directly the individual machines via their browsers, while bypassing all the tools for data-flow analysis, because everything is transmitted through an encrypted channel. Therefore, in order to detect or prevent the attacks, many systems operate by stopping the encrypted channel and continuing to analyse the data packets in the clear, which thus affects badly the confidentiality. This thesis is going to study the cryptographic mechanisms that allow guaranteeing the confidentiality of data, at the same time permitting the analysis to ensure the security of the users and systems. This will require adapting the techniques of *functional encryption* (FE) or *attribute-based encryption* (ABE), which enable the monitors to extract only the useful information for the cybersecurity purposes.

The main setting of our studies is FE with *multiple users*, in particular where we allow multiple *clients* to independently encrypt their partial data, or multiple *senders* to independently generate their partial functional decryption keys. These partial ciphertexts or partial keys can be later jointly combined, only if they are associated to some identical tag, *e.g.* a timestamp. We obtain various results with respect to the security notions of FE in this setting, both definitionally and constructively. On one hand, we give a definitional framework for *multi-client* FE with fine-grained access control over keys, which is furthermore generalized to function classes that authorize some auxiliary public inputs at the time of encryption. On the other hand, we revisit the widely used security model of *decentralized multi-client* FE and refine existing unnatural constraints of the model. Last but not least, we provide concrete constructions in regards of the particular function class for computing inner products, by leveraging the power of *dual pairing vector spaces* in the bilinear group setting.

KEYWORDS

cryptography ⋆ privacy ⋆ cybersecurity