Public-key encryption, revisited : tight security and richer functionalities

Romain Gay

PSL UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École normale supérieure

Public-Key Encryption, Revisited: Tight Security and Richer Functionalities

Soutenue par
Romain Gay
Le 18 mars 2019

Ecole doctorale n° 386
Sciences Mathématiques de Paris Centre

Spécialité
Informatique

Composition du jury :

M. Michel FERREIRA ABDALLA École Normale Supérieure	*Directeur de thèse*
M. Hoeteck WEE École Normale Supérieure	*Directeur de thèse*
M. Benoît LIBERT École normale supérieure de Lyon	*Rapporteur*
M. Katsuyuki TAKASHIMA Mitsubishi Electric	*Rapporteur*
M. Dario FIORE IMDEA Software Institute	*Examinateur*
M. Pierre Alain FOUQUE Université Rennes 1	*Examinateur*
M. Dennis HOFHEINZ Karlsruher Institut für Technologie	*Examinateur*
Mme. Huijia LIN University of Washington	*Examinatrice*

ENS ÉCOLE NORMALE SUPÉRIEURE

PSL RESEARCH UNIVERSITY PARIS

Public-key Encryption, Revisited: Tight Security and Richer Functionalities

Romain Gay

Supervised by Michel Ferreira Abdalla and Hoeteck Wee

Abstract

Our work revisits public-key encryption in two ways: 1) we provide a stronger security guarantee than typical public-key encryption, which handles many users than can collude to perform sophisticated attacks. This is necessary when considering widely deployed encryption schemes, where many sessions are performed concurrently, as in the case on the Internet; 2) we consider so-called functional encryption, introduced by Boneh, Sahai, Waters in 2011, that permits selective computation on the encrypted data, as opposed to the coarse-grained access provided by traditional public-key encryption. It generalizes the latter, in that a master secret key is used to generate so-called functional decryption keys, each of which is associated with a particular function. An encryption of a message m, together with a functional decryption key associated with the function f, decrypts the value $f(m)$, without revealing any additional information about the encrypted message m. A typical scenario involves the encryption of sensitive medical data, and the generation of functional decryption keys for functions that compute statistics on this encrypted data, without revealing the individual medical records.

In this thesis, we present a new public-key encryption that satisfies a strong security guarantee, that does not degrade with the number of users, and that prevents adversaries from tampering ciphertexts. We also give new functional encryption schemes, whose security relies on well-founded assumptions. We follow a bottom-up approach, where we start from simple constructions that can handle a restricted class of functions, and we extend these to richer functionalities. We also focus on adding new features that make functional encryption more relevant to practical scenarios, such as multi-input functional encryption, where encryption is split among different non-cooperative users. We also give techniques to decentralize the generation of functional decryption keys, and the setup of the functional encryption scheme, in order to completely remove the need for a trusted third party holding the master secret key.

Résumé

Nos travaux revisitent le chiffrement à clé publique de deux facons: 1) nous donnons une meilleure garantie de sécurité que les chiffrements à clé publique typiques, qui gère de nombreux utilisateurs pouvant coopérer pour réaliser des attaques sophistiquées. Une telle sécurité est nécessaire lorsque l'on considère des schémas de chiffrement largement déployés, où de nombreuses sessions ont lieu de manière concurrente, ce qui est le cas sur internet; 2) nous considérons le chiffrement fonctionnel, introduit en 2011 par Boneh, Sahai et Waters, qui permet un calcul sélectif sur les données chiffrées, par opposition à l'accès tout ou rien permis par les schémas de chiffrement à clé publique traditionnels. Il généralise ce dernier dans le sens où une clé secrète maîtresse permet de générer des clés de chiffrement fonctionnelles, qui sont chacune associées à une fonction particulière. Le déchiffrement du chiffrement d'un message m avec une clé de déchiffrement fonctionnelle associée à une fonction f obtiendra la valeur $f(m)$, et aucune autre information à propos du message chiffré m. Un scénario typique: des données médicales privées sont chiffrées, et des clés de déchiffrement fonctionnelles sont générées pour des fonctions qui permettent de calculer des statistiques, sans révéler les données individuelles chiffrées.

Dans cette thèse, nous présentons un nouveau schéma de chiffrement à clé publique satisfaisant une garantie de sécurité forte, qui ne se dégrade pas avec le nombre de clients utilisant le schéma, et qui empêche les adversaires de modifier activement les chiffrés. Nos donnons aussi des schémas de chiffrement fonctionnels, dont la sécurité repose sur des hypothèses calculatoires robustes. L'approche suivie est bottom-up, où des constructions simples qui permettent de générer des clés pour une classe restreinte de fonctions sont étendues à des classes de fonctions plus riches. Un intérêt a aussi été porté à l'étude d'améliorations qui rendent le chiffrement fonctionnel plus utilisable en pratique, tel que le chiffrement fonctionnel multientré, où le chiffrement est partagé entre différents utilisateurs, sans coopération. Nous donnons aussi des techniques permettant de décentraliser la génération de clés de déchiffrement fonctionnelles, et la mise en place du schéma de chiffrement, de sorte que la présence d'un tiers de confiance possédant la clé secrète principale ne soit plus nécessaire.

Contents

Chapter 1

Introduction

Cryptography helps resolve the tension between the ubiquitous use of mistrusted third-party servers to store sensitive data, and the desire for privacy. Concentrating data in a few powerful centers induces economies of scale, and provides an unprecedented availability of computing power and data storage. However, giving away sensitive data in the clear implies that clients have to trust their providers. Advanced encryption mechanisms overcome this issue by allowing users to encrypt their data in a way that still permits servers to perform selective computation on this encrypted data. The information revealed is exactly what is required by the server to provide its service to the clients, and nothing else. Moreover, given its unprecedented world-wide deployment, public-key cryptography needs to fulfill a strong security, which prevents sophisticated attacks using multiple concurrent sessions, which are inevitable on the Internet.

The work presented in this thesis addresses the following two limitations of traditional public-key cryptography: 1) it provides stronger security for public-key encryption, that does not degrade with the number of users, as is necessary for largely deployed systems, 2) it presents encryption schemes, known as functional encryption schemes, which permit fine-grained access and selective computation on the encrypted data.

Public-key cryptography. Following the tradition in cryptography, we exemplify public-key encryption using fictional characters Alice and Bob. Alice wants to send sensitive data to Bob through an insecure channel. Without sharing any information a priori, Alice and Bob can use public-key cryptography to prevent Eve, the eavesdropper, to intercept and read the content of the data. Namely, Bob produces a public key, which can be thought of as the digital analog of a safe, together with a key that opens the safe. The key is kept secret by Bob, whereas the safe itself is published for anybody to use (in the digital world, objects can be copied at will, and used indefinitely many times). Alice puts her message in the safe, closes it (think of a safe that can be closed without the key; this process is referred to as encrypting the message), and sends the safe (known as the ciphertext) to Bob, who can open it with his key (this process is referred to as decrypting the ciphertext). Eve doesn't see the content inside the safe, since it's opaque, thus, the message remains confidential. The only information that is revealed is an upper bound on the size of the message, since the safe has to be at least as large as the message it contains. Originally put forth by [DH76, Mer78], public-key encryption has become ubiquitous, in particular with the Transport Layer Security (TLS) protocol, which has widespread use on the Internet, such as web browsing or instant messaging.

Methodology: defining security. The security of public-key encryption is defined formally as a game between an adversary that tries to win, that is, to trigger a particular event, or learn some particular information (for instance, in an encryption scheme, the adversary wins if it can recover the encrypted message only knowing the public key), and a challenger that interacts with the adversary. The game simply specifies which messages are sent by the challenger

depending on the adversary's behavior, and the winning condition for the adversary. The security game is defined in such a way that the adversary's capabilities encompass all possible attacks that could reasonably occur in a real-life scenario. The winning condition is defined so as to capture security breaches.

Defining security is a challenging task that has prompted fundamental research papers, such as [GM84], which defined the notion of semantic security for public-key encryption, and the indistinguishability-based notion of security. Security definitions always have to keep up with the apparition of new practical attacks allowed by new technologies. For instance, the practical attack of Bleichenbacher [Ble98] on certain standardized and widely used protocols prompted the adoption of a stronger security definition (known as Chosen-Ciphertext Attacks security, originally studied in [DDN03, RS92]) as the de facto security notion for encryption.

Provable security. Given a well-defined security game, to prove the security of a particular scheme, it remains to prove that no *efficient* adversary can win the security game with *good* probability. Influenced by complexity theory, cryptographers use a so-called security parameter that measures the input size of a computational problem, and adversaries are defined as probabilistic Turing machines, whose running time is *polynomial* in the security parameter. Since an adversary can run multiple times on different independent random tapes to increase its winning probability, the natural choice for the bound on the winning probability is any negligible function in the security parameter, that is, any function that is asymptotically dominated by all functions of the form $1/P$ for any polynomial P. A more practically oriented approach estimates the running time of the security reduction and its advantage in breaking the underlying assumption more precisely than polynomial running time, and negligible winning advantage. The reduction can thus be used to choose *concrete* security parameters for the underlying assumption. See for instance [BDJR97b, BR96] which pioneered concrete security.

Standard assumptions. To prove that there exists no polynomial time adversary that can win a security game with non-negligible probability, we use a reductionist approach. Namely, we build an efficient algorithm (called the reduction) that leverages the adversary's success in winning the security game, to find a solution to a *hard* problem, that is, a problem that is impossible to solve efficiently with non-negligible probability, or at least, conjectured to be so. The tradition in cryptography departs from complexity theory at this point, given that basing cryptography on NP-hard problems has remained open for many years. Instead, security of cryptographic schemes relies on a more heuristic approach, where security is proven via a reduction to a well-defined assumption, which states that some problem is hard in practice, that is, for which there exists no known efficient solution. Of course, the robustness of the security depends on how much this assumption is trusted. Provable security makes sense as long as it relies on assumptions that have been extensively studied. Typically, they involve decade-old mathematical problems, where finding an efficient algorithm would represent a huge breakthrough. Instead of using ad hoc cryptanalysis for every possible cryptographic scheme, one can rely on a small set of simple-to-state assumptions, leveraging years of mathematical research. Assumptions whose validity is widely trusted are called standard assumptions. For example, this is the case of the discrete logarithm assumption, which states that given a cyclic group of prime order p, generated by g, and an element g^a for a random exponent a in $\mathbb{Z}_p$ (we use multiplicative notation here), it is hard to compute the discrete logarithm a (of course, the choice of the underlying group is crucial to the validity of the assumption, and only for certain well-chosen groups is this assumption considered standard).

Tight Security

As explained in the paragraph about provable security, a security reduction can serve as a tool to choose concrete security parameters. Indeed, an adversary that can win a security

game can be used by a reduction to break a computational problem that is assumed to be hard. However, the reduction may be slightly less efficient at breaking the hard problem than the adversary can win the security game. This gap in efficiency is referred to as the security loss. When choosing the security parameter according to the reduction, it is necessary to take into account this security loss. For instance, say we want 128 bits of security for a particular scheme, which means no efficient adversary should be able to break the security of the scheme with advantage more than 2^{-128}. Suppose the reduction leverages the adversary to break the discrete logarithm problem with advantage $2^{-128}/L$, where L is the security loss. Typically, the security loss grows with the number of challenge ciphertexts involved in the security game. That is, the more deployed the scheme, the larger the security loss. This can be an issue for widespread cryptographic protocols, such as TLS, where sophisticated attacks using many concurrent sessions can be mounted. For instance, L can be as large as 2^{30} in widely deployed systems. Then, it is necessary to choose a group where it is assumed to be impossible to solve the discrete logarithm problem efficiently with an advantage of more than 2^{-158}. In other words, a large security loss implies large parameters, and a less efficient scheme overall. Security is said to be tight when the security loss is small and in particular, independent of the number of clients using the scheme.

State of the Art in Tight Security

The most basic security guarantee required from a public-key encryption scheme is IND-CPA security, which stands for INDistinguishability against Chosen-Plaintext Attacks, defined in [GM84], which captures passive, eavesdropping attacks. Many existing IND-CPA-secure encryption schemes have a tight security. For instance, this is the case of El Gamal encryption scheme [ElG85], whose security tightly reduces to the Decisional Diffie Hellman (DDH) assumption [DH76], a standard assumption that implies the discrete logarithm assumption. This directly follows from the fact that the DDH assumption is random self-reducible: it is as easy to break many instances of the DDH assumption than just one instance, for a given prime-order group. However, the de facto security definition for public-key encryption is a stronger so-called IND-CCA, which stands for INDistinguishability against Chosen-Ciphertexts Attacks, originally introduced in [DDN03, RS92], where the adversary can actively manipulate and tamper with ongoing ciphertexts. Such attacks have been shown to be practically realizable in real life, such as the attack from [Ble98] on a widely used cryptographic protocol. Unfortunately, most CCA-secure public-key encryption schemes, such as the seminal construction from [CS98], or its improvements in [KD04, HK07], do not have a tight security proof: the security loss is proportional to the number of challenge ciphertexts in the security game. The first CCA-secure public-key encryption with a tight security proof was given in [HJ12], and a long line of works [LJYP14, LPJY15, HKS15, AHY15a, GCD+16, Hof17] improved efficiency considerably. However, the security of all of these schemes rely on a qualitatively stronger assumption than non-tightly secure schemes [CS98, KD04, HK07], in particular, they require pairing-friendly elliptic curves (henceforth simply referred to as pairings), an object first used for cryptography in [BF01, BF03, Jou00, Jou04]. This situation prompted the following natural question: does tight security intrinsically require a qualitatively stronger assumption, for CCA-secure public-key encryption? This question falls into the broad theoretical agenda that aims at minimizing the assumptions required to build cryptographic objects as fundamental as public-key encryption. Besides, eliminating the use of pairings is also important in practice, because it broadens the class of groups that can be used for the underlying computational assumption. In particular, it makes it possible to choose groups that admit more efficient group operations and more compact representations, and also avoid the use of expensive pairing operations.

Reference	$\|\mathsf{pk}\|$	$\|\mathsf{ct}\|$	security loss	assumption
[CS98]	3	3	$O(Q)$	DDH
[KD04, HK07]	2	2	$O(Q)$	DDH
[HJ12]	$O(1)$	$O(\lambda)$	$O(1)$	pairings
[LJYP14, LPJY15]	$O(\lambda)$	47	$O(\lambda)$	pairings
[AHY15a]	$O(\lambda)$	12	$O(\lambda)$	pairings
[GCD+16]	$O(\lambda)$	6	$O(\lambda)$	pairings
[GHKW16]	2λ	3	$O(\lambda)$	DDH
[Hof17]	28	6	$O(\lambda)$	pairings
[Hof17]	20	28	$O(\lambda)$	DCR
[GHK17]	6	3	$O(\lambda)$	DDH

Figure 1.1: Comparison amongst CCA-secure encryption schemes, where Q is the number of challenge ciphertexts, $|\mathsf{pk}|$ denotes the size (in groups elements) of the public key, and $|\mathsf{ct}|$ denotes the ciphertext overhead, ignoring smaller contributions from symmetric-key encryption. DCR stands for Decisional Composite Residuosity, a standard assumption that relies on the fact that factorizing larger numbers is heuristically hard, originally introduced in [Pai99] (see Definition 16).

Contribution 1: Tightly CCA-Secure Encryption without Pairing

In [GHKW16], which is presented in Chapter 3 of this thesis, we answer this question negatively. Namely, we present the first CCA-secure public-key encryption scheme based on DDH where the security loss is independent of the number of challenge ciphertexts and the number of decryption queries, whereas all prior constructions [LJYP14, LPJY15, HKS15, AHY15a, GCD+16, Hof17] rely on the use of pairings. Moreover, our construction improves upon the concrete efficiency of prior schemes, reducing the ciphertext overhead by about half (to only 3 group elements under DDH), in addition to eliminating the use of pairings. Figure 1.1 gives a comparison between existing CCA-secure public-key encryption schemes.

One limitation of our construction is its large public key: unlike the schemes with looser security reduction from [CS98, KD04, HK07], which admit a public key that only contains a constant number of group elements, our public key contains λ group elements, where λ denotes the security parameter. Using techniques from [Hof17], we present in [GHK17] the first CCA-secure public-key encryption with a tight security reduction to the DDH assumption (without pairings), whose public key only contains a constant number of group elements. The efficiency is comparable with [GHKW16], since the ciphertexts only contain three group elements. We choose to only present in this thesis the work from the precursor [GHKW16].

Functional Encryption

We now proceed to address another limitation of traditional public-key encryption: it only provides an all-or-nothing access to the encrypted data. Namely, with the secret key, one can decrypt the ciphertext and recover the message entirely; without the secret key, nothing is revealed about the encrypted message (beyond its size). To broaden the scope of applications of public-key encryption, [O'N10, BSW11] introduced the concept of functional encryption, which permits selective computations on the encrypted data, that is, it allows some authorized users to compute partial information on the encrypted data. In a functional encryption scheme,

a public key is generated, so as to allow anyone to encrypt any private message m. So-called functional decryption keys are generated from a master secret key, each of which is associated to a particular function f. Decrypting an encryption of a message m with a functional decryption key associated with a function f reveals the value $f(m)$, but no more information on the message m. The level of information that is revealed about the encrypted message is controlled by whoever generates the functional decryption keys. This is particularly useful when data is highly sensitive, such as medical data, but when revealing aggregated information on this data does not violate the privacy of the users whose data is collected, and yields many applications, such as useful statistics for medical research. The security of a functional encryption scheme guarantees that even a collusion of functional decryption keys for different functions does not reveal anything more than what each individual functional decryption key allows a user to learn. A description of the algorithms involved in a functional encryption scheme is given in Figure 1.2.

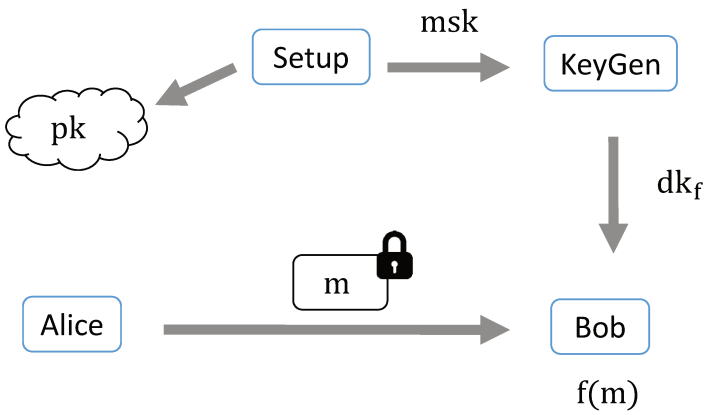


Figure 1.2: Illustration of Functional Encryption. In this scenario, the setup generates a public key pk that Alice uses to encrypt the message m, and sends the ciphertext to Bob. The setup also generates a master secret key msk, that is used by a key generation algorithm to generate a functional decryption key dk_f associated with the function f. Upon receiving this decryption key, Bob recovers the value $f(m)$ from the encryption of m.

Applications of functional encryption. We present several use cases of functional encryption.

- Consider the scenario where a hospital records medical data of its patients. For medical research, it would be useful to compute statistics on this data. Using functional encryption, the hospital can delegate the storage of this sensitive data to a mistrusted cloud server, by providing the data encrypted. Then, it can generate the functional decryption keys that allow medical researchers to learn the statistics they need to conduct their research, without revealing the individual records of each patient.

- Suppose a user Alice generates a public key and secret key encryption pair, so that Bob can send an encrypted email to Alice. The latter arrives at Alice's email provider server, which Alice does not trust. Using functional encryption, Alice can send to the server functional decryption keys that would allow the server to process her emails and take appropriate actions without her intervention, such as spam filtering or other operations on email that do not require to reveal the entire content of the emails (recall Alice has limited trust in the server). For instance, the server could classify the email into appropriate folders, or learn whether an email is urgent, and if so, notify Alice. It could even generate automatic answers to Bob. This use case is presented in [DGP18].

- Using functional encryption, one can perform machine learning on encrypted data. Namely, after a classifier is learned on plain data, one can generate a functional decryption key associated with this classifier, which allows decryption to run the classification on encrypted data, and reveals only the result of the classification. In [DGP18], a concrete implementation of functional encryption performs classification of hand-written digits from the MNIST dataset, with 97.54% accuracy, where the encryption and decryption only take a few seconds.

Difference with respect to fully homomorphic encryption. In a fully homomorphic encryption scheme, it is possible to publicly evaluate any function on the encrypted data. This differs from functional encryption in two major ways: first, the result of evaluating a function f on an encryption of message m does not reveal the evaluation $f(m)$ in the clear, but only an encryption of it. Consider the email filtering scenario: using fully homomorphic encryption, the email server would not be able to decide whether an incoming encrypted email is spam, without the intervention of the client, who is the only one who can decrypt the result of the evaluation on encrypted data. Second, using fully homomorphic encryption, anyone can compute arbitrary functions on the encrypted data: there is no guarantee that the computation was performed correctly. In a functional encryption scheme, the owner of the functional decryption key associated with function f can extract $f(m)$, from an encryption of m, and nothing else. In particular, this gives verifiability for free, unlike fully homomorphic encryption, which requires additional costly zero-knowledge proofs to verify that the proper function has been evaluated on the encrypted data.

Security of functional encryption. Security notions for functional encryption were first given in [O'N10, BSW11]. These works present a simulation-based security definition, where an efficient simulator is required to generate the view of the adversary in the security game, only knowing the information that leaks from the encrypted values and corrupted functional decryption keys. They prove that such a security notion is impossible to achieve in general, and give another indistinguishability-based variant of the security definition, essentially a security definition similar to [GM84], generalized to the context of functional encryption. In this security game, an adversary receives the public key of the encryption scheme, and then, it can obtain functional decryption keys for functions f of its choice. It also sends two messages, m_0 and m_1, to the challenger, in the security game, which samples a random bit $b \leftarrow_{\mathrm{R}} \{0,1\}$, and sends back an encryption of the message m_b. Assuming the functional encryption keys that are obtained by the adversary are associated with functions f that do not distinguish these two messages, that is, for which $f(m_0) = f(m_1)$, the adversary should not be able to guess which bit b was used with a probability significantly more than $1/2$, which can be obtained by random guessing. Intuitively, if the functions f do not help distinguish these two messages, then no information should be revealed about which message m_b was encrypted. An artificial but useful weakening of the security model is the so-called selective security, where the game is identical to the description above, except the adversary is required to decide on which messages m_0 and m_1 to choose beforehand, that is, before seeing the public key or obtaining any functional decryption keys. This notion is useful as a stepping stone towards full-fledged security. Moreover, a guessing argument can convert any selectively-secure scheme into a fully-secure scheme, albeit with a quantitative gap in the quality of the security.

State of the Art in Functional Encryption

Identity-based encryption. Historically, the first functional encryption scheme beyond traditional public-key encryption dates back to identity-based encryption, where a constant-size public key is used to encrypt messages to different users, represented by their identity. Functional decryption keys are also associated with an identity, and decryption succeeds to

recover the encrypted message if the identities associated with the ciphertext and the functional decryption key match. For instance, identities can be email addresses, and with a single public key, it is possible to encrypt a message to any user whose email address is known. The concept was thought of in [Sha84], and the first constructions whose security relied on standard assumptions were given in [BF01, Coc01].

Attribute-based encryption. Later, a more general concept was introduced: attribute-based encryption, where ciphertexts are associated with an access policy, and functional decryption keys are associated with a set of attributes. Decryption recovers the encrypted message if the attributes associated with the functional decryption key satisfy the access policy embedded in the ciphertext. Note that the role can be switched, that is, ciphertexts can be associated with attributes, and functional decryption keys embed access policies, as in [BSW07]. These are referred to as key-policy and ciphertext-policy attributed-based encryption, respectively. Such attribute-based encryption schemes have been first realized from standard assumptions in [SW05, GPSW06] for policies that can be represented as Boolean formulas, or in [GVW13, GVW15a, BGG+14] for policies that can be represented as any arbitrary circuit of polynomial size. Note that a ciphertext only hides the underlying message it encrypts, but reveals the associated access policy (or attributes, depending on whether we consider ciphertext-policy or key-policy attribute-based encryption).

Predicate encryption. Predicate encryption schemes are even more powerful than attribute-based encryption schemes, since the access policy associated with a ciphertext remains hidden (or the attributes, depending on whether we consider the ciphertext-policy or the key-policy variant). The first constructions from standard assumptions were given in [BW07] for comparison and subset queries, in [KSW08, KSW13] for constant-depth Boolean formulas, and in [GVW15b] for all circuits. Such predicate encryption schemes are sometimes referred to as private-index predicate encryption, whereas attribute-based encryption (which do not hide the policy or attributes underlying each ciphertext) are referred to as public-index predicate encryption. It is important to note that the construction from [GVW15b] only hides the attributes underlying each ciphertext (they build a key-policy predicate encryption, where attributes are associated with ciphertexts) when the adversary can only obtain functional decryption keys for access policies which are not satisfied by the attribute of the challenge ciphertext. This is referred to as weakly-hiding the attributes. Prior works [BW07, KSW08, KSW13] fully hide the attributes associated with each ciphertext, the only information that leaks being the value of the predicate evaluation, namely, whether or not the decryption succeeds. In fact, fully-hiding predicate encryption for all circuits essentially implies functional encryption for all circuits, for which we have no construction based on standard assumptions. We defer the interested reader to [GVW15b, 1.3 Discussion] for further details on the connections between predicate encryption and functional encryption for all circuits.

Functional encryption beyond predicates. So far, we have only discussed special kinds of functional encryption where decryption successfully recovers the *entire* message if the attributes associated with the ciphertext (resp. the functional decryption key) satisfy the access policy embedded in the key (resp. the ciphertext). While this is a fruitful generalization of traditional public-key encryption, since it permits embedding complex access policy into the encrypted data, this is still an all-or-nothing encryption: either the message is entirely recovered by the decryption, or no information whatsoever is revealed about the message. Not much is known about functional encryption with fine-grained access to the encrypted data, that is, where decryption recovers *partial information* about the encrypted data. In [ABDP15], the authors build the first construction of functional encryption from standard assumptions beyond predicates. In [ABDP15], messages to be encrypted are vectors of integers, in $\mathbb{Z}^d$, for some dimension $d \in \mathbb{N}$ that is fixed during the setup of the scheme. Functional decryption keys

are associated with vectors $\mathbf{y} \in \mathbb{Z}^d$. Decryption of an encryption of $\mathbf{x} \in \mathbb{Z}^d$ with a functional decryption key associated with $\mathbf{y} \in \mathbb{Z}^d$ recovers $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}$, which denotes the inner product between $\mathbf{x}$ and $\mathbf{y}$. Otherwise stated, this encryption scheme lets owners of functional decryption keys compute weighted sum on the encrypted data. Moreover, it is possible to encode any constant-depth formula as a polynomial of constant degree, which can be evaluated *via* functional encryption for inner products. That is, this scheme handles computation of NC0 circuits on encrypted data. Later, [ALS16] gave fully-secure functional encryption schemes (the original schemes from [ABDP15] being only selectively-secure). In this thesis, we present extensions of these functional encryption for inner products, and new functional encryption schemes with succinct ciphertexts that supports the evaluation of degree-2 polynomials on encrypted data. More details on the contributions of this thesis are given below.

Related works: functional encryption for bounded collusion. The case where security is guaranteed only when a *constant* number of functional decryption keys are corrupted has been considered in prior works. [SS10] built the first functional encryption for all circuits, where security handles the corruption of *one* functional decryption key, using garbled circuits and public-key encryption. In this functional encryption, the ciphertext size depends on the size of the circuit associated with the functional decryption keys (which thus needs to be bounded during the setup of the scheme). [GKP+13] improves upon [SS10] since the ciphertext size depends only on the size of the output of the function for which functional decryption keys are generated. They use attributed-based encryption for all circuits, and fully homomorphic encryption, both of which admits construction from standard assumptions. Note that the security of both of these constructions breaks down as soon as *two* functional decryption keys are corrupted. [GVW12, Agr17] show how to generically turn any functional encryption secure only when one functional decryption key is corrupted, into a functional encryption scheme where security handles an a priori bounded polynomial number of collusions. We now consider the case of general functional encryption with *unbounded* collusions.

Theoretical motivation: the power of general purpose functional encryption. As mentioned before, the existing functional encryption schemes from standard assumptions only permit the evaluation of degree-1 (inner products) or degree-2 polynomials on the encrypted data. However, there are feasibility results for functional encryption schemes where functions associated to functional decryption keys can be any arbitrary circuits (such schemes are called general purpose functional encryption schemes). The first candidate construction for general purpose functional encryption appeared in [GGH+13b, GGH+16]. It relies on Indistinguishability Obfuscation, a powerful object, originally defined in [BGI+01, BGI+12], that has been remarkably successful at providing an all-purpose tool for solving cryptographic problems, as shown in [SW14]. [GGH+13b, GGH+16] gave a construction for Indistinguishability Obfuscation that relies on cryptographic multilinear maps, for which there is currently no construction from standard assumptions. Other works [BLR+15, GGHZ16] gave direct candidate constructions of functional encryption from multilinear maps.

Follow-ups [Lin16, LV16, Lin17, AS17, LT17] focused on reducing the degree of the required multilinear map, all the way down to 3 in [LT17] (the degree of the multilinear map required in prior works depends on the complexity of the circuits for which functional decryption keys are generated). Namely, in [LT17], general purpose functional encryption is built from succinct functional encryption which handles evaluation of degree-3 polynomials on encrypted data (which can be built from degree 3 multilinear maps), together with some assumptions on the existence of special kind of pseudo-random generators[1]. Here, succinctness refers to the fact that the ciphertext size only depends on the underlying message, and not the functions for which functional decryption keys are generated. Unfortunately, there is no construction of even

[1] Namely, the existence of pseudo-random generators of block-wise locality 3.

degree-3 multilinear from standard assumptions. To sum up, all existing general purpose functional encryption schemes either rely on multilinear maps, or Indistinguishability Obfuscation, both of which rely on non-standard assumptions. In fact, general purpose functional encryption has been shown to imply Indistinguishability Obfuscation in [AJ15, BV15, BNPW16].

Contribution 2: Functional Encryption with New Features, and Richer Functionalities

Motivated by the quest for succinct functional encryption for richer classes of functions, we follow the bottom-up approach initiated by [ABDP15], which consists of building functional encryption as expressive as possible from standard assumptions. The benefit of this approach is two-fold: first, it aims at bridging the gap between the powerful Indistinguishability Obfusction, and the current constructions from standard assumptions; second, it gives practically relevant schemes based from concrete assumptions, which are interesting in their own right. We present extensions of the original functional encryption for inner products from [ABDP15, ALS16] with additional features: in contribution 2.1, we extend functional encryption for inner products to the multi-input setting, and to the multi-client setting in contribution 2.2, both of which generalize the standard single-input setting. Then, we expand functional encryption for richer classes of functions in contribution 2.3. These contributions are presented in more details below.

Contribution 2.1: multi-input encryption for inner products.

We present here an extension of the original functional encryption from [ABDP15, ALS16] to the more general multi-input setting.

Definition of multi-input functional encryption. As explained above, in a functional encryption (FE) scheme [SW05, BSW11], an authority can generate restricted decryption keys that allow users to learn specific functions of the encrypted messages and nothing else. That is, each FE decryption key dk_f is associated with a function f and decrypting a ciphertext $\mathsf{Enc}(x)$ with dk_f results in $f(x)$. Multi-input functional encryption (MIFE) introduced by [GGG+14] is a generalization of functional encryption to the setting of multi-input functions. A MIFE, the scheme has several encryption slots and each decryption key dk_f for a multi-input function f decrypts jointly ciphertexts $\mathsf{Enc}(x_1), \ldots, \mathsf{Enc}(x_n)$ for all slots to obtain $f(x_1, \ldots, x_n)$ without revealing anything more about the encrypted messages. The MIFE functionality provides the capability to encrypt independently messages for different slots. This facilitates scenarios where information, which will be processed jointly during decryption, becomes available at different points of time or is provided by different parties. MIFE has many applications related to computation and data mining over encrypted data coming from multiple sources, which include examples such as executing search queries over encrypted data, processing encrypted streaming data, non-interactive differentially private data releases, multi-client delegation of computation, order-revealing encryption [GGG+14, BLR+15].

Application of multi-input functional encryption for inner products. For instance, consider a database that contains profiles of the employees in company, where each profile describes the qualifications that the person has and the position that she can hold. Each such profile can be represented as an integer vector that contains the scores that person has received for her qualifications in her last evaluation. The employee profiles are sensitive information and only direct managers can access the profile information of the people in their teams. Therefore, the information of profiles needs to be protected from everyone else in the company. At the same time when the company starts a new project, the manager assigned to lead the project needs to select people for the new team. According to the needs of the project, the team

should have people serving different roles; the qualifications of each team member have different importance for every project. The selection criterion for the team members can be described as an integer vector that assigns weights to the different qualifications for the members in all team positions. In order to evaluate and compare potential teams, the manager needs to obtain the team score for each of them, which is the weighted sum of the individual qualifications.

A MIFE for inner products provides a perfect solution for the above scenario that protects the privacy of the profiles while enabling managers to evaluate possible team configurations. MIFE encryption slots will correspond to different team positions. Each person's profile will be a vector of her scores, which will be encrypted for the slot corresponding to the position she is qualified to hold. When a new project is established, the leading manager is granted a decryption key that is associated with an integer vector that assigns appropriate weight to each qualification of different team members. The manager can use this key to evaluate different combinations of people for the team while learning nothing more about the people's profiles than the team score. A similar example is the construction of a complex machine that requires parts from different manufacturers. Each part is rated based on different quality characteristics and prices, which are all manufacturer's proprietary information until a contract has been signed. The ultimate goal is to assemble a construction of parts that achieve a reasonable trade-off between quality and price. In order to evaluate different construction configurations, the company wants to compute cumulative score for each configuration that is a weighted sum over the quality rates and price of each of the parts.

State of the art for multi-input functional encryption. There are several constructions of MIFE schemes, which can be broadly classified as follows: (i) feasibility results for general circuits [GGG+14, BGJS15, AJ15, BKS16], and (ii) constructions for specific functionalities, notably comparison, which corresponds to order-revealing encryption [BLR+15]. Unfortunately, all of these constructions rely on indistinguishability obfuscation, single-input FE for circuits, or multilinear maps [GGH+13b, GGH13a], which we do not know how to instantiate under standard and well-understood cryptographic assumptions.[2]

A new construction of MIFE for inner products. In [AGRW17], we present a multi-input functional encryption scheme (MIFE) for inner products based on standard assumptions in prime-order bilinear groups. Our construction works for any polynomial number of encryption slots and achieves adaptive security against unbounded collusion, while relying on standard polynomial hardness assumptions. Prior to this work, we did not even have a candidate for 3-slot MIFE for inner products in the generic bilinear group model. Our work is also the first MIFE scheme for a non-trivial functionality based on standard cryptographic assumptions, as well as the first to achieve polynomial security loss for a super-constant number of slots under falsifiable assumptions. Prior works required stronger non-standard assumptions such as indistinguishability obfuscation or multilinear maps. Later, in [ACF+18], we put forward a novel methodology to convert single-input functional encryption for inner products into multi-input schemes for the same functionality. Our transformation is surprisingly simple, general and efficient. In particular, it does not require pairings and it can be instantiated with all known single-input schemes. This leads to two main advances. First, we enlarge the set of assumptions this primitive can be based on, notably, obtaining new MIFEs for inner products from plain DDH, LWE, and Decisional Composite Residuosity. Second, we obtain the first MIFE schemes from standard assumptions where decryption works efficiently even for messages of super-polynomial size. In this thesis, we strengthen the security of these constructions to handle corruption of the input slots. That is, to encrypt, each input slot $i \in [n]$ requires an encryption key ek_i. We consider the private-key setting, where encryption keys remain secret.

[2] Here, we refer only to unbounded collusions (i.e. the adversary can request for any number of secret keys). See the paragraph about related works for results on bounded collusions.

This is actually more relevant than the public-key setting, where the encryption keys ek_i are revealed to everyone. Indeed, in such a case, anyone can encrypt arbitrary message for any input slot. That weakens security drastically, since a challenge ciphertext $\mathsf{Enc}(\mathsf{ek}_i, m_b)$ for message m_i^b, where $b \leftarrow_{\mathrm{R}} \{0,1\}$ is chosen by the security game, can be combined with encryption of arbitrary messages for the other input slots during decryption. That means that given even a single functional decryption key for a function f, one can learn $f(*, \cdots, *, m_i^b, *, x \cdots, *)$, where each $*$ can be any arbitrary message. This is simply too much information in most relevant use cases. Thus, we consider the setting where encryption keys ek_i aren't public, which avoids precisely this kind of leakage of information. In the schemes presented in Chapter 4 and Chapter 5, the security holds even when some ek_i are corrupted. That means that even given ek_i for some slots $i \in [n]$, the security remains for other slots $j \neq i$. This is an important security feature, since that means even colluding users cannot learn any information about the encrypted messages by other users. This is relevant to assume such collusions, since in a multi-input encryption scheme, users do not communicate with each other, and do not trust each other. This is a novelty compared to [AGRW17, ACF+18]. A summary of our results and prior works on functional encryption for inner products is shown in Figure 1.3.

Reference	# inputs	setting	security	assumption	pairing
[ABDP15]	1	public-key	many-SEL-IND	DDH	no
[ALS16, ABDP16]	1	public-key	many-AD-IND	DDH	no
[BSW11]	1	any	many-SEL-SIM	impossible	
[LL18]	2	private-key	many-SEL-IND	SXDH + T3DH	yes
[KLM+18]	2	private-key	single-key many-AD-IND[3]	function-private FE	yes
Chapter 4	multi	private-key	many-AD-IND	SXDH	yes
Chapter 5	multi	private-key	many-AD-IND	DDH, DCR, LWE	no

Figure 1.3: Summary of constructions from cyclic or bilinear groups. We have 8 security notions xx-yy-zzz where xx $\in$ {one, many} refers to the number of challenge ciphertexts; yy $\in$ {SEL, AD} refers to the fact that encryption queries are selectively or adaptively chosen; zzz $\in$ {IND, SIM} refers to indistinguishability vs simulation-based security. SXDH stands for Symmetric eXternal Diffie Hellman assumption, DDH stands for Decisional Diffie Hellman assumption, DCR stands for Decisional Composite Residuosity assumption, and LWE stands for Learning With Errors assumption.

Contribution 2.2: multi-client functional encryption for inner products.

We now present another contribution of this thesis, which is an extension of multi-input functional encryption, where the encryption can additionally handle labels, which prevents mixing and matching different ciphertexts with different labels, thereby giving a stronger security notion. The labels are typically set to be time stamps, for the application we have in mind.

Definition of multi-client functional encryption. In multi-client functional encryption, as defined in [GGG+14, GKL+13], the single input x to the encryption procedure is broken down into an input vector $(x_1, \ldots, x_n)$ where the components are independent. An index i for each client and a (typically time-based) label ℓ are used for every encryption: $(c_1 = \mathsf{Enc}(1, x_1, \ell), \ldots, c_n = \mathsf{Enc}(n, x_n, \ell))$. Anyone owning a functional decryption key dk_f, for an n-ary function f and multiple ciphertexts *for the same label* ℓ, $c_1 = \mathsf{Enc}(1, x_1, \ell), \ldots, c_n =$

$\mathsf{Enc}(n, x_n, \ell)$, can compute $f(x_1, \ldots, x_n)$ but nothing else about the individual x_i's. The combination of ciphertexts generated for different labels does not give a valid global ciphertext and the adversary learns nothing from it. This is different from multi-input functional encryption, where every ciphertext for every slot can be combined with any other ciphertext for any other slot, and used with functional decryption keys to decrypt an exponential number of values, as soon as there is more than one ciphertext per slot. This "mix-and-match" feature is crucial for some of the applications of MIFE, such as building Indistinguishability Obfuscation [GGG+14]. However, it also means the information leaked about the underlying plaintext is too much for some applications. In the multi-client setting, however, since only ciphertexts with the same label can be combined for decryption, the information leaked about the encrypted messages is drastically reduced.

Decentralized multi-client functional encryption. While it allows independent generation of the ciphertexts, multi-client functional encryption (like multi-input functional encryption) still assumes the existence of a trusted third party who runs the Setup algorithm and distributes the functional decryption keys. This third party, if malicious or corrupted, can easily undermine any client's privacy. We are thus interested in building a scheme in which such a third party is entirely taken out of the equation. In [CDG+18a], we introduce the notion of decentralized multi-client functional encryption, in which the authority is removed and the clients work together to generate appropriate functional decryption keys. We stress that the authority is not simply *distributed* to a larger number of parties, but that the resulting protocol is indeed *decentralized*: each client has complete control over their individual data and the functional keys they authorize the generation of.

A new decentralized multi-client functional encryption for inner products. In [CDG+18a], we give the first decentralized multi-client functional encryption from standard assumptions, for inner products. Security is proven using bilinear pairing groups, and handles corruption of input slots. We first give an efficient centralized scheme whose security does not take into account the information leaked when decrypting incomplete ciphertexts, that is, ciphertexts for some, but not all, slots $i \in [n]$. Moreover, this scheme is only secure when there is only one challenge ciphertext per pair (i, ℓ), where $i \in [n]$ is an input slot, and ℓ is a label. The construction we give in Chapter 6 is a generalization of [CDG+18a] to encrypt vectors (instead of scalars in [CDG+18a]). Then, we deal with the limitation in the security model that requires for complete ciphertexts only. Our solution is quite generic, as this is an additional layer that is applied to the ciphertexts so that, unless the ciphertext is complete (with all the encrypted components), no information leaks about the individual ciphertexts, and thus on each component. This technique relies on a linear secret sharing scheme, hence the name *Secret Sharing Encapsulation* (SSE). It can also be seen as a decentralized version of *All-Or-Nothing Transforms* [Riv97, Boy99, CDH+00]. We propose a concrete instantiation in pairing-friendly groups, under the Decisional Bilinear Diffie-Hellman problem, in the random oracle model. This transformation works on *any* MCFE, and not only MCFE for inner products. Secondly, we show how another independent layer of single-input functional encryption for inner products authorizes repetitions: more precisely, we remove the restriction of a unique input per client and per label. Finally, we propose an efficient decentralized algorithm to generate a sum of private inputs, which can convert an MCFE for inner products into a decentralized MCFE for inner products: this technique is inspired from [KDK11], and only applies to the functional decryption key generation algorithm, and so this is compatible with the two above conversions. The resulting scheme is completely decentralized, in the sense that users do not need a trusted third party, even for setting up parameters (they just need to agree on a specific pairing group and a hash function that will be used later). These techniques used to strengthen the security of MCFE, as well as decentralize the key generation and setup, appeared in [CDG+18b].

A use case. Consider a financial firm that wants to compute aggregates of several companies' private data (profits, number of sales) so that it can better understand the dynamics of a sector. The companies may be willing to help the financial firm understand the sector as whole, or may be offered compensation for their help, but they don't trust the financial firm or each other with their individual data. After setting up a DMCFE, each company encrypts its private data with a time-stamp label under its private key. Together, they can give the financial firm a decryption aggregation key that only reveals a sum on the companies' private data weighted by public information (employee count, market value) for a given time-stamp. New keys can retroactively decrypt aggregates on old data.

Private stream aggregation (PSA). This notion, also referred to as Privacy-Preserving Aggregation of Time-Series Data, is an older primitive introduced by Shi *et al.* [SCR+11]. Even though it is quite similar to our target DMCFE scheme, PSA does not consider the possibility of adaptively generating different keys for different inner-product evaluations, but only enables the aggregator to compute the *sum* of the clients' data for each time period. PSA also typically involves a Differential Privacy component, which has yet to be studied in the larger setting of DMCFE. Further research on PSA has focused on achieving new properties or better efficiency [CSS12, Emu17, JL13, LC13, LC12, BJL16] but not on enabling new functionalities.

Contribution 2.3: Functional encryption for quadratic functions.

In [BCFG17], we build the first functional encryption scheme based on standard assumptions that supports a functionality beyond inner products, or predicates. Our scheme allows to compute *bilinear maps over the integers*: messages are expressed as pairs of vectors $(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}^n \times \mathbb{Z}^m$, secret keys are associated with $n \cdot m$ coefficients $\alpha_{i,j}$, and decryption allows to compute $\sum_{i,j} \alpha_{i,j} x_i y_j$. Bilinear maps represent a very general class of quadratic functions that includes, for instance, multivariate quadratic polynomials. These functions have several practical applications. For instance, a quadratic polynomial can express many statistical functions (e.g. (weighted) mean, variance, covariance, root-mean-square), the Euclidean distance between two vectors, and the application of a linear or quadratic classifier (e.g., linear or quadratic regression).

In [DGP18], we implement a functional encryption scheme for bilinear maps to perform machine learning on encrypted data. Namely, a quadratic classifier is learned on plain data, then, a functional decryption key is generated for a function that corresponds to the quadratic classifier. Using functional encryption, users can encrypt data, and the owner of the functional decryption key can perform classification of the encrypted data, without ever decrypting the data. In particular, no information apart from the result of the classification[4] is revealed about the encrypted data. In [DGP18], the quadratic classifier has an accuracy of 97.54% on MNIST data set of hand-written digits, where encryption and decryption only take a few seconds. In [BCFG17], we present a fully-secure construction whose security is proven in an idealized model, called the Generic Group Model (GGM), where the adversary cannot use the structure of the underlying pairing group. This is justified in practice, since for well-chosen elliptic curves, the only known attacks are generic, they do not use the structure of the underlying group. The security of the construction from [DGP18] also relies on the generic group model. In Chapter 7, we present the construction from [BCFG17] that is proven selectively-secure under standard assumptions, as opposed to relying on the generic group model. Note that [AS17, Lin17] concurrently exhibited functional encryption schemes supporting the evaluation of degree-2 polynomials, but on the arguably simpler *private-key* setting, where encryption

[4] in fact, to be technically accurate, the functional decryption keys in [DGP18] leak slightly more information than just the result of the classification: they leak the probability that a given instance belongs to each possible class.

References	security	public or private key
[AS17]	sel. GGM	private-key
[Lin17]	sel. standard	private-key
[BCFG17, DGP18]	ad. GGM	public-key
[BCFG17]	sel. standard	public-key

Figure 1.4: Existing functional encryption for quadratic functions. Here, ad. and sel. denote adaptive and selective security respectively and GGM stands for Generic Group Model.

requires a secret key. A comparison of existing functional encryption schemes for quadratic functions is given in Figure 7.1.

Other contributions

In this manuscript, we focus on presenting tightly-secure encryption, and functional encryption schemes. During this thesis, we have been also working on other topics, which led to papers accepted in peer-reviewed conferences. We give a brief description of these contributions here. A list of personal publications appears at the end of this manuscript.

- In [GMW15], we construct a lattice-based predicate encryption scheme for multi-dimensional range and multi-dimensional subset queries. Our scheme is selectively-secure and weakly attribute-hiding, and its security is based on the standard Learning With Errors (LWE) assumption. Multi-dimensional range and subset queries capture many interesting applications pertaining to searching on encrypted data. To the best of our knowledge, these were the first lattice-based predicate encryption schemes for functionalities beyond IBE and inner products.

- In [CGW15], we present a modular framework for the design of efficient adaptively secure attribute-based encryption (ABE) schemes for a large class of predicates under the standard k-Lin assumption in prime-order groups; this is the first uniform treatment of dual system ABE across different predicates and across both composite and prime-order groups. Via this framework, we obtain concrete efficiency improvements for several ABE schemes. Our framework has three novel components over prior works: (i) new techniques for simulating composite-order groups in prime-order ones (ii) a refinement of prior encodings framework for dual system ABE in composite-order groups (iii) an extension to weakly attribute-hiding predicate encryption (which includes anonymous identity-based encryption as a special case).

- In [GKW15], we initiate a systematic treatment of the communication complexity of conditional disclosure of secrets (CDS), where two parties want to disclose a secret to a third party if and only if their respective inputs satisfy some predicate. We present a general upper bound and the first non-trivial lower bounds for conditional disclosure of secrets. Moreover, we achieve tight lower bounds for many interesting setting of parameters for CDS with linear reconstruction, the latter being a requirement in the application to attribute-based encryption. In particular, our lower bounds explain the trade-off between ciphertext and secret key sizes of several existing attribute-based encryption schemes based on the dual system methodology.

- In [FGKO17], we build new Access Control Encryption (ACE), which is a novel paradigm for encryption which allows to control not only what users in the system are allowed to *read* but also what they are allowed to *write*. The original work of Damgård et al. [DHO16] introducing this notion left several open questions, in particular whether it

is possible to construct ACE schemes with polylogarithmic complexity (in the number of possible identities in the system) from standard cryptographic assumptions. In this work we answer the question in the affirmative by giving (efficient) constructions of ACE for an interesting class of predicates which includes equality, comparison, interval membership, and more. We instantiate our constructions based both on standard pairing assumptions (SXDH) or more efficiently in the generic group model.

- In [AGRW17], we present a multi-input functional encryption scheme (MIFE) for inner products based on the k-Lin assumption in prime-order bilinear groups. Our construction works for any polynomial number of encryption slots and achieves adaptive security against unbounded collusion, while relying on standard polynomial hardness assumptions. Prior to this work, we did not even have a candidate for 3-slot MIFE for inner products in the generic bilinear group model. Our work is also the first MIFE scheme for a non-trivial functionality based on standard cryptographic assumptions, as well as the first to achieve polynomial security loss for a super-constant number of slots under falsifiable assumptions. Prior works required stronger non-standard assumptions such as indistinguishability obfuscation or multilinear maps.

- In [BCFG17], we present two practically efficient functional encryption schemes for a large class of quadratic functionalities. Specifically, our constructions enable the computation of so-called bilinear maps on encrypted vectors. This represents a practically relevant class of functions that includes, for instance, multivariate quadratic polynomials (over the integers). Our realizations work over asymmetric bilinear groups and are surprisingly efficient and easy to implement. For instance, in our most efficient scheme the public key and each ciphertext consists of $2n+1$ and $4n+2$ group elements respectively, where n is the dimension of the encrypted vectors, while secret keys are only two group elements. Our two schemes build on similar ideas, but develop them in a different way in order to achieve distinct goals. Our first scheme is proved (selectively) secure under standard assumptions, while our second construction is concretely more efficient and is proved (adaptively) secure in the generic group model. As a byproduct of our functional encryption schemes, we show new predicate encryption schemes for degree-two polynomial evaluations, where ciphertexts consist of only $O(n)$ group elements. This significantly improves the $O(n^2)$ bound one would get from predicate encryption for inner products.

- In [ABGW17], we propose, implement, and evaluate fully automated methods for proving security of ABE in the Generic Bilinear Group Model ([BBG05, Boy08]), an idealized model which admits simpler and more efficient constructions, and can also be used to find attacks. Our method is applicable to Rational-Fraction Induced ABE, a large class of ABE that contains most of the schemes from the literature, and relies on a Master Theorem, which reduces security in the GGM to a (new) notion of symbolic security, which is amenable to automated verification using constraint- based techniques. We relate our notion of symbolic security for Rational-Fraction Induced ABE to prior notions for Pair Encodings. Finally, we present several applications, including automated proofs for new schemes.

- In [FG18], we focus on structure-preserving signatures on equivalence classes, or equivalence-class signatures for short (EQS), are signature schemes defined over bilinear groups whose messages are vectors of group elements. Signatures are perfectly randomizable and given a signature on a vector, anyone can derive a signature on any multiple of the vector; EQS thus sign projective equivalence classes. Applications of EQS include the first constant-size anonymous attribute-based credentials, efficient round-optimal blind signatures without random oracles and efficient access-control encryption. To date, the only existing instantiation of EQS is proven secure in the generic-group model. In this work we show that by relaxing the definition of unforgeability, which makes it efficiently

verifiable, we can construct EQS from standard assumptions, namely the Matrix-Diffie-Hellman assumptions. We then show that our unforgeability notion is sufficient for most applications.

- In [GHKP18], We provide a structure-preserving signature (SPS) scheme with an (almost) tight security reduction to a standard assumption. Compared to the state-of-the-art tightly secure SPS scheme of Abe et al. [AHN+17], our scheme has smaller signatures and public keys (of about 56%, resp. 40% of the size of signatures and public keys in Abe et al.'s scheme), and a lower security loss (of $O(\log Q)$ instead of $O(\lambda)$, where λ is the security parameter, and $Q = \mathsf{poly}(\lambda)$ is the number of adversarial signature queries). While our scheme is still less compact than structure-preserving signature schemes *without* tight security reduction, it significantly lowers the price to pay for a tight security reduction. In fact, when accounting for a non-tight security reduction with larger key (i.e., group) sizes, the computational efficiency of our scheme becomes at least comparable to that of non-tightly secure SPS schemes. Technically, we combine and refine recent existing works on tightly secure encryption and SPS schemes. Our technical novelties include a modular treatment (that develops an SPS scheme out of a basic message authentication code), and a refined hybrid argument that enables a lower security loss of $O(\log Q)$ (instead of $O(\lambda)$).

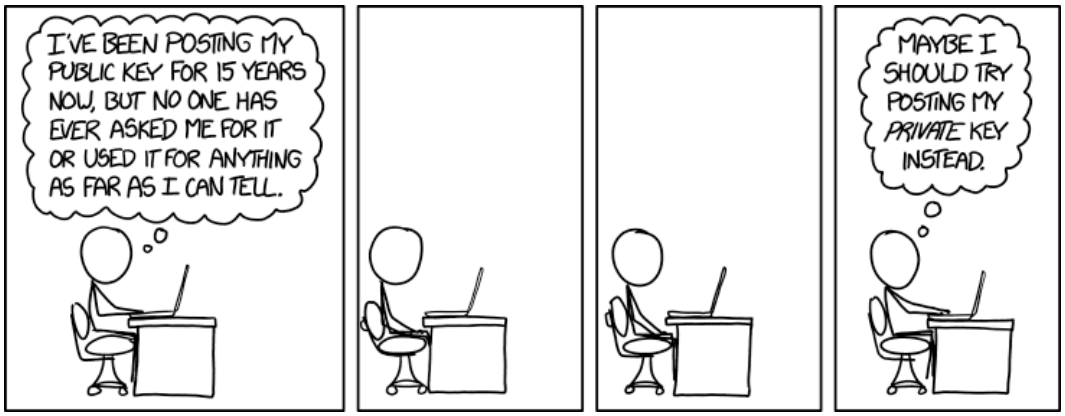
- In [ACF+18], we present new constructions of multi-input functional encryption (MIFE) schemes for the inner-product functionality that improve the state of the art solution of Abdalla et al. [AGRW17] in two main directions. First, we put forward a novel methodology to convert single-input functional encryption for inner products into multi-input schemes for the same functionality. Our transformation is surprisingly simple, general, and efficient. In particular, it does not require pairings and it can be instantiated with all known single-input schemes. This leads to two main advances. First, we enlarge the set of assumptions this primitive can be based on, notably obtaining new MIFEs for inner products from plain DDH, LWE and Composite Residuosity. Second, we obtain the first MIFE schemes from standard assumptions where decryption works efficiently even for messages of super-polynomial size. Our second main contribution is the first function-hiding MIFE scheme for inner products based on standard assumptions. To this end, we show how to extend the original, pairing-based, MIFE by Abdalla et al. [AGRW17] in order to make it function hiding, thus obtaining a function-hiding MIFE from the MDDH assumption.

- In [GKW18], we present a new public-key broadcast encryption scheme where both the ciphertext and secret keys consist of a constant number of group elements. Our result improves upon the work of Boneh, Gentry, and Waters [BGW05] in two ways: (i) we achieve adaptive security instead of selective security, and (ii) our construction relies on the decisional k-Linear Assumption in prime-order groups (as opposed to q-type assumptions or subgroup decisional assumptions in composite-order groups); our improvements come at the cost of a larger public key. Finally, we show that our scheme achieves adaptive security in the multi-ciphertext setting with a security loss that is independent of the number of challenge ciphertexts.

- In [CDG+18a], we consider a situation where multiple parties, owning data that have to be frequently updated, agree to share weighted sums of these data with some aggregator, but where they do not wish to reveal their individual data, and do not trust each other. We combine techniques from Private Stream Aggregation (PSA) and Functional Encryption (FE), to introduce a primitive we call Decentralized Multi-Client Functional Encryption (DMCFE), for which we give a practical instantiation for inner products. This primitive allows various senders to *non-interactively* generate ciphertexts which support inner-product evaluation, with functional decryption keys that can also be generated

non-interactively, in a distributed way, among the senders. Interactions are required during the setup phase only. We prove adaptive security of our constructions, while allowing corruptions of the clients, in the random oracle model.

Road-map. The rest of this thesis is organized as follows. In Chapter 2, we give the relevant background on public-key encryption and functional encryption, including security definitions and concrete assumptions that will be used throughout this thesis. In Chapter 3, we give our tightly CCA-secure encryption without pairings. Then, in Chapter 4, we present our multi-input functional encryption for inner products from pairings. In Chapter 5, we present our multi-input functional encryption for inner products without pairings. In Chapter 6, we exhibit our multi-client functional encryption for inner products. Finally, in Chapter 7, we present our functional encryption for quadratic functions, before concluding in Chapter 8.



Chapter 2

Preliminaries

Notations and Basics

For any set S, we denote by $x \leftarrow_{\mathrm{R}} S$ an element x that is picked uniformly at random over S. Adversaries or algorithms refer to Turing machines. PPT stands for Probabilistic Polynomial Time. For any PPT algorithm $\mathcal{A}$, we denote by $x \leftarrow \mathcal{A}$ an output of $\mathcal{A}$ which is sampled at random in the output space of $\mathcal{A}$, over the random coins of $\mathcal{A}$. For any Turing machine $\mathcal{A}$, we denote by $\mathbf{T}(\mathcal{A})$ its running time. Let p be a prime, and $n, m \in \mathbb{N}$. For any matrix $\mathbf{A} \in \mathbb{Z}_p^{n \times m}$, we denote by $\mathsf{Span}(\mathbf{A})$ the (column) span of $\mathbf{A}$. For any dimension $d \in \mathbb{N}$, we denote by $\mathsf{GL}_d(p)$ the set of invertible matrices in $\mathbb{Z}_p^{d \times d}$. We denote by $\mathrm{ID}_{d \times d}$ the identity matrix in $\mathbb{Z}_p^{d \times d}$. For any vector $\mathbf{x} \in \mathbb{R}^d$, we denote by $\|\mathbf{x}\|_2$ the Euclidian norm of $\mathbf{x}$, that is $\sqrt{\sum_{i=1}^{d} x_i^2}$. Throughout this paper, we denote by λ the security parameter, and we use the notation 1^λ to indicate that the security parameter is written in unary basis. For any function in parameter λ, we denote by $f(\lambda) = \mathsf{poly}(\lambda)$ the fact that f is a polynomial. We denote by $f(\lambda) = \mathsf{negl}(\lambda)$, if for all polynomials P, f is asymptotically dominated by $1/P$, that is, for λ large enough, $f(\lambda) < 1/P(\lambda)$.

Collision resistant hashing

A hash function generator is a PPT algorithm $\mathcal{H}$ that, on input 1^λ , outputs an efficiently computable function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^\lambda$.

Definition 1: Collision Resistance

We say that a hash function generator $\mathcal{H}$ outputs collision-resistant hash functions H if for all PPT adversaries $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{CR}}_{\mathcal{H},\mathcal{A}}(\lambda) := \Pr[x \neq x' \wedge \mathsf{H}(x) = \mathsf{H}(x') | \mathsf{H} \leftarrow_{\mathrm{R}} \mathcal{H}(1^\lambda), (x, x') \leftarrow \mathcal{A}(1^\lambda, \mathsf{H})] = \mathsf{negl}(\lambda).$$

Symmetric-Key Encryption

Definition 2: Symmetric-Key Encryption

A symmetric key encryption $(\mathsf{SEnc}, \mathsf{SDec})$ with key space $\mathcal{K}$ is defined as:

- $\mathsf{SEnc}(K, m)$: given a key K and a message m, outputs a ciphertext ct.
- $\mathsf{SDec}(K, \mathsf{ct})$: given a key K and a ciphertext ct, outputs a plaintext.

The following must hold.

Correctness. For all messages m in the message space, $\Pr[\mathsf{SDec}(K, \mathsf{SEnc}(K, m)) = m] = 1$, where the probability is taken over $K \leftarrow_{\mathrm{R}} \mathcal{K}$.

One-time Security. For any PPT adversary $\mathcal{A}$, the following advantage is negligible:

$$\mathsf{Adv}^{\mathsf{OT}}_{\mathcal{SKE},\mathcal{A}}(\lambda) := \left| \Pr\left[b' = b : \begin{array}{l} (m_0, m_1) \leftarrow \mathcal{A}(1^\lambda) \\ K \leftarrow_{\mathrm{R}} \mathcal{K}, b \leftarrow_{\mathrm{R}} \{0,1\}, \mathsf{ct} = \mathsf{SEnc}(K, m_b) \\ b' \leftarrow \mathcal{A}(\mathsf{ct}) \end{array} \right] - \frac{1}{2} \right|.$$

Authenticated Encryption

Definition 3: Authenticated Encryption

An authenticated symmetric encryption (AE) with message-space $\mathcal{M}$ and key-space $\mathcal{K}$ consists of two polynomial-time deterministic algorithms $(\mathsf{Enc}_{\mathsf{AE}}, \mathsf{Dec}_{\mathsf{AE}})$:

- The encryption algorithm $\mathsf{Enc}_{\mathsf{AE}}(K, M)$ generates C, encryption of the message M with the secret key K.
- The decryption algorithm $\mathsf{Dec}_{\mathsf{AE}}(K, C)$, returns a message M or $\perp$.

The following must hold.

Perfect correctness. For all λ, for all $K \in \mathcal{K}$ and $m \in \mathcal{M}$, we have

$$\mathsf{Dec}_{\mathsf{AE}}(K, \mathsf{Enc}_{\mathsf{AE}}(K, M)) = m.$$

One-time Privacy and Authenticity. For any PPT adversary $\mathcal{A}$, we have:

$$\mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{A}}(\lambda) := \left| \Pr\left[b' = b : \begin{array}{l} K \leftarrow_{\mathrm{R}} \mathcal{K}; b \leftarrow_{\mathrm{R}} \{0,1\} \\ b' \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathsf{EncO}(\cdot,\cdot),\mathsf{DecO}(\cdot)}(1^\lambda, \mathcal{M}, \mathcal{K}) \end{array} \right] - 1/2 \right| = \mathsf{negl}(\lambda),$$

where $\mathsf{EncO}(m_0, m_1)$, on input two messages m_0 and m_1, returns $\mathsf{Enc}_{\mathsf{AE}}(K, m_b)$, and $\mathsf{DecO}(\phi)$ returns $\mathsf{Dec}_{\mathsf{AE}}(K, \phi)$ if $b = 0$, $\perp$ otherwise. $\mathcal{A}$ is allowed at most one call to each oracle EncO and DecO, and the query to DecO must be different from the output of EncO. $\mathcal{A}$ is also given the description of the key-space $\mathcal{K}$ as input.

Public-Key Encryption

Definition 4: Public-Key Encryption

A Public-Key Encryption (PKE) consists of the following PPT algorithms ($\mathsf{Param}_{\mathsf{PKE}}$, $\mathsf{Gen}_{\mathsf{PKE}}, \mathsf{Enc}_{\mathsf{PKE}}, \mathsf{Dec}_{\mathsf{PKE}}$):

- $\mathsf{Gen}_{\mathsf{PKE}}(1^\lambda)$: on input the security parameter, generates a pair of public and secret keys $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Enc}_{\mathsf{PKE}}(\mathsf{pk}, M)$: on input the public key and a message, returns a ciphertext ct.
- $\mathsf{Dec}_{\mathsf{PKE}}(\mathsf{pk}, \mathsf{sk}, \mathsf{ct})$: deterministic algorithm that returns a message M or $\perp$, where $\perp$ is a special rejection symbol.

The following must hold.

Perfect correctness. For all λ, we have

$$\Pr\left[\mathsf{Dec}_{\mathsf{PKE}}(\mathsf{pk}, \mathsf{sk}, \mathsf{ct}) = M \;\middle|\; \begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Gen}_{\mathsf{PKE}}(1^\lambda); \\ \mathsf{ct} \leftarrow_{\mathrm{R}} \mathsf{Enc}_{\mathsf{PKE}}(\mathsf{pk}, M)\end{array}\right] = 1.$$

Definition 5: Multi-ciphertext CCA security [BBM00]

A public-key encryption $\mathcal{PKE}$ is IND-CCA secure if for any PPT adversary $\mathcal{A}$, we have:

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{PKE},\mathcal{A}}(\lambda) := \left|\Pr\left[\; b = b' \;\middle|\; \begin{array}{l}\mathcal{C}_{\mathsf{Enc}} := \emptyset, b \leftarrow_{\mathrm{R}} \{0,1\} \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Gen}_{\mathsf{PKE}}(1^\lambda) \\ b' \leftarrow \mathcal{A}^{\mathsf{DecO}(\cdot),\mathsf{EncO}(\cdot,\cdot)}(1^\lambda, \mathsf{pk})\end{array}\right] - 1/2\right| = \mathsf{negl}(\lambda)$$

where:

- On input the pair of messages (m_0, m_1), $\mathsf{EncO}(m_0, m_1)$ returns $\mathsf{Enc}_{\mathsf{PKE}}(\mathsf{pk}, m_b)$ and sets $\mathcal{C}_{\mathsf{Enc}} := \mathcal{C}_{\mathsf{Enc}} \cup \{\mathsf{ct}\}$.
- $\mathsf{DecO}(\mathsf{ct})$ returns $\mathsf{Dec}_{\mathsf{PKE}}(\mathsf{pk}, \mathsf{sk}, \mathsf{ct})$ if $\mathsf{ct} \notin \mathcal{C}_{\mathsf{Enc}}$, $\perp$ otherwise.

Key-Encapsulation Mechanism

Definition 6: Tag-based KEM

A tag-based Key-Encapsulation Mechanism (KEM) for tag space $\mathcal{T}$ and key space $\mathcal{K}$ consists of three PPT algorithms ($\mathsf{Gen}_{\mathsf{KEM}}, \mathsf{Enc}_{\mathsf{KEM}}, \mathsf{Dec}_{\mathsf{KEM}}$):

- $\mathsf{Gen}_{\mathsf{KEM}}(1^\lambda)$: on input the security parameter, generates a pair of public and secret keys $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Enc}_{\mathsf{KEM}}(\mathsf{pk}, \tau)$: on input the public key and a tag τ, returns a pair (K, C) where K is a uniformly distributed symmetric key in $\mathcal{K}$ and C is a ciphertext, with respect to the tag $\tau \in \mathcal{T}$.
- $\mathsf{Dec}_{\mathsf{KEM}}(\mathsf{pk}, \mathsf{sk}, \tau, C)$: deterministic algorithm that returns a key $K \in \mathcal{K}$, or a special rejection symbol $\perp$ if it fails.

The following must hold.

Perfect correctness. For all λ, for all tags $\tau \in \mathcal{T}$, we have

$$\Pr\left[\mathsf{Dec}_{\mathsf{KEM}}(\mathsf{pk},\mathsf{sk},\tau,C) = K \;\middle|\; \begin{array}{l}(\mathsf{pk},\mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Gen}_{\mathsf{KEM}}(1^\lambda);\\ (K,C) \leftarrow_{\mathrm{R}} \mathsf{Enc}_{\mathsf{KEM}}(\mathsf{pk},\tau)\end{array}\right] = 1.$$

Definition 7: Multi-ciphertext PCA security [OP01].

A key encapsulation mechanism $\mathcal{KEM}$ is IND-PCE secure if for any adversary $\mathcal{A}$, we have:

$$\mathsf{Adv}^{\mathsf{IND\text{-}PCA}}_{\mathcal{KEM},\mathcal{A}}(\lambda) := \left|\Pr\left[\; b = b' \;\middle|\; \begin{array}{l}\mathcal{T}_{\mathsf{Enc}} = \mathcal{T}_{\mathsf{Dec}} := \emptyset, b \leftarrow_{\mathrm{R}} \{0,1\}\\ (\mathsf{pk},\mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Gen}_{\mathsf{KEM}}(1^\lambda)\\ b' \leftarrow \mathcal{A}^{\mathsf{DecO}(\cdot,\cdot,\cdot),\mathsf{EncO}(\cdot)}(1^\lambda,\mathsf{pk})\end{array}\right] - 1/2\right| = \mathsf{negl}(\lambda)$$

where:

- The decryption oracle $\mathsf{DecO}(\tau, C, \widehat{K})$ computes $K := \mathsf{Dec}_{\mathsf{KEM}}(\mathsf{pk},\mathsf{sk},\tau,C)$. It returns 1 if $\widehat{K} = K \wedge \tau \notin \mathcal{T}_{\mathsf{Enc}}$, 0 otherwise. Then it sets $\mathcal{T}_{\mathsf{Dec}} := \mathcal{T}_{\mathsf{Dec}} \cup \{\tau\}$.
- The oracle $\mathsf{EncO}(\tau)$ computes $(K, C) \leftarrow_{\mathrm{R}} \mathsf{Enc}_{\mathsf{KEM}}(\mathsf{pk}, \tau)$, sets $K_0 := K$ and $K_1 \leftarrow_{\mathrm{R}} \mathcal{K}$. If $\tau \notin \mathcal{T}_{\mathsf{Dec}} \cup \mathcal{T}_{\mathsf{Enc}}$, it returns (C, K_b), and sets $\mathcal{T}_{\mathsf{Enc}} := \mathcal{T}_{\mathsf{Enc}} \cup \{\tau\}$; otherwise it returns $\perp$.

Cryptographic Assumptions

Prime-Order Groups

Let GGen be a PPT algorithm that on input 1^λ returns a description $\mathcal{G} = (\mathbb{G}, q, P)$ of an additive cyclic group $\mathbb{G}$ of order p for a 2λ-bit prime p, whose generator is P.

We use implicit representation of group elements as introduced in [EHK+13]. For $a \in \mathbb{Z}_p$, define $[a] = aP \in \mathbb{G}$ as the *implicit representation* of a in $\mathbb{G}$. More generally, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_p^{n\times m}$ we define $[\mathbf{A}]$ as the implicit representation of $\mathbf{A}$ in $\mathbb{G}$:

$$[\mathbf{A}] := \begin{pmatrix} a_{11}P & \dots & a_{1m}P \\ & & \\ a_{n1}P & \dots & a_{nm}P \end{pmatrix} \in \mathbb{G}^{n\times m}$$

We will always use this implicit notation of elements in $\mathbb{G}$, i.e., we let $[a] \in \mathbb{G}$ be an element in $\mathbb{G}$. Note that from $[a] \in \mathbb{G}$ it is generally hard to compute the value a (discrete logarithm problem in $\mathbb{G}$). Obviously, given $[a], [b] \in \mathbb{G}$ and a scalar $x \in \mathbb{Z}_p$, one can efficiently compute $[ax] \in \mathbb{G}$ and $[a + b] \in \mathbb{G}$.

Definition 8: Computational Diffie-Hellman Assumption

The Computational Diffie-Hellman (CDH) assumption [DH76] states that, in a prime-order group $\mathcal{G} \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$, no PPT adversary can compute $[xy]$, from $[x]$ and $[y]$ for $x, y \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, with non-negligible success probability.

Equivalently, this assumption states it is hard to compute $[a^2]$ from $[a]$ for $a \leftarrow_{\mathrm{R}} \mathbb{Z}_p$. This comes from the fact that $4 \cdot [xy] = [(x + y)^2] - [(x - y)^2]$.

Pairing Groups

The use of pairing friendly elliptic curves for cryptography has been initiated by [BF01, BF03, Jou00, Jou04]. We refer to [GPS08] for further details on the use of pairing for cryptography. Let PGGen be a PPT algorithm that on input 1^λ returns a description $\mathcal{PG} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e)$ of asymmetric pairing groups where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are cyclic group of order p for a 2λ-bit prime p, P_1 and P_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable (non-degenerate) bilinear map. Define $P_T := e(P_1, P_2)$, which is a generator of $\mathbb{G}_T$. We again use implicit representation of group elements. For $s \in \{1, 2, T\}$ and $a \in \mathbb{Z}_p$, define $[a]_s = aP_s \in \mathbb{G}_s$ as the implicit representation of a in G_s . More generally, for a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{Z}_p^{n\times m}$ we define $[\mathbf{A}]_s$ as the implicit representation of $\mathbf{A}$ in $\mathbb{G}_s$:

$$[\mathbf{A}]_s := \begin{pmatrix} a_{11}P & \dots & a_{1m}P \\ & & \\ a_{n1}P & \dots & a_{nm}P \end{pmatrix} \in \mathbb{G}_s^{n\times m}$$

We will always use this implicit notation of elements in $\mathbb{G}_s$, i.e., we let $[a]_s \in \mathbb{G}_s$ be an element in $\mathbb{G}_s$. Note that from $[b]_T \in \mathbb{G}_T$, it is hard to compute the value $[b]_1 \in \mathbb{G}_1$ and $[b]_2 \in \mathbb{G}_2$ (pairing inversion problem). Obviously, given $[a]_s \in \mathbb{G}_s$ and a scalar $x \in \mathbb{Z}_p$, one can efficiently compute $[ax]_s \in \mathbb{G}_s$. Further, Given $[a]_1$, $[a]_2$, one can efficiently compute $[ab]_T$ using the pairing e. For two matrices $\mathbf{A}$, $\mathbf{B}$ with matching dimensions define $e([\mathbf{A}]_1, [\mathbf{B}]_2) := [\mathbf{AB}]_T$ in $\mathbb{G}_T$.

Matrix Diffie-Hellman

We recall the definitions of the Matrix Decision Diffie-Hellman (MDDH) assumption from [EHK+13].

Definition 9: Matrix Distribution

Let $k, \ell \in \mathbb{N}$, with $\ell > k$, and a prime p. We call $\mathcal{D}_{\ell,k}(p)$ a matrix distribution if it outputs in polynomial time matrices in $\mathbb{Z}_p^{\ell\times k}$ of full rank k and satisfying the following property:

$$\Pr[\mathsf{orth}(\mathbf{A}) \subseteq \mathsf{Span}(\mathbf{B})] = \frac{1}{\Omega(p)},$$

where $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_{\ell,k}(p)$. We write $\mathcal{D}_k(p) := \mathcal{D}_{k+1,k}(p)$.

Without loss of generality, we assume the first k rows of $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_{\ell,k}(p)$ form an invertible matrix. The $\mathcal{D}_{\ell,k}(p)$-Matrix Diffie-Hellman problem in a group $\mathbb{G}_s$ of order p, is to distinguish the two distributions $([\mathbf{A}]_s, [\mathbf{Aw}]_s)$ and $([\mathbf{A}]_s, [\mathbf{u}]_s)$ where $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_{\ell,k}(p)$, $\mathbf{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ and $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^\ell$.

Definition 10: $\mathcal{D}_{\ell,k}(p)$-Matrix Diffie-Hellman assumption, $\mathcal{D}_{\ell,k}(p)$-MDDH

Let $\mathcal{D}_{\ell,k}(p)$ be a matrix distribution. We say that the $\mathcal{D}_{\ell,k}(p)$-Matrix Diffie-Hellman ($\mathcal{D}_{\ell,k}(p)$-MDDH) assumption holds in a group $\mathbb{G}_s$, if for all PPT adversaries $\mathcal{A}$:

$$\mathsf{Adv}^{\mathcal{D}_{\ell,k}(p)\text{-MDDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\mathbb{G}_s, [\mathbf{A}]_s, [\mathbf{Aw}]_s) = 1] - \Pr[\mathcal{A}(\mathbb{G}_s, [\mathbf{A}]_s, [\mathbf{u}]_s) = 1]| = \mathsf{negl}(\lambda),$$

where the probability is taken over $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_{\ell,k}(p), \mathbf{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k, \mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^\ell$.

Let $Q \geq 1$. For $\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times Q}, \mathbf{U} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{\ell\times Q}$, we consider the Q-fold $\mathcal{D}_{\ell,k}(p)$-MDDH assumption in the group $\mathbb{G}$, which consists in distinguishing the distributions $([\mathbf{A}]_s, [\mathbf{AW}]_s)$ from $([\mathbf{A}]_s, [\mathbf{U}]_s)$. That is, a challenge for the Q-fold $\mathcal{D}_{\ell,k}(p)$-MDDH assumption consists of Q independent challenges of the $\mathcal{D}_{\ell,k}(p)$-MDDH assumption (with the same $\mathbf{A}$ but different randomness $\mathbf{w}$). As shown in [EHK+13] (and recalled in Lemma 1), the $\mathcal{D}_{\ell,k}(p)$-MDDH assumption is random self reducible, that is, it implies its Q-fold variant.

Definition 11: Q-fold $\mathcal{D}_{\ell,k}(p)$-MDDH assumption

Let $Q \geq 1$, and $\mathcal{D}_{\ell,k}(p)$ be a matrix distribution. We say that the Q-fold $\mathcal{D}_{\ell,k}(p)$-MDDH assumption holds in a group $\mathbb{G}_s$, if for all PPT adversaries $\mathcal{A}$:

$$\mathsf{Adv}^{Q\text{-}\mathcal{D}_{\ell,k}(p)\text{-MDDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\mathbb{G}_s, [\mathbf{A}]_s, [\mathbf{AW}]_s) = 1] - \Pr[\mathcal{A}(\mathbb{G}_s, [\mathbf{A}]_s, [\mathbf{U}]_s) = 1]| = \mathsf{negl}(\lambda),$$

where the probability is taken over $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_{\ell,k}(p), \mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times Q}, \mathbf{U} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{\ell\times Q}$.

Lemma 1: $\mathcal{D}_{\ell,k}(p)$-MDDH $\Rightarrow$ Q-fold $\mathcal{D}_{\ell,k}(p)$-MDDH [EHK+13]

Let $Q, \ell, k \in \mathbb{N}^*$ such that $\ell > k$, and a group $\mathbb{G}_s$ of prime order p. For any PPT adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ such that:

$$\mathsf{Adv}^{Q\text{-}\mathcal{D}_{\ell,k}(p)\text{-MDDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) \leq \begin{cases} Q \cdot \mathsf{Adv}^{\mathcal{D}_{\ell,k}(p)\text{-MDDH}}_{\mathbb{G}_s,\mathcal{B}}(\lambda) & \text{if } \ 1 \leq Q \leq \ell - k \\ (\ell - k) \cdot \mathsf{Adv}^{\mathcal{D}_{\ell,k}(p)\text{-MDDH}}_{\mathbb{G}_s,\mathcal{B}}(\lambda) + \frac{1}{p-1} & \text{if } \ Q > \ell - k \end{cases}$$

where the probability is taken over $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{U}_{\ell,k}(p), \mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times Q}, \mathbf{U} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{\ell\times Q}$.

For each $k \geq 1$, [EHK+13] specifies distributions $\mathcal{L}_k$, $\mathcal{SC}_k$, $\mathcal{C}_k$ (and others) over $\mathbb{Z}_p^{(k+1)\times k}$ such that the corresponding $\mathcal{D}_k(p)$-MDDH assumptions are generically secure in prime-order groups and form a hierarchy of increasingly weaker assumptions. $\mathcal{L}_k$-MDDH is the well known k-Linear assumption, denote as k-Lin for short, with 1-Lin = DDH, the decisional Diffie-Hellman assumption. In this work we are particularly interested in the uniform matrix distribution $\mathcal{U}_{\ell,k}(p)$.

Definition 12: Uniform distribution

Let $\ell, k \in \mathbb{N}$, with $\ell > k$, and p be a prime. We denote by $\mathcal{U}_{\ell,k}(p)$ the uniform distribution over all full-rank $\ell \times k$ matrices over $\mathbb{Z}_p$. Let $\mathcal{U}_k(p) := \mathcal{U}_{k+1,k}(p)$.

In [GHKW16], it shown that for any $\ell, k \in \mathbb{N}^*$ such that $\ell > k$, the $\mathcal{U}_{\ell,k}(p)$-MDDH assumption is equivalent to the $\mathcal{U}_k(p)$-MDDH assumption.

Lemma 2: $\mathcal{U}_{\ell,k}(p)$-MDDH $\Leftrightarrow$ $\mathcal{U}_k(p)$-MDDH [GHKW16]

Let $\ell, k \in \mathbb{N}^*$, with $\ell > k$, $s \in \{1, 2, T\}$, and a group $\mathbb{G}_s$ of prime-order p. For any PPT adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ (and vice versa) such that:

$$\mathsf{Adv}^{\mathcal{U}_{\ell,k}(p)\text{-MDDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G}_s,\mathcal{B}}(\lambda).$$

Together with Lemma 1, this implies the following corollary.

Corollary 1: $\mathcal{U}_k(p)$-MDDH $\Rightarrow$ Q-fold $\mathcal{U}_{\ell,k}(p)$-MDDH

Let $Q, \ell, k \in \mathbb{N}^*$, with $\ell > k$, and a group $\mathbb{G}_s$ of prime order p. For any PPT adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ such that:

$$\mathsf{Adv}^{Q\text{-}\mathcal{U}_{\ell,k}(p)\text{-}\mathrm{MDDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G}_s,\mathcal{B}}(\lambda) + \frac{1}{p-1}.$$

Among all possible matrix distributions $\mathcal{D}_{\ell,k}(p)$, the uniform matrix distribution $\mathcal{U}_k(p)$ is the hardest possible instance as stated in Lemma 3, so in particular k-Lin $\Rightarrow \mathcal{U}_k$-MDDH.

Lemma 3: $\mathcal{D}_{\ell,k}(p)$-MDDH $\Rightarrow \mathcal{U}_{\ell,k}(p)$-MDDH, [EHK+13]

Let $\mathcal{D}_{\ell,k}(p)$ be a matrix distribution, and $\mathbb{G}_s$ be a group of prime order p. For any PPT adversary $\mathcal{A}$, there exists a PPT adversary $\mathcal{B}$ such that:

$$\mathsf{Adv}^{\mathcal{U}_{\ell,k}\text{-}\mathrm{MDDH}}_{\mathbb{G}_s,\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathcal{D}_{\ell,k}(p)\text{-}\mathrm{MDDH}}_{\mathbb{G}_s,\mathcal{B}}(\lambda).$$

We now present a standard assumption in asymmetric pairing groups, known as the Decisional Bilinear Diffie Hellman (DBDH) assumption.

Definition 13: DBDH assumption

We say that the DBDH assumption holds in a pairing group $\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, p, P_1, P_2, e)$, if for all PPT adversaries $\mathcal{A}$:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{PG},\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\mathcal{PG}, [a]_1, [b]_1, [b]_2, [c]_2, [abc]_T) = 1] \\ - \Pr[\mathcal{A}(\mathcal{PG}, [a]_1, [b]_1, [b]_2, [c]_2, [s]_T) = 1]| \\ = \mathsf{negl}(\lambda),\end{aligned}$$

where the probability is taken over $a, b, c, s \leftarrow_{\mathrm{R}} \mathbb{Z}_p$.

As for the $\mathcal{D}_k(p)$-MDDH assumption, we define a Q-fold variant of the DBDh assumption, and prove its random self-reducibility.

Definition 14: Q-fold DBDH assumption

For any $Q \geq 1$, we say that the Q-fold DBDH assumption holds in a pairing group $\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, p, P_1, P_2, e)$, if for all PPT adversaries $\mathcal{A}$:

$$\begin{aligned}\mathsf{Adv}^{Q\text{-}\mathsf{DBDH}}_{\mathcal{PG},\mathcal{A}}(\lambda) := |\Pr\left[\mathcal{A}\left(\mathcal{PG}, [a]_1, [b]_1, [b]_2, \{[c_i]_2, [abc_i]_T\}_{i\in[Q]}\right) = 1\right] \\ - \Pr\left[\mathcal{A}\left(\mathcal{PG}, [a]_1, [b]_1, [b]_2, \{[c_i]_2, [s_i]_T\}_{i\in[Q]}\right) = 1\right]| \\ = \mathsf{negl}(\lambda),\end{aligned}$$

where the probability is taken over $a, b \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, and for all $i \in [Q]$, $c_i, s_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$.

Lemma 4: DBDH $\Rightarrow$ Q-fold DBDH

Let $Q \geq 1$, and a pairing group $\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, p, P_1, P_2, e)$. For any PPT adversary $\mathcal{A}$,

there exists a PPT adversary $\mathcal{B}$ such that:

$$\mathsf{Adv}^{Q\text{-}\mathsf{DBDH}}_{\mathcal{PG},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{DBDH}}_{\mathcal{PG},\mathcal{B}}(\lambda).$$

Proof of Lemma 4. Upon receiving a DBDH challenge $(\mathcal{PG}, [a]_1, [b]_1, [b]_2, [c]_2, [s]_T)$, $\mathcal{B}$ samples $\alpha_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^*, c_i' \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ computes $[c_i]_2 := [\alpha_i \cdot c]_2 + [c_i']_2$, $[s_i]_T := [\alpha_i \cdot s]_T + [c_i' \cdot ab]_T$ for all $i \in [Q]$, and gives the challenge $(\mathcal{PG}, [a]_1, [b]_1, [b]_2, \{[c_i]_2, [s_i]_T\}_{i\in[Q]})$ to $\mathcal{A}$. □

We now recall the definition another standard assumption in asymmetric pairing groups, first introduced in [BSW06].

Definition 15: 3-PDDH assumption

We say that the 3-party Decision Diffie-Hellman (3-PDDH) assumption holds in a pairing group $\mathcal{PG} \leftarrow \mathsf{PGGen}(1^\lambda)$ if for all PPT adversaries $\mathcal{A}$:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{3\text{-}PDDH}}_{\mathcal{PG},\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(\mathcal{PG}, [a]_1, [b]_2, [c]_1, [c]_2, [abc]_1) = 1] \\ - \Pr[\mathcal{A}(\mathcal{PG}, [a]_1, [b]_2, [c]_1, [c]_2, [d]_1) = 1]| \\ = \mathsf{negl}(\lambda)\end{aligned}$$

where the probability is taken over $a, b, c, d \leftarrow_{\mathrm{R}} \mathbb{Z}_p$.

Decisional Composite Residuosity

In [Pai99], the Decisional Composite Residuosity assumption is used to build a linearly homomorphic encryption scheme where the message is $\mathbb{Z}_N$, for an RSA modulus N.

Definition 16: Decisional Composite Residuosity assumption

Let $N = pq$, for prime numbers p, q. We say the Decisional Composite Residuosity (DCR) assumption holds if for all PPT adversaries $\mathcal{A}$:

$$\mathsf{Adv}^{\mathsf{DCR}}_{N,\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(N, z_0^N) = 1] - \Pr[\mathcal{A}(N, z) = 1]| = \mathsf{negl}(\lambda),$$

where the probability is taken over $z_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_N^*$, $z \leftarrow_{\mathrm{R}} \mathbb{Z}_{N^2}^*$.

Learning With Errors

We now provide minimal background on lattice-based cryptography.

Gaussian distributions. For any vector $\mathbf{c} \in \mathbb{R}^n$ and any parameter $\sigma \in \mathbb{R}_{>0}$, let $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) := \exp\left(\frac{-\pi\|\mathbf{x}-\mathbf{c}\|_2^2}{\sigma^2}\right)$ be the Gaussian function on $\mathbb{R}^n$ with center $\mathbf{c}$ and parameter σ. Let $\rho_{\sigma,\mathbf{c}}(\Lambda) := \sum_{\mathbf{x}\in\Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$ be the discrete integral of $\rho_{\sigma,\mathbf{c}}$ over Λ, and let $D_{\Lambda,\sigma,\mathbf{c}}$ be the discrete Gaussian distribution over Λ with center $\mathbf{c}$ and parameter σ. Namely, for all $\mathbf{y} \in \Lambda$,

$$D_{\Lambda,\sigma,\mathbf{c}}(\mathbf{y}) := \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}(\Lambda)}}.$$

To keep notation simple, we abbreviate $\rho_{\sigma,\mathbf{0}}$ and $D_{\Lambda,\sigma,\mathbf{0}}$ as ρ_σ and $D_{\Lambda,\sigma}$, respectively.

Definition 17: $\mathsf{LWE}_{q,\alpha,m}$ assumption

Let $q, m \in \mathbb{N}$ and $\alpha \in (0,1)$ be functions of the security parameter $\lambda \in \mathbb{N}$. We say that the $\mathsf{LWE}_{q,\alpha,m}$ assumption holds if for all PPT adversaries $\mathcal{A}$:

$$\mathsf{Adv}^{\mathsf{LWE}}_{q,\alpha,m,\mathcal{A}} := |\Pr[\mathcal{A}(q, \mathbf{A}, \mathbf{As}+\mathbf{e}) = 1] - \Pr[\mathcal{A}(q, \mathbf{A}, \mathbf{u}) = 1]| = \mathsf{negl}(\lambda),$$

where the probability is taken over $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{m\times\lambda}$, $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{\lambda}$, $\mathbf{e} \leftarrow D^m_{\mathbb{Z},\alpha q}$.

[Reg05] gives a quantum reduction from a worst-case lattice problem to LWE. We now present a so-called multi-hint extended LWE assumption, which is stronger than the latter in general. For some parameters, it has been shown in [ALS16] to be no stronger than LWE.

Definition 18: $\mathsf{mheLWE}_{q,\alpha,m,t,\mathcal{D}}$ assumption

Let $q, m, t \in \mathbb{N}$, $\alpha \in (0,1)$, $\mathcal{D}$ be a distribution over $\mathbb{Z}^{t\times m}$, all functions of the security parameter $\lambda \in \mathbb{N}$. We say that the the multi-hint extended LWE assumtpion, $\mathsf{mheLWE}_{q,\alpha,m,t,\mathcal{D}}$, holds, if for all PPT adversaries $\mathcal{A}$:

$$\mathsf{Adv}^{\mathsf{mheLWE}}_{q,\alpha,m,t,\mathcal{D},\mathcal{A}} := |\Pr[\mathcal{A}(q, \mathbf{A}, \mathbf{As}+\mathbf{e}, \mathbf{Z}, \mathbf{Ze}) = 1] - \Pr[\mathcal{A}(q, \mathbf{A}, \mathbf{Z}, \mathbf{Ze}, \mathbf{u}) = 1]| = \mathsf{negl}(\lambda),$$

where the probability is taken over $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{m\times\lambda}$, $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{\lambda}$, $\mathbf{Z} \leftarrow_{\mathrm{R}} \mathcal{D}$, $\mathbf{e} \leftarrow D^m_{\mathbb{Z},\alpha q}$.

Theorem 1: Reduction from $\mathsf{LWE}_{q,\alpha',m}$ to $\mathsf{mheLWE}_{q,\alpha,m}$[ALS16]

Let $n \geq 100$, $q \geq 2$, $t < n$, and $m \in \mathbb{N}$ such that $m \geq \Omega(n \log n)$ and $m \leq n^{O(1)}$. There exists $\xi \leq O(n^4 m^2 \log^{5/2}(n))$ and a distribution $\mathcal{D}$ over $\mathbb{Z}^{t\times m}$ such that the following statements hold:

- There is a reduction from $\mathsf{LWE}_{q,\alpha,m}$ in dimension $\lambda - t$ to $\mathsf{mheLWE}_{q,\alpha\xi,m,t,\mathcal{D}}$ that reduces the advantage by at most $2^{\Omega(t-n)}$.
- It is possible to sample from $\mathcal{D}$ in time polynomial in λ.
- Each entry of matrix $\mathcal{D}$ is an independent discrete Gaussian $\mathcal{D}_{i,j} = D_{\mathbb{Z},\sigma_{i,j},c_{i,j}}$ for some $c_{i,j} \in \{0,1\}$ and $\sigma_{i,j} \geq \Omega(mn \log m)$.
- With probability at least $1 - n^{-\omega(1)}$, all rows from a sample of $\mathcal{D}$ have norms at most ξ.

Definitions for Single-Input Functional Encryption

We now proceed to give definitions of functional encryption, originally given in [O'N10, BSW11].

Definition 19: Functional Encryption

A functionality F defined over $(\mathcal{K}, \mathcal{X})$ is a function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Z}$. The set $\mathcal{K}$ is called the key space, the set $\mathcal{X}$ is called the message space, and $\mathcal{Z}$ is called the output space. A functional encryption scheme consists of the following PPT algorithms:

- $\mathsf{GSetup}(1^\lambda, F)$: on input the security parameter λ and a functionality F, outputs

global public key gpk.

- $\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F)$: on input the security parameter λ, the global public key gpk, and a functionality F, outputs an encryption key ek, and a master secret key msk.
- $\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, x)$: on the global public parameters gpk, an encryption key ek, and a message $x \in \mathcal{X}$, outputs a ciphertext ct.
- $\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, k)$: on input the global public key gpk, a master secret key msk and a key $k \in \mathcal{K}$, outputs a decryption key dk_k.
- $\mathsf{Dec}(\mathsf{gpk}, \mathsf{dk}_k, \mathsf{ct})$: on input the global public key gpk, a decryption key dk_k and a ciphertext ct, outputs $z \in \mathcal{Z}$, or a special rejection symbol $\perp$ if it fails.

The scheme $\mathcal{FE}$ for functionality F is correct if for all $k \in \mathcal{K}$ and all $x \in \mathcal{X}$, we have:

$$\Pr\left[\begin{array}{r} \mathsf{gpk} \leftarrow \mathsf{GSetup}(1^\lambda, F); \\ (\mathsf{ek}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{gpk}, F); \\ \mathsf{dk}_k \leftarrow \mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, k); \\ \mathsf{Dec}(\mathsf{gpk}, \mathsf{dk}_k, \mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, x)) = F(k, x) \end{array}\right] = 1 - \mathsf{negl}(\lambda),$$

where the probability is taken over the coins of GSetup, Setup, KeyGen and Enc. The scheme is said to be public-key if ek is public, private-key otherwise.

Remark 1: Need for a global setup, multi-instance security

We split the setup, which is typically a single algorithm, into two algorithms: the global setup, that produces a global public key, and another setup that uses the global public key to produce the encryption key and master secret key. We do so since we will use many instances of FE as part of larger schemes, and they must share common public parameters, so as to ensure compatibility. For instance, in Chapter 4, we will use different instances of (single-input) FE, to build multi-input FE (defined below) with independent encryption and master secret keys, but working on the same group.

Security notions

Following [AGVW13], we may consider 8 security notions xx-yy-zzz where xx ∈ {one, many} refers to the number of challenge ciphertexts; yy ∈ {SEL, AD} refers to the fact that encryption queries are selectively or adaptively chosen; zzz ∈ {IND, SIM} refers to indistinguishability vs simulation-based security. We have the following trivial relations: many ⇒ one, AD ⇒ SEL, and the following standard relations: SIM ⇒ IND, and one-yy-IND ⇒ many-yy-IND, the latter in the public-key setting. We start by describing the strongest notion, namely, many-AD-SIM. We then present the weaker notions. All the definitions we present are in the multi-instance setting (see Remark 1).

Definition 20: multi-instance, many-AD-SIM secure FE

A functional encryption $\mathcal{FE} := (\mathsf{GSetup}, \mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ is many-AD-SIM secure for n instances, if there exists a PPT simulator $(\widetilde{\mathsf{GSetup}}, \widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}})$ such that for every PPT adversary $\mathcal{A}$ and every security parameter $\lambda \in \mathbb{N}$, the following two distributions are computationally indistinguishable:

Experiment $\mathbf{REAL}^{\mathcal{FE}}(1^\lambda, \mathcal{A})$:	Experiment $\mathbf{IDEAL}^{\mathcal{FE}}(1^\lambda, \mathcal{A})$:
$\mathsf{gpk} \leftarrow \mathsf{GSetup}(1^\lambda, F)$	$(\widetilde{\mathsf{gpk}}, \mathsf{td}) \leftarrow \widetilde{\mathsf{GSetup}}(1^\lambda, F)$
$\forall i \in [n]$: $(\mathsf{ek}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{gpk}, F)$	$\forall i \in [n]$: $(\widetilde{\mathsf{ek}}_i, \widetilde{\mathsf{msk}}_i) \leftarrow \widetilde{\mathsf{Setup}}(1^\lambda, \widetilde{\mathsf{gpk}}, F)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot),\mathsf{OKeygen}(\cdot,\cdot)}(\mathsf{gpk}, (\mathsf{ek}_i)_{i\in[n]})$	$\alpha \leftarrow \mathcal{A}^{\widetilde{\mathsf{OEnc}}(\cdot,\cdot),\widetilde{\mathsf{OKeyGen}}(\cdot,\cdot)}(\widetilde{\mathsf{gpk}}, (\widetilde{\mathsf{ek}})_{i\in[n]})$

The encryption keys (highlighted in gray) are only given to the adversary in a public-key scheme. The oracle $\mathsf{OKeygen}(i,k)$, on input an instance $i \in [n]$, and a key $k \in \mathcal{K}$, returns $\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}_i, k)$; $\mathsf{OEnc}(i,x)$, on input an instance $i \in [n]$, and a message $x \in \mathcal{X}$, returns $\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}_i, x)$; $\widetilde{\mathsf{OKeyGen}}(i,k)$, on input $i \in [n]$ and $k \in \mathcal{K}$, adds k to $\mathcal{Q}_{\mathsf{dk}}^{(i)}$ (the set of all decryption key queried for instance i, initially empty), and returns $\widetilde{\mathsf{KeyGen}}\left(\mathsf{td}, \widetilde{\mathsf{msk}}_i, k, \{F(k,x)\}_{x\in\mathcal{Q}_{\mathsf{ct}}^{(i)}}\right)$, where $\mathcal{Q}_{\mathsf{ct}}^{(i)}$ denotes the sets of queries to $\widetilde{\mathsf{OEnc}}$ (initially empty); $\widetilde{\mathsf{OEnc}}(i,x)$, on input $i \in [n]$ and $x \in \mathcal{X}$, adds x to $\mathcal{Q}_{\mathsf{ct}}^{(i)}$, and returns $\widetilde{\mathsf{Enc}}\left(\mathsf{td}, \widetilde{\mathsf{ek}}_i, \widetilde{\mathsf{msk}}_i, \{k, F(k,x)\}_{k\in\mathcal{Q}_{\mathsf{dk}}^{(i)}}\right)$.

Weaker notion of many-AD-SIM security. The definition above is stronger than the standard simulation-based definition, where the algorithm $\widetilde{\mathsf{Enc}}$ and $\widetilde{\mathsf{KeyGen}}$ take all the information leaked by the ideal functionality. In particular, to generate a simulated decryption key for key $k \in \mathcal{K}$ and instance $i \in [n]$, $\widetilde{\mathsf{KeyGen}}$ takes as input not only the values $\{F(k,x)\}_{x\in\mathcal{Q}_{\mathsf{ct}}^{(i)}}$, but also all the values $\{k', F(k',x)\}_{k'\in\mathcal{Q}_{\mathsf{dk}}^{(i)}, x\in\mathcal{Q}_{\mathsf{ct}}^{(i)}}$, for keys k' for which decryption keys were previously issued. The same applies to the algorithm $\widetilde{\mathsf{Enc}}$. We choose to work with the stronger simulation definition above, for simplicity, since the schemes presented in this work achieve it anyway.

We now consider the indistinguishability variant of the previous notion.

Definition 21: multi-instance, many-AD-IND secure FE

A functional encryption scheme $\mathcal{FE} := (\mathsf{GSetup}, \mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$, is many-AD-IND secure for n instance if for every stateful PPT adversary $\mathcal{A}$, we have:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{A},n}(\lambda) &= \left|\Pr\left[\mathsf{AD\text{-}IND}_0^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{A}) = 1\right] - \Pr\left[\mathsf{AD\text{-}IND}_1^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{A}) = 1\right]\right| \\ &= \mathsf{negl}(\lambda),\end{aligned}$$

where the experiments are defined for $\beta \in \{0,1\}$ as follows:

Experiment $\mathsf{AD\text{-}IND}_\beta^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{A})$:
$\mathsf{gpk} \leftarrow \mathsf{GSetup}(1^\lambda, F)$
$\forall i \in [n] : (\mathsf{ek}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{gpk}, F)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot),\mathsf{OKeygen}(\cdot,\cdot)}\left(\mathsf{gpk}, (\mathsf{ek}_i)_{i\in[n]}\right)$
Output: α

The encryption key (highlighted in gray) is only given to the adversary in a public-key scheme. The oracle $\mathsf{OEnc}(i, (x^0, x^1))$, on input an instance $i \in [n]$ and a pair of messages $(x^0, x^1) \in \mathcal{X}^2$, returns $\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}_i, x^\beta)$. The oracle $\mathsf{OKeygen}(i,k)$, on input an instance $i \in [n]$ and a key $k \in \mathcal{K}$, returns $\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}_i, k)$. For any instance $i \in [n]$, the

queries k of adversary $\mathcal{A}$ to $\mathsf{OKeygen}(i,\cdot)$ must satisfy the following condition, for all queries (x^0, x^1) to $\mathsf{OEnc}(i,\cdot)$: $F(k, x^0) = F(k, x^1)$. That is, for a given instance $i \in [n]$, the decryption keys should not be able to distinguish any challenge message pairs.

Clearly, single-instance (that is, $n = 1$ in the above definition) is implied by the multi-instance security ($n > 1$). By a standard hybrid argument over the n instances, the converse is also true.

Lemma 5: Single-instance implies multi-instance security

For any scheme $\mathcal{FE}$, PPT adversary $\mathcal{A}$, xx $\in$ {many,one}, yy $\in$ {AD,SEL}, there exists a PPT adversary $\mathcal{B}$ such that for all security parameters λ:

$$\mathsf{Adv}^{\text{xx-yy-IND}}_{\mathcal{FE},\mathcal{A},n}(\lambda) \leq n \cdot \mathsf{Adv}^{\text{xx-yy-IND}}_{\mathcal{FE},\mathcal{B},1}(\lambda).$$

Proof of Lemma 5 (sketch). We only give a high-level sketch of the proof, which uses a standard hybrid argument over the n instances. Namely, we define n games, where the i'th game answers all the queries $(j, (x^0, x^1))$ to OEnc for $j \leq i$ with $\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}_j, x^1)$, and for $j > i$, answers with $\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}_i, x^0)$. To transition from hybrid i to $i+1$, we use the single instance security for the queries to OEnc on the $i+1$'st instance. The rest can be simulated simply by sampling $(\mathsf{ek}_j, \mathsf{msk}_j) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{gpk}, F)$, for all $j \neq i+1$, since gpk is known. □

We consider the following weaker notions of security.

One ciphertext, one-yy-zzz: the adversary $\mathcal{A}$ can only query its encryption oracle OEnc once per instance $i \in [n]$.

Selective security, xx-SEL-zzz: the adversary $\mathcal{A}$ must send its queries to OEnc beforehand, that is, before receiving the gpk (and the $(\mathsf{ek}_i)_{i\in[n]}$, in the public-key setting) from the experiment, and before querying $\mathsf{OKeygen}$.

These weaker security notions may appear artificial, and indeed, the desirable security notions are many-AD-IND or many-AD-SIM, both of which capture natural attacks. However, they are still useful as a first step towards many-yy-IND security. For instance, as explained below, in the public-key setting, one-yy-IND implies many-yy-IND. Also, using a guessing argument (see, for instance, [BB04], in the context of Identity-Based Encryption), one can turn any selectively-secure scheme into an adaptively-secure scheme, albeit with an exponential security loss.

Remark 2: Semi-adaptive security

In the context of Attribute-Based Encryption (which is a particular case of Functional Encryption), [CW14] put forth the notion of semi-adaptive security, where the adversary has to send its challenge messages before querying any decryption keys, but after receiving the public key from its experiment. This notion lies in between adaptive and selective security, namely, it is implied by the former, and implies the latter. In [GKW16], the authors give a generic transformation that turns any selectively-secure FE into a semi-adaptive secure FE, only using Public-Key Encryption.

It is also known that one-xx-IND security implies many-xx-IND security, in the public-key setting.

Lemma 6: one-xx-IND security implies many-xx-IND security

For any public-key scheme $\mathcal{FE}$, PPT adversary $\mathcal{A}$, xx $\in$ {AD,SEL}, there exists a PPT adversary $\mathcal{B}$ such that for all security parameters λ:

$$\mathsf{Adv}^{\mathsf{many\text{-}xx\text{-}IND}}_{\mathcal{FE},\mathcal{A},n}(\lambda) \leq Q \cdot \mathsf{Adv}^{\mathsf{one\text{-}xx\text{-}IND}}_{\mathcal{FE},\mathcal{B},n}(\lambda),$$

where Q is an upper bound on the number of queries to $\mathsf{OEnc}(i,\cdot)$, for any $i \in [n]$.

Proof of Lemma 6 (sketch). We only give a high-level sketch of the proof, which uses a standard hybrid argument over the challenge ciphertexts. Namely, we define Q games, where the i'th game answers the first i query to $\mathsf{OEnc}(j,(x^0,x^1))$ for any $j \in [n]$, with $\mathsf{Enc}(\mathsf{gpk},\mathsf{ek}_j,x^1)$, and the last queries with $\mathsf{Enc}(\mathsf{gpk},\mathsf{ek}_j,x^1)$. To transition from hybrid i to $i+1$'st, we use the one-yy-IND security to switch the $i+1$'st query from $\mathsf{Enc}(\mathsf{gpk},\mathsf{ek}_j,x^0)$ to $\mathsf{Enc}(\mathsf{gpk},\mathsf{ek}_j,x^1)$ simultaneously for all instances $j \in [n]$. The other queries can be addressed using the public encryption keys ek_j. □

Definitions for Multi-Input Functional Encryption

We recall the definition of multi-input functional encryption, that has been first introduced in [GGG+14]. It generalizes functional encryption as follows. In a multi-input functional encryption, encryption is split among n different users, or input slots; each of which encrypts separately an input x_i independently, without any interaction. Then, given a functional decryption key for an n-ary function f, decryption operates on all the n independently generated ciphertexts and recovers $f(x_1,\cdots,x_n)$. This generalization is useful in applications where the data to encrypt is distributed among users that do not trust each other; or when the same user wants to encrypt data at different point in time (without memorizing the randomness used for prior encryption).

Definition 22: Multi-input Function Encryption

Let $\{F_n\}_{n\in\mathbb{N}}$ be a set of functionality where for each $n \in \mathbb{N}$, F_n defined over $(\mathcal{K}_n,\mathcal{X}_1,\cdots,\mathcal{X}_n)$ is a function $F_n : \mathcal{K}_n \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Z}$. Each $i \in [n]$ is called an input slot. The key space $\mathcal{K}_n$, depends on the arity n. A multi-input functional encryption scheme $\mathcal{MIFE}$ for the set of functionality $\{F_n\}_{n\in\mathbb{N}}$ consists of the following algorithms:

- $\mathsf{Setup}(1^\lambda, F_n)$: on input the security parameter λ and a functionality F_n, outputs a public key pk, encryption keys ek_i for each input slot $i \in [n]$, and a master secret key msk.
- $\mathsf{Enc}(\mathsf{pk},\mathsf{ek}_i,x_i)$: on input the public key pk, encryption key ek_i for the input slot $i \in [n]$, and a message $x_i \in \mathcal{X}_i$, outputs a ciphertext ct. We assume that each ciphertext has an associated index i, which denotes what slot this ciphertext can be used for.
- $\mathsf{KeyGen}(\mathsf{pk},\mathsf{msk},k)$: on input the public key pk, the master secret key msk and a function $k \in \mathcal{K}_n$, outputs a decryption key dk_k.
- $\mathsf{Dec}(\mathsf{pk},\mathsf{dk}_k,\mathsf{ct}_1,\ldots,\mathsf{ct}_n)$: on input the public key pk, a decryption key dk_k and n ciphertexts, outputs $z \in \mathcal{Z}$, or a sepcial rejection symbol $\perp$ if it fails.

The scheme $\mathcal{MIFE}$ is correct if for all $k \in \mathcal{K}_n$ and all $x_i \in \mathcal{X}_i$ for $i \in [n]$, we have:

$$\Pr\left[\begin{array}{r}(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, F_n);\\ \mathsf{dk}_k \leftarrow \mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, k);\\ \mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_k, \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_1, x_1), \ldots, \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_n, x_n)) = F_n(k, x_1, \ldots, x_n)\end{array}\right] = 1-\mathsf{negl}(\lambda),$$

where the probability is taken over the coins of Setup, KeyGen and Enc.

The scheme is public-key if $\mathsf{ek}_i = \emptyset$, that is, the encryption algorithm Enc only requires the public pk to encrypt messages. It is private-key otherwise.

Security notions

As for the case of single-input FE, we may consider 8 security notions xx-yy-zzz where xx ∈ {one, many} refers to the number of challenge ciphertexts; yy ∈ {SEL, AD} refers to the fact that encryption queries are selectively or adaptively chosen; zzz ∈ {IND, SIM} refers to indistinguishability vs simulation-based security. Since simulation-security is impossible in general as proven in [BSW11], we will restrict ourselves to indistinguishability-based security definition. We defer to [BLR+15] for a description of simulation-based security definitions. Although the multi-instance setting for single-input FE is relevant to this work, the multi-instance for the multi-input setting is not. For simplicity, we focus on the single-instance setting here.

One novelty compared to the single-input setting is that some input slots can collude, and should not be able to break the security of the encryption for the other slots. This is captured, in the security game, by the oracle OCorrupt, that on input a slot $i \in [n]$, returns the corresponding encryption key ek_i. The public-key setting essentially corresponds to the case where all ek_i are public. In particular, the adversary can encrypt any message for any slot, and decrypt them with the challenge ciphertexts for the other slots. This inherent leakage of information (it is allowed for an adversary to learn this information, by correctness of the MIFE) is captured by the **Condition 1** in the many-AD-IND security game.

Definition 23: many-AD-IND secure MIFE

A multi-input functional encryption $\mathcal{MIFE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the set of functionalities $\{F_n\}_{n\in\mathbb{N}}$, is many-AD-IND secure if for every stateful PPT adversary $\mathcal{A}$, we have:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) &= \left|\Pr\left[\mathsf{AD\text{-}IND}_0^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\right] - \Pr\left[\mathsf{AD\text{-}IND}_1^{\mathcal{MIFE}}(1^\lambda, \mathcal{A}) = 1\right]\right| \\ &= \mathsf{negl}(\lambda),\end{aligned}$$

where the experiments are defined for all $\beta \in \{0, 1\}$ as follows:

Experiment $\mathsf{AD\text{-}IND}_\beta^{\mathcal{MIFE}}(1^\lambda, \mathcal{A})$:
$(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, F_n)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{OKeygen}(\cdot),\mathsf{OCorrupt}(\cdot)}(\mathsf{pk})$
Output: α

The oracle OEnc, on input (i, x_i^0, x_i^1), returns $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, x_i^\beta)$. For all input slots $i \in [n]$, we denote by $\mathcal{Q}_i$ the set of queries to OEnc for slot i, and Q_i the size of $\mathcal{Q}_i$. The oracle OKeygen, on input $k \in \mathcal{K}_n$, returns $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, k)$. The oracle OCorrupt, on input $i \in [n]$, returns ek_i. We denote by $\mathcal{CS} \subseteq [n]$ the set of corrupted slots. The queries of adversary $\mathcal{A}$ must satisfy the following condition.

Condition 1:

- For all $i \in \mathcal{CS}$, all $(x_i^0, x_i^1) \in \mathcal{Q}_i$, we have $x_i^0 = x_i^1$.
- $\mathcal{A}$ only makes queries k to $\mathsf{OKeygen}(\cdot)$ satisfying

$$F_n(k, x_1^0, \ldots, x_n^0) = F_n(k, x_1^1, \ldots, x_n^1)$$

for all possible vectors $(x_i^b)_{i\in[n],b\in\{0,1\}}$, where for all $i \in [n]$, we have: either $(x_i^0, x_i^1) \in \mathcal{Q}_i$, or ($i \in \mathcal{CS}$ and $x_i^0 = x_i^1$).

If the condition is not satisfied, the experiment outputs 0 instead of α.

Remark 3: Winning condition

Note that **Condition 1** is in general not efficiently checkable because of the combinatorial explosion in the restriction of the queries.

We consider the following weaker security notions.

One ciphertext, one-yy-IND: the adversary $\mathcal{A}$ can only query OEnc once per input slot $i \in [n]$, that is, $Q_i \leq 1$ for all $i \in [n]$.

Selective security, xx-SEL-IND: the adversary $\mathcal{A}$ must send its challenge $\{x_i^{j,b}\}_{b\in\{0,1\},i\in[n],j\in[Q_i]}$ beforehand, that is, before receiving the public key from the experiment, and before querying $\mathsf{OKeygen}$ or $\mathsf{OCorrupt}$.

Static corruption, xx-yy-IND-static: the adversary $\mathcal{A}$ must send its queries to $\mathsf{OCorrupt}$ before any other query.

Zero decryption keys, xx-yy-IND-zero: the adversary $\mathcal{A}$ does not query $\mathsf{OKeygen}$.

Extra condition, xx-yy-IND-weak: the adversary $\mathcal{A}$ must send at least one challenge per slot that is not corrupted, that is, for all $i \in [n] \setminus \mathcal{CS}$, we have: $Q_i \geq 1$.

These weaker security notions may appear to impose unrealistic restrictions on the adversary. As for the case of single-input FE, it is useful to start building a simpler scheme which only satisfies a weak security notion, then turn it into a many-AD-IND secure scheme. In fact, we show how to generically transform any xx-yy-IND-weakly and xx-yy-IND-zero secure MIFE into a full-fledged xx-yy-IND secure MIFE, only using symmetric-key encryption.

Removing the extra condition generically

Here we show how to remove the extra condition from any multi-input FE that is both xx-yy-IND-weak and xx-yy-IND-zero secure, for any xx $\in$ {one,many}, and yy $\in$ {AD,SEL}, using an extra layer of symmetric-key encryption. A similar approach is used in [AGRW17]. Namely, [AGRW17] uses a symmetric key to encrypt the original ciphertexts. The symmetric key is shared across users, and the i'th share is given as part of any ciphertext for input slot $i \in [n]$. Thus, when ciphertexts are known for all slots $i \in [n]$, the decryption recovers all shares of the symmetric key, and decrypt the outer layer, to get the original ciphertext. The rest of decryption is performed as in the original multi-input FE.

The problem with this approach is that the encryption algorithm needs to know the symmetric key (and not just a share of it). Thus, corrupting one input slot allows the adversary

to recover the entire symmetric key, and break the security of the scheme. Such problem did not arise in [AGRW17], which does not consider corruptions of input slots. To circumvent this issue, as in [DOT18], we use the symmetric key to encrypt the functional decryption keys, instead of encrypting the ciphertexts. Each encryption key ek_i for input slot $i \in [n]$ contains the i'th share of the symmetric key, but the full symmetric key is only needed by the key generation algorithm, which knows msk. If one share is missing, all the functional decryption keys are random. We conclude the security proof using the security of the overall multi-input FE when zero functional decryption keys are queried.

$\underline{\mathsf{Setup}(1^\lambda, F_n)}$:
$(\mathsf{pk}', \mathsf{msk}', (\mathsf{ek}'_i)_{i\in[n]}) \leftarrow \mathsf{Setup}'(1^\lambda, F_n)$
$\mathsf{K} \leftarrow_{\mathrm{R}} \mathcal{K}$
$k_1, \ldots, k_{n-1} \leftarrow_{\mathrm{R}} \{0,1\}^\lambda$, $k_n = \left(\bigoplus_{i\in[n-1]} k_i\right) \oplus \mathsf{K}$
$\mathsf{pk} := \mathsf{pk}', \mathsf{msk} := (\mathsf{msk}', \mathsf{K}), \forall i \in [n] : \ \mathsf{ek}_i := (\mathsf{ek}'_i, k_i)$
return $\left(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}\right)$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, x_i)}$:
parse $\mathsf{ek}_i = (\mathsf{ek}'_i, k_i)$
$\mathsf{ct}' \leftarrow \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x_i)$
return (k_i, ct')

$\underline{\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, k)}$:
parse $\mathsf{msk} = (\mathsf{msk}', \mathsf{K})$
$\mathsf{dk}'_k \leftarrow \mathsf{KeyGen}'(\mathsf{pk}', \mathsf{msk}', k)$
$\mathsf{dk}_k \leftarrow \mathsf{Enc}_{\mathsf{SE}}(\mathsf{K}, \mathsf{dk}'_k)$
return dk_k

$\underline{\mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_k, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)}$:
parse $\{\mathsf{ct}_i = (k_i, \mathsf{ct}'_i)\}_{i\in[n]}$
$\mathsf{K} \leftarrow \bigoplus_{i\in[n]} k_i$
$\mathsf{dk}'_k \leftarrow \mathsf{Dec}_{\mathsf{SE}}(\mathsf{K}, \mathsf{dk}_k)$
return $\mathsf{Dec}'(\mathsf{dk}'_k, \mathsf{ct}'_1, \ldots, \mathsf{ct}'_n)$.

Figure 2.1: Compiler from any $\mathcal{MIFE}' := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ with xx-yy-weak and xx-yy-zero security to the $\mathcal{MIFE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ with xx-yy security. Here, $(\mathsf{Enc}_{\mathsf{SE}}, \mathsf{Dec}_{\mathsf{SE}})$ is a symmetric key encryption scheme with key space $\mathcal{K}$ as defined in Definition 2.

Theorem 2: Removing the extra condition

Let $\mathcal{MIFE}'$ be a xx-yy-IND-weak and xx-yy-IND-zero secure MIFE, for any xx $\in$ {one,many}, and any yy $\in$ {AD,SEL}, and $(\mathsf{Gen}, \mathsf{Enc}_{\mathsf{SE}}, \mathsf{Dec}_{\mathsf{SE}})$ be a symmetric encryption scheme. The scheme $\mathcal{MIFE}$ defined in Figure 2.1 is xx-yy-IND secure.

Proof of Theorem 2 (sketch). We consider two cases:

- Case 1: there exists some $i \in [n]$ for which $Q_i = 0$, and $i \notin \mathcal{CS}$. That is, the adversary never queries OEnc or $\mathsf{OCorrupt}$ on slot i. Here, k_i and thus K is perfectly hidden from the adversary. Then, by semantic security of $(\mathsf{Gen}_{\mathsf{SE}}, \mathsf{Enc}_{\mathsf{SE}}, \mathsf{Dec}_{\mathsf{SE}})$, the decryption keys are pseudo-random. We conclude using the xx-yy-IND-zero security of $\mathcal{MIFE}'$.
- Case 2: for all i, $Q_i \geq 1$. Here, security follows immediately from the xx-yy-IND-weak security of the underlying $\mathcal{MIFE}'$.

□

Definitions for Multi-Client Functional Encryption

We now present the definition of multi-client functional encryption (MCFE), originally given in [GGG+14], which enhances multi-input functional encryption in the following way. In MCFE, the encryption algorithm takes as an additional input a label (typically a time-stamp), and ciphertexts from different input slots can only be combined when they are encrypted under the same label. The limits the leakage of information from the encrypted messages. Multi-input functional encryption corresponds to the case where every message is encrypted under the same label.

Definition 24: Multi-Client Function Encryption

Let $\{F_n\}_{n\in\mathbb{N}}$ be a set of functionality where for each $n \in \mathbb{N}$, F_n defined over $(\mathcal{K}_n, \mathcal{X}_1, \cdots, \mathcal{X}_n)$ is a function $F_n : \mathcal{K}_n \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Z}$. Each i is called an input slot. The key space $\mathcal{K}_n$, depends on the arity n. A multi-client functional encryption scheme $\mathcal{MCFE}$ for the set of functionality $\{F_n\}_{n\in\mathbb{N}}$ consists of the following algorithms:

- $\mathsf{Setup}(1^\lambda, F_n)$: on input the security parameter λ and a functionality F_n, outputs a public key pk, encryption keys ek_i for each input slot $i \in [n]$, and a master secret key msk.
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, x_i, \ell)$: on input the public key pk, encryption key ek_i for the input slot $i \in [n]$, a message $x_i \in \mathcal{X}_i$, and a label ℓ, it outputs a ciphertext ct.
- $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, k)$: on input the public key pk, the master secret key msk and a function $k \in \mathcal{K}_n$, it outputs a decryption key dk_k.
- $\mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_k, \mathsf{ct}_1, \ldots, \mathsf{ct}_n, \ell)$: on input the public key pk, a decryption key dk_k, n ciphertexts and a label ℓ, outputs $z \in \mathcal{Z}$, or a special rejection symbol $\perp$ if it fails.

The scheme $\mathcal{MCFE}$ is correct if for all $k \in \mathcal{K}_n$, all $x_i \in \mathcal{X}_i$ for $i \in [n]$, and all label ℓ, we have:

$$\Pr\left[\begin{array}{r}(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, F_n);\\ \mathsf{dk}_k \leftarrow \mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, k);\\ \mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_k, \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_1, x_1, \ell), \ldots, \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_n, x_n, \ell), \ell) = F_n(k, x_1, \ldots, x_n)\end{array}\right] = 1 - \mathsf{negl}(\lambda),$$

where the probability is taken over the coins of Setup, KeyGen and Enc.

The scheme is public-key if $\mathsf{ek}_i = \emptyset$, that is, the encryption algorithm Enc only requires the public pk to encrypt messages. It is private-key otherwise.

Definition 25: many-AD-IND secure MCFE

A multi-client functional encryption $\mathcal{MCFE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the set of functionalities $\{F_n\}_{n\in\mathbb{N}}$, is many-AD-IND secure if for every stateful PPT adversary $\mathcal{A}$, we have:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{MCFE},\mathcal{A}}(\lambda) &= \left|\Pr\left[\mathsf{AD\text{-}IND}_0^{\mathcal{MCFE}}(1^\lambda, \mathcal{A}) = 1\right] - \Pr\left[\mathsf{AD\text{-}IND}_1^{\mathcal{MCFE}}(1^\lambda, \mathcal{A}) = 1\right]\right| \\ &= \mathsf{negl}(\lambda),\end{aligned}$$

where the experiments are defined for $\beta \in \{0, 1\}$ as follows:

Experiment $\mathsf{AD\text{-}IND}_{\beta}^{\mathcal{MCFE}}(1^{\lambda}, \mathcal{A})$:
$(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^{\lambda}, F_n)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{OKeygen}(\cdot),\mathsf{OCorrupt}(\cdot)}(\mathsf{pk})$
Output: α

The oracle OEnc, on input $(i, (x_i^0, x_i^1), \ell)$, returns $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, x_i^{\beta}, \ell)$. For all input slots $i \in [n]$, and label ℓ, we denote by $\mathcal{Q}_{i,\ell}$ the set of queries to OEnc for slot i and label ℓ, and $Q_{i,\ell}$ the size of $\mathcal{Q}_{i,\ell}$. The oracle $\mathsf{OKeygen}$, on input $k \in \mathcal{K}_n$, returns $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, k)$. The oracle $\mathsf{OCorrupt}$, on input $i \in [n]$, returns ek_i. We denote by $\mathcal{CS} \subseteq [n]$ the set of corrupted slots. The queries of adversary $\mathcal{A}$ must satisfy the following condition.

Condition 1:

- For all $i \in \mathcal{CS}$, all labels ℓ, all $(x_i^0, x_i^1) \in \mathcal{Q}_{i,\ell}$, we have $x_i^0 = x_i^1$.
- $\mathcal{A}$ only makes queries k to $\mathsf{OKeygen}(\cdot)$ satisfying

$$F_n(k, x_1^0, \ldots, x_n^0) = F_n(k, x_1^1, \ldots, x_n^1)$$

  for all labels ℓ and all vectors $(x_i^b)_{i\in[n],b\in\{0,1\}}$ such that for all $i \in [n]$, we have: either $(x_i^0, x_i^1) \in \mathcal{Q}_{i,\ell}$, or ($i \in \mathcal{CS}$ and $x_i^0 = x_i^1$).

If the condition is not satisfied, the experiment outputs 0 instead of α.

We consider the following weaker security notions.

one-AD-IND security: the adversary $\mathcal{A}$ can only query OEnc once for each input slot $i \in [n]$ and label ℓ, that is, $Q_{i,\ell} \leq 1$ for all $i \in [n]$ and all labels ℓ.

xx-AD-IND-weak security: The queries of adversary $\mathcal{A}$ must satisfy the following **extra condition**: if there exists a label ℓ and a slot $i \in [n]$ such that $(x_i^0, x_i^1) \in \mathcal{Q}_{i,\ell}$ with $x_i^0 \neq x_i^1$, then for all $j \in [n]$, we must have either $j \in \mathcal{CS}$ or $Q_{j,\ell} > 1$. Intuitively, this condition restricts the adversary to use challenge ciphertexts for *all* input slots $i \in [n]$ for a given label ℓ. In fact, **Condition 1** does not consider the information that may be leaked from partial ciphertexts, since for all $i \in [n]$, we must have either a query $(x_i^0, x_i^1) \in \mathcal{Q}_{i,\ell}$, or $i \in \mathcal{CS}$. The **extra condition** simply prevents the occurrence of such partial ciphertexts in the security game. This artificial notion will be a useful stepping stone towards full-fledged xx-AD-IND security.

We now present a decentralized variant of multi-client functional encryption, where the generation of functional decryption keys does not require a trusted third party: the master secret key is split across users into several keys; each user can generate a share of the functional decryption keys, without any interaction; then the shares can be publicly combined to obtain a functional decryption key.

Definition 26: Decentralized Multi-Client Function Encryption

Let $\{F_n\}_{n\in\mathbb{N}}$ be a set of functionality where for each $n \in \mathbb{N}$, F_n defined over $(\mathcal{K}_n, \mathcal{X}_1, \cdots, \mathcal{X}_n)$ is a function $F_n : \mathcal{K}_n \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Z}$. Each i is called an input slot. The key space $\mathcal{K}_n$, depends on the arity n. A decentralized multi-client functional encryption scheme $\mathcal{DMCFE}$ for the set of functionality $\{F_n\}_{n\in\mathbb{N}}$ consists of the following algorithms:

- $\mathsf{Setup}(1^\lambda, F_n)$: on input the security parameter λ and a functionality F_n, outputs a public key pk, encryption keys ek_i for each input slot $i \in [n]$, and secret keys sk_i for each input slot $i \in [n]$.
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, x_i, \ell)$: on input the public key pk, encryption key ek_i for the input slot $i \in [n]$, a message $x_i \in \mathcal{X}_i$, and a label ℓ, it outputs a ciphertext ct.
- $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}_i, k)$: on input the public key pk, the secret key sk_i for slot $i \in [n]$, and a function $k \in \mathcal{K}_n$, it outputs a partial decryption key $\mathsf{dk}_{k,i}$.
- $\mathsf{KeyComb}(\mathsf{pk}, \{\mathsf{dk}_{k,i}\}_{i\in[n]}, k)$: on input the public key pk, n partial decryption keys keys, and a key k, it combines its input to produce a decryption key dk_k.
- $\mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_k, \mathsf{ct}_1, \ldots, \mathsf{ct}_n, \ell)$: on input the public key pk, a decryption key dk_k, n ciphertexts and a label ℓ, outputs $z \in \mathcal{Z}$, or a special rejection symbol $\perp$ if it fails.

The scheme $\mathcal{DMCFE}$ is correct if for all $k \in \mathcal{K}_n$, all $x_i \in \mathcal{X}_i$ for $i \in [n]$, and all label ℓ, we have:

$$\Pr\left[\begin{array}{r} (\mathsf{pk}, (\mathsf{ek}_i, \mathsf{sk}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, F_n); \\ \forall i \in [n] : \mathsf{dk}_{k,i} \leftarrow \mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}_i, k); \\ \mathsf{dk}_k \leftarrow \mathsf{KeyComb}(\mathsf{pk}, (\mathsf{dk}_{i,k})_{i\in[n]}, k); \\ \mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_k, \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_1, x_1, \ell), \ldots, \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_n, x_n, \ell), \ell) = F_n(k, x_1, \ldots, x_n) \end{array}\right] = 1 - \mathsf{negl}(\lambda),$$

where the probability is taken over the coins of Setup, KeyGen, $\mathsf{KeyComb}$ and Enc.

The scheme is public-key if $\mathsf{ek}_i = \emptyset$, that is, the encryption algorithm Enc only requires the public pk to encrypt messages. It is private-key otherwise.

We now present the many-AD-IND security notion for decentralized multi-client functional encryption. The difference with centralized multi-client functional encryption is that the shares of the functional decryption keys can be corrupted, instead of the functional decryption keys themselves. The oracle $\mathsf{OCorrupt}$ also give out the secret key sk_i in addition of ek_i, when queried on input slot $i \in [n]$.

Definition 27: many-AD-IND secure DMCFE

A decentralized multi-client functional encryption $\mathcal{DMCFE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{KeyComb}, \mathsf{Dec})$ for the set of functionalities $\{F_n\}_{n\in\mathbb{N}}$, is many-AD-IND secure if for every stateful PPT adversary $\mathcal{A}$, we have:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{DMCFE},\mathcal{A}}(\lambda) &= \left|\Pr\left[\mathsf{AD\text{-}IND}_0^{\mathcal{DMCFE}}(1^\lambda, \mathcal{A}) = 1\right] - \Pr\left[\mathsf{AD\text{-}IND}_1^{\mathcal{DMCFE}}(1^\lambda, \mathcal{A}) = 1\right]\right| \\ &= \mathsf{negl}(\lambda),\end{aligned}$$

where the experiments are defined for $\beta \in \{0, 1\}$ as follows:

Experiment $\mathsf{AD\text{-}IND}_{\beta}^{\mathcal{MCFE}}(1^{\lambda}, \mathcal{A})$:
$(\mathsf{pk}, (\mathsf{ek}_i, \mathsf{sk}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^{\lambda}, F_n)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{OKeygen}(\cdot,\cdot),\mathsf{OCorrupt}(\cdot)}(\mathsf{pk})$
Output: α

The oracle OEnc, on input $(i, (x_i^0, x_i^1), \ell)$, returns $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, x_i^{\beta}, \ell)$. For any input slot $i \in [n]$, and label ℓ, we denote by $\mathcal{Q}_{i,\ell}$ the set of queries to OEnc for slot i and label ℓ, and $Q_{i,\ell}$ the size of $\mathcal{Q}_{i,\ell}$. The oracle $\mathsf{OKeygen}(i, k)$, on input $i \in [n]$, and $k \in \mathcal{K}_n$, returns $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}_i, k)$. The oracle $\mathsf{OCorrupt}$, on input $i \in [n]$, returns $(\mathsf{ek}_i, \mathsf{sk}_i)$. We denote by $\mathcal{CS} \subseteq [n]$ the set of corrupted slots. The queries of adversary $\mathcal{A}$ must satisfy the following condition.

Condition 1:

- For all $i \in \mathcal{CS}$, all labels ℓ, all $(x_i^0, x_i^1) \in \mathcal{Q}_{i,\ell}$, we have $x_i^0 = x_i^1$.
- if $\mathcal{A}$ queries $\mathsf{OKeygen}(\cdot,\cdot)$ on the same key k for all slots $i \in [n]$, then it must be that:
$$F_n(k, x_1^0, \ldots, x_n^0) = F_n(k, x_1^1, \ldots, x_n^1)$$
for all labels ℓ and all vectors $(x_i^b)_{i\in[n],b\in\{0,1\}}$ such that for all $i \in [n]$, we have: either $(x_i^0, x_i^1) \in \mathcal{Q}_{i,\ell}$, or ($i \in \mathcal{CS}$ and $x_i^0 = x_i^1$).

If the condition is not satisfied, the experiment outputs 0 instead of α.

Concrete Instances of Functional Encryption for Inner Products

In this section, we recall the public-key single-input functional encryption schemes from [ALS16], which are proven many-AD-IND secure for the inner products.

We recall the additional properties defined in [ACF+18], which will be useful to obtain multi-input FE from single-input FE for inner products, in Chapter 4.

Inner-Product FE from MDDH

Here we present the FE for bounded norm inner products from [ALS16, Section 3], generalized to the $\mathcal{D}_k(p)$-MDDH setting, as in [AGRW17, Figure 15]. It handles the following functionality $F_{\mathsf{IP}}^{m,X,Y} : \mathcal{K} \times \mathcal{X} \to \mathcal{Z}$, with $\mathcal{X} := [0, X]^m$, $\mathcal{K} := [0, Y]^m$, $\mathcal{Z} := \mathbb{Z}$, and for all $\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$, we have:

$$F_{\mathsf{IP}}^m(\mathbf{y}, \mathbf{x}) = \langle \mathbf{x}, \mathbf{y} \rangle.$$

This restriction on the norm of $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{K}$ is necessary for the correctness of the scheme. Note that the scheme actually supports vector of arbitrary norms, as long as we only want to decrypt the result in the exponent (see Remark 4).

In [ALS16], it was proven many-AD-IND secure under the DDH assumption. We extend the one-SEL-SIM security proof given in [AGRW17] to the multi-instance setting. Note that in the public-key setting, one-SEL-IND security (which is implied by one-SEL-SIM security) implies many-SEL-IND security. Finally, we also extend the many-AD-IND security proof from [AGRW17] to the multi-instance setting. We also show that is satisfies Property 1 (two-step decryption) and Property 2 (linear encryption).

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathsf{gpk} := (\mathcal{G}, [\mathbf{A}])$
Return gpk

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, $\mathsf{ek} := [\mathbf{WA}]$, $\mathsf{msk} := \mathbf{W}$
Return $(\mathsf{ek}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x})}$:
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
Return $\begin{bmatrix} -\mathbf{Ar} \\ \mathbf{x} + \mathbf{WAr} \end{bmatrix} \in \mathbb{G}^{k+m+1}$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \mathbf{y})}$:
Return $\begin{pmatrix} \mathbf{W}^\top \mathbf{y} \\ \mathbf{y} \end{pmatrix} \in \mathbb{Z}_p^{k+m+1}$

$\underline{\mathsf{Dec}(\mathsf{pk}, [\mathbf{c}], \mathbf{d})}$:
$C := [\mathbf{c}^\top \mathbf{d}]$
Return $\log(C)$

Figure 2.2: $\mathcal{FE}$, a functional encryption scheme for the functionality $F_{\mathsf{IP}}^{m,X,Y}$, whose one-SEL-SIM security is based on the $\mathcal{D}_k(p)$-MDDH assumption.

Correctness. We have $C = [\mathbf{x}^\top \mathbf{y}] \in \mathbb{G}$. Since $\mathbf{x} \in [0, X]^m$ and $\mathbf{y} \in [0, Y]^m$, we have $\langle \mathbf{x}, \mathbf{y} \rangle < m \cdot X \cdot Y$. Thus, we can efficiently recover the discrete log $\langle \mathbf{x}, \mathbf{y} \rangle$ as long as m, X, Y are polynomials in the security parameter.

Remark 4: Correctness for vectors with large norm

Note that the the functional encryption scheme $\mathcal{FE}$ presented in Figure 5.7 supports vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^m$ of arbitrary norm, where the decryption efficiently recovers $[\langle \mathbf{x}, \mathbf{y} \rangle] \in \mathbb{G}$. This feature will be used in Chapter 4 to build multi-input FE from single-input FE for inner products.

Theorem 3: Multi-instance, one-SEL-SIM security

If the $\mathcal{D}_k(p)$-MDDH assumption holds in $\mathbb{G}$, then the single-input FE in Figure 5.7 is one-SEL-SIM secure, for n instances.

Games: G_0, $\boxed{\mathsf{G}_1}$, $\boxed{\mathsf{G}_2}$:

$\{\mathbf{x}_i\}_{i \in I \subseteq [n]} \leftarrow \mathcal{A}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})$

$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\boxed{\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \{\mathbf{0}\} \text{ s.t. } \mathbf{A}^\top \mathbf{a}^\perp = \mathbf{0}}$, $\mathsf{gpk} := (\mathcal{G}, [\mathbf{A}])$.

For all $i \in [n]$: $\mathbf{W}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m \times (k+1)}$, $\mathsf{ek}_i := [\mathbf{W}_i \mathbf{A}], \mathsf{ct}_i := \mathsf{OEnc}(\mathbf{x}_i)$

$\alpha \leftarrow \mathcal{A}^{\mathsf{OKeygen}(\cdot,\cdot)}(\mathsf{gpk}, \{\mathsf{ek}_i\}_{i \in [n]}, \{\mathsf{ct}_i\}_{i \in I})$

Return α.

$\underline{\mathsf{OEnc}(\mathbf{x}_i)}$:

$\mathbf{r}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathbf{c}_i := \mathbf{A}\mathbf{r}_i$, $\boxed{\mathbf{c}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \text{ s.t. } \mathbf{c}_i^\top \mathbf{a}^\perp = 1}$, $\mathbf{c}'_i := \mathbf{x}_i + \mathbf{W}_i \mathbf{c}_i$, $\boxed{\mathbf{c}'_i := \mathbf{W}_i \mathbf{c}_i}$, return $\begin{bmatrix} -\mathbf{c}_i \\ \mathbf{c}'_i \end{bmatrix}$

$\underline{\mathsf{OKeygen}(i, \mathbf{y})}$:

$\mathsf{dk}_\mathbf{y} := \begin{pmatrix} \mathbf{W}_i^\top \mathbf{y} \\ \mathbf{y} \end{pmatrix}$. $\boxed{\text{If } i \in I,\ \mathsf{dk}_\mathbf{y} := \begin{pmatrix} \mathbf{W}_i^\top \mathbf{y} - \langle \mathbf{x}_i, \mathbf{y} \rangle \cdot \mathbf{a}^\perp \\ \mathbf{y} \end{pmatrix}}$.

Return $\mathsf{dk}_\mathbf{y}$.

Figure 2.3: Games for the proof of Theorem 3. In each procedure, the components inside a solid (dotted) frame are only present in the games marked by a solid (dotted) frame. Here, $I \subseteq [n]$ denotes the set of instances for which a challenge ciphertext is queried.

Proof of Theorem 3. Let $\mathcal{A}$ be a PPT adversary, and $\lambda \in \mathbb{N}$ be the security parameter. We proceed with a series of hybrid games, described in Figure 2.3. For any game G, we denote by $\mathsf{Adv}_\mathsf{G}(\mathcal{A})$ the advantage of $\mathcal{A}$ in game G, that is, the probability that the game G outputs 1 when interacting with $\mathcal{A}$.

Game G_0: is the experiment $\mathbf{REAL}^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{A})$.

Game G_1: is as game G_0, except we replace the vector $[\mathbf{c}_i] := [\mathbf{A}\mathbf{r}_i]$ computed by $\mathsf{OEnc}(\mathbf{x}_i)$ with $[\mathbf{c}_i] \leftarrow_{\mathrm{R}} \mathbb{G}^{k+1}$ such that $\mathbf{c}_i^\top \mathbf{a}^\perp = 1$, where $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \{\mathbf{0}\}$ such that $\mathbf{A}^\top \mathbf{a}^\perp = \mathbf{0}$, using the $\mathcal{D}_k(p)$-MDDH assumption. We do so for all instances $i \in I$ simultaneously (recall we denote by $I \subseteq [n]$ the set of instances for which a challenge ciphertext is queried). Namely, we prove in Lemma 7 that there exists a PPT adversary $\mathcal{B}$ such that

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + \frac{1}{p}.$$

$\underline{\widetilde{\mathsf{GSetup}}(1^\lambda, F_{\mathsf{IP}}^m)}$:
$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \{\mathbf{0}\}$ s.t. $\mathbf{A}^\top \mathbf{a}^\perp = \mathbf{0}$, $\widetilde{\mathsf{gpk}} := (\mathcal{G}, [\mathbf{A}])$, $\mathsf{td} := \mathbf{a}^\perp$. Return $(\widetilde{\mathsf{gpk}}, \mathsf{td})$.

$\underline{\widetilde{\mathsf{Setup}}(\widetilde{\mathsf{gpk}}, F_{\mathsf{IP}}^m)}$:
$\widetilde{\mathbf{W}} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m \times (k+1)}$, $\widetilde{\mathsf{ek}} := [\widetilde{\mathbf{W}}\mathbf{A}]$, $\widetilde{\mathsf{msk}} := \widetilde{\mathbf{W}}$. Return $(\widetilde{\mathsf{ek}}, \widetilde{\mathsf{msk}})$.

$\underline{\widetilde{\mathsf{KeyGen}}\left(\mathsf{td}, \widetilde{\mathsf{msk}}, \mathbf{y}, \langle \mathbf{x}, \mathbf{y} \rangle\right)}$:
Return $\begin{pmatrix} \widetilde{\mathbf{W}}^\top \mathbf{y} - \langle \mathbf{x}, \mathbf{y} \rangle \cdot \mathbf{a}^\perp \\ \mathbf{y} \end{pmatrix}$.

$\underline{\widetilde{\mathsf{Enc}}(\mathsf{td}, \widetilde{\mathsf{ek}}, \widetilde{\mathsf{msk}})}$:
$\mathbf{c} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$ s.t. $\mathbf{c}^\top \mathbf{a}^\perp = 1$. Return $\begin{bmatrix} -\mathbf{c} \\ \widetilde{\mathbf{W}}\mathbf{c} \end{bmatrix}$.

Figure 2.4: Simulator $(\widetilde{\mathsf{GSetup}}, \widetilde{\mathsf{Setup}}, \widetilde{\mathsf{KeyGen}}, \widetilde{\mathsf{Enc}})$ for the one-SEL-SIM security of the FE from Figure 5.7.

Game G_2: is the experiment $\mathbf{IDEAL}^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{A})$, where the simulator $(\widetilde{\mathsf{GSetup}}, \widetilde{\mathsf{Setup}}, \widetilde{\mathsf{KeyGen}}, \widetilde{\mathsf{Enc}})$ is described in 2.4. In Lemma 8, we show that game G_2 and game G_1 are perfectly indistinguishable, using a statistical argument, that crucially relies on the fact that game G_1 and G_2 are selective. Namely, we prove in Lemma 8 that

$$\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}).$$

Putting everything together, we obtain:

$$\mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}SIM}}_{\mathcal{FE},\mathcal{A},n}(\lambda) \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + \frac{1}{p}.$$

□

Lemma 7: Game G_0 to G_1

There exists a PPT adversary $\mathcal{B}$ such that

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + \frac{1}{p}.$$

Proof of Lemma 7. In game G_1, we replace the vectors $[\mathbf{A}\mathbf{r}_i]$ computed by $\mathsf{OEnc}(\mathbf{x}_i)$, with $[\mathbf{c}_i] \leftarrow_{\mathrm{R}} \mathbb{G}^{k+1}$ such that $\mathbf{c}_i^\top \mathbf{a}^\perp = 1$, simultaneously for all instances $i \in [n]$. This replacement is justified by the facts that:

- The following are identically distributed: $\{\mathbf{A}\mathbf{r}_i\}_{i\in[n]}$ and $\{\mathbf{A}\mathbf{r}_i + \mathbf{A}\mathbf{r}\}_{i\in[n]}$, where for all $i \in [n]$, $\mathbf{r}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$.
- By the $\mathcal{D}_k(p)$-MDDH assumption, we can switch $([\mathbf{A}], [\mathbf{A}\mathbf{r}])$ to $([\mathbf{A}], [\mathbf{u}])$, where $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$.
- The uniform distribution over $\mathbb{Z}_p^{k+1}$ and $\mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$ are $\frac{1}{p}$-close, for any $\mathbf{A} \in \mathbb{Z}_p^{(k+1)\times k}$ of rank k. So we can take $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$ instead of uniformly random over $\mathbb{Z}_p^{k+1}$.

Combining these facts, we obtain a PPT adversary $\mathcal{B}$ such that $|\mathsf{Adv}_1(\mathcal{A}) - \mathsf{Adv}_0(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{D}_k\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + \frac{1}{p}$. □

Lemma 8: Game G_1 to G_2

$$\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}).$$

Proof of Lemma 8. We use the fact that the following are identically distributed:

$$\{\mathbf{W}_i\}_{i\in[n]} \text{ and } \{\mathbf{W}_i - \mathbf{x}_i(\mathbf{a}^\perp)^\top\}_{i\in[n]},$$

where for all $i \in [n]$: $\mathbf{W}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, and $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$ such that $\mathbf{A}^\top\mathbf{a}^\perp = \mathbf{0}$ and for all $i \in [n]$, $\mathbf{c}_i^\top\mathbf{a}^\perp = 1$.

The leftmost distribution corresponds to game G_1, whereas the rightmost distribution corresponds to game G_2. We crucially rely on the fact that these games are selective, thus, the matrices $\mathbf{W}_i$ are picked after the adversary $\mathcal{A}$ sends its challenge $\{\mathbf{x}_i\}_{i\in I}$, and therefore, *independently* of it.

Namely:

$$\begin{aligned}(\mathbf{W}_i - \mathbf{x}_i(\mathbf{a}^\perp)^\top)\mathbf{A} &= \mathbf{W}_i\mathbf{A}\\ \mathbf{x}_i + (\mathbf{W}_i - \mathbf{x}_i(\mathbf{a}^\perp)^\top)\mathbf{c}_i &= \mathbf{W}_i\mathbf{c}_i\\ (\mathbf{W}_i - \mathbf{x}_i(\mathbf{a}^\perp)^\top)^\top\mathbf{y} &= \mathbf{W}_i^\top\mathbf{y} - \langle\mathbf{x}_i,\mathbf{y}\rangle\cdot\mathbf{a}^\perp\end{aligned}$$

which coincides precisely with the output of the simulator. This proves $\mathsf{Adv}_2(\mathcal{A}) = \mathsf{Adv}_1(\mathcal{A})$. □

Theorem 4: Multi-instance, many-AD-IND security

If the $\mathcal{D}_k(p)$-MDDH assumption holds in $\mathbb{G}$, then the single-input FE in Figure 5.7 is many-AD-IND secure for n instances.

Games: $\mathsf{G}_{0,\beta}$, $\boxed{\mathsf{G}_{1,\beta}, \boxed{\mathsf{G}^\star_{1,\beta}}}$, for $\beta \in \{0,1\}$:

$\boxed{(\mathbf{x}_0,\mathbf{x}_1) \leftarrow \mathcal{A}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})}$

$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathsf{gpk} := (\mathcal{G}, [\mathbf{A}])$, $\boxed{\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \{\mathbf{0}\} \text{ s. t. } \mathbf{A}^\top\mathbf{a}^\perp = \mathbf{0}}$,

$\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, $\mathsf{ek} := [\mathbf{WA}]$, $\boxed{\mathsf{ct} := \mathsf{OEnc}(\mathbf{x}_0,\mathbf{x}_1)}$.

$\alpha \leftarrow \mathcal{A}^{\mathsf{OKeygen}(\cdot),\ \mathsf{OEnc}(\cdot)}(\mathsf{ek}, \boxed{\mathsf{ct}})$

Return α.

$\underline{\mathsf{OEnc}(\mathbf{x}_0,\mathbf{x}_1)}$:

$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathbf{c} := \mathbf{Ar}$, $\boxed{\mathbf{c} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \text{ s.t. } \mathbf{c}^\top\mathbf{a}^\perp = 1}$, $\mathbf{c}' := \mathbf{x}^\beta + \mathbf{Wc}$, return $\begin{bmatrix}-\mathbf{c}\\ \mathbf{c}'\end{bmatrix}$.

$\underline{\mathsf{OKeygen}(\mathbf{y})}$:

Return $\begin{pmatrix}\mathbf{W}^\top\mathbf{y}\\ \mathbf{y}\end{pmatrix}$

Figure 2.5: Games for the proof of Theorem 4. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame. The encryption oracle OEnc can only be called once by adversary $\mathcal{A}$.

Proof of Theorem 4. First,because $\mathcal{FE}$ described in Figure 5.7 is a public key encryption scheme, it suffices to prove one-AD-IND security: many-AD-IND follows by a standard hybrid argument over all challenge ciphertexts (cf Lemma 6). Second, it suffices to prove security for a single instance, since it implies its many-instance variant, as shown in Lemma 5. We now prove one-AD-IND security for a single instance.

Let $\mathcal{A}$ be a PPT adversary, and $\lambda \in \mathbb{N}$ be the security parameter. We proceed with a series of hybrid games, described below. For any game G, we denote by $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ the advantage of $\mathcal{A}$ in game G, that is, the probability that the game G outputs 1 when interacting with $\mathcal{A}$.

Games $\mathsf{G}_{0,\beta}$, for $\beta \in \{0,1\}$: are such that $\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{A},1}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_{0,0}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{0,1}}(\mathcal{A})|$ (see Definition 21).

Games $\mathsf{G}_{1,\beta}$, for $\beta \in \{0,1\}$: are as games $\mathsf{G}_{0,\beta}$, except we replace the vector $[\mathbf{Ar}]$ computed by $\mathsf{OEnc}(\mathbf{x}_0, \mathbf{x}_1)$ with $[\mathbf{c}] \leftarrow_{\mathrm{R}} \mathbb{G}^{k+1}$, such that $\mathbf{c}^\top \mathbf{a}^\perp = 1$, where $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \{\mathbf{0}\}$ such that $\mathbf{A}^\top \mathbf{a}^\perp = \mathbf{0}$, using the $\mathcal{D}_k(p)$-MDDH assumption. Namely, we prove in Lemma 9 that there exists a PPT adversary $\mathcal{B}_\beta$ such that

$$|\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_\beta}(\lambda) + \frac{1}{p}.$$

At this point, we show that $\mathsf{Adv}_{\mathsf{G}_{1,0}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{1,1}}(\mathcal{A})$ in three steps. First, we consider the selective variant of game $\mathsf{G}_{1,\beta}$, called $\mathsf{G}^\star_{1,\beta}$, where the adversary must commit to its challenge $\{\mathbf{x}_b\}_{b\in\{0,1\}}$ beforehand. By a guessing argument, we show in Lemma 10 that there exists PPT adversary $\mathcal{A}^\star$ such that

$$\mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) = (X+1)^{2m} \cdot \mathsf{Adv}_{\mathsf{G}^\star_{1,\beta}}(\mathcal{A}^\star).$$

Then we prove in Lemma 11 that the game $\mathsf{G}^\star_{1,0}$ is identical to game $\mathsf{G}^\star_{1,1}$ using a statistical argument, which is only true in the selective setting. Namely, for any adversary $\mathcal{A}'$:

$$\mathsf{Adv}_{\mathsf{G}^\star_{1,0}}(\mathcal{A}') = \mathsf{Adv}_{\mathsf{G}^\star_{1,1}}(\mathcal{A}').$$

Putting everything together, we obtain:

$$\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{A},1}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + \frac{2}{p}.$$

□

Lemma 9: Game $\mathsf{G}_{0,\beta}$ to $\mathsf{G}_{1,\beta}$

There exists a PPT adversary $\mathcal{B}_\beta$ such that

$$|\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_\beta}(\lambda) + \frac{1}{p}.$$

Proof of Lemma 9. This is proof is similar to the proof of Lemma 7, for the one-SEL-SIM security of $\mathcal{FE}$. We replace the vectors $[\mathbf{Ar}]$ computed by $\mathsf{OEnc}(\mathbf{x}_0, \mathbf{x}_1)$ with $[\mathbf{c}] \leftarrow_{\mathrm{R}} \mathbb{G}^{k+1}$ such that $\mathbf{c}^\top \mathbf{a}^\perp = 1$. This replacement is justified by the facts that:

- By the $\mathcal{D}_k(p)$-MDDH assumption, we can switch $([\mathbf{A}], [\mathbf{Ar}])$ to $([\mathbf{A}], [\mathbf{u}])$, where $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$.

- The uniform distribution over $\mathbb{Z}_p^{k+1}$ and $\mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$ are $\frac{1}{p}$-close, for any $\mathbf{A} \in \mathbb{Z}_p^{(k+1)\times k}$ of rank k. Thus, we can chose $\mathbf{u} \leftarrow \mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$ instead of uniformly random over $\mathbb{Z}_p^{k+1}$.

Combining these facts, we obtain a PPT adversary $\mathcal{B}_\beta$ such that $|\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_\beta}(\lambda) + \frac{1}{p}$. □

Lemma 10: Game $\mathsf{G}_{1,\beta}$ to $\mathsf{G}^\star_{1,\beta}$

There exists a PPT adversary $\mathcal{A}^\star$ such:

$$\mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) = (X+1)^{-2m} \cdot \mathsf{Adv}_{\mathsf{G}^\star_{1,\beta}}(\mathcal{A}^\star).$$

Proof of Lemma 10. First, $\mathcal{A}^\star$ guesses the challenge by picking random: $\{\mathbf{x}^\star_b\}_{b\in\{0,1\}} \leftarrow_{\mathrm{R}} [0, X]^{2m}$, and sends its to the game $\mathsf{G}^\star_{1,\beta}$, which is a selective variant of game $\mathsf{G}_{1,\beta}$. These games are described in Figure 2.5. Whenever $\mathcal{A}$ queries $\mathsf{OKeygen}$, $\mathcal{A}^\star$ forwards the query to its own oracle, and gives back the answer to $\mathcal{A}$. When $\mathcal{A}$ calls $\mathsf{OEnc}(\mathbf{x}_0, \mathbf{x}_1)$, $\mathcal{A}^\star$ verifies its guess was correct, that is $(\mathbf{x}_0, \mathbf{x}_1) = (\mathbf{x}^\star_0, \mathbf{x}^\star_1)$. If the guess is incorrect, $\mathcal{A}^\star$ ends the simulation, and sends $\alpha := 0$ to the game $\mathsf{G}^\star_{1,\beta}$. Otherwise, it keeps answering $\mathcal{A}$'s queries to $\mathsf{OKeygen}$ as explained, and forwards $\mathcal{A}$'s output α to the game $\mathsf{G}^\star_{1,\beta}$.

When $\mathcal{A}^\star$ guesses correctly, it simulates $\mathcal{A}$'s view perfectly. When it fails to guess, it outputs $\alpha := 0$. Thus, the probability that $\mathcal{A}^\star$ outputs 1 in $\mathsf{G}^\star_{1,\beta}$ is exactly $(X+1)^{-2m} \cdot \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A})$. □

Lemma 11: Game $\mathsf{G}^\star_{1,0}$ to $\mathsf{G}^\star_{1,1}$

For all adversaries $\mathcal{A}'$, we have:

$$\mathsf{Adv}_{\mathsf{G}^\star_{1,0}}(\mathcal{A}') = \mathsf{Adv}_{\mathsf{G}^\star_{1,1}}(\mathcal{A}').$$

Proof of Lemma 11. We use the fact that the following distributions are identical:

$$\mathbf{W} \quad \text{and} \quad \mathbf{W} + (\mathbf{x}_1 - \mathbf{x}_0)(\mathbf{a}^\perp)^\top,$$

where $\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, and $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$ such that $\mathbf{A}^\top\mathbf{a}^\perp = \mathbf{0}$.

The leftmost distribution corresponds to game $\mathsf{G}^\star_{1,0}$, while the rightmost distribution corresponds to $\mathsf{G}^\star_{1,1}$, since we have:

$$\begin{aligned}
(\mathbf{W} + (\mathbf{x}_1 - \mathbf{x}_0)(\mathbf{a}^\perp)^\top)\mathbf{A} &= \mathbf{W}\mathbf{A} \\
\mathbf{x}_0 + (\mathbf{W} + (\mathbf{x}_1 - \mathbf{x}_0)(\mathbf{a}^\perp)^\top)\mathbf{c} &= \mathbf{x}_1 + \mathbf{W}\mathbf{c} \\
(\mathbf{W} + (\mathbf{x}_1 - \mathbf{x}_0)(\mathbf{a}^\perp)^\top)^\top\mathbf{y} &= \mathbf{W}^\top\mathbf{y} + (\langle\mathbf{x}_1, \mathbf{y}\rangle - \langle\mathbf{x}_0, \mathbf{y}\rangle)\mathbf{a}^\perp \\
&= \mathbf{W}^\top\mathbf{y}
\end{aligned}$$

The first equality uses the fact that $\mathbf{A}^\top\mathbf{a}^\perp = \mathbf{0}$, the second equality uses the fact that $\mathbf{c}^\top\mathbf{a}^\perp = 1$, and the third equality uses the fact that $\langle\mathbf{x}_0, \mathbf{y}\rangle = \langle\mathbf{x}_1, \mathbf{y}\rangle$ for any $\mathbf{y}$ queried to $\mathsf{OKeygen}$.

Note that we are relying on the fact that in these games, $\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$ is picked after the adversary $\mathcal{A}$ sends its selective challenge $\{\mathbf{x}_b\}_{b\in\{0,1\}}$, and therefore, independently of it. □

Inner-Product FE from LWE

Here we present the many-AD-IND secure Inner-Product FE from [ALS16, Section 4.1].

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})}$:
Let integers $M, q \geq 2$, real $\alpha \in (0,1)$, and distribution $\mathcal{D}$ over $\mathbb{Z}^{m \times M}$. Set $K := m \cdot X \cdot Y$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{M \times \lambda}$, $\mathsf{gpk} := (K, \mathbf{A})$.
Return gpk

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathbf{Z} \leftarrow_{\mathrm{R}} \mathcal{D}$, $\mathbf{U} := \mathbf{ZA} \in \mathbb{Z}_q^{m \times \lambda}$, $\mathsf{ek} := \mathbf{U}$, $\mathsf{msk} := \mathbf{Z}$.
Return $(\mathsf{ek}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x} \in \mathbb{Z}^m)}$:
$\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^\lambda$, $\mathbf{e}_0 \leftarrow_{\mathrm{R}} D_{\mathbb{Z},\alpha q}^M$, $\mathbf{e}_1 \leftarrow_{\mathrm{R}} D_{\mathbb{Z},\alpha q}^m$
$\mathbf{c}_0 := \mathbf{As} + \mathbf{e}_0 \in \mathbb{Z}_q^M$
$\mathbf{c}_1 := \mathbf{Us} + \mathbf{e}_1 + \mathbf{x} \cdot \lfloor \frac{q}{K} \rfloor \in \mathbb{Z}_q^m$
Return $\mathsf{ct}_{\mathbf{x}} := (\mathbf{c}_0, \mathbf{c}_1)$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \mathbf{y} \in \mathbb{Z}^m)}$:
Return $\mathsf{dk}_{\mathbf{y}} := \begin{pmatrix} \mathbf{Z}^\top \mathbf{y} \\ \mathbf{y} \end{pmatrix} \in \mathbb{Z}^{M+m}$

$\underline{\mathsf{Dec}(\mathsf{gpk}, \mathsf{dk}_{\mathbf{y}}, \mathsf{ct}_{\mathbf{x}})}$:
$\mu' := \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \end{pmatrix}^\top \mathsf{dk}_{\mathbf{y}} \bmod q$.
Return $\mu \in \{-K+1, \ldots, K-1\}$ that minimizes $\left| \lfloor \frac{q}{K} \rfloor \mu - \mu' \right|$.

Figure 2.6: Functional encryption scheme for the class $F_{\mathsf{IP}}^{m,X,Y}$, based on the LWE assumption.

Choice of parameters. Following the analysis given in [ALS16], we choose:

- $\sigma_1 := \Theta\left(\sqrt{\lambda \log(M) \max(\sqrt{M}, K)}\right)$
- $\sigma_2 := \Theta\left(\lambda^{7/2} M^{1/2} \max(M, K^2) \log^{5/2}(M)\right)$
- $\mathcal{D} := D_{\mathbb{Z},\sigma_1}^{m \times M/2} \times D_{\mathbb{Z}^{M/2},\sigma_2,\mathbf{u}_1} \times \cdots \times D_{\mathbb{Z}^{M/2},\sigma_2,\mathbf{u}_m}$, where for all $i \in [m]$, $\mathbf{u}_i$ denotes the i'th canonical vector.
- Let $B_{\mathcal{D}}$ be such that with probability at least $1 - \lambda^{\omega(1)}$, each row of a sample from $\mathcal{D}$ has norm at most $B_{\mathcal{D}}$. For correctness, we must have: $\alpha^{-1} \geq K^2 B_{\mathcal{D}} \omega(\sqrt{\log \lambda})$, $q \geq \alpha^{-1} \omega(\sqrt{\log(\lambda)})$.
- $M \geq 4\lambda \log q$, $m \leq \lambda^{O(1)}$, $q > mK^2$

Theorem 5: many-AD-IND security [ALS16]

The FE from Figure 5.8 is correct and many-AD-IND secure under the $\mathsf{mheLWE}_{q,\alpha,M,m,\mathcal{D}}$ assumption (see Definition 18).

Inner-Product FE from DCR

Here we present the many-AD-IND secure Inner-Product FE from [ALS16, Section 5.1].

Theorem 6: many-AD-IND security [ALS16]

The FE from Figure 5.9 is correct and many-AD-IND secure under the DCR assumption

(see Definition 16).

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})}$:
Choose primes $p = 2p' + 1$, $q = q' + 1$ with prime $p', q' > 2^{l(\lambda)}$ for an $l(\lambda) = \mathsf{poly}(\lambda)$ such that factoring is λ-hard, and set $N := pq$ ensuring that $m \cdot X \cdot Y < N$. Sample $g' \leftarrow_{\mathrm{R}} \mathbb{Z}_{N^2}^*$, $g := g'^{2N} \bmod N^2$.
Return $\mathsf{gpk} := (N, g)$

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathbf{s} \leftarrow_{\mathrm{R}} D_{\mathbb{Z}^m,\sigma}$, for standard deviation $\sigma > \sqrt{\lambda} \cdot N^{5/2}$, and for all $j \in [m]$, $h_j := g^{s_j} \bmod N^2$.
$\mathsf{ek} := \{h_j\}_{j\in[m]}$, $\mathsf{msk} := \{s_j\}_{j\in[m]}$
Return $(\mathsf{ek}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x} \in \mathbb{Z}^m)}$:
$r \leftarrow_{\mathrm{R}} \{0, \ldots, \lfloor N/4 \rfloor\}$, $C_0 := g^r \in \mathbb{Z}_{N^2}$, for all $j \in [m]$, $C_j := (1 + x_j N) \cdot h_j^r \in \mathbb{Z}_{N^2}$
Return $\mathsf{ct}_{\mathbf{x}} := (C_0, \ldots, C_m) \in \mathbb{Z}_{N^2}^{m+1}$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \mathbf{y} \in \mathbb{Z}^m)}$:
$d := \sum_{j\in[m]} y_j s_j \in \mathbb{Z}$.
Return $\mathsf{sk}_{\mathbf{y}} := (d, \mathbf{y})$

$\underline{\mathsf{Dec}\big(\mathsf{gpk}, \mathsf{sk}_{\mathbf{y}} := (d, \mathbf{y}), \mathsf{ct}_{\mathbf{x}}\big)}$:
$C := \left(\prod_{j\in[m]} C_j^{y_j}\right) \cdot C_0^{-d} \bmod N^2$.
Return $\log_{(1+N)}(C) := \frac{C-1 \bmod N^2}{N}$.

Figure 2.7: Functional encryption scheme for the class $F_{\mathsf{IP}}^{m,X,Y}$, based on the DCR assumption.

Chapter 3

Tightly CCA-Secure Encryption without Pairings

We present the construction from [GHKW16], which was the first CCA-secure public-key encryption with a tight security reduction to DDH, without relying on the use of pairings. We refer to Figure 1.1 for a comparison with related works.

Overview of our construction. In this overview, we will consider a weaker notion of security, namely tag-based KEM security against plaintext check attacks (PCA) [OP01]. In the PCA security experiment, the adversary gets no decryption oracle (as with CCA security), but a PCA oracle that takes as input a tag and a ciphertext/plaintext pair and checks whether the ciphertext decrypts to the plaintext. Furthermore, we restrict the adversary to only query the PCA oracle on tags different from those used in the challenge ciphertexts. PCA security is strictly weaker than the CCA security we actually strive for, but allows us to present our solution in a clean and simple way. (We show how to obtain full CCA security separately.)

The starting point of our construction is the Cramer-Shoup KEM. The public key is given by $\mathsf{pk} := ([\mathbf{M}], [\mathbf{M}^\top \mathbf{k}_0], [\mathbf{M}^\top \mathbf{k}_1])$ for $\mathbf{M} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{(k+1)\times k}$. On input pk and a tag τ, the encryption algorithm outputs the ciphertext/plaintext pair

$$([\mathbf{y}], [z]) = ([\mathbf{M}\mathbf{r}], [\mathbf{r}^\top \mathbf{M}^\top \mathbf{k}_\tau]), \tag{3.1}$$

where $\mathbf{k}_\tau = \mathbf{k}_0 + \tau \mathbf{k}_1$ and $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$. Decryption relies on the fact that $\mathbf{y}^\top \mathbf{k}_\tau = \mathbf{r}^\top \mathbf{M}^\top \mathbf{k}_\tau$. The KEM is PCA-secure under k-Lin, with a security loss that depends on the number of ciphertexts Q (via a hybrid argument) but independent of the number of PCA queries [CS03, ABP15].

Following the "randomized Naor-Reingold" paradigm introduced by Chen and Wee on tightly secure IBE [CW13], our starting point is (3.1), where we replace $\mathbf{k}_\tau = \mathbf{k}_0 + \tau \mathbf{k}_1$ with

$$\mathbf{k}_\tau = \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$$

and $\mathsf{pk} := ([\mathbf{M}], [\mathbf{M}^\top \mathbf{k}_{j,b}]_{j=1,\ldots,\lambda, b=0,1})$, where $(\tau_1, \ldots, \tau_\lambda)$ denotes the binary representation of the tag $\tau \in \{0,1\}^\lambda$.

Following [CW13], we want to analyze this construction by a sequence of games in which we first replace $[\mathbf{y}]$ in the challenge ciphertexts by uniformly random group elements via random self-reducibility of MDDH (k-Lin), and then incrementally replace $\mathbf{k}_\tau$ in both the challenge ciphertexts and in the PCA oracle by $\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}(\tau)$, where RF is a truly random function and $\mathbf{M}^\perp$ is a random element from the kernel of $\mathbf{M}$, i.e., $\mathbf{M}^\top \mathbf{M}^\perp = 0$. Concretely, in Game i, we will replace $\mathbf{k}_\tau$ with $\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_i(\tau)$ where RF_i is a random function on $\{0,1\}^i$ applied to the i-bit prefix of τ. We proceed to outline the two main ideas needed to carry out this transition. Looking ahead, note that once we reach Game λ, we would have replaced $\mathbf{k}_\tau$ with

$\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}(\tau)$, upon which security follows from a straight-forward information-theoretic argument (and the fact that ciphertexts and decryption queries carry pairwise different τ).

First idea. First, we show how to transition from Game i to Game $i+1$, under the restriction that the adversary is only allowed to query the encryption oracle on tags whose $i+1$-st bit is 0; we show how to remove this unreasonable restriction later. Here, we rely on an *information-theoretic* argument similar to that of Cramer and Shoup to increase the entropy from RF_i to RF_{i+1}. This is in contrast to prior works which rely on a computational argument; note that the latter requires encoding secret keys as group elements and thus a pairing to carry out decryption.

More precisely, we pick a random function RF'_i on $\{0,1\}^i$, and implicitly define RF_{i+1} as follows:

$$\mathsf{RF}_{i+1}(\tau) = \begin{cases} \mathsf{RF}_i(\tau) & \text{if } \tau_{i+1} = 0 \\ \mathsf{RF}'_i(\tau) & \text{if } \tau_{i+1} = 1 \end{cases}$$

Observe all of the challenge ciphertexts leak no information about RF'_i or $\mathbf{k}_{i+1,1}$ since they all correspond to tags whose $i+1$-st bit is 0. To handle a PCA query $(\tau, [\mathbf{y}], [z])$, we proceed via a case analysis:

- if $\tau_{i+1} = 0$, then $\mathbf{k}_\tau + \mathsf{RF}_{i+1}(\tau) = \mathbf{k}_\tau + \mathsf{RF}_i(\tau)$ and the PCA oracle returns the same value in both Games i and $i+1$.
- if $\tau_{i+1} = 1$ and $\mathbf{y}$ lies in the span of $\mathbf{M}$, we have

  $$\mathbf{y}^\top \mathbf{M}^\perp = 0 \Longrightarrow \mathbf{y}^\top(\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_i(\tau)) = \mathbf{y}^\top(\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_{i+1}(\tau)),$$

  and again the PCA oracle returns the same value in both Games i and $i+1$.
- if $\tau_{i+1} = 1$ and $\mathbf{y}$ lies outside the span of $\mathbf{M}$, then $\mathbf{y}^\top \mathbf{k}_{i+1,1}$ is uniformly random given $\mathbf{M}, \mathbf{M}^\top \mathbf{k}_{i+1,1}$. (Here, we crucially use that the adversary does not query encryptions with $\tau_{i+1} = 1$, which ensures that the challenge ciphertexts do not leak additional information about $\mathbf{k}_{i+1,1}$.) This means that $\mathbf{y}^\top \mathbf{k}_\tau$ is uniformly random from the adversary's viewpoint, and therefore the PCA oracle will reject with high probability in both Games i and $i+1$. (At this point, we crucially rely on the fact that the PCA oracle only outputs a *single* check bit and not all of $\mathbf{k}_\tau + \mathsf{RF}(\tau)$.)

Via a hybrid argument, we may deduce that the distinguishing advantage between Games i and $i+1$ is at most Q/q where Q is the number of PCA queries.

Second idea. Next, we remove the restriction on the encryption queries using an idea of Hofheinz, Koch and Striecks [HKS15] for tightly-secure IBE in the multi-ciphertext setting, and its instantiation in prime-order groups [GCD+16]. The idea is to create two "independent copies" of $(\mathbf{M}^\perp, \mathsf{RF}_i)$; we use one to handle encryption queries on tags whose $i+1$-st bit is 0, and the other to handle those whose $i+1$-st bit is 1. We call these two copies $(\mathbf{M}_0^*, \mathsf{RF}_i^{(0)})$ and $(\mathbf{M}_1^*, \mathsf{RF}_i^{(1)})$, where $\mathbf{M}^\top \mathbf{M}_0^* = \mathbf{M}^\top \mathbf{M}_1^* = \mathbf{0}$.

Concretely, we replace $\mathbf{M} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{(k+1)\times k}$ with $\mathbf{M} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{3k\times k}$. We decompose $\mathbb{Z}_q^{3k}$ into the span of the respective matrices $\mathbf{M}, \mathbf{M}_0, \mathbf{M}_1$, and we will also decompose the span of $\mathbf{M}^\perp \in \mathbb{Z}_q^{3k\times 2k}$ into that of $\mathbf{M}_0^*, \mathbf{M}_1^*$. Similarly, we decompose $\mathbf{M}^\perp \mathsf{RF}_i(\tau)$ into $\mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau)$. We then refine the prior transition from Games i to $i+1$ as follows:

- Game $i.0$ (= Game i): pick $\mathbf{y} \leftarrow \mathbb{Z}_q^{3k}$ for ciphertexts, and replace $\mathbf{k}_\tau$ with $\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau)$;

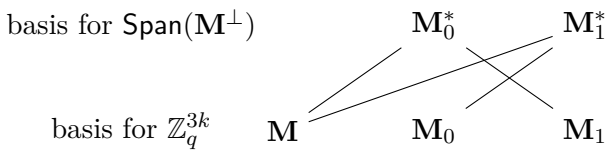


Figure 3.1: Solid lines mean orthogonal, that is: $\mathbf{M}^\top\mathbf{M}_0^* = \mathbf{M}_1^\top\mathbf{M}_0^* = \mathbf{0} = \mathbf{M}^\top\mathbf{M}_1^* = \mathbf{M}_0^\top\mathbf{M}_1^*$.

- Game $i.1$: replace $\mathbf{y} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{3k}$ with $\mathbf{y} \leftarrow_{\mathrm{R}} \mathsf{Span}(\mathbf{M}, \mathbf{M}_{\tau_{i+1}})$;
- Game $i.2$: replace $\mathsf{RF}_i^{(0)}(\tau)$ with $\mathsf{RF}_{i+1}^{(0)}(\tau)$;
- Game $i.3$: replace $\mathsf{RF}_i^{(1)}(\tau)$ with $\mathsf{RF}_{i+1}^{(1)}(\tau)$;
- Game $i.4$ (= Game $i+1$): replace $\mathbf{y} \leftarrow_{\mathrm{R}} \mathsf{Span}(\mathbf{M}, \mathbf{M}_{\tau_{i+1}})$ with $\mathbf{y} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{3k}$.

For the transition from Game $i.0$ to Game $i.1$, we rely on the fact that the uniform distributions over $\mathbb{Z}_q^{3k}$ and $\mathsf{Span}(\mathbf{M}, \mathbf{M}_{\tau_{i+1}})$ encoded in the group are computationally indistinguishable, even given a random basis for $\mathsf{Span}(\mathbf{M}^\perp)$ (in the clear). This extends to the setting with multiple samples, with a tight reduction to the $\mathcal{D}_k(p)$-MDDH Assumption independent of the number of samples.

For the transition from Game $i.1$ to $i.2$, we rely on an information-theoretic argument like the one we just outlined, replacing $\mathsf{Span}(\mathbf{M})$ with $\mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$ and $\mathbf{M}^\perp$ with $\mathbf{M}_0^*$ in the case analysis. In particular, we will exploit the fact that if $\mathbf{y}$ lies outside $\mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$, then $\mathbf{y}^\top\mathbf{k}_{i+1,1}$ is uniformly random even given $\mathbf{M}, \mathbf{M}\mathbf{k}_{i+1,1}, \mathbf{M}_1, \mathbf{M}_1\mathbf{k}_{i+1,1}$. The transition from Game $i.2$ to $i.3$ is completely analogous.

From PCA to CCA. Using standard techniques from [CS03, KD04, Kil06, BCHK07, AGK08], we could transform our basic tag-based PCA-secure scheme into a "full-fledged" CCA-secure encryption scheme by adding another hash proof system (or an authenticated symmetric encryption scheme) and a one-time signature scheme. However, this would incur an additional overhead of several group elements in the ciphertext. Instead, we show how to directly modify our tag-based PCA-secure scheme to obtain a more efficient CCA-secure scheme with the minimal additional overhead of a single symmetric-key authenticated encryption. In particular, the overall ciphertext overhead in our tightly CCA-secure encryption scheme is merely *one* group element more than that for the best known non-tight schemes [KD04, HK07].

To encrypt a message M in the CCA-secure encryption scheme, we will (i) pick a random $\mathbf{y}$ as in the tag-based PCA scheme, (ii) derive a tag τ from $\mathbf{y}$, (iii) encrypt M using a one-time authenticated encryption under the KEM key $[\mathbf{y}^\top\mathbf{k}_\tau]$. The naive approach is to derive the tag τ by hashing $[\mathbf{y}] \in \mathbb{G}^{3k}$, as in [KD04]. However, this creates a circularity in Game $i.1$ where the distribution of $[\mathbf{y}]$ depends on the tag. Instead, we will derive the tag τ by hashing $[\overline{\mathbf{y}}] \in \mathbb{G}^k$, where $\overline{\mathbf{y}} \in \mathbb{Z}_q^k$ are the top k entries of $\mathbf{y} \in \mathbb{Z}_q^{3k}$. We then modify $\mathbf{M}_0, \mathbf{M}_1$ so that the top k rows of both matrices are zero, which avoids the circularity issue. In the proof of security, we will also rely on the fact that for any $\mathbf{y}_0, \mathbf{y}_1 \in \mathbb{Z}_q^{3k}$, if $\overline{\mathbf{y}}_0 = \overline{\mathbf{y}}_1$ and $\mathbf{y}_0 \in \mathsf{Span}(\mathbf{M})$, then either $\mathbf{y}_0 = \mathbf{y}_1$ or $\mathbf{y}_1 \notin \mathsf{Span}(\mathbf{M})$. This allows us to deduce that if the adversary queries the CCA oracle on a ciphertext which shares the same tag as some challenge ciphertext, then the CCA oracle will reject with overwhelming probability.

Alternative view-point. Our construction can also be viewed as applying the IBE-to-PKE transform from [BCHK07] to the scheme from [HKS15], and then writing the exponents of the secret keys in the clear, thereby avoiding the pairing. This means that we can no longer apply a computational assumption and the randomized Naor-Reingold argument to the secret key space. Indeed, we replace this with an information-theoretic Cramer-Shoup-like argument as outlined above.

Prior approaches. Several approaches to construct tightly CCA-secure PKE schemes exist: first, the schemes of [HJ12, ACD+12, ADK+13, LPJY14, LJYP14, LPJY15] construct a tightly secure NIZK scheme from a tightly secure signature scheme, and then use the tightly secure NIZK in a CCA-secure PKE scheme following the Naor-Yung double encryption paradigm [NY90, DDN00]. Since these approaches build on the public verifiability of the used NIZK scheme (in order to faithfully simulate a decryption oracle), their reliance on a pairing seems inherent.

Next, the works of [CW13, BKP14, HKS15, AHY15b, GCD+16] used a (Naor-Reingold-based) MAC instead of a signature scheme to design tightly secure IBE schemes. Those IBE schemes can then be converted (using the BCHK transformation [BCHK07]) into tightly CCA-secure PKE schemes. However, the derived PKE schemes still rely on pairings, since the original IBE schemes do (and the BCHK does not remove the reliance on pairings).

In contrast, our approach directly fuses a Naor-Reingold-like randomization argument with the encryption process. We are able to do so since we substitute a computational randomization argument (as used in the latter line of works) with an information-theoretic one, as described above. Hence, we can apply that argument to *exponents* rather than group elements. This enables us to trade pairing operations for exponentiations in our scheme.

Road-map. The rest of this chapter is organized as follows. First, we present our key-encapsulation mechanism (KEM) that is only PCA-secure when there is multiple challenge ciphertext, with a tight security reduction from DDH. Its security proof already captures most technical novelties. Then, we show how to upgrade this encryption scheme to obtain tightly, CCA-secure encryption, using an additional layer of symmetric authenticated encryption, à la [KD04, HK07].

Multi-ciphertext PCA-secure KEM

In this section we describe a tag-based Key Encapsulation Mechanism $\mathcal{KEM}$ that is IND-PCA-secure (see Definition 6).

For simplicity, we use the matrix distribution $\mathcal{U}_{3k,k}(p)$ in our scheme in Figure 3.2, and prove it secure under the $\mathcal{U}_k(p)$-MDDH assumption ($\Leftrightarrow \mathcal{U}_{3k,k}(p)$-MDDH assumption, by Lemma 2). However, using a matrix distribution $\mathcal{D}_{3k,k}(p)$ with more compact representation yields a more efficient scheme, secure under the $\mathcal{D}_{3k,k}(p)$-MDDH assumption (see Remark 5).

Our construction

$\underline{\mathsf{Gen}_{\mathsf{KEM}}(1^\lambda)}$:
$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$; $\mathbf{M} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}(p)$
$\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$
$\mathsf{pk} := \left(\mathcal{G}, [\mathbf{M}], \left([\mathbf{M}^\top \mathbf{k}_{j,\beta}]\right)_{1\le j\le\lambda, 0\le\beta\le 1}\right)$
$\mathsf{sk} := (\mathbf{k}_{j,\beta})_{1\le j\le\lambda, 0\le\beta\le 1}$
Return $(\mathsf{pk}, \mathsf{sk})$

$\underline{\mathsf{Enc}_{\mathsf{KEM}}(\mathsf{pk}, \tau)}$:
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$; $C := [\mathbf{r}^\top \mathbf{M}^\top]$
$\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$
$K := [\mathbf{r}^\top \cdot \mathbf{M}^\top \mathbf{k}_\tau]$
Return $(C, K) \in \mathbb{G}^{1\times 3k} \times \mathbb{G}$

$\underline{\mathsf{Dec}_{\mathsf{KEM}}(\mathsf{pk}, \mathsf{sk}, \tau, C)}$:
$\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$
Return $K := C \cdot \mathbf{k}_\tau$

Figure 3.2: $\mathcal{KEM}$, an IND-PCA-secure KEM under the $\mathcal{U}_k(p)$-MDDH assumption, with tag-space $\mathcal{T} = \{0,1\}^\lambda$. Here, GGen is a prime-order group generator (see Section 2.2.1).

Remark 5: On the use of the $\mathcal{U}_k(p)$-MDDH assumption

In our scheme, we use a matrix distribution $\mathcal{U}_{3k,k}(p)$ for the matrix $\mathbf{M}$, therefore proving security under the $\mathcal{U}_{3k,k}(p)$-MDDH assumption $\Leftrightarrow$ $\mathcal{U}_k(p)$-MDDH assumption (see Lemma 3). This is for simplicity of the presentation. However, for efficiency, one may want to use an assumption with a more compact representation, such as the $\mathcal{CI}_{3k,k}$-MDDH assumption [MRV16] with representation size $2k$ instead of $3k^2$ for $\mathcal{U}_{3k,k}(p)$.

Perfect correctness. It follows readily from the fact that for all $\mathbf{r} \in \mathbb{Z}_p^k$ and $C = \mathbf{r}^\top \mathbf{M}^\top$, for all $\mathbf{k} \in \mathbb{Z}_p^{3k}$:

$$\mathbf{r}^\top (\mathbf{M}^\top \mathbf{k}) = C \cdot \mathbf{k}.$$

Security proof

Theorem 7: IND-PCA security

The tag-based Key Encapsulation Mechanism $\mathcal{KEM}$ defined in Figure 3.2 is IND-PCA secure if the $\mathcal{U}_k(p)$-MDDH assumption holds in $\mathbb{G}$. Namely, for any adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ such that $\mathbf{T}(\mathcal{B}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot \mathsf{poly}(\lambda)$ and

$$\mathsf{Adv}^{\mathsf{IND\text{-}PCA}}_{\mathcal{KEM},\mathcal{A}}(\lambda) \leq (4\lambda + 1) \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + (Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot 2^{-\Omega(\lambda)},$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

game	$\mathbf{y}$ uniform in:	$\mathbf{k}'_\tau$ used by EncO and DecO	justification/remark
G_0	$\mathsf{Span}(\mathbf{M})$	$\mathbf{k}_\tau$	actual scheme
G_1	$\boxed{\mathbb{Z}_q^{3k}}$	$\mathbf{k}_\tau$	$\mathcal{U}_{3k,k}$-MDDH on $[\mathbf{M}]$
$\mathsf{G}_{2.i}$	$\mathbb{Z}_q^{3k}$	$\mathbf{k}_\tau + \boxed{\mathbf{M}^\perp \mathsf{RF}_i(\tau_{\mid i})}$	$\mathsf{G}_1 \equiv \mathsf{G}_{2.0}$
$\mathsf{G}_{2.i.1}$	$\tau_{i+1} = 0$: $\boxed{\mathsf{Span}(\mathbf{M}, \mathbf{M}_0)}$ $\tau_{i+1} = 1$: $\boxed{\mathsf{Span}(\mathbf{M}, \mathbf{M}_1)}$	$\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_i(\tau_{\mid i})$	$\mathcal{U}_{3k,k}$-MDDH on $[\mathbf{M}_0]$ $\mathcal{U}_{3k,k}$-MDDH on $[\mathbf{M}_1]$
$\mathsf{G}_{2.i.2}$	$\tau_{i+1} = 0$: $\mathsf{Span}(\mathbf{M}, \mathbf{M}_0)$ $\tau_{i+1} = 1$: $\mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$	$\mathbf{k}_\tau + \boxed{\mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{\mid i+1})} + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{\mid i})$	Cramer-Shoup argument
$\mathsf{G}_{2.i.3}$	$\tau_{i+1} = 0$: $\mathsf{Span}(\mathbf{M}, \mathbf{M}_0)$ $\tau_{i+1} = 1$: $\mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$	$\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{\mid i+1}) + \boxed{\mathbf{M}_1^* \mathsf{RF}_{i+1}^{(1)}(\tau_{\mid i+1})}$	Cramer-Shoup argument
$\mathsf{G}_{2.i+1}$	$\boxed{\mathbb{Z}_q^{3k}}$	$\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_{i+1}(\tau_{\mid i+1})$	$\mathcal{U}_{3k,k}$-MDDH on $[\mathbf{M}_0]$ and $[\mathbf{M}_1]$

Figure 3.3: Sequence of games for the proof of Theorem 7. Throughout, we have (i) $\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$; (ii) $\mathsf{EncO}(\tau) = ([\mathbf{y}], K_b)$ where $K_0 = [\mathbf{y}^\top \mathbf{k}'_\tau]$ and $K_1 \leftarrow_{\mathrm{R}} \mathbb{G}$; (iii) $\mathsf{DecO}(\tau, [\mathbf{y}], \widehat{K})$ computes the encapsulation key $K := [\mathbf{y}^\top \cdot \mathbf{k}'_\tau]$. Here, $(\mathbf{M}_0^*, \mathbf{M}_1^*)$ is a basis for $\mathsf{Span}(\mathbf{M}^\perp)$, so that $\mathbf{M}_1^\top \mathbf{M}_0^* = \mathbf{M}_0^\top \mathbf{M}_1^* = \mathbf{0}$, and we write $\mathbf{M}^\perp \mathsf{RF}_i(\tau_{\mid i}) := \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{\mid i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{\mid i})$. The second column shows which set $\mathbf{y}$ is uniformly picked from by EncO, the third column shows the value of $\mathbf{k}'_\tau$ used by both EncO and DecO.

Proof of Theorem 7. We proceed via a series of hybrid games described in Figure 3.4 and 3.5 and for any game G, we use $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ to denote the advantage of $\mathcal{A}$ in game G. We also give a high-level picture of the proof in Figure 3.3, summarizing the sequence of games.

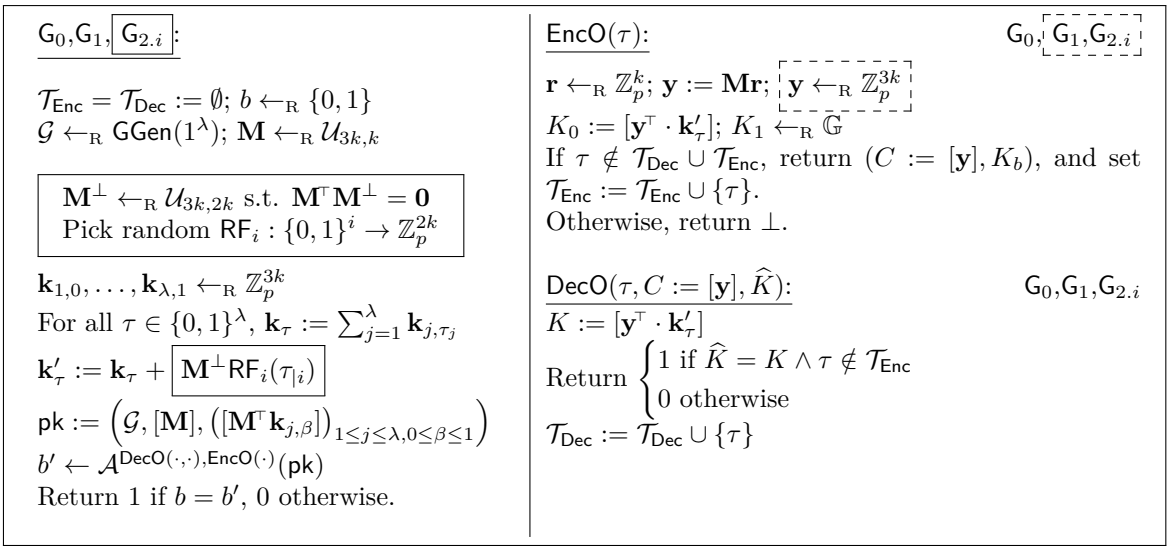


Figure 3.4: Games for the proof of Theorem 7. In each procedure, the components inside a solid (dotted) frame are only present in the games marked by a solid (dotted) frame.

- To go from game G_0 to G_1, we use the MDDH assumption to "tightly" switch the distribution of all the challenge ciphertexts. In Lemma 12, we build an adversary $\mathcal{B}_0$ such that:
$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \le \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_0}(\lambda) + \frac{1}{p-1}.$$
- In Lemma 13, we show that the game G_1 and $\mathsf{G}_{2.0}$ are identically distributed.
- For all $0 \le i \le \lambda - 1$, we build in Lemma 14 an adversary $\mathcal{B}_{2.i}$ such that:
$$|\mathsf{Adv}_{\mathsf{G}_{2.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i+1}}(\mathcal{A})| \le 4 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{2.i}}(\lambda) + \frac{4Q_{\mathsf{Dec}} + 2k}{p} + \frac{4}{p-1},$$
where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO.
- In Lemma 19, we show that $\mathsf{Adv}_{\mathsf{G}_{2.\lambda}}(\mathcal{A}) \le \frac{Q_{\mathsf{Enc}}}{p}$, using a statistical argument.

Putting everything together, we obtain an adversary $\mathcal{B}$ such that $\mathbf{T}(\mathcal{B}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot \mathsf{poly}(\lambda)$ and
$$\mathsf{Adv}^{\mathsf{IND\text{-}PCA}}_{\mathcal{KEM},\mathcal{A}}(\lambda) \le (4\lambda + 1) \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + (Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot 2^{-\Omega(\lambda)},$$
where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$. □

Lemma 12: From game G_0 to game G_1

There exists an adversary $\mathcal{B}_0$ such that $\mathbf{T}(\mathcal{B}_0) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and
$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \le \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_0}(\lambda) + \frac{1}{p-1},$$
where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 12. To go from G_0 to G_1, we switch the distribution of the vectors $[\mathbf{y}]$ sampled by EncO, using the Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH assumption on $[\mathbf{M}]$ (see Definition 12).

We build an adversary $\mathcal{B}'_0$ against the Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH assumption, such that $\mathbf{T}(\mathcal{B}'_0) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ with $\mathsf{poly}(\lambda)$ independent of $\mathbf{T}(\mathcal{A})$, and

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq \mathsf{Adv}^{Q_{\mathsf{Enc}}\text{-}\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}'_0}(\lambda).$$

This implies the lemma by Corollary 1 ($\mathcal{U}_k(p)$-MDDH $\Rightarrow$ Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH).

Upon receiving a challenge $(\mathcal{G}, [\mathbf{M}] \in \mathbb{G}^{3k\times k}, [\mathbf{H}] := [\mathbf{h}_1|\ldots|\mathbf{h}_{Q_{\mathsf{Enc}}}] \in \mathbb{G}^{3k\times Q_{\mathsf{Enc}}})$ for the Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH assumption, $\mathcal{B}'_0$ picks $b \leftarrow_{\mathrm{R}} \{0,1\}$, $\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, generates pk and simulates the oracle DecO as described in Figure 3.4. To simulate EncO on its j'th query, for $j = 1, \ldots, Q_{\mathsf{Enc}}$, $\mathcal{B}'_0$ sets $[\mathbf{y}] := [\mathbf{h}_j]$, and computes K_b as described in Figure 3.4. □

Lemma 13: From game G_1 to game $\mathsf{G}_{2.0}$

For any adversary $\mathcal{A}$, we have: $|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.0}}(\mathcal{A})| = 0$.

Proof of Lemma 13. To go from G_1 to $\mathsf{G}_{2.0}$, we change the distribution of $\mathbf{k}_{1,\beta} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$ for $\beta = 0, 1$, to $\mathbf{k}_{1,\beta} + \mathbf{M}^{\perp}\mathsf{RF}_0(\varepsilon)$, where $\mathbf{k}_{1,\beta} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, $\mathsf{RF}_0(\varepsilon) \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{2k}$, and $\mathbf{M}^{\perp} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,2k}(p)$ such that $\mathbf{M}^{\top}\mathbf{M}^{\perp} = \mathbf{0}$. Note that the extra term $\mathbf{M}^{\perp}\mathsf{RF}_0(\varepsilon)$ does not appear in pk, since $\mathbf{M}^{\top}(\mathbf{k}_{1,\beta} + \mathbf{M}^{\perp}\mathsf{RF}_0(\varepsilon)) = \mathbf{M}^{\top}\mathbf{k}_{1,\beta}$. □

Lemma 14: From game $\mathsf{G}_{2.i}$ to game $\mathsf{G}_{2.i+1}$

For all $0 \leq i \leq \lambda - 1$, there exists an adversary $\mathcal{B}_{2.i}$ such that $\mathbf{T}(\mathcal{B}_{2.i}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and

$$|\mathsf{Adv}_{\mathsf{G}_{2.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i+1}}(\mathcal{A})| \leq 4 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}_{2.i}}(\lambda) + \frac{4Q_{\mathsf{Dec}} + 2k}{p} + \frac{4}{p-1},$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 14. To go from $\mathsf{G}_{2.i}$ to $\mathsf{G}_{2.i+1}$, we introduce intermediate games $\mathsf{G}_{2.i.1}$, $\mathsf{G}_{2.i.2}$ and $\mathsf{G}_{2.i.3}$, defined in Figure 3.5.

- To go from game $\mathsf{G}_{2.i}$ to game $\mathsf{G}_{2.i.1}$, we use the MDDH assumption to "tightly" switch the distribution of all the challenge ciphertexts. We proceed in two steps, first, by changing the distribution of all the ciphertexts with a tag τ such that $\tau_{i+1} = 0$, and then, for those with a tag τ such that $\tau_{i+1} = 1$. We use the MDDH assumption with respect to an independent matrix for each step. We build an adversary in $\mathcal{B}_{2.i.0}$ Lemma 15 such that:

  $$|\mathsf{Adv}_{\mathsf{G}_{2.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.1}}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}_{2.i.0}}(\lambda) + \frac{2}{p-1},$$

  where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively.

- To go from game $\mathsf{G}_{2.i.1}$ to game $\mathsf{G}_{2.i.2}$, we use a variant of the Cramer-Shoup information-theoretic argument to move from RF_i to RF_{i+1}, thereby increasing the entropy of $\mathbf{k}'_\tau$. For the sake of readability, we proceed in two steps: in Lemma 16, we move from RF_i to an hybrid between RF_i and RF_{i+1}, and in Lemma 17, we move to RF_{i+1}. In Lemma 16, we show that:

  $$|\mathsf{Adv}_{\mathsf{G}_{2.i.1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.2}}(\mathcal{A})| \leq \frac{2Q_{\mathsf{Dec}} + 2k}{p},$$

  where Q_{Dec} is the number of times $\mathcal{A}$ queries DecO.

- In Lemma 17, we show that

$$|\mathsf{Adv}_{\mathsf{G}_{2.i.2}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.3}}(\mathcal{A})| \leq \frac{2Q_{\mathsf{Dec}}}{p},$$

  where Q_{Dec} is the number of times $\mathcal{A}$ queries DecO, using a statistical argument.

- The transition between $\mathsf{G}_{2.i.3}$ and game $\mathsf{G}_{2.i+1}$ is symmetric to the transition between game $\mathsf{G}_{2.i}$ and game $\mathsf{G}_{2.i.1}$ (cf. Lemma 15): we use the MDDH assumption to "tightly" switch the distribution of all the challenge ciphertexts in two steps; first, by changing the distribution of all the ciphertexts with a tag τ such that $\tau_{i+1} = 0$, and then, the distribution of those with a tag τ such that $\tau_{i+1} = 1$, using the MDDH assumption with respect to an independent matrix for each step. We build an adversary $\mathcal{B}_{2.i.3}$ in Lemma 18 such that:

$$|\mathsf{Adv}_{\mathsf{G}_{2.i.3}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i+1}}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{2.i.3}}(\lambda) + \frac{2}{p-1},$$

  where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively.

Putting everything together, we obtain the lemma. □

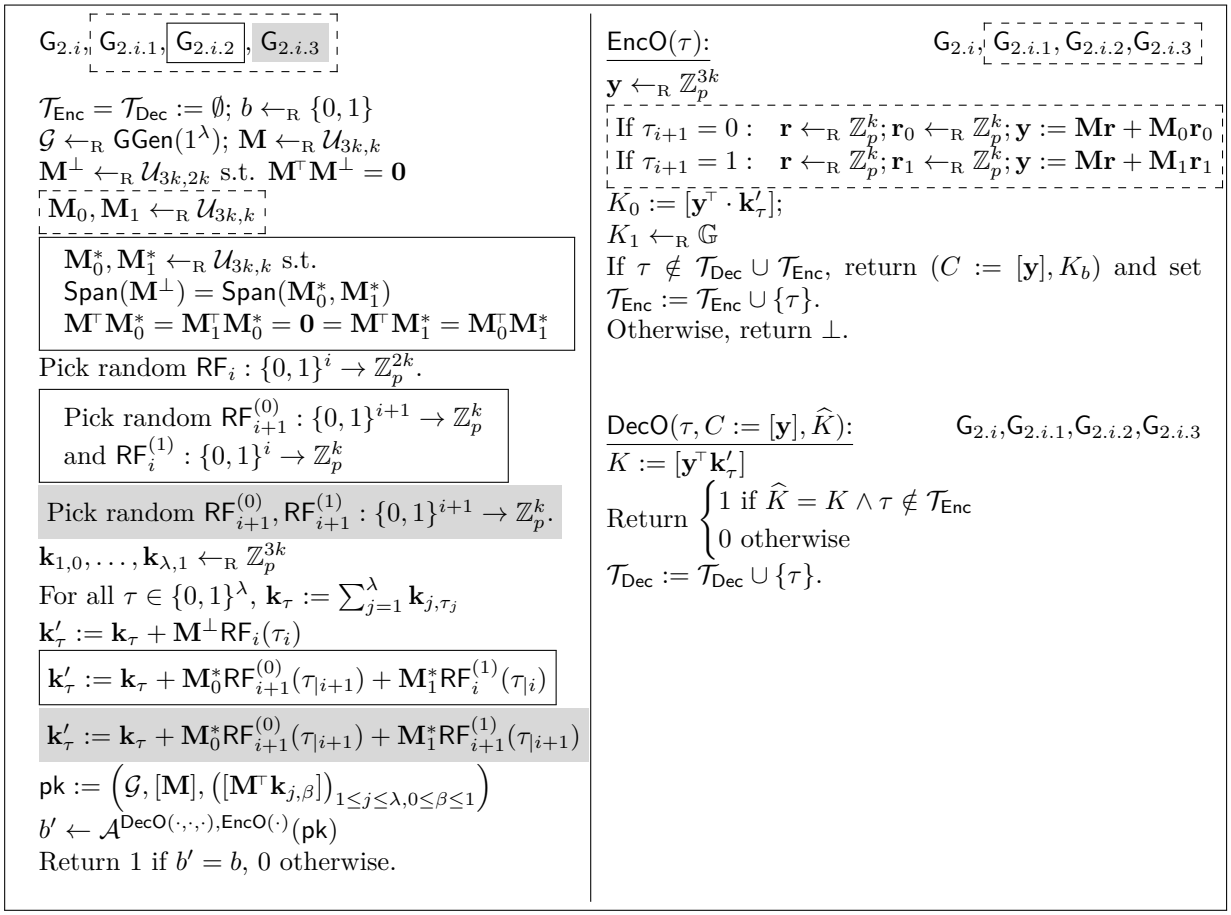


Figure 3.5: Games $\mathsf{G}_{2.i}$ (for $0 \leq i \leq \lambda$),$\mathsf{G}_{2.i.1}$, $\mathsf{G}_{2.i.2}$ and $\mathsf{G}_{2.i.3}$ (for $0 \leq i \leq \lambda - 1$) for the proof of Lemma 14. For all $\tau \in \{0,1\}^\lambda$, we denote by $\tau_{|i}$ the i-bit prefix of τ. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame.

Lemma 15: From game $\mathsf{G}_{2.i}$ to game $\mathsf{G}_{2.i.1}$

For all $0 \leq i \leq \lambda - 1$, there exists an adversary $\mathcal{B}_{2.i.0}$ such that $\mathbf{T}(\mathcal{B}_{2.i.0}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and

$$|\mathsf{Adv}_{\mathsf{G}_{2.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.1}}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{2.i.0}}(\lambda) + \frac{2}{p-1},$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 15. To go from $\mathsf{G}_{2.i}$ to $\mathsf{G}_{2.i.1}$, we switch the distribution of the vectors $[\mathbf{y}]$ sampled by EncO, using the Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH assumption.

We introduce an intermediate game $\mathsf{G}_{2.i.0}$ where $\mathsf{EncO}(\tau)$ is computed as in $\mathsf{G}_{2.i.1}$ if $\tau_{i+1} = 0$, and as in $\mathsf{G}_{2.i}$ if $\tau_{i+1} = 1$. The public key pk, and the oracle DecO are as in $\mathsf{G}_{2.i.1}$. We build adversaries $\mathcal{B}'_{2.i.0}$ and $\mathcal{B}''_{2.i.0}$ such that $\mathbf{T}(\mathcal{B}'_{2.i.0}) \approx \mathbf{T}(\mathcal{B}''_{2.i.0}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ with $\mathsf{poly}(\lambda)$ independent of $\mathbf{T}(\mathcal{A})$, and

Claim 1: $|\mathsf{Adv}_{\mathsf{G}_{2.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.0}}(\mathcal{A})| \leq \mathsf{Adv}^{Q_{\mathsf{Enc}}\text{-}\mathcal{U}_{3k,k}(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}'_{2.i.0}}(\lambda)$.

Claim 2: $|\mathsf{Adv}_{\mathsf{G}_{2.i.0}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.1}}(\mathcal{A})| \leq \mathsf{Adv}^{Q_{\mathsf{Enc}}\text{-}\mathcal{U}_{3k,k}(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}''_{2.i.0}}(\lambda)$.

This implies the lemma by Corollary 1 ($\mathcal{U}_k(p)$-MDDH $\Rightarrow$ Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH).

Let us prove Claim 1. Upon receiving a challenge $(\mathcal{G}, [\mathbf{M}_0] \in \mathbb{G}^{3k \times k}, [\mathbf{H}] := [\mathbf{h}_1 | \ldots | \mathbf{h}_{Q_{\mathsf{Enc}}}] \in \mathbb{G}^{3k \times Q_{\mathsf{Enc}}})$ for the Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH assumption with respect to $\mathbf{M}_0 \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}(p)$, $\mathcal{B}'_{2.i.0}$ does as follows:

pk**:** $\mathcal{B}'_{2.i.0}$ picks $\mathbf{M} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}$, $\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, and computes pk as described in Figure 3.5. For each τ queried to EncO or DecO, it computes on the fly $\mathsf{RF}_i(\tau_{|i})$ and $\mathbf{k}'_\tau := \mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_i(\tau_{|i})$, where $\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$, $\mathsf{RF}_i : \{0,1\}^i \to \mathbb{Z}_p^{2k}$ is a random function, and $\tau_{|i}$ denotes the i-bit prefix of τ (see Figure 3.5). Note that $\mathcal{B}'_{2.i.0}$ can compute efficiently $\mathbf{M}^\perp$ from $\mathbf{M}$.

EncO**:** To simulate the oracle $\mathsf{EncO}(\tau)$ on its j'th query, for $j = 1, \ldots, Q_{\mathsf{Enc}}$, $\mathcal{B}'_{2.i.0}$ computes $[\mathbf{y}]$ as follows:

$$\begin{array}{ll} \text{if } \tau_{i+1} = 0: & \mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k;\ [\mathbf{y}] := [\mathbf{M}\mathbf{r} + \mathbf{h}_j] \\ \text{if } \tau_{i+1} = 1: & [\mathbf{y}] \leftarrow_{\mathrm{R}} \mathbb{G}^{3k} \end{array}$$

This way, $\mathcal{B}'_{2.i.0}$ simulates EncO as in $\mathsf{G}_{2.i.0}$ when $[\mathbf{h}_j] := [\mathbf{M}_0 \mathbf{r}_0]$ with $\mathbf{r}_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and as in $\mathsf{G}_{2.i}$ when $[\mathbf{h}_j] \leftarrow_{\mathrm{R}} \mathbb{G}^{3k}$.

DecO**:** Finally, $\mathcal{B}'_{2.i.0}$ simulates DecO as described in Figure 3.5.

Therefore, $|\mathsf{Adv}_{\mathsf{G}_{2.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.0}}(\mathcal{A})| \leq \mathsf{Adv}^{Q_{\mathsf{Enc}}\text{-}\mathcal{U}_{3k,k}(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}'_{2.i.0}}(\lambda)$.

To prove Claim 2, we build an adversary $\mathcal{B}''_{2.i.0}$ against the Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH assumption with respect to a matrix $\mathbf{M}_1 \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}(p)$, independent from $\mathbf{M}_0$, similarly than $\mathcal{B}'_{2.i.0}$. □

Lemma 16: From game $\mathsf{G}_{2.i.1}$ to game $\mathsf{G}_{2.i.2}$

For all $0 \leq i \leq \lambda - 1$,

$$|\mathsf{Adv}_{\mathsf{G}_{2.i.1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.2}}(\mathcal{A})| \leq \frac{2Q_{\mathsf{Dec}} + 2k}{p},$$

where Q_{Dec} is the number of times $\mathcal{A}$ queries DecO.

Proof of Lemma 16. In $\mathsf{G}_{2.i.2}$, we decompose $\mathsf{Span}(\mathbf{M}^\perp)$ into two subspaces $\mathsf{Span}(\mathbf{M}_0^*)$ and $\mathsf{Span}(\mathbf{M}_1^*)$, and we increase the entropy of the components of $\mathbf{k}'_\tau$ which lie in $\mathsf{Span}(\mathbf{M}_0^*)$. To argue that $\mathsf{G}_{2.i.1}$ and $\mathsf{G}_{2.i.2}$ are statistically close, we use a Cramer-Shoup argument [CS03].

Let us first explain how the matrices $\mathbf{M}_0^*$ and $\mathbf{M}_1^*$ are sampled. Note that with probability at least $1 - \frac{2k}{p}$, $(\mathbf{M}\|\mathbf{M}_0\|\mathbf{M}_1)$ forms a basis of $\mathbb{Z}_p^{3k}$. Therefore, we have $\mathsf{Span}(\mathbf{M}^\perp) = \mathsf{Ker}(\mathbf{M}^\top) = \mathsf{Ker}((\mathbf{M}\|\mathbf{M}_1)^\top) \oplus \mathsf{Ker}((\mathbf{M}\|\mathbf{M}_0)^\top)$. We pick uniformly $\mathbf{M}_0^*$ and $\mathbf{M}_1^*$ in $\mathbb{Z}_p^{3k\times k}$ that generate $\mathsf{Ker}((\mathbf{M}\|\mathbf{M}_1)^\top)$ and $\mathsf{Ker}((\mathbf{M}\|\mathbf{M}_0)^\top)$, respectively (see Figure 3). This way, for all $\tau \in \{0,1\}^\lambda$, we can write

$$\mathbf{M}^\perp \mathsf{RF}_i(\tau_{|i}) := \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}),$$

where $\mathsf{RF}_i^{(0)}, \mathsf{RF}_i^{(1)} : \{0,1\}^i \to \mathbb{Z}_p^k$ are independent random functions.

We define $\mathsf{RF}_{i+1}^{(0)} : \{0,1\}^{i+1} \to \mathbb{Z}_p^k$ as follows:

$$\mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) := \begin{cases} \mathsf{RF}_i^{(0)}(\tau_{|i}) & \text{if } \tau_{i+1} = 0 \\ \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathsf{RF}'^{(0)}_i(\tau_{|i}) & \text{if } \tau_{i+1} = 1 \end{cases}$$

where $\mathsf{RF}'^{(0)}_i : \{0,1\}^i \to \mathbb{Z}_p^k$ is a random function independent from $\mathsf{RF}_i^{(0)}$. This way, $\mathsf{RF}_{i+1}^{(0)}$ is a random function.

We show that the outputs of EncO and DecO are statistically close in $\mathsf{G}_{2.i.1}$ and $\mathsf{G}_{2.i.2}$. We decompose the proof in two cases (delimited with ■): the queries with a tag $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 0$, and the queries with a tag τ such that $\tau_{i+1} = 1$.

Queries with $\tau_{i+1} = 0$:

The only difference between $\mathsf{G}_{2.i.1}$ and $\mathsf{G}_{2.i.2}$ is that $\mathbf{k}'_\tau$ is computed using the random function $\mathsf{RF}_i^{(0)}$ in $\mathsf{G}_{2.i.1}$, whereas it uses the random function $\mathsf{RF}_{i+1}^{(0)}$ in $\mathsf{G}_{2.i.2}$ (see Figure 3.5). Therefore, by definition of $\mathsf{RF}_{i+1}^{(0)}$, for all $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 0$, $\mathbf{k}'_\tau$ is the same in $\mathsf{G}_{2.i.1}$ and $\mathsf{G}_{2.i.2}$, and the outputs of EncO and DecO are identically distributed. ■

Queries with $\tau_{i+1} = 1$:

Observe that for all $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$ and all $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 1$,

$$\begin{aligned} &\overbrace{\mathbf{y}^\top \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}) + \mathbf{M}_0^* \mathsf{RF}'^{(0)}_i(\tau_{|i})\Big)}^{\mathsf{G}_{2.i.2}} \\ &= \mathbf{y}^\top \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i})\Big) + \underbrace{\mathbf{y}^\top \mathbf{M}_0^* \mathsf{RF}'^{(0)}_i(\tau_{|i})}_{=0} \\ &= \overbrace{\mathbf{y}^\top \cdot \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i})\Big)}^{\mathsf{G}_{2.i.1}} \end{aligned}$$

where the second equality uses the fact that $\mathbf{M}^\top \mathbf{M}_0^* = \mathbf{M}_1^\top \mathbf{M}_0^* = \mathbf{0}$ and thus $\mathbf{y}^\top \mathbf{M}_0^* = \mathbf{0}$.

This means that:

- the output of EncO on any input τ such that $\tau_{i+1} = 1$ is identically distributed in $\mathsf{G}_{2.i.1}$ and $\mathsf{G}_{2.i.2}$;

- the output of DecO on any input $(\tau, [\mathbf{y}], \widehat{K})$ where $\tau_{i+1} = 1$, and $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$ is the same in $\mathsf{G}_{2.i.1}$ and $\mathsf{G}_{2.i.2}$.

Henceforth, we focus on the *ill-formed* queries to DecO, namely those corresponding to $\tau_{i+1} = 1$, and $\mathbf{y} \notin \mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$. We introduce intermediate games $\mathsf{G}_{2.i.1.j}$, and $\mathsf{G}'_{2.i.1.j}$ for $j = 0, \ldots, Q_{\mathsf{Dec}}$, defined as follows:

- $\mathsf{G}_{2.i.1.j}$: DecO is as in $\mathsf{G}_{2.i.1}$ except that for the first j times it is queried, it outputs 0 to any ill-formed query. EncO is as in $\mathsf{G}_{2.i.2}$.
- $\mathsf{G}'_{2.i.1.j}$: DecO as in $\mathsf{G}_{2.i.2}$ except that for the first j times it is queried, it outputs 0 to any ill-formed query. EncO is as in $\mathsf{G}_{2.i.2}$.

We show that:

$$\mathsf{G}_{2.i.1} \equiv \mathsf{G}_{2.i.1.0} \approx_s \mathsf{G}_{2.i.1.1} \approx_s \ldots \approx_s \mathsf{G}_{2.i.1.Q_{\mathsf{Dec}}} \equiv \mathsf{G}'_{2.i.1.Q_{\mathsf{Dec}}} \approx_s \mathsf{G}'_{2.i.1.Q_{\mathsf{Dec}}-1} \approx_s \ldots \approx_s \mathsf{G}'_{2.i.1.0} \equiv \mathsf{G}_{2.i.2}$$

where we denote statistical closeness with $\approx_s$ and statistical equality with $\equiv$.
It suffices to show that for all $j = 0, \ldots, Q_{\mathsf{Dec}} - 1$:

Claim 1: in $\mathsf{G}_{2.i.1.j}$, if the $j+1$-st query is ill-formed, then DecO outputs 0 with overwhelming probability $1 - 1/q$ (this implies $\mathsf{G}_{2.i.1.j} \approx_s \mathsf{G}_{2.i.1.j+1}$, with statistical difference $1/q$);

Claim 2: in $\mathsf{G}'_{2.i.1.j}$, if the $j+1$-st query is ill-formed, then DecO outputs 0 with overwhelming probability $1 - 1/q$ (this implies $\mathsf{G}'_{2.i.1.j} \approx_s \mathsf{G}'_{2.i.1.j+1}$, with statistical difference $1/q$)

where the probabilities are taken over the random coins used to generate pk.

Let us prove Claim 1.

Recall that in $\mathsf{G}_{2.i.1.j}$, on its $j+1$-st query, $\mathsf{DecO}(\tau, [\mathbf{y}], \widehat{K})$ computes $K := [\mathbf{y}^\top \mathbf{k}'_\tau]$, where $\mathbf{k}'_\tau := (\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}))$ (see Figure 3.5). We prove that if $(\tau, [\mathbf{y}], \widehat{K})$ is ill-formed, then K is completely hidden from $\mathcal{A}$, up to its $j+1$-st query to DecO. The reason is that the vector $\mathbf{k}_{i+1,1}$ in sk contains some entropy that is hidden from $\mathcal{A}$. This entropy is "released" on the $j+1$-st query to DecO if it is ill-formed. More formally, we use the fact that the vector $\mathbf{k}_{i+1,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$ is identically distributed as $\mathbf{k}_{i+1,1} + \mathbf{M}_0^* \mathbf{w}$, where $\mathbf{k}_{i+1,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, and $\mathbf{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. We show that $\mathbf{w}$ is completely hidden from $\mathcal{A}$, up to its $j+1$-st query to DecO.

- The public key pk does not leak any information about $\mathbf{w}$, since

$$\mathbf{M}^\top (\mathbf{k}_{i+1,1} + \mathbf{M}_0^* \mathbf{w}) = \mathbf{M}^\top \mathbf{k}_{i+1,1}.$$

  This is because $\mathbf{M}^\top \mathbf{M}_0^* = \mathbf{0}$.

- The outputs of EncO also hide $\mathbf{w}$.
  - For τ such that $\tau_{i+1} = 0$, $\mathbf{k}'_\tau$ is independent of $\mathbf{k}_{i+1,1}$, and therefore, so does $\mathsf{EncO}(\tau)$.
  - For τ such that $\tau_{i+1} = 1$, and for any $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$, we have:

$$\mathbf{y}^\top (\mathbf{k}'_\tau + \mathbf{M}_0^* \mathbf{w}) = \mathbf{y}^\top \mathbf{k}'_\tau \tag{3.2}$$

    since $\mathbf{M}^\top \mathbf{M}_0^* = \mathbf{M}_1^\top \mathbf{M}_0^* = \mathbf{0}$, which implies $\mathbf{y}^\top \mathbf{M}_0^* = \mathbf{0}$.

- The first j outputs of DecO also hide $\mathbf{w}$.
  - For τ such that $\tau_{i+1} = 0$, $\mathbf{k}'_\tau$ is independent of $\mathbf{k}_{i+1,1}$, and therefore, so does $\mathsf{DecO}([\mathbf{y}], \tau, \widehat{K})$.
  - For τ such that $\tau_{i+1} = 1$ and $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$, the fact that $\mathsf{DecO}(\tau, [\mathbf{y}], \widehat{K})$ is independent of $\mathbf{w}$ follows readily from Equation (3.2).

– For τ such that $\tau_{i+1} = 1$ and $\mathbf{y} \notin \mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$, that is, for an ill-formed query, DecO outputs 0, independently of $\mathbf{w}$, by definition of $\mathsf{G}_{2.i.1.j}$.

This proves that $\mathbf{w}$ is uniformly random from $\mathcal{A}$'s viewpoint.

Finally, because the $j+1$-st query $(\tau, [\mathbf{y}], \widehat{K})$ is ill-formed, we have $\tau_{i+1} = 1$, and $\mathbf{y} \notin \mathsf{Span}(\mathbf{M}, \mathbf{M}_1)$, which implies that $\mathbf{y}^\top \mathbf{M}_0^* \neq \mathbf{0}$. Therefore, the value

$$K = [\mathbf{y}^\top(\mathbf{k}'_\tau + \mathbf{M}_0^* \mathbf{w})] = [\mathbf{y}^\top \mathbf{k}'_\tau + \underbrace{\mathbf{y}^\top \mathbf{M}_0^*}_{\neq \mathbf{0}} \mathbf{w}]$$

computed by DecO is uniformly random over $\mathbb{G}$ from $\mathcal{A}$'s viewpoint. Thus, with probability $1 - 1/q$ over $K \leftarrow_{\mathrm{R}} \mathbb{G}$, we have $\widehat{K} \neq K$, and $\mathsf{DecO}(\tau, [\mathbf{y}], \widehat{K}) = 0$.

We prove Claim 2 similarly, arguing than in $\mathsf{G}'_{2.i.1.j}$, the value $K := [\mathbf{y}^\top \mathbf{k}'_\tau]$, where $\mathbf{k}'_\tau := (\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}))$, computed by $\mathsf{DecO}(\tau, [\mathbf{y}], \widehat{K})$ on its $j+1$-st query, is completely hidden from $\mathcal{A}$, up to its $j+1$-st query to DecO, if $(\tau, [\mathbf{y}], \widehat{K})$ is ill-formed. The argument goes exactly as for Claim 1. ■ □

Lemma 17: From game $\mathsf{G}_{2.i.2}$ to game $\mathsf{G}_{2.i.3}$

For all $0 \leq i \leq \lambda - 1$,

$$|\mathsf{Adv}_{\mathsf{G}_{2.i.2}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i.3}}(\mathcal{A})| \leq \frac{2Q_{\mathsf{Dec}}}{p},$$

where Q_{Dec} is the number of times $\mathcal{A}$ queries DecO.

Proof of Lemma 17. In $\mathsf{G}_{2.i.3}$, we use the same decomposition $\mathsf{Span}(\mathbf{M}^\perp) = \mathsf{Span}(\mathbf{M}_0^*, \mathbf{M}_1^*)$ as that in $\mathsf{G}_{2.i.2}$. The entropy of the components of $\mathbf{k}'_\tau$ that lie in $\mathsf{Span}(\mathbf{M}_1^*)$ increases from $\mathsf{G}_{2.i.2}$ to $\mathsf{G}_{2.i.3}$. To argue that these two games are statistically close, we use a Cramer-Shoup argument [CS03], exactly as for Lemma 16.

We define $\mathsf{RF}_{i+1}^{(1)}\{0,1\}^{i+1} \to \mathbb{Z}_p^k$ as follows:

$$\mathsf{RF}_{i+1}^{(1)}(\tau_{|i+1}) := \begin{cases} \mathsf{RF}_i^{(1)}(\tau_{|i}) + \mathsf{RF}'^{(1)}_i(\tau_{|i}) & \text{if } \tau_{i+1} = 0 \\ \mathsf{RF}_i^{(1)}(\tau_{|i}) & \text{if } \tau_{i+1} = 1 \end{cases}$$

where $\mathsf{RF}'^{(1)}_i : \{0,1\}^i \to \mathbb{Z}_p^k$ is a random function independent from $\mathsf{RF}_i^{(1)}$. This way, $\mathsf{RF}_{i+1}^{(1)}$ is a random function.

We show that the outputs of EncO and DecO are statistically close in $\mathsf{G}_{2.i.1}$ and $\mathsf{G}_{2.i.2}$. We decompose the proof in two cases (delimited with ■): the queries with a tag $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 0$, and the queries with tag τ such that $\tau_{i+1} = 1$.

Queries with $\tau_{i+1} = 1$:

The only difference between $\mathsf{G}_{2.i.2}$ and $\mathsf{G}_{2.i.3}$ is that $\mathbf{k}'_\tau$ is computed using the random function $\mathsf{RF}_i^{(1)}$ in $\mathsf{G}_{2.i.2}$, whereas it uses the random function $\mathsf{RF}_{i+1}^{(1)}$ in $\mathsf{G}_{2.i.3}$ (see Figure 3.5). Therefore, by definition of $\mathsf{RF}_{i+1}^{(1)}$, for all $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 1$, $\mathbf{k}'_\tau$ is the same in $\mathsf{G}_{2.i.2}$ and $\mathsf{G}_{2.i.3}$, and the outputs of EncO and DecO are identically distributed. ■

Queries with $\tau_{i+1} = 0$:

Observe that for all $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \mathbf{M}_0)$ and all $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 0$,

$$\begin{aligned}
&\overbrace{\mathbf{y}^\top \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}'^{(1)}_i(\tau_{|i}) \Big)}^{\mathsf{G}_{2.i.3}} \\
&= \mathbf{y}^\top \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}) \Big) + \underbrace{\mathbf{y}^\top \mathbf{M}_1^* \mathsf{RF}'^{(1)}_i(\tau_{|i})}_{=0} \\
&= \overbrace{\mathbf{y}^\top \cdot \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}) \Big)}^{\mathsf{G}_{2.i.2}}
\end{aligned}$$

where the second equality uses the fact $\mathbf{M}^\top \mathbf{M}_1^* = \mathbf{M}_0^\top \mathbf{M}_1^* = \mathbf{0}$, which implies $\mathbf{y}^\top \mathbf{M}_1^* = \mathbf{0}$.

This means that:

- the output of EncO on any input τ such that $\tau_{i+1} = 0$ is identically distributed in $\mathsf{G}_{2.i.2}$ and $\mathsf{G}_{2.i.3}$;
- the output of DecO on any input $(\tau, [\mathbf{y}], \widehat{K})$ where $\tau_{i+1} = 0$, and $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \mathbf{M}_0)$ is the same in $\mathsf{G}_{2.i.2}$ and $\mathsf{G}_{2.i.3}$.

Henceforth, we focus on the *ill-formed* queries to DecO, namely those corresponding to $\tau_{i+1} = 0$, and $\mathbf{y} \notin \mathsf{Span}(\mathbf{M}, \mathbf{M}_0)$. The rest of the proof goes similarly than the proof of Lemma 16. See the latter for further details. ■ □

Lemma 18: From game $\mathsf{G}_{2.i.3}$ to game $\mathsf{G}_{2.i+1}$

For all $0 \leq i \leq \lambda - 1$, there exists an adversary $\mathcal{B}_{2.i.3}$ such that $\mathbf{T}(\mathcal{B}_{2.i.3}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and

$$|\mathsf{Adv}_{\mathsf{G}_{2.i.3}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2.i+1}}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G}, \mathcal{B}_{2.i.3}}(\lambda) + \frac{2}{p-1}$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 18. First, we use the fact that for all $\tau \in \{0,1\}^\lambda$, the vector $\mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) + \mathbf{M}_1^* \mathsf{RF}_{i+1}^{(1)}(\tau_{|i+1})$ is identically distributed to $\mathbf{M}^\perp \mathsf{RF}_{i+1}(\tau_{|i+1})$, where $\mathsf{RF}_{i+1} : \{0,1\}^{i+1} \to \mathbb{Z}_p^{2k}$ is a random function. This is because $(\mathbf{M}_0^*, \mathbf{M}_1^*)$ is a basis of $\mathsf{Span}(\mathbf{M}^\perp)$. That means $\mathcal{A}$'s view can be simulated only knowing $\mathbf{M}^\perp$, and not $\mathbf{M}_0^*, \mathbf{M}_1^*$ explicitly. Then, to go from $\mathsf{G}_{2.i.3}$ to $\mathsf{G}_{2.i+1}$, we switch the distribution of the vectors $[\mathbf{y}]$ sampled by EncO, using the Q_{Enc}-fold $\mathcal{U}_{3k,k}(p)$-MDDH assumption (which is equivalent to the $\mathcal{U}_k(p)$-MDDH assumption, see Lemma 2) twice: first with respect to a matrix $\mathbf{M}_0 \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}(p)$ for ciphertexts with $\tau_{i+1} = 0$, then with respect to an independent matrix $\mathbf{M}_1 \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}(p)$ for ciphertexts with $\tau_{i+1} = 1$ (see the proof of Lemma 15 for further details). □

Lemma 19: Game $\mathsf{G}_{2.\lambda}$

For any PPT adversary $\mathcal{A}$, we have: $\mathsf{Adv}_{\mathsf{G}_{2.\lambda}}(\mathcal{A}) \leq \frac{Q_{\mathsf{Enc}}}{p}$.

Proof of Lemma 19. We show that the joint distribution of all the values K_0 computed by EncO is statistically close to uniform over $\mathbb{G}^{Q_{\mathsf{Enc}}}$. Recall that on input τ, $\mathsf{EncO}(\tau)$ computes

$$K_0 := [\mathbf{y}^\top(\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_\lambda(\tau))],$$

where $\mathsf{RF}_\lambda : \{0,1\}^\lambda \to \mathbb{Z}_p^{2k}$ is a random function, and $\mathbf{y} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$ (see Figure 3.4).

We make use of the following properties:

Property 1: all the tags τ queried to EncO, such that $\mathsf{EncO}(\tau) \neq \perp$, are distinct.

Property 2: the outputs of DecO are independent of $\{\mathsf{RF}(\tau) : \tau \in \mathcal{T}_{\mathsf{Enc}}\}$. This is because for all queries $(\tau, [\mathbf{y}], \widehat{K})$ to DecO such that $\tau \in \mathcal{T}_{\mathsf{Enc}}$, $\mathsf{DecO}(\tau, [\mathbf{y}], \widehat{K}) = 0$, independently of $\mathsf{RF}_\lambda(\tau)$, by definition of $\mathsf{G}_{2.\lambda}$.

Property 3: with probability at least $1 - \frac{Q_{\mathsf{Enc}}}{p}$ over the random coins of EncO, all the vectors $\mathbf{y}$ sampled by EncO are such that $\mathbf{y}^\top \mathbf{M}^\perp \neq \mathbf{0}$.

We deduce that the joint distribution of all the values $\mathsf{RF}_\lambda(\tau)$ computed by EncO is uniformly random over $(\mathbb{Z}_p^{2k})^{Q_{\mathsf{Enc}}}$ (from Property 1), independent of the outputs of DecO (from Property 2). Finally, from Property 3, we get that the joint distribution of all the values K_0 computed by EncO is statistically close to uniform over $\mathbb{G}^{Q_{\mathsf{Enc}}}$, since:

$$K_0 := [\mathbf{y}^\top(\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_\lambda(\tau)) = [\mathbf{y}^\top \mathbf{k}_\tau + \underbrace{\mathbf{y}^\top \mathbf{M}^\perp}_{\neq \mathbf{0} \text{ w.h.p.}} \mathsf{RF}_\lambda(\tau)].$$

This means that the values K_0 and K_1 are statistically close, and therefore, $\mathsf{Adv}_{\mathsf{G}_3}(\mathcal{A}) \leq \frac{Q_{\mathsf{Enc}}}{p}$. □

Multi-ciphertext CCA-secure Public Key Encryption scheme

Our construction

We now describe the optimized IND-CCA-secure PKE scheme. Compared to the PCA-secure KEM from Section 3.1, we add an authenticated (symmetric) encryption scheme $(\mathsf{Enc}_{\mathsf{AE}}, \mathsf{Dec}_{\mathsf{AE}})$, and set the KEM tag τ as the hash value of a suitable part of the KEM ciphertext (as explained in the introduction). A formal definition with highlighted differences to our PCA-secure KEM appears in Figure 3.6. We prove the security under the $\mathcal{U}_k(p)$-MDDH assumption.

Perfect correctness. It follows from the perfect correctness of $\mathcal{AE}$ and the fact that for all $\mathbf{r} \in \mathbb{Z}_p^k$ and $\mathbf{y} = \mathbf{M}\mathbf{r}$, for all $\mathbf{k} \in \mathbb{Z}_p^{3k}$:

$$\mathbf{r}^\top(\mathbf{M}^\top \mathbf{k}) = \mathbf{y}^\top \cdot \mathbf{k}.$$

$\underline{\mathsf{Gen}_{\mathsf{PKE}}(1^\lambda)}$:
$\mathcal{G} \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$; $\mathsf{H} \leftarrow_{\mathrm{R}} \mathcal{H}(1^\lambda)$; $\mathbf{M} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}$
$\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{3k}$
$\mathsf{pk} := \left(\mathcal{G}, [\mathbf{M}], \mathsf{H}, \left([\mathbf{M}^\top \mathbf{k}_{j,\beta}]\right)_{1 \le j \le \lambda, 0 \le \beta \le 1}\right)$
$\mathsf{sk} := (\mathbf{k}_{j,\beta})_{1 \le j \le \lambda, 0 \le \beta \le 1}$
Return $(\mathsf{pk}, \mathsf{sk})$

$\underline{\mathsf{Enc}_{\mathsf{PKE}}(\mathsf{pk}, m)}$:
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$; $\mathbf{y} := \mathbf{M}\mathbf{r}$
$\tau := \mathsf{H}([\overline{\mathbf{y}}])$
$\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$
$K := [\mathbf{r}^\top \cdot \mathbf{M}^\top \mathbf{k}_\tau]$
$\phi := \mathsf{Enc}_{\mathsf{AE}}(K, m)$
Return $([\mathbf{y}], \phi)$

$\underline{\mathsf{Dec}_{\mathsf{PKE}}(\mathsf{pk}, \mathsf{sk}, ([\mathbf{y}], \phi))}$:
$\tau := \mathsf{H}([\overline{\mathbf{y}}])$; $\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$;
$K := [\mathbf{y}^\top \mathbf{k}_\tau]$
Return $\mathsf{Dec}_{\mathsf{AE}}(K, \phi)$.

Figure 3.6: $\mathcal{PKE}$, an IND-CCA-secure PKE. We color in gray the differences with $\mathcal{KEM}$, the IND-PCA-secure KEM in Figure 3.2. Here, GGen is a prime-order group generator (see Section 2.2.1) , and $\mathcal{AE} := (\mathsf{Enc}_{\mathsf{AE}}, \mathsf{Dec}_{\mathsf{AE}})$ is an Authenticated Encryption scheme with key-space $\mathcal{K} := \mathbb{G}$ (see Definition 3).

Security proof of $\mathcal{PKE}$

Theorem 8: IND-CCA security

The Public Key Encryption scheme $\mathcal{PKE}$ defined in Figure 3.6 is IND-CCA secure, if the $\mathcal{U}_k(p)$-MDDH assumption holds in $\mathbb{G}$, $\mathcal{AE}$ has one-time privacy and authenticity, and $\mathcal{H}$ generates collision resistant hash functions. Namely, for any adversary $\mathcal{A}$, there exist adversaries $\mathcal{B}$, $\mathcal{B}'$, $\mathcal{B}''$ such that $\mathbf{T}(\mathcal{B}) \approx \mathbf{T}(\mathcal{B}') \approx \mathbf{T}(\mathcal{B}'') \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot \mathsf{poly}(\lambda)$ and

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{PKE},\mathcal{A}}(\lambda) \leq\ & (4\lambda+1) \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) \\ & + (Q_{\mathsf{Enc}}Q_{\mathsf{Dec}} + (4\lambda+2)Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}''}(\lambda) \\ & + \mathsf{Adv}^{\mathsf{CR}}_{\mathcal{H},\mathcal{B}'}(\lambda) + Q_{\mathsf{Enc}}(Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot 2^{-\Omega(\lambda)},\end{aligned} \tag{3.3}$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

We note that the Q_{Enc} and Q_{Dec} factors in (3.4) are only related to $\mathcal{AE}$. Hence, when using a statistically secure authenticated encryption scheme, the corresponding terms in (3.4) become exponentially small.

Remark 6: Extension to the multi-user CCA security

We only provide an analysis in the multi-ciphertext (but single-user) setting. However, we remark (without proof) that our analysis generalizes to the multi-user, multi-ciphertext scenario, similar to [BBM00, HJ12, HKS15]. Indeed, all computational steps (not counting the steps related to the AE scheme) modify all ciphertexts simultaneously, relying for this on the re-randomizability of the $\mathcal{U}_k(p)$-MDDH assumption relative to a fixed matrix $\mathbf{M}$. The same modifications can be made to many $\mathcal{PKE}$ simultaneously by using that the $\mathcal{U}_k(p)$-MDDH Assumption is also re-randomizable across many matrices $\mathbf{M}_i$. (A similar property for the DDH, DLIN, and bilinear DDH assumptions is used in [BBM00], [HJ12], and [HKS15], respectively.)

Proof of Theorem 8. We proceed via a series of hybrid games described in Figures 3.7 and 3.8. Let $\mathcal{A}$ be a PPT adversary. For any game G, we use $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ to denote the advantage of $\mathcal{A}$ in game G_i.

- We transition from game G_0 to game G_1 using the collision resistance of $\mathcal{H}$ and the one-time authenticity of $\mathcal{AE}$ to restrict the oracles DecO and EncO, as described in Figure 3.7. In Lemma 51, we build adversaries $\mathcal{B}_0$ and $\mathcal{B}'_0$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| = 2Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_0}(\lambda) + \mathsf{Adv}^{\mathsf{CR}}_{\mathcal{H},\mathcal{B}'_0}(\lambda) + \frac{Q_{\mathsf{Enc}}(Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}})}{p^k},$$

  where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

- To go from game G_1 to G_2, we use the MDDH assumption to "tightly" switch the distribution of all the challenge ciphertexts. Similarly than in Lemma 12, we obtain an adversary $\mathcal{B}_1$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}_1}(\lambda) + \frac{1}{p-1}.$$

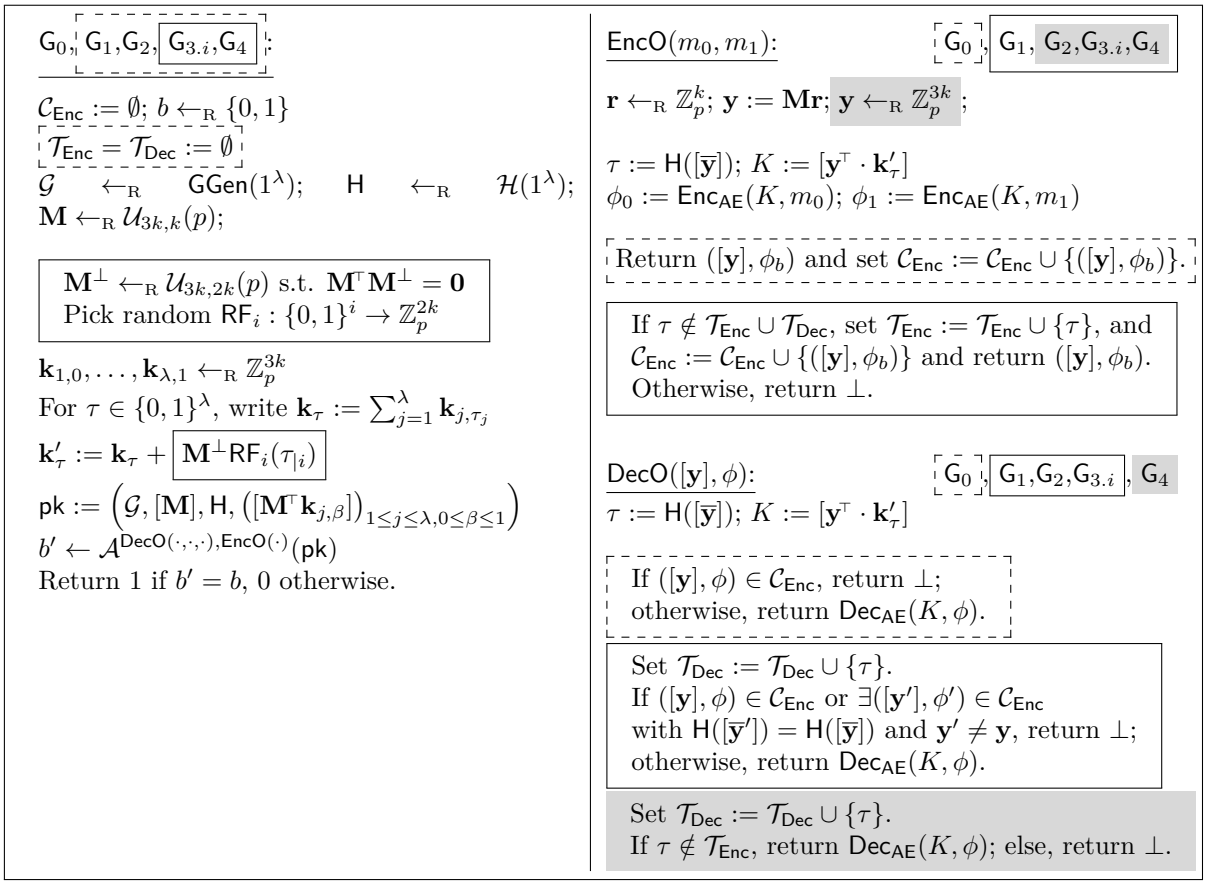


Figure 3.7: Games for the proof of Theorem 8. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame.

- The game G_2 and $\mathsf{G}_{3.0}$ are identically distributed. The argument is exactly as in Lemma 13, thus omitted.

- We build in Lemma 21 adversaries $\mathcal{B}_{3.i}$ and $\mathcal{B}'_{3.i}$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_{3.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i+1}}(\mathcal{A})| \leq 4 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.i}}(\lambda) + 4Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}'_{3.i}}(\lambda) + \frac{4}{p-1} + \frac{2k}{p},$$

  where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively.

- To go from game $\mathsf{G}_{3.\lambda}$ to G_4, we use the one-time authenticity of $\mathcal{AE}$ to restrict the decryption oracle DecO. Namely, in Lemma 26, we build an adversary $\mathcal{B}_{3.\lambda}$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_{3.\lambda}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A})| \leq Q_{\mathsf{Dec}}Q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.\lambda}}(\lambda) + \frac{Q_{\mathsf{Dec}}}{p},$$

  where Q_{Enc}, Q_{Dec} are the number of queries to EncO and DecO, respectively.

- We show in Lemma 27 that there exists an adversary $\mathcal{B}_4$ such that:

$$\mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A}) \leq Q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_4}(\lambda) + \frac{Q_{\mathsf{Enc}}}{p},$$

  where Q_{Enc} denotes the number queries to EncO.

Putting everything together, we obtain adversaries $\mathcal{B}$, $\mathcal{B}'$, $\mathcal{B}''$ such that $\mathbf{T}(\mathcal{B}) \approx \mathbf{T}(\mathcal{B}') \approx \mathbf{T}(\mathcal{B}'') \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot \mathsf{poly}(\lambda)$ and

$$\begin{aligned}\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathcal{PKE},\mathcal{A}}(\lambda) \leq\ & (4\lambda+1) \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) \\ & + (Q_{\mathsf{Enc}}Q_{\mathsf{Dec}} + (4\lambda+2)Q_{\mathsf{Dec}} + Q_{\mathsf{Enc}}) \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}''}(\lambda) \\ & + \mathsf{Adv}^{\mathsf{CR}}_{\mathcal{H},\mathcal{B}'}(\lambda) + Q_{\mathsf{Enc}}(Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot 2^{-\Omega(\lambda)},\end{aligned} \tag{3.4}$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$. □

Lemma 20: From game G_0 to game G_1

There exist adversaries $\mathcal{B}_0$ and $\mathcal{B}'_0$ such that $\mathbf{T}(\mathcal{B}_0) \approx \mathbf{T}(\mathcal{B}'_0) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| = 2Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_0}(\lambda) + \mathsf{Adv}^{\mathsf{CR}}_{\mathcal{H},\mathcal{B}'_0}(\lambda) + \frac{Q_{\mathsf{Enc}}(Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}})}{p^k},$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 20. First, we use the one-time authenticity of $\mathcal{AE}$ to argue that if $\mathcal{A}$ queries DecO on a vector $[\mathbf{y}]$ such that $\mathbf{y} \notin \mathsf{Span}(\mathbf{M})$, then, DecO outputs $\perp$, with all but negligible probability. Second, we use the collision resistance of H to argue that:
(i) if $\mathcal{A}$ queries DecO on $([\mathbf{y}'], \phi')$, where for some previous output $([\mathbf{y}], \phi)$ of EncO, we have: $\mathsf{H}([\overline{\mathbf{y}}]) = \mathsf{H}([\overline{\mathbf{y}'}])$ and $\mathbf{y}' \neq \mathbf{y}$, then, with all but negligible probability, DecO outputs $\perp$;
(ii) every time EncO outputs a vector $[\mathbf{y}]$, its tag $\mathsf{H}([\overline{\mathbf{y}}])$ is fresh (no $[\mathbf{y}']$ with the same tag has been output by EncO or queried to DecO before), with overwhelming probability over EncO's random coins.

We introduce intermediate games $\mathsf{G}_{0.j}$ (resp. $\mathsf{G}_{1.j}$) for $j = 0, \ldots, Q_{\mathsf{Dec}}$, defined as follows: DecO is as in G_0 (resp. G_1) except that for the first j times it is queried, it outputs $\perp$ to any query $([\mathbf{y}], \phi)$ such that $\mathbf{y} \notin \mathsf{Span}(\mathbf{M})$. The public key and EncO are as in G_0 (resp. G_1).

We show that:

$$\mathsf{G}_0 \equiv \mathsf{G}_{0.0} \approx_{\mathcal{AE}} \mathsf{G}_{0.1} \approx_{\mathcal{AE}} \ldots \approx_{\mathcal{AE}} \mathsf{G}_{0.Q_{\mathsf{Dec}}} \approx_{CR} \mathsf{G}_{1.Q_{\mathsf{Dec}}} \approx_{\mathcal{AE}} \ldots \approx_{\mathcal{AE}} \mathsf{G}_{1.0} \equiv \mathsf{G}_1$$

where $\equiv$ denotes statistical equality, $\approx_{\mathcal{AE}}$ denotes indistinguishability based on the security of $\mathcal{AE}$, and $\approx_{CR}$ denotes indistinguishability based on the collision resistance of $\mathcal{H}$.

Namely, we build adversaries $\mathcal{B}_{0.j}$, $\mathcal{B}_{1.j}$ for $j = 0, \ldots, Q_{\mathsf{Dec}} - 1$, and $\mathcal{B}'_0$ such that $\mathbf{T}(\mathcal{B}_{0.j}) \approx \mathbf{T}(\mathcal{B}_{1.j}) \approx \mathbf{T}(\mathcal{B}'_0) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$, where $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$, and such that

Claim 1: $|\mathsf{Adv}_{\mathsf{G}_{0.j}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{0.j+1}}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{0.j}}(\lambda)$ and $|\mathsf{Adv}_{1.j} - \mathsf{Adv}_{1.j+1}| \leq \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{0.j}}(\lambda)$, for $j = 0, \ldots, Q_{\mathsf{Dec}} - 1$.

Claim 2: $|\mathsf{Adv}_{0.Q_{\mathsf{Dec}}} - \mathsf{Adv}_{1.Q_{\mathsf{Dec}}}| \leq \mathsf{Adv}^{\mathsf{CR}}_{\mathcal{H},\mathcal{B}'_0}(\lambda)$.

This implies the lemma.

Let us prove Claim 1. It suffices to show that in $\mathsf{G}_{0.j}$ and $\mathsf{G}_{1.j}$, with all but negligible probability, DecO outputs $\perp$ to its $j+1$-st query if it contains $[\mathbf{y}]$ such that $\mathbf{y} \notin \mathsf{Span}(\mathbf{M})$.

Recall that in both $\mathsf{G}_{0.j}$ and $\mathsf{G}_{1.j}$, on its $j+1$-st query $([\mathbf{y}], \phi)$, DecO computes

$$K := [\mathbf{y}^\top \cdot \mathbf{k}_\tau], \text{ where } \tau = \mathsf{H}([\overline{\mathbf{y}}]) \text{ and } \mathbf{k}_\tau := \sum_{\rho=1}^{\lambda} \mathbf{k}_{\rho,\tau_\rho},$$

and returns $\mathsf{Dec}_{\mathsf{AE}}(K, \phi)$ (or $\perp$, see Figure 3.7). We prove that this value K is hidden from $\mathcal{A}$ up to its $j+1$-st query to DecO. Then, we use the one-time authenticity of $\mathcal{AE}$ to argue that $\mathsf{Dec}_{\mathsf{AE}}(K, \phi) = \perp$ with overwhelming probability.

To prove K is hidden from $\mathcal{A}$, we show that the vectors $\mathbf{k}_{1,0}, \mathbf{k}_{1,1}$ in sk contain some entropy that is hidden from $\mathcal{A}$. More formally, we use the fact that the vectors $\mathbf{k}_{1,\beta} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$ are identically distributed than $\mathbf{k}_{1,\beta} + \mathbf{M}^{\perp}\mathbf{w}$ for $\beta = 0, 1$, where $\mathbf{k}_{1,\beta} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, $\mathbf{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k}$, and $\mathbf{M}^{\perp} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,2k}$ such that $\mathbf{M}^{\top}\mathbf{M}^{\perp} = \mathbf{0}$. We show that $\mathbf{w}$ is hidden from $\mathcal{A}$, up to its $j+1$-st query to DecO.

- The public key pk does not leak any information about $\mathbf{w}$, since

$$\mathbf{M}^{\top}(\mathbf{k}_{1,\beta} + \mathbf{M}^{\perp}\mathbf{w}) = \mathbf{M}^{\top}\mathbf{k}_{1,\beta}.$$

  This is because $\mathbf{M}^{\top}\mathbf{M}^{\perp} = \mathbf{0}$.

- The outputs of EncO also hide $\mathbf{w}$, since for any $\mathbf{y} \in \mathsf{Span}(\mathbf{M})$, we have:

$$\mathbf{y}^{\top}(\mathbf{k}_{\tau} + \mathbf{M}^{\perp}\mathbf{w}) = \mathbf{y}^{\top}\mathbf{k}'_{\tau} \tag{3.5}$$

  since $\mathbf{M}^{\top}\mathbf{M}^{\perp} = \mathbf{0}$ which implies $\mathbf{y}^{\top}\mathbf{M}^{\perp} = \mathbf{0}$.

- The first j outputs of DecO also hide $\mathbf{w}$.
  - For $\mathbf{y} \in \mathsf{Span}(\mathbf{M})$, $\mathsf{DecO}([\mathbf{y}], \phi)$ is independent of $\mathbf{w}$, from Equation (3.5).
  - For $\mathbf{y} \notin \mathsf{Span}(\mathbf{M})$, $\mathsf{DecO}([\mathbf{y}], \phi) = \perp$, independently of $\mathbf{w}$, by definition of $\mathsf{G}_{0.j}$.

Therefore, the value

$$K = [\mathbf{y}^{\top}(\mathbf{k}_{\tau} + \mathbf{M}^{\perp}\mathbf{w})] = [\mathbf{y}^{\top}\mathbf{k}_{\tau} + \underbrace{\mathbf{y}^{\top}\mathbf{M}^{\perp}}_{\neq \mathbf{0}}\mathbf{w}]$$

computed by DecO on its $j+1$-st query, is uniformly random over $\mathbb{G}$ from $\mathcal{A}$'s view, since $\mathbf{y} \notin \mathsf{Span}(\mathbf{M}) \Leftrightarrow \mathbf{y}^{\top}\mathbf{M}^{\perp} \neq \mathbf{0}$.

Then, by one-time authenticity of $\mathcal{AE}$, there exists an adversary $\mathcal{B}_{0.j}$ such that $\mathbf{T}(\mathcal{B}_{0,j}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$, where $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$, and

$$|\mathsf{Adv}_{\mathsf{G}_{0.j}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{0.j+1}}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{0.j}}(\lambda).$$

Let us prove Claim 2. It suffices to show that in $\mathsf{G}_{0.Q_{\mathsf{Dec}}}$:
(i) if DecO is queried on $([\mathbf{y}], \phi)$, and there exists $([\mathbf{y}'], \phi')$ output previously by EncO, with $\mathsf{H}([\overline{\mathbf{y}}]) = \mathsf{H}([\overline{\mathbf{y}'}])$ and $\mathbf{y}' \neq \mathbf{y}$, then, with all but negligible probability, DecO outputs $\perp$;
(ii) every time EncO outputs a vector $[\mathbf{y}]$, its tag $\mathsf{H}([\overline{\mathbf{y}}])$ is fresh (no $[\mathbf{y}']$ with the same tag has been output by EncO or queried to DecO before), with overwhelming probability over its random coins.

We define $\mathcal{B}'_0$ as follows. Upon receiving a challenge $\mathsf{H} \leftarrow_{\mathrm{R}} \mathcal{H}(1^{\lambda})$ for the collision resistance of $\mathcal{H}$, $\mathcal{B}'_0$ picks $b \leftarrow_{\mathrm{R}} \{0,1\}$, $\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, and generates the public key pk, simulates the oracle EncO and DecO as in $\mathsf{G}_{0.Q_{\mathsf{Dec}}}$.

(i) Suppose $\mathcal{B}'_0$ receives some $[\mathbf{y}]$ through a DecO query, such that there is a $[\mathbf{y}']$ from an earlier EncO query with $\mathsf{H}([\overline{\mathbf{y}}]) = \mathsf{H}([\overline{\mathbf{y}'}])$, and $\mathbf{y} \neq \mathbf{y}'$. Then, we distinguish the following cases:

Case 1: $\overline{\mathbf{y}} \neq \overline{\mathbf{y}}'$. Then there is a collision $\mathsf{H}([\overline{\mathbf{y}}]) = \mathsf{H}([\overline{\mathbf{y}}'])$ that $\mathcal{B}'_0$ can directly output.

Case 2: $\overline{\mathbf{y}} = \overline{\mathbf{y}}'$ (but $\mathbf{y} \neq \mathbf{y}'$). Then, $\mathbf{y} \notin \mathsf{Span}(\mathbf{M})$ (because $\mathbf{y} \neq \mathbf{y}'$), and DecO outputs $\perp$, as would happen both in $\mathsf{G}_{0.Q_{\mathsf{Dec}}}$ and $\mathsf{G}_{1.Q_{\mathsf{Dec}}}$.

(ii) First, note that with probability at least $1 - \frac{Q_{\mathsf{Enc}}(Q_{\mathsf{Enc}}+Q_{\mathsf{Dec}})}{p^k}$ over its random coins, EncO samples vectors $[\mathbf{y}]$ whose upper parts $[\overline{\mathbf{y}}]$ are fresh (they are distinct from those previously sampled by EncO, or queried to DecO). Therefore, conditioned on this fact, if $\mathcal{B}'_0$ samples $\tau := \mathsf{H}([\overline{\mathbf{y}}])$ that is not fresh, i.e there exists a pair $([\mathbf{y}'], \mathsf{H}([\overline{\mathbf{y}}']) = \tau)$ previously output by EncO or queried to DecO (along with some symmetric ciphertext ϕ), then we have $\mathsf{H}([\overline{\mathbf{y}}]) = \mathsf{H}([\overline{\mathbf{y}}'])$, and $[\overline{\mathbf{y}}] \neq [\overline{\mathbf{y}}']$, that is, $\mathcal{B}'_0$ finds a collision.

Summarizing, both games $\mathsf{G}_{0.Q_{\mathsf{Dec}}}$ and $\mathsf{G}_{1.Q_{\mathsf{Dec}}}$ proceed identically (as simulated by $\mathcal{B}'_0$), unless (i) Case 1 occurs, or (ii) EncO samples a tag that was output or queried before, in which case $\mathcal{B}'_0$ finds a collision, with overwhelming probability over its random coins. □

Lemma 21: From game $\mathsf{G}_{3.i}$ to game $\mathsf{G}_{3.i+1}$

For all $0 \leq i \leq \lambda - 1$, there exist adversaries $\mathcal{B}_{3.i}$ and $\mathcal{B}'_{3.i}$ such that $\mathbf{T}(\mathcal{B}_{3.i}) \approx \mathbf{T}(\mathcal{B}'_{3.i}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and

$$|\mathsf{Adv}_{\mathsf{G}_{3.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i+1}}(\mathcal{A})| \leq 4 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.i}}(\lambda) + 4Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}'_{3.i}}(\lambda) + \frac{4}{p-1} + \frac{2k}{p},$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 21. To go from $\mathsf{G}_{3.i}$ to $\mathsf{G}_{3.i+1}$, we introduce intermediate games $\mathsf{G}_{3.i.1}$, $\mathsf{G}_{3.i.2}$ and $\mathsf{G}_{3.i.3}$, defined in Figure 3.5.

- To go from game $\mathsf{G}_{3.i}$ to game $\mathsf{G}_{3.i.1}$, we use the MDDH Assumption to "tightly" switch the distribution of all the challenge ciphertexts, as in Lemma 15 in Section 3.1. We proceed in two steps, first, by changing the distribution of all the ciphertexts with a tag τ such that $\tau_{i+1} = 0$, and then, for those with a tag τ such that $\tau_{i+1} = 1$. We use the MDDH Assumption with respect to an independent matrix for each step. We build an adversary $\mathcal{B}_{3.i.0}$ in Lemma 22 such that:

$$|\mathsf{Adv}_{\mathsf{G}_{3.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i.1}}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.i.0}}(\lambda) + \frac{2}{p-1}.$$

- To go from game $\mathsf{G}_{3.i.1}$ to game $\mathsf{G}_{3.i.2}$, we use a computational variant of the Cramer-Shoup information-theoretic argument to move from RF_i to RF_{i+1}, thereby increasing the entropy of $\mathbf{k}'_\tau$, as in Lemma 16, in Section 3.1. For the sake of readability, we proceed in two steps: in Lemma 23, we move from RF_i to an hybrid between RF_i and RF_{i+1}, and in Lemma 24, we move to RF_{i+1}. Overall, we build in Lemma 23 an adversary $\mathcal{B}_{3.i.1}$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_{3.i.1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i.2}}(\mathcal{A})| \leq 2Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.i.1}}(\lambda) + \frac{2k}{p},$$

  where Q_{Dec} denotes the number of queries to DecO.

- In Lemma 24, we build an adversary $\mathcal{B}_{3.i.2}$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_{3.i.2}}(\mathcal{A}) - \mathsf{Adv}_{3.i.3}| \leq 2Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.i.2}}(\lambda),$$

  where Q_{Dec} denotes the number of queries to DecO.

- The transition between $\mathsf{G}_{3.i.3}$ and game $\mathsf{G}_{3.i+1}$ is symmetric to the transition between $\mathsf{G}_{3.i}$ and $\mathsf{G}_{3.i.1}$ (cf. Lemma 22): we use the MDDH Assumption to "tightly" switch the distribution of all the challenge ciphertexts in two steps; first, by changing the distribution

$\mathsf{G}_{3.i}$, $\mathsf{G}_{3.i.1}$, $\mathsf{G}_{3.i.2}$, $\mathsf{G}_{3.i.3}$:

$\mathcal{C}_{\mathsf{Enc}} := \emptyset$; $b \leftarrow_{\mathrm{R}} \{0,1\}$
$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$; $\mathsf{H} \leftarrow_{\mathrm{R}} \mathcal{H}(1^\lambda)$;
$\mathbf{M} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}(p)$
$\mathbf{M}^\perp \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,2k}$ s.t. $\mathbf{M}^\top \mathbf{M}^\perp = \mathbf{0}$
$\mathbf{M}_0, \mathbf{M}_1 \leftarrow_{\mathrm{R}} \mathcal{U}_{2k,k}$
$\mathbf{M}_0^*, \mathbf{M}_1^* \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}$ s.t.
$\mathsf{Span}(\mathbf{M}^\perp) = \mathsf{Span}(\mathbf{M}_0^*, \mathbf{M}_1^*)$
$\mathbf{M}^\top \mathbf{M}_0^* = \binom{\mathbf{0}}{\mathbf{M}_1}^\top \mathbf{M}_0^* = \mathbf{0}$
$\mathbf{M}^\top \mathbf{M}_1^* = \binom{\mathbf{0}}{\mathbf{M}_0}^\top \mathbf{M}_1^* = \mathbf{0}$
Pick random $\mathsf{RF}_i : \{0,1\}^i \to \mathbb{Z}_q^{2k}$.
Pick random $\mathsf{RF}_{i+1}^{(0)} : \{0,1\}^{i+1} \to \mathbb{Z}_q^k$ and $\mathsf{RF}_i^{(1)} : \{0,1\}^i \to \mathbb{Z}_q^k$
Pick random $\mathsf{RF}_{i+1}^{(0)}, \mathsf{RF}_{i+1}^{(1)} : \{0,1\}^{i+1} \to \mathbb{Z}_q^k$.
$\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{3k}$
For all $\tau \in \{0,1\}^\lambda$, $\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$
$\mathbf{k}'_\tau := \mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_i(\tau_{|i})$
$\mathbf{k}'_\tau := \mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i})$
$\mathbf{k}'_\tau := \mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) + \mathbf{M}_1^* \mathsf{RF}_{i+1}^{(1)}(\tau_{|i+1})$
Return $\mathsf{pk} := \left(\mathcal{G}, [\mathbf{M}], \mathsf{H}, \left([\mathbf{M}^\top \mathbf{k}_{j,\beta}]\right)_{1 \le j \le \lambda, 0 \le \beta \le 1}\right)$

$\underline{\mathsf{EncO}(m_0, m_1)}$: $\mathsf{G}_{3.i}$, $\mathsf{G}_{3.i.1}$, $\mathsf{G}_{3.i.2}$, $\mathsf{G}_{3.i.3}$
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$; $\overline{\mathbf{y}} := \overline{\mathbf{M}}\mathbf{r}$; $\tau := \mathsf{H}([\overline{\mathbf{y}}])$; $\underline{\mathbf{y}} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{2k}$
If $\tau_{i+1} = 0$: $\mathbf{r}_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$; $\underline{\mathbf{y}} := \underline{\mathbf{M}}\mathbf{r} + \mathbf{M}_0\mathbf{r}_0$
If $\tau_{i+1} = 1$: $\mathbf{r}_1 \leftarrow_{\mathrm{R}} \mathbb{Z}_q^k$; $\underline{\mathbf{y}} := \underline{\mathbf{M}}\mathbf{r} + \mathbf{M}_1\mathbf{r}_1$
$K := [\mathbf{y}^\top \cdot \mathbf{k}'_\tau]$
$\phi_0 := \mathsf{Enc}_{\mathsf{AE}}(K, m_0)$; $\phi_1 := \mathsf{Enc}_{\mathsf{AE}}(K, m_1)$
If $\tau \notin \mathcal{T}_{\mathsf{Enc}} \cup \mathcal{T}_{\mathsf{Dec}}$, return $([\mathbf{y}], \phi_b)$, set $\mathcal{T}_{\mathsf{Enc}} := \mathcal{T}_{\mathsf{Enc}} \cup \{\tau\}$ and $\mathcal{C}_{\mathsf{Enc}} := \mathcal{C}_{\mathsf{Enc}} \cup \{([\mathbf{y}], \phi_b)\}$. Otherwise, return $\perp$.

$\underline{\mathsf{DecO}([\mathbf{y}], \phi)}$: $\mathsf{G}_{3.i}$,$\mathsf{G}_{3.i.1}$,$\mathsf{G}_{3.i.2}$,$\mathsf{G}_{3.i.3}$
$\tau := \mathsf{H}(\overline{\mathbf{y}})$; $K := [\mathbf{y}^\top \mathbf{k}'_\tau]$
If $([\mathbf{y}], \phi) \in \mathcal{C}_{\mathsf{Enc}}$ or $\exists([\mathbf{y}'], \phi') \in \mathcal{C}_{\mathsf{Enc}}$ with $\mathsf{H}([\overline{\mathbf{y}}']) = \mathsf{H}([\overline{\mathbf{y}}])$ and $\mathbf{y}' \neq \mathbf{y}$, return $\perp$; otherwise, return $\mathsf{Dec}_{\mathsf{AE}}(K, \phi)$. Set $\mathcal{T}_{\mathsf{Dec}} := \mathcal{T}_{\mathsf{Dec}} \cup \{\tau\}$.

Figure 3.8: Games $\mathsf{G}_{3.i}$ (for $0 \le i \le \lambda$),$\mathsf{G}_{3.i.1}$, $\mathsf{G}_{3.i.2}$ and $\mathsf{G}_{3.i.3}$ (for $0 \le i \le \lambda - 1$) for the proof of Lemma 21. For all $\tau \in \{0,1\}^\lambda$, we denote by $\tau_{|i}$ the i-bit prefix of τ. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame.

of all the ciphertexts with a tag τ such that $\tau_{i+1} = 0$, and then, the distribution of those with a tag τ such that $\tau_{i+1} = 1$, using the MDDH assumption with respect to an independent matrix for each step. In Lemma 24, we build an adversary $\mathcal{B}_{3.i.3}$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_{3.i.2}}(\mathcal{A}) - \mathsf{Adv}_{3.i.3}| \leq\leq 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.i.3}}(\lambda) + \frac{2}{p-1}.$$

Putting everything together, we obtain the lemma. □

Lemma 22: From game $\mathsf{G}_{3.i}$ to game $\mathsf{G}_{3.i.1}$

For all $0 \le i \le \lambda - 1$, there exists an adversary $\mathcal{B}_{3.i.0}$ such that $\mathbf{T}(\mathcal{B}_{3.i.0}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and

$$|\mathsf{Adv}_{\mathsf{G}_{3.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i.1}}(\mathcal{A})| \le 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.i.0}}(\lambda) + \frac{2}{p-1},$$

where $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 22. The proof of this lemma is essentially as the proof of Lemma 15, in Section 3.1. The difference is that now, only the lower part of the vectors $[\mathbf{y}]$ sampled by EncO

is randomized using the Q_{Enc}-fold $\mathcal{U}_{2k,k}$-MDDH Assumption. The upper part of $[\mathbf{y}]$ is used to compute the tag τ. We call $\overline{\mathbf{y}}$ and $\underline{\mathbf{y}}$ the upper and lower part of $\mathbf{y}$, respectively.

We introduce an intermediate game $\mathsf{G}_{3.i.0}$ where EncO first picks $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, computes $[\overline{\mathbf{y}}] := [\overline{\mathbf{M}}\mathbf{r}]$, $\tau := \mathsf{H}([\overline{\mathbf{y}}])$, and computes the rest of its output as in $\mathsf{G}_{3.i.1}$ if $\tau_{i+1} = 0$, and as in $\mathsf{G}_{3.i}$ if $\tau_{i+1} = 1$; the public key pk and DecO are as in $\mathsf{G}_{3.i.1}$. We build adversaries $\mathcal{B}'_{3.i.0}$ and $\mathcal{B}''_{3.i.0}$ such that $\mathbf{T}(\mathcal{B}'_{3.i.0}) \approx \mathbf{T}(\mathcal{B}''_{3.i.0}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ with $\mathsf{poly}(\lambda)$ independent of $\mathbf{T}(\mathcal{A})$, and

Claim 1: $|\mathsf{Adv}_{\mathsf{G}_{3.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i.0}}(\mathcal{A})| \leq \mathsf{Adv}^{Q_{\mathsf{Enc}}\text{-}\mathcal{U}_{2k,k}\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}'_{3.i.0}}(\lambda)$.

Claim 2: $|\mathsf{Adv}_{\mathsf{G}_{3.i.0}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i.1}}(\mathcal{A})| \leq \mathsf{Adv}^{Q_{\mathsf{Enc}}\text{-}\mathcal{U}_{2k,k}\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}''_{3.i.0}}(\lambda)$.

This implies the lemma by Corollary 1 ($\mathcal{U}_k(p)$-MDDH $\Rightarrow$ Q_{Enc}-fold $\mathcal{U}_{2k,k}(p)$-MDDH).

Let us prove Claim 1. Upon receiving a challenge $(\mathcal{G}, [\mathbf{M}_0] \in \mathbb{G}^{2k\times k}, [\mathbf{H}] := [\mathbf{h}_1 | \ldots | \mathbf{h}_{Q_{\mathsf{Enc}}}] \in \mathbb{G}^{2k\times Q_{\mathsf{Enc}}})$ for the Q_{Enc}-fold $\mathcal{U}_{2k,k}$-MDDH Assumption with respect to $\mathbf{M}_0 \leftarrow_{\mathrm{R}} \mathcal{U}_{2k,k}$, $\mathcal{B}'_{3.i.0}$ does as follows:

pk: $\mathcal{B}'_{3.i.0}$ picks $\mathbf{M} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}$, $\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, $\mathsf{H} \leftarrow_{\mathrm{R}} \mathcal{H}(1^\lambda)$, and computes pk as described in Figure 3.8. For each τ computed while simulating EncO or DecO, $\mathcal{B}'_{3.i.0}$ computes on the fly $\mathsf{RF}_i(\tau_{|i})$, $\mathbf{k}'_\tau := \mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_i(\tau_{|i})$, where $\mathsf{RF}_i : \{0,1\}^i \to \mathbb{Z}_p^{2k}$ is a random function, $\mathbf{k}_\tau := \sum_{j=1}^{\lambda} \mathbf{k}_{j,\tau_j}$, and $\tau_{|i}$ denotes the i-bit prefix of τ (see Figure 3.8). Note that $\mathcal{B}'_{3.i.0}$ can compute efficiently $\mathbf{M}^\perp$ from $\mathbf{M}$.

$\mathsf{EncO}(m_0, m_1)$: on the j'th query, for $j = 1, \ldots, Q_{\mathsf{Enc}}$, $\mathcal{B}'_{3.i.0}$ samples $\mathbf{r} \leftarrow \mathbb{Z}_p^k$, computes $[\overline{\mathbf{y}}] := [\overline{\mathbf{M}}\mathbf{r}]$, $\tau := \mathsf{H}([\overline{\mathbf{y}}])$, and computes $[\underline{\mathbf{y}}]$ as follows:

$$\begin{array}{ll} \text{if } \tau_{i+1} = 0: & [\underline{\mathbf{y}}] := [\underline{\mathbf{M}}\mathbf{r} + \mathbf{h}_j] \\ \text{if } \tau_{i+1} = 1: & [\underline{\mathbf{y}}] \leftarrow_{\mathrm{R}} \mathbb{G}^{2k} \end{array}$$

This way, $\mathcal{B}'_{3.i.0}$ simulates EncO as in $\mathsf{G}_{3.i.0}$ when $[\mathbf{h}_j] := [\mathbf{M}_0\mathbf{r}_0]$ with $\mathbf{r}_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and as in $\mathsf{G}_{3.i}$ when $[\mathbf{h}_j] \leftarrow_{\mathrm{R}} \mathbb{G}^{2k}$.

$\mathsf{DecO}(C, \phi)$: Finally, $\mathcal{B}'_{3.i.0}$ simulates DecO as described in Figure 3.8.

Therefore, $|\mathsf{Adv}_{\mathsf{G}_{3.i}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i.0}}(\mathcal{A})| \leq \mathsf{Adv}^{Q_{\mathsf{Enc}}\text{-}\mathcal{U}_{2k,k}(p)\text{-}\mathrm{MDDH}}_{\mathbb{G},\mathcal{B}'_{3.i.0}}(\lambda)$.

To prove Claim 2, we build an adversary $\mathcal{B}''_{3.i.0}$ against the Q_{Enc}-fold $\mathcal{U}_{2k,k}(p)$-MDDH assumption with respect to a matrix $\mathbf{M}_1 \leftarrow_{\mathrm{R}} \mathcal{U}_{2k,k}$, independent from $\mathbf{M}_0$, similarly than $\mathcal{B}'_{3.i.0}$. □

Lemma 23: From game $\mathsf{G}_{3.i.1}$ to game $\mathsf{G}_{3.i.2}$

For all $0 \leq i \leq \lambda - 1$, there exists an adversary $\mathcal{B}_{3.i.1}$ such that $\mathbf{T}(\mathcal{B}_{3.i.1}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$, and

$$|\mathsf{Adv}_{\mathsf{G}_{3.i.1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.i.2}}(\mathcal{A})| \leq 2Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.i.1}}(\lambda) + \frac{2k}{p}$$

where Q_{Enc}, Q_{Dec} are the number of queries to EncO and DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 23. In $\mathsf{G}_{3.i.2}$, we decompose $\mathsf{Span}(\mathbf{M}^\perp)$ into two spaces $\mathsf{Span}(\mathbf{M}_0^*)$ and $\mathsf{Span}(\mathbf{M}_1^*)$, and we increase the entropy of the vector $\mathbf{k}'_\tau$ computed by EncO and DecO. More precisely, the entropy of the components of $\mathbf{k}'_\tau$ that lie in $\mathsf{Span}(\mathbf{M}_0^*)$ increases from $\mathsf{G}_{3.i.1}$ to $\mathsf{G}_{3.i.2}$. To argue that these two games are computationally indistinguishable, we use a Cramer-Shoup argument [CS03], together with the one-time authenticity of $\mathcal{AE}$.

Let us first explain how the matrices $\mathbf{M}_0^*$ and $\mathbf{M}_1^*$ are sampled. Note that with probability $1 - \frac{2k}{p}$, $(\mathbf{M} \| \binom{\mathbf{0}}{\mathbf{M}_0} \| \binom{\mathbf{0}}{\mathbf{M}_1})$ forms a basis of $\mathbb{Z}_p^{3k}$. Therefore, we have $\mathsf{Span}(\mathbf{M}^\perp) = \mathsf{Ker}(\mathbf{M}^\top) = \mathsf{Ker}\big((\mathbf{M} \| \binom{\mathbf{0}}{\mathbf{M}_1})^\top\big) \oplus \mathsf{Ker}\big((\mathbf{M} \| \binom{\mathbf{0}}{\mathbf{M}_0})^\top\big)$.

We pick uniformly $\mathbf{M}_0^*$ and $\mathbf{M}_1^*$ in $\mathbb{Z}_p^{3k \times k}$ that generates $\mathsf{Ker}\big((\mathbf{M} \| \binom{\mathbf{0}}{\mathbf{M}_1})^\top\big)$ and $\mathsf{Ker}\big((\mathbf{M} \| \binom{\mathbf{0}}{\mathbf{M}_0})^\top\big)$, respectively. This way, for all $\tau \in \{0,1\}^\lambda$, we can write

$$\mathbf{M}^\perp \mathsf{RF}_i(\tau_{|i}) := \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}),$$

where $\mathsf{RF}_i^{(0)}$, $\mathsf{RF}_i^{(1)} : \{0,1\}^i \to \mathbb{Z}_p^k$ are independent random functions.

We define $\mathsf{RF}_{i+1}^{(0)} : \{0,1\}^{i+1} \to \mathbb{Z}_p^k$ as follows:

$$\mathsf{RF}_{i+1}^{(0)}(\tau_{|i+1}) := \begin{cases} \mathsf{RF}_i^{(0)}(\tau_{|i}) & \text{if } \tau_{i+1} = 0 \\ \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathsf{RF}'^{(0)}_i(\tau_{|i}) & \text{if } \tau_{i+1} = 1 \end{cases}$$

where $\mathsf{RF}'^{(0)}_i : \{0,1\}^i \to \mathbb{Z}_p^k$ is a random function independent from $\mathsf{RF}_i^{(0)}$. This way, $\mathsf{RF}_{i+1}^{(0)}$ is a random function.

We show that the outputs of EncO and DecO are computationally indistinguishable in $\mathsf{G}_{3.i.1}$ and $\mathsf{G}_{3.i.2}$. We decompose the proof in two cases (delimited with ■): the queries corresponding to a tag $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 0$, and the queries corresponding to a tag τ such that $\tau_{i+1} = 1$.

Queries with $\tau_{i+1} = 0$:

The only difference between $\mathsf{G}_{3.i.1}$ and $\mathsf{G}_{3.i.2}$ is that $\mathbf{k}'_\tau$ is computed using the random function $\mathsf{RF}_i^{(0)}$ in $\mathsf{G}_{3.i.1}$, whereas it uses the random function $\mathsf{RF}_{i+1}^{(0)}$ in $\mathsf{G}_{3.i.2}$ (see Figure 3.8). Therefore, by definition of $\mathsf{RF}_{i+1}^{(0)}$, for all $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 0$, $\mathbf{k}'_\tau$ is the same in $\mathsf{G}_{3.i.1}$ and $\mathsf{G}_{3.i.2}$, and the outputs of EncO and DecO are identically distributed. ■

Queries with $\tau_{i+1} = 1$:

Observe that for all $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \binom{\mathbf{0}}{\mathbf{M}_1})$ and all $\tau \in \{0,1\}^\lambda$ such that $\tau_{i+1} = 1$,

$$\begin{aligned} &\overbrace{\mathbf{y}^\top \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}) + \mathbf{M}_0^* \mathsf{RF}'^{(0)}_i(\tau_{|i}) \Big)}^{\mathsf{G}_{3.i.2}} \\ &= \mathbf{y}^\top \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}) \Big) + \underbrace{\mathbf{y}^\top \mathbf{M}_0^* \mathsf{RF}'^{(0)}_i(\tau_{|i})}_{=0} \\ &= \overbrace{\mathbf{y}^\top \cdot \Big(\mathbf{k}_\tau + \mathbf{M}_0^* \mathsf{RF}_i^{(0)}(\tau_{|i}) + \mathbf{M}_1^* \mathsf{RF}_i^{(1)}(\tau_{|i}) \Big)}^{\mathsf{G}_{3.i.1}} \end{aligned}$$

where the second equality uses the fact $\mathbf{M}^\top \mathbf{M}_0^* = \binom{\mathbf{0}}{\mathbf{M}_1}^\top \mathbf{M}_0^* = \mathbf{0}$ and thus $\mathbf{y}^\top \mathbf{M}_0^* = \mathbf{0}$.

This means that:

- the outputs of EncO that contains $[\mathbf{y}]$ whose tag $\tau = \mathsf{H}([\overline{\mathbf{y}}])$ is such that $\tau_{i+1} = 1$ are identically distributed in $\mathsf{G}_{3.i.1}$ and $\mathsf{G}_{3.i.2}$;

- the output of DecO on any input $([\mathbf{y}], \phi)$ where $\tau = \mathsf{H}([\overline{\mathbf{y}}])$, $\tau_{i+1} = 1$, and $\mathbf{y} \in \mathsf{Span}(\mathbf{M}, \binom{\mathbf{0}}{\mathbf{M}_1})$ is the same in $\mathsf{G}_{3.i.1}$ and $\mathsf{G}_{3.i.2}$.

Henceforth, we focus on the *ill-formed* queries to DecO, namely those corresponding to $\tau_{i+1} = 1$, and $\mathbf{y} \notin \mathsf{Span}(\mathbf{M}, \binom{\mathbf{0}}{\mathbf{M}_1})$. We introduce intermediate games $\mathsf{G}_{3.i.1.j}$, and $\mathsf{G}'_{3.i.1.j}$ for $j = 0, \ldots, Q_{\mathsf{Dec}}$, defined as follows:

- $\mathsf{G}_{3.i.1.j}$: DecO is as in $\mathsf{G}_{3.i.1}$ except that for the first j times it is queried, it outputs $\perp$ to any ill-formed query. EncO is as in $\mathsf{G}_{3.i.2}$.
- $\mathsf{G}'_{3.i.1.j}$: DecO is as in $\mathsf{G}_{3.i.2}$ except that for the first j times it is queried, it outputs $\perp$ to any ill-formed query. EncO is as in $\mathsf{G}_{3.i.2}$.

We show that:

$$\mathsf{G}_{3.i.1} \equiv \mathsf{G}_{3.i.1.0} \approx_{\mathcal{AE}} \mathsf{G}_{3.i.1.1} \approx_{\mathcal{AE}} \ldots \approx_{\mathcal{AE}} \mathsf{G}_{3.i.1.Q_{\mathsf{Dec}}} \equiv \mathsf{G}'_{3.i.1.Q_{\mathsf{Dec}}}$$
$$\mathsf{G}'_{3.i.1.Q_{\mathsf{Dec}}} \approx_{\mathcal{AE}} \mathsf{G}'_{3.i.1.Q_{\mathsf{Dec}}-1} \approx_{\mathcal{AE}} \ldots \approx_{\mathcal{AE}} \mathsf{G}'_{3.i.1.0} \equiv \mathsf{G}_{3.i.2}$$

where $\equiv$ denote statistical equality, and $\approx_{\mathcal{AE}}$ denotes indistinguishability based on the security of $\mathcal{AE}$.

It suffices to show that for all $j = 0, \ldots, Q_{\mathsf{Dec}} - 1$, there exist adversaries $\mathcal{B}_{3.i.1.j}$ and $\mathcal{B}'_{3.i.1.j}$ against the one-time authenticity of $\mathcal{AE}$, such that $\mathbf{T}(\mathcal{B}_{3.i.1.j}) \approx \mathbf{T}(\mathcal{B}'_{3.i.1.j}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$, with $\mathsf{poly}(\lambda)$ independent of $\mathbf{T}(\mathcal{A})$, and such that:

Claim 1: in $\mathsf{G}_{3.i.1.j}$, if the $j+1$-st query is ill-formed, then DecO outputs $\perp$ with overwhelming probability $1 - \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.i.1.j}}(\lambda)$ (this implies $\mathsf{G}_{3.i.1.j} \approx_{\mathcal{AE}} \mathsf{G}_{3.i.1.j+1}$).

Claim 2: in $\mathsf{G}'_{3.i.1.j}$, if the $j+1$-st query is ill-formed, then DecO outputs 0 with overwhelming probability $1 - \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}'_{3.i.1.j}}(\lambda)$ (this implies $\mathsf{G}'_{3.i.1.j} \approx_{\mathcal{AE}} \mathsf{G}'_{3.i.1.j+1}$).

We prove Claim 1 and 2 as in Lemma 16, in Section 3.1, arguing that the encapsulation key K computed by DecO on an ill-formed $j+1$-st query, is completely hidden from $\mathcal{A}$, up to its $j+1$-st query to DecO. The reason is that the vector $\mathbf{k}_{i+1,1}$ in sk contains some entropy that is hidden from $\mathcal{A}$, and that is "released" on the $j+1$-st query, if it is ill-formed. Then, we use the one-time authenticity of $\mathcal{AE}$ to argue that DecO outputs $\perp$ with all but negligible probability.■

□

Lemma 24: From game $\mathsf{G}_{3.i.2}$ to game $\mathsf{G}_{3.i.3}$

For all $0 \leq i \leq \lambda - 1$, there exists an adversary $\mathcal{B}_{3.i.2}$ such that $\mathbf{T}(\mathcal{B}_{3.i.2}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$,

$$|\mathsf{Adv}_{\mathsf{G}_{3.i.2}}(\mathcal{A}) - \mathsf{Adv}_{3.i.3}| \leq 2Q_{\mathsf{Dec}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.i.2}}(\lambda),$$

where Q_{Enc}, Q_{Dec} are the number of queries to EncO and DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 24. In $\mathsf{G}_{3.i.3}$, we use the same decomposition $\mathsf{Span}(\mathbf{M}^{\perp}) = \mathsf{Span}(\mathbf{M}^*_0, \mathbf{M}^*_1)$ as that in $\mathsf{G}_{3.i.2}$. The entropy of the component of $\mathbf{k}'_\tau$ that lies in $\mathsf{Span}(\mathbf{M}^*_1)$ increases from $\mathsf{G}_{3.i.2}$ to $\mathsf{G}_{3.i.3}$. That is, we use a random function $\mathsf{RF}^{(1)}_{i+1} : \{0,1\}^{i+1} \to \mathbb{Z}_p^k$ in place of the random function $\mathsf{RF}^{(1)}_i : \{0,1\}^i \to \mathbb{Z}_p^k$. To argue that these two games are computationally indistinguishable, we use a computational variant of the Cramer-Shoup argument [CS03], exactly as in the proof of Lemma 23.

We define $\mathsf{RF}^{(1)}_{i+1} \to \mathbb{Z}_p^k$ as follows:

$$\mathsf{RF}^{(1)}_{i+1}(\tau_{|i+1}) := \begin{cases} \mathsf{RF}^{(1)}_i(\tau_{|i}) + \mathsf{RF}'^{(1)}_i(\tau_{|i}) & \text{if } \tau_{i+1} = 0 \\ \mathsf{RF}^{(1)}_i(\tau_{|i}) & \text{if } \tau_{i+1} = 1 \end{cases}$$

where $\mathsf{RF}'^{(1)}_i : \{0,1\}^i \to \mathbb{Z}_p^k$ is a random function independent from $\mathsf{RF}^{(1)}_i$. This way, $\mathsf{RF}^{(1)}_{i+1}$ is a random function.

We show that the outputs of EncO and DecO are computationally indistinguishable in $\mathsf{G}_{3.i.1}$ and $\mathsf{G}_{3.i.2}$, similarly that in the proof of Lemma 17, in Section 3.1 (see the latter for further details). □

Lemma 25: From game $\mathsf{G}_{3.i.3}$ to game $\mathsf{G}_{3.i+1}$

For all $0 \leq i \leq \lambda - 1$, there exists an adversary $\mathcal{B}_{3.i.3}$ such that $\mathbf{T}(\mathcal{B}_{3.i.3}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$ and

$$|\mathsf{Adv}_{3.i.3} - \mathsf{Adv}_{\mathsf{G}_{3.i+1}}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.i.3}}(\lambda) + \frac{2}{p-1},$$

where Q_{Enc}, Q_{Dec} are the number of times $\mathcal{A}$ queries EncO, DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 25. First, we use the fact that for all $\tau \in \{0,1\}^\lambda$, the vector $\mathbf{M}^*_0\mathsf{RF}^{(0)}_{i+1}(\tau_{|i+1}) + \mathbf{M}^*_1\mathsf{RF}^{(1)}_{i+1}(\tau_{|i+1})$ is identically distributed to $\mathbf{M}^\perp\mathsf{RF}_{i+1}(\tau_{|i+1})$, where $\mathsf{RF}_{i+1} : \{0,1\}^{i+1} \to \mathbb{Z}_p^{2k}$ is a random function. This is because $(\mathbf{M}^*_0, \mathbf{M}^*_1)$ is a basis of $\mathsf{Span}(\mathbf{M}^\perp)$. That means $\mathcal{A}$'s view can be simulated only knowing $\mathbf{M}^\perp$, and not $\mathbf{M}^*_0, \mathbf{M}^*_1$ explicitly. Then, to go from $\mathsf{G}_{3.i.3}$ to $\mathsf{G}_{3.i+1}$, we switch the distribution of the vectors $[\mathbf{y}]$ sampled by EncO, using the Q_{Enc}-fold $\mathcal{U}_{2k,k}(p)$-MDDH Assumption (equivalent to the $\mathcal{U}_k$-MDDH Assumption, see Lemma 2) twice: first with respect to a matrix $\mathbf{M}_0 \leftarrow_{\mathrm{R}} \mathcal{U}_{2k,k}(p)$ for ciphertexts with $\tau_{i+1} = 0$, then with respect to an independent matrix $\mathbf{M}_1 \leftarrow_{\mathrm{R}} \mathcal{U}_{2k,k}(p)$ for ciphertexts with $\tau_{i+1} = 1$ (see the proof of Lemma 22 for further details). □

Lemma 26: From game $\mathsf{G}_{3.\lambda}$ to G_4

There exists an adversary $\mathcal{B}_{3.\lambda}$ such that $\mathbf{T}(\mathcal{B}_{3.\lambda}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$, and

$$|\mathsf{Adv}_{\mathsf{G}_{3.\lambda}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A})| \leq Q_{\mathsf{Dec}}Q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.\lambda}}(\lambda) + \frac{Q_{\mathsf{Dec}}}{p},$$

where Q_{Enc}, Q_{Dec} are the number of queries to EncO and DecO, respectively, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 26. We use the one-time authenticity of $\mathcal{AE}$ to argue that with all but negligible probability, DecO outputs $\perp$ on any input $([\mathbf{y}], \phi)$ such that for some previous output $([\mathbf{y}'], \phi')$ of EncO, $\mathsf{H}([\overline{\mathbf{y}'}]) = \mathsf{H}([\overline{\mathbf{y}}])$.

We introduce intermediate games $\mathsf{G}_{3.\lambda.j}$ for $j = 0, \ldots, Q_{\mathsf{Dec}}$, defined as $\mathsf{G}_{3.\lambda}$, except that on its first j query, DecO is as in G_4, that is, it outputs $\perp$ to any query corresponding to a tag τ previously output by EncO.

We show that :

$$\mathsf{G}_{3.\lambda} \equiv \mathsf{G}_{3.\lambda.0} \approx_{\mathcal{AE}} \mathsf{G}_{3.\lambda.1} \approx_{\mathcal{AE}} \ldots \approx_{\mathcal{AE}} \mathsf{G}_{3.\lambda.Q_{\mathsf{Dec}}} \equiv \mathsf{G}_4,$$

where $\equiv$ denotes statistical equality, and $\approx_{\mathcal{AE}}$ denotes indistinguishability based on the security of $\mathcal{AE}$.

Namely, we build adversaries $\mathcal{B}_{3.\lambda.j}$ for $j = 0, \ldots, Q_{\mathsf{Dec}} - 1$, such that $\mathbf{T}(\mathcal{B}_{3.\lambda.j}) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$, where $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$, and

$$|\mathsf{Adv}_{\mathsf{G}_{3.\lambda.j}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.\lambda.j+1}}(\mathcal{A})| \leq Q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.\lambda.j}}(\lambda) + \frac{1}{p}.$$

This implies the lemma.

It suffices to show that in $\mathsf{G}_{3.\lambda.j}$, with all but negligible probability, DecO outputs $\perp$ to its $j+1$-st query if it contains $[\mathbf{y}^\star]$ such that $\mathsf{H}([\overline{\mathbf{y}}^\star]) = \mathsf{H}([\overline{\mathbf{y}}])$, for $[\mathbf{y}]$ that was output previously by EncO.

We build $\mathcal{B}_{3.\lambda.j}$ as follows.

pk **:** Upon receiving the description of $\mathcal{K} := \mathcal{G}$, $\mathcal{B}_{3.\lambda.j}$ picks $\mathbf{M} \leftarrow_{\mathrm{R}} \mathcal{U}_{3k,k}$, $\mathbf{k}_{1,0}, \ldots, \mathbf{k}_{\lambda,1} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, $\mathsf{H} \leftarrow_{\mathrm{R}} \mathcal{H}(1^\lambda)$, and outputs pk as in G_4 (see Figure 3.7). It also picks $j^\star \leftarrow_{\mathrm{R}} \{1, \ldots, Q_{\mathsf{Enc}}\}$, and $b \leftarrow_{\mathrm{R}} \{0,1\}$.

$\mathsf{EncO}(m_0, m_1)$ **:** On the $j^\star$'th query, $\mathcal{B}_{3.\lambda.j}$ picks $\mathbf{y} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$, calls the encryption oracle for $\mathcal{AE}$, $\mathsf{EncO}(m_b, m_b)$ to get $\phi_b := \mathsf{Enc}_{\mathsf{AE}}(K^\star, m_b)$, for a random $K^\star \leftarrow_{\mathrm{R}} \mathbb{G}$. The rest of the simulation goes as in G_4 (see Figure 3.7), that is: if $\mathsf{H}([\overline{\mathbf{y}}]) \notin \mathcal{T}_{\mathsf{Enc}} \cup \mathcal{T}_{\mathsf{Dec}}$, $\mathcal{B}_{3.\lambda.j}$ returns $([\mathbf{y},], \phi_b)$, sets $\mathcal{T}_{\mathsf{Enc}} := \mathcal{T}_{\mathsf{Enc}} \cup \{\mathsf{H}([\overline{\mathbf{y}}])\}$ and $\mathcal{C}_{\mathsf{Enc}} := \mathcal{C}_{\mathsf{Enc}} \cup \{([\mathbf{y}], \phi_b)\}$, otherwise, it returns $\perp$. The other $j \neq j^\star$ queries are simulated as in G_4.

$\mathsf{DecO}([\mathbf{y}], \phi)$**:** the first j queries are simulated as in G_4, the last $Q_{\mathsf{Enc}} - j - 1$ as in $\mathsf{G}_{3.\lambda}$. For the $j+1$-st query $([\mathbf{y}^\star], \phi^\star)$, $\mathcal{B}_{3.\lambda.j}$ calls the decryption oracle for $\mathcal{AE}$, $\mathsf{DecO}([\mathbf{y}^\star], \phi^\star)$ to get $\mathsf{Dec}_{\mathsf{AE}}(K^\star, \phi^\star)$. The rest of the simulation goes as in $\mathsf{G}_{3.i}$, that is, if $([\mathbf{y}^\star], \phi^\star) \in \mathcal{C}_{\mathsf{Enc}}$ or $\exists([\mathbf{y}], \phi) \in \mathcal{C}_{\mathsf{Enc}}$ with $\mathsf{H}([\overline{\mathbf{y}}^\star]) = \mathsf{H}([\overline{\mathbf{y}}])$ and $\mathbf{y}^\star \neq \mathbf{y}$, $\mathcal{B}_{3.\lambda.j}$ returns $\perp$. Otherwise, it returns $\mathsf{Dec}_{\mathsf{AE}}(K^\star, \phi^\star)$. Finally, it sets $\mathcal{T}_{\mathsf{Dec}} := \mathcal{T}_{\mathsf{Dec}} \cup \{\mathsf{H}([\overline{\mathbf{y}}^\star])\}$.

Assume the $j+1$-st query $([\mathbf{y}^\star], \phi^\star)$ to DecO is such that $\mathsf{DecO}([\mathbf{y}^\star], \phi^\star) = \perp$ in G_4, but not in $\mathsf{G}_{3.\lambda.j}$. In particular, that means that there exists $([\mathbf{y}], \phi) \in \mathcal{C}_{\mathsf{Enc}}$ such that $\mathbf{y} = \mathbf{y}^\star$ and $\phi \neq \phi^\star$. Then, with probability $1/Q_{\mathsf{Enc}}$ over the choice of $j^\star$, $([\mathbf{y}], \phi)$ is the $j^\star$'th query of EncO. In that case, we show that $\mathcal{A}$'s view is simulated as in $\mathsf{G}_{3.\lambda.j}$ if DecO is the real decryption oracle, and as in G_4 if it is the "always $\perp$" function. This implies the lemma.

Indeed, the key $K^\star := [\mathbf{y}^{\star\top}(\mathbf{k}_{\tau^\star} + \mathbf{M}^\perp \mathsf{RF}_\lambda(\tau^\star))]$ for $\tau^\star := \mathsf{H}([\overline{\mathbf{y}}^\star])$ is random, independent from $\mathcal{A}$'s view up to its $j+1$-st query on DecO (except what leaks through $\mathsf{Enc}_{\mathsf{AE}}(K^\star, m_b)$). This is because:

1. with probability $1/q$ over the random coins of $\mathcal{B}_{3.\lambda.j}$, $\mathbf{y}^\star \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k} \notin \mathsf{Span}(\mathbf{M})$.
2. for all $[\mathbf{y}]$ contained in EncO outputs or DecO queries that don't output $\perp$, prior to the $j+1$-st DecO query, we have $\mathsf{H}([\overline{\mathbf{y}}]) \neq \tau^\star$, by definition of $\mathsf{G}_{3.\lambda.j}$. That is, the tag $\tau^\star$ is "fresh". Therefore, the key

$$K^\star := [\mathbf{y}^{\star\top}(\mathbf{k}_{\tau^\star} + \mathbf{M}^\perp \mathsf{RF}_\lambda(\tau^\star))] = [\mathbf{y}^\top \mathbf{k}_{\tau^\star} + \underbrace{\mathbf{y}^{\star\top}\mathbf{M}^\perp}_{\neq \mathbf{0}} \mathsf{RF}_\lambda(\tau^\star)]$$

is random, independent of $\mathcal{A}$'s view up to its $j+1$-st query (except what leaks through $\mathsf{Enc}_{\mathsf{AE}}(K^\star, m_b)$).

This proves that

$$|\mathsf{Adv}_{\mathsf{G}_{3.\lambda.j}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.\lambda.j+1}}(\mathcal{A})| \leq Q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_{3.\lambda.j}}(\lambda) + \frac{1}{p}.$$

□

Lemma 27: Game G_4

There exists an adversary $\mathcal{B}_4$ such that $\mathbf{T}(\mathcal{B}_4) \approx \mathbf{T}(\mathcal{A}) + (Q_{\mathsf{Enc}} + Q_{\mathsf{Dec}}) \cdot \mathsf{poly}(\lambda)$, such that

$$\mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A}) \leq Q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\mathsf{ae\text{-}ot}}_{\mathcal{AE},\mathcal{B}_4}(\lambda) + \frac{Q_{\mathsf{Enc}}}{p},$$

where Q_{Enc} denotes the number queries to EncO, and $\mathsf{poly}(\lambda)$ is independent of $\mathbf{T}(\mathcal{A})$.

Proof of Lemma 27. First, we show that the joint distribution of all the values K computed by EncO is statistically close to uniform over $\mathbb{G}^{Q_{\mathsf{Enc}}}$. Then, we use the one-time privacy of $\mathcal{AE}$ on each one of the Q_{Enc} symmetric ciphertexts.

Recall that on input τ, $\mathsf{EncO}(\tau)$ computes

$$K := [\mathbf{y}^\top(\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_\lambda(\tau))],$$

where $\mathsf{RF}_\lambda : \{0,1\}^\lambda \to \mathbb{Z}_p^{2k}$ is a random function, and $\mathbf{y} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{3k}$.

We make use of the following properties:

Property 1: all the tags τ computed by $\mathsf{EncO}(m_0, m_1)$, such that $\mathsf{EncO}(m_0, m_1) \neq \perp$, are distinct.

Property 2: the outputs of DecO are independent of $\{\mathsf{RF}(\tau) : \tau \in \mathcal{T}_{\mathsf{Enc}}\}$. This is because for all queries $([\mathbf{y}], \phi)$ to DecO such that $\mathsf{H}([\overline{\mathbf{y}}]) \in \mathcal{T}_{\mathsf{Enc}}$, $\mathsf{DecO}([\mathbf{y}], \phi) = \perp$, independently of $\mathsf{RF}_\lambda(\tau)$, by definition of G_4.

Property 3: with probability at least $1 - \frac{Q_{\mathsf{Enc}}}{p}$ over the random coins of EncO, all the vectors $\mathbf{y}$ sampled by EncO are such that $\mathbf{y}^\top \mathbf{M}^\perp \neq \mathbf{0}$.

We deduce that the joint distribution of all the values $\mathsf{RF}_\lambda(\tau)$ computed by EncO is uniformly random over $(\mathbb{Z}_p^{2k})^{Q_{\mathsf{Enc}}}$ (from Property 1), independent of the outputs of DecO (from Property 2). Finally, from Property 3, we get that the joint distribution of all the values K computed by EncO is statistically close to uniformly random over $\mathbb{G}^{Q_{\mathsf{Enc}}}$, since:

$$K := [\mathbf{y}^\top(\mathbf{k}_\tau + \mathbf{M}^\perp \mathsf{RF}_\lambda(\tau)) = [\mathbf{y}^\top \mathbf{k}_\tau + \underbrace{\mathbf{y}^\top \mathbf{M}^\perp}_{\neq \mathbf{0} \text{ w.h.p.}} \mathsf{RF}_\lambda(\tau)].$$

Therefore, we can use the one-time privacy of $\mathcal{AE}$ to argue that all symmetric ciphertexts ϕ_b computed by EncO don't reveal b (this uses a hybrid argument over the Q_{Enc} challenge ciphertexts). □

Chapter 4

Multi-Input Inner-Product Functional Encryption from Pairings

Overview of the construction

In this chapter, we present a multi-input functional encryption scheme (MIFE) for inner products based on the MDDH assumption in prime-order *bilinear* groups. The construction appeared in [AGRW17], and was the first MIFE scheme for a non-trivial functionality based on standard cryptographic assumptions with polynomial security loss, for any polynomial number of slots and secure against unbounded collusions. We prove in this thesis a stronger security guarantee than in [AGRW17]. Namely, the novelty here, is that input slots can collude, and should not be able to break the security of the encryption for the other slots. The security notion that captures corruption of input slots is formally described in Definition 23. Moreover, using a single-input FE that is secure in a multi-instance setting, we obtain a multi-input FE (see Figure 4.6) that is more efficient that the original scheme from [AGRW17].

Concretely, the set of functionality $\{F_n\}_{n\in\mathbb{N}}$ we consider is that of "bounded-norm" multi-input inner products: each key is specified by a vector $(\mathbf{y}_1\| \cdots \|\mathbf{y}_n) \in \mathbb{Z}^{mn}$, takes as input n vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$, each of dimension m, and outputs

$$F_n((\mathbf{y}_1\| \ldots, \|\mathbf{y}_n), \mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle.$$

We require that the $\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{y}_1, \ldots, \mathbf{y}_n$ have bounded norm, and inner product is computed over the integers. The functionality is a natural generalization of single-input inner product functionality introduced by Abdalla et. al [ABDP15], and studied in [ABDP15, BJK15, DDM16, ALS16, ABDP16], and captures several useful computations arising in the context of data-mining.

Prior approaches. Prior constructions of MIFE schemes in [BLR+15] require (at least) nm-linear maps for n slots with m-bit inputs as they encode each input bit for each slot into a fresh level of a multilinear map. In addition, there is typically a security loss that is exponential in n due to the combinatorial explosion arising from combining different ciphertexts across the slots. In the case of inner products, one can hope to reduce the multilinearity to n by exploiting linearity as in the single-input FE; indeed, this was achieved in two independent works [LL16, KLM+18][1] showing how to realize a two-slot MIFE for inner products over bilinear groups. We stress that our result is substantially stronger: we show how to realize n-slot MIFE for inner products for any polynomial n over bilinear groups under standard assumptions, while in addition avoiding the exponential security loss. In particular, we deviate

[1] This work is independent of both works.

from the prior approaches of encoding each slot into a fresh level of a multilinear map. We stress that prior to [AGRW17], we did not even have a candidate for 3-slot MIFE for inner products in the generic bilinear group model.

A public-key scheme. Our first observation is that we can build a public-key MIFE for inner product by running n independent copies of a single-input FE for inner products. Combined with existing instantiations of the latter in [ABDP15], this immediately yields a public-key MIFE for inner products under the standard DDH in cyclic group $\mathbb{G}$ (we use the implicit representation of group elements as defined in Section 2.2.1).

In a bit more detail, we recall the DDH-based public-key single-input FE scheme from [ABDP15]:

$$\mathsf{pk} := [\mathbf{w}],\ \mathsf{ct}_{\mathbf{x}} = ([s], [\mathbf{x} + \mathbf{w}s]),\ \mathsf{sk}_{\mathbf{y}} := \langle \mathbf{w}, \mathbf{y} \rangle.$$

Decryption computes $[\langle \mathbf{x}, \mathbf{y} \rangle] = [\mathbf{x} + \mathbf{w}s]^\top \mathbf{y} - [s] \cdot \langle \mathbf{w}, \mathbf{y} \rangle$ and then recovers $\langle \mathbf{x}, \mathbf{y} \rangle$ by computing the discrete log.

Our public-key MIFE scheme is as follows:

$$\begin{aligned} \mathsf{pk} &:= ([\mathbf{w}_1], \ldots, [\mathbf{w}_n]), \\ \mathsf{ct}_{\mathbf{x}_i} &:= ([s_i], [\mathbf{x}_i + \mathbf{w}_i s_i]), \\ \mathsf{sk}_{\mathbf{y}_1, \ldots, \mathbf{y}_n} &:= (\langle \mathbf{w}_1, \mathbf{y}_1 \rangle, \ldots, \langle \mathbf{w}_n, \mathbf{y}_n \rangle). \end{aligned}$$

We note that the encryption of $\mathbf{x}_i$ uses fresh randomness s_i; to decrypt, we need to know each $\langle \mathbf{w}_i, \mathbf{y}_i \rangle$, and not just $\langle \mathbf{w}_1, \mathbf{y}_1 \rangle + \cdots + \langle \mathbf{w}_n, \mathbf{y}_n \rangle$. In particular, an adversary can easily recover each $[\langle \mathbf{x}_i, \mathbf{y}_i \rangle]$, whereas the ideal functionality should only leak the sum $\sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i \rangle$. In the *public-key* setting, it is easy to see that $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$ is in fact inherent leakage from the ideal functionality. Concretely, an adversary can always pad an encryption of $\mathbf{x}_i$ in the i'th slot with encryptions of $\mathbf{0}$'s in the remaining $n - 1$ slots and then decrypt.

Our main scheme. The bulk of this work lies in constructing a multi-input FE for inner product in the *private-key* setting, where we can no longer afford to leak $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$. We modify the previous scheme by introducing additional rerandomization into each slot with the use of bilinear groups as follows:

$$\begin{aligned} \mathsf{msk} &:= \{[\mathbf{w}_i]_2, [v_i]_2, [z_i]_T\}_{i \in [n]}, \\ \mathsf{ek}_i &:= ([\mathbf{w}_i]_1, [v_i]_1, [z_i]_1), \\ \mathsf{ct}_{\mathbf{x}_i} &:= ([s_i]_1, [\mathbf{x}_i + \mathbf{w}_i s_i]_1, [z_i + v_i s_i]_1), \\ \mathsf{sk}_{\mathbf{y}_1, \ldots, \mathbf{y}_n} &:= ([\langle \mathbf{w}_1, \mathbf{y}_1 \rangle + v_1 r]_2, \ldots, [\langle \mathbf{w}_n, \mathbf{y}_n \rangle + v_n r]_2, [r]_2, [(z_1 + \cdots + z_n) r]_T). \end{aligned}$$

The ciphertext $\mathsf{ct}_{\mathbf{x}_i}$ can be viewed as encrypting $\mathbf{x}_i \| z_i$ using the single-input FE, where $z_1, \ldots, z_n$ are part of msk. In addition, we provide a single-input FE key for $\mathbf{y}_i \| r$ in the secret key, where a fresh r is sampled for each key. Decryption proceeds as follows: first compute

$$[\langle \mathbf{x}_i, \mathbf{y}_i \rangle + z_i r]_T = e([\mathbf{x}_i + \mathbf{w}_i s_i]_1^\top, [\mathbf{y}_i]_2) + e([z_i + v_i s_i]_1^\top, [r]_2) - e([s_i], [\langle \mathbf{w}_i, \mathbf{y}_i \rangle + v_i r]_2)$$

and then

$$[\sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i \rangle]_T = -[(z_1 + \cdots + z_n) r]_T + \sum_{i=1}^n [\langle \mathbf{x}_i, \mathbf{y}_i \rangle + z_i r]_T.$$

The intuition underlying security is that by the DDH assumption $[z_i r]_T$ is pseudorandom and helps mask the leakage about $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$ in $[\langle \mathbf{x}_i, \mathbf{y}_i \rangle + z_i r]_T$; in particular,

$$[\langle \mathbf{x}_1, \mathbf{y}_1 \rangle + z_1 r]_T, \ldots, [\langle \mathbf{x}_n, \mathbf{y}_n \rangle + z_n r]_T, [(z_1 + \cdots + z_n) r]_T$$

constitutes a computational secret-sharing of $[\langle \mathbf{x}_1, \mathbf{y}_1 \rangle + \cdots + \langle \mathbf{x}_n, \mathbf{y}_n \rangle]_T$, even upon reusing $z_1, \ldots, z_n$ as long as we pick a fresh r. In addition, sharing the same exponent r across n elements in the secret key helps prevent mix-and-match attacks across secret keys.

Our main technical result is that a variant of the private-key MIFE scheme we just described satisfies adaptive indistinguishability-based security under the k-Lin assumption in bilinear groups; a straight-forward extension of an impossibility result in [BSW11, AGVW13] rules out simulation-based security. Our final scheme, described in Figure 4.6, remains quite simple and achieves good concrete efficiency. We focus on selective security in this overview, and explain at the end the additional ideas needed to achieve adaptive security.

Overview of the security proof. There are two main challenges in the security proof: (i) avoiding leakage beyond the ideal functionality, (ii) avoiding super-polynomial hardness assumptions. Our proof proceeds in two steps: first, we establish security with a single challenge ciphertext per slot, and from which we bootstrap to achieve security with multiple challenge ciphertexts per slot. We will address the first challenge in the first step and the second challenge in the second. For notation simplicity, we focus on the setting with $n = 2$ slots and a single key query $\mathbf{y}_1 \| \mathbf{y}_2$.

Step 1. To prove indistinguishability-based security, we want to switch encryptions $\mathbf{x}_1^0, \mathbf{x}_2^0$ to encryptions of $\mathbf{x}_1^1, \mathbf{x}_2^1$. Here, the leakage from the ideal functionality imposes the restriction that

$$\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle + \langle \mathbf{x}_2^0, \mathbf{y}_2 \rangle = \langle \mathbf{x}_1^1, \mathbf{y}_1 \rangle + \langle \mathbf{x}_2^1, \mathbf{y}_2 \rangle$$

and this is the only restriction we can work with. The natural proof strategy is to introduce an intermediate hybrid that generates encryptions of $\mathbf{x}_1^1, \mathbf{x}_2^0$. However, to move from encryptions $\mathbf{x}_1^0, \mathbf{x}_2^0$ to this hybrid, we would require that $\langle \mathbf{x}_1^0 \| \mathbf{x}_2^0, \mathbf{y}_1 \| \mathbf{y}_2 \rangle = \langle \mathbf{x}_1^1 \| \mathbf{x}_2^0, \mathbf{y}_1 \| \mathbf{y}_2 \rangle$, which implies the extraneous restriction $\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle = \langle \mathbf{x}_1^1, \mathbf{y}_1 \rangle$. (Indeed, the single-input inner-product scheme in [BJK15] imposes extraneous restrictions to overcome similar difficulties in the function-hiding setting.)

To overcome this challenge, we rely on a single-input FE that achieves simulation-based security, which allows us to avoid the intermediate hybrid. See Theorem 9 and Remark 11 for further details.

Step 2. Next, we consider the more general setting with Q_1 challenge ciphertexts in the first slot and Q_2 in the second, but still a single key query. We achieve security loss $O(Q_1 + Q_2)$ for two slots, and more generally, $O(Q_1 + \cdots + Q_n)$ —as opposed to $Q_1 Q_2 \cdots Q_n$ corresponding to all possible combinations of the challenge ciphertexts— for n slots.

Our first observation is that we can bound the leakage from the ideal functionality by $O(Q_1 + Q_2)$ relations (the trivial bound being $Q_1 \cdot Q_2$). Denote the j'th ciphertext query in the i'th slot by $\mathbf{x}_i^{j,b}$, where b is the challenge bit. By decrypting the encryptions of $\mathbf{x}_1^{2,b}, \mathbf{x}_2^{1,b}$ and $\mathbf{x}_1^{1,b}, \mathbf{x}_2^{1,b}$ and substracting the two, the adversary learns $\langle \mathbf{x}_1^{2,b} - \mathbf{x}_1^{1,b}, \mathbf{y}_1 \rangle$ and more generally, $\langle \mathbf{x}_i^{j,b} - \mathbf{x}_i^{1,b}, \mathbf{y}_i \rangle$. Indeed, these are essentially the only constraints we need to work with, namely:

$$\begin{aligned} \langle \mathbf{x}_1^{1,0}, \mathbf{y}_1 \rangle + \langle \mathbf{x}_2^{1,0}, \mathbf{y}_2 \rangle &= \langle \mathbf{x}_1^{1,1}, \mathbf{y}_1 \rangle + \langle \mathbf{x}_2^{1,1}, \mathbf{y}_2 \rangle, \\ \langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle &= \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle, j = 2, \ldots, Q_i, i = 1, 2. \end{aligned}$$

Next, we need to translate the bound on the constraints to a $O(Q_1 + Q_2)$ bound on the security loss in the security reduction. We will switch from encryptions of $\mathbf{x}_i^{j,0}$ to those of $\mathbf{x}_i^{j,1}$ as follows: we write

$$\mathbf{x}_i^{j,0} = \mathbf{x}_i^{1,0} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}).$$

We can switch the first terms in the sums from $\mathbf{x}_i^{1,0}$ to $\mathbf{x}_i^{1,1}$ using security for a single challenge ciphertext, and then switch $\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}$ to $\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}$ by relying on security of the underlying single-input FE and the fact that $\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$. Here, we will require that the underlying single-input FE satisfies a malleability property, namely given Δ, we can maul an encryption of $\mathbf{x}$ into that of $\mathbf{x} + \Delta$. Note that this does not violate security because given $\langle \mathbf{x}, \mathbf{y} \rangle, \mathbf{y}, \Delta$, we can efficiently compute $\langle \mathbf{x} + \Delta, \mathbf{y} \rangle$. See Theorem 10 for further details.

Extension to adaptive security. The previous argument for selective security requires to embed the challenge into the setup parameters. To circumvent this issue, we use a two-step strategy for the adaptive security proof of MIFE. The first step uses an adaptive argument (this is essentially the argument used for the selective case, but applied to parameters that are picked at setup time), while the second step uses a selective argument, with *perfect security*. Thus, we can afford to use to simply guess the challenge beforehand, which incurs an exponential security loss, since the exponential term is multiplied by a zero term. The idea of using complexity leveraging to deduce adaptive security from selective security when the security is perfect, also appears in [Wee14, Remark 1]. See Remark 12 for further details.

Security against corruption of input slots. Proving the stronger security notion requires solving technical challenges that did not arise in [AGRW17]. In particular, to obtain full fledged many-AD-IND security, [AGRW17] use a generic transformation that uses an extra layer of symmetric encryption, to encrypt the original ciphertext. The symmetric key is shared across input slots, and the i'th share is given as part of any ciphertext for input slot $i \in [n]$. Thus, when ciphertexts are known for all slots $i \in [n]$, the decryption recovers all shares of the symmetric key, and decrypt the outer layer, to get the original ciphertext. The rest of decryption is performed as in the original multi-input FE.

The problem with this approach is that the encryption algorithm needs to know the symmetric key (and not simply a share of it). Thus, corrupting one input slot allows the adversary to recover the entire symmetric key, and break the security of the scheme. Such problem did not arise in [AGRW17], which does not consider corruptions of input slots. To circumvent this issue, as in [DOT18], we use the symmetric key to encrypt the functional secret keys, instead of encrypting the ciphertexts. Each encryption key ek_i for input slot $i \in [n]$ contains the i'th share of the symmetric key, but the full symmetric key is only needed by the key generation algorithm, which knows msk. If one share is missing, all the functional secret keys are random. Security of the overall multi-input FE when zero functional secret keys are queried concludes the security proof. See Section 2.4.2 for further details.

Theoretical perspective. The focus of this work is on obtaining constructions for a specific class of functions with good concrete efficiency. Nonetheless, we believe that our results do shed some new insights into general feasibility results for MIFE. Namely, we presented the first MIFE for a non-trivial functionality that polynomial security loss for a super-constant number of slots under falsifiable assumptions. Recall that indistinguishability obfuscation and generic multilinear maps are not falsifiable, whereas the constructions based on single-input FE in [AJ15, BV15, BKS16] incur a security loss which is exponential in the number of slots. Indeed, there is a reason why prior works relied on non-falsifiable assumptions or super-polynomial security loss. Suppose an adversary makes Q_0 key queries, and $Q_1, \ldots, Q_n$ ciphertext queries for the n slots. By combining the ciphertexts and keys in different ways, the adversary can learn $Q_0 Q_1 \cdots Q_n$ different decryptions. When n is super-constant, the winning condition in the security game may not be efficiently checkable in polynomial-time, hence the need for either

[1] The security notion achieved in [KLM+18] is actually a weaker variant of many-AD-IND in which the adversary is only allowed to perform a single key query at the beginning of the security game.

a non-falsifiable assumption or a super-polynomial security loss. To overcome this difficulty, we show that for inner products, we can exploit linearity to succinctly characterize the $Q_0 Q_1 \cdots Q_n$ constraints by roughly $Q_0 \cdot (Q_1 + \cdots Q_n)$ constraints.

Discussion. Our constructions and techniques may seem a-priori largely tailored to the inner product functionality and properties of bilinear groups. We clarify here that our high-level approach (which builds upon [Wee14, BKP14]) may be applicable beyond inner products, namely:

i. start with a multi-input FE that is only secure for a single ciphertext per slot and one secret key, building upon a single-input FE whose security is simulation-based for a single ciphertext (in our case, this corresponds to introducing the additional $z_1, \ldots, z_n$ to hide the intermediate computation $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$);

ii. achieve security for a single ciphertext per slot and multiple secret keys, by injecting additional randomness to the secret keys to prevent mix-and-match attacks (for this, we replaced $z_1, \ldots, z_n$ with $z_1 r, \ldots, z_n r, r$ in the exponent);

iii. "bootstrap" to multiple ciphertexts per slot, where we also showed how to avoid incurring an exponential security loss.

In particular, using simulation-based security for i. helped us avoid additional leakage beyond what is allowed by the ideal functionality.

Additional related work. Goldwasser et al. [GGG+14] showed that both two-input public-key MIFE as well as n-input private-key MIFE for circuits already implies indistinguishability obfuscation for circuits.

There have also been several works that proposed constructions for private-key multi-input functional encryption. The work of Boneh et al. [BLR+15] constructs a single-key MIFE in the private key setting, which is based on multilinear maps and is proven secure in the idealized generic multilinear map model. Two other papers explore the question how to construct multi-input functional encryption starting from the single input variant. In their work [AJ15] Ananth and Jain demonstrate how to obtain selectively secure MIFE in the private key setting starting from any general-purpose public key functional encryption. In an independent work, Brakerski et al. [BKS16] reduce the construction of private key MIFE to general-purpose private key (single input) functional encryption. The resulting scheme achieves selective security when the starting private key FE is selectively secure. Additionally in the case when the MIFE takes any constant number of inputs, adaptive security for the private key FE suffices to obtain adaptive security for the MIFE construction as well. The constructions in that work provide also function hiding properties for the MIFE encryption scheme.

While this line of work reduces MIFE to single-input FE for general-purpose constructions, the only known instantiations of construction for public and private key functional encryption with unbounded number of keys require either indistinguishability obfuscation [GGH+13b] or multilinear maps with non-standard assumptions [GGHZ16]. We stress that the transformations from single-input to MIFE in [AJ15, BKS16] are not applicable in the case of inner products since these transformations require that the single-input FE for complex functionalities related to computing a PRF, which is not captured by the simple inner functionality.

Road-map. In the rest of this chapter, we first present the selectively-secure MIFE in Section 4.1, then show in Section 4.2 how to obtain adaptive security.

Selectively-Secure, Private-Key MIFE for Inner Products

In this section, we present a private-key MIFE for bounded-norm inner products over $\mathbb{Z}$, that is, for the set of functionalities $\{F_n^{m,X,Y}\}_{n\in\mathbb{N}}$ defined as $F_n^{m,X,Y} : \mathcal{K}_n \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Z}$, with $\mathcal{K}_n := [0, Y]^{mn}$, for all $i \in [n]$, $\mathcal{X}_i := [0, X]^m$, $\mathcal{Z} := \mathbb{Z}$, such that for any $(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n) \in \mathcal{K}_n$, $\mathbf{x}_i \in \mathcal{X}_i$, we have:

$$F_n^{m,X,Y}\big((\mathbf{y}_1 \| \cdots \| \mathbf{y}_n), \mathbf{x}_1, \ldots, \mathbf{x}_n\big) = \sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle.$$

Remark 7: on leakage

Let $(\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1})_{i\in[n],j\in[Q_i]}$ be the ciphertext queries, and $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$ be a secret key query. For all slots $i \in [n]$, all $j \in [Q_i]$, and all bits $b \in \{0, 1\}$, the adversary can learn

$$\langle \mathbf{x}_i^{j,b} - \mathbf{x}_i^{1,b}, \mathbf{y}_i \rangle$$

via the ideal functionality. In the IND security game, this means the adversary is restricted to queries satisfying

$$\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle.$$

In the hybrid, we want to avoid additional constraints such as

$$\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle.$$

We prove many-SEL security, for static corruptions (see Definition 23), using an asymmetric pairing group $\mathcal{PG} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e)$ with $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ of prime order p, where p is a 2λ-bit prime. Our construction relies on the Matrix Decisional Diffie-Hellman assumption in $\mathbb{G}_1$ and in $\mathbb{G}_2$ (see Definition 10), and build upon any single-input FE for inner products, that satisfies one-SEL-SIM security, along with some additional structural properties. Such single-input FE can be obtained by straightforwardly adapting the scheme from [ALS16, Section 3], and is recalled in Section 2.6.1 for completeness. For correctness, we require $n; m; X; Y$ to be polynomials in the security parameter. This implies that:

$$n \cdot m \cdot X \cdot Y \ll p.$$

Our generic single-to-multi input construction is described in Figure 4.1. We present a self-contained description of the scheme in Figure 4.6.

Selectively-secure, multi-input scheme from single-input scheme

Main construction. We present in Figure 4.1 a private key multi-input FE, $\mathcal{MIFE}$, for the bounded-norm inner products over $\mathbb{Z}$, starting from any one-SEL-SIM secure, single-input inner products FE, $\mathcal{FE}$, that additionally satisfies the following requirements.

Additional requirements. The construction and the analysis requires that $\mathcal{FE} := (\mathsf{GSetup}', \mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ satisfies the following structural properties:

- The scheme can be instantiated over $\mathbb{G}_1$, where the ciphertext is a vector $[\mathbf{c}]_1$ over $\mathbb{G}_1$ and the secret key is a vector $\mathbf{d}_i$ over $\mathbb{Z}_p$.
- Enc' is linearly homomorphic. More specifically, we only that, given gpk', $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}', \mathbf{x})$, and $\mathbf{x}'$, we can generate a fresh random encryption of $\mathbf{x} + \mathbf{x}'$, i.e. $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}', \mathbf{x} + \mathbf{x}')$. This property is used in the proof of Lemma 31 and Lemma 32.

- For correctness, Dec' should be linear in its input $\mathbf{d}$ and $[\mathbf{c}]_1$, so that $\mathsf{Dec}'(\mathsf{gpk}', [\mathbf{d}]_2, [\mathbf{c}]_1) = [\mathsf{Dec}'(\mathsf{gpk}', \mathbf{d}, \mathbf{c})]_T \in \mathbb{G}_T$ can be computed using a pairing.
- For an efficient MIFE decryption, Dec' must work without any restriction on the norm of the output as long as the output is in the exponent.
- Let $(\widetilde{\mathsf{GSetup}}, \widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}})$ be the simulator for the one-SEL-SIM security of $\mathcal{FE}$. We require that $\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, \cdot, \cdot)$ is linear in its inputs $(\mathbf{y}, a)$, so that we can compute $\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, [\mathbf{y}]_2, [a]_2) = [\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, \mathbf{y}, a)]_2$. This property is used in the proof of Lemma 29.

$\underline{\mathsf{Setup}(1^\lambda, F_n^{m,X,Y})}$:
$\mathsf{gpk}' \leftarrow \mathsf{GSetup}'(1^\lambda, F_{\mathsf{IP}}^{m+k,X,Y})$, where gpk' contains $\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, p, P_1, P_2, e) \leftarrow \mathsf{PGGen}(1^\lambda)$
For all $i \in [n]$: $(\mathsf{ek}'_i, \mathsf{msk}'_i) \leftarrow \mathsf{Setup}'(1^\lambda, \mathsf{gpk}', F_{\mathsf{IP}}^{m+k,X,Y})$, $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathsf{ek}_i := (\mathsf{ek}'_i, \mathbf{z}_i)$
$\mathsf{pk} := \mathsf{gpk}', \mathsf{msk} := \{\mathsf{msk}'_i, \mathbf{z}_i\}_{i\in[n]}$
Return $(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i)}$:
Return $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{x}_i \| \mathbf{z}_i)$

$\underline{\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $z := \langle \mathbf{z}_1 + \cdots + \mathbf{z}_n, \mathbf{r} \rangle$
For all $i \in [n]$: $\mathbf{d}_i \leftarrow \mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}'_i, \mathbf{y}_i \| \mathbf{r})$
$\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := \big((\mathbf{y}_1 \| \cdots \| \mathbf{y}_n), \{[\mathbf{d}_i]_2\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T\big)$
Return $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

$\underline{\mathsf{Dec}\big(\mathsf{pk}, \mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n\big)}$:
Parse $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} = \big((\mathbf{y}_1 \| \cdots \| \mathbf{y}_n), \{[\mathbf{d}_i]_2\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T\big)$
For all $i \in [n]$: $[a_i]_T \leftarrow \mathsf{Dec}'(\mathsf{gpk}', [\mathbf{d}_i]_2, \mathsf{ct}_i)$
Return the discrete log of $(\sum_{i=1}^n [a_i]_T) - [z]_T$

Figure 4.1: Multi-input functional encryption scheme $\mathcal{MIFE}$ for the bounded norm inner-product over $\mathbb{Z}$. $\mathcal{FE} := (\mathsf{GSetup}', \mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ refers to a single-input inner-product FE.

Correctness. By correctness of $\mathcal{FE}$, we have for all $i \in [n]$: $[a_i]_T = [\langle \mathbf{x}_i \| \mathbf{z}_i, \mathbf{y}_i \| \mathbf{r} \rangle]_T$. Thus, decryption computes:

$$\left[(\sum_{i=1}^{n} \langle \mathbf{x}_i \| \mathbf{z}_i, \mathbf{y}_i \| \mathbf{r} \rangle) - \langle \mathbf{z}_1 + \cdots + \mathbf{z}_n, \mathbf{r} \rangle \right]_T = [\langle \mathbf{x}_1 \| \cdots \| \mathbf{x}_n, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n \rangle]_T$$

We know $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \leq n \cdot m \cdot X \cdot Y$, which is bounded by a polynomial in the security parameter. Thus, decryption can efficiently recover the discrete log: $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \bmod p = \sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle$, where the equality holds since $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \leq n \cdot m \cdot X \cdot Y \ll p$.

Remark 8: Optimization

A more efficient version of our scheme would be to take $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ subject to $\sum_i \mathbf{z}_i = \mathbf{0}$. This way, we don't have to include the value $[z]_T$ in the secret keys, since it would cancel out. We choose to present the inefficient version which includes the value $[z]_T$ for simplicity.

Remark 9: Notations

We use subscripts and superscripts for indexing over multiple copies, and never for indexing over positions or exponentiation. Concretely, the j'th ciphertext query in slot i is $\mathbf{x}_i^j$.

Security. First, we prove the one-SEL-IND-static security of $\mathcal{MIFE}$, in Theorem 9, that is, in English: the scheme is secure for only one challenge ciphertext per input slot, in the selective setting, for static corruptions (see Definition 23). Then, in Theorem 10, we show how to upgrade the security of the $\mathcal{MIFE}$ to many-SEL-IND-static, that is, for many challenge ciphertexts.

Theorem 9: one-SEL-IND-static security of $\mathcal{MIFE}$

Suppose $\mathcal{FE}$ is one-SEL-SIM secure for n instances, and that the $\mathcal{U}_k(p)$-MDDH assumption holds in $\mathbb{G}_2$. Then, $\mathcal{MIFE}$ is one-SEL-IND-static secure.

Recall that the $\mathcal{U}_k(p)$-MDDH assumption is the weakest of all $\mathcal{D}_k(p)$-MDDH assumptions, for any matrix distribution $\mathcal{D}_k(p)$, according to Lemma 3. In particular, it is implied by the well-known k-Lin assumption.

game	ct_i:	$\{\mathbf{d}_i\}_{i\in[n]}$ in $\mathsf{sk}_\mathbf{y}$:	z in $\mathsf{sk}_\mathbf{y}$:	justification/remark
$\mathsf{G}_{0,\beta}$	$\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{x}_i^\beta \Vert \mathbf{z}_i)$	$\mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}_i', \mathbf{y}_i \Vert \mathbf{r})$	$z = \langle \mathbf{z}_1 + \ldots + \mathbf{z}_n, \mathbf{r} \rangle$	one-SEL-IND-static security game
$\mathsf{G}_{1,\beta}$	$\widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}}_i)$	$\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}_i, \mathbf{y}_i \Vert \mathbf{r}, \langle \mathbf{x}_i^\beta \Vert \mathbf{z}_i, \mathbf{y}_i \Vert \mathbf{r} \rangle)$	$z = \langle \mathbf{z}_1 + \ldots + \mathbf{z}_n, \mathbf{r} \rangle$	one-SEL-SIM security of $\mathcal{FE}$
$\mathsf{G}_{2,\beta}$	$\widetilde{\mathsf{Enc}}(\widetilde{\mathsf{msk}}_i)$	$\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}_i, \mathbf{y}_i \Vert \mathbf{r}, \langle \mathbf{x}_i^\beta, \mathbf{y}_i \rangle + \widetilde{z}_i)$	$z = \sum_{i\in\mathcal{CS}} \langle \mathbf{z}_i, \mathbf{r} \rangle + \sum_{i\in\mathcal{HS}} \widetilde{z}_i$	$\mathcal{D}_k$-MDDH

Figure 4.2: Sequence of games for the proof of Theorem 9. Here, for any slot $i \in [n]$, ct_i refers to the challenge ciphertext computed by oracle $\mathsf{OEnc}(i, (\mathbf{x}_i^0, \mathbf{x}_i^1))$, $\mathbf{d}_i$ and z refers to the vectors computed by the oracle $\mathsf{OKeygen}(\mathbf{y}_1 \Vert \cdots \Vert \mathbf{y}_n)$ as part of $\mathsf{dk}_{\mathbf{y}_1\Vert\cdots\Vert\mathbf{y}_n}$, and $(\widetilde{\mathsf{GSetup}}, \widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}})$ is the simulator for the one-SEL-SIM security of $\mathcal{FE}$.

Proof of Theorem 9. Using Theorem 2, it is sufficient to prove one-SEL-IND-zero-static (i.e. the scheme is secure when no decryption keys are queried), and one-SEL-IND-weak-static i.e. we assume the adversary requests a challenge ciphertext for all slots $i \in \mathcal{HS}$, where $\mathcal{HS} := [n] \setminus \mathcal{CS}$ denotes the set of slots that are not corrupted) to obtain one-SEL-IND-static security.

The one-SEL-IND-zero-static security of $\mathcal{MIFE}$ follows directly from the one-SEL-IND security of $\mathcal{FE}$ (which is implied by its one-SEL-SIM security). In what follows, we prove one-SEL-IND-weak-static security of $\mathcal{MIFE}$.

We proceed via a series of games $\mathsf{G}_{i,\beta}$ for $i \in \{0, \ldots, 2\}$, $\beta \in \{0, 1\}$, described in Figure 4.3. The transitions are summarized in Figure 4.2. Let $\mathcal{A}$ be a PPT adversary. For any game G, we denote by $\mathsf{Adv}_\mathsf{G}(\mathcal{A})$ the probability that the game G outputs 1 when interacting with $\mathcal{A}$. Note that the set of input slots for which a challenge ciphertext is queried, denoted by I in Figure 4.3, is such that $\mathcal{HS} \subseteq I$, since we want to prove one-SEL-IND-*weak* security.

Games $\mathsf{G}_{0,\beta}$, for $\beta \in \{0, 1\}$: are such that $\mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND\text{-}weak\text{-}static}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_{0,0}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{0,1}}(\mathcal{A})|$, according to Definition 21.

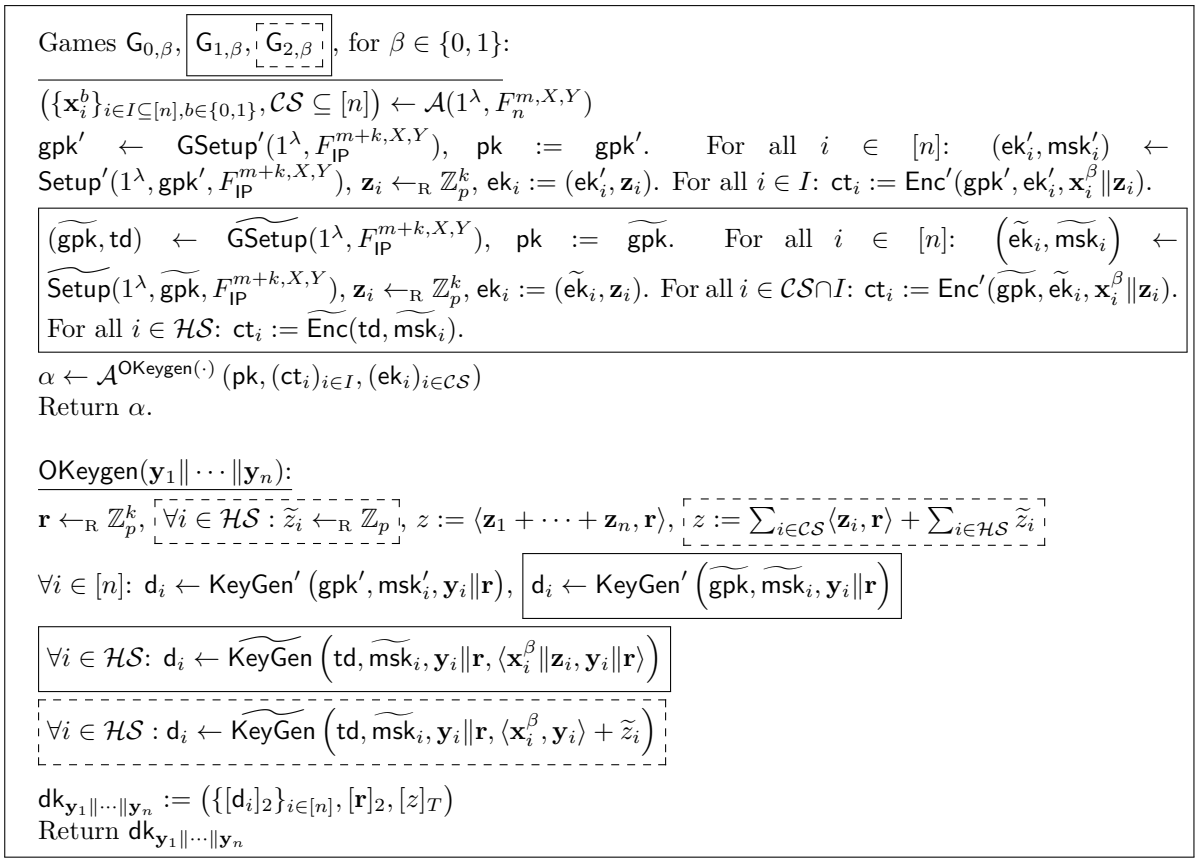


Figure 4.3: Games for the proof of Theorem 9. In each procedure, the components inside a solid (dotted) frame are only present in the games marked by a solid (dotted) frame. Here, $\mathcal{CS}$ denotes the set of corrupted slots, $\mathcal{HS} := [n] \setminus \mathcal{CS}$ denotes the set of honest slots, and $I \subseteq [n]$ denotes the set of input slots for which there is a challenge ciphertext. We have $\mathcal{HS} \subseteq I$.

Games $\mathsf{G}_{1,\beta}$, for $\beta \in \{0,1\}$: we replace $(\mathsf{GSetup}', \mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Enc}')$ by the simulator $(\widetilde{\mathsf{GSetup}}, \widetilde{\mathsf{Setup}}, \widetilde{\mathsf{KeyGen}}, \widetilde{\mathsf{Enc}})$, using the one-SEL-SIM security of $\mathcal{FE}$ for h instances, where h denotes the size of $\mathcal{HS}$, where $\mathcal{HS}$ is the set of honest input slots, that is, $\mathcal{HS} := [n] \setminus \mathcal{CS}$. We prove in Lemma 28 that there exists a PPT adversary $\mathcal{B}_1$ such that

$$|\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}SIM}}_{\mathcal{FE},\mathcal{B}_1,h}(\lambda).$$

Games $\mathsf{G}_{2,\beta}$, for $\beta \in \{0,1\}$: we replace the values $\langle \mathbf{z}_i, \mathbf{r} \rangle$ used by the oracle $\mathsf{OKeygen}$ to $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, for all slots $i \in \mathcal{HS}$, using the $\mathcal{U}_k(p)$-MDDH assumption in $\mathbb{G}_2$. Namely, we prove in Lemma 29 that there exists a PPT adversary $\mathcal{B}_2$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A})| \leq \mathsf{Adv}^{\mathcal{U}_k(p)\text{-}\mathrm{MDDH}}_{\mathbb{G}_2,\mathcal{B}_2}(\lambda) + \frac{1}{p-1}.$$

Finally, in Lemma 30, we prove that $\mathsf{G}_{2,0}$ and $\mathsf{G}_{2,1}$ are perfectly indistinguishable, using a statistical argument that crucially relies on the fact that we are in the selective security setting, and using the restrictions on the queries to $\mathsf{OKeygen}$ and the challenge $\{\mathbf{x}_i^b\}_{i \in I \subseteq [n], b \in \{0,1\}}$ imposed by the security game. We have:

$$\mathsf{Adv}_{\mathsf{G}_{2,0}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{2,1}}(\mathcal{A}).$$

Putting everything together, we obtain:

$$\mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND\text{-}weak\text{-}static}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}SIM}}_{\mathcal{FE},\mathcal{B}_0,h}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathcal{U}_k\text{-}\mathrm{MDDH}}_{\mathbb{G}_2,\mathcal{B}_2}(\lambda) + \frac{2}{p-1},$$

where $h \leq n$ is the number of honest input slots. □

Lemma 28: Game $\mathsf{G}_{0,\beta}$ to $\mathsf{G}_{1,\beta}$

There exists a PPT adversary $\mathcal{B}_{1,\beta}$ such that

$$\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}SIM}}_{\mathcal{FE},\mathcal{B}_{1,\beta},h}(\lambda),$$

where h denotes the size of $\mathcal{HS}$, where $\mathcal{HS}$ is the set of honest input slots, that is, $\mathcal{HS} := [n] \setminus \mathcal{CS}$.

Proof of Lemma 28. In the game $\mathsf{G}_{1,\beta}$, we replace $(\mathsf{GSetup}', \mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}')$ by the simulator $(\widetilde{\mathsf{GSetup}}, \widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{KeyGen}})$, whose existence is ensured by the one-SEL-SIM security of $\mathcal{FE}$ (see Definition 20). A complete description of games $\mathsf{G}_{0,\beta}$ and $\mathsf{G}_{1,\beta}$ is given in Figure 4.3.

The adversary $\mathcal{B}_{0,\beta}$ proceeds as follows.

-Simulation of $(\mathsf{pk}, \{\mathsf{ct}_i\}_{i \in I}, \{\mathsf{ek}_i\}_{i \in \mathcal{CS}})$:

Upon receiving the challenge $\{\mathbf{x}_i^b\}_{i \in I, b \in \{0,1\}}$, and the set of corrupted user $\mathcal{CS} \subseteq [n]$ from $\mathcal{A}$, adversary $\mathcal{B}_{0,\beta}$ samples $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ for all $i \in [n]$, and sends $\{(\mathbf{x}_i^\beta \| \mathbf{z}_i)\}_{i \in \mathcal{HS}}$ to the experiment it is interacting with, upon which it receives the global public key gpk and ciphertexts $\{\mathsf{ct}_i\}_{i \in \mathcal{HS}}$. The global public key gpk is either of the form $\mathsf{gpk} = \mathsf{gpk}'$ with $\mathsf{gpk}' \leftarrow \mathsf{GSetup}'(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})$ if $\mathcal{B}_{0,\beta}$ is interacting with the experiment $\mathbf{REAL}^{\mathcal{FE}}(1^\lambda, \mathcal{B}_{0,\beta,\ell})$, and $\mathsf{gpk} = \widetilde{\mathsf{gpk}}$ with $(\widetilde{\mathsf{gpk}}, \mathsf{td}) \leftarrow \widetilde{\mathsf{GSetup}}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})$ if $\mathcal{B}_{0,\beta}$ is interacting with the experiment $\mathbf{IDEAL}^{\mathcal{FE}}(1^\lambda, \mathcal{B}_{0,\beta,\ell})$ (see Definition 20 for a description of these experiments, with the one-SEL restriction). The ciphertexts are of the form $\mathsf{ct}_i := \mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{x}_i^\beta \| \mathbf{z}_i)$ or $\widetilde{\mathsf{Enc}}(\mathsf{td}, \widetilde{\mathsf{ek}}_i, \widetilde{\mathsf{msk}}_i)$, depending on which experiment $\mathcal{B}_{0,\beta}$ is interacting with.

For all $i \in \mathcal{CS}$, $\mathcal{B}_{0,\beta}$ samples $(\mathsf{ek}_i, \mathsf{msk}_i) \leftarrow \mathsf{Setup}'(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})$. For all $\mathcal{CS} \cap I$, it computes $\mathsf{ct}_i := \mathsf{Enc}'(\mathsf{gpk}, \mathsf{ek}_i, \mathbf{x}_i^\beta \| \mathbf{z}_i)$. It sets $\mathsf{pk} := \mathsf{gpk}$, and returns $(\mathsf{pk}, \{\mathsf{ct}_i\}_{i \in I}, \{\mathsf{ek}_i\}_{i \in \mathcal{CS}})$ to $\mathcal{A}$.

-Simulation of $\mathsf{OKeygen}(\mathbf{y}_1\| \dots \|\mathbf{y}_n)$:

For any query $(\mathbf{y}_1\| \dots \|\mathbf{y}_n)$, $\mathcal{B}_{0,\beta,\ell}$ picks $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. Then, for all $i \in \mathcal{CS}$, it computes $\mathbf{d}_i \leftarrow \mathsf{KeyGen}'(\mathsf{gpk}, \mathsf{msk}_i, \mathbf{y}_i\|\mathbf{r})$. It can do so since it knows gpk and msk_i for all $i \in \mathcal{CS}$. For all $i \in \mathcal{HS}$, $\mathcal{B}_{0,\beta}$ queries its own decryption key oracle on $\mathbf{y}_i\|\mathbf{r}$, to obtain $\mathbf{d}_i := \mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}'_i, \mathbf{y}_i\|\mathbf{r})$ if it is interacting with the real experiment, or $\mathbf{d}_i := \widetilde{\mathsf{KeyGen}}(\mathsf{td}, \widetilde{\mathsf{msk}}_i, \mathbf{y}_i\|\mathbf{r}, \langle \mathbf{x}_i^{\beta}\|\mathbf{z}_i, \mathbf{y}_i\|\mathbf{r}\rangle)$ if it is interacting with the ideal experiment.

Then, it computes $z := \langle \mathbf{z}_1 + \dots + \mathbf{z}_n, \mathbf{r}\rangle$ and returns $\mathsf{dk}_{\mathbf{y}_1\|\dots\|\mathbf{y}_n} := \left(\{[\mathbf{d}_i]_2\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T\right)$ to $\mathcal{A}$.

Finally, $\mathcal{B}_{0,\beta}$ forwards $\mathcal{A}$'s output α to its own experiment. It is clear that when $\mathcal{B}_{0,\beta}$ interacts with the experiment $\mathbf{REAL}^{\mathcal{FE}}(1^\lambda, \mathcal{B}_{0,\beta})$, it simulates the game $\mathsf{G}_{0,\beta}$, whereas it simulates the game $\mathsf{G}_{,\beta}$ when it interacts with $\mathbf{IDEAL}^{\mathcal{FE}}(1^\lambda, \mathcal{B}_{0,\beta})$. Therefore,

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}SIM}}_{\mathcal{FE},\mathcal{B}_{0,\beta}}(\lambda)\\
&= \left|\Pr\left[\mathbf{REAL}^{\mathcal{FE}}(1^\lambda, \mathcal{B}_{0,\beta}) = 1\right] - \Pr\left[\mathbf{IDEAL}^{\mathcal{FE}}(1^\lambda, \mathcal{B}_{0,\beta}) = 1\right]\right|\\
&= |\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A})|
\end{aligned}$$

□

Lemma 29: Game $\mathsf{G}_{1,\beta}$ to $\mathsf{G}_{2,\beta}$

There exists a PPT adversary $\mathcal{B}_{2,\beta}$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathcal{U}_k\text{-MDDH}}_{\mathbb{G}_2,\mathcal{B}_{2,\beta}}(\lambda) + \frac{1}{p-1}.$$

Recall, from Lemma 3, that for any matrix distribution $\mathcal{D}_k(p)$, we have $\mathcal{D}_k(p)$-MDDH $\Rightarrow \mathcal{U}_k(p)$-MDDH.

Proof of Lemma 29. Here, we switch $\{[\mathbf{r}]_2, [\langle \mathbf{z}_i, \mathbf{r}\rangle]_2\}_{i\in\mathcal{HS}}$ used by the oracle $\mathsf{OKeygen}$ to $\{[\mathbf{r}]_2, [\widetilde{z}_i]_2\}_{i\in\mathcal{HS}}$, where $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, and $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. Recall that $\mathcal{HS}$ denotes the set of honest slots, that is $\mathcal{HS} := [n] \setminus \mathcal{CS}$.

This is justified by the fact that $\{[\mathbf{r}]_2, [\langle \mathbf{z}_i, \mathbf{r}\rangle]_2\}_{i\in\mathcal{HS}} \in \mathbb{G}_2^{(k+h)}$, where h is the size of $\mathcal{HS}$, is identically distributed to $[\mathbf{U}\mathbf{r}]_2$ where $\mathbf{U} \leftarrow_{\mathrm{R}} \mathcal{U}_{k+h,k}(p)$ (wlog. we assume that the upper k rows of $\mathbf{U}$ are full rank), which is indistinguishable from a uniformly random vector over $\mathbb{G}_2^{k+h}$, that is, of the form: $\{[\mathbf{r}]_2, [\widetilde{z}_i]_2\}_{i\in\mathcal{HS}}$, according to the $\mathcal{U}_{k+h,k}(p)$-MDDH assumption. To do the switch simultaneously for all calls to $\mathsf{OKeygen}$, that is, to switch $\{[\mathbf{r}^j]_2, [\langle \mathbf{z}_i, \mathbf{r}^j\rangle]_2\}_{i\in\mathcal{HS}, j\in[Q_0]}$ to $\{[\mathbf{r}^j]_2, [\widetilde{z}_i^j]_2\}_{i\in\mathcal{HS}, j\in[Q_0]}$, where $\widetilde{z}_i^j \leftarrow \mathbb{Z}_p$ and $\mathbf{r}^j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ for all $j \in [Q_0]$, where Q_0 denotes the number of calls to $\mathsf{OKeygen}$, we use the Q_0-fold $\mathcal{U}_{k+h,k}(p)$-MDDH assumption. Namely, we build a PPT adversary $\mathcal{B}'_{2,\beta}$ such that

$$\mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A}) \le \mathsf{Adv}^{Q_0\text{-}\mathcal{U}_{k+h,k}(p)\text{-MDDH}}_{\mathbb{G}_2,\mathcal{B}'_{2,\beta}}(\lambda).$$

This, together with Corollary 1 ($\mathcal{U}_k(p)$-MDDH $\Rightarrow$ Q_0-fold $\mathcal{U}_{k+h,k}(p)$-MDDH), implies the lemma. The adversary $\mathcal{B}'_{2,\beta}$ proceeds as follows.

-Simulation of $(\mathsf{pk}, \{\mathsf{ct}_i\}_{i\in I}, \{\mathsf{ek}_i\}_{i\in\mathcal{CS}})$**:**

Upon receiving an Q_0-fold $\mathcal{U}_{k+h,k}$-MDDH challenge

$$\left(\mathcal{PG}, [\mathbf{U}]_2 \in \mathbb{G}_2^{(k+h)\times k}, \left[\mathbf{h}^1 \| \cdots \| \mathbf{h}^{Q_0}\right]_2 \in \mathbb{G}_2^{(k+h)\times Q_0}\right),$$

together with the challenge $\{\mathbf{x}_i^b\}_{i\in I, b\in\{0,1\}}$ and the set $\mathcal{CS} \subseteq [n]$ from $\mathcal{A}$, $\mathcal{B}'_{2,\beta}$ samples $(\widetilde{\mathsf{gpk}}, \mathsf{td}) \leftarrow \widetilde{\mathsf{GSetup}}(1^\lambda, F_{\mathsf{IP}}^{m+k,X,Y})$. For all $i \in [n]$, it samples $(\widetilde{\mathsf{ek}}_i, \widetilde{\mathsf{msk}}_i) \leftarrow \widetilde{\mathsf{Setup}}(1^\lambda, \widetilde{\mathsf{gpk}}, F_{\mathsf{IP}}^{m+k,X,Y})$, $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and sets $\mathsf{ek}_i := (\widetilde{\mathsf{ek}}_i, \mathbf{z}_i)$. For all $i \in \mathcal{HS}$, it samples $\mathsf{ct}_i := \widetilde{\mathsf{Enc}}(\mathsf{td}, \widetilde{\mathsf{ek}}_i, \widetilde{\mathsf{msk}}_i)$. For all $i \in \mathcal{CS} \cap I$, it samples $\mathsf{ct}_i := \mathsf{Enc}'(\widetilde{\mathsf{gpk}}, \widetilde{\mathsf{ek}}_i, \mathbf{x}_i^\beta \| \mathbf{z}_i)$. It sets $\mathsf{pk} := \widetilde{\mathsf{gpk}}$, and returns $(\mathsf{pk}, \{\mathsf{ct}_i\}_{i\in I}, \{\mathsf{ek}_i\}_{i\in\mathcal{CS}})$ to $\mathcal{A}$.

-Simulation of $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$**:**

On the j'th query $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$ of $\mathcal{A}$, $\mathcal{B}'_{2,\beta}$ sets $[\mathbf{r}^j]_2 := [\overline{\mathbf{h}^j}]_2$, where $\overline{\mathbf{h}^j} \in \mathbb{Z}_p^k$ denotes the k-upper components of $\mathbf{h}^j \in \mathbb{Z}_p^{k+n}$. For all $i \in \mathcal{CS}$, it computes $\mathbf{d}_i := \mathsf{KeyGen}'(\widetilde{\mathsf{gpk}}, \widetilde{\mathsf{msk}}_i, \mathbf{y}_i \| \mathbf{r}^j)$. For all $i \in \mathcal{HS}$, it computes $[\mathbf{d}_i] := \widetilde{\mathsf{KeyGen}}\left(\mathsf{td}, \widetilde{\mathsf{msk}}_i, [\mathbf{y}_i \| \mathbf{r}^j]_2, [\langle \mathbf{x}_i^\beta, \mathbf{y}_i\rangle + \mathbf{h}_{k+i}^j]_2\right)$, where $\mathbf{h}_{k+i}^j$ denotes the $k+i$'th coordinate of the vector $\mathbf{h}^j \in \mathbb{Z}_p^{k+h}$. Here we rely on the fact that $\widetilde{\mathsf{KeyGen}}(\mathsf{td}, \widetilde{\mathsf{msk}}, \cdot, \cdot)$ is linear in its inputs $(\mathbf{y}, a)$, so that $\mathcal{B}'_{2,\beta}$ can compute $\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, [\mathbf{y}]_2, [a]_2) = [\widetilde{\mathsf{KeyGen}}(\widetilde{\mathsf{msk}}, \mathbf{y}, a)]_2$. Note that when $\left[\mathbf{h}^1 \| \cdots \| \mathbf{h}^{Q_0}\right]_2$ is a real MDDH challenge, $\mathcal{B}'_{2,\beta}$ simulate game $\mathsf{G}_{1,\beta}$, whereas it simulates game $\mathsf{G}_{2,\beta}$ when $\left[\mathbf{h}^1 \| \cdots \| \mathbf{h}^{Q_0}\right]_2$ is uniformly random over $\mathbb{G}_2^{(k+n)\times Q_0}$. □

Lemma 30: Game $\mathsf{G}_{2,0}$ to $\mathsf{G}_{2,1}$

$$\mathsf{Adv}_{\mathsf{G}_{2,0}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{2,1}}(\mathcal{A}).$$

Proof of Lemma 30. We show that $\mathsf{G}_{2,\beta}$ does not depend on β, using the fact that for all $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n \in (\mathbb{Z}_p^m)^n$, for all $\{\mathbf{x}_i^b \in \mathbb{Z}_p^m\}_{i\in[n], b\in\{0,1\}}$, the following are identically distributed:

$$\{\widetilde{z}_i\}_{i\in\mathcal{HS}} \quad \text{and} \quad \{\widetilde{z}_i - \langle \mathbf{x}_i^\beta, \mathbf{y}_i\rangle\}_{i\in\mathcal{HS}},$$

where $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ for all $i \in \mathcal{HS}$.

For each query $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$, $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ picks values $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ for $i \in \mathcal{HS}$ that are *independent* of $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$ and the challenge $\{\mathbf{x}_i^b \in \mathbb{Z}_p^m\}_{i \in [n], b \in \{0,1\}}$ (note that here we crucially rely on the fact the games $\mathsf{G}_{2,0}$ and $\mathsf{G}_{2,1}$ are *selective*), therefore, using the previous fact, we can switch $\widetilde{z}_i$ to $\widetilde{z}_i - \langle \mathbf{x}_i^\beta, \mathbf{y}_i \rangle$ for all $i \in \mathcal{HS}$, without changing the distribution of the game. This way, for all $i \in \mathcal{HS}$, $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ computes $\mathbf{d}_i \leftarrow \widetilde{\mathsf{KeyGen}}(\mathsf{td}, \widetilde{\mathsf{msk}}_i, \mathbf{y}_i \| \mathbf{r}, \widetilde{z}_i)$, which does not depend on β, and

$$z := \sum_{i \in \mathcal{CS}} \langle \mathbf{z}_i, \mathbf{r} \rangle + \sum_{i \in \mathcal{HS}} \widetilde{z}_i - \sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^\beta, \mathbf{y}_i \rangle .$$

By definition of the security game, we have $\mathbf{x}_i^0 = \mathbf{x}_i^1$ for all $i \in \mathcal{CS} \cap I$. Thus, we have:

$$z := \sum_{i \in \mathcal{CS}} \langle \mathbf{z}_i, \mathbf{r} \rangle + \sum_{i \in \mathcal{HS}} \widetilde{z}_i - \sum_{i \in I} \langle \mathbf{x}_i^\beta, \mathbf{y}_i \rangle .$$

Finally, by definition of the security game, we have: $\sum_{i \in I} \langle \mathbf{x}_i^0, \mathbf{y}_i \rangle = \sum_{i \in I} \langle \mathbf{x}_i^1, \mathbf{y}_i \rangle$. This is implied by **Condition 1** in Definition 23, and the fact that $\mathcal{HS} \subseteq I$. That means the value $[z]_T$ computed by $\mathsf{OKeygen}$ does not depend on β. Finally, for all $i \in \mathcal{CS}$, $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ computes $\mathbf{d}_i := \mathsf{KeyGen}'(\widetilde{\mathsf{gpk}}, \widetilde{\mathsf{msk}}_i, \mathbf{y}_i \| \mathbf{r})$, which does not depend on β. Putting everything together, we get that $\mathsf{G}_{2,\beta}$ is independent of β. □

Remark 10: decryption capabilities

As a sanity check, we note that the simulated secret keys will correctly decrypt a simulated ciphertext. However, unlike schemes proven secure via the standard dual system encryption methodology [Wat09], a simulated secret key will incorrectly decrypt a normal ciphertext. This is not a problem because we are in the private-key setting, so a distinguisher will not be able to generate normal ciphertexts by itself.

Remark 11: why a naive argument is inadequate

We cannot afford to do a naive hybrid argument across the n slots for the challenge ciphertext as it would introduce extraneous restrictions on the adversary's queries. Concretely, suppose we want to use a hybrid argument to switch from encryptions of $\mathbf{x}_1^0, \mathbf{x}_2^0$ in game 0 to those of $\mathbf{x}_1^1, \mathbf{x}_2^1$ in game 2 with an intermediate hybrid that uses encryptions of $\mathbf{x}_1^1, \mathbf{x}_2^0$ in Game_1. To move from game 0 to game 1, the adversary's query $\mathbf{y}_1 \| \mathbf{y}_2$ must satisfy $\langle \mathbf{x}_1^0 \| \mathbf{x}_2^0, \mathbf{y}_1 \| \mathbf{y}_2 \rangle = \langle \mathbf{x}_1^1 \| \mathbf{x}_2^0, \mathbf{y}_1 \| \mathbf{y}_2 \rangle$, which implies the extraneous restriction $\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle = \langle \mathbf{x}_1^1, \mathbf{y}_1 \rangle$.

As described in the proof above, we overcome the limitation by using simulation-based security. Note that what essentially happens in the first slot in our proof is as follows (for $k = 1$, that is, DDH): we switch from $\mathsf{Enc}'(\mathsf{pk}'_1, \mathbf{x}_1^0 \| z_1)$ to $\mathsf{Enc}'(\mathsf{pk}'_1, \mathbf{x}_1^1 \| z_1)$ while giving out a secret key which contains $\mathsf{KeyGen}'(\mathsf{msk}'_1, \mathbf{y}_1 \| r^1)$ and $[r^1]_2$. Observe that

$$\langle \mathbf{x}_1^0 \| z_1, \mathbf{y}_1 \| r^1 \rangle = \langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle + z_1 r^1, \quad \langle \mathbf{x}_1^1 \| z_1, \mathbf{y}_1 \| r^1 \rangle = \langle \mathbf{x}_1^1, \mathbf{y}_1 \rangle + z_1 r^1$$

may not be equal, since we want to avoid the extraneous restriction $\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle = \langle \mathbf{x}_1^1, \mathbf{y}_1 \rangle$. This means that one-SEL-IND security does not provide any guarantee that the ciphertexts are indistinguishable. However, one-SEL-SIM security does provide such a guarantee, because

$$([\langle \mathbf{x}_1^0, \mathbf{y}_1 \rangle + z_1 r^1]_2, [r^1]_2) \approx_c ([\langle \mathbf{x}_1^1, \mathbf{y}_1 \rangle + z_1 r^1]_2, [r^1]_2)$$

via the DDH assumption in $\mathbb{G}_2$. Since the outcomes of the decryption are computationally

indistinguishable, the output of the simulated ciphertext would also be computationally indistinguishable.

Theorem 10: many-yy-IND-static security of $\mathcal{MIFE}$

Let yy $\in$ {AD,SEL}. Suppose $\mathcal{FE}$ is many-yy-IND secure and $\mathcal{MIFE}$ is one-yy-IND-static secure. Then, $\mathcal{MIFE}$ is many-yy-IND-static secure.

Since the construction $\mathcal{MIFE}$ from Figure 4.1 is proven one-SEL-IND-static secure in Theorem 9, we obtain the following corollary.

Corollary 2: many-SEL-IND-static security of $\mathcal{MIFE}$

The scheme $\mathcal{MIFE}$ from Figure 4.1 is many-SEL-IND secure, assuming the underlying $\mathcal{FE}$ is many-SEL-IND secure.

That is, we show that our multi-input FE is selectively secure in the setting with multiple challenge ciphertexts (and since our multi-input FE is a private key scheme, this is not immediately implied by the one-SEL-IND security).

Proof overview.

- We first switch encryptions of $\mathbf{x}_1^{1,0}, \ldots, \mathbf{x}_n^{1,0}$ to those of $\mathbf{x}_1^{1,1}, \ldots, \mathbf{x}_n^{1,1}$, and for the remaining ciphertexts, we switch from an encryption of $\mathbf{x}_i^{j,0} = (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,0}$ to that of $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$. This uses the one-yy-IND security of $\mathcal{MIFE}$, and the fact that its encryption algorithm is linearly homomorphic, thanks to which encryption of $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,\beta}$ can be publicly computed from an encryption of $\mathbf{x}_i^{1,\beta}$.
- Then, we switch from encryptions of
$$(\mathbf{x}_i^{2,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_i,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$$
to those of
$$(\mathbf{x}_i^{2,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_i,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}.$$
This uses the many-yy-IND security of $\mathcal{FE}$.

As described earlier, to carry out the latter argument, the queries must satisfy the constraint

$$\langle (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle = \langle (\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle \iff \langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$$

where the latter is already imposed by the ideal functionality.

Proof of Theorem 10. We proceed via a series of games, described in Figure 4.5. The transitions are summarized in Figure 4.4. Let $\mathcal{A}$ be a PPT adversary. For any game G, we denote by $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ the probability that the game G outputs 1 when interacting with $\mathcal{A}$.

Game G_0: is such that $\mathsf{Adv}^{\mathsf{many\text{-}SEL\text{-}IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})|$, according to Definition 21.

Game G_1: is as game G_0, except we replace the challenge ciphertexts to $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})$ for all $i \in [n]$ and $j \in [Q_i]$, using the one-yy-IND security of $\mathcal{MIFE}$. Namely, we prove in Lemma 31 that there exists a PPT adversary $\mathcal{B}_1$ such that

$$\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{one\text{-}yy\text{-}IND}}_{\mathcal{MIFE},\mathcal{B}_1}(\lambda).$$

game	ct_i^j:	justification/remark
0	$\mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,0})$	many-yy-IND security game
1	$\mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})$	one-yy-IND security of $\mathcal{MIFE}$
2	$\mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_i^{j,1})$	many-yy-IND security for n instance of $\mathcal{FE}$

Figure 4.4: Sequence of games for the proof of Theorem 10. Here, for any slot $i \in [n]$, and $j \in [Q_i]$, ct_i^j refers to the j'th challenge ciphertext for slot $i \in [n]$. Changes are highlighted in gray for better visibility.

Games G_0, $\boxed{\mathsf{G}_1}$, $\boxed{\mathsf{G}_2}$:
$\mathcal{CS} \subseteq [n] \leftarrow \mathcal{A}(1^\lambda, F_n^{m,X,Y})$
$(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, F_n^{m,X,Y})$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot),\mathsf{OKeygen}(\cdot)}(\mathsf{pk}, \{\mathsf{ek}_i\}_{i\in\mathcal{CS}})$
Return α.

$\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}))$:
$\mathsf{ct}_i^j := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,0})$
$\boxed{\mathsf{ct}_i^j := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})}$
$\boxed{\mathsf{ct}_i^j := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1})}$
Return ct_i^j.

$\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:
Return $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$.

Figure 4.5: Games for the proof of Theorem 10. In the selective variants of these games, the adversary sends its challenges $\{\mathbf{x}_i^{j,b}\}_{i\in[n],j\in[Q_i],b\in\{0,1\}}$ before seeing the public key and querying any decryption keys.

Game G_2: we replace the challenge ciphertexts to $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1})$ for all $i \in [n]$ and $j \in [Q_i]$, using the many-yy-IND security of $\mathcal{FE}$ for n instances, which is implied by the single-instance security (see Lemma 5). We prove in Lemma 32 that there exists a PPT adversary $\mathcal{B}_2$ such that

$$\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_2(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{many\text{-}yy\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda).$$

Putting everything together, we obtain:

$$\mathsf{Adv}^{\mathsf{many\text{-}yy\text{-}IND}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{one\text{-}yy\text{-}IND}}_{\mathcal{MIFE},\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{many\text{-}yy\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda).$$

□

Lemma 31: Game G_0 to G_1

There exists a PPT adversary $\mathcal{B}_1$ such that

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{one\text{-}yy\text{-}IND}}_{\mathcal{MIFE},\mathcal{B}_1}(\lambda).$$

Proof of Lemma 31. In game G_1, which is described in Figure 4.5, we replace $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,0} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}))$ with $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}))$ for all $i \in [n], j \in [Q_i]$. This is justified by the following properties:

- one-yy-IND security of $\mathcal{MIFE}$;
- the fact Enc' is linearly homomorphic. Namely, for all $i \in [n]$, given $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{x}_i^{1,\beta})$, $\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}$ and gpk', we can create a fresh encryption $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{x}_i^{1,\beta} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0})$ (corresponding to challenge ciphertexts in slots i in game G_β).

The adversary $\mathcal{B}_1$ proceeds as follows.

-Simulation of pk:

In the adaptive variant, i.e. yy = AD, $\mathcal{B}$ receives the set $\mathcal{CS} \subseteq [n]$ from $\mathcal{A}$, sends it to its own experiment, receives a public key which it forwards to $\mathcal{A}$.

In the selective variant, i.e. yy = SEL, it receives the challenge $\{\mathbf{x}_i^{j,b}\}_{i\in[n],j\in[Q_i],b\in\{0,1\}}$, and the set $\mathcal{CS} \subseteq [n]$ from $\mathcal{A}$. It sends the pair of vectors $\{\mathbf{x}_i^{1,b}\}_{i\in I,b\in\{0,1\}}$ as its selective challenge to its experiment, where $I \subseteq [n]$ is the set of indices $i \in [n]$ for which $Q_i > 0$. It gets back pk, which it forwards to $\mathcal{A}$, and the challenge ciphertexts $\{\mathsf{ct}_i\}_{i\in I}$, where $\mathsf{ct}_i = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_1^{1,\beta})$, for $\beta \in \{0, 1\}$, when $\mathcal{B}_1$ is interacting with the experiment $\mathsf{SEL\text{-}IND}_\beta^{\mathcal{MIFE}}(1^\lambda, \mathcal{B}_1)$, which is the selective variant of $\mathsf{AD\text{-}IND}_\beta^{\mathcal{MIFE}}(1^\lambda, \mathcal{B}_1)$ from Definition 23.

-Simulation of $\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}))$:

In the adaptive variant, if $j = 1$, that is, it is the first query for slot $i \in [n]$, then $\mathcal{B}_1$ queries its own oracle to get $\mathsf{ct}_i := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}^{1,\beta})$, where $\beta \in \{0, 1\}$, depending on the experiment $\mathcal{B}_1$ is interacting with. If $j > 1$, $\mathcal{B}_1$ uses the fact that the single-input inner-product scheme is linearly homomorphic to generate all the remaining ciphertexts ct_i^j for $i \in I, j \in \{2, \ldots, Q_i\}$ by combining $\mathsf{ct}_i = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,\beta}) = \mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{x}_i^{1,\beta} \| \mathbf{z}_i)$ with the vector $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} \| \mathbf{0})$ to obtain $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{x}_i^{1,\beta} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} \| \mathbf{z}_i) = \mathsf{Enc}(\mathsf{ek}_i, \mathbf{x}_i^{1,\beta} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0})$ which matches the challenge ciphertexts in Game G_β. Note that this can be done using gpk'. $\mathcal{B}_1$ returns $\{\mathsf{ct}_i^j\}_{i\in[n],j\in[Q_i]}$ to $\mathcal{A}$.

In the selective variant, the same thing happens, except queries to OEnc are performed beforehand.

-Simulation of $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:

$\mathcal{B}_1$ simply uses its own secret key generation oracle on input $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$ and forwards the answer to $\mathcal{A}$.

Finally, $\mathcal{B}_1$ forwards the output α of $\mathcal{A}$ to its own experiment. It is clear that for all $\beta \in \{0, 1\}$, when $\mathcal{B}_1$ interacts with $\mathbf{one\text{-}SEL\text{-}IND}_\beta^{\mathcal{MIFE}}$, it simulates the game G_β to $\mathcal{A}$. Therefore,

$$\mathsf{Adv}^{\mathsf{one\text{-}yy\text{-}IND}}_{\mathcal{MIFE},\mathcal{B}_1}(\lambda) = \left|\Pr\left[\mathbf{one\text{-}yy\text{-}IND}_0^{\mathcal{MIFE}}(1^\lambda, \mathcal{B}_1) = 1\right] - \Pr\left[\mathbf{one\text{-}yy\text{-}IND}_1^{\mathcal{MIFE}}(1^\lambda, \mathcal{B}_1) = 1\right]\right| = |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})|.$$

□

Lemma 32: Game G_1 to G_2

There exists a PPT adversary $\mathcal{B}_2$ such that

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{many\text{-}yy\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda).$$

Proof of Lemma 32. In Game G_2, which is described in Figure 4.5, we replace $\mathsf{Enc}(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) \| \mathbf{z}_i)$ with $\mathsf{Enc}(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}) \| \mathbf{z}_i) = \mathsf{Enc}(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{x}_i^{j,1} \| \mathbf{z}_i)$, for all $i \in [n], j \in [Q_i]$. This follows from the many-yy-IND security of $\mathcal{FE}$ for n instances, which we can use since for each key query $\mathbf{y}_1 \| \dots \| \mathbf{y}_n$ and all $\mathbf{r}$, $\mathbf{z}$, we have

$$\langle \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} \| \mathbf{z}, \mathbf{y}_i \| \mathbf{r} \rangle = \langle \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} \| \mathbf{z}, \mathbf{y}_i \| \mathbf{r} \rangle.$$

The latter is equivalent to $\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$, which follows from the restriction imposed by the security game (see Remark 7).

We build a PPT adversary $\mathcal{B}_2$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{many\text{-}yy\text{-}IND}}_{\mathcal{FE},\mathcal{B}_1,n}(\lambda).$$

Adversary $\mathcal{B}_2$ proceeds as follows.

First, $\mathcal{B}_2$ samples $\mathbf{z}_i \leftarrow \mathbb{Z}_p^k$ for all $i \in [n]$. Then, it simulates all challenge ciphertexts ct_i^j using its own encryption oracle on input $(i, (\mathbf{x}_i^{j,0} \| \mathbf{z}_i, \mathbf{x}_i^{j,1} \| \mathbf{z}_i))$. It simulates all decryption keys $\mathsf{dk}_{\mathbf{y}_1 \| \cdots \| \mathbf{y}_n}$ by first sampling $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and setting $\mathbf{d}_i$ as the output of its own decryption key oracle on input $(i, \mathbf{y}_i \| \mathbf{r})$. It returns $\mathsf{dk}_{\mathbf{y}_1 \| \cdots \| \mathbf{y}_n} := (\{[\mathbf{d}_i]_2\}_{i \in [n]}, [\mathbf{r}]_2, [\sum_i \langle \mathbf{z}_i, \mathbf{r} \rangle]_T)$.

Finally, $\mathcal{B}_2$ forwards the outputs α of $\mathcal{A}$ to its own experiment. It is clear that for all $\beta \in \{0,1\}$, when $\mathcal{B}_2$ interacts with $\mathbf{many\text{-}yy\text{-}IND}_\beta^{\mathcal{MIFE}}$, it simulates the game $\mathsf{G}_{1+\beta}$ to $\mathcal{A}$. Therefore,

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{many\text{-}yy\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda) = \\
&\left| \Pr\left[\mathbf{many\text{-}yy\text{-}IND}_0^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{B}_2) = 1\right] - \Pr\left[\mathbf{many\text{-}yy\text{-}IND}_1^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{B}_2) = 1\right] \right| = \\
&|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})|.
\end{aligned}$$

□

Putting everything together

In Figure 4.6 we spell out the details of the scheme in the previous section with a concrete instantiation of the underlying single-input inner-product scheme, whose one-SEL-SIM security is proven under the $\mathcal{D}_k$-MDDH assumption, which is provided for completeness in Section 2.6.1.

$\underline{\mathsf{Setup}(1^\lambda, F_n^{m,X,Y})}$:
$\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, p, P_1, P_2, e) \leftarrow_{\mathrm{R}} \mathsf{PGGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$. For all $i \in [n]$: $\mathbf{W}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, $\mathbf{V}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times(k+1)}$, $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathsf{ek}_i := (\mathbf{z}_i, [\mathbf{W}_i\mathbf{A}]_1, [\mathbf{V}_i\mathbf{A}]_1)$, $\mathsf{pk} := (\mathcal{PG}, [\mathbf{A}]_1)$, $\mathsf{msk} := \{\mathbf{W}_i, \mathbf{V}_i, \mathbf{z}_i\}_{i\in[n]}$. Return $(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]})$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i)}$:
$\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, return $\begin{bmatrix} -\mathbf{A}\mathbf{s}_i \\ \mathbf{x}_i + \mathbf{W}_i\mathbf{A}\mathbf{s}_i \\ \mathbf{z}_i + \mathbf{V}_i\mathbf{A}\mathbf{s}_i \end{bmatrix}_1$.

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. For all $i \in [n]$: $\mathbf{d}_i := \begin{pmatrix} \mathbf{W}_i^\top \mathbf{y}_i + \mathbf{V}_i\mathbf{r} \\ \binom{\mathbf{y}_i}{\mathbf{r}} \end{pmatrix}$, $z := \langle \mathbf{z}_1 + \cdots + \mathbf{z}_n, \mathbf{r} \rangle$
Return $\big(\{[\mathbf{d}_i]_2\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T\big)$

$\underline{\mathsf{Dec}\Big(\big(\{[\mathbf{d}_i]_2\}_{i\in[n]}, [\mathbf{r}]_2, [z]_T\big), \{[\mathbf{c}_i]_1\}_{i\in[n]}\Big)}$:
$[d]_T := \sum_i e([\mathbf{c}_i]_1^\top, [\mathbf{d}_i]_2) - [z]_T$
Return the discrete log of $[d]_T$.

Figure 4.6: Our private-key MIFE scheme for the functionality $F_n^{m,X,Y}$, which is proven many-SEL-IND-static in Corollary 2, and many-AD-IND secure in Theorem 51. Both rely on the $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$.

Achieving Adaptive Security

In this section, we prove that MIFE from Figure 4.6 is many-AD-IND-static under $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$. That is, our scheme is secure with many challenge ciphertexts, chosen adaptively by the adversary, and handles static corruptions of input slots (see Definition 23).

Security. The security proof proceeds in two steps, similarly than the many-SEL-IND security proof in Section 4.1. First, we show in Theorem 11 that the MIFE in Figure 4.6 is one-AD-IND-static secure, that is, it is adaptively secure when there is only a single challenge ciphertext, and handles static corruption of input slots.

Then, using Theorem 10 (many-yy-IND security of $\mathcal{FE}$ & one-yy-IND security of $\mathcal{MIFE} \Rightarrow$ many-yy-IND of $\mathcal{MIFE}$) together with Theorem 11 (one-AD-IND security of $\mathcal{MIFE}$) and the many-AD-IND security of the underlying $\mathcal{FE}$ (proven in Theorem 4), we obtain many-AD-IND security of $\mathcal{MIFE}$ (Corollary 3)

Theorem 11: one-AD-IND-static security of $\mathcal{MIFE}$

Suppose the $\mathcal{D}_k(p)$-MDDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$. Then, the multi-input FE in Figure 4.6 is one-AD-IND-static secure.

That is, we show that our multi-input FE is adaptively secure when there is only a single challenge ciphertext.

Corollary 3: many-AD-IND-static security of $\mathcal{MIFE}$

Suppose the $\mathcal{D}_k(p)$-MDDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$. Then, the multi-input FE in Figure 4.6 is many-AD-IND-static secure.

Game	$\mathbf{c}_i$	$\mathbf{c}'_i$	$\mathbf{c}''_i$	$\mathbf{d}_i$:	z:	justification/remark	reference
$\mathsf{G}_{0,\beta}$	$\mathbf{A}\mathbf{s}_i$	$\mathbf{W}_i\mathbf{c}_i + \mathbf{x}_i^\beta$	$\mathbf{V}_i\mathbf{c}_i + \mathbf{z}_i$	$\mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r}$	$\langle \mathbf{z}_1 + \ldots + \mathbf{z}_n, \mathbf{r}\rangle$	one-AD-IND-static security game	Definition 23
$\mathsf{G}_{1,\beta}$	$\mathbf{A}\mathbf{s}_i + \mathbf{u}$	$\mathbf{W}_i\mathbf{c}_i + \mathbf{x}_i^\beta$	$\mathbf{V}_i\mathbf{c}_i + \mathbf{z}_i$	$\mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r}$	$\langle \mathbf{z}_1 + \ldots + \mathbf{z}_n, \mathbf{r}\rangle$	$\mathcal{D}_k$-MDDH in $\mathbb{G}_1$	Lemma 33
$\mathsf{G}_{2,\beta}$	$\mathbf{A}\mathbf{s}_i + \mathbf{u}$	$\mathbf{W}_i\mathbf{c}_i + \mathbf{x}_i^\beta$	$\mathbf{V}_i\mathbf{c}_i$	$\mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r} - \mathbf{a}^\perp\langle \mathbf{z}_i, \mathbf{r}\rangle$	$\langle \mathbf{z}_1 + \ldots + \mathbf{z}_n, \mathbf{r}\rangle$	inf. theoretic	Lemma 34
$\mathsf{G}_{3,\beta}$	$\mathbf{A}\mathbf{s}_i + \mathbf{u}$	$\mathbf{W}_i\mathbf{c}_i + \mathbf{x}_i^\beta$	$\mathbf{V}_i\mathbf{c}_i$	If $i \in \mathcal{HS}$: $\mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r} - \mathbf{a}^\perp\widetilde{z}_i$ If $i \in \mathcal{CS}$: $\mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r} - \mathbf{a}^\perp\langle \mathbf{z}_i, \mathbf{r}\rangle$	$\sum_{i\in\mathcal{CS}}\langle \mathbf{z}_i, \mathbf{r}\rangle + \sum_{i\in\mathcal{HS}}\widetilde{z}_i$	$\mathcal{D}_k$-MDDH in $\mathbb{G}_2$	Lemma 35
$\mathsf{G}^\star_{3,\beta}$	$\mathbf{A}\mathbf{s}_i + \mathbf{u}$	$\mathbf{W}_i\mathbf{c}_i + \mathbf{x}_i^\beta$	$\mathbf{V}_i\mathbf{c}_i$	If $i \in \mathcal{HS}$: $\mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r} - \mathbf{a}^\perp\widetilde{z}_i$ If $i \in \mathcal{CS}$: $\mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r} - \mathbf{a}^\perp\langle \mathbf{z}_i, \mathbf{r}\rangle$	$\sum_{i\in\mathcal{CS}}\langle \mathbf{z}_i, \mathbf{r}\rangle + \sum_{i\in\mathcal{HS}}\widetilde{z}_i$	selective variant	Lemma 36

Figure 4.7: Sequence of games for the proof of Theorem 11. Here, for any slot $i \in [n]$, $([-\mathbf{c}_i]_1, [\mathbf{c}'_i]_1, [\mathbf{c}''_i]_1)$ is the challenge ciphertext computed by $\mathsf{Enc}(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$; $[\mathbf{d}_i]_2$ and $[z]_T$ are part of the $\mathsf{sk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$ computed by $\mathsf{OKeygen}(\mathbf{y}_1\|\cdots\|\mathbf{y}_n)$. We use $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{k+1} \setminus \mathsf{Span}(\mathbf{A})$ and $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{k+1}$ such that $\mathbf{A}^\top\mathbf{a}^\perp = \mathbf{0}$ and $\mathbf{u}^\top\mathbf{a}^\perp = 1$. To analyze the games $\mathsf{G}_{3,\beta}$, we consider the selective variant of these games: $\mathsf{G}^\star_{3,\beta}$. We prove using an information-theoretic argument that $\mathsf{G}^\star_{3,0}$ and $\mathsf{G}^\star_{3,1}$ are identically distributed. Using a guessing argument, we prove the same holds for the adaptive games $\mathsf{G}_{3,0}$ and $\mathsf{G}_{3,1}$.

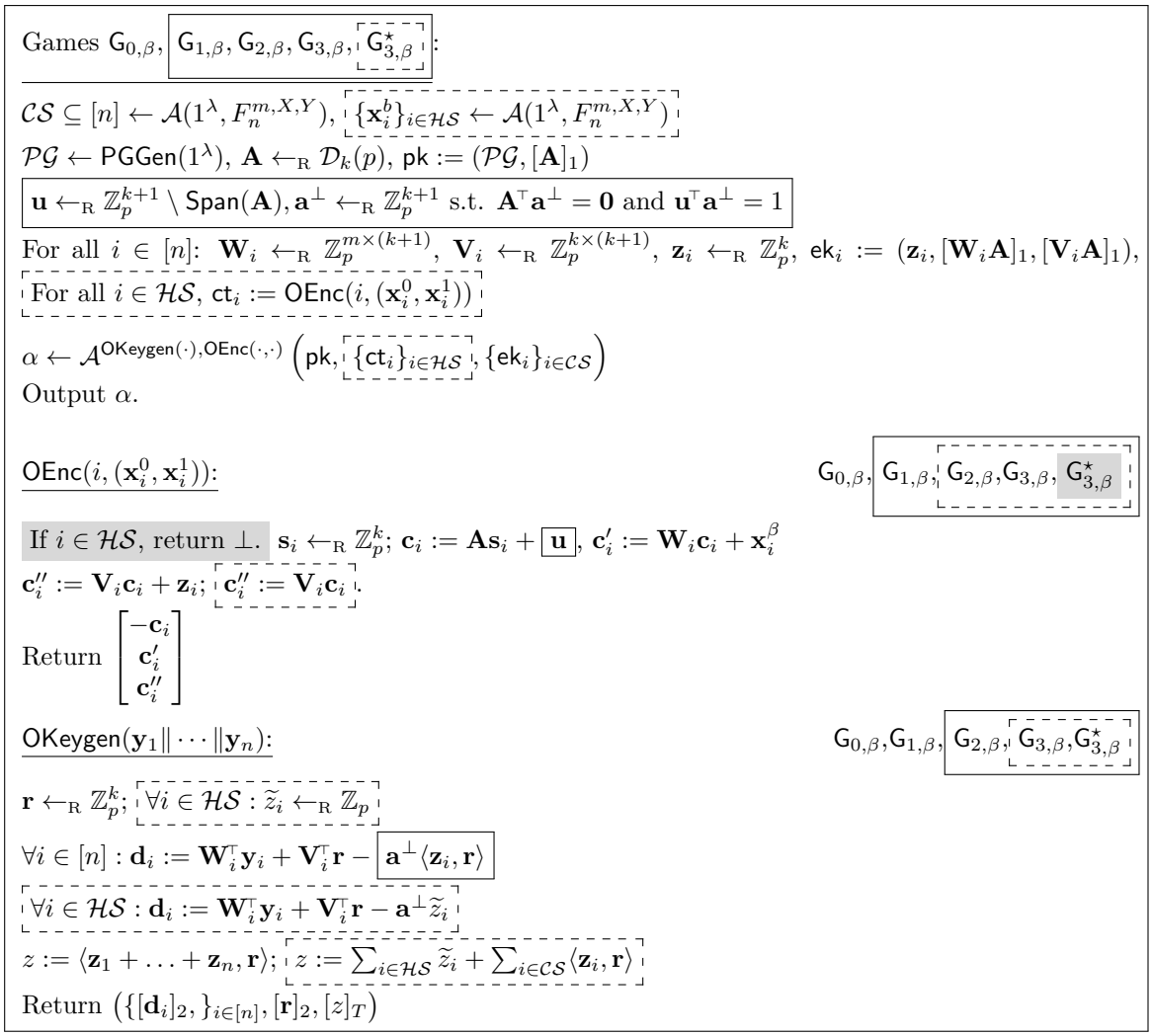


Figure 4.8: Games for the proof of Theorem 11. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame. Here, $\mathcal{CS}$ denotes the set of corrupted slots, and $\mathcal{HS} := [n] \setminus \mathcal{CS}$ is the set of honest input slots. The oracle OEnc can be queried at most once per input slot.

Proof of Theorem 11. Using Theorem 2, it is sufficient to prove one-AD-IND-zero-static (i.e. the scheme is secure when no decryption keys are queried), and one-AD-IND-weak-static i.e. we assume the adversary requests a challenge ciphertext for all slots $i \in \mathcal{HS}$, where $\mathcal{HS} := [n] \setminus \mathcal{CS}$ denotes the set of slots that are not corrupted) to obtain one-AD-IND-static security.

The one-AD-IND-zero-static security of $\mathcal{MIFE}$ follows directly from the one-AD-IND security for n instance of the underlying $\mathcal{FE}$ (recall that the construction from Figure 4.6 is simply the implementation of the generic construction from Figure 4.1, with the concrete $\mathcal{FE}$ from Section 2.6.1, which is one-AD-IND secure for n instance). In what follows, we prove one-AD-IND-weak-static security of $\mathcal{MIFE}$.

We proceed via a series of games described in Figure 4.8. The transitions are summarized in Figure 4.7. Let $\mathcal{A}$ be a PPT adversary, and $\lambda \in \mathbb{N}$ be the security parameter. For game G, we define $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ to be the probability that the game G outputs 1 when interacting with $\mathcal{A}$.

Games $\mathsf{G}_{0,\beta}$, for $\beta \in \{0,1\}$: are such that

$$\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}static}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_{0,0}} - \mathsf{Adv}_{\mathsf{G}_{0,1}}|,$$

according to Definition 23.

Games $\mathsf{G}_{1,\beta}$, for $\beta \in \{0,1\}$: we change the distribution of the vectors $[\mathbf{c}_i]_1$ computed by $\mathsf{OEnc}(i,\cdot,\cdot)$, for all queried $i \in [n]$, using the $\mathcal{D}_k(p)$-MDDH assumption. Namely, in Lemma 33, we prove that there exists a PPT adversary $\mathcal{B}_{1,\beta}$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_1,\mathcal{B}_{1,\beta}}(\lambda) + \frac{1}{p}.$$

Games $\mathsf{G}_{2,\beta}$, for $\beta \in \{0,1\}$: here, for all slots $i \in [n]$, we change the way the vectors $[\mathbf{c}''_i]_1$ and $[\mathbf{d}_i]_2$ are computed, respectively, by $\mathsf{OEnc}(i,\cdot,\cdot)$ and $\mathsf{OKeygen}$, using an information theoretic argument. The point is to make it possible to simulate the G_2 only knowing $[\mathbf{z}_i]_2$ (and not $[\mathbf{z}_i]_1$), which will be useful later, to use the $\mathcal{U}_k(p)$-MDDH assumption on $[\mathbf{z}_i]_2$, in $\mathbb{G}_2$. Namely, we show in Lemma 34 that

$$\mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A}).$$

Games $\mathsf{G}_{3,\beta}$, for $\beta \in \{0,1\}$: we use the $\mathcal{D}_k(p)$-MDDH assumption to switch simultaneously for all $i \in \mathcal{HS}$ the values $[\langle \mathbf{z}_i, \mathbf{r}\rangle]_2$ computed by $\mathsf{OKeygen}$, to uniformly random values over $\mathbb{G}_2$. Recall that $\mathcal{HS} \subseteq [n]$ denotes the set of honest (that is, non corrupted) input slots. This relies on the fact that it is not necessary to know $[\mathbf{z}_i]_1$ for $i \in \mathcal{HS}$ to simulate the games $\mathsf{G}_{2,\beta}$ or $\mathsf{G}_{3,\beta}$. Namely, in Lemma 35, we show that there exists a PPT adversary $\mathcal{B}_{3,\beta}$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathcal{U}_k\text{-MDDH}}_{\mathbb{G}_2,\mathcal{B}_{3,\beta}}(\lambda) + \frac{1}{p-1}.$$

At this point, we show that $\mathsf{Adv}_{\mathsf{G}_{3,0}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{3,1}}(\mathcal{A})$ in three steps. First, we consider the selective variant of game $\mathsf{G}_{3,\beta}$, called $\mathsf{G}^\star_{3,\beta}$, where the adversary must commit to its challenge $(\mathbf{x}^0_i, \mathbf{x}^1_i)_{i\in\mathcal{HS}}$ before receiving pk or making any decryption key queries, where $\mathcal{HS} \subseteq [n]$ denotes the set of input slots which are not corrupted. Further encryption queries can be made adaptively for slots $i \in \mathcal{CS}$. By a guessing argument, we show in Lemma 36 that there exists PPT adversary $\mathcal{A}^\star$ such that

$$\mathsf{Adv}_{\mathsf{G}_{3,\beta}}(\mathcal{A}) = (X+1)^{2hm} \cdot \mathsf{Adv}_{\mathsf{G}^\star_{3,\beta}}(\mathcal{A}^\star),$$

where h denotes the size of $\mathcal{HS}$. Then we prove in Lemma 37 that the game $\mathsf{G}^\star_{3,0}$ is identical to the game $\mathsf{G}^\star_{3,1}$ using a statistical argument, which is only true in the selective setting, and using the restrictions on the queries imposed by the security game. Namely, we show that for all adversaries $\mathcal{A}'$:

$$\mathsf{Adv}_{\mathsf{G}^\star_{3,0}}(\mathcal{A}') = \mathsf{Adv}_{\mathsf{G}^\star_{3,1}}(\mathcal{A}').$$

Putting everything together, we obtain:

$$\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}static}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_1,\mathcal{B}_1}(\lambda) + 2 \cdot \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G}_2,\mathcal{B}_2}(\lambda) + \frac{2}{p}.$$

Note that the $\mathcal{U}_k(p)$-MDDH is implied by $\mathcal{D}_k(p)$-MDDH for any matrix distribution $\mathcal{D}_k(p)$ according to Lemma 3. In particular, it is implied by the well-known k-Lin assumption.

□

Lemma 33: Game $\mathsf{G}_{0,\beta}$ to $\mathsf{G}_{1,\beta}$

There exists a PPT adversary $\mathcal{B}_{1,\beta}$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathcal{D}_k\text{-MDDH}}_{\mathbb{G}_1,\mathcal{B}_{1,\beta}}(\lambda) + \frac{1}{p}.$$

Proof of Lemma 33. Here, we switch $([\mathbf{A}]_1, [\mathbf{A}\mathbf{s}_i]_1)$ computed by $\mathsf{OEnc}(i,\cdot,\cdot)$ to $([\mathbf{A}]_1, [\mathbf{A}\mathbf{s}_i + \mathbf{u}]_1)$ simultaneously for all queried $i \in [n]$, where $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$, $\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$.

This change is justified by the facts that:

1. The distributions: $\{\mathbf{s}_i\}_{i\in[n]}$ and $\{\mathbf{s}_i + \mathbf{s}\}_{i\in[n]}$, where $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ and for all $i \in [n]$, $\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, are identically distributed.
2. By the $\mathcal{D}_k$-MDDH assumption, we can switch $([\mathbf{A}]_1, [\mathbf{A}\mathbf{s}]_1)$ to $([\mathbf{A}]_1, [\mathbf{u}]_1)$, where $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$.
3. The uniform distribution over $\mathbb{Z}_p^{k+1}$ and $\mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$ are $\frac{1}{p}$-close, for all $\mathbf{A}$ of rank k.

Combining these three facts, we obtain a PPT adversary $\mathcal{B}_{1,\beta}$ such that

$$\mathsf{Adv}_{\mathsf{G}_{0,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_1,\mathcal{B}_{1,\beta}}(\lambda) + \frac{1}{p}.$$

Now we describe the adversary $\mathcal{B}_{1,\beta}$. Upon receiving an MDDH challenge $(\mathcal{PG}, [\mathbf{A}]_1, [\mathbf{h}]_1)$, $\mathcal{B}_1$ picks $\mathbf{W}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, $\mathbf{V}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times(k+1)}$, and $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ for all $i \in [n]$, thanks to which it can compute and send $(\mathsf{pk}, \{\mathsf{ek}_i\}_{i\in\mathcal{CS}})$ to $\mathcal{A}$, and simulate the oracle $\mathsf{OKeygen}$, as described in Figure 4.8. To simulate $\mathsf{OEnc}(i,\cdot,\cdot)$, $\mathcal{B}_{1,\beta}$ picks $\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, sets $[\mathbf{c}_i]_1 := [\mathbf{A}]_1\mathbf{s}_i + [\mathbf{h}]_1$, and computes the rest of the challenge ciphertext from $[\mathbf{c}_i]_1$. Note that when $[\mathbf{h}]_1$ is a real MDDH challenge, this simulates game $\mathsf{G}_{0,\beta}$, whereas it simulates $\mathsf{G}_{1,\beta}$ when $[\mathbf{h}]_1$ is uniformly random over $\mathbb{G}_1^{k+1}$ (within $\frac{1}{p}$ statistical distance). □

Lemma 34: Game $\mathsf{G}_{1,\beta}$ to $\mathsf{G}_{2,\beta}$

$$\mathsf{Adv}_{\mathsf{G}_{1,\beta}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A}).$$

Proof of Lemma 34. We argue that games $\mathsf{G}_{1,\beta}$ and $\mathsf{G}_{2,\beta}$ are the same, using the fact that for all $\mathbf{A} \in \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{u} \in \mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$, and all $\mathbf{a}^\perp \in \mathbb{Z}_p^{k+1}$ such that $\mathbf{A}^\top\mathbf{a}^\perp = \mathbf{0}$ and $(\mathbf{a}^\perp)^\top\mathbf{u} = 1$, the following distributions are identical:

$$\{\mathbf{V}_i, \mathbf{z}_i\}_{i\in[n]} \text{ and } \{\mathbf{V}_i - \mathbf{z}_i(\mathbf{a}^\perp)^\top, \mathbf{z}_i\}_{i\in[n]},$$

where for all $i \in [n]$, $\mathbf{V}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times(k+1)}$, and $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. This is the case because the matrices $\mathbf{V}_i$ are picked uniformly, independently of the vectors $\mathbf{z}_i$. This way, we obtain

$$\mathbf{d}_i := \mathbf{W}_i^\top\mathbf{y}_i + \left(\mathbf{V}_i^\top - \mathbf{a}^\perp\mathbf{z}_i^\top\right)\mathbf{r} = \mathbf{W}_i^\top\mathbf{y}_i + \mathbf{V}_i^\top\mathbf{r} - \mathbf{a}^\perp\mathbf{z}_i^\top\mathbf{r}$$

and

$$\begin{aligned}
[\mathbf{c}_i'']_1 &:= \left[\left(\mathbf{V}_i - \mathbf{z}_i(\mathbf{a}^\perp)^\top\right)(\mathbf{A}\mathbf{s}_i + \mathbf{u})\right]_1 + [\mathbf{z}_i]_1 \\
&= \left[\mathbf{V}_i(\mathbf{A}\mathbf{s}_i + \mathbf{u})\right]_1 - \left[\mathbf{z}_i(\mathbf{a}^\perp)^\top\mathbf{u}\right]_1 + [\mathbf{z}_i]_1 \\
&= \left[\mathbf{V}_i(\mathbf{A}\mathbf{s}_i + \mathbf{u})\right]_1
\end{aligned}$$

where we use the fact that $(\mathbf{a}^\perp)^\top\mathbf{u} = 1$ is the last equality, and the fact that $\mathbf{A}^\top\mathbf{a}^\perp = \mathbf{0}$ in the penultimate equality. This corresponds to game $\mathsf{G}_{2,\beta}$. □

Lemma 35: Game $\mathsf{G}_{2,\beta}$ to $\mathsf{G}_{3,\beta}$

There exists a PPT adversary $\mathcal{B}_{3,\beta}$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathcal{U}_k(p)\text{-MDDH}}_{\mathbb{G}_2,\mathcal{B}_{3,\beta}}(\lambda) + \frac{1}{p-1}.$$

Proof of Lemma 35. Here, we switch $\{[\mathbf{r}]_2, [\langle \mathbf{z}_i, \mathbf{r}\rangle]_2\}_{i\in\mathcal{HS}}$ used by $\mathsf{OKeygen}$ to $\{[\mathbf{r}]_2, [\widetilde{z}_i]_2\}_{i\in\mathcal{HS}}$, where for all $i \in [n]$, $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\widetilde{z}_1, \ldots, \widetilde{z}_n \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ and $\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. This is justified by the fact that $\{[\mathbf{r}]_2, [\langle \mathbf{z}_i, \mathbf{r}\rangle]_2\}_{i\in\mathcal{HS}}$ is identically distributed to $[\mathbf{U}\mathbf{r}]_2$ where $\mathbf{U} \leftarrow_{\mathrm{R}} \mathcal{U}_{k+h,k}$, where h denotes the size of $\mathcal{HS}$ (wlog. we assume that the upper k rows of $\mathbf{U}$ are full rank), which is indistinguishable from a uniformly random vector over $\mathbb{G}_2^{k+h}$, that is, of the form: $\{[\mathbf{r}]_2, [\widetilde{z}_i]_2\}_{i\in\mathcal{HS}}$, according to the $\mathcal{U}_{k+h,k}(p)$-MDDH assumption. To do the switch simultaneously for all calls to $\mathsf{OKeygen}$, that is, to switch $\{[\mathbf{r}^j]_2, [\langle \mathbf{z}_i, \mathbf{r}^j\rangle]_2\}_{i\in\mathcal{HS},j\in[Q_0]}$ to $\{[\mathbf{r}^j]_2, [\widetilde{z}_i^j]_2\}_{i\in\mathcal{HS},j\in[Q_0]}$, where Q_0 denotes the number of calls to $\mathsf{OKeygen}$, and for all $j \in [Q_0]$, $\mathbf{r}^j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\widetilde{z}_i^j \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ for all $i \in \mathcal{HS}$, we use the Q_0-fold $\mathcal{U}_{k+h,k}(p)$-MDDH assumption. Namely, we build a PPT adversary $\mathcal{B}'_{3,\beta}$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{2,\beta}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3,\beta}}(\mathcal{A}) \leq \mathsf{Adv}^{Q_0\text{- }\mathcal{U}_{k+h,k}(p)\text{-MDDH}}_{\mathbb{G}_2,\mathcal{B}'_{3,\beta}}(\lambda).$$

This, together with Lemma 3 ($\mathcal{U}_k(p)$-MDDH $\Rightarrow$ Q_0-fold $\mathcal{U}_{k+h,k}(p)$-MDDH), implies the lemma. Adversary $\mathcal{B}'_{3,\beta}$ proceeds as follows.

-Simulation of $(\mathsf{pk}, \{\mathsf{ek}_i\}_{i\in\mathcal{CS}})$: Upon receiving an Q_0-fold $\mathcal{U}_{k+h,k}(p)$-MDDH challenge

$$\left(\mathcal{PG}, [\mathbf{U}]_2 \in \mathbb{G}_2^{(k+h)\times k}, \left[\mathbf{h}^1 \| \cdots \| \mathbf{h}^{Q_0}\right]_2 \in \mathbb{G}_2^{(k+n)\times Q_0}\right),$$

$\mathcal{B}'_{3,\beta}$ samples $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \mathsf{Span}(\mathbf{A})$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$ s.t. $\mathbf{A}^\top \mathbf{a}^\perp = \mathbf{0}$ and $\mathbf{u}^\top \mathbf{a}^\perp = 1$, for all $i \in [n]$: $\mathbf{W}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, $\mathbf{V}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times(k+1)}$. For all $i \in \mathcal{CS}$: $\mathbf{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$. It returns $\mathsf{pk} := (\mathcal{PG}, [\mathbf{A}]_1)$, $\{\mathsf{ek}_i := (\mathbf{z}_i, [\mathbf{W}_i\mathbf{A}]_1, [\mathbf{V}_i\mathbf{A}]_1)\}_{i\in\mathcal{CS}})$ to $\mathcal{A}$.

-Simulation of $\mathsf{Enc}(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$:

$\mathcal{B}'_{3,\beta}$ picks $\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, computes $[\mathbf{c}_i]_1 := [\mathbf{A}\mathbf{s}_i]_1 + [\mathbf{u}]_1$, $[\mathbf{c}'_i] := \mathbf{W}_i[\mathbf{c}_i]_1 + [\mathbf{x}_i^\beta]_1$, $[\mathbf{c}''_i]_1 := \mathbf{V}_i[\mathbf{c}_i]_1$, and returns $\begin{bmatrix} -\mathbf{c}_i \\ \mathbf{c}'_i \\ \mathbf{c}''_i \end{bmatrix}_1$ to $\mathcal{A}$.

-Simulation of $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:

On the j'th query $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$, $\mathcal{B}'_{3,\beta}$ sets $[\mathbf{r}]_2 := [\overline{\mathbf{h}^j}]_2$, where $\overline{\mathbf{h}^j} \in \mathbb{Z}_p^k$ denotes the k-upper components of $\mathbf{h}^j \in \mathbb{Z}_p^{k+h}$ (recall that h denotes the size of $\mathcal{HS}$). For each $i \in \mathcal{HS}$, it uses one of the h lowest components of $\mathbf{h}^j$, call it $\underline{\mathbf{h}^j}_i$ (a different one is used for each $i \in \mathcal{HS}$), to compute $[\mathbf{d}_i]_2 := [\mathbf{W}_i^\top \mathbf{y}_i]_2 + \mathbf{V}_i^\top [\overline{\mathbf{h}^j}]_2 - \mathbf{a}^\perp [\underline{\mathbf{h}^j}_i]_2$. For each $i \in \mathcal{CS}$, it computes $[\mathbf{d}_i]_2 := [\mathbf{W}_i^\top \mathbf{y}_i]_2 + \mathbf{V}_i^\top [\overline{\mathbf{h}^j}]_2 - \mathbf{a}^\perp [\langle \mathbf{z}_i, \overline{\mathbf{h}^j}\rangle]_2$.

Note that when $\left[\mathbf{h}^1 \| \cdots \| \mathbf{h}^{Q_0}\right]_2$ is a real MDDH challenge, $\mathcal{B}'_{3,\beta}$ simulate the game $\mathsf{G}_{2,\beta}$, whereas it simulates $\mathsf{G}_{3,\beta}$ when $\left[\mathbf{h}^1 \| \cdots \| \mathbf{h}^{Q_0}\right]_2$ is uniformly random over $\mathbb{G}_2^{(k+h)\times Q_0}$. □

Lemma 36: Game $\mathsf{G}_{3,\beta}$ to $\mathsf{G}^\star_{3,\beta}$

There exists a PPT adversary $\mathcal{A}^\star$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{3,\beta}}(\mathcal{A}) = (X+1)^{2hm} \cdot \mathsf{Adv}_{\mathsf{G}^\star_{3,\beta}}(\mathcal{A}^\star),$$

where h denotes the size of $\mathcal{HS} \subseteq [n]$, the set of honest input slots.

Proof of Lemma 36. Upon receiving a set $\mathcal{CS} \subseteq [n]$ from $\mathcal{A}$, $\mathcal{A}^\star$ guesses the challenge by picking random: $(\mathbf{z}_i^0, \mathbf{z}_i^1)_{i\in\mathcal{HS}} \leftarrow_{\mathrm{R}} [0,X]^{2hm}$, which it sends, together with $\mathcal{CS}$, to the game $\mathsf{G}^\star_{3,\beta}$, which is a selective variant of game $\mathsf{G}_{3,\beta}$. Then it receives a public key pk and ciphertexts $\{\mathsf{ct}_i\}_{i\in\mathcal{HS}}$. Whenever $\mathcal{A}$ queries $\mathsf{OKeygen}$, $\mathcal{A}^\star$ forwards the query to its own oracle, and gives back the answer to $\mathcal{A}$. When $\mathcal{A}$ calls $\mathsf{OEnc}(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$, if $i \in \mathcal{CS}$, then $\mathcal{A}^\star$ queries its own encryption oracle on $(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$ and forwards the answer to $\mathcal{A}$. If $i \in \mathcal{HS}$, then $\mathcal{A}^\star$ verifies its guess was correct, that is, whether $(\mathbf{x}_i^0, \mathbf{x}_i^1) = (\mathbf{z}_i^0, \mathbf{z}_i^1)$. If the guess is incorrect, $\mathcal{A}^\star$ ends the simulation, and sends $\alpha := 0$ to the game $\mathsf{G}^\star_{3,\beta}$. Otherwise, it returns ct_i to $\mathcal{A}$, and keeps answering $\mathcal{A}$'s queries as explained. Finally (if it didn't end the simulation before the end), it forwards $\mathcal{A}$'s output α to the game $\mathsf{G}^\star_{3,\beta}$.

When $\mathcal{A}^\star$ guesses correctly, it simulates $\mathcal{A}$'s view perfectly. When it fails to guess, it outputs $\alpha := 0$. Thus, the probability that $\mathcal{A}^\star$ outputs 1 in $\mathsf{G}^\star_{3,\beta}$ is exactly $(X+1)^{-2hm} \cdot \mathsf{Adv}_{\mathsf{G}_{3,\beta}}(\mathcal{A})$. □

Lemma 37: Game $\mathsf{G}^\star_{3,0}$ to $\mathsf{G}^\star_{3,1}$

For all adversaries $\mathcal{A}'$, we have:

$$\mathsf{Adv}_{\mathsf{G}^\star_{3,0}}(\mathcal{A}') = \mathsf{Adv}_{\mathsf{G}^\star_{3,1}}(\mathcal{A}').$$

Proof of Lemma 37. We show that game $\mathsf{G}^\star_{3,0}$ and $\mathsf{G}^\star_{3,1}$ are perfectly indistinguishable, using an information theoretic argument that crucially relies on the fact that these games are *selective*, and using the restrictions on the oracle queries imposed by the security game.

This proof is similar to the proof of Lemma 30 for the one-SEL-IND-static security of the $\mathcal{MIFE}$ in Figure 4.1.

Namely, We show that $\mathsf{G}^\star_{3,\beta}$ does not depend on β, using the fact that for all $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n \in (\mathbb{Z}_p^m)^n$, for all $\{\mathbf{x}_i^b \in [0,X]^m\}_{i\in\mathcal{HS},b\in\{0,1\}}$, the following are identically distributed:

$$\{\mathbf{W}_i, \widetilde{z}_i\}_{i\in\mathcal{HS}} \quad \text{and} \quad \{\mathbf{W}_i - \mathbf{x}_i^\beta(\mathbf{a}^\perp)^\top, \widetilde{z}_i - \langle \mathbf{x}_i^\beta, \mathbf{y}_i\rangle\}_{i\in\mathcal{HS}},$$

where $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ for all $i \in \mathcal{HS}$, and $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$ such that $\mathbf{A}^\top \mathbf{a}^\perp = 0$ and $\mathbf{u}^\top \mathbf{a}^\perp = 1$.

For each query $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$, $\mathsf{OKeygen}(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ picks values $\widetilde{z}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ and $\mathbf{W}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$ for $i \in \mathcal{HS}$ that are *independent* of $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$ and the challenge $\{\mathbf{x}_i^b \in [0,X]^m\}_{i\in\mathcal{HS},b\in\{0,1\}}$ (note that here we crucially rely on the fact the games $\mathsf{G}^\star_{3,0}$ and $\mathsf{G}^\star_{3,1}$ are *selective* here), therefore, using the previous fact, we can switch $\widetilde{z}_i$ to $\widetilde{z}_i - \langle \mathbf{x}_i^\beta, \mathbf{y}_i\rangle$ and $\mathbf{W}_i$ to $\mathbf{W}_i - \mathbf{x}_i^\beta(\mathbf{a}^\perp)^\top$, for all $i \in \mathcal{HS}$, without changing the distribution of the game.

This way, for all $i \in \mathcal{HS}$, $\mathsf{OEnc}(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$ computes:

$$\mathbf{c}_i' := (\mathbf{W}_i - \mathbf{x}_i^\beta(\mathbf{a}^\perp)^\top)\mathbf{c}_i + \mathbf{x}_i^\beta = (\mathbf{W}_i - \mathbf{x}_i^\beta(\mathbf{a}^\perp)^\top)(\mathbf{A}\mathbf{s}_i + \mathbf{u}) + \mathbf{x}_i^\beta = \mathbf{W}_i\mathbf{c}_i,$$

using the facts that $\mathbf{A}^\top\mathbf{a}^\perp = 0$ and $\mathbf{u}^\top\mathbf{a}^\perp = 1$. That is, $\mathsf{OEnc}(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$ is independent of β, for all $i \in \mathcal{HS}$. Moreover, for all $i \in \mathcal{CS} \cap I$, by definition of the security game, we have $\mathbf{x}_i^0 = \mathbf{x}_i^1$. Thus, $\mathsf{OEnc}(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$ is independent of β, for all $i \in [n]$.

Note that, for all $i \in \mathcal{HS}$, $\mathsf{OKeygen}(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$ computes

$$\mathbf{d}_i := (\mathbf{W}_i - \mathbf{x}_i^\beta(\mathbf{a}^\perp)^\top)^\top \mathbf{y}_i + \mathbf{V}_i^\top \mathbf{r} - \mathbf{a}^\perp \left(\widetilde{z}_i + \langle \mathbf{x}_i^\beta, \mathbf{y}_i\rangle\right) = \mathbf{W}_i^\top \mathbf{y}_i + \mathbf{V}_i^\top \mathbf{r} - \mathbf{a}^\perp \widetilde{z}_i,$$

which does not depend on β.

Finally, $\mathsf{OKeygen}$ also computes:

$$z := \sum_{i \in \mathcal{CS}} \langle \mathbf{z}_i, \mathbf{r} \rangle + \sum_{i \in \mathcal{HS}} \widetilde{z}_i - \sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^{\beta}, \mathbf{y}_i \rangle .$$

Finally, by definition of the security game, we have: $\sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^0, \mathbf{y}_i \rangle = \sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^1, \mathbf{y}_i \rangle$, by taking $\mathbf{x}_i^0 = \mathbf{x}_i^1 = \mathbf{0}$ for all $i \in \mathcal{CS}$ in **Condition 1** from Definition 23. Thus, $\mathsf{G}^{\star}_{3,\beta}$ is independent of β. □

Remark 12: On adaptive security

To achieve *adaptive* security, we split the selective, computational argument used for the proof of Theorem 9, in two steps: first, we use an adaptive, computational argument, that does not involve the challenges $\{\mathbf{x}_i^b\}_{i \in [n], b\{0,1\}}$ (this corresponds to the transition from game $\mathsf{G}_{0,\beta}$ to $\mathsf{G}_{3,\beta}$). Then, we prove security of game $\mathsf{G}_{3,\beta}$, using a selective argument, which involves the challenges $\{\mathbf{x}_i^b\}_{i \in [n], b\{0,1\}}$, but relies on perfect indistinguishability. That is, we prove that $\mathsf{G}_{3,\beta}$ is perfectly secure, by first proving the perfect security of its selective variant, $\mathsf{G}^{\star}_{3,\beta}$, and using a guessing argument to obtain security of the adaptive game $\mathsf{G}_{3,\beta}$. Guessing incurs an exponential security loss, which we can afford, since it is multiplied by a zero term. The proof of Theorem 9 essentially does the two steps at once, which prevents using the same guessing argument (since in that case, the exponential term would be multiplied by the computational advantage).

Chapter 5

Multi-Input Inner-Product Functional Encryption without Pairings

Overview of our construction.

In this chapter we give a (private-key) MIFE scheme for inner products based on a variety of assumptions, notably *without the need of bilinear maps*, and where *decryption works efficiently*, even for messages of super-polynomial size. We achieve this result by proposing a generic construction of MIFE from any single-input FE (for inner products) in which the encryption algorithm is linearly-homomorphic. Our transformation is surprisingly simple, general and efficient. In particular, it does not require pairings (as in the case of the multi-input inner-product FE from [AGRW17], presented in Chapter 4), and it can be instantiated with *all* known single-input functional encryption schemes (e.g., [ABDP15, ABDP16, ALS16]). This allows us to obtain new MIFE for inner products from plain DDH, composite residuosity, and LWE. Beyond the obvious advantage of enlarging the set of assumptions on which MIFE can be based, this result yields schemes that can be used with a much larger message space. Indeed, dropping the bilinear groups requirement allows us to employ schemes where the decryption time is polynomial, rather than exponential, in the message bit size. From a more theoretical perspective, our results also show that, contrary to what was previously conjectured [AGRW17], MIFE for inner product does not need any (qualitatively) stronger assumption than their single-input counterpart.

This result has been published in [ACF+18]. The novelty in this thesis is that security is guaranteed even when some encryption keys are corrupted. Namely, each user $i \in [n]$ receives a (private) encryption key ek_i. Even a collusion of ek_i for some malicious users i cannot break security for the encryption of other slots. This property is obtained without modifying the scheme from [ACF+18], but requires a novel security proof. It is desirable for practical use case of MIFE to assume no particular trust between different users, since the setting already assumes these users do not cooperate or communicate while performing encryption (this would corresponds to the single-input setting).

Our solution, in more detail. Informally, the scheme from the previous chapter builds upon a two-step decryption blueprint. The ciphertexts $\mathsf{ct}_1 = \mathsf{Enc}(\mathbf{x}_1), \ldots, \mathsf{ct}_n = \mathsf{Enc}(\mathbf{x}_n)$ (corresponding to slots $1, \ldots, n$) are all created using different instances of a single-input FE. Decryption is performed in two stages. One first decrypts each single ct_i separately using the secret key $\mathsf{dk}_{\mathbf{y}_i}$ of the underlying single-input FE, and then the outputs of these decryptions are added up to get the final result.

The main technical challenge of this approach is that the stage one of the above decryption

algorithm leaks information on each partial inner product $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$. To avoid this leakage, their idea is to let source i encrypt its plaintext vector $\mathbf{x}_i$ augmented with some fixed (random) value u_i, which is part of the secret key. Moreover, $\mathsf{dk}_{\mathbf{y}_i}$ are built by running the single-input FE key generation algorithm on input $\mathbf{y}_i || r$, i.e., the vector $\mathbf{y}_i$ augmented with fresh randomness r.

By these modifications, and skipping many technical details, stage-one decryption then consists of using pairings to compute, in $\mathbb{G}_T$, the values $[\langle \mathbf{x}_i, \mathbf{y}_i \rangle + u_i r]_T$ for every slot i. From these quantities, the result $[\langle \mathbf{x}, \mathbf{y} \rangle]_T$ is obtained as

$$\prod_{i=1}^{n} [\langle \mathbf{x}_i, \mathbf{y}_i \rangle + u_i r]_T - [\sum_{i=1}^{n} u_i r]_T$$

which can be easily computed if $[\sum_{i=1}^{n} u_i r]_T$ is included in the secret key.

Intuitively, the scheme is secure as the quantities $[u_i r]_T$ are all pseudo-random (under the DDH assumption) and thus hide all the partial information $[\langle \mathbf{x}_i, \mathbf{y}_i \rangle + u_i r]_T$ may leak. Notice that, in order for this argument to go through, it is crucial that the quantities $[\langle \mathbf{x}_i, \mathbf{y}_i \rangle + u_i r]_T$ are all encoded in the exponent, and thus decoding is possible only for small norm exponents. Furthermore, this technique seems to inherently require pairings, as both u_i and r have to remain hidden while allowing to compute an encoding of their product at decryption time. This is why the possibility of a scheme without pairings was considered as "quite surprising" in [AGRW17].

We overcome these difficulties via a new FE to MIFE transform, which manages to avoid leakage in a much simpler and efficient way. Our transformation works in two steps. First, we consider a simplified scheme where only one ciphertext query is allowed and messages live in the ring $\mathbb{Z}_L$, for some integer L. In this setting, we build the following multi-input scheme. For each slot i the (master) secret key for slot i consists of one random vector $\mathbf{u}_i \in \mathbb{Z}_L^m$. Encrypting $\mathbf{x}_i$ merely consists in computing $\mathbf{c}_i = \mathbf{x}_i + \mathbf{u}_i \bmod L$. The secret key for function $\mathbf{y} = (\mathbf{y}_1 \| \dots \| \mathbf{y}_n)$, is just $z_{\mathbf{y}} = \sum_{i=1}^{n} \langle \mathbf{u}_i, \mathbf{y}_i \rangle \bmod L$. To decrypt, one computes

$$\langle \mathbf{x}, \mathbf{y} \rangle \bmod L = \langle (\mathbf{c}_1, \dots, \mathbf{c}_n), \mathbf{y} \rangle - z_{\mathbf{y}} \bmod L$$

Security comes from the fact that, if only one ciphertext query is allowed, the above can be seen as the functional encryption equivalent of the one-time pad[1].

Next, to guarantee security in the more challenging setting where many ciphertext queries are allowed, we just add a layer of (functional) encryption on top of the above one-time encryption. More specifically, we encrypt each $\mathbf{c}_i$ using a FE (supporting inner products) that is both linearly homomorphic and whose message space is compatible with L. So, given ciphertexts $\{\mathsf{ct}_i = \mathsf{Enc}(\mathbf{c}_i)\}$ and secret key $\mathsf{dk}_{\mathbf{y}} = (\{\mathsf{dk}_{\mathbf{y}_i}\}_i, z_{\mathbf{y}})$, one can first obtain $\{\langle \mathbf{c}_i, \mathbf{y}_i \rangle = \mathsf{Dec}(\mathsf{ct}_i, \mathsf{dk}_{\mathbf{y}_i})\}$, and then extract the result as $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} \langle \mathbf{c}_i, \mathbf{y}_i \rangle - \langle \mathbf{u}, \mathbf{y} \rangle$.

Our transformation actually comes in two flavors: the first one addresses the case where the underlying FE computes inner products over some finite ring $\mathbb{Z}_L$; the second one instead considers FE schemes that compute bounded-norm inner products over the integers. In both cases the transformations are generic enough to be instantiated with known single-input FE schemes for inner products. This gives us new MIFE relying on plain DDH [ABDP15], LWE [ALS16] and Decisional Composite Residuosity [ALS16, ABDP16]. Moreover, the proposed transform is security-preserving in the sense that, if the underlying FE achieves adaptive security, so does our resulting MIFE.

From Single to Multi-Input FE for Inner Product

In this section, we give a generic construction of MIFE for inner product from any single-input FE for the same functionality. More precisely, we show two transformations: the first one

[1] We remark that a similar information theoretic construction was put forward by Wee in [Wee17], as a warm-up scheme towards an FE for inner products achieving simulation security.

$\underline{\mathsf{Setup}^{\mathsf{ot}}(1^\lambda, F_n^{m,L})}$:
For all $i \in [n]$, $\mathbf{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_L^m$, $\mathsf{ek}_i := \mathbf{u}_i$, $\mathsf{pk} = L$, $\mathsf{msk} := \{\mathbf{u}_i\}_{i\in[n]}$
Return $(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]})$.

$\underline{\mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i)}$:
Return $\mathbf{x}_i + \mathbf{u}_i \bmod L$.

$\underline{\mathsf{KeyGen}^{\mathsf{ot}}(\mathsf{pk}, \mathsf{msk}, (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n))}$:
$z := \sum_{i\in[n]} \langle \mathbf{u}_i, \mathbf{y}_i \rangle \bmod L$
Return $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n, z)$.

$\underline{\mathsf{Dec}^{\mathsf{ot}}\big(\mathsf{pk}, \mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n\big)}$:
Parse $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n, z)$.
Return $\sum_{i=1}^n \langle \mathsf{ct}_i, \mathbf{y}_i \rangle - z \bmod L$

Figure 5.1: Private-key, information theoretically secure, multi-input FE scheme $\mathcal{MIFE}^{\mathsf{ot}} = (\mathsf{Setup}^{\mathsf{ot}}, \mathsf{Enc}^{\mathsf{ot}}, \mathsf{KeyGen}^{\mathsf{ot}}, \mathsf{Dec}^{\mathsf{ot}})$ for the class $F_n^{m,L}$.

addresses FE schemes that compute the inner product functionality over a finite ring $\mathbb{Z}_L$ for some integer L, while the second transformation addresses FE schemes for bounded-norm inner product. The two transformations are almost the same, and the only difference is that in the case of bounded-norm inner product, we require additional structural properties on the single-input FE. Yet we stress that these properties are satisfied by all existing constructions. Both our constructions rely on a simple MIFE scheme that is one-AD-IND secure unconditionally. In particular, our constructions show how to use single-input FE in order to bootstrap the information-theoretic MIFE from one-time to many-time security.

Information-Theoretic MIFE with One-Time Security

Here we present the multi-input scheme $\mathcal{MIFE}^{\mathsf{ot}}$ for inner product over $\mathbb{Z}_L$, that is, for the set of functionalities $\{F_n^{m,L}\}_{n\in\mathbb{N}}$ defined as $F_n^{m,L} : \mathcal{K}_n \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Z}$, with $\mathcal{K}_n := \mathbb{Z}^{nm}$, for all $i \in [n]$, $\mathcal{X}_i := \mathbb{Z}^m$, $\mathcal{Z} := \mathbb{Z}_L$, such that for any $(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n) \in \mathcal{K}_n$, $\mathbf{x}_i \in \mathcal{X}_i$, we have:

$$F_n^{m,L}((\mathbf{y}_1 \| \cdots \| \mathbf{y}_n), \mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{i=1}^n \langle \mathbf{x}_i, \mathbf{y}_i \rangle \bmod L.$$

We prove its one-AD-IND security. The scheme is described in Figure 5.1.

Theorem 12: one-AD-IND security

The MIFE described in Figure 5.1 is one-AD-IND-weak secure. Namely, for any adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{A}}(\lambda) = 0$.

Proof of Theorem 12. Let $\mathcal{A}$ be an adversary against the one-AD-IND security of the MIFE. First, we use a guessing argument to build an adversary $\mathcal{B}$ such that:

$$\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq 2^{-n} \cdot L^{-2mn} \cdot \mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND\text{-}weak\text{-}static}}_{\mathcal{MIFE},\mathcal{B}}(\lambda).$$

First, $\mathcal{B}$ samples $\mathcal{CS} \subseteq [n]$ uniformly at random among all subset of $[n]$. We denote $\mathcal{HS} := [n] \setminus \mathcal{CS}$. Then, for all $i \in \mathcal{HS}$, it samples $(\mathbf{z}_i^1, \mathbf{z}_i^0) \leftarrow_{\mathrm{R}} \mathbb{Z}_L^{2m}$, which is a guess of the challenge ciphertexts. Then, $\mathcal{B}$ sends $\Big(\mathcal{CS}, \{\mathbf{z}_i^b\}_{i\in\mathcal{HS}, b\in\{0,1\}}\Big)$ to its own experiment, upon which it receives $(\mathsf{pk}, \{\mathsf{ek}_i\}_{i\in\mathcal{CS}}\{\mathsf{ct}_i\}_{i\in\mathcal{HS}})$, where for all $i \in \mathcal{HS}$, $\mathsf{ct}_i := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{z}_i^\beta)$, where $\beta \in \{0,1\}$ corresponds to the experiment $\mathbf{one\text{-}SEL\text{-}IND}_\beta(1^\lambda, \mathcal{A})$ the adversary $\mathcal{B}$ is interacting with. It sends pk to $\mathcal{A}$. For every query to $\mathsf{OKeygen}$, $\mathcal{B}$ queries its own decryption key oracle oracle on the same input, and returns the answer to $\mathcal{A}$. For every query $i \in [n]$ of $\mathcal{A}$ to $\mathsf{OCorrupt}$, $\mathcal{B}$ verifies its guess was correct, namely, that $i \in \mathcal{CS}$. If not, $\mathcal{B}$ ends the simulation and returns $\alpha = 0$ to its experiment. For every query $(i, \mathbf{x}_i^0, \mathbf{x}_i^1)$ to OEnc, $\mathcal{B}$ verifies its guess is correct, namely, whether ($i \in \mathcal{CS}$ and $\mathbf{x}_i^0 = \mathbf{x}_i^1$), or $(\mathbf{x}_i^0, \mathbf{x}_i^1) = (\mathbf{z}_i^0, \mathbf{z}_i^1)$. If it is not the case, $\mathcal{B}$ ends the simulation, and returns $\alpha = 0$ to its own experiment. If this is case, $\mathcal{B}$ does the

following: if $i \in \mathcal{CS}$, then it returns $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^0)$ to $\mathcal{A}$ (note that it can do so since it knows ek_i for all $i \in \mathcal{CS}$); if $i \notin \mathcal{CS}$, it returns ct_i to $\mathcal{A}$. Finally (if the simulation didn't end before), $\mathcal{B}$ forwards $\mathcal{A}$'s output α to its experiment.

When $\mathcal{B}$'s guess is correct, then it simulates $\mathcal{A}$'s view perfectly. The guess is correct with probability at least $2^{-n} \cdot L^{-2mn}$. When the guess is incorrect, then $\mathcal{B}$ returns $\alpha = 0$ to its experiment. Thus, we obtain $\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq 2^{-n} \cdot L^{-2mn} \cdot \mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND\text{-}weak\text{-}static}}_{\mathcal{MIFE},\mathcal{B}}(\lambda)$.

It remains to prove that the MIFE presented in Figure 5.1 satisfies perfect one-SEL-IND security, under static corruptions. Namely, for any adversary $\mathcal{B}$,

$$\mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND\text{-}weak\text{-}static}}_{\mathcal{MIFE},\mathcal{B}}(\lambda) = 0.$$

To do so, we introduce hybrid games $\mathsf{H}_\beta(1^\lambda, \mathcal{B})$ described in Figure 5.2. We prove that for all $\beta \in \{0,1\}$, $\mathsf{H}_\beta(1^\lambda, \mathcal{B})$ is identical to the experiment **one-SEL-IND-weak-static**$^{\mathcal{MIFE}}_\beta(1^\lambda, \mathcal{B})$ (this game is defined as **many-AD-**$^{\mathcal{MIFE}}_\beta(1^\lambda, \mathcal{B})$ from Definition 23, with the one, SEL, weak, and static restrictions). This can be seen using the fact that for all $\{\mathbf{x}_i^\beta \in \mathbb{Z}^m\}_{i \in \mathcal{HS}}$, where $\mathcal{HS} := [n] \setminus \mathcal{CS}$, the following distributions are identical: $\{\mathbf{u}_i \bmod L\}_{i \in \mathcal{HS}}$ and $\{\mathbf{u}_i - \mathbf{x}_i^\beta \bmod L\}_{i \in \mathcal{HS}}$, with $\mathbf{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_L^m$. Note that the independence of the $\mathbf{x}_i^\beta$ from the $\mathbf{u}_i$ is only true in the selective security game. We denote by $I \subseteq [n]$ the set of input slots that is queried by the adversary. We use the fact that for all $i \in I \cap \mathcal{CS}$, it must be that $\mathbf{x}_i^0 = \mathbf{x}_i^1$. This is implied by the definition of the security game, and the fact that $\mathcal{HS} \subseteq I$, that is, every honest slot is queried by the adversary, since we are only proving one-SEL-IND-*weak*-static security. Finally, we show that $\mathcal{B}$'s view in $\mathsf{H}_\beta(1^\lambda, \mathcal{B})$ is independent of β. Indeed, the only information about β that leaks in this experiment is $\sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^\beta, \mathbf{y}_i \rangle$. Moreover, by definition of the security game, we have $\sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^0, \mathbf{y}_i \rangle = \sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^1, \mathbf{y}_i \rangle$ (this follows by taking $\mathbf{x}_i^0 = \mathbf{x}_i^1 = \mathbf{0}$ for all $i \in \mathcal{CS}$ in **Condition 1** from Definition 23). □

$\underline{\mathsf{H}_\beta(1^\lambda, \mathcal{B})}$:	
$(\mathcal{CS}, \{\mathbf{x}_i^b\}_{i \in I \subseteq [n], b \in \{0,1\}}) \leftarrow \mathcal{B}(1^\lambda, F_n^{m,L})$ For all $i \in [n]$: $\mathbf{u}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_L^m$ For all $i \in \mathcal{CS}$, $\mathsf{ek}_i := \mathbf{u}_i$. For all $i \in \mathcal{HS}$, $\mathsf{ct}_i := \mathbf{u}_i$. For all $i \in I \cap \mathcal{CS}$: $\mathsf{ct}_i := \mathbf{u}_i + \mathbf{x}_i^0$ $\alpha \leftarrow \mathcal{B}^{\mathsf{OKeygen}(\cdot), \mathsf{OCorrupt}(\cdot)}(\mathsf{pk}, \{\mathsf{ek}_i\}_{i \in \mathcal{CS}}, \{\mathsf{ct}_i\}_{i \in I})$ Output α	$\underline{\mathsf{OKeygen}(\mathbf{y})}$: Return $\sum_{i \in [n]} \langle \mathbf{u}_i, \mathbf{y}_i \rangle - \sum_{i \in \mathcal{HS}} \langle \mathbf{x}_i^\beta, \mathbf{y}_i \rangle \bmod L$

Figure 5.2: Experiments for the proof of Theorem 12. Note that $\mathcal{HS} \subseteq I$, where I denotes the set of input slots that are queried by $\mathcal{A}$.

Remark 13: Linear homomorphism

We use the fact that $\mathsf{Enc}^{\mathsf{ot}}$ is linearly homomorphic, that is, for all $i \in [n]$, $\mathbf{x}_i, \mathbf{x}_i' \in \mathbb{Z}^m$, $\mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i) + \mathbf{x}_i' \bmod L = \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i + \mathbf{x}_i')$, with probability 1 over the choice of $(\mathsf{pk}, (\mathsf{ek}_i)_{i \in [n]}) \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, F_n^{m,L})$. This property will be used when using the one-time scheme $\mathcal{MIFE}^{\mathsf{ot}}$ from Figure 5.1 as a building block to obtain a full-fledged many-AD-IND MIFE.

Our Transformation for Inner Product over $\mathbb{Z}_L$

We present our multi-input scheme $\mathcal{MIFE}$ for the class $F_n^{m,L}$ in Figure 5.3. The construction relies on the one-time scheme $\mathcal{MIFE}^{\mathsf{ot}}$ of Figure 5.1, and any single-input, public-key FE for

the functionality $F_{\mathsf{IP}}^{m,L} : \mathcal{K} \times \mathcal{X} \to \mathcal{Z}$, with $\mathcal{K} := \mathbb{Z}^m$, $\mathcal{X} := \mathbb{Z}^m$, $\mathcal{Z} := \mathbb{Z}_L$, such that for any $\mathbf{y} \in \mathcal{K}$, $\mathbf{x} \in \mathcal{X}$, we have:

$$F_{\mathsf{IP}}^{m,L}(\mathbf{y}, \mathbf{x}) = \langle \mathbf{x}, \mathbf{y} \rangle \bmod L.$$

$\underline{\mathsf{Setup}(1^\lambda, F_n^{m,L})}$:
$(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \{\mathsf{ek}_i^{\mathsf{ot}}\}_{i\in[n]}) \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, F_n^{m,L})$, $\mathsf{gpk}' \leftarrow \mathsf{GSetup}'(1^\lambda, F_{\mathsf{IP}}^{m,L})$. For all $i \in [n]$, $(\mathsf{ek}_i', \mathsf{msk}_i') \leftarrow \mathsf{Setup}'(1^\lambda, \mathsf{gpk}', F_{\mathsf{IP}}^{m,L})$
$\mathsf{pk} := (\mathsf{pk}^{\mathsf{ot}}, \mathsf{gpk}', \{\mathsf{ek}_i'\}_{i\in[n]})$, $\mathsf{msk} := (\mathsf{msk}^{\mathsf{ot}}, \{\mathsf{msk}_i'\}_{i\in[n]})$, for all $i \in [n]$, $\mathsf{ek}_i := (\mathsf{ek}_i^{\mathsf{ot}}, \mathsf{ek}_i')$
Return $(\mathsf{pk}, \mathsf{msk}\{\mathsf{ek}_i\}_{i\in[n]})$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i)}$:
$\mathbf{w}_i := \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}_i^{\mathsf{ot}}, \mathbf{x}_i)$
Return $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{w}_i)$

$\underline{\mathsf{KeyGen}(\mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:
For all $i \in [n]$, $\mathsf{dk}_i' \leftarrow \mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}_i', \mathbf{y}_i)$, $z := \mathsf{KeyGen}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$
$\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\{\mathsf{dk}_i'\}_{i\in[n]}, z)$
Return $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

$\underline{\mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)}$:
Parse $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\{\mathsf{dk}_i'\}_{i\in[n]}, z)$. For all $i \in [n]$, $d_i := \mathsf{Dec}'(\mathsf{gpk}', \mathsf{dk}_i', \mathsf{ct}_i)$
Return $\sum_{i\in[n]} d_i - z \bmod L$

Figure 5.3: Private-key multi-input FE scheme $\mathcal{MIFE} := (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the functionality $F_n^{m,L}$ from a public-key single-input FE $\mathcal{FE} := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ for the functionality $F_{\mathsf{IP}}^{m,L}$, and the one-time multi-input FE $\mathcal{MIFE}^{\mathsf{ot}} = (\mathsf{Setup}^{\mathsf{ot}}, \mathsf{Enc}^{\mathsf{ot}}, \mathsf{KeyGen}^{\mathsf{ot}}, \mathsf{Dec}^{\mathsf{ot}})$ for the functionality $F_n^{m,L}$ from Figure 5.1.

Correctness of $\mathcal{MIFE}$ follows from the correctness properties of the single-input scheme $\mathcal{FE}$ and the multi-input scheme $\mathcal{MIFE}^{\mathsf{ot}}$. Indeed, correctness of the former implies that, for all $i \in [n]$, $d_i = \langle \mathbf{w}_i, \mathbf{y}_i \rangle \bmod L$, while correctness of $\mathcal{MIFE}^{\mathsf{ot}}$ implies that $\sum_{i\in[n]} d_i - z = \mathsf{Dec}^{\mathsf{ot}}(z, \mathbf{w}_1, \ldots, \mathbf{w}_n) = \sum_{i\in[n]} \langle \mathbf{x}_i, \mathbf{y}_i \rangle \bmod L$.

Theorem 13: many-AD-IND security

If $\mathcal{FE}$ is many-AD-IND secure, and $\mathcal{MIFE}^{\mathsf{ot}}$ is one-AD-IND-weak secure, then $\mathcal{MIFE}$ described in Figure 5.3 is many-AD-IND-secure.

Since the proof of the above theorem is almost the same as the one for the case of bounded-norm inner product, we only provide an overview here, and defer to the proof of Theorem 14 for further details.

Proof overview. First, we use Theorem 2 which prove that many-AD-IND security follows from many-AD-IND-weak and many-AD-IND-zero of $\mathcal{MIFE}$, using an extra layer of symmetric encryption on top of the decryption keys (see Figure 2.1). The many-AD-IND-zero of $\mathcal{MIFE}$ follows directly from the many-AD-IND security of $\mathcal{FE}$ for n instances (which is implied by many-AD-IND security of $\mathcal{FE}$ for one instance, see Lemma 5). Thus, it remains to prove many-AD-IND-weak security of $\mathcal{MIFE}$.

To do so, we first switch encryptions of $\mathbf{x}_1^{1,0}, \ldots, \mathbf{x}_n^{1,0}$ to those of $\mathbf{x}_1^{1,1}, \ldots, \mathbf{x}_n^{1,1}$, using the one-AD-IND security of $\mathcal{MIFE}^{\mathsf{ot}}$. For the remaining ciphertexts, we switch from an encryption of $\mathbf{x}_i^{j,0} = (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,0}$ to that of $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$. In this step we use the fact that one can

compute an encryption of $\mathsf{Enc}^{\mathsf{ot}}(\mathbf{u}, i, (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,0})$ from an encryption $\mathsf{Enc}^{\mathsf{ot}}(\mathbf{u}, i, \mathbf{x}_i^{1,0})$, because the encryption algorithm $\mathsf{Enc}^{\mathsf{ot}}$ of $\mathcal{MIFE}^{\mathsf{ot}}$ is linearly homomorphic (see Remark 13). Finally, we use the many-AD-IND security of $\mathcal{FE}$ for n instance (which is implied by many-AD-IND security of $\mathcal{FE}$ for one instance, see Lemma 5) to switch encryptions of

$$(\mathbf{x}_i^{2,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_i,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$$

to those of

$$(\mathbf{x}_i^{2,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_i,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}.$$

Instantiations. The construction in Figure 5.3 can be instantiated using the single-input public-key FE schemes from [ALS16] that are many-AD-IND-secure and allow for computing inner products over a finite ring. Specifically, we obtain:

- A MIFE for inner product over $\mathbb{Z}_p$ for a prime p, based on the LWE assumption. This is obtained using the LWE-based scheme of Agrawal et al. [ALS16, Section 4.2].
- A MIFE for inner product over $\mathbb{Z}_N$ where N is an RSA modulus, based on Paillier's Decisional Composite Residuosity assumption. This is obtained using the DCR-based scheme of Agrawal et al. [ALS16, Section 5.2].

We note that since both these schemes in [ALS16] have a stateful key generation, our MIFE inherits this stateful property. Stateless MIFE instantiations are obtained from the transformation in the next section.

Our Transformation for Inner Product over $\mathbb{Z}$

Here we present our transformation for the case of bounded-norm inner product. In particular, in Figure 5.4 we present a multi-input scheme $\mathcal{MIFE}$ for the set of functionalities $\{F_n^{m,X,Y}\}_{n\in\mathbb{N}}$ defined as $F_n^{m,X,Y} : \mathcal{K}_n \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Z}$, with $\mathcal{K}_n := [0, Y]^{mn}$, for all $i \in [n]$, $\mathcal{X}_i := [0, X]^m$, $\mathcal{Z} := \mathbb{Z}$, such that for any $(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n) \in \mathcal{K}_n$, $\mathbf{x}_i \in \mathcal{X}_i$, we have:

$$F_n^{m,X,Y}((\mathbf{y}_1 \| \cdots \| \mathbf{y}_n), \mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle.$$

Our transformation builds upon the one-time scheme $\mathcal{MIFE}^{\mathsf{ot}}$ of Figure 5.1, and a single-input, public-key scheme $\mathcal{FE}$ for the class $F_{\mathsf{IP}}^{m,3X,Y}$.[2] We require $\mathcal{FE}$ to satisfy two properties. The first one, that we call *two-step decryption*, intuitively says that the $\mathcal{FE}$ decryption algorithm works in two steps: the first step uses the decryption key to output an encoding of the result, while the second step returns the actual result $\langle \mathbf{x}, \mathbf{y} \rangle$ provided that the bounds $\|\mathbf{x}\|_\infty < X$, $\|\mathbf{y}\|_\infty < Y$ hold. The second property informally says that the $\mathcal{FE}$ encryption algorithm is additively homomorphic.

We note that the two-step property also says that the encryption algorithm accepts inputs $\mathbf{x}$ such that $\|\mathbf{x}\|_\infty > X$, yet correctness is guaranteed as long as the encrypted inputs are within the bound at the moment of invoking the second step of decryption.

Two-step decryption is formally defined as follows.

Property 1: Two-step decryption

An FE scheme $\mathcal{FE} = (\mathsf{GSetup}, \mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ satisfies *two-step decryption* if it admits PPT algorithms $\mathsf{GSetup}^\star, \mathsf{Dec}_1, \mathsf{Dec}_2$ and an encoding function $\mathcal{E}$ such that:

1. For all $\lambda, m, n, X, Y \in \mathbb{N}$, $\mathsf{GSetup}^\star(1^\lambda, F_{\mathsf{IP}}^{m,X,Y}, 1^n)$ outputs gpk which includes a

[2] The reason why we need $3X$ instead of X is due to maintain a correct distribution of the inputs in the security proof.

bound $B \in \mathbb{N}$, and the description of a group $\mathbb{G}$ (with group law $\circ$) of order $L > n \cdot m \cdot X \cdot Y$, which defines the encoding function $\mathcal{E} : \mathbb{Z}_L \times \mathbb{Z} \to \mathbb{G}$.

2. For all $\mathsf{gpk} \leftarrow \mathsf{GSetup}^\star(1^\lambda, F_{\mathsf{IP}}^{m,X,Y}, 1^n)$, $(\mathsf{ek}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})$, $\mathbf{x} \in \mathbb{Z}^m$, $\mathsf{ct}_\mathbf{x} \leftarrow \mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x})$, $\mathbf{y} \in \mathbb{Z}^m$, and $\mathsf{dk}_\mathbf{y} \leftarrow \mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \mathbf{y})$, we have

$$\mathsf{Dec}_1(\mathsf{gpk}, \mathsf{ct}_\mathbf{x}, \mathsf{dk}_\mathbf{y}) = \mathcal{E}(\langle \mathbf{x}, \mathbf{y} \rangle \bmod L, \mathsf{noise}),$$

for some $\mathsf{noise} \in \mathbb{N}$ that depends on $\mathsf{ct}_\mathbf{x}$ and $\mathsf{dk}_\mathbf{y}$. Furthermore, it holds that for all $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^m$, $\Pr[\mathsf{noise} < B] = 1 - \mathsf{negl}(\lambda)$, where the probability is taken over the random coins of $\mathsf{GSetup}^\star$, Setup, Enc and KeyGen. Note that there is no restriction on the magnitude of $\langle \mathbf{x}, \mathbf{y} \rangle$ here, and we are assuming that Enc accepts inputs $\mathbf{x}$ whose norm may be larger than the bound.

3. Given any $\gamma \in \mathbb{Z}_L$, and gpk, one can efficiently compute $\mathcal{E}(\gamma, 0)$.

4. The encoding $\mathcal{E}$ is linear, that is: for all $\gamma, \gamma' \in \mathbb{Z}_L, , \mathsf{noise}, \mathsf{noise}' \in \mathbb{Z}$, we have

$$\mathcal{E}(\gamma, \mathsf{noise}) \circ \mathcal{E}(\gamma', \mathsf{noise}') = \mathcal{E}(\gamma + \gamma' \bmod L, \mathsf{noise} + \mathsf{noise}').$$

5. For all $\gamma < n \cdot m \cdot X \cdot Y$, and $\mathsf{noise} < n \cdot B$, $\mathsf{Dec}_2(\mathsf{gpk}, \mathcal{E}(\gamma, \mathsf{noise})) = \gamma$.

The second property is as follows.

Property 2: Linear encryption

For any FE scheme $\mathcal{FE} = (\mathsf{GSetup}, \mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ satisfying the two-step property, we define the following additional property. There exists a deterministic algorithm Add that takes as input a ciphertext and a message, such that for all $\mathbf{x}, \mathbf{x}' \in \mathbb{Z}^m$, the following are identically distributed:

$$\mathsf{Add}(\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x}), \mathbf{x}'), \quad \text{and} \quad \mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, (\mathbf{x} + \mathbf{x}' \bmod L)),$$

where $\mathsf{gpk} \leftarrow \mathsf{GSetup}^\star(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})$, $(\mathsf{ek}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})$. Note that the value $L \in \mathbb{N}$ is defined as part of the output of the algorithm $\mathsf{Setup}^\star$ (see the two-step property above). We later use a single input FE with this property as a building block for a multi-input FE (see Figure 5.4); this property however is only used in the security proof of our transformation.

Instantiations. It is not hard to check that these two properties are satisfied by known functional encryption schemes for (bounded-norm) inner product. In particular, in Section 5.2, we show that this is satisfied by the many-AD-IND secure FE schemes from [ALS16]. This allows us to obtain MIFE schemes for bounded-norm inner product based on a variety of assumptions such as plain DDH, Decisional Composite Residuosity, and LWE. In addition to obtaining the first schemes without the need of pairing groups, we also obtain schemes where decryption works efficiently even for large outputs. This stands in contrast to the previous result in the previous chapter, where decryption requires to extract discrete logarithms.

Correctness. The correctness of the scheme $\mathcal{MIFE}$ follows from (i) the correctness and Property 1 (two-step decryption) of the single-input scheme, and (ii) from the correctness of $\mathcal{MIFE}^{\mathsf{ot}}$ and the linear property of its decryption algorithm $\mathsf{Dec}^{\mathsf{ot}}$.

More precisely, consider any vector $\mathbf{x} := (\mathbf{x}_1 \| \cdots \| \mathbf{x}_n) \in (\mathbb{Z}^m)^n$, $\mathbf{y} \in \mathbb{Z}^{mn}$, such that $\|\mathbf{x}\|_\infty < X$, $\|\mathbf{y}\|_\infty < Y$, and let $(\mathsf{pk}, \mathsf{msk}, \{\mathsf{ek}_i\}_{i \in [n]}) \leftarrow \mathsf{Setup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})$, $\mathsf{dk}_\mathbf{y} \leftarrow \mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \mathbf{y})$,

$\underline{\mathsf{Setup}(1^\lambda, F_n^{m,X,Y})}$:
$\big(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \{\mathsf{ek}_i^{\mathsf{ot}}\}_{i\in[n]}\big) \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, F_n^{m,X,Y})$, $\mathsf{gpk}' \leftarrow \mathsf{GSetup}^\star(1^\lambda, F_{\mathsf{IP}}^{m,3X,Y}, 1^n)$, for all $i \in [n]$,
$\big(\mathsf{ek}_i', \mathsf{msk}_i'\big) \leftarrow \mathsf{Setup}'(1^\lambda, \mathcal{F}_1^{m,3X,Y}, 1^n)$, $\mathsf{ek}_i := \mathsf{ek}_i^{\mathsf{ot}}$
$\mathsf{pk} := (\mathsf{gpk}, \{\mathsf{ek}_i'\}_{i\in[n]})$, $\mathsf{msk} := \big(\mathsf{msk}^{\mathsf{ot}}, \{\mathsf{msk}_i'\}_{i\in[n]}\big)$,
Return $\big(\mathsf{pk}, \mathsf{msk}, \{\mathsf{ek}_i\}_{i\in[n]}\big)$

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i)}$:
$\mathbf{w}_i := \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}_i^{\mathsf{ot}}, \mathbf{x}_i)$
Return $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{w}_i)$

$\underline{\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:
For all $i \in [n]$, $\mathsf{dk}_i' \leftarrow \mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}_i', \mathbf{y}_i)$, $z \leftarrow \mathsf{KeyGen}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$
$\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := \big(\{\mathsf{dk}_i'\}_{i\in[n]}, z\big)$
Return $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

$\underline{\mathsf{Dec}(\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)}$:
Parse $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\{\mathsf{dk}_i'\}_{i\in[n]}, z)$. For all $i \in [n]$, $\mathcal{E}(\langle \mathbf{x}_i + \mathbf{u}_i, \mathbf{y}_i\rangle \bmod L, \mathsf{noise}_i) \leftarrow \mathsf{Dec}_1(\mathsf{dk}_i', \mathsf{ct}_i)$
Return $\mathsf{Dec}_2\big(\mathcal{E}(\langle \mathbf{x}_1 + \mathbf{u}_1, \mathbf{y}_1\rangle \bmod L, \mathsf{noise}_1) \circ \cdots \circ \mathcal{E}(\langle \mathbf{x}_n + \mathbf{u}_n, \mathbf{y}_n\rangle \bmod L, \mathsf{noise}_n) \circ \mathcal{E}(-z, 0)\big)$

Figure 5.4: Private-key multi-input FE scheme $\mathcal{MIFE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the functionality $F_n^{m,X,Y}$ from public-key single-input FE scheme $\mathcal{FE} = (\mathsf{GSetup}', \mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ for the functionality $F_{\mathsf{IP}}^{m,3X,Y}$ and the one-time multi-input FE $\mathcal{MIFE}^{\mathsf{ot}} = (\mathsf{Setup}^{\mathsf{ot}}, \mathsf{Enc}^{\mathsf{ot}}, \mathsf{KeyGen}^{\mathsf{ot}}, \mathsf{Dec}^{\mathsf{ot}})$ from Figure 5.1.

and $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i)$ for all $i \in [n]$.

By (2) of Property 1, the decryption algorithm $\mathsf{Dec}(\mathsf{dk}_{\mathbf{y}}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$ computes $\mathcal{E}(\langle \mathbf{w}_i, \mathbf{y}_i\rangle \bmod L, \mathsf{noise}_i) \leftarrow \mathsf{Dec}_1(\mathsf{dk}_i', \mathsf{ct}_i)$ where for all $i \in [n]$, $\mathsf{noise}_i < B$, with probability $1 - \mathsf{negl}(\lambda)$.

By (4) of Property 1 (linearity of $\mathcal{E}$), and the correctness of $\mathcal{MIFE}^{\mathsf{ot}}$ we have:

$$\begin{aligned}&\mathcal{E}(\langle \mathbf{w}_1, \mathbf{y}_1\rangle \bmod L, \mathsf{noise}_1) \circ \cdots \circ \mathcal{E}(\langle \mathbf{w}_n, \mathbf{y}_n\rangle \bmod L, \mathsf{noise}_n) \circ \mathcal{E}(-z, 0)\\ &= \mathcal{E}\left(\mathsf{Dec}^{\mathsf{ot}}(z, \mathbf{w}_1, \ldots, \mathbf{w}_n), \sum_{i\in[n]} \mathsf{noise}_i\right) = \mathcal{E}\left(\langle \mathbf{x}, \mathbf{y}\rangle \bmod L, \sum_{i\in[n]} \mathsf{noise}_i\right).\end{aligned}$$

Since $\langle \mathbf{x}, \mathbf{y}\rangle < n \cdot m \cdot X \cdot Y < L$ and $\sum_{i\in[n]} \mathsf{noise}_i < n \cdot B$, we have

$$\mathsf{Dec}_2\big(\mathcal{E}(\langle \mathbf{x}, \mathbf{y}\rangle \bmod L, \sum_{i\in[n]} \mathsf{noise}_i)\big) = \langle \mathbf{x}, \mathbf{y}\rangle,$$

by (5) of Property 1.

Proof of Security. In the following theorem we show that our construction is a many-AD-IND-secure MIFE, assuming that the underlying single-input FE scheme is many-AD-IND-secure, and the scheme $\mathcal{MIFE}^{\mathsf{ot}}$ is one-AD-IND secure.

Theorem 14: many-AD-IND security

Assume that the single-input FE: $\mathcal{FE}$, is many-AD-IND secure and the multi-input FE $\mathcal{MIFE}^{\mathsf{ot}}$ is one-AD-IND-weak secure. Then the multi-input FE $\mathcal{MIFE}$ in Figure 5.4 is many-AD-IND secure.

Proof of Theorem 14. Using Theorem 2, it is sufficient to prove many-AD-IND-zero (i.e. the scheme is secure when no decryption keys are queried), and many-AD-IND-weak i.e. we assume

Game	ct_i^j	justification/remark
G_0	$\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,0})$	**many-AD-IND-weak**$_0^{\mathcal{MIFE}}(\mathcal{A}, 1^\lambda)$ security game
G_1	$\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})$	one-AD-IND-weak security of $\mathcal{MIFE}^{\mathsf{ot}}$, Lemma 33
G_2	$\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1})$	many-AD-IND security of $\mathcal{FE}$ for n instances, Lemma 34

Figure 5.5: An overview of the games used in the proof of Theorem 14.

Games: G_0, $\boxed{\mathsf{G}_1}$, $\boxed{\mathsf{G}_2}$:
$(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \{\mathsf{ek}_i^{\mathsf{ot}}\}_{i\in[n]} \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, F_n^{m,X,Y})$, $\mathsf{gpk}' \leftarrow \mathsf{GSetup}'(1^\lambda, F_{\mathsf{IP}}^{m,3X,Y})$, for all $i \in [n]$, $(\mathsf{ek}_i', \mathsf{msk}_i') \leftarrow \mathsf{Setup}'(1^\lambda, \mathsf{gpk}, F_n^{m,3X,Y})$, $\mathsf{ek}_i := \mathsf{ek}_i^{\mathsf{ot}}$, $\mathsf{pk} := \{\mathsf{gpk}', \mathsf{ek}_i'\}_{i\in[n]}$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OKeygen}(\cdot),\mathsf{OEnc}(\cdot,\cdot),\mathsf{OCorrupt}(\cdot)}(\mathsf{pk})$
Return α.

$\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}))$:
$\mathbf{w}_i^j := \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}_i^{\mathsf{ot}}, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,0})$
$\boxed{\mathbf{w}_i^j := \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}_i^{\mathsf{ot}}, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})}$
$\boxed{\mathbf{w}_i^j := \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}_i^{\mathsf{ot}}, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1})}$
Return $\mathsf{ct}_i^j := \mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{w}_i^j)$.

$\mathsf{OKeygen}(\mathbf{y})$:
For all $i \in [n]$, $\mathsf{dk}_i' \leftarrow \mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}_i', \mathbf{y}_i)$, $z \leftarrow \mathsf{KeyGen}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$, $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\{\mathsf{dk}_i'\}_{i\in[n]}, z)$
Return $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}$

$\mathsf{OCorrupt}(i)$:
Return $\mathsf{ek}_i^{\mathsf{ot}}$

Figure 5.6: Games for the proof of Theorem 14.

the adversary requests a challenge ciphertext for all slots $i \in \mathcal{HS}$, where $\mathcal{HS} := [n] \setminus \mathcal{CS}$ denotes the set of slots that are not corrupted) to obtain many-AD-IND security.

The many-AD-IND-zero security of $\mathcal{MIFE}$ follows directly from the many-AD-IND security of $\mathcal{FE}$ for n instances (which is implied by the security for a single instance, see Lemma 5). In what follows, we prove many-AD-IND-weak security of $\mathcal{MIFE}$.

We proceed via a series of games G_i for $i \in \{0, \ldots, 2\}$, described in Figure 5.6. The transitions are summarized in Figure 5.5. Let $\mathcal{A}$ be a PPT adversary. For any game G, we denote by $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ the probability that the game G outputs 1 when interacting with $\mathcal{A}$. Note that the set of input slots for which a challenge ciphertext is queried, denoted by I in Figure 5.6, is such that $\mathcal{HS} \subseteq I$, since we want to prove many-AD-IND-*weak* security.

According to Definition 21, we have: $\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})|$.

Game G_1: is as game G_0, except we replace the challenge ciphertexts to $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})$ for all $i \in [n]$ and $j \in [Q_i]$, using the one-AD-IND-weak security of $\mathcal{MIFE}^{\mathsf{ot}}$. Namely, we prove in Lemma 33 that there exists a PPT adversary $\mathcal{B}_1$ such that

$$\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{B}_1}(\lambda).$$

Game G_2: we replace the challenge ciphertexts to $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1})$ for all $i \in [n]$ and $j \in [Q_i]$, using the many-AD-IND security of $\mathcal{FE}$ for n instances, which is implied by the single-instance security (see Lemma 5). We prove in

Lemma 34 that there exists a PPT adversary $\mathcal{B}_2$ such that

$$\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda).$$

Putting everything together, we obtain:

$$\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda).$$

□

Lemma 38: Game G_0 to G_1

There exists a PPT adversary $\mathcal{B}_1$ such that

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{B}_1}(\lambda).$$

Proof of Lemma 38. In game G_1, which is described in Figure 5.6, we replace $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,0} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}))$ with $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}))$ for all $i \in [n], j \in [Q_i]$. This is justified by one-AD-IND-weak security of $\mathcal{MIFE}^{\mathsf{ot}}$. The adversary $\mathcal{B}_1$ proceeds as follows.

-Simulation of pk:

Adversary $\mathcal{B}_1$ receives $\mathsf{pk}^{\mathsf{ot}}$ from its experiment. Then, it samples $\mathsf{gpk}' \leftarrow \mathsf{GSetup}'(1^\lambda, F_{\mathsf{IP}}^{m,3X,Y})$, and for all $i \in [n]$, $(\mathsf{ek}_i', \mathsf{msk}_i') \leftarrow \mathsf{Setup}'(1^\lambda, \mathsf{gpk}', F_{\mathsf{IP}}^{m,3X,Y})$. It sends $\mathsf{pk} := (\mathsf{pk}^{\mathsf{ot}}, \{\mathsf{ek}_i'\}_{i\in[n]})$ to $\mathcal{A}$.

-Simulation of $\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}))$:

If $j = 1$, that is, the first query for slot $i \in [n]$, then $\mathcal{B}_1$ queries its own encryption oracle to get $\mathbf{w}_i^1 := \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}_i^{\mathsf{ot}}, \mathbf{x}_i^{1,\beta})$, where $\beta \in \{0,1\}$, depending on the experiment $\mathcal{B}_1$ is interacting with. If $j > 1$, $\mathcal{B}_1$ uses the fact that the $\mathcal{MIFE}^{\mathsf{ot}}$ from Figure 5.1 is linearly homomorphic (see Remark 13) to generate all the remaining $\mathbf{w}_i^j := \mathbf{w}_i^1 + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} \bmod L = \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}_i^{\mathsf{ot}}, \mathbf{x}_i^{j,0} + \mathbf{x}_i^{1,\beta} - \mathbf{x}_i^0)$, which corresponds to the challenge ciphertexts in game G_β. Finally, $\mathcal{B}_1$ returns $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}_i', \mathbf{w}_i^j)$ to $\mathcal{A}$.

-Simulation of $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:

$\mathcal{B}_1$ uses its own secret key generation oracle on input $\mathbf{y}_1 \| \cdots \| \mathbf{y}_n$ to get $z := \mathsf{KeyGen}^{\mathsf{ot}}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$. For all $i \in [n]$, it computes $\mathsf{dk}_i' := \mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}_i', \mathbf{y}_i)$. It sends $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\{\mathsf{dk}_i'\}_{i\in[n]}, z)$ to $\mathcal{A}$.

Finally, $\mathcal{B}_1$ forwards the output α of $\mathcal{A}$ to its own experiment. It is clear that for all $\beta \in \{0,1\}$, when $\mathcal{B}_1$ interacts with $\mathbf{many\text{-}AD\text{-}IND\text{-}weak}_\beta^{\mathcal{MIFE}^{\mathsf{ot}}}$, it simulates the game G_β to $\mathcal{A}$. Therefore,

$$\begin{aligned}
&\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MIFE}^{\mathsf{ot}},\mathcal{B}_1}(\lambda) = \\
&\big|\Pr\left[\mathbf{many\text{-}AD\text{-}IND\text{-}weak}_0^{\mathcal{MIFE}^{\mathsf{ot}}}(1^\lambda, \mathcal{B}_1) = 1\right] \\
&- \Pr\left[\mathbf{many\text{-}AD\text{-}IND\text{-}weak}_1^{\mathcal{MIFE}^{\mathsf{ot}}}(1^\lambda, \mathcal{B}_1) = 1\right]\big| = \\
&|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})|.
\end{aligned}$$

□

Lemma 39: Game G_1 to G_2

There exists a PPT adversary $\mathcal{B}_2$ such that

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda).$$

Proof of Lemma 39. In Game G_2, we replace $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) \| \mathbf{z}_i)$ with $\mathsf{Enc}(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}) \| \mathbf{z}_i) = \mathsf{Enc}(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{x}_i^{j,1} \| \mathbf{z}_i)$, for all $i \in [n], j \in [Q_i]$. This follows from the many-AD-IND security of $\mathcal{FE}$ for n instances, which we can use since for each key query $\mathbf{y}_1 \| \dots \| \mathbf{y}_n$ and all $\mathbf{r}$, $\mathbf{z}$, we have

$$\begin{aligned}\langle \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}^{\mathsf{ot}}, \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i\rangle &= \langle \mathbf{u}_i + \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i\rangle \\ &= \langle \mathbf{u}_i + \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i\rangle \\ &= \langle \mathsf{Enc}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{ek}^{\mathsf{ot}}, \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i\rangle\end{aligned}$$

The second equality is equivalent to $\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i\rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i\rangle$, which follows from the restriction imposed by the security game (see Remark 7).

We build a PPT adversary $\mathcal{B}_2$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda).$$

Adversary $\mathcal{B}_2$ proceeds as follows.

-Simulation of pk:

Adversary $\mathcal{B}_2$ receives $(\mathsf{gpk}', \{\mathsf{ek}'_i\}_{i\in[n]}$ from its experiment. Then, it samples $(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \{\mathsf{ek}_i^{\mathsf{ot}}\}_{i\in[n]}) \leftarrow \mathsf{Setup}^{\mathsf{ot}}(1^\lambda, F_n^{m,X,Y})$, and sends $\mathsf{pk} := (\mathsf{pk}^{\mathsf{ot}}, \mathsf{gpk}', \{\mathsf{ek}'_i\}_{i\in[n]})$, to $\mathcal{A}$.

-Simulation of $\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}))$:

For all $b \in \{0,1\}$, $\mathcal{B}_1$ computes $\mathbf{w}_i^{j,b} := \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,b} - \mathbf{x}_i^{1,b}$, and queries its own encryption oracle on input $(i, \mathbf{w}_i^{j,0}, \mathbf{w}_i^{j,1})$, to get $\mathsf{Enc}'(\mathsf{gpk}', \mathsf{ek}'_i, \mathbf{w}_i^{j,\beta})$, which it forwards to $\mathcal{A}$, where $\beta \in \{0,1\}$, depending on the experiment $\mathcal{B}_2$ is interacting with.

-Simulation of $\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:

for all $i \in [n]$, $\mathcal{B}_1$ uses its own decryption key generation oracle on input $\mathbf{y}_i$ to get $\mathsf{dk}'_i := \mathsf{KeyGen}'(\mathsf{gpk}', \mathsf{msk}'_i, \mathbf{y}_i)$. It computes $z := \mathsf{KeyGen}^{\mathsf{ot}}(\mathsf{pk}^{\mathsf{ot}}, \mathsf{msk}^{\mathsf{ot}}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$, which it can do since it knows $\mathsf{msk}^{\mathsf{ot}}$. It sends $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\{\mathsf{dk}'_i\}_{i\in[n]}, z)$ to $\mathcal{A}$.

-Simulation of $\mathsf{OCorrupt}(i)$:

$\mathcal{B}_1$ returns $\mathsf{ek}_i^{\mathsf{ot}}$ to $\mathcal{A}$.

Finally, $\mathcal{B}_2$ forwards the outputs α of $\mathcal{A}$ to its own experiment. It is clear that for all $\beta \in \{0,1\}$, when $\mathcal{B}_2$ interacts with $\mathbf{many\text{-}AD\text{-}IND}_\beta^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{B}_2)$, it simulates the game $\mathsf{G}_{1+\beta}$ to $\mathcal{A}$. Therefore,

$$\begin{aligned}&\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{FE},\mathcal{B}_2,n}(\lambda) = \\ &\left|\Pr\left[\mathbf{many\text{-}AD\text{-}IND}_0^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{B}_2) = 1\right] - \Pr\left[\mathbf{many\text{-}AD\text{-}IND}_1^{\mathcal{FE}}(1^\lambda, 1^n, \mathcal{B}_2) = 1\right]\right| = \\ &|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})|.\end{aligned}$$

□

Concrete instances of FE for Inner Product

In this section we discuss three instantiations of our generic construction from Section 5.1.3. In particular, we show that the existing (single-input) public-key FE schemes proposed by [ALS16] (that are proven many-AD-IND-secure) satisfy Property 1 (two-step decryption) and Property 2 (linear encryption). These schemes are presented Section 2.6, recalled here for completeness.

Inner Product FE from MDDH

Here we present the FE for bounded norm inner product from [ALS16, Section 3], generalized to the $\mathcal{D}_k(p)$-MDDH setting, as in [AGRW17, Figure 15]. It handles the following functionality $F_{\mathsf{ip}}^{m,X,Y} : \mathcal{K} \times \mathcal{X} \to \mathcal{Z}$, with $\mathcal{X} := [0, X]^m$, $\mathcal{K} := [0, Y]^m$, $\mathcal{Z} := \mathbb{Z}$, and for all $\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$, we have:

$$F_{\mathsf{ip}}^{m}(\mathbf{y}, \mathbf{x}) = \langle \mathbf{x}, \mathbf{y} \rangle.$$

In [ALS16], it was proven many-AD-IND secure under the DDH assumption. In Section 2.6.1, we extend the many-AD-IND security proof from [AGRW17] to the multi-instance setting. We also show in this section that it satisfies Property 1 (two-step decryption) and Property 2 (linear encryption).

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow \mathsf{GGen}(1^\lambda)$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathsf{gpk} := (\mathcal{G}, [\mathbf{A}])$
Return gpk

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathbf{W} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m \times (k+1)}$, $\mathsf{ek} := [\mathbf{WA}]$, $\mathsf{msk} := \mathbf{W}$
Return $(\mathsf{ek}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x})}$:
$\mathbf{r} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
Return $\begin{bmatrix} -\mathbf{Ar} \\ \mathbf{x} + \mathbf{WAr} \end{bmatrix} \in \mathbb{G}^{k+m+1}$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \mathbf{y})}$:
Return $\begin{pmatrix} \mathbf{W}^\top \mathbf{y} \\ \mathbf{y} \end{pmatrix} \in \mathbb{Z}_p^{k+m+1}$

$\underline{\mathsf{Dec}(\mathsf{gpk}, [\mathbf{c}], \mathbf{d})}$:
$C := [\mathbf{c}^\top \mathbf{d}]$
Return $\log(C)$

Figure 5.7: $\mathcal{FE}$, a functional encryption scheme for the functionality $F_{\mathsf{IP}}^{m,X,Y}$, whose many-AD-IND security is based on the $\mathcal{D}_k(p)$-MDDH assumption.

Proof of Property 1 (two-step decryption).

1. The algorithm $\mathsf{GSetup}^\star(1^\lambda, F_{\mathsf{IP}}^{m,X,Y}, 1^n)$ works the same as GSetup except that it additionally uses n to ensure that $n \cdot m \cdot X \cdot Y = \mathsf{poly}(\lambda)$ (which implies $n \cdot m \cdot X \cdot Y < p$). Also, it returns the bound $B := 0$, $L := p$, $\mathbb{G}$ as the same group of order p generated by $\mathsf{GGen}(1^\lambda)$, and the encoding function $\mathcal{E} : \mathbb{Z}_p \times \mathbb{Z} \to \mathbb{G}$ defined for all $\gamma \in \mathbb{Z}_p, \mathsf{noise} \in \mathbb{Z}$ as

$$\mathcal{E}(\gamma, \mathsf{noise}) := [\gamma].$$

We let Dec_1 and Dec_2 be the first and second line of Dec in Figure 5.7 respectively.

2. We have for all $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^m$,

$$\mathsf{Dec}_1(\mathsf{dk}_\mathbf{y}, \mathsf{ct}_\mathbf{x} := [\mathbf{c}]) := [\mathbf{c}]^\top \mathsf{dk}_\mathbf{y} = [\langle \mathbf{x}, \mathbf{y} \rangle] = \mathcal{E}(\langle \mathbf{x}, \mathbf{y} \rangle \bmod p, 0).$$

3. It is straightforward to see that $\mathcal{E}(\gamma, 0)$ is efficiently and publicly computable.

4. It is also easy to see that $\mathcal{E}$ is linear.

5. Finally, for all $\gamma \in \mathbb{Z}$ such that $\gamma < n \cdot m \cdot X \cdot Y$,

$$\mathsf{Dec}_2(\mathcal{E}(\gamma \bmod p, 0)) := \log([\gamma \bmod p]) = \gamma \bmod p = \gamma,$$

where the log can be computed efficiently since $\gamma < n \cdot m \cdot X \cdot Y$ is assumed to lie in a polynomial size range.

Proof of Property 2 (linear encryption).

For all $\mathbf{x}' \in \mathbb{Z}^m$ and $[\mathbf{c}] \in \mathbb{G}^{m+k+1}$, let $\mathsf{Add}([\mathbf{c}], \mathbf{x}') := [\mathbf{c}] + \begin{bmatrix} \mathbf{0} \\ \mathbf{x}' \end{bmatrix}$. Then, for all $\mathbf{x}, \mathbf{x}' \in \mathbb{Z}^m$, and $[\mathbf{c}] := \mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x}) = \begin{bmatrix} -\mathbf{Ar} \\ \mathbf{x} + \mathbf{WAr} \end{bmatrix}$, we have:

$$\mathsf{Add}([\mathbf{c}], \mathbf{x}') = [\mathbf{c}] + \begin{bmatrix} \mathbf{0} \\ \mathbf{x}' \end{bmatrix} = \begin{bmatrix} -\mathbf{Ar} \\ \mathbf{x} + \mathbf{x}' + \mathbf{WAr} \end{bmatrix} = \mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, (\mathbf{x} + \mathbf{x}' \bmod p)).$$

Inner Product FE from LWE

Here we show that the many-AD-IND secure Inner Product FE from [ALS16, Section 4.1] and recalled in Figure 5.8, satisfies Property 1 (two-step decryption) and Property 2 (linear encryption).

Property 1 (two-step decryption).

1. The algorithm $\mathsf{GSetup}^\star(1^\lambda, F_{\mathsf{IP}}^{m,X,Y}, 1^n)$ works the same as Setup except that it uses n to set $K := n \cdot m \cdot X \cdot Y$, and it also returns the bound $B := \lfloor \frac{q}{K} \rfloor$, $L := q$, $\mathbb{G} := (\mathbb{Z}_q, +)$, and the encoding function $\mathcal{E} : \mathbb{Z}_q \times \mathbb{Z} \to \mathbb{G}$ defined for all $\gamma \in \mathbb{Z}_q, \mathsf{noise} \in \mathbb{Z}$ as

$$\mathcal{E}(\gamma \bmod q, \mathsf{noise}) := \gamma \cdot \left\lfloor \frac{q}{K} \right\rfloor + \mathsf{noise} \bmod q.$$

Also, parameters M, q, α and distribution $\mathcal{D}$ are chosen as explained in Section 2.6.2, as if working with input vectors of dimension $n \cdot m$.

We let Dec_1 and Dec_2 be the first and second line of Dec in Figure 5.8 respectively.

2. We have for all $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^m$,

$$\begin{aligned} \mathsf{Dec}_1(\mathsf{dk}_\mathbf{y}, \mathsf{ct}_\mathbf{x} := (\mathbf{c}_0, \mathbf{c}_1)) &= \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \end{pmatrix}^\top \mathsf{dk}_\mathbf{y} \bmod q \\ &= \langle \mathbf{x}, \mathbf{y} \rangle \cdot \left\lfloor \frac{q}{K} \right\rfloor + \mathbf{y}^\top \mathbf{e}_1 - \mathbf{e}_0^\top \mathbf{Z}^\top \mathbf{y} \bmod q \\ &= \mathcal{E}(\langle \mathbf{x}, \mathbf{y} \rangle \bmod q, \mathsf{noise}), \end{aligned}$$

where $\mathsf{noise} := \mathbf{y}^\top \mathbf{e}_1 - \mathbf{e}_0^\top \mathbf{Z}^\top \mathbf{y}$, and $\Pr[\mathsf{noise} < B] = 1 - \mathsf{negl}(\lambda)$.

3. It is straightforward to see that $\mathcal{E}(\gamma, 0)$ is efficiently and publicly computable.

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})}$:
Let integers $M, q \geq 2$, real $\alpha \in (0,1)$, and distribution $\mathcal{D}$ over $\mathbb{Z}^{m\times M}$ chosen as explained in Section 2.6.2; $K := m \cdot X \cdot Y$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{M\times\lambda}$, $\mathsf{gpk} := (K, \mathbf{A})$.
Return gpk

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathbf{Z} \leftarrow_{\mathrm{R}} \mathcal{D}$, $\mathbf{U} := \mathbf{ZA} \in \mathbb{Z}_q^{m\times\lambda}$, $\mathsf{ek} := \mathbf{U}$, $\mathsf{msk} := \mathbf{Z}$.
Return $(\mathsf{ek}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x} \in \mathbb{Z}^m)}$:
$\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^\lambda$, $\mathbf{e}_0 \leftarrow_{\mathrm{R}} D_{\mathbb{Z},\alpha q}^M$, $\mathbf{e}_1 \leftarrow_{\mathrm{R}} D_{\mathbb{Z},\alpha q}^m$
$\mathbf{c}_0 := \mathbf{As} + \mathbf{e}_0 \in \mathbb{Z}_q^M$
$\mathbf{c}_1 := \mathbf{Us} + \mathbf{e}_1 + \mathbf{x} \cdot \lfloor \frac{q}{K} \rfloor \in \mathbb{Z}_q^m$
Return $\mathsf{ct}_{\mathbf{x}} := (\mathbf{c}_0, \mathbf{c}_1)$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \mathbf{y} \in \mathbb{Z}^m)}$:
Return $\mathsf{dk}_{\mathbf{y}} := \begin{pmatrix} \mathbf{Z}^\top \mathbf{y} \\ \mathbf{y} \end{pmatrix} \in \mathbb{Z}^{M+m}$

$\underline{\mathsf{Dec}(\mathsf{gpk}, \mathsf{dk}_{\mathbf{y}}, \mathsf{ct}_{\mathbf{x}})}$:
$\mu' := \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \end{pmatrix}^\top \mathsf{dk}_{\mathbf{y}} \bmod q$.
Return $\mu \in \{-K+1, \ldots, K-1\}$ that minimizes $\left| \lfloor \frac{q}{K} \rfloor \mu - \mu' \right|$.

Figure 5.8: Functional encryption scheme for the class $F_{\mathsf{IP}}^{m,X,Y}$, based on the LWE assumption.

4. It is also easy to see that $\mathcal{E}$ is linear.

5. Finally, for all $\gamma \in \mathbb{Z}$ such that $\gamma < n \cdot m \cdot X \cdot Y$, and $\mathsf{noise} < n \cdot B$,

$$\mathsf{Dec}_2(\mathcal{E}(\gamma \bmod q, \mathsf{noise})) = \gamma,$$

follows by the same decryption correctness argument in [ALS16], with the only difference that here we used a larger bound K.

Property 2 (linear encryption). For all $\mathbf{x}' \in \mathbb{Z}^m$ and $(\mathbf{c}_0, \mathbf{c}_1) \in \mathbb{Z}_q^{M+m}$, let $\mathsf{Add}((\mathbf{c}_0, \mathbf{c}_1), \mathbf{x}') := (\mathbf{c}_0, \mathbf{c}_1) + (0, \mathbf{x}' \cdot \lfloor \frac{q}{K} \rfloor) \bmod q$. Then, for all $\mathbf{x}, \mathbf{x}' \in \mathbb{Z}^m$, and $(\mathbf{c}_0, \mathbf{c}_1) := (\mathbf{As} + \mathbf{e}_0, \mathbf{Us} + \mathbf{e}_1 + \mathbf{x} \cdot \lfloor \frac{q}{K} \rfloor) \in \mathbb{Z}_q^{M+m}$, we have:

$$\mathsf{Add}((\mathbf{c}_0, \mathbf{c}_1), \mathbf{x}') = (\mathbf{As} + \mathbf{e}_0, \mathbf{Us} + \mathbf{e}_1 + (\mathbf{x} + \mathbf{x}') \cdot \left\lfloor \frac{q}{K} \right\rfloor) \bmod q = \mathsf{Enc}(\mathsf{mpk}, (\mathbf{x} + \mathbf{x}' \bmod q)).$$

Inner Product FE from DCR

Here we show that the Inner Product FE from [ALS16, Section 5.1] and recalled in Figure 5.9 satisfies Property 1 (two-step decryption) and Property 2 (linear encryption).

Property 1 (two-step decryption).

1. The algorithm $\mathsf{GSetup}^\star(1^\lambda, F_{\mathsf{IP}}^{m,X,Y}, 1^n)$ works the same as Setup except that it additionally uses n to ensure $n \cdot m \cdot X \cdot Y < N$. Also, it returns the bound $B := 0$, $L := N$,

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})}$:
Choose primes $p = 2p' + 1$, $q = q' + 1$ with prime $p', q' > 2^{l(\lambda)}$ for an $l(\lambda) = \mathsf{poly}(\lambda)$ such that factoring is λ-hard, and set $N := pq$ ensuring that $m \cdot X \cdot Y < N$. Sample $g' \leftarrow_{\mathrm{R}} \mathbb{Z}_{N^2}^*$, $g := g'^{2N} \bmod N^2$.
Return $\mathsf{gpk} := (N, g)$

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})}$:
$\mathbf{s} \leftarrow_{\mathrm{R}} D_{\mathbb{Z}^m,\sigma}$, for standard deviation $\sigma > \sqrt{\lambda} \cdot N^{5/2}$, and for all $j \in [m]$, $h_j := g^{s_j} \bmod N^2$.
$\mathsf{ek} := \{h_j\}_{j\in[m]}$, $\mathsf{msk} := \{s_j\}_{j\in[m]}$
Return $(\mathsf{ek}, \mathsf{msk})$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, \mathbf{x} \in \mathbb{Z}^m)}$:
$r \leftarrow_{\mathrm{R}} \{0, \ldots, \lfloor N/4 \rfloor\}$, $C_0 := g^r \in \mathbb{Z}_{N^2}$, for all $j \in [m]$, $C_j := (1 + x_j N) \cdot h_j^r \in \mathbb{Z}_{N^2}$
Return $\mathsf{ct}_{\mathbf{x}} := (C_0, \ldots, C_m) \in \mathbb{Z}_{N^2}^{m+1}$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \mathbf{y} \in \mathbb{Z}^m)}$:
$d := \sum_{j\in[m]} y_j s_j \in \mathbb{Z}$.
Return $\mathsf{sk}_{\mathbf{y}} := (d, \mathbf{y})$

$\underline{\mathsf{Dec}\big(\mathsf{gpk}, \mathsf{sk}_{\mathbf{y}} := (d, \mathbf{y}), \mathsf{ct}_{\mathbf{x}}\big)}$:
$C := \left(\prod_{j\in[m]} C_j^{y_j}\right) \cdot C_0^{-d} \bmod N^2$.
Return $\log_{(1+N)}(C) := \frac{C-1 \bmod N^2}{N}$.

Figure 5.9: Functional encryption scheme for the class $F_{\mathsf{IP}}^{m,X,Y}$, based on the DCR assumption.

$\mathbb{G}$ as the subgroup of $\mathbb{Z}_{N^2}^*$ of order N generated by $(1 + N)$, and the encoding function $\mathcal{E} : \mathbb{Z}_N \times \mathbb{Z} \to \mathbb{G}$ defined for all $\gamma \in \mathbb{Z}_N, \mathsf{noise} \in \mathbb{Z}$ as

$$\mathcal{E}(\gamma, \mathsf{noise}) := 1 + \gamma \cdot N \bmod N^2.$$

We let Dec_1 and Dec_2 be the first and second line of Dec in Figure 5.9.

2. We have for all $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^m$,

$$\mathsf{Dec}_1(\mathsf{dk}_{\mathbf{y}} := (d, \mathbf{y}), \mathsf{ct}_{\mathbf{x}}) := \left(\prod_{j\in[m]} C_j^{y_j}\right) \cdot C_0^{-d} \bmod N^2 = \mathcal{E}(\langle \mathbf{x}, \mathbf{y}\rangle \bmod N, 0).$$

3. It is straightforward to see that see that $\mathcal{E}(\gamma, 0)$ can be efficiently computed from public information.

4. It is also easy to see that $\mathcal{E}$ is linear.

5. Finally, for all $\gamma \in \mathbb{Z}$ such that $\gamma \le n \cdot m \cdot X \cdot Y < N$, it holds

$$\mathsf{Dec}_2(\mathcal{E}(\gamma, 0)) := \frac{\mathcal{E}(\gamma, 0) - 1 \bmod N^2}{N} = \gamma.$$

Property 2 (linear encryption). For all $\mathbf{x}' \in \mathbb{Z}^m$ and $(C_0, C_1', \ldots, C_m') \in \mathbb{Z}_{N^2}^{m+1}$, let $\mathsf{Add}((C_0, C_1, \ldots, C_m), \mathbf{x}')$ computes $C_j' := C_j \cdot (1 + x_j' N) \bmod N^2$ for all $j \in [m]$ and outputs $(C_0, C_1', \ldots, C_m')$. Then, for all $\mathbf{x}, \mathbf{x}' \in \mathbb{Z}^m$, and $(C_0, C_1, \ldots, C_m) := (g^r, (1 + x_1 N) \cdot h_1^r, \ldots, (1 + x_m N) \cdot h_m^r) \in \mathbb{Z}_{N^2}^m$, we have:

$$\begin{aligned}\mathsf{Add}((C_0, C_1, \ldots, C_m), \mathbf{x}') &= (g^r, (1 + (x_1 + x_1')N) \cdot h_1^r \bmod N^2, \ldots \\ &\quad , (1 + (x_m + x_m')N) \cdot h_m^r \bmod N^2) \\ &= \mathsf{Enc}\big(\mathsf{mpk}, (\mathbf{x} + \mathbf{x}' \bmod N)\big).\end{aligned}$$

Chapter 6

Multi-Client Inner Product Functional Encryption

Overview of our construction.

We build the first MCFE for inner product from standard assumptions. Our construction goes in four steps. First, we build an MCFE for inner product that only satisfies a weak notion of security, namely, one-AD-IND-weak security (see Definition 51). That is, our scheme is only secure when there is only one challenge ciphertext per input slot $i \in [n]$ and label ℓ. Moreover, the security notion does not take into account the information that can be extracted from a partial decryption of ciphertexts. Recall that decryption usually operates on pk, msk, and ciphertexts ct_i for *all* slots $i \in [n]$. But it is still possible to extract information from ciphertexts ct_i for some, bot not all slots $i \in [n]$. The information on the underlying messages that is leaked by such partial decryption is not captured by the weak security notion. The security of this construction relies on the DDH assumption, in the random oracle model. This work has appeared in [CDG+18a].

Second, we show how to transform our one-AD-IND secure MCFE for inner product into a many-AD-IND secure MCFE, thereby allowing an adversary to obtain many challenge ciphertexts, using an extra layer of single-input FE for inner product.

Third, we show how to remove the aforementioned limitation in the security model, using a layer of secret sharing on top of the original MCFE. This layer ensures that given only ciphertexts ct_i for some, but not all input slots $i \in [n]$, one learns no information whatsoever on the underlying messages. This transformation is generic: it takes as input any MCFE with xx-AD-IND-weak security and turns it into an xx-AD-IND secure MCFE, where xx $\in$ {many,one}. It can also be seen as a decentralized version of *All-Or-Nothing Transforms* [Riv97, Boy99, CDH+00]. We propose a concrete instantiation in pairing-friendly groups, under the Decisional Bilinear Diffie-Hellman problem, in the random oracle model. When applied on our one-AD-IND-weak secure MCFE, we get an one-AD-IND secure MCFE.

Fourth, we propose an efficient decentralized algorithm to generate a sum of private inputs, which can convert our many-AD-IND secure MCFE for inner product into a decentralized many-AD-IND secure MCFE. This technique is inspired from [KDK11], and only applies to the functional decryption key generation algorithm, and so this is compatible with the two above conversions. We now expose our MCFE and SSE constructions in more details.

MCFE for inner product with one-AD-IND-weak security. We briefly showcase the techniques that allow us to build efficient MCFE for inner product. The schemes we introduce later enjoy adaptive security (aka full security), where encryption queries are made adaptively by the adversary against the security game, but for the sake of clarity, we will here give an informal description of a selectively-secure scheme from the DDH assumption, where queries

Scheme	MCFE	[ABDP15]
$\mathsf{Setup}:$	$\forall i \in [n]$: $\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^m$, $\mathsf{ek}_i := \mathbf{s}_i$	$\forall i \in [n]$: $\mathbf{s}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^m$, $\mathsf{ek}_i := [\mathbf{s}_i]$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i, \ell):$	$[r] := \mathsf{H}(\ell)$ return $[\mathbf{c}_i] := [\mathbf{x}_i + \mathbf{s}_i r]$	$r \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ return $([r], [\mathbf{c}_i] := [\mathbf{x}_i + \mathbf{s}_i r])$
$\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \mathbf{y}_1 \Vert \cdots \Vert \mathbf{y}_n):$	$d := \sum_i \mathbf{y}_i^\top \mathbf{s}_i$ returns $(\mathbf{y}_1 \Vert \cdots \Vert \mathbf{y}_n, d)$	$d := \sum_i \mathbf{y}_i^\top \mathbf{s}_i$ returns $(\mathbf{y}_1 \Vert \cdots \Vert \mathbf{y}_n, d)$
$\mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_{\mathbf{y}_1 \Vert \cdots \Vert \mathbf{y}_n}, \mathsf{ct}_1, \cdots, \mathsf{ct}_n, \ell):$	Discrete logarithm of $\sum_i [\mathbf{c}_i^\top \mathbf{y}_i] - [r \cdot d]$ where $[r] := \mathsf{H}(\ell)$	Discrete logarithm of $\sum_i [\mathbf{c}_i^\top \mathbf{y}_i] - [r \cdot d]$

Figure 6.1: Comparison of the Inner-Product FE scheme from [ABDP15] and a similar MCFE obtained by introducing a hash function H.

are made beforehand. Namely, the standard security notion for FE is indistinguishability-based, where the adversary has access to a encryption oracle, that on input (m_0, m_1) either always encrypts m_0 or always encrypts m_1. While for the adaptive security, the adversary can query this oracle adaptively, in the *selective* setting, all queries are made at the beginning, before seeing the public parameters.

We first design a secret-key MCFE scheme building up from the public-key FE scheme introduced by [ABDP15] (itself a selectively-secure scheme) where we replace the global randomness with a hash function (modeled as a random oracle for the security analysis), in order to make the generation of the ciphertexts independent for each client. The comparison is illustrated in Figure 6.1. Note that for the final decryption to be possible, one needs the function evaluation to be small enough, within this discrete logarithm setting. This is one limitation, which is still reasonable for real-world applications that use concrete numbers, that are not of cryptographic size.

Correctness then follows from:

$$\sum_i \mathbf{c}_i^\top \mathbf{y}_i - r \cdot d = \sum_i (\mathbf{x}_i + \mathbf{s}_i r)^\top \mathbf{y}_i - r \cdot \sum_i \mathbf{y}_i^\top \mathbf{s}_i = \sum_i \mathbf{x}_i^\top \mathbf{y}_i.$$

In [CDG+18a, Appendix B], this scheme is proven selectively secure under the DDH assumption. To obtain adaptive security, we adapt the adaptively secure inner product FE from [ALS16] in the same manner than described for the FE from [ABDP15].

Secret Sharing Encapsulation. AS explained, in order to deal with partial ciphertexts, we introduce a new tool, called *Secret Sharing Encapsulation* (SSE). In fact, the goal is to allow a user to recover the ciphertexts from the n senders only when he gets the contributions of all of them. At first glance, one may think this could be achieved by using *All-Or-Nothing Transforms* or (n, n)-*Secret Sharing*. However, these settings require an authority who operates on the original messages or generates the shares. Consequently, they are incompatible with our multi-client schemes. Our SSE tool can be seen as a decentralized version of *All-Or-Nothing Transforms* or of (n, n)-*Secret Sharing*: for each label ℓ, each user $i \in [n]$ can generate, on his own, the share $S_{\ell,i}$. And, unless all the shares $S_{i,\ell}$ have been generated, the encapsulated keys are random and perfectly mask all the inputs.

We believe that SSE could be used in other applications. As an example, AONT was used in some traitor tracing schemes [KY02, CPP05]. By using SSE instead of AONT, one can get *decentralized* traitor tracing schemes in which the tracing procedure can only be run if all the authorities agree on the importance of tracing a suspected decoder. This might be meaningful in practice to avoid the abuse of tracing, in particular on-line tracing, which might break the privacy of the users, in case the suspected decoders are eventually legitimate decoders.

MCFE with one-AD-IND-weak security

Here we present a multi-client scheme $\mathcal{MCFE}$ for inner product over $\mathbb{Z}$, that is, for the set of functionalities $\{F_n^{m,X,Y}\}_{n\in\mathbb{N}}$ defined as $F_n^{m,X,Y} : \mathcal{K}_n \times \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Z}$, with $\mathcal{K}_n := [0, Y]^{mn}$, for all $i \in [n]$, $\mathcal{X}_i := [0, X]^m$, $\mathcal{Z} := \mathbb{Z}$, such that for any $(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n) \in \mathcal{K}_n$, $\mathbf{x}_i \in \mathcal{X}_i$, we have:

$$F_n^{m,X,Y}\big((\mathbf{y}_1 \| \cdots \| \mathbf{y}_n), \mathbf{x}_1, \ldots, \mathbf{x}_n\big) = \sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{y}_i \rangle.$$

We prove its one-AD-IND-weak security under the $\mathcal{D}_k(p)$ in prime-order group (a particular case being the DDH assumption). Note that we do not require pairing-friendly groups. As explained in the introduction of this chapter, this scheme will be used to build many-AD-IND secure MCFE for inner product. The scheme is described in Figure 6.2.

$\underline{\mathsf{Setup}(1^\lambda, F_n^{m,X,Y})}$:
$\mathcal{G} := (\mathbb{G}, p, P) \leftarrow \mathsf{GGen}(1^\lambda)$, $\mathsf{H} : \{0,1\}^* \to \mathbb{G}^{k+1}$ be a full domain hash function modeled as a random oracle.
For all $i \in [n]$, $\mathbf{S}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, $\mathsf{ek}_i := \mathbf{S}_i$, $\mathsf{pk} = \mathcal{G}$, $\mathsf{msk} := \{\mathbf{S}_i\}_{i\in[n]}$.
Return $(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]})$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i, \ell)}$:
Compute $[\mathbf{r}] := \mathsf{H}(\ell)$.
Return $[\mathbf{c}_i] := [\mathbf{x}_i + \mathbf{S}_i\mathbf{r}]$.

$\underline{\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n))}$:
$\mathbf{d} := \sum_{i\in[n]} \mathbf{S}_i^\top \mathbf{y}_i$
Return $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n, \mathbf{d})$.

$\underline{\mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n}, [\mathbf{c}_1], \ldots, [\mathbf{c}_n], \ell)}$:
Parse $\mathsf{dk}_{\mathbf{y}_1\|\cdots\|\mathbf{y}_n} := (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n, \mathbf{d})$.
Compute $[\mathbf{r}] := \mathsf{H}(\ell)$.
Return the discrete log of $\sum_{i=1}^n [\mathbf{c}_i^\top \mathbf{y}_i] - [\mathbf{r}^\top \mathbf{d}]$.

Figure 6.2: Private-key, one-AD-IND-weak secure, multi-client FE scheme $\mathcal{MCFE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$ for the class $F_n^{m,X,Y}$, one-AD-IND-weak secure under the $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}$.

Correctness of $\mathcal{MCFE}$ follows from:

$$\sum_i [\mathbf{c}_i^\top \mathbf{y}_i] - [\mathbf{r}^\top \mathbf{d}] = \sum_i [(\mathbf{x}_i + \mathbf{S}_i\mathbf{r})^\top \mathbf{y}_i] - [\mathbf{r}^\top \sum_i \mathbf{S}_i^\top \mathbf{y}_i] = \sum_i [\mathbf{x}_i^\top \mathbf{y}_i].$$

We know $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \le n \cdot m \cdot X \cdot Y$, which is bounded by a polynomial in the security parameter. Thus, decryption can efficiently recover the discrete log: $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \bmod p = \sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle$, where the equality holds since $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \le n \cdot m \cdot X \cdot Y \ll p$.

Theorem 15: one-AD-IND-weak security

The scheme $\mathcal{MCFE}$ from Figure 6.2 is one-AD-IND-weak secure assuming the $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}$, in the random oracle model.

Proof of Theorem 15. We proceed via a series of games G_i for $i \in \{0, \ldots, 2\}$, described in Figure 6.3. The transitions are summarized in Figure 5.5. Let $\mathcal{A}$ be a PPT adversary. For any game G, we denote by $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ the probability that the game G outputs 1 when interacting with $\mathcal{A}$.

According to Definition 21, we have:

$$\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE},\mathcal{A}}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A})|.$$

Games G_0, G_1, $\boxed{\mathsf{G}_2, (\mathsf{G}_{3.q.1})_{q\in[Q+1]}, (\mathsf{G}_{3.q.2}, \mathsf{G}_{3.q.3})_{q\in[Q]}}$, G_4

$\mathcal{G} := (\mathbb{G}, p, g) \leftarrow \mathsf{GGen}(1^\lambda)$, $\mathsf{pk} := \mathcal{G}$. For all $i \in [n]$, $\mathbf{S}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$, $\mathsf{ek}_i := \mathbf{S}_i$, $\mathsf{msk} := \{\mathbf{S}_i\}_i$.
$\boxed{\mathbf{A} \leftarrow \mathcal{D}_k(p),\ \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \{\mathbf{0}\} \text{ s.t. } \mathbf{A}^\top\mathbf{a}^\perp = 0}$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{OKeygen}(\cdot),\mathsf{OCorrupt}(\cdot),\mathsf{RO}(\cdot)}(\mathsf{pk})$.
Return α if **Condition 1** and **Extra condition** from Definition 25 of one-AD-IND-weak security are satisfied, 0 otherwise.

$\underline{\mathsf{RO}(\ell)}$: // G_0, $\lceil\mathsf{G}_1\rceil$, $\boxed{\mathsf{G}_2, \mathsf{G}_{3.q.1}, \mathsf{G}_{3.q.2}, \mathsf{G}_{3.q.3}}$, G_4

$[\mathbf{u}_\ell] := \mathsf{H}(\ell)$, $\lceil[\mathbf{u}_\ell] := \mathsf{RF}(\ell)\rceil$, $\boxed{[\mathbf{u}_\ell] := [\mathbf{A}\cdot\mathbf{r}_\ell], \text{ with } \mathbf{r}_\ell := \mathsf{RF}'(\ell)}$
On the q'th (fresh) query: $[\mathbf{u}_\ell] := \mathbf{A}\cdot\mathsf{RF}'(\ell) + \mathsf{RF}''(\ell)\cdot\mathbf{a}^\perp$
Return $[\mathbf{u}_\ell]$.

$\underline{\mathsf{OEnc}(i, (\mathbf{x}_i^0, \mathbf{x}_i^1), \ell)}$: // G_0, G_1, G_2, $\boxed{\mathsf{G}_{3.q.1}, \mathsf{G}_{3.q.2}, \mathsf{G}_{3.q.3}}$, $\lceil\mathsf{G}_4\rceil$

$[\mathbf{u}_\ell] := \mathsf{RO}(\ell)$,
$[\mathbf{c}_i] := [\mathbf{x}_i^0 + \mathbf{S}_i\mathbf{u}_\ell]$
$\boxed{\text{If } [\mathbf{u}_\ell] \text{ is computed on the } j\text{-th } \mathsf{RO}\text{-query, for } j < q\text{: } [\mathbf{c}_i] := [\mathbf{x}_i^1 + \mathbf{S}_i\mathbf{u}_\ell]}$
If $[\mathbf{u}_\ell]$ is computed on the q-th RO-query: $[\mathbf{c}_i] := [\mathbf{x}_i^1 + \mathbf{S}_i\mathbf{u}_\ell]$
$\lceil[\mathbf{c}_i] := [\mathbf{x}_i^1 + \mathbf{S}_i\mathbf{u}_\ell]\rceil$
Return $[\mathbf{c}_i]$

$\underline{\mathsf{OKeygen}(\mathbf{y})}$: Return $\sum_i \mathbf{S}_i^\top\mathbf{y}_i$. //$\mathsf{G}_0$, G_1, G_2, $\mathsf{G}_{3.q.1}$, $\mathsf{G}_{3.q.2}$, $\mathsf{G}_{3.q.3}$, G_4

$\underline{\mathsf{OCorrupt}(i)}$: Return $\mathbf{S}_i$. // G_0, G_1, G_2, $\mathsf{G}_{3.q.1}$, $\mathsf{G}_{3.q.2}$, $\mathsf{G}_{3.q.3}$, G_4

Figure 6.3: Games for the proof of Theorem 15. Here, RF, RF', RF'' are random functions onto $\mathbb{G}^{k+1}$, $\mathbb{Z}_p^k$, and $\mathbb{Z}_p^*$, respectively, that are computed on the fly. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame. Note that $\mathcal{A}$'s queries must satisfy the condition from Definition 25, including the extra condition, since we are only proving one-AD-IND-*weak* security.

Game G_1: we replace the hash function H by a truly random function onto $\mathbb{G}^2$, that is computed on the fly. This uses the pseudorandomness of the hash function H. Namely, in the Random Oracle Model:

$$\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}).$$

Game G_2: here, the outputs of RO are uniformly random in the span of $[\mathbf{A}]$ for $\mathbf{A} \leftarrow \mathcal{D}_k(p)$. This uses the Q-fold $\mathcal{D}_k(p)$-MDDH assumption, where Q is the number of call to $\mathsf{RO}(\cdot)$, which tightly reduces to its 1-fold variant, using the random-self reducibility (see Lemma 1). Namely, there exists a PPT adversary $\mathcal{B}$ such that

$$\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}) \le \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda) + \frac{1}{p-1}.$$

Note that we use the fact that the **Condition 1** and **Extra condition** from Definition 25 of one-AD-IND-weak security are efficiently checkable. This allows adversary $\mathcal{B}$ to decide efficiently whether to forward the output α of $\mathcal{A}$, or 0 (in case the conditions are not satisfied) to its own experiment.

Game $\mathsf{G}_{3.1.1}$: is exactly game G_2. Thus,

$$\mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{3.1.1}}(\mathcal{A}).$$

From game $\mathsf{G}_{3.q.1}$ to game $\mathsf{G}_{3.q.2}$: we first change the distribution of the output of RO on its q'th query (note that two queries with the same input are counted once, that is, we only count *fresh* queries), from uniformly random in the span of $[\mathbf{A}]$ to uniformly random over $\mathbb{G}^{k+1}$, using the $\mathcal{D}_k(p)$-MDDH assumption. Then, we use the basis $(\mathbf{A}\|\mathbf{a}^\perp)$ of $\mathbb{Z}_p^{k+1}$, to write a uniformly random vector over $\mathbb{Z}_p^{k+1}$ as $\mathbf{A}\mathbf{u}_1 + u_2 \cdot \mathbf{a}^\perp$, where $\mathbf{u}_1 \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $u_2 \leftarrow_{\mathrm{R}} \mathbb{Z}_p$. Finally, we switch to $\mathbf{A}\mathbf{u}_1 + u_2 \cdot \mathbf{a}^\perp$ where $\mathbf{u}_1 \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $u_2 \leftarrow_{\mathrm{R}} \mathbb{Z}_p^*$, which only changes the adversary view by a statistical distance of $1/p$. Thus, there exists a PPT adversary $\mathcal{B}_{3.q.1}$ such that

$$\mathsf{Adv}_{\mathsf{G}_{3.q.1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_{3.q.2}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.q.1}}(\lambda) + \frac{1}{p}.$$

Once again, we rely on the fact that **Condition 1** and **Extra condition** from Definition 25 of one-AD-IND-weak security are efficiently checkable.

From game $\mathsf{G}_{3.q.2}$ to game $\mathsf{G}_{3.q.3}$: We prove:

$$\mathsf{Adv}_{\mathsf{G}_{3.q.2}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{3.q.3}}(\mathcal{A}).$$

Note that if the output of the q'th fresh query to RO is not used by OEnc, then the games $\mathsf{G}_{3.q.2}$ and $\mathsf{G}_{3.q.3}$ are identical. We consider the case where the output of the q'th fresh query to RO is used by OEnc. We show that we also have $\mathsf{Adv}_{\mathsf{G}_{3.q.2}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}_{3.q.3}}(\mathcal{A})$ in that case, in two steps.

In Step 1, we show that for all PPT adversaries $\mathcal{B}_{3.q.2}$ and $\mathcal{B}^\star_{3.q.3}$, there exist PPT adversaries $\mathcal{B}^\star_{3.q.2}$ and $\mathcal{B}_{3.q.3}$ such that $\mathsf{Adv}_{\mathsf{G}_{3.q.2}}(\mathcal{B}_{3.q.2}) = (p^{2m}+1)^n \cdot \mathsf{Adv}_{\mathsf{G}^\star_{3.q.2}}(\mathcal{B}^\star_{3.q.2})$ and $\mathsf{Adv}_{\mathsf{G}_{3.q.3}}(\mathcal{B}_{3.q.3}) = (p^{2m}+1)^n \cdot \mathsf{Adv}_{\mathsf{G}^\star_{3.q.3}}(\mathcal{B}^\star_{3.q.3})$, where the games $\mathsf{G}^\star_{3.q.2}$ and $\mathsf{G}^\star_{3.q.3}$ are selective variants of games $\mathsf{G}_{3.q.2}$ and $\mathsf{G}_{3.q.3}$ respectively (see Figure 6.4). Note that those advantage are conditioned on the fact that the output of the q'th fresh query to RO is used by OEnc.

In Step 2, we show that for all PPT adversaries $\mathcal{B}^\star$, we have $\mathsf{Adv}_{\mathsf{G}^\star_{3.q.2}}(\mathcal{B}^\star) = \mathsf{Adv}_{\mathsf{G}^\star_{3.q.3}}(\mathcal{B}^\star)$, where again, these advantages are conditioned on the fact that the output of the q'th fresh query to RO is used by OEnc.

Step 1. We build a PPT adversary $\mathcal{B}^\star_{3.q.2}$ playing against $\mathsf{G}^\star_{3.q.2}$, such that $\mathsf{Adv}_{\mathsf{G}_{3.q.2}}(\mathcal{B}_{3.q.2}) = (p^{2m}+1)^n \cdot \mathsf{Adv}_{\mathsf{G}^\star_{3.q.2}}(\mathcal{B}^\star_{3.q.2})$.

Adversary $\mathcal{B}^\star_{3.q.2}$ first guesses for all $i \in [n]$, $z_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{2m} \cup \{\perp\}$, which it sends to its selective game $\mathsf{G}^\star_{3.q.2}$. That is, the guess z_i is either a pair of vectors $(\mathbf{x}_i^0, \mathbf{x}_i^1) \in \mathbb{Z}_p^{2m}$ queried to OEnc, or $\perp$, which means no query to OEnc. Then, it simulates $\mathcal{A}$'s view using its own oracles. When $\mathcal{B}^\star_{3.q.2}$ guesses successfully (call E that event), it simulates $\mathcal{B}_{3.q.2}$'s view exactly as in $\mathsf{G}_{3.q.2}$. Since event E happens with probability $(p^{2m}+1)^{-n}$, we obtain:

$$\begin{aligned}
&\mathsf{Adv}_{\mathsf{G}^\star_{3.q.2}}(\mathcal{B}^\star_{3.q.2}) \\
&= \Big| \underbrace{\Pr[1 \leftarrow \mathsf{G}^\star_{3.q.2} | E]}_{=\Pr[1 \leftarrow \mathsf{G}_{3.q.2}]} \cdot \Pr[E] + \underbrace{\Pr[1 \leftarrow \mathsf{G}^\star_{3.q.2} | \neg E]}_{=0} \cdot \Pr[\neg E] \\
&= \Pr[E] \cdot |\Pr[1 \leftarrow \mathsf{G}_{3.q.2}]| \\
&= (p^{2m}+1)^{-n} \cdot \mathsf{Adv}_{\mathsf{G}_{3.q.2}}(\mathcal{B}_{3.q.2})
\end{aligned}$$

Adversary $\mathcal{B}_{3.q.3}$ is built similarly. As for prior reductions, we use the fact that **Condition 1** and **Extra condition** from Definition 25 of one-AD-IND-weak security, and the validity of the guess $\{z_i\}_{i\in[n]}$, can be checked efficiently.

Step 2. We assume the values $(z_i)_{i\in[n]}$ sent by $\mathcal{B}^\star$ are consistent, that is, they don't make the game end and return 0. We also assume **Condition 1** and **Extra condition** from Definition 25 of one-AD-IND-weak security are satisfied. We call E this event.

We show that games $\mathsf{G}^\star_{3.q.2}$ and $\mathsf{G}^\star_{3.q.3}$ are identically distributed, conditioned on E. To prove so, we use the fact that the following are identically distributed: $(\mathbf{S}_i)_{i\in[n],z_i=(\mathbf{x}_i^0,\mathbf{x}_i^1)}$ and $\left(\mathbf{S}_i+\gamma(\mathbf{x}_i^1-\mathbf{x}_i^0)(\mathbf{a}^\perp)^\top\right)_{i\in[n],z_i=(\mathbf{x}_i^0,\mathbf{x}_i^1)}$, where $\mathbf{a}^\perp\leftarrow_{\mathrm{R}}\mathbb{Z}_p^{k+1}\setminus\{\mathbf{0}\}$ such that $\mathbf{A}^\top\mathbf{a}^\perp=\mathbf{0}$, and for all $i\in[n]$: $\mathbf{S}_i\leftarrow_{\mathrm{R}}\mathbb{Z}_p^{m\times(k+1)}$, and $\gamma\leftarrow_{\mathrm{R}}\mathbb{Z}_p$. This is true since the $\mathbf{S}_i$ are independent of the z_i (note that this is not true in adaptive games). Thus, we can re-write $\mathbf{S}_i$ into $\mathbf{S}_i+\gamma(\mathbf{x}_i^1-\mathbf{x}_i^0)(\mathbf{a}^\perp)^\top$ without changing the distribution of the game.

We now take a look at where the extra terms $\gamma(\mathbf{x}_i^1-\mathbf{x}_i^0)(\mathbf{a}^\perp)^\top$ actually appear in the adversary's view. They do not appear in the output of $\mathsf{OCorrupt}$, because we assume event E holds, which implies for all $i\in[n]$, either $z_i=\perp$, and there is no extra term; or $z_i=(\mathbf{x}_i^0,\mathbf{x}_i^1)$, but by **Condition 1**, we must have $\mathbf{x}_i^0=\mathbf{x}_i^1$, which means there is again no extra term.

They appear in $\mathsf{OKeygen}(\mathbf{y})$ as

$$\mathsf{dk}_\mathbf{y}=\sum_{i\in[n]}\mathbf{S}_i^\top\mathbf{y}_i+\boxed{\mathbf{a}^\perp\cdot\gamma\textstyle\sum_{i:z_i=(\mathbf{x}_i^0,\mathbf{x}_i^1)}(\mathbf{x}_i^1-\mathbf{x}_i^0)^\top\mathbf{y}_i},$$

where the gray term equals $\mathbf{0}$ by **Condition 1** and **Extra condition** from Definition 25 of one-AD-IND-weak security.

Finally, the extra terms $\gamma(\mathbf{x}_i^1-\mathbf{x}_i^0)(\mathbf{a}^\perp)^\top$ only appear in the output of the queries to OEnc which use $[\mathbf{u}_\ell]$ computed on the q'th query to RO, since for all others, the vector $[\mathbf{u}_\ell]$ lies in the span of $[\mathbf{A}]$, and $\mathbf{A}^\top\mathbf{a}^\perp=\mathbf{0}$. For the former, we have $[\mathbf{c}]:=[\mathbf{S}_i\mathbf{u}_\ell+\mathbf{x}_i^0+\gamma(\mathbf{x}_i^1-\mathbf{x}_i^0)(\mathbf{a}^\perp)^\top\mathbf{u}_\ell]$. Since $\mathbf{u}_\ell^\top\mathbf{a}^\perp\neq 0$, the above $[\mathbf{c}]$ is identically distributed to $[\mathbf{S}_i\mathbf{u}_\ell+\mathbf{x}_i^1+\gamma(\mathbf{x}_i^1-\mathbf{x}_i^0)(\mathbf{a}^\perp)^\top\mathbf{u}_\ell]$. Finally, reverting these statistically perfect changes, we obtain that $[\mathbf{c}]$ is identically distributed to $[\mathbf{S}_i\mathbf{u}_\ell+\mathbf{x}_i^1]$, as in game $\mathsf{G}^\star_{3.q.3}$.

Thus, when event E happens, the games are identically distributed. When $\neg E$ happens, the games both return 0. Thus, we have

$$\mathsf{Adv}_{\mathsf{G}^\star_{3.q.2}}(\mathcal{B}^\star)=\mathsf{Adv}_{\mathsf{G}^\star_{3.q.3}}(\mathcal{B}^\star).$$

From game $\mathsf{G}_{3.q.3}$ to game $\mathsf{G}_{3.q+1.1}$: this transition is the reverse of the transition from game $\mathsf{G}_{3.q.1}$ to game $\mathsf{G}_{3.q.2}$, namely, we use the $\mathcal{D}_k(p)$-MDDH assumption to switch back the distribution of $[\mathbf{u}_\ell]$ computed on the q'th (fresh) query to RO from uniformly random over $\mathbb{G}^{k+1}$ (conditioned on the fact that $\mathbf{u}_\ell^\top\mathbf{a}^\perp\neq 0$) to uniformly random in the span of $[\mathbf{A}]$. We obtain a PPT adversary $\mathcal{B}_{3.q.3}$ such that

$$\mathsf{Adv}_{\mathsf{G}_{3.q.3}}(\mathcal{A})-\mathsf{Adv}_{\mathsf{G}_{3.q+1.1}}(\mathcal{A})\leq\mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_{3.q.3}}(\lambda)+\frac{1}{p}.$$

From game $\mathsf{G}_{3.Q+1.1}$ to G_4: First, we switch the distribution of all the vectors $[\mathbf{u}_\ell]$ output by the random oracle to uniformly random over $\mathbb{G}^{k+1}$, using the $\mathcal{D}_k(p)$-MDDH simultaneously for all queried labels ℓ, using the random self reducibility of the MDDH assumption (cf Lemma 1). Then, we using the random oracle model to argue that the output of the real hash function H are distributed as the output of a truly random function computed on the fly (this is the reserve transition than transition from gma eG_0 to game G_1). We obtain a PPT adversary $\mathcal{B}_4$ such that:

$$\mathsf{Adv}_{\mathsf{G}_{3.Q+1.1}}(\mathcal{A})-\mathsf{G}_4\leq\mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}_4}(\lambda)+\frac{1}{p-1}.$$

Putting everything together, we obtain a PPT adversary $\mathcal{B}$ such that

$$\mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE},\mathcal{A}}(\lambda)\leq(2Q+2)\cdot\mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G},\mathcal{B}}(\lambda)+\frac{2Q}{p}+\frac{2}{p-1},$$

where Q denotes the number of calls to the random oracle. □

Games $(\mathsf{G}^{\star}_{3.q.2}, \mathsf{G}^{\star}_{3.q.3})_{q\in[Q]}$:
$\big(\mathsf{state}, (z_i \in \mathbb{Z}_p^{2m} \cup \{\bot\})_{i\in[n]}\big) \leftarrow \mathcal{A}(1^\lambda, 1^n)$
$\mathcal{S} := \emptyset$, $\mathcal{G} := (\mathbb{G}, p, P) \leftarrow \mathsf{GGen}(1^\lambda)$, $\mathsf{pk} := \mathcal{G}$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\mathbf{a}^\bot \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1} \setminus \{\mathbf{0}\}$ s.t. $\mathbf{A}^\top \mathbf{a}^\bot = \mathbf{0}$. For all $i \in [n]$, $\mathbf{S}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{m\times(k+1)}$.
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{OKeygen}(\cdot),\mathsf{OCorrupt}(\cdot),\mathsf{RO}(\cdot)}(\mathsf{pk}, \mathsf{state})$.
If $\exists i \in [n] \setminus \mathcal{S}$ such that $z_i \neq \bot$, the game ends, and returns 0.
Return α if **Condition 1** and **Extra condition** from Definition 25 of one-AD-IND-weak security are satisfied, 0 otherwise.

$\mathsf{RO}(\ell)$: // $\mathsf{G}^{\star}_{3.q.2}$, $\mathsf{G}^{\star}_{3.q.3}$
$[\mathbf{u}_\ell] := [\mathbf{A}\mathbf{r}_\ell]$, with $\mathbf{r}_\ell := \mathsf{RF}'(\ell)$
On the q'th (fresh) query: $[\mathbf{u}_\ell] := [\mathbf{A} \cdot \mathsf{RF}'(\ell) + \mathsf{RF}''(\ell) \cdot \mathbf{a}^\bot]$
Return $[\mathbf{u}_\ell]$.

$\mathsf{OEnc}(i, (\mathbf{x}^0, \mathbf{x}^1), \ell)$: // $\boxed{\mathsf{G}^{\star}_{3.q.2}}$, $\mathsf{G}^{\star}_{3.q.3}$
$[\mathbf{u}_\ell] := \mathsf{RO}(\ell)$,
$[\mathbf{c}] := [\mathbf{x}^0 + \mathbf{S}_i \mathbf{u}_\ell]$
If $[\mathbf{u}_\ell]$ is computed on the j'th (fresh) query to RO with $j < q$: $[\mathbf{c}] := [\mathbf{x}^1 + \mathbf{S}_i \mathbf{u}_\ell]$.
If $[\mathbf{u}_\ell]$ is computed on the q'th (fresh) query to RO, then:
• if $(\mathbf{x}^0, \mathbf{x}^1) \neq z_i$, the game ends and returns 0.
• otherwise, $[\mathbf{c}] := \Big[\boxed{\mathbf{x}^0} + \mathbf{x}^1 + \mathbf{S}_i \mathbf{u}_\ell\Big]$, $\mathcal{S} := \mathcal{S} \cup \{i\}$.
Return $[\mathbf{c}]$.

$\mathsf{OKeygen}(\mathbf{y})$: //$\mathsf{G}^{\star}_{3.q.2}$, $\mathsf{G}^{\star}_{3.q.3}$
Return $\sum_i \mathbf{S}_i^\top \mathbf{y}_i$.

$\mathsf{OCorrupt}(i)$: // $\mathsf{G}^{\star}_{3.q.2}$, $\mathsf{G}^{\star}_{3.q.3}$
Return $\mathbf{S}_i$.

Figure 6.4: Games $\mathsf{G}^{\star}_{3.q.2}$ and $\mathsf{G}^{\star}_{3.q.3}$, with $q \in [Q]$, for the proof of Theorem 15. Here, RF, RF' are random functions onto $\mathbb{G}^{k+1}$, and $\mathbb{Z}_p^k$, respectively, that are computed on the fly. In each procedure, the components inside a solid (gray) frame are only present in the games marked by a solid (gray) frame.

From one to many ciphertext for MCFE

In this section, we add an extra layer of public-key, single-input inner product FE on top of the inner product MCFE from Section 6.1, to remove the restriction of having a unique challenge ciphertext per client and per label. Our construction works for any public-key single-input inner product FE that is compatible with the inner product MCFE from Section 6.1, that is, an FE whose message space is the ciphertext space of the MCFE. Namely, we use a single-input FE whose encryption algorithm can act on vectors of group elements, in $\mathbb{G}^m$, where $\mathbb{G}$ is a prime-order group, as opposed to vectors over $\mathbb{Z}$. Decryption recovers the inner product in the group $\mathbb{G}$, without any restriction on the size of the input of the encryption and decryption key generation algorithms. The message space of the FE is $\mathbb{G}^m$, for some dimension m, its decryption key space is $\mathbb{Z}_p^m$, where p is the order of $\mathbb{G}$, and for any $[\mathbf{x}] \in \mathbb{G}^m$, $\mathbf{y} \in \mathbb{Z}_p^m$,the decryption of the encryption of $[\mathbf{x}]$ together with the functional decryption key associated with $\mathbf{y}$ yields $[\mathbf{x}^\top \mathbf{y}]$.

For correctness, we exploit the fact that decryption of the MCFE from Section 6.1 computes the inner product of the ciphertext together with the decryption keys. For security, we exploit the fact that the MCFE is linearly homomorphic, in the sense that given an input $\mathbf{x}$, one can publicly maul an encryption of $\mathbf{x}'$ into an encryption of $\mathbf{x} + \mathbf{x}'$. This is used to bootstrap the security from one to many challenge ciphertexts per (user,label) pair, similarly to the security proof in Chapter 4 in the context of multi-input inner product FE. In fact, the construction in Chapter 5 uses a one-time secure multi-input FE as inner layer, and a single-input inner product FE as outer layer, while we use an inner product MCFE as inner layer, and a single-input inner product FE as outer layer.

Before presenting our construction in Figure 6.5, we remark that the MCFE from Section 6.1 satisfies the following properties.

- *Linear Homomorphism of ciphertexts*: for any $i \in [n]$, $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{Z}_p^m$, and any label ℓ, we have $[\mathbf{c}_i] + [\mathbf{x}'_i] = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i + \mathbf{x}'_i, \ell)$, where $[\mathbf{c}_i] = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i, \ell)$.
- *Deterministic Encryption.* In particular, together with the linear homomorphism of ciphertexts, this implies that for any $\mathbf{x}_i, \mathbf{x}'_i \in \mathbb{Z}_p^m$ and any label ℓ, we have: $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i, \ell) - \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}'_i, \ell) = [\mathbf{x}_i - \mathbf{x}'_i]$.

Correctness. By correctness of $\mathcal{IPFE}$, we have for all $i \in [n]$, and any label ℓ: $[\alpha_{\ell,i}] = [\langle \mathbf{y}_i, \mathbf{x}_i + \mathbf{S}_i \mathbf{u}_\ell \rangle] = [\langle \mathbf{y}_i, \mathbf{x}_i \rangle] + [\mathbf{u}_\ell]^\top \mathbf{S}_i^\top \mathbf{y}_i$. Thus, $\sum_i [\alpha_{\ell,i}] = [\langle \mathbf{y}, \mathbf{x} \rangle] + [\mathbf{u}_\ell]^\top (\sum_i \mathbf{S}_i^\top \mathbf{y}_i)$. Since $\mathbf{d} = \sum_i \mathbf{S}_i^\top \mathbf{y}_i$, we have $\sum_i [\alpha_{\ell,i}] = [\langle \mathbf{y}, \mathbf{x} \rangle] + [\mathbf{u}_\ell]^\top \mathbf{d}$, hence $[\alpha] = [\langle \mathbf{x}, \mathbf{y} \rangle]$.

We know $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \leq n \cdot m \cdot X \cdot Y$, which is bounded by a polynomial in the security parameter. Thus, decryption can efficiently recover the discrete logarithm: $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \bmod p = \sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle$, where the equality holds since $\sum_i \langle \mathbf{x}_i, \mathbf{y}_i \rangle \leq n \cdot m \cdot X \cdot Y \ll p$.

Security proof.

Theorem 16: many-AD-IND-weak security of $\mathcal{MCFE}$

The scheme $\mathcal{MCFE}$ from Figure 6.5 is many-AD-IND-weak secure, assuming the underlying single-input FE $\mathcal{IPFE}$ is many-AD-IND secure, and using the fact that the scheme $\mathcal{MCFE}'$ from Figure 6.2 is one-AD-IND-weak secure.

Proof overview. The proof is similar than the proof of Theorem 10, in Chapter 4, which proves the many-time security of our multi-input FE from its one-time security. In the one-AD-IND-weak security game, the adversary only queries OEnc on one input $(i, (\mathbf{x}_i^0, \mathbf{x}_i^1), \ell)$ per

$\underline{\mathsf{Setup}(1^\lambda, F_n^{m,X,Y})}$:
$(\mathsf{pk}', \mathsf{msk}', (\mathsf{ek}_i')_{i\in[n]}) \leftarrow \mathsf{Setup}'(1^\lambda, F_n^{m,X,Y})$, $\mathsf{gpk} \leftarrow \mathsf{IP.GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})$, for all $i \in [n]$, $(\mathsf{IP.ek}_i, \mathsf{IP.msk}_i) \leftarrow \mathsf{IP.Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})$, $\mathsf{ek}_i := \mathsf{ek}_i'$, $\mathsf{pk} := (\mathsf{pk}', \mathsf{gpk}, \{\mathsf{IP.ek}_i\}_{i\in[n]})$, $\mathsf{msk} := (\mathsf{msk}', \{\mathsf{IP.msk}_i\}_{i\in[n]})$.
Return $(\mathsf{pk}, \mathsf{msk}, \{\mathsf{ek}_i\}_{i\in[n]})$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i, \ell)}$:
$[\mathbf{c}_{\ell,i}] \leftarrow \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}_i', \mathbf{x}_i, \ell)$
Return $C_{\ell,i} := \mathsf{IP.Enc}(\mathsf{gpk}, \mathsf{IP.ek}_i, [\mathbf{c}_{\ell,i}])$

$\underline{\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \mathbf{y} := \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)}$:
$\mathsf{dk}_{\mathbf{y}}' \leftarrow \mathsf{KeyGen}'(\mathsf{pk}', \mathsf{msk}', \mathbf{y})$, and for all $i \in [n]$: $\mathsf{dk}_{\mathbf{y}_i} \leftarrow \mathsf{IP.KeyGen}(\mathsf{gpk}, \mathsf{IP.msk}_i, \mathbf{y}_i)$.
Return $\mathsf{dk}_{\mathbf{y}} := (\mathsf{dk}_{\mathbf{y}}', \{\mathsf{dk}_{\mathbf{y}_i}\}_{i\in[n]})$.

$\underline{\mathsf{Dec}\big(\mathsf{pk}, \mathsf{dk}_{\mathbf{y}}, \{C_{\ell,i}\}_{i\in[n]}, \ell\big)}$:
Parse $\mathsf{dk}_{\mathbf{y}} = (\mathsf{dk}_{\mathbf{y}}', \{\mathsf{dk}_{\mathbf{y}_i}\}_{i\in[n]})$, where $\mathsf{dk}_{\mathbf{y}}' = (\mathbf{y}, \mathbf{d})$. For all $i \in [n]$, compute $[\alpha_{\ell,i}] \leftarrow \mathsf{IP.Dec}(\mathsf{gpk}, C_{\ell,i}, \mathsf{dk}_{\mathbf{y}_i})$. Then $[\mathbf{u}_\ell] = \mathsf{H}(\ell)$, $[\alpha] = [\sum_i \alpha_{\ell,i}] - [\mathbf{u}_\ell]^\top \mathbf{d}$. Finally, it returns the discrete logarithm $\alpha \in \mathbb{Z}_p$.

Figure 6.5: $\mathcal{MCFE}$, a many-AD-IND-weak secure MCFE for inner product. Here, $\mathcal{MCFE}' := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$ is the one-AD-IND-weak secure from Section 6.1, and $\mathcal{IPFE} := (\mathsf{IP.GSetup}, \mathsf{IP.Setup}, \mathsf{IP.Enc}, \mathsf{IP.KeyGen}, \mathsf{IP.Dec})$ is a many-AD-IND secure, public-key, single-input inner product FE. Here, H denotes the hash function that is part of pk'.

input slot $i \in [n]$ and label ℓ. In the many-AD-IND-weak security game, however, we may have many such queries, and we use an index $j \in [Q_{i,\ell}]$ to enumerate over such queries, where $Q_{i,\ell}$ denotes the number of queries to OEnc which contain the input $i \in [n]$ and the label ℓ. That is, we call $(\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1})$ the j'th query to OEnc on label ℓ and slot i. The proof goes in two steps:

- We first switch encryptions of $\mathbf{x}_1^{1,0}, \ldots, \mathbf{x}_n^{1,0}$ to those of $\mathbf{x}_1^{1,1}, \ldots, \mathbf{x}_n^{1,1}$ all at once, and for the remaining ciphertexts, we switch from an encryption of $\mathbf{x}_i^{j,0} = (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,0}$ to that of $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$. We can do so using the one-AD-IND-weak security of $\mathcal{MCFE}$, and the fact that its encryption algorithm is linear homomorphic. In particular, given an encryption of $\mathbf{x}_i^{1,\beta}$ for $\beta \in \{0,1\}$, and the vector $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0})$, we can produce (only with the public key) an encryption of $(\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,\beta}$. Thus, we can generate all the challenge ciphertexts only from the security game where there is only a single ciphertext in each slot and label.

- Then, we switch from encryptions of

$$(\mathbf{x}_i^{2,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_i,0} - \mathbf{x}_i^{1,0}) + \mathbf{x}_i^{1,1}$$

  to those of

$$(\mathbf{x}_i^{2,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}, \ldots, (\mathbf{x}_i^{Q_i,1} - \mathbf{x}_i^{1,1}) + \mathbf{x}_i^{1,1}.$$

To carry out the latter hybrid argument, we use the fact that the queries must satisfy the constraint:

$$\begin{aligned}
[\mathbf{c}_{0,i}^\top \mathbf{y}_i] &= [\mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}]^\top \mathbf{y}_i + [\mathbf{S}_i \mathbf{u}_\ell]^\top \mathbf{y}_i \\
&= [\mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}]^\top \mathbf{y}_i + [\mathbf{S}_i \mathbf{u}_\ell]^\top \mathbf{y}_i \\
&= [\mathbf{c}_{1,i}]^\top \mathbf{y}_i,
\end{aligned}$$

where Enc' denotes the encryption algorithm of $\mathcal{MCFE}'$ from Figure 6.2, and for all $b \in \{0,1\}$, $[\mathbf{c}_{b,i}] := \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,b} - \mathbf{x}_i^{1,b}, \ell)$.

The second equality is equivalent to $\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$, which follows from the restriction imposed by the security game (see Remark 7).

Thus, we can use the many-AD-IND security of the single-input FE $\mathcal{IPFE}$ for n instances (which is implied by the single instance many-AD-IND security, see Lemma 5), to switch simultaneously all the challenge ciphertexts for all slots $i \in [n]$. As explained in the beginning of this section, the construction is essentially the same construction than multi-input FE for inner product as in Section 5.4, except we replace the perfectly, one-time secure MIFE used in the inner layer, by the one-time secure MCFE from Figure 6.2.

Proof of Theorem 16. We proceed via a series of games, described in Figure 6.6. Let $\mathcal{A}$ be a PPT adversary. For any game G, we denote by $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ the probability that the game G outputs 1 when interacting with $\mathcal{A}$. Note that we have:

$$\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE},\mathcal{A}}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})|,$$

according to Definition 25.

Game G_1: is as game G_0, except we replace the challenge ciphertexts to $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})$ for all $i \in [n]$ and $j \in [Q_i]$, using the one-AD-IND-weak security of $\mathcal{MIFE}'$. Namely, we prove in Lemma 40 that there exists a PPT adversary $\mathcal{B}_1$ such that:

$$\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE}',\mathcal{B}_1}(\lambda).$$

Game G_2: we replace the challenge ciphertexts to $\mathsf{ct}_i^j = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1})$ for all $i \in [n]$ and $j \in [Q_i]$, using the many-AD-IND security of $\mathcal{IPFE}$ for n instances, which is implied by the single-instance security (see Lemma 5). We prove in Lemma 41 that there exists a PPT adversary $\mathcal{B}_2$ such that:

$$\mathsf{Adv}_1(\mathcal{A}) - \mathsf{Adv}_2(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{IPFE},\mathcal{B}_2,n}(\lambda).$$

Putting everything together, we obtain:

$$\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE}',\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{IPFE},\mathcal{B}_2,n}(\lambda).$$

□

Lemma 40: Game G_0 to G_1

There exists a PPT adversary $\mathcal{B}_1$ such that

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq \mathsf{Adv}^{\mathsf{one\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE}',\mathcal{B}_1}(\lambda).$$

Proof of Lemma 40. In game G_1, which is described in Figure 6.6, we replace $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0}, \ell) = \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,0} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}), \ell)$ with $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}), \ell)$ for all $i \in [n], j \in [Q_i]$. This is justified by the following properties:

- one-AD-IND-weak security of $\mathcal{MCFE}'$;
- the fact that Enc' is linearly homomorphic. Namely, for all $i \in [n]$, given $\mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, \mathbf{x}_i^{1,\beta})$, $\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}$ and pk', we can create an encryption $\mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, \mathbf{x}_i^{1,\beta} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0})$ (corresponding to challenge ciphertexts in slot i in game G_β).

The adversary $\mathcal{B}_1$ proceeds as follows.

Games G_0, $\boxed{\mathsf{G}_1}$, $\boxed{\mathsf{G}_2}$:

$(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, F_n^{m,X,Y})$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{OKeygen}(\cdot),\mathsf{OCorrupt}(\cdot)}(\mathsf{pk}, \{\mathsf{ek}_i\}_{i\in\mathcal{CS}})$
Return α if **condition 1** and **extra condition** from Definition 25 of many-AD-IND-weak security are satisfied; otherwise, return 0.

$\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}), \ell)$:
$\mathsf{ct}_i^j := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,0})$
$\boxed{\mathsf{ct}_i^j := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0} + \mathbf{x}_i^{1,1})}$
$\boxed{\mathsf{ct}_i^j := \mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1} + \mathbf{x}_i^{1,1})}$
Return ct_i^j.

$\mathsf{OKeygen}(\mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:
Return $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$.

$\mathsf{OCorrupt}(i)$:
Return ek_i.

Figure 6.6: Games for the proof of Theorem 16.

-Simulation of pk:

The adversary $\mathcal{B}$ samples $\mathsf{gpk} \leftarrow \mathsf{GSetup}(1^\lambda, F_{\mathsf{IP}}^{m,X,Y})$, and for all $i \in [n]$, $(\mathsf{ek}_i, \mathsf{msk}_i) \leftarrow \mathsf{IP.Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{IP}}^{m,X,Y})$. It receives a public key pk' from its own experiment. It returns $\mathsf{pk} := (\mathsf{pk}', \mathsf{gpk}, \{\mathsf{IP.ek}_i\}_{i\in[n]})$ to $\mathcal{A}$.

-Simulation of $\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}), \ell)$:

If $j = 1$, that is, it is the first query for slot $i \in [n]$ and label ℓ, then $\mathcal{B}_1$ queries its own oracle to get $[\mathbf{c}_i^1] := \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}_i', \mathbf{x}_i^{1,\beta}, \ell)$, where $\beta \in \{0, 1\}$, depending on the experiment $\mathcal{B}_1$ is interacting with. If $j > 1$, $\mathcal{B}_1$ uses the fact that $\mathcal{MCFE}'$ is linearly homomorphic to generate all the remaining ciphertexts ct_i^j for $i \in [n]$, $j \in \{2, \ldots, Q_i\}$ by combining $\mathsf{ct}_i = \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}_i', \mathbf{x}_i^{1,\beta}, \ell)$ with the vector $\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}$ to obtain an encryption $\mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}_i', \mathbf{x}_i^{1,\beta} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \ell)$, which matches the challenge ciphertexts in Game G_β. Note that this can be done using pk' only. Moreover, there is no need to rerandomize the challenge ciphertext, since the encryption is deterministic in $\mathcal{MCFE}'$. Then, for all $i \in [n]$ and all $j \in [Q_i]$, $\mathcal{B}_1$ computes $\mathsf{ct}_i^j := \mathsf{IP.Enc}(\mathsf{gpk}, \mathsf{IP.ek}_i, [\mathbf{c}_i^j])$, and returns $\{\mathsf{ct}_i^j\}_{i\in[n], j\in[Q_i]}$ to $\mathcal{A}$.

-Simulation of $\mathsf{OKeygen}(\mathbf{y} := \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:

$\mathcal{B}_1$ uses its own secret key generation oracle to get $\mathsf{dk}_{\mathbf{y}}' \leftarrow \mathsf{OKeygen}'(\mathbf{y})$, and for all $i \in [n]$, computes $\mathsf{dk}_{\mathbf{y}_i} \leftarrow \mathsf{IP.KeyGen}(\mathsf{gpk}, \mathsf{IP.msk}_i, \mathbf{y}_i)$. It returns $(\mathsf{dk}_{\mathbf{y}}', \{\mathsf{dk}_{\mathbf{y}_i}\}_{i\in[n]})$ to $\mathcal{A}$.

-Simulation of $\mathsf{OCorrupt}(i)$:

$\mathcal{B}_1$ uses its own oracle to get $\mathsf{ek}_i' \leftarrow \mathsf{OCorrupt}'(i)$, which it returns to $\mathcal{A}$.

Finally, $\mathcal{B}_1$ forwards the output α of $\mathcal{A}$ to its own experiment. It is clear that for all $\beta \in \{0, 1\}$, when $\mathcal{B}_1$ interacts with **one-AD-IND**$_\beta^{\mathcal{MCFE}'}$, it simulates the game G_β to $\mathcal{A}$.

Therefore,

$$\mathsf{Adv}^{\text{one-AD-IND}}_{\mathcal{MCFE}',\mathcal{B}_1}(\lambda) = \left|\Pr\left[\textbf{one-AD-IND}_0^{\mathcal{MCFE}'}(1^\lambda,\mathcal{B}_1)=1\right] - \Pr\left[\textbf{one-AD-IND}_1^{\mathcal{MCFE}'}(1^\lambda,\mathcal{B}_1)=1\right]\right| = |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})|.$$

□

Lemma 41: Game G_1 to G_2

There exists a PPT adversary $\mathcal{B}_2$ such that

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq \mathsf{Adv}^{\text{many-AD-IND}}_{\mathcal{IPFE},\mathcal{B}_2,n}(\lambda).$$

Proof of Lemma 41. In Game G_2, we replace $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}), \ell)$ with $\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, \mathbf{x}_i^{1,1} + (\mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}), \ell)$ for all $i \in [n], j \in [Q_i]$. This follows from the many-AD-IND security of $\mathcal{IPFE}$ for n instances, which we can use since for each key query $\mathbf{y}_1 \| \dots \| \mathbf{y}_n$, we have

$$\begin{aligned}[\mathbf{c}_{0,i}^\top \mathbf{y}_i] &= [\mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}]^\top \mathbf{y}_i + [\mathbf{S}_i \mathbf{u}_\ell]^\top \mathbf{y}_i \\ &= [\mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}]^\top \mathbf{y}_i + [\mathbf{S}_i \mathbf{u}_\ell]^\top \mathbf{y}_i \\ &= [\mathbf{c}_{1,i}]^\top \mathbf{y}_i,\end{aligned}$$

where for all $b \in \{0,1\}$, $[\mathbf{c}_{b,i}] := \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}_i', \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,b} - \mathbf{x}_i^{1,b}, \ell)$.

The second equality is equivalent to $\langle \mathbf{x}_i^{j,0} - \mathbf{x}_i^{1,0}, \mathbf{y}_i \rangle = \langle \mathbf{x}_i^{j,1} - \mathbf{x}_i^{1,1}, \mathbf{y}_i \rangle$, which follows from the restriction imposed by the security game (see Remark 7).

We build a PPT adversary $\mathcal{B}_2$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq \mathsf{Adv}^{\text{many-AD-IND}}_{\mathcal{IPFE},\mathcal{B}_2,n}(\lambda).$$

Adversary $\mathcal{B}_2$ proceeds as follows.

-Simulation of pk:

Adversary $\mathcal{B}_2$ receives $(\mathsf{gpk}, \{\mathsf{IP.ek}_i\}_{i\in[n]})$ from its experiment. Then, it samples $(\mathsf{pk}', \mathsf{msk}', \{\mathsf{ek}_i'\}_{i\in[n]}) \leftarrow \mathsf{Setup}'(1^\lambda, F_n^{m,X,Y})$, and sends $\mathsf{pk} := (\mathsf{pk}', \mathsf{gpk}, \{\mathsf{IP.ek}_i\}_{i\in[n]})$, to $\mathcal{A}$.

-Simulation of $\mathsf{OEnc}(i, (\mathbf{x}_i^{j,0}, \mathbf{x}_i^{j,1}), \ell)$:

For all $b \in \{0,1\}$, $\mathcal{B}_1$ computes $[\mathbf{c}_i^{j,b}] \leftarrow \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}_i', \mathbf{x}_i^{1,1} + \mathbf{x}_i^{j,b} - \mathbf{x}_i^{1,b}, \ell)$, and queries its own encryption oracle on input $(i, ([\mathbf{c}_i^{j,0}], [\mathbf{c}_i^{j,1}]))$, to get $\mathsf{IP.Enc}(\mathsf{gpk}, \mathsf{IP.ek}_i, [\mathbf{c}_i^{j,\beta}])$, which it forwards to $\mathcal{A}$, where $\beta \in \{0,1\}$, depending on the experiment $\mathcal{B}_2$ is interacting with.

-Simulation of $\mathsf{OKeygen}(\mathbf{y} := \mathbf{y}_1 \| \cdots \| \mathbf{y}_n)$:

For all $i \in [n]$, $\mathcal{B}_1$ uses its own decryption key generation oracle on input $\mathbf{y}_i$ to get $\mathsf{dk}_{\mathbf{y}_i} := \mathsf{IP.KeyGen}(\mathsf{gpk}, \mathsf{IP.msk}_i, \mathbf{y}_i)$. It computes $\mathsf{dk}_{\mathbf{y}} := \mathsf{KeyGen}'(\mathsf{pk}', \mathsf{msk}', \mathbf{y})$, which it can do since it knows msk'. It returns $(\mathsf{dk}_{\mathbf{y}}', \{\mathsf{dk}_{\mathbf{y}_i}\}_{i\in[n]})$ to $\mathcal{A}$.

-Simulation of $\mathsf{OCorrupt}(i)$:

$\mathcal{B}_2$ returns ek'_i to $\mathcal{A}$.

Finally, $\mathcal{B}_2$ checks whether **condition 1** and **extra condition** from Definition 25 are satisfied. Note that involves checking an exponential number of equation for general functionalities. But in the case of inner-product, $\mathcal{B}_2$ just has to look at spanned vector sub-spaces. Namely, all queries $(i, \mathbf{x}_i^{j_i,0}, \mathbf{x}_i^{j_i,1}, \ell)_{i\in[n],j_i\in[Q_i]}$ to OEnc and all queries $\mathbf{y} := (\mathbf{y}_1\| \cdots \|\mathbf{y}_n)$ to $\mathsf{OKeygen}$ must satisfy: $\sum_i \langle \mathbf{x}_i^{j_i,0}, \mathbf{y}_i\rangle = \sum_i \langle \mathbf{x}_i^{j_i,1}, \mathbf{y}_i\rangle$. This is an exponential number of linear equations, but, as noted in the beginning of Chapter 4, it suffices to verify the linearly independent equations, of which there can be at most $n \cdot m$. This can be done efficiently given the queries.

If these conditions are satisfied, then $\mathcal{B}_2$ forwards $\mathcal{A}$'s output α to its own experiment, otherwise it sends 0 to its own experiment. It is clear that for all $\beta \in \{0, 1\}$, when $\mathcal{B}_2$ interacts with $\mathbf{many\text{-}AD\text{-}IND}_\beta^{\mathcal{IPFE}}(1^\lambda, 1^n, \mathcal{B}_2)$, it simulates the game $\mathsf{G}_{1+\beta}$ to $\mathcal{A}$. Therefore,

$$\mathsf{Adv}^{\mathsf{many\text{-}AD\text{-}IND}}_{\mathcal{IPFE},\mathcal{B}_2,n}(\lambda) =$$
$$\left|\Pr\left[\mathbf{many\text{-}AD\text{-}IND}_0^{\mathcal{IPFE}}(1^\lambda, 1^n, \mathcal{B}_2) = 1\right] - \Pr\left[\mathbf{many\text{-}AD\text{-}IND}_1^{\mathcal{IPFE}}(1^\lambda, 1^n, \mathcal{B}_2) = 1\right]\right| =$$
$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})|.$$

□

Secret Sharing Encapsulation

As explained in the introduction of this chapter, in the xx-AD-IND-weak security notion, incomplete ciphertexts were considered illegitimate. This was with the intuition that no adversary should use it since this leaks no information. But actually, an adversary could exploit that in the real-life. We wish to obtain xx-AD-IND security, where the adversary can use incomplete ciphertexts. We upgrade the scheme from the previous section so that no information is leaked in such a case.

Namely, we present a generic layer, called the Secret Sharing Encapsulation (SSE), that we will use to encapsulate ciphertexts. It allows a user to recover the ciphertexts from the n senders only when he gets the contributions of all the servers. That is, if one sender did not send anything, the user cannot get any information from any of the ciphertexts of the other senders. More concretely, a share of a key $S_{\ell,i}$ is generated for each user $i \in [n]$ and each label ℓ. Unless all the shares $S_{i,\ell}$ have been generated, the encapsulation keys are random and mask all the ciphertexts.

After giving the definition of SSE, we provide a construction whose security is based on the DBDH assumption in asymmetric pairing groups.

Definitions

Definition 28: Secret Sharing Encapsulation (SSE)

A secret sharing encapsulation on $\mathcal{K}$ over a set of n senders is defined by four algorithms:

- $\mathsf{SSE.Setup}(1^\lambda)$: Takes as input a security parameter 1^λ and generates the public parameters $\mathsf{pk}_{\mathsf{sse}}$ and the personal encryption keys are $\mathsf{ek}_{\mathsf{sse},i}$ for all $i \in [n]$;
- $\mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell)$: Takes as input the public parameters $\mathsf{pk}_{\mathsf{sse}}$ and the label ℓ and outputs a ciphertext C_ℓ and an encapsulation key $K_\ell \in \mathcal{K}$;
- $\mathsf{SSE.Share}(\mathsf{ek}_{\mathsf{sse},i}, \ell)$: Takes as input a personal encryption $\mathsf{ek}_{\mathsf{sse},i}$ and the label ℓ, outputs the share $S_{\ell,i}$;

- $\mathsf{SSE.Decaps}(\mathsf{pk_{sse}}, (S_{\ell,i})_{i\in[n]}, \ell, C_\ell)$: Takes as input all the shares $S_{\ell,i}$ for all $i \in [n]$, a label ℓ, and a ciphertext C_ℓ, and outputs the encapsulation key K_ℓ.

Correctness. For any label ℓ, we have: $\Pr[\mathsf{SSE.Decaps}(\mathsf{pk_{sse}}, (S_{\ell,i})_{i\in[n]}, \ell, C_\ell) = K_\ell] = 1$, where the probability is taken over $(\mathsf{pk_{sse}}, (\mathsf{ek}_{\mathsf{sse},i})_{i\in[n]}) \leftarrow \mathsf{SSE.Setup}(\lambda)$, $(C_\ell, K_\ell) \leftarrow \mathsf{SSE.Encaps}(\mathsf{pk_{sse}}, \ell)$, and $S_{\ell,i} \leftarrow \mathsf{SSE.Share}(\mathsf{ek}_{\mathsf{sse},i}, \ell)$ for all $i \in [n]$.

Security. We want to show that the encapsulated keys are indistinguishable from random if not all the shares are known to the adversary. We could define a Real-or-Random security game [BDJR97a] for all the masks. Instead, we limit the Real-or-Random queries to one label only (whose index is chosen in advance), and for all the other labels, the adversary can do the encapsulation by itself, since it just uses a public key. This is well-known that a hybrid proof among the label indices (the order they appear in the game) shows that the One-Label security is equivalent to the Many-Label security. The One-Label definition will be enough for our applications.

Definition 29: 1-label-IND security for SSE

An SSE scheme $\mathcal{SSE} := (\mathsf{SSE.Setup}, \mathsf{SSE.Encaps}, \mathsf{SSE.Share}, \mathsf{SSE.Decaps})$ over n users is 1-label-IND secure if for every stateful PPT adversary $\mathcal{A}$, we have:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{1\text{-}label\text{-}IND}}_{\mathcal{SSE},\mathcal{A}}(\lambda) &= \left|\Pr\left[\mathsf{1\text{-}label\text{-}IND}^{\mathcal{SSE}}_0(1^\lambda, \mathcal{A}) = 1\right] - \Pr\left[\mathsf{1\text{-}label\text{-}IND}^{\mathcal{SSE}}_1(1^\lambda, \mathcal{A}) = 1\right]\right| \\ &= \mathsf{negl}(\lambda),\end{aligned}$$

where the experiments are defined for $\beta \in \{0, 1\}$ as follows:

Experiment $\mathsf{1\text{-}label\text{-}IND}^{\mathcal{SSE}}_\beta(1^\lambda, \mathcal{A})$:
$i^\star \leftarrow \mathcal{A}(1^\lambda, 1^n)$
$(\mathsf{pk_{sse}}, (\mathsf{ek}_{\mathsf{sse},i})_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda)$
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEncaps}(\cdot),\mathsf{OShare}(\cdot,\cdot),\mathsf{OCorrupt}(\cdot)}(\mathsf{pk})$
Output: α

On input a label ℓ, the oracle $\mathsf{OEncaps}(\ell)$ computes $(C_\ell, K_\ell) \leftarrow \mathsf{SSE.Encaps}(\mathsf{pk_{sse}}, \ell)$, $K_0 := K_\ell$, $K_1 \leftarrow_{\mathrm{R}} \mathcal{K}$, and returns (C_ℓ, K_β). On input $i \in [n]$, and a label ℓ, the oracle $\mathsf{OShare}(i, \ell)$ returns $S_{i,\ell} \leftarrow \mathsf{SSE.Share}(\mathsf{ek}_{\mathsf{sse},i}, \ell)$. On input $i \in [n]$, the oracle $\mathsf{OCorrupt}(i)$ returns $\mathsf{ek}_{\mathsf{sse},i}$.

We require that the oracle $\mathsf{OEncaps}$ is only called on one label $\ell^\star$, OShare is never called on input $(i^\star, \ell^\star)$, and $\mathsf{OCorrupt}$ is never called on $i^\star$. If this condition is not satisfied, the experiment outputs 0 instead of α.

Construction of the Secret Sharing Encapsulation

We build an SSE from the DBDH assumption in asymmetric pairing groups, in the random oracle model, in Figure 6.7.

We stress here that K_ℓ is not unique for each label ℓ: whereas $S_{\ell,i}$ deterministically depends on ℓ and the slot i, K_ℓ is randomized by the random coins r. Hence, with all the shares, using a specific C_ℓ one can recover the associated K_ℓ. Correctness follows from the fact that the above decapsulated key K_ℓ is equal to

$$e\left(\sum_{i\in[n]} t_i \cdot \mathsf{H}(\ell), [r]_2\right) = e\left(\mathsf{H}(\ell), [r \cdot \sum_{i\in[n]} t_i]_2\right),$$

$\underline{\mathsf{SSE.Setup}(1^\lambda)}$:
$\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, p, P_1, P_2) \leftarrow \mathsf{PGGen}(1^\lambda)$, $\mathsf{H} : \{0,1\}^* \to \mathbb{G}_1$ be a full domain hash function modeled as a random oracle.
For all $i \in [n]$, $t_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $\mathsf{ek}_{\mathsf{sse},i} := t_i$, $\mathsf{pk}_{\mathsf{sse}} = (\mathcal{PG}, \mathsf{H}, [\sum_{i\in[n]} t_i]_2)$.
Return $(\mathsf{pk}_{\mathsf{sse}}, (\mathsf{ek}_{\mathsf{sse},i})_{i\in[n]})$.

$\underline{\mathsf{SSE.Share}(\mathsf{pk}_{\mathsf{sse}}, \mathsf{ek}_{\mathsf{sse},i}, \ell)}$:
Return $S_{\ell,i} := t_i \cdot \mathsf{H}(\ell) \in \mathbb{G}_1$.

$\underline{\mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell)}$:
$r \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $C_\ell := [r]_2$, $K_\ell := e(\mathsf{H}(\ell), r \cdot \sum_{i\in[n]} t_i)$. Return (C_ℓ, K_ℓ).

$\underline{\mathsf{SSE.Decaps}(\mathsf{pk}_{\mathsf{sse}}, (S_{i,\ell})_{i\in[n]}, \ell, C_\ell)}$:
Return $K_\ell := e(\sum_{i\in[n]} S_{\ell,i}, C_\ell)$.

Figure 6.7: SSE based on DBDH in asymmetric pairing groups.

where the pair (C_ℓ, K_ℓ) has been generated by the same $\mathsf{SSE.Encaps}$ call, with the same random r. The intuition for the security is that given all the $S_{\ell,i} = t_i \cdot \mathsf{H}(\ell)$ for a label ℓ, one can recover the masks $K_\ell = e(\mathsf{H}(\ell), [r \cdot \sum_{i\in[n]} t_i]_2)$ using $C_\ell = [r]_2$. However if $S_{\ell,i}$ is missing for one slot i, then all the encapsulation keys K_ℓ are pseudo-random, from the DBDH assumption.

Our construction is reminiscent from the Identity-Based Encryption from [BF01], where a ciphertext for an identity ℓ is of the form $e(\mathsf{H}(\ell), [\mathsf{msk} \cdot r]_2)$ for a random $r \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, and a functional decryption key for identity ℓ is of the form $\mathsf{H}(\ell)^{\mathsf{msk}}$. In our construction, we share the master secret msk into the $\{t_i\}_{i\in[n]}$, and each $S_{\ell,i}$ represents a share of the functional decryption key for identity ℓ.

Security proof.

Theorem 17: 1-label-IND security of SSE

The SSE scheme presented in Figure 6.7 is 1-label-IND secure under the DBDH assumption, in the random oracle model.

Proof of Theorem 17. We build a PPT adversary $\mathcal{B}$ such that

$$\mathsf{Adv}^{\mathsf{1\text{-}label\text{-}IND}}_{\mathcal{SSE},\mathcal{A}}(\lambda) \leq (1 + q_{\mathsf{H}}) \cdot \mathsf{Adv}^{q_{\mathsf{Enc}}\text{-}\mathsf{DBDH}}_{\mathcal{PG},\mathcal{B}}(\lambda),$$

where q_{H} denotes the number of calls to the random oracle prior to any query to $\mathsf{OEncaps}$, either direct calls, or indirect via OShare. The integer q_{Enc} denotes the number of calls to the oracle $\mathsf{OEncaps}$. We will then conclude using the random self reducibility of the DBDH assumption (see Lemma 4).

The adversary $\mathcal{B}$ receives a q_{Enc}-fold DBDH challenge $\left(\mathcal{PG}, [a]_1, [b]_1, [b]_2, \{[c_i]_2, [s_i]_T\}_{i\in[q_{\mathsf{Enc}}]}\right)$, where q_{Enc} denotes the number of queries of $\mathcal{A}$ to its oracle $\mathsf{OEncaps}$, and receives $i^\star \in [n]$ from $\mathcal{A}$.

Then, $\mathcal{B}$ guesses $\rho \leftarrow_{\mathrm{R}} \{0, \ldots, q_{\mathsf{H}}\}$. Intuitively, ρ is a guess on when the random oracle is going to be queried on $\ell^\star$, the first label used as input to $\mathsf{OEncaps}$ (without loss of generality, we can assume $\mathsf{OEncaps}$ is queried at least once by $\mathcal{A}$, otherwise the security is trivially satisfied), with $\rho = 0$ indicating that the adversary never queries H on $\ell^\star$ before querying $\mathsf{OEncaps}$.

Then, $\mathcal{B}$ samples $t_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ and sets $\mathsf{ek}_{\mathsf{sse},i} := t_i$ for all $i \in [n]$, $i \neq i^\star$, and sets $[t_{i^\star}]_2 := [b]_2$. It returns $\mathsf{pk}_{\mathsf{sse}} := (\mathcal{PG}, [\sum_{i\in[n]} t_i]_2)$ to $\mathcal{A}$.

For any query $\mathsf{OCorrupt}(i)$: if $i \neq i^\star$, $\mathcal{B}$ returns $\mathsf{ek}_{\mathsf{sse},i}$, otherwise $\mathcal{B}$ stops simulating the experiment for $\mathcal{A}$ and returns 0 to its own experiment.

For any query to the random oracle H, if this the ρ'th new query, then $\mathcal{B}$ sets $\mathsf{H}(\ell_\rho) := [a]_1$. For others queries, $\mathcal{B}$ outputs $[h]_1$ for a random $h \leftarrow_{\mathrm{R}} \mathbb{Z}_p$. $\mathcal{B}$ keeps track of the queries and outputs to the random oracle H, so that it answers two identical queries with the same output.

For any query to $\mathsf{OEncaps}(\ell)$: if ℓ has never been queried to the random oracle H before (directly, or indirectly via OShare) and $\rho = 0$, then $\mathcal{B}$ sets $\mathsf{H}(\ell) := [a]_1$; if ℓ was queried to random oracle as the ρ'th new query (again, we consider direct and indirect queries to H, the latter coming from OShare), then we already have $\mathsf{H}(\ell) = [a]_1$. In both cases, $\mathcal{B}$ sets $C_\ell \leftarrow [c_j]_2$, for the next index j in the q_{Enc}-fold DBDH instance, computes $K_\ell \leftarrow [s_j]_T + e([a]_1, (\sum_{i \neq i^\star} t_i) \cdot [c_j]_2)$, and returns (C_ℓ, K_ℓ) to $\mathcal{A}$. Otherwise, the guess ρ was incorrect: $\mathcal{B}$ stops simulating the experiment for $\mathcal{A}$, and returns 0 to its own experiment. Moreover, if $\mathcal{A}$ ever calls $\mathsf{OEncaps}$ on different labels ℓ, then $\mathcal{B}$ stops simulating this experiment for $\mathcal{A}$ and returns 0 to its own experiment.

For any query to $\mathsf{OShare}(i, \ell)$: if the random oracle has been called on ℓ, then $\mathcal{B}$ uses the already computed input $\mathsf{H}(\ell)$; otherwise, it computes $\mathsf{H}(\ell)$ for the first time as explained above. If $i = i^\star$ and $\ell = \ell_\rho$, then $\mathcal{B}$ stops simulating the experiment for $\mathcal{A}$ and returns 0 to its own experiment. Otherwise, that means either $i \neq i^\star$, in which case $\mathcal{B}$ knows $t_i \in \mathbb{Z}_p$, or $\ell \neq \ell_\rho$, in which case $\mathcal{B}$ the discrete logarithm of $\mathsf{H}(\ell)$. In both cases, $\mathcal{B}$ can compute $S_{\ell,i} := t_i \cdot \mathsf{H}(\ell) \in \mathbb{G}_1$, which it returns to $\mathcal{A}$.

At the end of the experiment, $\mathcal{B}$ receives the output α from $\mathcal{A}$. If its guess ρ was correct, $\mathcal{B}$ outputs α to its own experiment, otherwise, it ignores α and returns 0.

When $\mathcal{B}$'s guess is incorrect, it returns 0 to its experiment. Otherwise, when it is given as input a real q_{Enc}-fold DBDH challenge, that is $s_j = abc_j$ for all indices $j \in [q_{\mathsf{Enc}}]$, then $\mathcal{B}$ simulates the 1-label-IND security game with $b = 0$. Indeed, since $b = t_{i^\star}$, for the j-th query to $\mathsf{OEncaps}$, we have:

$$\begin{aligned} K_{\ell^\star} &= [s_j]_T + e([a]_1, (\sum_{i \neq i^\star} t_i) \cdot [c_j]_2) = [abc_j]_T + e([a]_1, (\sum_{i \neq i^\star} t_i) \cdot [c_j]_2) \\ &= e([a]_1, [bc_j]_2) + e([a]_1, (\sum_{i \neq i^\star} t_i) \cdot [c_j]_2) = e([a]_1, [bc_j]_2 + (\sum_{i \neq i^\star} t_i) \cdot [c_j]_2) \\ &= e([a]_1, (b + \sum_{i \neq i^\star} t_i) \cdot [c_j]_2) = e([a]_1, (\sum_{i} t_i) \cdot [c_j]_2) = e(\mathsf{H}(\ell^\star), c_j \cdot T_2) \end{aligned}$$

where $C_{\ell^\star} = [c_j]_2$. When given as input a a random q_{Enc}-fold DBDH challenge, the simulation corresponds to the case $b = 1$. Finally, we conclude using the fact that the guess ρ is correct with probability exactly $\frac{1}{q_{\mathsf{H}}+1}$. □

Strengthening the Security of MCFE Using SSE

We now show how we can enhance the security of any MCFE for any set of functionality $\{F_n\}_{n \in \mathbb{N}}$, using a Secret Sharing Layer as defined in Section 6.3. Namely, we show that the construction from Figure 6.8 is xx-AD-IND secure if the underlying MCFE is xx-AD-IND secure, for any xx ∈ {one,many}, thereby removing the complete-ciphertext restriction. We stress our transformation is not restricted to MCFE for inner product, but works for any functionality.

Generic construction of xx-AD-IND security for MCFE

We present an xx-AD-IND secure MCFE, where xx ∈ {one,many}, for the set of functionalities $\{F_n\}_{n \in \mathbb{N}}$, from any xx-AD-IND-weak secure MCFE for $\{F_n\}_{n \in \mathbb{N}}$, 1-label-IND secure SSE, and symmetric encryption scheme. The generic construction is presented in Figure 6.8.

$\underline{\mathsf{Setup}(1^\lambda, F_n)}$:
$(\mathsf{pk}', \mathsf{msk}', (\mathsf{ek}'_i)_{i\in[n]}) \leftarrow \mathsf{Setup}'(1^\lambda, F_n)$, $(\mathsf{pk}_{\mathsf{sse}}, (\mathsf{ek}_{\mathsf{sse},i})_{i\in[n]}) \leftarrow \mathsf{SSE.Setup}(1^\lambda)$.
$\mathsf{pk} := (\mathsf{pk}', \mathsf{pk}_{\mathsf{sse}})$, $\mathsf{msk} := \mathsf{msk}'$, and for all $i \in [n]$, $\mathsf{ek}_i := (\mathsf{ek}'_i, \mathsf{ek}_{\mathsf{sse},i})$.
Return $(\mathsf{pk}, \mathsf{msk}, (\mathsf{ek}_i)_{i\in[n]})$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{ek}_i, x_i, \ell)}$:
$C'_{\ell,i} \leftarrow \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x_i, \ell)$, $(C_\ell, K_\ell) \leftarrow \mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell)$, $S_{\ell,i} \leftarrow \mathsf{SSE.Share}(\mathsf{pk}_{\mathsf{sse}}, \mathsf{ek}_{\mathsf{sse},i}, \ell)$.
Return the ciphertext $C_{\ell,i} := (D_{\ell,i} := \mathsf{SEnc}(K_\ell, C'_{\ell,i}), C_\ell, S_{\ell,i})$.

$\underline{\mathsf{KeyGen}(\mathsf{pk}, \mathsf{msk}, k)}$:
Return $\mathsf{KeyGen}'(\mathsf{msk}', k)$.

$\underline{\mathsf{Dec}(\mathsf{pk}, \mathsf{dk}_k, \{C_{\ell,i}\}_{i\in[n]}, \ell)}$:
For all $i \in [n]$, parse $C_{\ell,i} = (D_{\ell,i}, C_\ell, S_{\ell,i})$. Compute $K_\ell \leftarrow \mathsf{SSE.Decaps}(\mathsf{pk}_{\mathsf{sse}}, (S_{\ell,i})_{i\in[n]}, \ell, C_\ell)$. For all $i \in [n]$, computes $C'_{\ell,i} \leftarrow \mathsf{SDec}(K_\ell, D_{\ell,i})$.
Return $\mathsf{Dec}'(\mathsf{pk}', \mathsf{dk}_k, \{C'_{\ell,i}\}_{i\in[n]})$.

Figure 6.8: MCFE with xx-AD-IND security from any 1-label-xx-IND secure MCFE $\mathcal{MCFE}' := (\mathsf{Setup}', \mathsf{Enc}', \mathsf{KeyGen}', \mathsf{Dec}')$, SSE scheme $\mathcal{SSE} := (\mathsf{SSE.Setup}, \mathsf{SSE.Encaps}, \mathsf{SSE.Share}, \mathsf{SSE.Decaps})$, and symmetric encryption $\mathcal{SKE} := (\mathsf{SEnc}, \mathsf{SDec})$. Here, xx $\in$ {one,many}. Recall that the algorithm $\mathsf{SSE.Encaps}$ is randomized, thus, different invocation of $\mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell)$ on the same input will produce different outputs.

Correctness: follows straightforwardly from the correctness of the underlying $\mathcal{MCFE}'$, $\mathcal{SSE}$ and $\mathcal{SKE}$.

Security proof.

Theorem 18: Security

The MCFE from Figure 6.8 is xx-AD-IND secure assuming $\mathcal{MCFE}'$ is xx-AD-IND-weak secure, $\mathcal{SSE}$ is 1-label-IND secure, and $\mathcal{SKE}$ is one-time secure.

We stress that this security result keeps all the properties of $\mathcal{MCFE}'$ and $\mathcal{SSE}$:

- if $\mathcal{MCFE}'$ and $\mathcal{SSE}$ are both secure against adaptive corruptions, then, so is $\mathcal{MCFE}$;
- if $\mathcal{MCFE}'$ is many time secure (xx = many), then, so is $\mathcal{MCFE}$.

Proof of Theorem 18. The proof uses a hybrid argument that goes over all the labels $\ell_1, \ldots, \ell_L$ used as input to the queries $\mathcal{A}$ makes to the oracle OEnc. We define the hybrid games G_ρ, for all $\rho \in \{0, \ldots, L\}$ in Figure 6.9. For any hybrid game G_ρ, we denote by $\mathsf{Adv}_{\mathsf{G}_\rho}(\mathcal{A})$ the probability that the game G_ρ outputs 1 when interacting with $\mathcal{A}$. Note that $\mathsf{Adv}^{\mathsf{xx\text{-}AD\text{-}IND}}_{\mathcal{MCFE},\mathcal{A}}(\lambda) = |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_L}(\mathcal{A})|$. Lemma 42 states that for all $i \in [L]$, $|\mathsf{Adv}_{\mathsf{G}_{i-1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_i}(\mathcal{A})|$ is negligible, which concludes the proof. □

Games G_ρ, $\boxed{\mathsf{G}^\star_\rho,\ \mathsf{H}_{\rho,\beta}}$, for all $\rho \in \{0,\dots,L\}$:

$\boxed{i^\star \leftarrow_{\mathrm{R}} \{0,\dots,n\}}$, $(\mathsf{pk}', \mathsf{msk}', (\mathsf{ek}'_i)_{i\in[n]}) \leftarrow \mathsf{Setup}'(1^\lambda, F_n)$, $(\mathsf{pk}_{\mathsf{sse}}, (\mathsf{ek}_{\mathsf{sse},i})_{i\in[n]}) \leftarrow \mathsf{SSE.Setup}(1^\lambda)$, $\mathsf{pk} := (\mathsf{pk}', \mathsf{pk}_{\mathsf{sse}})$, $\mathsf{msk} := \mathsf{msk}'$, and for all $i \in [n]$, $\mathsf{ek}_i := (\mathsf{ek}'_i, \mathsf{ek}_{\mathsf{sse},i})$.
$\alpha \leftarrow \mathcal{A}^{\mathsf{OEnc}(\cdot,\cdot,\cdot),\mathsf{OKeygen}(\cdot),\mathsf{OCorrupt}(\cdot)}(\mathsf{pk})$
Return α if **Condition 1** from Definition 25 is satisfied,

$\boxed{\text{and: } \begin{array}{l}(i^\star \neq 0 \text{ is never queried to } \mathsf{OCorrupt} \text{ and } (\ell_{\rho+1}, i^\star) \text{ is never part of a query to } \mathsf{OEnc}) \text{ OR} \\ (i^\star = 0 \text{ and } \mathsf{OEnc} \text{ is queried on all slots } i \in \mathcal{HS} \text{ for label } \ell_{\rho+1})\end{array}}$;
0 otherwise.

$\mathsf{OEnc}(i, (x_i^0, x_i^1), \ell_j)$:
If $j \le \rho$, $C'_{\ell_j,i} \leftarrow \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x_i^1, \ell_j)$. If $j > \rho$, $C'_{\ell_j,i} \leftarrow \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x_i^0, \ell_j)$.
$(C_{\ell_j}, K_{\ell_j}) \leftarrow \mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell_j)$, $S_{\ell_j,i} \leftarrow \mathsf{SSE.Share}(\mathsf{pk}_{\mathsf{sse}}, \mathsf{ek}_{\mathsf{sse},i})$.
If $j = \rho$, $C'_{\ell_j,i} \leftarrow \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x_i^\beta, \ell_j)$, $K_{\ell_j} \leftarrow_{\mathrm{R}} \mathcal{K}$.
Return $(D_{\ell_j,i} := \mathsf{SEnc}(K_{\ell_j}, C'_{\ell_j,i}), C_{\ell_j}, S_{\ell_j,i})$.

$\mathsf{OKeygen}(k)$: return $\mathsf{KeyGen}(\mathsf{msk}, k)$

$\mathsf{OCorrupt}(i)$: return ek_i

Figure 6.9: Games for the proof of Theorem 18. Here, $\mathcal{HS} := [n] \setminus \mathcal{CS}$, the set of honest slots, where $\mathcal{CS}$ is the set of slots queried to $\mathsf{OCorrupt}$. Recall that the algorithm $\mathsf{SSE.Encaps}$ is randomized, thus, different invocation of $\mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell_j)$ on the same input will produce different outputs.

Lemma 42: From game $\mathsf{G}_{\rho-1}$ to game G_ρ

For any PPT adversary $\mathcal{A}$, for all $\rho \in [L]$, there exist PPT adversaries $\mathcal{B}_\rho$, $\mathcal{B}'_\rho$, and $\mathcal{B}''_\rho$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_{\rho-1}}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_\rho}(\mathcal{A})| \leq (n+1) \cdot \left(\begin{array}{l} \mathsf{Adv}^{\mathsf{xx\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE},\mathcal{B}_\rho}(\lambda) + \\ 2 \cdot \mathsf{Adv}^{\mathsf{1\text{-}label\text{-}IND}}_{\mathcal{SSE},\mathcal{B}'_\rho}(\lambda) + q_e \cdot \mathsf{Adv}^{\mathsf{OT}}_{\mathcal{SKE},\mathcal{B}''_\rho}(\lambda) \end{array} \right),$$

where q_e denotes the number of queries to OEnc.

Proof of Lemma 42. Two cases can happen between games $\mathsf{G}_{\rho-1}$ and G_ρ, for each $\rho \in [L]$: either all the challenge ciphertexts are generated under ℓ_ρ or not all of them. We first make the guess, and then deal with the two cases: if they are all generated (for honest slots, that is, slots that are not queried to $\mathsf{OCorrupt}$), we use the xx-AD-IND-weak security of $\mathcal{MCFE}'$, otherwise there is an honest slot $i^\star$ for which the ciphertext has not been generated, and we use the 1-label-IND security of $\mathcal{SSE}$, together with the one-time security of the symmetric encryption scheme.

Guess of the Case for the ℓ_ρ: We define a new sequence of hybrid games $\mathsf{G}^\star_\rho$ for all $\rho \in \{0, \ldots, L\}$, which is exactly as G_ρ, except that a guess for the missing honest-slot ciphertext $i^\star$ under ℓ_ρ is performed ($i^\star = 0$ means that all the honest-client ciphertexts are expected to be generated under ℓ_ρ). Recall that a slot is called honest if it is not queried to $\mathsf{OCorrupt}$. The games are presented in Figure 6.9. Since $\mathsf{G}^\star_\rho$ and G_ρ are the same unless the guess is incorrect, which happens with probability exactly $1/(n+1)$, for any adversary $\mathcal{A}$: $\mathsf{Adv}_{\mathsf{G}_\rho}(\mathcal{A}) = (n+1) \cdot \mathsf{Adv}_{\mathsf{G}^\star_\rho}(\mathcal{A})$.

All the ciphertexts are generated under ℓ_ρ: We build a PPT adversary $\mathcal{B}_\rho$ against the xx-AD-IND-weak security of $\mathcal{MCFE}'$ such that

$$|\mathsf{Adv}_{\mathsf{G}^\star_{\rho-1}}(\mathcal{A} \wedge i^\star = 0) - \mathsf{Adv}_{\mathsf{G}^\star_\rho}(\mathcal{A} \wedge i^\star = 0)| \leq \mathsf{Adv}^{\mathsf{xx\text{-}AD\text{-}IND\text{-}weak}}_{\mathcal{MCFE}'}(\mathcal{B}_\rho).$$

The adversary $\mathcal{B}_\rho$ simulates $\mathcal{A}$'s view as follows:

- First, it obtains pk' from its own xx-AD-IND-weak security game for $\mathcal{MCFE}'$, samples $(\mathsf{pk}_{\mathsf{sse}}, (\mathsf{ek}_{\mathsf{sse},i})_{i\in[n]}) \leftarrow \mathsf{SSE.Setup}(1^\lambda)$ and returns $\mathsf{pk} = (\mathsf{pk}', \mathsf{pk}_{\mathsf{sse}})$ to the adversary $\mathcal{A}$.
- $\mathsf{OEnc}(i, (x^0, x^1), \ell_j)$: if $j < \rho$, it uses its own encryption oracle OEnc' to get $C \leftarrow \mathsf{OEnc}'(i, (x^1, x^1), \ell_j)$; if $j > \rho$, it uses its own encryption oracle OEnc' to get $C \leftarrow \mathsf{OEnc}'(i, (x^0, x^0), \ell_j)$; if $j = \rho$, then it uses its own encryption oracle to get $C \leftarrow \mathsf{OEnc}'(i, (x^0, x^1), \ell_\rho)$. Then, it computes $(C_{\ell_j}, K_{\ell_j}) \leftarrow \mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell_j)$, and $S_{\ell_j,i} \leftarrow \mathsf{SSE.Share}(\mathsf{ek}_{\mathsf{sse},i}, \ell_j)$. Finally, it computes and returns the ciphertext $(\mathsf{SEnc}(K_{\ell_j}, C), C_{\ell_j}, S_{\ell_j,i})$.
- $\mathsf{OKeygen}(k)$: it uses its own oracle to get $\mathsf{dk}'_k \leftarrow \mathsf{OKeygen}'(k)$, which it returns to $\mathcal{A}$.
- $\mathsf{OCorrupt}(i)$: it uses its own corruption oracle to get $\mathsf{ek}'_i \leftarrow \mathsf{OCorrupt}'(i)$, and returns $\mathsf{ek}_i = (\mathsf{ek}'_i, \mathsf{ek}_{\mathsf{sse},i})$.
- Finally, $\mathcal{B}_\rho$ checks that OEnc is queried on all slots $i \in \mathcal{HS}$ for label ℓ_ρ. If this is the case, it forwards the output α from $\mathcal{A}$. Otherwise, it returns 0 to its own experiment.

First, note that when simulating $\mathcal{A}$'s view, $\mathcal{B}_\rho$ only queries its encryption oracle on input (x^0, x^1) with $x^0 \neq x^1$ for a unique label ℓ_ρ. Moreover, when the guess $i^\star = 0$ is correct, then the **extra condition** from Definition 25 is satisfied: OEnc is queried for label ℓ_ρ on all slots $i \in \mathcal{HS}$

(that is, all slots which are not queried to $\mathsf{OCorrupt}$). Thus, we can use the xx-AD-IND-weak security of $\mathcal{MCFE}'$ to switch $\mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x^0, \ell_\rho)$, as in game $\mathsf{G}^\star_{\rho-1}$ to $\mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x^1, \ell_\rho)$, as in game $\mathsf{G}^\star_\rho$.

Some ciphertexts are missing under ℓ_ρ: For $\beta \in \{0,1\}$, we define the games $\mathsf{H}_{\rho,\beta}$ for all $\rho \in \{0,\ldots,L\}$, and $\beta \in \{0,1\}$, as $\mathsf{G}^\star_\rho$, except that $\mathsf{OEnc}(i,(x^0,x^1),\ell_\rho)$ computes the encryption of x^β, and samples $K_{\ell_\rho} \leftarrow_{\mathrm{R}} \mathcal{K}$ instead of using $(C_{\ell_\rho}, K_{\ell_\rho}) \leftarrow \mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell)$. These games are described in Figure 6.9.

Now, we build PPT adversaries $\mathcal{B}_{\rho,0}$ and $\mathcal{B}_{\rho,1}$ against the 1-label-IND security of $\mathcal{SSE}$ such that:

$$|\mathsf{Adv}_{\mathsf{G}^\star_{\rho-1}}(\mathcal{A} \wedge i^\star \neq 0) - \mathsf{Adv}_{\mathsf{H}_{\rho,0}}(\mathcal{A} \wedge i^\star \neq 0)| \leq \mathsf{Adv}^{\mathsf{1\text{-}label\text{-}IND}}_{\mathcal{SSE},\mathcal{B}_{\rho,0}}(\lambda);$$
$$|\mathsf{Adv}_{\mathsf{G}^\star_{\rho}}(\mathcal{A} \wedge i^\star \neq 0) - \mathsf{Adv}_{\mathsf{H}_{\rho,1}}(\mathcal{A} \wedge i^\star \neq 0)| \leq \mathsf{Adv}^{\mathsf{1\text{-}label\text{-}IND}}_{\mathcal{SSE},\mathcal{B}_{\rho,1}}(\lambda).$$

Let $\beta \in \{0,1\}$. We proceed to describe $\mathcal{B}_{\rho,\beta}$. First, $\mathcal{B}_{\rho,\beta}$ samples the guess $i^\star \leftarrow_{\mathrm{R}} \{0,\ldots,n\}$. If $i^\star = 0$, then $\mathcal{B}_{\rho,\beta}$ behaves exactly as the game $\mathsf{G}^\star_{\rho-1+\beta}$. Otherwise, it does the following, using the 1-label-IND security game against $\mathcal{SSE}$:

- First, it generates $(\mathsf{pk}', \mathsf{msk}', (\mathsf{ek}'_i)_{i\in[n]}) \leftarrow \mathsf{Setup}'(1^\lambda)$, and sends $i^\star$ to receive $\mathsf{pk}_{\mathsf{sse}}$ from its own experiment. It returns $\mathsf{pk} = (\mathsf{pk}', \mathsf{pk}_{\mathsf{sse}})$ to the adversary $\mathcal{A}$.
- $\mathsf{OEnc}(i,(x^0,x^1),\ell_j)$: if $j < \rho$, it computes $C = \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x^1, \ell_j)$; if $j > \rho$, it computes $C = \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}'_i, x^0, \ell_j)$; and if $j = \rho$, it computes $C = \mathsf{Enc}'(\mathsf{pk}', \mathsf{ek}_i, x^\beta, \ell_j)$. Then it calls its own oracle to get $S_{\ell_j,i} = \mathsf{OShare}(i, \ell_j)$. If $j \neq \rho$, it computes $(C_{\ell_j}, K_{\ell_j}) \leftarrow \mathsf{SSE.Encaps}(\mathsf{pk}_{\mathsf{sse}}, \ell_j)$, if $j = \rho$ it calls $(C_{\ell_\rho}, K_{\ell_\rho}) \leftarrow \mathsf{OEncaps}(\ell_\rho)$. Finally, it returns the ciphertext $(\mathsf{SEnc}(K_{\ell_j}, C), C_{\ell_j}, S_{\ell_j,i})$.
- $\mathsf{OKeygen}(k)$: it returns $\mathsf{KeyGen}'(\mathsf{msk}', k)$.
- $\mathsf{OCorrupt}(i)$: it uses its own corruption oracle to get $\mathsf{ek}_{\mathsf{sse},i} \leftarrow \mathsf{OCorrupt}(i)$, and returns $\mathsf{ek}_i = (\mathsf{ek}'_i, \mathsf{ek}_{\mathsf{sse},i})$.
- Finally, $\mathcal{B}_{\rho,\beta}$ forwards $\mathcal{A}$'s output α to its own experiment.

Game $\mathsf{G}^\star_\rho$, which encrypts x^1 under ℓ_ρ just differs from $\mathsf{H}_{\rho,1}$ with real vs. random keys K_{ℓ_ρ}, as emulated by $\mathcal{B}_{\rho,1}$, according to the real-or-random behavior of the 1-label-IND game for $\mathcal{SSE}$. Game $\mathsf{G}^\star_{\rho-1}$, which encrypts x^0 under ℓ_ρ just differs from $\mathsf{H}_{\rho,0}$ with real vs. random keys K_{ℓ_ρ}, as emulated by $\mathcal{B}_{\rho,0}$, according to the real-or-random behavior of the 1-label-IND game for $\mathcal{SSE}$. Note that if adversary $\mathcal{A}$ makes queries that satisfy **condition 1** and that the guess $i^\star$ is correct, and different from 0, then the queries of $\mathcal{B}_{\rho,\beta}$ satisfy the conditions required by the 1-label-IND security game for $\mathcal{SSE}$, namely, $\mathsf{OEncaps}$ is only queried on one label ℓ_ρ, $\mathsf{OCorrupt}$ is never queried on $i^\star$, and OShare is never queried on $(i^\star, \ell_\rho)$.

Since the encapsulation keys K_{ℓ_ρ} are uniformly random in games $\mathsf{H}_{\rho,0}$ and $\mathsf{H}_{\rho,1}$, we can use the one-time security of $\mathcal{SKE}$, for each ciphertext for the label ℓ_ρ, to obtain a PPT adversary $\mathcal{B}''_\rho$ such that:

$$|\mathsf{Adv}_{\mathsf{H}_{\rho,0}}(\mathcal{A} \wedge i^\star \neq 0) - \mathsf{Adv}_{\mathsf{H}_{\rho,1}}(\mathcal{A} \wedge i^\star \neq 0)| \leq q_e \cdot \mathsf{Adv}^{\mathsf{OT}}_{\mathcal{SKE},\mathcal{B}''_\rho}(\lambda),$$

where q_e denotes maximum number of ciphertexts generated under a label.

Putting everything together, for the case $i^\star \neq 0$, we obtain PPT adversaries $\mathcal{B}'_\rho$ and $\mathcal{B}''_\rho$ such that:

$$|\mathsf{Adv}_{\mathsf{G}^\star_{\rho-1}}(\mathcal{A} \wedge i^\star \neq 0) - \mathsf{Adv}_{\mathsf{G}^\star_{\rho}}(\mathcal{A} \wedge i^\star \neq 0)| \leq 2 \cdot \mathsf{Adv}^{\mathsf{1\text{-}label\text{-}IND}}_{\mathcal{SSE}}(\mathcal{B}'_\rho) + q_e \cdot \mathsf{Adv}^{\mathsf{OT}}_{\mathcal{SKE}}(\mathcal{B}''_\rho))$$

Since for any game G and any adversary $\mathcal{A}$, $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A}) = \mathsf{Adv}_{\mathsf{G}}(\mathcal{A} \wedge i^\star = 0) + \mathsf{Adv}_{\mathsf{G}}(\mathcal{A} \wedge i^\star \neq 0)$, this concludes the proof of Lemma 42.

□

Decentralizing MCFE

In decentralized MCFE, the master secret key msk is split into $[n]$ secret keys sk_i, on for each client and the generation of the functional decryption keys is distributed among the clients. We focus on non-interactive protocols to generate the decryption keys, namely, clients can first run independently an algorithm $\mathsf{KeyGenShare}$ that only requires the secret key ek_i, and that generates a partial key. Then, all these partial decryption keys can be combined via $\mathsf{KeyComb}$, that only requires the public key. This way, there is no need for different clients to interact with each other. The master secret key is only used during the setup. See Definition 26 for further details.

The correctness property essentially states the combined key corresponds to the functional decryption key. The security model is quite similar to the one for MCFE, except that

- for the KeyGen protocol: the adversary has access to transcripts of the communications, thus modeled by a query $\mathsf{OKeyShare}(i, f)$ that executes $\mathsf{KeyGenShare}(\mathsf{sk}_i, f)$.
- corruption queries additionally reveal the secret keys sk_i;
- the distributed key generation must guarantee that without all the shares, no information is known about the functional decryption key.

Distributed Sum

In the MCFE for inner product from Section 6.1 the functional decryption keys are of the form $\mathsf{dk}_{\mathbf{y}} = \left(\mathbf{y}, \sum_i \mathbf{S}_i^\top \mathbf{y}_i\right)$, and $\mathsf{msk} = \{\mathbf{S}_i\}_{i\in[n]}$. We split the master secret key into $\mathsf{sk}_i := \mathbf{S}_i$ for all $i \in [n]$, and we use a non-interactive prototol to compute the sum of all the $\mathbf{S}_i^\top \mathbf{y}_i$, each of which can be computed by each client $i \in [n]$ independently.

The same protocol can be used to decentralize the setup of the SSE scheme from Section 6.3, since the public key $\mathsf{pk}_{\mathsf{sse}}$ contains $[\sum_i t_i]_2$.. In this section, we present such a protocol that is similar to [KDK11].

Definition 30: Ideal Protocol DSum

A DSum on abelian groups G, G' among n senders is defined by three algorithms:

- $\mathsf{DSSetup}(1^\lambda)$: Takes as input the security parameter 1^λ. Generates the public parameters pp and the personal secret keys sk_i for all $i \in [n]$.
- $\mathsf{DSEncode}(x_i, \ell, \mathsf{sk}_i)$: Takes the group element $x_i \in G$ to encode, a label ℓ, and the personal secret key sk_i of the user i. Returns the share $M_{\ell,i} \in G'$
- $\mathsf{DSCombine}(\{M_{\ell,i}\}_{i\in[n]})$: Takes the shares $\{M_{\ell,i}\}_{i\in[n]}$, and returns the value $\sum_i M_{\ell,i} \in G'$.

Correctness. For any label ℓ, we want $\Pr[\mathsf{DSCombine}(\{M_{\ell,i}\}_{i\in[n]}) = \sum_i x_i] = 1$, where the probability is taken over $M_{\ell,i} \leftarrow \mathsf{DSEncode}(x_i, \ell, \mathsf{sk}_i)$ for all $i \in [n]$, and $(\mathsf{pp}, (\mathsf{sk}_i)_i) \leftarrow \mathsf{DSSetup}(1^\lambda)$.

Security Notion. This protocol must guarantee the privacy of the x_i's, possibly excepted their sum when all the shares are known. This is the classical security notion for multi-party computation, where the security proof is performed by simulating the view of the adversary from the output of the result: nothing when not all the shares are asked, and just the sum of the inputs when all the shares are queried. We also have to deal with the corruptions, which give the users' secret keys.

Our DSum Protocol

We present a DSum protocol for n users, with groups $G = G' = \mathbb{Z}_p^m$. The security relies on the CDH assumption in a group $\mathbb{G}$ of primer order p. Similar protocol can be found in [KDK11].

- $\mathsf{DSSetup}(1^\lambda)$: generates $\mathcal{G} := (\mathbb{G}, p, P) \leftarrow \mathsf{GGen}(1^\lambda)$, and a hash function H onto $\mathbb{Z}_p^m$. For all $i \in [n]$, $t_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $\mathsf{sk}_i := t_i$, $\mathsf{pp} := (\mathcal{G}, \mathsf{H}, ([t_i])_i)$. It returns $\mathsf{pp}, \{\mathsf{sk}_i\}_{i\in[n]}$.

- $\mathsf{DSEncode}(\mathbf{x}_i \in \mathbb{Z}_p^m, \ell, \mathsf{sk}_i)$: computes $\mathbf{h}_{\ell,i,j} = \mathsf{H}([t_{\min\{i,j\}}], [t_{\max\{i,j\}}], t_i \cdot [t_j], \ell) = \mathbf{h}_{\ell,j,i} \in \mathbb{Z}_p^m$ for all $i, j \in [n]$, and returns:

$$M_{\ell,i} = \mathbf{x}_i - \sum_{j<i} \mathbf{h}_{\ell,i,j} + \sum_{j>i} \mathbf{h}_{\ell,i,j}.$$

- $\mathsf{DSCombine}(\{M_{\ell,i}\}_{i\in[n]})$: returns $\sum_i M_{\ell,i}$.

Correctness. The correctness should show that the sum of the shares is equal to the sum of the x_i's: the former is equal to

$$\begin{aligned}\sum_i \left(\mathbf{x}_i - \sum_{j<i} \mathbf{h}_{\ell,i,j} + \sum_{j>i} \mathbf{h}_{\ell,i,j}\right) &= \sum_i \mathbf{x}_i - \sum_i \sum_{j<i} \mathbf{h}_{\ell,i,j} + \sum_i \sum_{j>i} \mathbf{h}_{\ell,j,i} \\ = \sum_i \mathbf{x}_i - \sum_i \sum_{j<i} \mathbf{h}_{\ell,i,j} + \sum_j \sum_{i<j} \mathbf{h}_{\ell,j,i} &= \sum_i \mathbf{x}_i\end{aligned}$$

Security Analysis

We will prove that there exists a simulator that generates the view of the adversary from the output only. In this proof, we will assume static corruptions (the set $\mathcal{CS}$ of the corrupted clients is known from the beginning) and the hardness of the CDH problem. However, this construction will only tolerate up to $n-2$ corruptions, so that there are at least 2 honest users. But this is also the case for the MCFE.

W.l.o.g., we can assume that $\mathcal{HS} = \{1, \ldots, n-c\}$ and $\mathcal{CS} = \{n-c+1, \ldots, n\}$, by simply reordering the clients, when $\mathcal{CS}$ is known. We will gradually modify the behavior of the simulator, with less and less powerful queries. At the beginning, the DSEncode-query takes all the same inputs as in the real game, including the secret keys. At the end, it should just take the sum (when all the queries have been asked), as well as the corrupted $\mathbf{x}_j$'s.

Game G_0: The simulator runs as in the real game, with known $\mathcal{CS}$.

Game G_1: The simulator is given a pair $([t], [t^2])$.

- DSSetup: for all $1 \le i \le n-c$: $\alpha_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $[t_i] := [t + \alpha_i]$. For all $n-c < i \le n$: $t_i \leftarrow \mathbb{Z}_p$. For all $1 \le i, j \le n-c$, $Y_{i,j} := [t^2 + (\alpha_i + \alpha_j) \cdot t + \alpha_i\alpha_j]$. For all $1 \le i \le n-c$, and $n-c < j \le n$, $Y_{i,j} := [(t + \alpha_i)t_j]$, and $Y_{j,i} = Y_{i,j}$. For all $n-c < i, j \le n$, $Y_{i,j} := [t_i \cdot t_j]$. It returns $\mathsf{pp} := \{[t_i]\}_{i\in[n]}$ and the secret keys t_i of the corrupted users.

- $\mathsf{DSEncode}(\mathbf{x}_i, \ell)$: the simulator generates all the required $\mathbf{h}_{\ell,i,j}$ using the X_j's and $Y_{i,j}$'s, querying the hash function, and returns $M_{\ell,i} = \mathbf{x}_i - \sum_{j<i} \mathbf{h}_{\ell,i,j} + \sum_{j>i} \mathbf{h}_{\ell,i,j}$.

Game G_2: The simulator does as above, but just uses a random $[t'] \leftarrow_{\mathrm{R}} \mathbb{G}$ instead of $[t^2]$, to answer the $\mathsf{DSEncode}$-queries.

This can make a difference for the adversary if the latter asks for the hash function on some tuple $(X_{\min\{i,j\}}, X_{\max\{i,j\}}, [t_i \cdot t_j], \ell)$, for $i, j \leq n-c$, as this will not be the value $\mathbf{h}_{\ell,i,j}$, which has been computed using $Y_{i,j} \neq [t_i \cdot t_j]$. In such a case, one can find $[t_i \cdot t_j] = [t^2 + (\alpha_i + \alpha_j) \cdot t + \alpha_i \alpha_j$ in the list of the hash queries, and thus extract $t^2 = [t^2]$. As a consequence, under the hardness of the square Diffie-Hellman problem (which is equivalent to the CDH problem), this simulation is indistinguishable from the previous one.

Game G_3: The simulator does as above excepted for the $\mathsf{DSEncode}$-queries. If this is not the last-honest query under label ℓ, the simulator returns $M_{\ell,i} = -\sum_{j<i} \mathbf{h}_{\ell,i,j} + \sum_{j>i} \mathbf{h}_{\ell,i,j}$; for the last honest query, it returns $M_{\ell,i} = S_H - \sum_{j<i} \mathbf{h}_{\ell,i,j} + \sum_{j>i} \mathbf{h}_{\ell,i,j}$, where $S_H = \sum_{j \in \mathcal{HS}} \mathbf{x}_j$.

Actually, for a label ℓ, if we denote i_ℓ the index of the honest player involved in the last query, the view of the adversary is exactly the same as if, for every $i \neq i_\ell$, we have replaced $\mathbf{h}_{\ell,i,i_\ell}$ by $\mathbf{h}_{\ell,i,i_\ell} + \mathbf{x}_i$ (if $i_\ell > i$) or by $\mathbf{h}_{\ell,i,i_\ell} - \mathbf{x}_i$ (if $i_\ell < i$). We thus replace uniformly distributed variables by other uniformly distributed variables: this simulation is perfectly indistinguishable from the previous one.

Game G_4: The simulator now ignores the values $\mathbf{h}_{\ell,i,j}$ for honest i, j. But for each label, it knows the corrupted $\mathbf{x}_j$'s, and can thus compute the values $M_{\ell,j}$ for the corrupted users, using the corrupted $\mathbf{x}_j$'s and secret keys. If this is not the last honest query, it returns a random $M_{\ell,i}$. For the last honest query, knowing $S = \sum_j \mathbf{x}_j$, it outputs $M_{\ell,i} = S - \sum_{j \neq i} M_{\ell,j}$.

As in the previous analysis, if one first sets all the $\mathbf{h}_{\ell,i,j}$, for $j \neq i_\ell$, this corresponds to define $\mathbf{h}_{\ell,i,i_\ell}$ from $M_{\ell,i}$, for $i \neq i_\ell$.

Application to DMCFE for Inner Products

One can convert the MCFE from Section 6.1 whose decryption keys are of the form $\sum_i \mathbf{S}_i^\top \mathbf{y}_i$ into an decentralized MCFE. Each client computes $\mathbf{S}_i^\top \mathbf{y}_i$ independently, and we use the DSum protocol to compute the sum, where the label is the vector $\mathbf{y}$ itself. Namely, we have:

- $\mathsf{KeyGenShare}(\mathsf{sk}_i, \mathbf{y} := (\mathbf{y}_1 \| \cdots \| \mathbf{y}_n))$: outputs $M_{\mathbf{y},i} \leftarrow \mathsf{DSEncode}(\mathbf{S}_i^\top \mathbf{y}_i, \mathbf{y}, \mathsf{sk}_i)$;
- $\mathsf{KeyComb}((M_{\mathbf{y},i})_{i \in [n]}, \mathbf{y})$: outputs $\mathsf{dk}_\mathbf{y} = (\mathbf{y}, \mathbf{d}_\mathbf{y})$, where $\mathbf{d}_\mathbf{y}$ is publicly computed as $\mathsf{DSCombine}(\{M_{\mathbf{y},i}\}_{i \in [n]})$;

Using the last simulation game, we can now show that all the $\mathsf{KeyGenShare}(\mathsf{sk}_i, \mathbf{y})$ are first simulated at random, and just the last query needs to ask the KeyGen-query to the MCFE scheme to get the sum and program the output. Hence, unless all the honest queries are asked, the functional decryption key is unknown.

Consequently, we can convert the MCFE from Section 6.1 into a decentralized MCFE. Note that the transformation from Section 6.2 and Section 6.4, which remove the one challenge ciphertext restriction, and the incomplete ciphertext restriction, respectively, preserve the decentralized feature of the DCMFE obtained from using the DSum on the MCFE from Section 6.1. At the end, combining all transformations, we obtain a decentralized MCFE for inner product that is many-AD-IND secure.

Decentralizing the setup. Note that the setup of the MCFE from Section 6.1 is already decentralized, in the sense that each $\mathsf{ek}_i, \mathsf{msk}_i$ can be generated independently for all $i \in [n]$, and dynamically (the users only have to agree on a particular group and hash function to use). Applying the transformation from Section 6.2 preserves that feature, since an independent single-input FE is used for each slot $i \in [n]$. Finally, the SSE from Section 6.3 can have a distributed setup if we use a DSum protocol to compute the value $[\sum_i t_i]_2$ from the public key $\mathsf{pk}_{\mathsf{sse}}$. Consequently, we obtain a scheme where there is no need of a trusted authority.

Chapter 7

Functional Encryption for Quadratic Functions

In this section, we present the first public-key FE scheme based on a standard assumption that supports a functionality beyond inner product, or predicates. In our scheme, ciphertexts are associated with a set of values, and secret keys are associated with a degree-two polynomial. This way, the decryption of a ciphertext $\mathsf{ct}_{(x_1,\ldots,x_n)\in\mathbb{Z}_p^n}$ with a secret key $\mathsf{dk}_{P\in\mathbb{Z}_p[X_1,\ldots,X_n],\mathsf{deg}(P)\leq 2}$ recovers $P(x_1,\ldots,x_n)$. The ciphertext size is $O(n)$ group elements, improving upon [ABDP15, ALS16], which would require $O(n^2)$ group elements, since they build an FE scheme for inner product. Our FE scheme is proved selectively secure under the Matrix Diffie-Hellman assumption [EHK+13], which generalizes standard assumptions such as DLIN or k-Lin for $k \geq 1$, and the 3-PDDH assumption [BSW06]. Constructions whose security is justified in the generic group model can be found in [BCFG17, DGP18]. See also [Lin17, AS17] for private-key variants. The state of the art for functional encryption for quadratic functions is summarized in Figure 7.1.

Overview of our construction

The difficulty is to have ciphertexts $\mathsf{ct}_{(x_1,\ldots,x_n)}$ of $O(n)$ group elements, that must hide the message $(x_1,\ldots,x_n) \in \mathbb{Z}_p^n$, but still contain enough information to recover the n^2 values $x_i \cdot x_j$ for $i,j \in [n]$. To ensure the message is hidden, the ciphertext will contain an encryption of each value x_i. Since we want to multiply together these encryptions to compute products $x_i \cdot x_j$, and since these encryption are composed of group elements, we require a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are additively written, prime-order groups. Namely, decryption pairs encrypted values in $\mathbb{G}_1$ with encrypted values in $\mathbb{G}_2$. For this reason, it makes sense to re-write

References	security	public or private key
[AS17]	sel. GGM	private-key
[Lin17]	sel. SXDH	private-key
[BCFG17, DGP18]	ad. GGM	public-key
[BCFG17]	sel. standard	public-key

Figure 7.1: Existing functional encryption for quadratic functions. Here, ad. and sel. denote adaptive and selecive security respectively, SXDH stands for Symmetric eXternal Diffie Hellman assumption, and GGM stands for Generic Group Model.

the function as: $\mathcal{X} := \mathbb{Z}_p^n \times \mathbb{Z}_p^m$, $\mathcal{K} := \mathbb{Z}_p^{n \cdot m}$, and for all $(\mathbf{x}, \mathbf{y}) \in \mathcal{X}$, $\alpha \in \mathcal{K}$,

$$F((\mathbf{x}, \mathbf{y}), \alpha) = \sum_{i \in [n], j \in [m]} \alpha_{i,j} x_i y_j.$$

Private-key, one-ciphertext secure FE. Our starting point is a private-key FE for inner product, that is only secure for one challenge ciphertext:

$$\mathsf{ct}_{(\mathbf{x},\mathbf{y})} := \{[\mathbf{A}\mathbf{r}_i + \mathbf{b}^\perp x_i]_1\}_{i \in [n]}, \{[\mathbf{B}\mathbf{s}_j + \mathbf{a}^\perp y_j]_2\}_{j \in [m]}, \ \mathsf{dk}_\alpha := [\sum_{i,j} \alpha_{i,j} \mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j]_T,$$

where $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, and $(\mathbf{A}|\mathbf{b}^\perp)$, $(\mathbf{B}|\mathbf{a}^\perp)$ are bases of $\mathbb{Z}_p^{k+1}$ such that $\mathbf{a}^\perp \in \mathsf{orth}(\mathbf{A})$ and $\mathbf{b}^\perp \in \mathsf{orth}(\mathbf{B})$, à la [CGW15]. The vectors $[\mathbf{A}\mathbf{r}_i]_1$ and $[\mathbf{B}\mathbf{s}_j]_2$ for $i \in [n], j \in [m]$, $\mathbf{a}^\perp$ and $\mathbf{b}^\perp$ are part of a master secret key, used to (deterministically) generate $\mathsf{ct}_{\mathbf{x},\mathbf{y}}$ and dk_α. Correctness follows from the orthogonality property: decryption computes $\sum_{i,j} \alpha_{i,j} e([\mathbf{A}\mathbf{r}_i + \mathbf{b}^\perp x_i]_1^\top, [\mathbf{B}\mathbf{s}_j + \mathbf{a}^\perp y_j]_2) = \mathsf{dk}_{\mathbf{F}} + (\mathbf{a}^\perp)^\top \mathbf{b}^\perp \cdot [F(\mathbf{F}, (\mathbf{x}, \mathbf{y}))]_T$, from which one can extract $F(\alpha, (\mathbf{x}, \mathbf{y}))$ since $[(\mathbf{a}^\perp)^\top \mathbf{b}^\perp]_T$ is public, simply by enumerating all the possible values for $F(\alpha, (\mathbf{x}, \mathbf{y}))$. This is efficient as long as the output always lies in a polynomial size domain.

Security relies on the $\mathcal{D}_k$-MDDH Assumption [EHK+13], which stipulates that given $[\mathbf{A}]_1, [\mathbf{B}]_2$ drawn from a matrix distribution $\mathcal{D}_k$ over $\mathbb{Z}_p^{(k+1) \times k}$,

$$[\mathbf{A}\mathbf{r}]_1 \approx_c [\mathbf{u}]_1 \approx_c [\mathbf{A}\mathbf{r} + \mathbf{b}^\perp]_1 \text{ and } [\mathbf{B}\mathbf{s}]_2 \approx_c [\mathbf{v}]_2 \approx_c [\mathbf{B}\mathbf{s} + \mathbf{a}^\perp]_2,$$

where $\mathbf{r}, \mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, and $\mathbf{u}, \mathbf{v} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$. This allows us to change $\mathsf{ct}_{(\mathbf{x}^{(0)},\mathbf{y}^{(0)})}$ into $\mathsf{ct}_{(\mathbf{x}^{(1)},\mathbf{y}^{(1)})}$, but creates an extra term $\left[\mathbf{x}^{(1)\top}\mathbf{F}\mathbf{y}^{(1)} - \mathbf{x}^{(0)\top}\mathbf{F}\mathbf{y}^{(0)}\right]_T$ in the secret keys dk_α. We conclude the proof using the fact that for all the α queried to $\mathsf{OKeygen}$, $F(\alpha, (\mathbf{x}^{(0)}, \mathbf{y}^{(0)})) = F(\alpha, (\mathbf{x}^{(1)}, \mathbf{y}^{(1)}))$, as required by the security definition for FE (see Definition 19), which cancels out the extra term in all secret keys.

Public-key FE. We now present how to obtain to modify this simple scheme to obtain a public-key FE.

- In the public-key setting, for the encryption to compute $[\mathbf{A}\mathbf{r}_i + \mathbf{b}^\perp x_i]$ and $[\mathbf{B}\mathbf{s}_j + \mathbf{a}^\perp y_j]$ for $i \in [n], j \in [m]$ and any $\mathbf{x} \in \mathbb{Z}_p^n, \mathbf{y} \in \mathbb{Z}_p^m$, the vectors $[\mathbf{a}^\perp]_2$ and $[\mathbf{b}^\perp]_1$ would need to be part of the public key, which is incompatible with the MDDH assumption on $[\mathbf{A}]_1$ or $[\mathbf{B}]_2$. To solve this problem, we add an extra dimension, namely, we use bases $\left(\begin{array}{c|c} \mathbf{A}|\mathbf{b}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)$ and $\left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)$ where the extra dimension will be used for correctness, while $(\mathbf{A}|\mathbf{b}^\perp)$ and $(\mathbf{B}|\mathbf{a}^\perp)$ will be used for security (using the MDDH assumption, since $\mathbf{a}^\perp$ and $\mathbf{b}^\perp$ are not part of the public key anymore).
- To avoid mix and match attacks, the encryption randomizes the bases

$$\left(\begin{array}{c|c} \mathbf{A}|\mathbf{b}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \text{ and } \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)$$

  into

$$\mathbf{W}^{-1}\left(\begin{array}{c|c} \mathbf{A}|\mathbf{b}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \text{ and } \mathbf{W}^\top\left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)$$

  for $\mathbf{W} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}$ a random invertible matrix. This "glues" the components of a ciphertext that are in G_1 to those that are in G_2.
- We randomize the ciphertexts so as to contain $[\mathbf{A}\mathbf{r}_i \cdot \gamma]_1$ and $[\mathbf{B}\mathbf{s}_j \cdot \sigma]_2$, where $\gamma, \sigma \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ are the same for all $i \in [n]$, and $j \in [m]$, but fresh for each ciphertext. The ciphertexts also contain $[\gamma \cdot \sigma]_1$, for correctness.

Related works. We note that the techniques used here share some similarities with Dual Pairing Vector Space constructions (e.g., [OT08, OT09, Lew12, CLL+13]). In particular, our produced ciphertexts and private keys are distributed as in their corresponding counterparts in [OT08]. The similarities end here though. These previous constructions all rely on the Dual System Encryption paradigm [Wat09], where the security proof uses a hybrid argument over all secret keys, leaving the distribution of the public key untouched. Our approach, on the other hand, manages to avoid this inherent security loss by changing the distributions of *both* the secret and public keys. Our approach also differs from [BSW06] and follow-up works [BW06, GKSW10] in that they focus on the comparison predicate, a function that can be expressed via a quadratic function that is significantly simpler than those considered here. Indeed, for the case of comparisons predicates it is enough to consider vectors of the form: $[\mathbf{A}\mathbf{r}_i + x_i\mathbf{b}^\perp]_1, [\mathbf{B}\mathbf{s}_j + y_j\mathbf{a}^\perp]_2$, where x_i and y_j are either 0, or some random value (fixed at setup time, and identical for all ciphertexts and secret keys), or are just random garbage.

The work of [Lin17, AS17] present constructions of private-key functional encryption schemes for degree-D poly- nomials based on D-linear maps. As a special case for D = 2, these schemes support quadratic polynomials from bilinear maps, as ours. Also, in terms of security, the construction of [Lin17] is proven selectively secure based on the SXDH assumption, while the scheme of [AS17] is selectively secure based on ad-hoc assumptions that are justified in the multilinear group model.

In comparison to these works, our scheme has the advantage of working in the (arguably more challenging) public key setting. [BCFG17] also gave an adaptively secure construction in the generic group model. We only present the construction whose security is based on standard assumption. Namely, we start by giving the private-key FE whose security only handles one challenge ciphertext. We then present the full-fledged public-key FE.

Private-key FE with one-SEL-IND security

We give in Figure 7.2 a private-key FE for quadratic functions, that is, the functionality $F_{\mathsf{quad}}^{K,X,Y} : \mathcal{K} \times \mathcal{X} \to \mathcal{Z}$, with $\mathcal{K} := [0, K]^{nm}$, $\mathcal{X} := [0, X]^n \times [0, Y]^m$, $\mathcal{Z} := [0, nmKXY]$, such that for any $\alpha \in \mathcal{K}$, $(\mathbf{x}, \mathbf{y}) \in \mathcal{X}$, we have:

$$F_{\mathsf{quad}}^{K,X,Y}(\alpha, (\mathbf{x}, \mathbf{y})) = \sum_{i,j} \alpha_{i,j} x_i y_j.$$

For correctness, we require that $nmKXY$ is of polynomial size in the security parameter. The one-SEL-SIM security relies on the $\mathcal{D}_k(p)$-MDDH assumption in asymmetric pairing groups.

Correctness. For any $(\mathbf{x}, \mathbf{y}) \in \mathcal{X}$, $i \in [n], j \in [m]$, we have:

$$e([\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2) = [\mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j + (\mathbf{b}^\perp)^\top \mathbf{a}^\perp x_i y_j]_T,$$

since $\mathbf{A}^\top \mathbf{a}^\perp = \mathbf{B}^\top \mathbf{b}^\perp = \mathbf{0}$. Therefore, for any $(\alpha_{i,j})_{i,j} \in \mathcal{K}$, the decryption computes

$$\begin{aligned} D &:= [\sum_{i,j} \alpha_{i,j} \mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j + \sum_{i,j} \alpha_{i,j} x_i y_j \cdot (\mathbf{b}^\perp)^\top \mathbf{a}^\perp]_T - e(K, [1]_2) - e([1]_1, \widehat{K}) \\ &= \sum_{i,j} \alpha_{i,j} x_i y_j \cdot [(\mathbf{b}^\perp)^\top \mathbf{a}^\perp]_T. \end{aligned}$$

Note that $(\mathbf{b}^\perp)^\top \mathbf{a}^\perp \neq 0$ with probability $1 - \frac{1}{\Omega(p)}$ over the choices of $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A})$, and $\mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})$ (see Definition 9). Therefore, one can enumerate all possible $v \in \mathcal{Z}$ and check if $v \cdot [(\mathbf{b}^\perp)^\top \mathbf{a}^\perp]_T = D$. This can be done in time $|\mathcal{Z}| = nmKXY + 1$, which is of polynomial size in the security parameter.

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{quad}}^{K,X,Y})}$:
$\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e) \leftarrow_{\mathrm{R}} \mathsf{PGGen}(1^\lambda)$
Return $\mathsf{gpk} := \mathcal{PG}$

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F)}$:
$\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})$. For $i \in [n], j \in [m]$, $\mathbf{r}_i, \mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$.
$\mathsf{ek} := ([\mathbf{A}]_1, [\mathbf{b}^\perp]_1, [\mathbf{B}]_2, [\mathbf{a}^\perp]_2)$, $\mathsf{msk} := \left(\mathbf{A}, \mathbf{a}^\perp, \mathbf{B}, \mathbf{b}^\perp, \{\mathbf{r}_i, \mathbf{s}_j\}_{i\in[n], j\in[m]}\right)$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{ek}, (\mathbf{x}, \mathbf{y}) \in \mathcal{X})}$:
For $i \in [n]$: $\mathbf{c}_i := \mathbf{A}\mathbf{r}_i + \mathbf{b}^\perp x_i$,
For $j \in [m]$: $\widehat{\mathbf{c}}_j := \mathbf{B}\mathbf{s}_j + \mathbf{a}^\perp y_j$,
$\mathsf{ct}_{(\mathbf{x},\mathbf{y})} := \{[\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2\}_{i\in[n], j\in[m]}$
Return $\mathsf{ct}_{(\mathbf{x},\mathbf{y})} \in \mathbb{G}_1^{n(k+1)} \times \mathbb{G}_2^{m(k+1)}$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \alpha \in \mathcal{K})}$:
$u \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $K := [\sum_{i\in[n], j\in[m]} \alpha_{i,j} \mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j]_1 - [u]_1$, $\widehat{K} := [u]_2$
Return $\mathsf{dk}_\alpha := (K, \widehat{K}) \in \mathbb{G}_1 \times \mathbb{G}_2$

$\underline{\mathsf{Dec}(\mathsf{gpk}, \mathsf{ct}_{(\mathbf{x},\mathbf{y})}, \mathsf{dk}_\alpha)}$:
$D := \sum_{i\in[n], j\in[m]} \alpha_{i,j} \cdot e([\mathbf{c}_i]_1^\top, [\widehat{\mathbf{c}}_j]_2) - e(K, [1]_2) - e([1]_1, \widehat{K})$.
Return $v \in \mathbb{Z}_p$ such that $[v \cdot (\mathbf{b}^\perp)^\top \mathbf{a}^\perp]_T = D$.

Figure 7.2: $\mathsf{FE}_{\mathsf{one}}$, a private-key FE for inner product, selectively secure under the $\mathcal{D}_k(p)$-MDDH assumption in asymmetric pairing groups.

Theorem 19: one-SEL-IND security

The FE from Figure 7.2 is one-SEL-IND secure under the $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$.

Remark 14: one-SEL-SIM security

WE note that the FE from Figure 7.2 is in fact one-SEL-SIM secure, which implies one-SEL-IND security. This is clear from the fact that in the last hybrid game in the proof of Theorem 19, the simulator is only required to know the value $\alpha_{i,j} x_i y_j$. Since we only need one-SEL-IND for our public-key FE, which is the main focus of this chapter, we omit the one-SEL-SIM security proof of the private-key FE.

Proof of Theorem 19. We use a sequence of hybrid games defined in Figure 7.3. Let $\mathcal{A}$ be a PPT adversary. For any game G, we denote by $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ the probability that the game G returns 1 when interacting with $\mathcal{A}$.

Note that we have: $\mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND}}_{\mathsf{FE}_{\mathsf{one}}}(\mathcal{A}) = 2 \times |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - 1/2|$. This follows from the fact that for all $i \in [n]$, $j \in [m]$, we have:

$$\mathbf{c}_i^\top \widehat{\mathbf{c}}_j = \mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j + x_i^{(\beta)} y_j^{(\beta)} (\mathbf{b}^\perp)^\top \mathbf{a}^\perp.$$

$\underline{\mathsf{G}_0, \boxed{\mathsf{G}_1}, \boxed{\boxed{\mathsf{G}_2}}}$:
$(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)})) \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e) \leftarrow_{\mathrm{R}} \mathsf{GGen}(1^\lambda)$, $\mathsf{gpk} := \mathcal{PG}$, $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})$, $\mathsf{pk} := [(\mathbf{a}^\perp)^\top \mathbf{b}^\perp]_T$, $\beta \leftarrow_{\mathrm{R}} \{0,1\}$. For $i \in [n], j \in [m]$: $\mathbf{r}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
$\mathbf{c}_i := \mathbf{A}\mathbf{r}_i + x_i^{(\beta)} \mathbf{b}^\perp$, $\boxed{\mathbf{c}_i \leftarrow_{\mathrm{R}} \mathbb{G}_1^{k+1}}$
$\widehat{\mathbf{c}}_j := \mathbf{B}\mathbf{s}_j + y_j^{(\beta)} \mathbf{a}^\perp$, $\boxed{\widehat{\mathbf{c}}_j \leftarrow_{\mathrm{R}} \mathbb{G}_2^{k+1}}$
$\mathsf{ct} := \{[\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_1\}_{i \in [n], j \in [m]}$
$\beta' \leftarrow \mathcal{A}^{\mathsf{OKeygen}(\cdot)}(\mathsf{gpk}, \mathsf{pk}, \mathsf{ct})$
Return 1 if $\beta' = \beta$, 0 otherwise.

$\underline{\mathsf{OKeygen}(\alpha \in \mathcal{K})}$:
$u \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $K := [\sum_{i,j} \alpha_{i,j} \mathbf{c}_i^\top \widehat{\mathbf{c}}_j]_1 - [\sum_{i,j} \alpha_{i,j} x_i^{(\beta)} y_j^{(\beta)} \cdot (\mathbf{b}^\perp)^\top \mathbf{a}^\perp]_1 - [u]_1$, $\widehat{K} := [u]_2$
Return $\mathsf{dk}_\alpha := (K, \widehat{K})$

Figure 7.3: Games for the proof of Theorem 19. In each procedure, the components inside a solid (dotted) frame are only present in the games marked by a solid (dotted) frame.

Thus, in game G_0, for all $\alpha \in \mathbb{Z}_p^{n \times m}$, $\mathsf{OKeygen}(\alpha)$ computes:

$$\begin{aligned} K &:= \sum_{i,j} \alpha_{i,j} [\mathbf{c}_i^\top \widehat{\mathbf{c}}_j]_1 - [\sum_{i,j} \alpha_{i,j} x_i^{(\beta)} y_j^{(\beta)} \cdot (\mathbf{b}^\perp)^\top \mathbf{a}^\perp]_1 - [u]_1 \\ &= \sum_{i,j} \alpha_{i,j} [\mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B} \mathbf{s}_j]_1 - [u]_1. \end{aligned}$$

Game G_1: is the same as game G_0 except that the vectors $[\mathbf{c}_i]$ from the challenge ciphertext are uniformly random over $\mathbb{G}_1^{k+1}$. In Lemma 43 we show that G_0 is computationally indistinguishable from G_1 under the $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}_1$.

Game G_2: is the same as game G_1 except that the vectors $\widehat{\mathbf{c}}_j$ from the challenge ciphertext are uniformly random over $\mathbb{G}_2^{k+1}$. In Lemma 44 we show that G_1 is computationally indistinguishable from G_2 under the $\mathcal{D}_k(p)$-MDDH assumption. Finally, we show in Lemma 45 that the adversary's view in this game is independent of the bit β, and thus the adversary's advantage in this game is zero, which concludes the proof. □

Lemma 43: From game G_0 to G_1

There exists a PPT adversary $\mathcal{B}_0$ such that

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_1, \mathcal{B}_0}(\lambda) + 2^{-\Omega(\lambda)}.$$

Proof of Lemma 43. Here, we use the $\mathcal{D}_k(p)$-MDDH assumption on $[\mathbf{A}]_1$ to change the distribution of the challenge ciphertext, after arguing that one can simulate the game without knowing $\mathbf{a}^\perp$ or $[\mathbf{A}]_2$.

Namely, we build a PPT adversary $\mathcal{B}_0'$ against the n-fold $\mathcal{D}_k$-MDDH assumption in $\mathbb{G}_1$ such that $|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{n\text{-}\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_1, \mathcal{B}_0}(\lambda) + 2^{-\Omega(\lambda)}$. Then, by Lemma 1, this implies the existence of a PPT adversary $\mathcal{B}_0$ such that $|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_1, \mathcal{B}_0}(\lambda) + 2^{-\Omega(\lambda)}$.

$\underline{\mathcal{B}'_0(\mathcal{PG}, [\mathbf{A}]_1, [\mathbf{h}_1|\cdots|\mathbf{h}_n]_1)}$:
$\big((\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)})\big) \leftarrow \mathcal{A}(1^\lambda)$
$\mathsf{gpk} := \mathcal{PG}$, $\mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})$, $\mathbf{z} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$, $\mathsf{pk} := [(\mathbf{b}^\perp)^\top \mathbf{z}]_T$, $\beta \leftarrow_{\mathrm{R}} \{0,1\}$. For all $j \in [m]$: $\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\widehat{\mathbf{c}}_j := \mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{z}$. For all $i \in [n]$: $\mathbf{c}_i := \mathbf{h}_i + x_i^{(\beta)}\mathbf{b}^\perp$, $\mathsf{ct} := \{[\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2\}_{i\in[n], j\in[m]}$
$\beta' \leftarrow \mathcal{A}(\mathsf{gpk}, \mathsf{pk}, \mathsf{ct})$
Return 1 if $\beta' = \beta$, 0 otherwise.

$\underline{\mathsf{OKeygen}(\alpha \in \mathbb{Z}_p^{n\times m})}$:
$u \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $K := \sum_{i,j} \alpha_{i,j}[\mathbf{c}_i^\top \widehat{\mathbf{c}}_j]_1 - [u]_1 - \sum_{i,j} \alpha_{i,j} x_i^{(\beta)} y_j^{(\beta)} \cdot [(\mathbf{b}^\perp)^\top \mathbf{z}]_1$, $\widehat{K} := [u]_2$
Return $\mathsf{dk}_\alpha := (K, \widehat{K})$

Figure 7.4: Adversary $\mathcal{B}'_0$ against the n-fold $\mathcal{D}_k(p)$-MDDH assumption, for the proof of Lemma 43.

Adversary $\mathcal{B}'_0$ simulates the game to $\mathcal{A}$ as described in Figure 7.4. We show that when $\mathcal{B}'_0$ is given a real MDDH challenge, that is, $[\mathbf{h}_1|\cdots|\mathbf{h}_n]_1 := [\mathbf{AR}]$ for $\mathbf{R} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times n}$, then it simulates the game G_0, whereas it simulates the game G_1 when given a fully random challenge, i.e. when $[\mathbf{h}_1|\cdots|\mathbf{h}_n]_1 \leftarrow_{\mathrm{R}} \mathbb{G}_1^{(k+1)\times n}$, which implies the lemma.

We use the following facts.

1. For all $\mathbf{s} \in \mathbb{Z}_p^k$, $\mathbf{B} \in \mathbb{Z}_p^{(k+1)\times k}$, $\mathbf{b}^\perp \in \mathsf{orth}(\mathbf{B})$, and $\mathbf{a}^\perp \in \mathbb{Z}_p^{k+1}$, we have:
$$(\mathbf{b}^\perp)^\top \mathbf{a}^\perp = (\mathbf{b}^\perp)^\top (\mathbf{B}\mathbf{s} + \mathbf{a}^\perp).$$

2. For all $y_j^{(\beta)} \in \mathbb{Z}_p$, $\mathbf{s} \in \mathbb{Z}_p^k$:
$$\Big(\{\mathbf{s}_j\}_{j\in[m]}\Big)_{\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k} \equiv \Big(\{\mathbf{s}_j + y_j^{(\beta)}\mathbf{s}\}_{j\in[m]}\Big)_{\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k}.$$

3.
$$\Big(\mathbf{B}\mathbf{s} + \mathbf{a}^\perp\Big)_{\mathbf{A},\mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k} \approx_{\frac{1}{\Omega(p)}} (\mathbf{z})_{\mathbf{z} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}},$$
since $(\mathbf{B}|\mathbf{a}^\perp)$ is a basis of $\mathbb{Z}_p^{k+1}$, with probability $1 - \frac{1}{\Omega(p)}$ over the choices of $\mathbf{A}, \mathbf{B}$, and $\mathbf{a}^\perp$ (see Definition 9).

Recall that we use $\equiv$ to denote equality of distribution, and $\approx_\varepsilon$ to indicate that two distributions are statistically ε-close.

Therefore, we have for all $\mathbf{y}^{(\beta)} \in \mathbb{Z}_p^m$:

$\Big(\mathbf{A}, \mathbf{b}^\perp, \{\mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{a}^\perp\}_{j\in[m]}, (\mathbf{b}^\perp)^\top \mathbf{a}^\perp\Big)$
where $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B}), \mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$
$\equiv \Big(\mathbf{A}, \mathbf{b}^\perp, \{\mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{a}^\perp\}_{j\in[m]}, (\mathbf{b}^\perp)^\top (\mathbf{B}\mathbf{s} + \mathbf{a}^\perp)\Big)$
where $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B}), \mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k, \mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ (by 1.)
$\equiv \Big(\mathbf{A}, \mathbf{b}^\perp, \{\mathbf{B}\mathbf{s}_j + y_j^{(\beta)}(\mathbf{B}\mathbf{s} + \mathbf{a}^\perp)\}_{j\in[m]}, (\mathbf{b}^\perp)^\top (\mathbf{B}\mathbf{s} + \mathbf{a}^\perp)\Big)$
where $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B}), \mathbf{s}, \mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ (by 2.)
$\approx_{\frac{1}{\Omega(p)}} \Big(\mathbf{A}, \mathbf{b}^\perp, \{\mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{z}\}_{j\in[m]}, (\mathbf{b}^\perp)^\top \mathbf{z}\Big)$
where $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B}), \mathbf{z} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}, \mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ (by 3.)

□

$\underline{\mathcal{B}_1(\mathcal{PG}, [\mathbf{B}]_2, [\mathbf{h}_1|\cdots|\mathbf{h}_m]_2):}$
$((\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)})) \leftarrow \mathcal{A}(1^\lambda)$
$\mathsf{gpk} := \mathcal{PG}$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\beta \leftarrow_{\mathrm{R}} \{0,1\}$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A})$, $v \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $\mathsf{pk} := [v]_T$. For all $i \in [n]$: $\mathbf{c}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$. For all $j \in [m]$: $\widehat{\mathbf{c}}_j := \mathbf{h}_j + y_j^{(\beta)}\mathbf{a}^\perp$, $\mathsf{ct} := \{[\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2\}_{i\in[n], j\in[m]}$
$\beta' \leftarrow \mathcal{A}^{\mathsf{OKeygen}(\cdot)}(\mathsf{gpk}, \mathsf{pk}, \mathsf{ct})$
Return 1 if $\beta' = \beta$, 0 otherwise.

$\underline{\mathsf{OKeygen}(\alpha \in \mathbb{Z}_p^{n\times m}):}$
$u \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, $\widehat{K} := \sum_{i,j} \alpha_{i,j}[\mathbf{c}_i^\top \widehat{\mathbf{c}}_j]_2 - [u]_2 - \sum_{i,j} \alpha_{i,j} x_i^{(\beta)} y_j^{(\beta)} \cdot [v]_1$, $K := [u]_1$
Return $\mathsf{dk}_\alpha := (K, \widehat{K})$

Figure 7.5: Adversary $\mathcal{B}_1$ against the $\mathcal{D}_k(p)$-MDDH assumption, for the proof of Lemma 44.

Lemma 44: From game G_1 to game G_2

There exists a PPT adversary $\mathcal{B}_1$ such that

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_2, \mathcal{B}_2}(\lambda) + \frac{2}{p-1}.$$

Proof of Lemma 44. Here, we use the $\mathcal{D}_k(p)$-MDDH assumption on $[\mathbf{B}]_2$ to change the distribution of the challenge ciphertext, after arguing that one can simulate the game without knowing $\mathbf{b}^\perp$ or $[\mathbf{B}]_1$.

Namely, we build a PPT adversary $\mathcal{B}_1'$ against the m-fold $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}_2$ such that $|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{m\text{-}\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_2, \mathcal{B}_2'}(\lambda)$. Then, by Lemma 1, this implies the existence of a PPT adversary $\mathcal{B}_1$ such that $|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathcal{D}_k(p)\text{-MDDH}}_{\mathbb{G}_2, \mathcal{B}_2}(\lambda) + \frac{2}{p-1}$.

Adversary $\mathcal{B}_1'$ simulates the game to $\mathcal{A}$ as described in Figure 7.5. We show that when $\mathcal{B}_1'$ is given a real MDDH challenge, that is, $[\mathbf{h}_1|\cdots|\mathbf{h}_m]_2 := [\mathbf{BS}]_2$ for $\mathbf{S} \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k\times m}$, then it simulates the game G_1, whereas it simulates the game G_2 when given a uniformly random challenge, i.e. when $[\mathbf{h}_1|\cdots|\mathbf{h}_m]_2 \leftarrow_{\mathrm{R}} \mathbb{G}_2^{(k+1)\times m}$, which implies the lemma.

We use the fact that for all $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_p^{(k+1)\times k}$,

$$(\mathbf{B}, \mathbf{a}^\perp, (\mathbf{b}^\perp)^\top \mathbf{a}^\perp)_{\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})} \equiv (\mathbf{B}, \mathbf{a}^\perp, v)_{v \leftarrow_{\mathrm{R}} \mathbb{Z}_p)}.$$

Note that the leftmost distribution corresponds to gpk, pk, $\{\mathbf{c}_i\}_{i\in[n]}$, and $\mathsf{OKeygen}$ distributed as in games G_1 or G_2 (these are identically distributed in these two games), while the last distribution corresponds to gpk, pk, $\{\mathbf{c}_i\}_{i\in[n]}$, and $\mathsf{OKeygen}$ simulated by $\mathcal{B}_1'$.

Finally, when $\mathcal{B}_1'$ is given a real MDDH challenge, i.e., when for all $j \in [m]$, $\mathbf{h}_j := \mathbf{B}\mathbf{s}_j$, for $\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, we have $\widehat{\mathbf{c}}_j := \mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{a}^\perp$, exactly as in game G_1, whereas $\widehat{\mathbf{c}}_j$ is uniformly random over $\mathbb{Z}_p^{k+1}$ when $\mathcal{B}_1'$ is given a random challenge, i.e., when for all $j \in [m]$, $\mathbf{h}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^{k+1}$, as in game G_2. □

Lemma 45: Game G_2

$\mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}) = 0$.

Proof of Lemma 45. By definition of the security game, for all α queried to $\mathsf{OKeygen}$, we have: $\sum_{i,j} \alpha_{i,j} x_i^{(\beta)} y_j^{(\beta)} = \sum_{i,j} \alpha_{i,j} x_i^{(0)} y_j^{(0)}$. Therefore, the view of the adversary in G_2 is completely independent from the random bit $\beta \leftarrow_{\mathrm{R}} \{0,1\}$. □

Public-key FE

We give in Figure 7.6 a public-key FE for quadratic functions, that is, the functionality $F_{\mathsf{quad}}^{K,X,Y}$ defined in the previous section. It builds upon the private-key from the previous section, as explained in the overview. We prove one-SEL-IND security, which implies many-SEL-IND security via a standard argument, since we are in the public-key setting. This is proved under the $\mathcal{D}_k(p)$-MDDH assumption in both $\mathbb{G}_1$ and $\mathbb{G}_2$, as well as the 3-PDDH assumption (see Definition 15).

$\underline{\mathsf{GSetup}(1^\lambda, F_{\mathsf{quad}}^{K,X,Y})}$:
$\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e) \leftarrow \mathsf{PGGen}(1^\lambda)$, $\mathsf{gpk} := \mathcal{PG}$
Return gpk

$\underline{\mathsf{Setup}(1^\lambda, \mathsf{gpk}, F_{\mathsf{quad}}^{K,X,Y})}$:
$\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$. For $i \in [2n], j \in [2m]$, $\mathbf{r}_i, \mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$.
Return $\mathsf{pk} := \{[\mathbf{A}\mathbf{r}_i]_1, [\mathbf{B}\mathbf{s}_j]_2\}_{i\in[2n], j\in[2m]}$
and $\mathsf{msk} := \left(\mathbf{A}, \mathbf{B}, \{\mathbf{r}_i, \mathbf{s}_j\}_{i\in[2n], j\in[2m]}\right)$

$\underline{\mathsf{KeyGen}(\mathsf{gpk}, \mathsf{msk}, \alpha \in \mathbb{Z}_p^{n\times m})}$:
$K := [\sum_{i\in[n], j\in[m]} \alpha_{i,j}\left(\mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j + \mathbf{r}_{i+n}^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_{j+m}\right)]_1 - [u]_1 \in \mathbb{G}_1$
$\widehat{K} := [u]_2 \in \mathbb{G}_2$, where $u \leftarrow_{\mathrm{R}} \mathbb{Z}_p$.
Return $\mathsf{dk}_\alpha := (K, \widehat{K}) \in \mathbb{G}_1 \times \mathbb{G}_2$

$\underline{\mathsf{Enc}(\mathsf{gpk}, \mathsf{pk}, (\mathbf{x}, \mathbf{y}) \in \mathbb{Z}_p^n \times \mathbb{Z}_p^m)}$:
$\mathbf{W}, \mathbf{V} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$, $\gamma \leftarrow_{\mathrm{R}} \mathbb{Z}_p$; $c_0 = \widehat{c}_0 := \gamma$; for all $i \in [n], j \in [m]$:
$\mathbf{c}_i := \begin{pmatrix} \gamma \cdot \mathbf{A}\mathbf{r}_i \\ x_i \end{pmatrix}^\top \mathbf{W}^{-1}$, $\mathbf{c}_{n+i} := \begin{pmatrix} \gamma \cdot \mathbf{A}\mathbf{r}_{n+i} \\ 0 \end{pmatrix}^\top \mathbf{V}^{-1}$,
$\widehat{\mathbf{c}}_j := \mathbf{W}\begin{pmatrix} \mathbf{B}\mathbf{s}_j \\ y_j \end{pmatrix}$, $\widehat{\mathbf{c}}_{m+j} := \mathbf{V}\begin{pmatrix} \mathbf{B}\mathbf{s}_{m+j} \\ 0 \end{pmatrix}$
$\mathsf{ct}_{(\mathbf{x},\mathbf{y})} := \{[c_0]_1, [\widehat{c}_0]_2, [\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2\}_{i\in[2n], j\in[2m]} \in \mathbb{G}_1^{2n(k+2)+1} \times \mathbb{G}_2^{2m(k+2)+1}$

$\underline{\mathsf{Dec}(\mathsf{gpk}, \mathsf{pk}, \mathsf{ct}_{(\mathbf{x},\mathbf{y})}, \mathsf{dk}_\alpha)}$:
$[d]_T := \sum_{i\in[n], j\in[m]} \alpha_{i,j}\left(e([\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2) + e([\mathbf{c}_{n+i}]_1, [\widehat{\mathbf{c}}_{m+j}]_2)\right) - e([c_0]_1, \widehat{K}) - e(K, [\widehat{c}_0]_2)$
Return d.

Figure 7.6: $\mathcal{FE}$, a scheme for the functionality $F_{\mathsf{quad}}^{K,X,Y}$, whose one-SEL-IND security relies on the $\mathcal{D}_k(p)$-MDDH assumption and 3-PDDH assumption in asymmetric pairing groups.

Correctness. For any $(\mathbf{x}, \mathbf{y}) \in \mathcal{X}$, $i \in [n], j \in [m]$, we have:

$$e([\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2) = [\gamma \cdot \mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j + x_i y_j]_T.$$

Moreover, for any $i \in \{n+1, \ldots, 2n\}$, $j \in \{m+1, \ldots, 2m\}$, we have:

$$e([\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2) = [\gamma \cdot \mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j]_T.$$

Therefore, for any $\alpha \in \mathcal{K}$, the decryption computes

$$\begin{aligned}[d]_T &:= [\sum_{i,j} \alpha_{i,j}\gamma \cdot \mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j + \sum_{i,j} \alpha_{i,j} x_i y_j]_T - e(K, [\widehat{c}_0]_2) - e([c_0]_1, \widehat{K}) \\ &= [\sum_{i,j} \alpha_{i,j} x_i y_j]_T.\end{aligned}$$

Games G_0, $\boxed{\mathsf{G}_1, \boxed{\mathsf{G}_2}, \mathsf{G}_3}$, G_4, G_5:

$((\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)})) \leftarrow \mathcal{A}(1^\lambda)$
$\mathcal{PG} := (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P_1, P_2, e) \leftarrow_{\mathrm{R}} \mathsf{PGGen}(1^\lambda)$, $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k(p)$, $\beta \leftarrow_{\mathrm{R}} \{0,1\}$,
$\boxed{\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})}$
For $i \in [2n], j \in [2m]$: $\mathbf{r}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$

$$\mathsf{gpk} := \mathcal{PG},\ \mathsf{pk} := \left\{ \left[\mathbf{A}\mathbf{r}_i + \boxed{x_i^{(\beta)}\mathbf{b}^\perp}\right]_1, \left[\mathbf{A}\mathbf{r}_{n+i} - \boxed{x_i^{(0)}\mathbf{b}^\perp}\right]_1, \left[\mathbf{B}\mathbf{s}_j + \boxed{y_j^{(\beta)}\mathbf{a}^\perp}\right]_2 \right.$$
$$\left. \left[\mathbf{B}\mathbf{s}_{m+j} + \boxed{y_j^{(0)}\mathbf{a}^\perp}\right]_2 \right\}_{i\in[n], j\in[m]}$$

$\mathbf{W} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$, $\gamma \leftarrow_{\mathrm{R}} \mathbb{Z}_p$; $\boxed{v \leftarrow_{\mathrm{R}} \mathbb{Z}_p}$; $c_0 = \widehat{c}_0 := \gamma$

$$\mathbf{c}_i := \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_i + \boxed{\gamma x_i^{(\beta)}\mathbf{b}^\perp} + \boxed{v x_i^{(\beta)}\mathbf{b}^\perp} \\ x_i^{(\beta)} - x_i^{(\beta)} \end{pmatrix}^\top \mathbf{W}^{-1}$$

$$\mathbf{c}_{n+i} := \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_{n+i} - \boxed{\gamma x_i^{(0)}\mathbf{b}^\perp} - \boxed{v x_i^{(0)}\mathbf{b}^\perp} \\ 0 + x_i^{(0)} \end{pmatrix}^\top \mathbf{V}^{-1}$$

$$\widehat{\mathbf{c}}_j := \mathbf{W} \begin{pmatrix} \mathbf{B}\mathbf{s}_j + \boxed{y_j^{(\beta)}\mathbf{a}^\perp} \\ y_j^{(\beta)} - y_j^{(\beta)} \end{pmatrix};\ \widehat{\mathbf{c}}_{m+j} := \mathbf{V} \begin{pmatrix} \mathbf{B}\mathbf{s}_{m+j} + \boxed{y_j^{(0)}\mathbf{a}^\perp} \\ 0 + y_j^{(0)} \end{pmatrix}$$

$\mathsf{ct} := \{[c_0]_1, [\widehat{c}_0]_2, [\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2\}_{i\in[2n], j\in[2m]}$
$\beta' \leftarrow \mathcal{A}^{\mathsf{OKeygen}(\cdot)}(\mathsf{gpk}, \mathsf{pk}, \mathsf{ct})$
Return 1 if $\beta' = \beta$, 0 otherwise.

$\mathsf{OKeygen}(\alpha \in \mathbb{Z}_p^{n\times m})$:
$u \leftarrow_{\mathrm{R}} \mathbb{Z}_p$
$K := [\sum_{i\in[n], j\in[m]} \alpha_{i,j}\left(\mathbf{r}_i^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_j + \mathbf{r}_{n+i}^\top \mathbf{A}^\top \mathbf{B}\mathbf{s}_{m+j}\right)]_1 - [u]_1 \in \mathbb{G}_1$
$\widehat{K} := [u]_2 \in \mathbb{G}_2$
Return $\mathsf{dk}_\alpha := (K, \widehat{K}) \in \mathbb{G}_1 \times \mathbb{G}_2$

Figure 7.7: Games for the proof of Theorem 20. In each procedure, the components inside a solid (dotted, gray) frame are only present in the games marked by a solid (dotted, gray) frame.

Since $\sum_{i,j} \alpha_{i,j} x_i y_j \in [0, nmKXY]$ which is of polynomials size, one can efficiently recover the discrete logarithm $d \in \mathbb{Z}$.

Theorem 20: one-SEL-IND security

The scheme from Figure 7.6 is one-SEL-IND secure, assuming the $\mathcal{D}_k(p)$-MDDH assumption in $\mathbb{G}_1$ and $\mathbb{G}_2$, as well as the 3-PDDH assumption.

Proof of Theorem 20. The proof uses hybrid games defined in Figure 7.7. Let $\mathcal{A}$ be a PPT adversary. For any game G, we denote by $\mathsf{Adv}_{\mathsf{G}}(\mathcal{A})$ the probability that the game G returns 1 when interacting with $\mathcal{A}$.

Game G_0: is such that $\mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND}}_{\mathcal{FE},\mathcal{A}}(\lambda) = 2 \times |\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - 1/2|$. For the sake of the proof, we look at the public key elements $\{[\mathbf{A}\mathbf{r}_i]_1, [\mathbf{B}\mathbf{s}_j]_2\}_{i\in[2n], j\in[2m]}$ as a ciphertext of the $\mathsf{FE}_{\mathsf{one}}$ scheme encrypting vectors $(\mathbf{0}, \mathbf{0}) \in \mathbb{Z}_p^{2n} \times \mathbb{Z}_p^{2m}$.

Game G_1: with the above observation in mind, in this game we change the distribution of the public key elements so as to be interpreted as an $\mathsf{FE}_{\mathsf{one}}$ ciphertext encrypting the vectors

$$(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}}) = \left(\begin{pmatrix} \mathbf{x}^{(\beta)} \\ \hline -\mathbf{x}^{(0)} \end{pmatrix}, \begin{pmatrix} \mathbf{y}^{(\beta)} \\ \hline \mathbf{y}^{(0)} \end{pmatrix} \right) \in \mathbb{Z}_p^{2n} \times \mathbb{Z}_p^{2m}$$

In Lemma 46 we show how to argue that game G_1 is computationally indistinguishable from game G_0 based on the selective, single-ciphertext security of $\mathsf{FE}_{\mathsf{one}}$ (that in turn reduces to $\mathcal{D}_k(p)$-MDDH).

Game G_2: in this game we change the distribution of the $\mathbf{c}_i$ components of the challenge ciphertext. We switch from using $\{\gamma \mathbf{A}\mathbf{r}_i + \widetilde{x}_i \cdot \gamma \mathbf{b}^\perp\}_{i \in [2n]}$ to $\{\gamma \mathbf{A}\mathbf{r}_i + \widetilde{x}_i \cdot (\gamma + v)\mathbf{b}^\perp\}_{i \in [2n]}$, for a random $v \leftarrow_{\mathrm{R}} \mathbb{Z}_p$. In Lemma 47 we prove we can do this switch using the 3-PDDH assumption.

Game G_3 : by using a statistical argument we show that in this game the challenge ciphertexts can be rewritten as

$$\mathbf{c}_i := \begin{pmatrix} \gamma \mathbf{A}\mathbf{r}_i + (\gamma + v) x_i^{(\beta)} \mathbf{b}^\perp \\ \hline 0 \end{pmatrix}^\top \mathbf{W}^{-1};$$

$$\mathbf{c}_{n+i} := \begin{pmatrix} \gamma \mathbf{A}\mathbf{r}_{n+i} - (\gamma + v) x_i^{(0)} \mathbf{b}^\perp \\ \hline x_i^{(0)} \end{pmatrix}^\top \mathbf{V}^{-1};$$

$$\widehat{\mathbf{c}}_j := \mathbf{W} \begin{pmatrix} \mathbf{B}\mathbf{s}_j + y_j^{(\beta)} \mathbf{a}^\perp \\ \hline 0 \end{pmatrix}; \widehat{\mathbf{c}}_{m+j} := \mathbf{V} \begin{pmatrix} \mathbf{B}\mathbf{s}_{m+j} + y_j^{(0)} \mathbf{a}^\perp \\ \hline y_j^{(0)} \end{pmatrix}.$$

This step essentially shows that the change in game G_2 made the ciphertexts less dependent on the bit β.

Game G_4: in this game we change again the distribution of the challenge ciphertext components $\mathbf{c}_i$ switching from using $\{\gamma \mathbf{A}\mathbf{r}_i + \widetilde{x}_i \cdot (\gamma + v)\mathbf{b}^\perp\}_{i \in [2n]}$ to $\{\gamma \mathbf{A}\mathbf{r}_i + \widetilde{x}_i \cdot \gamma \mathbf{b}^\perp\}_{i \in [2n]}$. This change is analogous to that introduced in game G_2, and its indistinguishability follows from the 3-PDDH assumption.

The crucial observation is that the public key in this game can be seen as an $\mathsf{FE}_{\mathsf{one}}$ ciphertext encrypting vector $(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}})$, while the challenge ciphertext of game G_4 can be seen as an encryption of vectors

$$\left(\begin{pmatrix} \mathbf{0} \\ \hline \mathbf{x}^{(0)} \end{pmatrix}, \begin{pmatrix} \mathbf{0} \\ \hline \mathbf{y}^{(0)} \end{pmatrix} \right) \in \mathbb{Z}_p^{2n} \times \mathbb{Z}_p^{2m}$$

using such public key. At a high level, the idea is that we moved to a game in which the dependence on the challenge messages $(\mathbf{x}^{(\beta)}, \mathbf{y}^{(\beta)})$ is only in the public key.

Game G_5: in this game we change back the distribution of the public key elements so as to be interpreted as an $\mathsf{FE}_{\mathsf{one}}$ ciphertext encrypting vectors $(\mathbf{0}, \mathbf{0})$. The fact that game G_3 and game G_4 are computationally indistinguishable can be argued based on the selective, single-ciphertext security of the $\mathsf{FE}_{\mathsf{one}}$ scheme.

The proof is concluded by arguing that in this game the view of the adversary is independent of the bit β.

□

We now prove the lemmas needed to prove the above theorem.

Lemma 46: from game G_0 to game G_1

There exists a PPT adversary $\mathcal{B}_0$:

$$|\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathsf{one\text{-}SEL\text{-}IND}}_{\mathsf{FE}_{\mathsf{one}},\mathcal{B}_0}(\lambda).$$

Proof of Lemma 46. Using the one-SEL-IND security of the underlying private-key scheme (which is exactly the scheme in Figure 7.2), we can change the distribution of the public key elements from $\{[\mathbf{A}\mathbf{r}_i]_1, [\mathbf{B}\mathbf{s}_j]_2\}_{i\in[2n],j\in[2m]}$ to

$$\Big\{[\mathbf{A}\mathbf{r}_i + x_i^{(\beta)}\mathbf{b}^\perp]_1, [\mathbf{A}\mathbf{r}_{n+i} - x_i^{(0)}\mathbf{b}^\perp]_1,$$
$$[\mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{a}^\perp]_2, [\mathbf{B}\mathbf{s}_{m+j} + y_j^{(0)}\mathbf{a}^\perp]_2\Big\}_{i\in[n],j\in[m]}$$

In order to apply the one-SEL-IND security of the private-key FE (Theorem19) we rely on the fact that the public key of $\mathcal{FE}$ can be seen as an $\mathsf{FE}_{\mathsf{one}}$ encryption of longer vectors

$$\widetilde{\mathbf{x}}^{(0)} = \mathbf{0} \in \mathbb{Z}_p^{2n} \text{ and } \widetilde{\mathbf{y}}^{(0)} = \mathbf{0} \in \mathbb{Z}_p^{2m} \text{ in } \mathsf{G}_0,$$

$$\widetilde{\mathbf{x}}^{(1)} = (\mathbf{x}^{(\beta)}||-\mathbf{x}^{(0)}) \in \mathbb{Z}_p^{2n} \text{ and } \widetilde{\mathbf{y}}^{(1)} = (\mathbf{y}^{(\beta)}||\mathbf{y}^{(0)}) \in \mathbb{Z}_p^{2m} \text{ in } \mathsf{G}_1.$$

Also, secret keys in $\mathcal{FE}$ can be seen as $\mathsf{FE}_{\mathsf{one}}$ secret keys corresponding to matrices

$$\widetilde{\alpha} = \left(\begin{array}{c|c} \alpha & \mathbf{0} \\ \hline \mathbf{0} & \alpha \end{array}\right) \in \mathbb{Z}_p^{2n\times 2m}.$$

Note that we are using the matrix representation for functions $\alpha \in \mathbb{Z}_p^{nm}$, since more convenient here. In particular, for any vector $\mathbf{x} \in \mathbb{Z}_p^n$, $\mathbf{y} \in \mathbb{Z}_p^m$, we denote by $\mathbf{x}^\top\alpha\mathbf{y} = \sum_{i,j}\alpha_{i,j}x_iy_j$. With this observation in mind, it can be seen that the restriction

$$\mathbf{x}^{(1)\top}\,\alpha\,\mathbf{y}^{(1)} = \mathbf{x}^{(0)\top}\,\alpha\,\mathbf{y}^{(0)}$$

in the queries made by $\mathcal{A}$ translates into legitimate queries by $\mathcal{B}_0$ since $\mathbf{x}^{(\beta)\top}\,\alpha\,\mathbf{y}^{(\beta)} - \mathbf{x}^{(0)\top}\,\alpha\,\mathbf{y}^{(0)} = 0$ and $\widetilde{\mathbf{x}}^{(0)\top}\,\widetilde{\alpha}\,\widetilde{\mathbf{y}}^{(0)} = \widetilde{\mathbf{x}}^{(1)\top}\,\widetilde{\alpha}\,\widetilde{\mathbf{y}}^{(1)} = 0$. Thus, by Theorem 19 (one-SEL-IND security of private-key scheme), we obtain the lemma. □

Lemma 47: From game G_1 to game G_2

There exists a PPT adversary $\mathcal{B}_1$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathsf{3\text{-}PDDH}}_{\mathcal{PG},\mathcal{B}_1}(\lambda) + 2^{-\Omega(\lambda)}.$$

Here, we change the distribution of the challenge ciphertexts, using the 3-PDDH assumption.

Proof of Lemma 47. Upon receiving a 3-PDDH challenge $(\mathcal{PG}, [a]_1, [b]_2, [c]_1, [c]_2, [z]_1)$ (see Definition 15), and the challenge messages $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)})$, $\mathcal{B}_1$ picks $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$; $\beta \leftarrow_{\mathrm{R}} \{0,1\}$; $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})$, and sets $[\gamma]_1 := [c]_1$ and $[\gamma]_2 := [c]_2$. Then, for $i \in [2n], j \in [2m]$, $\mathcal{B}_2$ picks $\mathbf{r}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ and computes

$$\mathsf{pk} := \left\{\left[\mathbf{A}\mathbf{r}_i + a x_i^{(\beta)}\mathbf{b}^\perp\right]_1, \left[\mathbf{A}\mathbf{r}_{n+i} - a x_i^{(0)}\mathbf{b}^\perp\right]_1, \left[\mathbf{B}\mathbf{s}_j + b y_j^{(\beta)}\mathbf{a}^\perp\right]_2, \left[\mathbf{B}\mathbf{s}_{m+j} + b y_j^{(0)}\mathbf{a}^\perp\right]_2\right\}_{i\in[n],j\in[m]}.$$

It picks $\widetilde{\mathbf{W}}, \widetilde{\mathbf{V}} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$ and implicitly sets

$$\mathbf{W} := \widetilde{\mathbf{W}} \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \text{ and } \mathbf{V} := \widetilde{\mathbf{V}} \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1}.$$

Here we use the fact that $(\mathbf{B}|b\mathbf{a}^\perp)$ is full rank with probability $1-\frac{1}{\Omega(p)}$ over $\mathbf{A},\mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A})$, and $b \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ (see Definition 9).

Then, for $i \in [n], j \in [m]$, it computes

$$[\mathbf{c}_i]_1 := \left[\begin{pmatrix} \gamma\mathbf{r}_i \\ z\cdot x_i^{(\beta)} \\ x_i^{(\beta)} \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & 0 \\ \hline 0 & 0 & 1 \end{array}\right) \widetilde{\mathbf{W}}^{-1} \right]_1 \text{ and } [\widehat{\mathbf{c}}_j]_2 := \left[\widetilde{\mathbf{W}} \begin{pmatrix} \mathbf{s}_j \\ y_j^{(\beta)} \\ y_j^{(\beta)} \end{pmatrix} \right]_2$$

$$[\mathbf{c}_{n+i}]_1 := \left[\begin{pmatrix} \gamma\mathbf{r}_{n+i} \\ -z\cdot x_i^{(0)} \\ 0 \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & 0 \\ \hline 0 & 0 & 1 \end{array}\right) \widetilde{\mathbf{V}}^{-1} \right]_1 \text{ and } [\widehat{\mathbf{c}}_{m+j}]_2 := \left[\widetilde{\mathbf{V}} \begin{pmatrix} \mathbf{s}_{m+j} \\ y_j^{(0)} \\ 0 \end{pmatrix} \right]_2.$$

$\mathcal{B}_2$ computes $[c_0]_1 := [\gamma]_1$, $[\widehat{c}_0]_2 := [\gamma]_2$, $\mathsf{gpk} := \mathcal{PG}$, $\mathsf{ct} := \{[c_0]_1, [\widehat{c}_0]_2, [\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2\}_{i\in[2n], j\in[2m]}$. It returns $(\mathsf{gpk}, \mathsf{pk}, \mathsf{ct})$ to $\mathcal{A}$. Then, it simulates $\mathsf{OKeygen}$ as in G_2 (see Figure 7.7). Finally, when $\mathcal{A}$ outputs β', $\mathcal{B}_2$ outputs 1 if $\beta' = \beta$, and 0 otherwise.

It can be seen that when $[z]_1$ is a real 3-PDDH challenge, i.e., $[z]_1 = [abc]_1$, then $\mathcal{B}_2$ simulates game G_1; whereas it simulates game G_2 when $[z]_1 \leftarrow_{\mathrm{R}} \mathbb{G}_1$. In particular, while this is easy to see for the elements of the public key and for ciphertexts $[\widehat{\mathbf{c}}_j]_2$, $[\widehat{\mathbf{c}}_{m+j}]_2$, for the ciphertext elements $[\mathbf{c}_i]_1$, $[\mathbf{c}_{n+i}]_1$ we observe that they can be written as

$$\mathbf{c}_i := \begin{pmatrix} \gamma\mathbf{B}^\top\mathbf{A}\mathbf{r}_i \\ z\cdot x_i^{(\beta)}\cdot(\mathbf{b}^\perp)^\top\mathbf{a}^\perp \\ x_i^{(\beta)} \end{pmatrix}^\top \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \mathbf{W}^{-1} = \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_i + zb^{-1}\cdot x_i^{(\beta)}\mathbf{b}^\perp \\ x_i^{(\beta)} \end{pmatrix}^\top \mathbf{W}^{-1}$$

$$\mathbf{c}_{n+i} := \begin{pmatrix} \gamma\mathbf{B}^\top\mathbf{A}\mathbf{r}_{n+i} \\ -z\cdot x_i^{(0)}\cdot(\mathbf{b}^\perp)^\top\mathbf{a}^\perp \\ 0 \end{pmatrix}^\top \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \mathbf{V}^{-1} = \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_{n+i} + zb^{-1}\cdot x_i^{(0)}\mathbf{b}^\perp \\ 0 \end{pmatrix}^\top \mathbf{V}^{-1}.$$

So, if $z = abc$, then $zb^{-1} = a\gamma$ and the ciphertexts are distributed as in G_1; otherwise if z is random zb^{-1} is identically distributed to $(a\gamma+v)$ as in G_2. This proves $|\mathsf{Adv}_{\mathsf{G}_1}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A})| \le \mathsf{Adv}^{\mathsf{3\text{-}PDDH}}_{\mathcal{PG},\mathcal{B}_2}(\lambda) + 2^{-\Omega(\lambda)}$. □

Lemma 48: From game G_2 to G_3

$|\mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_3}(\mathcal{A})| \le 2^{-\Omega(\lambda)}$.

Here, we change the distribution of the challenge ciphertexts, using a statistical argument.

Proof of Lemma 48. First, we use the fact that for all $\gamma \in \mathbb{Z}_p$:

$$(\gamma, v+\gamma)_{v\leftarrow_{\mathrm{R}}\mathbb{Z}_p} \equiv (\gamma, v)_{v\leftarrow_{\mathrm{R}}\mathbb{Z}_p}.$$

Therefore, we can write the challenge ciphertexts as follows. For all i$\in [n], j \in [m]$:

$$\mathbf{c}_i := \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_i + vx_i^{(\beta)}\mathbf{b}^\perp \\ x_i^{(\beta)} \end{pmatrix}^\top \mathbf{W}^{-1}, \mathbf{c}_{n+i} := \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_{n+i} - vx_i^{(0)}\mathbf{b}^\perp \\ 0 \end{pmatrix}^\top \mathbf{V}^{-1}.$$

Then, we use the facts that:

- $(v \leftarrow_{\mathrm{R}} \mathbb{Z}_p) \approx_{\frac{1}{p}} (v \leftarrow_{\mathrm{R}} \mathbb{Z}_p)$ such that $v+1 \neq 0 \bmod p$.

- $(\mathbf{A}, \mathbf{B}, \mathbf{a}^\perp)_{\mathbf{A},\mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A})} \approx_{\frac{1}{\Omega(p)}} (\mathbf{A}, \mathbf{B}, \mathbf{a}^\perp)_{\mathbf{A},\mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k, \mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}) \setminus \mathsf{Span}(\mathbf{B})}$, by Definition 9.

- For any $v \in \mathbb{Z}_p$ such that $v+1 \neq 0 \bmod p$, $\mathbf{W} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$ is identically distributed than $\widetilde{\mathbf{W}} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \cdot \left(\begin{array}{c|c|c} \mathrm{ID}_{k\times k} & 0 & 0 \\ \hline 0 & \frac{v}{v+1} & \frac{1}{v+1} \\ \hline 0 & -1 & 1 \end{array}\right) \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1}$, where $\widetilde{\mathbf{W}} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$, $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, and $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}) \setminus \mathsf{Span}(\mathbf{B})$.

Therefore, we can change the distribution of $\{\mathbf{c}_i, \widehat{\mathbf{c}}_j\}_{i\in[n],j\in[m]}$ as follows:

$$\begin{aligned}
\widehat{\mathbf{c}}_j &= \widetilde{\mathbf{W}} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \cdot \left(\begin{array}{c|c|c} \mathrm{ID}_{k\times k} & 0 & 0 \\ \hline 0 & \frac{v}{v+1} & \frac{1}{v+1} \\ \hline 0 & -1 & 1 \end{array}\right) \begin{pmatrix} \mathbf{s}_j \\ y_j^{(\beta)} \\ y_j^{(\beta)} \end{pmatrix} \\
&= \widetilde{\mathbf{W}} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \cdot \begin{pmatrix} \mathbf{s}_j \\ y_j^{(\beta)} \\ 0 \end{pmatrix} \\
&= \widetilde{\mathbf{W}} \cdot \begin{pmatrix} \mathbf{B}\mathbf{s}_j + y_j^{(\beta)} \mathbf{a}^\perp \\ 0 \end{pmatrix}
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{c}_i &= \begin{pmatrix} \gamma\mathbf{r}_i \\ v x_i^{(\beta)} \\ x_i^{(\beta)} \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & 0 \\ \hline 0 & 0 & 1 \end{array}\right) \cdot \left(\begin{array}{c|c|c} \mathrm{ID}_{k\times k} & 0 & 0 \\ \hline 0 & \frac{v}{v+1} & \frac{1}{v+1} \\ \hline 0 & -1 & 1 \end{array}\right)^{-1} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \cdot \widetilde{\mathbf{W}}^{-1} \\
&= \begin{pmatrix} \gamma\mathbf{r}_i \\ v \cdot x_i^{(\beta)} \\ x_i^{(\beta)} \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & \frac{-1}{v+1} \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & \frac{v}{v+1} \end{array}\right) \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \cdot \widetilde{\mathbf{W}}^{-1} \\
&= \begin{pmatrix} \gamma\mathbf{r}_i \\ (v+1) \cdot x_i^{(\beta)} \\ 0 \end{pmatrix}^\top \cdot \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & 0 \\ \hline 0 & 0 & 1 \end{array}\right) \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \cdot \widetilde{\mathbf{W}}^{-1} \\
&= \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_i + (v+1) \cdot x_i^{(\beta)} \mathbf{b}^\perp \\ 0 \end{pmatrix}^\top \cdot \widetilde{\mathbf{W}}^{-1}
\end{aligned}$$

Then, we use the facts that:

- $v \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ such that $v+1 \neq 0 \bmod p \approx_{\frac{1}{p}} v \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ such that $v+1 \neq 0 \bmod p$ and $v \neq 0 \bmod p$.

- For any $v \in \mathbb{Z}_p$ such that $v+1 \neq 0 \bmod p$ and $v \neq 0 \bmod p$, $\mathbf{V} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$ is identically distributed than $\widetilde{\mathbf{V}} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \cdot \left(\begin{array}{c|c|c} \mathrm{ID}_{k\times k} & 0 & 0 \\ \hline 0 & 1 & \frac{1}{v} \\ \hline 0 & 1 & 1+\frac{1}{v} \end{array}\right) \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1}$, where $\widetilde{\mathbf{V}} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$, $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, and $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}) \setminus \mathsf{Span}(\mathbf{B})$.

Therefore, we can change the distribution of $\{\mathbf{c}_{n+i}, \widehat{\mathbf{c}}_{m+j}\}_{i\in[n],j\in[m]}$ as follows:

$$\begin{aligned}
\widehat{\mathbf{c}}_{m+j} &= \widetilde{\mathbf{V}} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \cdot \left(\begin{array}{c|c|c} \mathrm{ID}_{k\times k} & 0 & 0 \\ \hline 0 & 1 & \frac{1}{v} \\ \hline 0 & 1 & 1+\frac{1}{v} \end{array}\right) \begin{pmatrix} \mathbf{s}_j \\ y_j^{(0)} \\ 0 \end{pmatrix} \\
&= \widetilde{\mathbf{V}} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right) \cdot \begin{pmatrix} \mathbf{s}_j \\ y_j^{(0)} \\ y_j^{(0)} \end{pmatrix} \\
&= \widetilde{\mathbf{V}} \cdot \begin{pmatrix} \mathbf{B}\mathbf{s}_j + y_j^{(0)}\mathbf{a}^\perp \\ y_j^{(0)} \end{pmatrix}
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{c}_{n+i} &= \begin{pmatrix} \gamma\mathbf{r}_{n+i} \\ -vx_i^{(0)} \\ 0 \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & 0 \\ \hline 0 & 0 & 1 \end{array}\right) \cdot \left(\begin{array}{c|c|c} \mathrm{ID}_{k\times k} & 0 & 0 \\ \hline 0 & 1 & \frac{1}{v} \\ \hline 0 & 1 & 1+\frac{1}{v} \end{array}\right)^{-1} \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \cdot \widetilde{\mathbf{V}}^{-1} \\
&= \begin{pmatrix} \gamma\mathbf{r}_{n+i} \\ -vx_i^{(0)} \\ 0 \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp \cdot (1+\frac{1}{v}) & \frac{-1}{v} \\ \hline 0 & -(\mathbf{b}^\perp)^\top\mathbf{a}^\perp & 1 \end{array}\right) \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \cdot \widetilde{\mathbf{V}}^{-1} \\
&= \begin{pmatrix} \gamma\mathbf{r}_{n+i} \\ -(v+1)x_i^{(0)} \\ x_i^{(0)} \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & 0 & 0 \\ \hline 0 & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & 0 \\ \hline 0 & 0 & 1 \end{array}\right) \cdot \left(\begin{array}{c|c} \mathbf{B}|\mathbf{a}^\perp & 0 \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \cdot \widetilde{\mathbf{V}}^{-1} \\
&= \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_{n+i} - (v+1)x_i^{(0)}\mathbf{b}^\perp \\ x_i^{(0)} \end{pmatrix}^\top \cdot \widetilde{\mathbf{V}}^{-1}
\end{aligned}$$

Finally, we use the fact that for any $\gamma \in \mathbb{Z}_p$: $(v+1)$ where $v \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ such that $v+1 \neq 0 \bmod p$ and $v \neq 0 \bmod p \approx_{\frac{2}{p}} (v+\gamma)$, where $v \leftarrow_{\mathrm{R}} \mathbb{Z}_p$. Thus, we obtain, for all $i \in [n]$ and $j \in [m]$:

$$\mathbf{c}_i := \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_i + (v+\gamma)x_i^{(\beta)}\mathbf{b}^\perp \\ 0 \end{pmatrix}^\top \widetilde{\mathbf{W}}^{-1}, \mathbf{c}_{n+i} := \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_{n+i} - (v+\gamma)x_i^{(0)}\mathbf{b}^\perp \\ x_i^{(0)} \end{pmatrix}^\top \widetilde{\mathbf{V}}^{-1},$$

$\widehat{\mathbf{c}}_j := \widetilde{\mathbf{W}} \begin{pmatrix} \gamma\mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{a}^\perp \\ 0 \end{pmatrix}$, $\widehat{\mathbf{c}}_{m+j} := \widetilde{\mathbf{V}} \begin{pmatrix} \gamma\mathbf{B}\mathbf{s}_j + y_j^{(0)}\mathbf{a}^\perp \\ y_j^{(0)} \end{pmatrix}$, as in game G_3.

This proves $|\mathsf{Adv}_{\mathsf{G}_2}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_3}(\mathcal{A})| \leq 2^{-\Omega(\lambda)}$. □

Lemma 49: From game G_3 to game G_4

There exists an adversary $\mathcal{B}_3$ such that:

$$|\mathsf{Adv}_{\mathsf{G}_3}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A})| \leq 2 \cdot \mathsf{Adv}^{\mathsf{3\text{-}PDDH}}_{\mathcal{PG},\mathcal{B}_2}(\lambda) + 2^{-\Omega(\lambda)}.$$

Here, we change the distribution of the challenge ciphertext, using the 3-PDDH assumption, as for Lemma 47.

Proof of Lemma 49. Upon receiving a 3-PDDH challenge $(\mathcal{PG}, [a]_1, [b]_2, [c]_1, [c]_2, [z]_1)$ (see Definition 15), and the challenge messages $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)})$, $\mathcal{B}_1$ samples $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$;

$b \leftarrow_{\mathrm{R}} \{0,1\}$; $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A}), \mathbf{b}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{B})$, and sets $[\gamma]_1 := [c]_1$ and $[\gamma]_2 := [c]_2$. Then, for $i \in [2n], j \in [2m]$, $\mathcal{B}_2$ picks $\mathbf{r}_i \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$, $\mathbf{s}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_p^k$ and computes

$$\mathsf{pk} := \left\{ \left[\mathbf{A}\mathbf{r}_i + a x_i^{(\beta)} \mathbf{b}^\perp\right]_1, \left[\mathbf{A}\mathbf{r}_{n+i} - a x_i^{(0)} \mathbf{b}^\perp\right]_1, \left[\mathbf{B}\mathbf{s}_j + b y_j^{(\beta)} \mathbf{a}^\perp\right]_2, \left[\mathbf{B}\mathbf{s}_{m+j} + b y_j^{(0)} \mathbf{a}^\perp\right]_2 \right\}_{i\in[n], j\in[m]}.$$

It picks $\widetilde{\mathbf{W}}, \widetilde{\mathbf{V}} \leftarrow_{\mathrm{R}} \mathsf{GL}_{k+2}(p)$ and implicitly sets

$$\mathbf{W} := \widetilde{\mathbf{W}} \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & \begin{matrix}0\\0\end{matrix} \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \text{ and } \mathbf{V} := \widetilde{\mathbf{V}} \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & \begin{matrix}0\\0\end{matrix} \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1}.$$

Here we use the fact that $(\mathbf{B}|b\mathbf{a}^\perp)$ is full rank with probability $1 - \frac{1}{\Omega(p)}$ over $\mathbf{A}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathcal{D}_k$, $\mathbf{a}^\perp \leftarrow_{\mathrm{R}} \mathsf{orth}(\mathbf{A})$, and $b \leftarrow_{\mathrm{R}} \mathbb{Z}_p$ (see Definition 9).

Then, for $i \in [n], j \in [m]$, it computes

$$[\mathbf{c}_i]_1 := \left[\begin{pmatrix} \gamma\mathbf{r}_i \\ z\cdot x_i^{(\beta)} \\ x_i^{(\beta)} \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & \begin{matrix}0\\0\end{matrix} & \begin{matrix}0\\0\end{matrix} \\ \hline \begin{matrix}0 & 0\end{matrix} & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & \begin{matrix}0\\0\end{matrix} \\ \hline \begin{matrix}0 & 0\end{matrix} & \begin{matrix}0\\0\end{matrix} & 1 \end{array}\right) \widetilde{\mathbf{W}}^{-1} \right]_1 \text{ and } [\widehat{\mathbf{c}}_j]_2 := \left[\widetilde{\mathbf{W}} \begin{pmatrix} \mathbf{s}_j \\ y_j^{(\beta)} \\ y_j^{(\beta)} \end{pmatrix} \right]_2$$

$$[\mathbf{c}_{n+i}]_1 := \left[\begin{pmatrix} \gamma\mathbf{r}_{n+i} \\ -z\cdot x_i^{(0)} \\ 0 \end{pmatrix}^\top \left(\begin{array}{c|c|c} \mathbf{A}^\top\mathbf{B} & \begin{matrix}0\\0\end{matrix} & \begin{matrix}0\\0\end{matrix} \\ \hline \begin{matrix}0 & 0\end{matrix} & (\mathbf{b}^\perp)^\top\mathbf{a}^\perp & \begin{matrix}0\\0\end{matrix} \\ \hline \begin{matrix}0 & 0\end{matrix} & \begin{matrix}0\\0\end{matrix} & 1 \end{array}\right) \widetilde{\mathbf{V}}^{-1} \right]_1 \text{ and } [\widehat{\mathbf{c}}_{m+j}]_2 := \left[\widetilde{\mathbf{V}} \begin{pmatrix} \mathbf{s}_{m+j} \\ y_j^{(0)} \\ 0 \end{pmatrix} \right]_2.$$

$\mathcal{B}_2$ computes $[c_0]_1 := [\gamma]_1$, $[\widehat{c}_0]_2 := [\gamma]_2$, $\mathsf{gpk} := \mathcal{PG}$, and $\mathsf{ct} := \{[c_0]_1, [\widehat{c}_0]_2, [\mathbf{c}_i]_1, [\widehat{\mathbf{c}}_j]_2\}_{i\in[2n], j\in[2m]}$. It returns $(\mathsf{gpk}, \mathsf{pk}, \mathsf{ct})$ to $\mathcal{A}$.

Then, it simulates $\mathsf{OKeygen}$ as in G_4 (see Figure 7.7). Finally, if $\mathcal{A}$ outputs β', $\mathcal{B}_2$ outputs 1 if $\beta' = \beta$, and 0 otherwise.

It can be seen that when $[z]_1$ is a real 3-PDDH challenge, i.e., $[z]_1 = [abc]_1$, then $\mathcal{B}_3$ simulates game G_4; whereas it simulates game G_3 when $[z]_1 \leftarrow_{\mathrm{R}} \mathsf{G}_1$. In particular, while this is easy to see for the elements of the public key and for ciphertexts $[\widehat{\mathbf{c}}_j]_2$, $[\widehat{\mathbf{c}}_{m+j}]_2$, for the ciphertext elements $[\mathbf{c}_i]_1$, $[\mathbf{c}_{n+i}]_1$ we observe that they can be written as

$$\mathbf{c}_i := \begin{pmatrix} \gamma\mathbf{B}^\top\mathbf{A}\mathbf{r}_i \\ z\cdot x_i^{(\beta)}\cdot(\mathbf{b}^\perp)^\top\mathbf{a}^\perp \\ x_i^{(\beta)} \end{pmatrix}^\top \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & \begin{matrix}0\\0\end{matrix} \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \mathbf{W}^{-1} = \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_i + zb^{-1}\cdot x_i^{(\beta)}\mathbf{b}^\perp \\ x_i^{(\beta)} \end{pmatrix}^\top \mathbf{W}^{-1}$$

$$\mathbf{c}_{n+i} := \begin{pmatrix} \gamma\mathbf{B}^\top\mathbf{A}\mathbf{r}_{n+i} \\ -z\cdot x_i^{(0)}\cdot(\mathbf{b}^\perp)^\top\mathbf{a}^\perp \\ 0 \end{pmatrix}^\top \left(\begin{array}{c|c} \mathbf{B}|b\cdot\mathbf{a}^\perp & \begin{matrix}0\\0\end{matrix} \\ \hline \mathbf{0} & 1 \end{array}\right)^{-1} \mathbf{V}^{-1} = \begin{pmatrix} \gamma\mathbf{A}\mathbf{r}_{n+i} + zb^{-1}\cdot x_i^{(0)}\mathbf{b}^\perp \\ 0 \end{pmatrix}^\top \mathbf{V}^{-1}.$$

So, if $z = abc$, then $zb^{-1} = a\gamma$ and the ciphertexts are distributed as in G_4; otherwise, if z is random, zb^{-1} is identically distributed to $(a\gamma + v)$ as in G_3. This proves $|\mathsf{Adv}_{\mathsf{G}_3}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A})| \le \mathsf{Adv}^{\mathsf{3\text{-}PDDH}}_{\mathcal{PG},\mathcal{B}_3}(\lambda) + 2^{-\Omega(\lambda)}$. □

Lemma 50: From game G_4 to game G_5

There exists an adversary $\mathcal{B}_4$ such that

$$|\mathsf{Adv}_{\mathsf{G}_4}(\mathcal{A}) - \mathsf{Adv}_{\mathsf{G}_5}(\mathcal{A})| \le 2\cdot\mathsf{Adv}^{\mathsf{one\text{-}SEI\text{-}IND}}_{\mathsf{FE}_{\mathsf{one}},\mathcal{B}_4}(\lambda).$$

Proof of Lemma 50. This transition is symmetric to that between G_0 and G_1: we use the selective, single-ciphertext security of the underlying private-key scheme (in Figure 7.2), to switch: $\{[\mathbf{A}\mathbf{r}_i + x_i^{(\beta)}\mathbf{b}^\perp]_1, [\mathbf{A}\mathbf{r}_{n+i} - x_i^{(0)}\mathbf{b}^\perp]_1, [\mathbf{B}\mathbf{s}_j + y_j^{(\beta)}\mathbf{a}^\perp]_2, [\mathbf{B}\mathbf{s}_{m+j} + y_j^{(0)}\mathbf{a}^\perp]_2\}_{i\in[n],j\in[m]}$ to $\{[\mathbf{A}\mathbf{r}_i]_1, [\mathbf{B}\mathbf{s}_j]_2\}_{i\in[2n],j\in[2m]}$, since $\mathbf{x}_i^{(\beta)\top}\alpha\mathbf{y}_j^{(\beta)} - \mathbf{x}_i^{(0)\top}\alpha y_j^{(0)} = 0$, by definition of the security game. Thus, by Theorem 19 (one-SEL-IND security of $\mathsf{FE}_{\mathsf{one}}$), we obtain the lemma. □

Lemma 51: Game G_5:

$\mathsf{Adv}_{\mathsf{G}_5}(\mathcal{A}) = 0$.

Proof. This follows directly from inspection of game G_5 in Figure 7.7, which does not depend on the bit $\beta \leftarrow_{\mathrm{R}} \{0,1\}$. □

Chapter 8

Conclusion

Summary of the Contributions

In this thesis, we presented a new public-key encryption that satisfies a strong security notion, which prevents many users to collude and perform complex, large-scale attacks. Our construction, which appeared in [GHKW16], was the first CCA-secure encryption scheme with a tight security reduction from the DDH assumption, without using pairings. It also has short ciphertexts (they only contain three group elements). Figure 1.1 gives the state of the art for tightly CCA-secure encryption.

Our proof techniques depart from the long line of prior works [HJ12, LJYP14, LPJY15] that uses non-interactive zero-knowledge proofs with tight simulation soundness, for which we have no efficient construction from standard assumptions without pairings. Other works [HKS15, AHY15a, GCD+16] first build a tightly-secure IBE to then obtain CCA-secure encryption. However, IBE is notoriously hard to build without pairings in the standard model [BPR+08], let alone with a tight security proof and short ciphertexts. To get rid of the pairings, we revisit techniques from [CW13] together with the hash-proof system approach used in [CS98].

We address the major limitation of our construction in [GHK17], where the size of the public key is reduced to a constant number of group elements, using techniques from [Hof17]. We chose to only present the predecessor [GHKW16] here.

We also presented new functional encryption schemes from standard assumptions. We followed a bottom-up approach, where we explored new constructions for larger classes of functions, with new features, starting from simple constructions, that rely on well-understood assumptions. Namely, we extended the original functional encryption schemes for inner products from [ABDP15, ALS16] to a multi-input setting:

- in Chapter 4, we present a multi-input functional encryption scheme (MIFE) for inner products based on the MDDH assumption in prime-order bilinear groups. Our construction works for any polynomial number of encryption slots and achieves adaptive security against unbounded collusion, while relying on standard polynomial hardness assumptions. Prior to this work, which was published in [AGRW17], we did not even have a candidate for 3-slot MIFE for inner products in the generic bilinear group model. Our work is also the first MIFE scheme for a non-trivial functionality based on standard cryptographic assumptions, as well as the first to achieve polynomial security loss for a super-constant number of slots under falsifiable assumptions. Prior works required stronger non-standard assumptions such as indistinguishability obfuscation or multilinear maps. The construction presented in Chapter 4 improves upon [AGRW17] in that security handles corruption of input slots, with no additional efficiency cost or extra assumption.

- in Chapter 5, we present constructions of multi-input functional encryption (MIFE)

schemes for the inner-product functionality that improve those from Chapter 4 in two main directions.

First, we put forward a novel methodology to convert single-input functional encryption for inner products into multi-input schemes for the same functionality. Our transformation is surprisingly simple, general and efficient. In particular, it does not require pairings and it can be instantiated with *all* known single-input schemes. This leads to two main advances. First, we enlarge the set of assumptions this primitive can be based on, notably, obtaining new MIFEs for inner products from plain DDH, LWE, and Decisional Composite Residuosity. Second, we obtain the first MIFE schemes from standard assumptions where decryption works efficiently even for messages of super-polynomial size. This work appeared in [ACF+18]. As for the pairing-based MIFE presented in Chapter 4, the novelty of the work presented in Chapter 5 of this thesis is that its security handles corruptions of input slots.

Then, we turned our attention to multi-client functional encryption for inner products, which enhances multi-input functional encryption in the following way. In MCFE, the encryption algorithm takes as an additional input a label (typically a time-stamp), and ciphertexts from different input slots can only be combined when they are encrypted under the same label. This limits the leakage of information from the encrypted messages. Multi-input functional encryption corresponds to the case where every message is encrypted under the same label.

In Chapter 6, we give the first MCFE for inner products from standard assumptions, namely, bilinear groups. We first give a simple construction whose security is based on the Decisional Diffie Hellman assumption in the random oracle model, which only satisfies a somewhat weak security model. This construction appeared in [CDG+18a].

Then, we give several transformations to strengthen security, using a new primitive that we called Secret Sharing Encapsulation; and an extra layer of single-input functional encryption on top of the original scheme. The resulting scheme is fully secure, and relies on bilinear groups, in the random oracle model. We also show a generic way to decentralize the generation of the functional decryption keys, and the setup of the scheme. These can be performed independently by all users, without interaction. We obtain a multi-client functional encryption where there is no need for trusted authority holding any master secret key. These transformations appeared in [CDG+18b].

Finally, in Chapter 7, we give the first functional encryption that supports the evaluation of degree-2 polynomials on encrypted data, from standard assumptions. This work appeared in [BCFG17]. The ciphertexts are succinct: their size only depends linearly on the encrypted message, and not the functions for which functional decryption keys are generated. This is as far as it goes in terms of functional encryption beyond predicate encryption, for constant degree polynomials, from standard assumptions. Recall that in [LT17], it is shown that succinct functional encryption which supports the evaluation of degree-3 polynomials on encrypted data already implies indistinguishability obfuscation (together with the existence of block-wise 3-local PRG), a powerful tool that has surprisingly many applications in cryptography, including solving long standing open problems (see [SW14]). Unfortunately, there is no known construction of such functional encryption (with unbounded collusion) from standard assumptions.

Open Problems

Tight security. Can we exhibit tight security reduction for more advanced encryption schemes, such as attribute-based encryption, or functional encryption? Even though tightly secure identity-based encryption are known [CW13, HKS15, AHY15a, GCD+16], all of these schemes have a large public key (it contains $\Omega(\lambda)$ group elements, where λ denotes the security parameter), or rely on composite-order pairings [CGW17], which are less efficient than their prime-order counterpart. Current techniques, such as adaptive partitioning from [Hof17], have

thus far been unsuccessful at providing a tightly-secure IBE with compact ciphertexts and public key, in the prime-order setting.

More generally, we believe bridging the gap between currently known attacks against cryptographic schemes and their security proof is a fruitful research agenda. Finding tighter security reductions is one way to bridge that gap, by ruling out more attacks than traditional, asymptotic security reductions. Another approach consists of finding explicit attacks against particular cryptosystems that match as much as possible the security proof. As far as we know, there are no known attacks against concrete public-key encryption schemes which make use of the fact that the security reduction is not tight. This deserves to be investigated.

Functional encryption. Interesting open problems include building functional encryption that supports the evaluation of degree-2 polynomials on encrypted data with *large messages*. Current constructions [BCFG17, DGP18] crucially rely on the use of pairings, which only allows decryption to recover the value in the exponent of a group element. Since correctness involves solving a discrete logarithm in this group, we require the size of the message to be bounded by a polynomial in the security parameter (note that discrete logarithm should be hard to compute for large values, for the security of the scheme). Because they would probably require radically new techniques, and most likely avoid the use of pairings, such functional encryption with large messages would be much insightful.

Besides, exploring larger classes of functions from standard assumptions, in particular getting degree-3 succinct functional encryption from standard assumptions (and thereby, indistinguishability obfuscation, given the result of [LT17]) would be a breakthrough.

On the more practical side, mitigating the reliance on trusted third party (which holds a master secret key) would increase the practical relevance of functional encryption. Decentralized multi-client functional encryption goes into that direction. We hope this work will inspire further research following the same approach for other classes of functions, or predicate encryption.

Acknowledgments

Foremost, I wish to thank my advisor Hoeteck Wee. He influenced my work greatly, and his care and dedication went far beyond what I could have expected. You like to quote: "There are no two words [...] more harmful than good job"; I want to say there are no two words more appropriate than thank you. I thank Michel Abdalla, for his perspicacious advice, and his precious support, especially when I needed it. David Pointcheval, for running the lab in a seemingly effortless way. The legend has it that David has a twin brother that helps him do all the work. But that is myth, since that much work would require at least triplets.

Je remercie chaleureusement Benoît Libert d'avoir pris le soin de relire mon manuscript de thèse en détail, et d'avoir fourni de multiples conseils qui ont nettement contribué à améliorer la qualité de cette thèse. I thank Katsuyuki Takashima for accepting to review my PhD thesis. It's an honor to have you come all the way from Japan to attend my defense.

I thank Sophie Laplante and Iordanis Kerenidis for a valuable guidance and an insightful first exposure to doing research in cryptography.

I thank Eike Kiltz for hosting me in Bochum before my PhD, and for introducing me to an algebraic viewpoint in cryptography that hasn't left me since then. I thank Carla Ràfols, for teaching me all the intricacies of the Groth-Sahai proofs, and more. I only regret your French is so good I'm not even incited to practice my Spanish with you. I thank Sakib Kakvi for his random anecdotes, and my office mate Bertram Poettering, for impromptu nespresso tasting. Jiaxin Pan for making my stay sportive, Daniel Masny for making me discover light and delicate German meals, such as Schweinshaxe. I thank Olivier Blazy, Manuel Fersch, Federico Giacon, Stefan Guido Hoffmann, Gottfried Herold, Felix Heuer, Elena Kirshanova, Alexander May, Ilya Ozerov, Frank Quedenfeld, Marion Reinhardt-Kalender, Sven Schäge.

I thank Lucas Trevisan for hosting me at UC Berkeley, and showing me around the campus. I thank Tal Rabin for organizing the amazing cryptography workshop at the Simon's Institute, and my office mates Marshall Ball, Sasha Berkoff, Tobias Boelter, Paul Kirchner, Mukul Kulkarni, Tianren Liu, Manuel Sabin. I thank Tancrède Lepoint for touristic visits and somehow convincing me to wake up at 7AM to go to the gym (those who know me can appreciate how unlikely this was).

Je tiens à remercier mes collègues de l'ENS: Balthazar Bauer, compétiteur talentueux au championnat d'étourderie, Sonia Belaïd, Fabrice Benhamouda, Raphaël Bost, Florian Bourse et Geoffroy Couteau aka les bolosses, pour nos aventures hollandaises et allemandes (is this acknowledgement too big for you?), Céline Chevalier, Jérémy Chotard, Simon Cogliani, Mario Cornejo, Angelo De Caro, Léo Colisson, Rafaël Del Pino qui a peut être finalement trouvé la route du Groenland?, Itai Dinur, Léo Ducas, Edouard Dufour Sans, l'expert des big data dans tout le sud ouest: g^{merci}, Aurélien Dupin, pour m'avoir fait connaître tous les décathlons d'île de France, et la fameuse pizzeria d'Igny, Pierre-Alain Dupont, Ehsan Ebrahimi, Pooya Farshim, Houda Ferradi, Georg Fuchsbauer que j'ai apprécié avoir comme co-auteur, Rémi Géraud, Junqing Gong, Dahmun Goudarzi pour ses poses sexy sur les parterres de fleur en terres australes, Giuseppe Guagliardo, Chloé Hébant, compétitrice talentueuse au championnat de (l'absence de) tact, Duong Hieu Phan, Quoc Huy Vu, Louiza Khati, Baptiste Louf, Vadim Lyubashevsky for teaching an inspiring crypto class, Pierrick Méaux, confident des pauses café, et prof de tact à ses heures perdues, pour son impact aquatique sur le labo, Thierry Mefenza,

Brice Minaud, Michele Minelli à qui je dois envoyer la recette des pâtes aux micro-ondes, Nicky Mouha, David Naccache, Anca Nitulescu, pour égayer le labo de son style coloré, Michele Orrù, Alain Passelègue, pour d'intéressantes conversations, notamment quand il s'agissait de basher les États-Unis, Thomas Peters, Duong-Hieu Phan, Antoine Plouviez, Thomas Prest, Razvan Rosie, Mélissa Rossi, compagne de voyage en Australie, Sylvain Ruhault, Théo Ryffel, Olivier Sanders, Antonia Schmidt-Lademann, Azam Soleimanian, Adrian Thillard, Bogdan Ursu, parti en Allemagne, mais toujours un peu là, je suis ravi de notre collaboration, Damien Vergnaud. Je remercie aussi Camilla et Ilaria, visiteuses régulières pour les pasta party. I also thank visitors Luke Kowalczyk, who taught me the importance of the quality of the ice used in cocktails, the emphatic Tal Malkin, for great conversations, Claudio Orlandi, friendly co-author, for a nice collaboration. Je remercie également le personnel administratif de l'ENS et les membres du SPI: Jacques Beigbeder, Lise-Marie Bivard, Isabelle Delais, Nathalie Gaudechoux, Joëlle Isnard, Valérie Mongiat, Ludovic Ricardou, and Sophie Jaudon.

Je remercie chaleureusement Pierre-Alain Fouque d'avoir accepté de faire partie de mon jury de thèse, et Adeline Langlois de m'avoir invité pour un séminaire à Rennes.

I thank Dennis Hofheinz for inviting me to visit at KIT, whose humility and kindness only equal his brightness and love for research. Special thanks to Julia Hesse and Lisa Kohl for giving me a warm welcome in Karlsruhe. I have been lucky to meet Thomas Agrikola, Brandon Broadnax, Rafael Dowsley, Dingding Jia, Alexander Koch, Jessica Koch, Carmen Manietta, Jörn Müller-Quade, Matthias Nagel, Jiaxin Pan, Andy Rupp, Mario Strefler, Bogdan Ursu, Akin Ünal, Cong Zhang. Je remercie Gilles Barthe de m'avoir invité au IMDEA Software Institute et pour notre collaboration enrichissante. I also thank Dario Fiore, for being a great co-author, and for accepting to be part of the jury for my PhD defense. I'm glad I met: Matteo Campanelli, Antonio Faonio, Artem Khyzha, Bogdan Kulynych, Vincent Laporte, Yuri Meshman, Luca Nizzardo, Nataliia Stulova. Especialmente muchas gracias a Miguel Ambrona, con quien me ha gustado trabajar y visitar a Madrid. Mola mucho!

I thank Rachel Lin for inviting me to visit UCSB, who impressed me by her research and by how friendly she was. I am also grateful to have her as a member of the jury for my PhD defense. Together with Stefano Tessaro, they made my stay very pleasant and insightful. I am fortunate for my interactions with Priyanka Bose, Binyi Chen, Sandro Coretti, Yevgeniy Dodis, Pooya Farshim, Joseph Jaeger, Harish Karthikeyan, Christian Matt, Pratik Soni, Ben Terner. Je remercie particulièrement Fabrice Benhamouda que j'ai eu du (Buddah's) bowl d'avoir comme coloc. I'm especially thankful to Aishwarya Thiruvengadam, for good conversations, and great kayaking skills (although the combination of the two can be perilous).

I thank Sanjam Garg for hosting me at UC Berkeley, making this an enjoyable and fruitful stay for me. I'm very excited and grateful to start a postdoc with you. I thank Daniel Apon, Prashant Vasudevan, Mohammad Hajiabadi, for being an amazing co-author, Daniel Masny, for welcoming me at Berkeley, Xiao Liang, for teaching me French (summer boy for always), and Sruthi Sekar, for linear algebra talks. Our friendship has reached the top since then.

Je remercie Damien Stehlé de m'avoir invité à Lyon, où j'ai eu la chance de rencontrer Junqing Gong, Gottfried Herold, Elena Kirshanova, Fabien Laguillaumie, Benoît Libert, Fabrice Mouhartem, Alice Pellet–Mary, Miruna Rosca, Weiqiang Wen.

Lors de mon premier stage de recherche, j'ai eu la chance d'être encadré par Brahim Chaibdraa de l'université de Laval, au Québec. Dans cette contrée au dialecte comique, j'ai eu le plaisir de rencontrer Sophie Létourneau, Josiane Ménard, Alejandro Sanchez. Muchas gracias tambien a Oriol Serra, que me ha recibido por una visita a la UPC, m'ha agradat molt treballar amb tu! Ahi he encontrado tambien Florent Foucaud y Guillem Perarnau.

Remerciement tribal à Julie Gauthier et Léo Girardin, des colocs high en couleurs, David, Ogg, Bathilde, Zoé et Boris, pour nos soirées Futgeuze, Varrax, parti trop tôt en terre bretonne, et Sarah.

Bien sur je remercie vivement les nadines: Président, Présidente, pour m'avoir hébergé à l'hotel Halfon pendant ma période de nomadisme, ainsi que Juan Isaak Carlos miniprez, pour

nous avoir fait apprécier le suspens des prénoms; Malefoy Alimasse, fondateur du mouvement Nadine unifié, Joris générateur de citations Kamelot, ou plus généralement d'entropie, Guillaume Davy générateur de conversations sur la sécurité de Whatsapp, Samickey générateur de conversation sur le communadinisme, et en général je remercie les Jazirez, pour être plus aptes à garder mes clés que je ne le suis moi même, Seya générateur de gifs, Skippy, Doc, Japan boy, Mlle Razakarison. Je remercie mes collègues du MeuPRI, en particulier Laurent Feuilloley et Nathan Grosshans.

Je remercie Fabrice Lembrez, mon prof de spé math, qui par sa pédagogie, a renforcé mon goût pour les mathématiques. Je tiens à remercier mes amis PC1, tous divins: Dounia Arcens, Pierre-Louis Alzieu, Anne Bernhart, Sylvain Borie, Farinelli Boyeldieu, Antoine Buges, Maxime Collodel (Mr adiabatique), Blandine Darfeuil, Florent Delval, Juliette Deu, Alexandre Plazanet, Milena Suarez, Marine Uribesalgo, Agnès Verdier.

Je remercie Jean-Baptiste, pour une amitié qui dure depuis le lycée jusqu'à ces jours, où je squatte son/mon/notre canapé, et pour m'avoir fait découvrir des musiques à la valeur artistique parfois (très) insoupconnée.

Je remercie Marta d'avoir passé de belles et nombreuses années à mes côtés. Je remercie mes parents et ma soeur pour leur soutien indéfectible, leur support inconditionnel, aussi, les toasts au magret. Merci!

Personal Publications

[ABGW17] M. Ambrona, G. Barthe, R. Gay, and H. Wee. Attribute-based encryption in the generic group model: Automated proofs and new constructions. In *ACM CCS 17*, pages 647–664. ACM Press, October / November 2017.

[ACF+18] M. Abdalla, D. Catalano, D. Fiore, R. Gay, and B. Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In *CRYPTO 2018, Part I*, *LNCS* 10991, pages 597–627. Springer, Heidelberg, August 2018.

[AGRW17] M. Abdalla, R. Gay, M. Raykova, and H. Wee. Multi-input inner-product functional encryption from pairings. In *EUROCRYPT 2017, Part I*, *LNCS* 10210, pages 601–626. Springer, Heidelberg, April / May 2017.

[BCFG17] C. E. Z. Baltico, D. Catalano, D. Fiore, and R. Gay. Practical functional encryption for quadratic functions with applications to predicate encryption. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 67–98. Springer, Heidelberg, August 2017.

[CDG+18] J. Chotard, E. Dufour Sans, R. Gay, D. H. Phan, and D. Pointcheval. Decentralized multi-client functional encryption for inner product. In *ASIACRYPT 2018, Part II*, *LNCS* 11273, pages 703–732. Springer, Heidelberg, December 2018.

[CGW15] J. Chen, R. Gay, and H. Wee. Improved dual system ABE in prime-order groups via predicate encodings. In *EUROCRYPT 2015, Part II*, *LNCS* 9057, pages 595–624. Springer, Heidelberg, April 2015.

[FG18] G. Fuchsbauer and R. Gay. Weakly secure equivalence-class signatures from standard assumptions. In *PKC 2018, Part II*, *LNCS* 10770, pages 153–183. Springer, Heidelberg, March 2018.

[FGKO17] G. Fuchsbauer, R. Gay, L. Kowalczyk, and C. Orlandi. Access control encryption for equality, comparison, and more. In *PKC 2017, Part II*, *LNCS* 10175, pages 88–118. Springer, Heidelberg, March 2017.

[GHK17] R. Gay, D. Hofheinz, and L. Kohl. Kurosawa-desmedt meets tight security. In *CRYPTO 2017, Part III*, *LNCS* 10403, pages 133–160. Springer, Heidelberg, August 2017.

[GHKP18] R. Gay, D. Hofheinz, L. Kohl, and J. Pan. More efficient (almost) tightly secure structure-preserving signatures. In *EUROCRYPT 2018, Part II*, *LNCS* 10821, pages 230–258. Springer, Heidelberg, April / May 2018.

[GHKW16] R. Gay, D. Hofheinz, E. Kiltz, and H. Wee. Tightly CCA-secure encryption without pairings. In *EUROCRYPT 2016, Part I*, *LNCS* 9665, pages 1–27. Springer, Heidelberg, May 2016.

[GKW15] R. Gay, I. Kerenidis, and H. Wee. Communication complexity of conditional disclosure of secrets and attribute-based encryption. In *CRYPTO 2015, Part II*, *LNCS* 9216, pages 485–502. Springer, Heidelberg, August 2015.

[GKW18] R. Gay, L. Kowalczyk, and H. Wee. Tight adaptively secure broadcast encryption with short ciphertexts and keys. In *SCN 18*, *LNCS* 11035, pages 123–139. Springer, Heidelberg, September 2018.

[GMW15] R. Gay, P. Méaux, and H. Wee. Predicate encryption for multi-dimensional range queries from lattices. In *PKC 2015*, *LNCS* 9020, pages 752–776. Springer, Heidelberg, March / April 2015.

Bibliography

[ABDP15] M. Abdalla, F. Bourse, A. De Caro, and D. Pointcheval. Simple functional encryption schemes for inner products. In *PKC 2015*, *LNCS* 9020, pages 733–751. Springer, Heidelberg, March / April 2015.

[ABDP16] M. Abdalla, F. Bourse, A. De Caro, and D. Pointcheval. Better security for functional encryption for inner product evaluations. Cryptology ePrint Archive, Report 2016/011, 2016. http://eprint.iacr.org/2016/011.

[ABGW17] M. Ambrona, G. Barthe, R. Gay, and H. Wee. Attribute-based encryption in the generic group model: Automated proofs and new constructions. In *ACM CCS 17*, pages 647–664. ACM Press, October / November 2017.

[ABP15] M. Abdalla, F. Benhamouda, and D. Pointcheval. Public-key encryption indistinguishable under plaintext-checkable attacks. In *PKC 2015*, *LNCS* 9020, pages 332–352. Springer, Heidelberg, March / April 2015.

[ACD+12] M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo. Constant-size structure-preserving signatures: Generic constructions and simple assumptions. In *ASIACRYPT 2012*, *LNCS* 7658, pages 4–24. Springer, Heidelberg, December 2012.

[ACF+18] M. Abdalla, D. Catalano, D. Fiore, R. Gay, and B. Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In *CRYPTO 2018, Part I*, *LNCS* 10991, pages 597–627. Springer, Heidelberg, August 2018.

[ADK+13] M. Abe, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo. Tagged one-time signatures: Tight security and optimal tag size. In *PKC 2013*, *LNCS* 7778, pages 312–331. Springer, Heidelberg, February / March 2013.

[AGK08] M. Abe, R. Gennaro, and K. Kurosawa. Tag-KEM/DEM: A new framework for hybrid encryption. *Journal of Cryptology*, 21(1):97–130, January 2008.

[Agr17] S. Agrawal. Stronger security for reusable garbled circuits, general definitions and attacks. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 3–35. Springer, Heidelberg, August 2017.

[AGRW17] M. Abdalla, R. Gay, M. Raykova, and H. Wee. Multi-input inner-product functional encryption from pairings. In *EUROCRYPT 2017, Part I*, *LNCS* 10210, pages 601–626. Springer, Heidelberg, April / May 2017.

[AGVW13] S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption: New perspectives and lower bounds. In *CRYPTO 2013, Part II*, *LNCS* 8043, pages 500–518. Springer, Heidelberg, August 2013.

[AHN+17] M. Abe, D. Hofheinz, R. Nishimaki, M. Ohkubo, and J. Pan. Compact structure-preserving signatures with almost tight security. In *CRYPTO 2017, Part II*, *LNCS* 10402, pages 548–580. Springer, Heidelberg, August 2017.

[AHY15a] N. Attrapadung, G. Hanaoka, and S. Yamada. A framework for identity-based encryption with almost tight security. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 521–549. Springer, 2015.

[AHY15b] N. Attrapadung, G. Hanaoka, and S. Yamada. A framework for identity-based encryption with almost tight security. In *ASIACRYPT 2015, Part I*, *LNCS* 9452, pages 521–549. Springer, Heidelberg, November / December 2015.

[AJ15] P. Ananth and A. Jain. Indistinguishability obfuscation from compact functional encryption. In *CRYPTO 2015, Part I*, *LNCS* 9215, pages 308–326. Springer, Heidelberg, August 2015.

[ALS16] S. Agrawal, B. Libert, and D. Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In *CRYPTO 2016, Part III*, *LNCS* 9816, pages 333–362. Springer, Heidelberg, August 2016.

[AS17] P. Ananth and A. Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In *EUROCRYPT 2017, Part I*, *LNCS* 10210, pages 152–181. Springer, Heidelberg, April / May 2017.

[BB04] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *CRYPTO 2004*, *LNCS* 3152, pages 443–459. Springer, Heidelberg, August 2004.

[BBG05] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In *EUROCRYPT 2005*, *LNCS* 3494, pages 440–456. Springer, Heidelberg, May 2005.

[BBM00] M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In *EUROCRYPT 2000*, *LNCS* 1807, pages 259–274. Springer, Heidelberg, May 2000.

[BCFG17] C. E. Z. Baltico, D. Catalano, D. Fiore, and R. Gay. Practical functional encryption for quadratic functions with applications to predicate encryption. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 67–98. Springer, Heidelberg, August 2017.

[BCHK07] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 36(5):1301–1328, 2007.

[BDJR97a] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403. IEEE Computer Society Press, October 1997.

[BDJR97b] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 394–403. IEEE, 1997.

[BF01] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In *CRYPTO 2001*, *LNCS* 2139, pages 213–229. Springer, Heidelberg, August 2001.

[BF03] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. *SIAM journal on computing*, 32(3):586–615, 2003.

[BGG+14] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT 2014*, *LNCS* 8441, pages 533–556. Springer, Heidelberg, May 2014.

[BGI+01] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *CRYPTO 2001*, *LNCS* 2139, pages 1–18. Springer, Heidelberg, August 2001.

[BGI+12] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im) possibility of obfuscating programs. *Journal of the ACM (JACM)*, 59(2):6, 2012.

[BGJS15] S. Badrinarayanan, D. Gupta, A. Jain, and A. Sahai. Multi-input functional encryption for unbounded arity functions. In *ASIACRYPT 2015, Part I*, *LNCS* 9452, pages 27–51. Springer, Heidelberg, November / December 2015.

[BGW05] D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO 2005*, *LNCS* 3621, pages 258–275. Springer, Heidelberg, August 2005.

[BJK15] A. Bishop, A. Jain, and L. Kowalczyk. Function-hiding inner product encryption. In *ASIACRYPT 2015, Part I*, *LNCS* 9452, pages 470–491. Springer, Heidelberg, November / December 2015.

[BJL16] F. Benhamouda, M. Joye, and B. Libert. A new framework for privacy-preserving aggregation of time-series data. *ACM Trans. Inf. Syst. Secur.*, 18(3):10:1–10:21, 2016.

[BKP14] O. Blazy, E. Kiltz, and J. Pan. (Hierarchical) identity-based encryption from affine message authentication. In *CRYPTO 2014, Part I*, *LNCS* 8616, pages 408–425. Springer, Heidelberg, August 2014.

[BKS16] Z. Brakerski, I. Komargodski, and G. Segev. Multi-input functional encryption in the private-key setting: Stronger security from weaker assumptions. In *EUROCRYPT 2016, Part II*, *LNCS* 9666, pages 852–880. Springer, Heidelberg, May 2016.

[Ble98] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the rsa encryption standard pkcs# 1. In *Annual International Cryptology Conference*, pages 1–12. Springer, 1998.

[BLR+15] D. Boneh, K. Lewi, M. Raykova, A. Sahai, M. Zhandry, and J. Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In *EUROCRYPT 2015, Part II*, *LNCS* 9057, pages 563–594. Springer, Heidelberg, April 2015.

[BNPW16] N. Bitansky, R. Nishimaki, A. Passelègue, and D. Wichs. From cryptomania to obfustopia through secret-key functional encryption. In *TCC 2016-B, Part II*, *LNCS* 9986, pages 391–418. Springer, Heidelberg, October / November 2016.

[Boy99] V. Boyko. On the security properties of OAEP as an all-or-nothing transform. In *CRYPTO'99*, *LNCS* 1666, pages 503–518. Springer, Heidelberg, August 1999.

[Boy08] X. Boyen. The uber-assumption family (invited talk). In *PAIRING 2008*, *LNCS* 5209, pages 39–56. Springer, Heidelberg, September 2008.

[BPR+08] D. Boneh, P. A. Papakonstantinou, C. Rackoff, Y. Vahlis, and B. Waters. On the impossibility of basing identity based encryption on trapdoor permutations. In *49th FOCS*, pages 283–292. IEEE Computer Society Press, October 2008.

[BR96] M. Bellare and P. Rogaway. The exact security of digital signatures-how to sign with rsa and rabin. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 399–416. Springer, 1996.

[BSW06] D. Boneh, A. Sahai, and B. Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *EUROCRYPT 2006*, *LNCS* 4004, pages 573–592. Springer, Heidelberg, May / June 2006.

[BSW07] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society Press, May 2007.

[BSW11] D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *TCC 2011*, *LNCS* 6597, pages 253–273. Springer, Heidelberg, March 2011.

[BV15] N. Bitansky and V. Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In *56th FOCS*, pages 171–190. IEEE Computer Society Press, October 2015.

[BW06] D. Boneh and B. Waters. A fully collusion resistant broadcast, trace, and revoke system. In *ACM CCS 06*, pages 211–220. ACM Press, October / November 2006.

[BW07] D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC 2007*, *LNCS* 4392, pages 535–554. Springer, Heidelberg, February 2007.

[CDG+18a] J. Chotard, E. Dufour Sans, R. Gay, D. H. Phan, and D. Pointcheval. Decentralized multi-client functional encryption for inner product. In *ASIACRYPT 2018, Part II*, *LNCS* 11273, pages 703–732. Springer, Heidelberg, December 2018.

[CDG+18b] J. Chotard, E. Dufour Sans, R. Gay, D. H. Phan, and D. Pointcheval. Multi-client functional encryption with repetition for inner product. Cryptology ePrint Archive, Report 2018/1021, 2018. https://eprint.iacr.org/2018/1021.

[CDH+00] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-resilient functions and all-or-nothing transforms. In *EUROCRYPT 2000*, *LNCS* 1807, pages 453–469. Springer, Heidelberg, May 2000.

[CGW15] J. Chen, R. Gay, and H. Wee. Improved dual system ABE in prime-order groups via predicate encodings. In *EUROCRYPT 2015, Part II*, *LNCS* 9057, pages 595–624. Springer, Heidelberg, April 2015.

[CGW17] J. Chen, J. Gong, and J. Weng. Tightly secure IBE under constant-size master public key. In *PKC 2017, Part I*, *LNCS* 10174, pages 207–231. Springer, Heidelberg, March 2017.

[CLL+13] J. Chen, H. W. Lim, S. Ling, H. Wang, and H. Wee. Shorter IBE and signatures via asymmetric pairings. In *PAIRING 2012*, *LNCS* 7708, pages 122–140. Springer, Heidelberg, May 2013.

[Coc01] C. Cocks. An identity based encryption scheme based on quadratic residues. In *IMA International Conference on Cryptography and Coding*, pages 360–363. Springer, 2001.

[CPP05] H. Chabanne, D. H. Phan, and D. Pointcheval. Public traceability in traitor tracing schemes. In *EUROCRYPT 2005*, *LNCS* 3494, pages 542–558. Springer, Heidelberg, May 2005.

[CS98] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO'98*, *LNCS* 1462, pages 13–25. Springer, Heidelberg, August 1998.

[CS03] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[CSS12] T.-H. H. Chan, E. Shi, and D. Song. Privacy-preserving stream aggregation with fault tolerance. In *FC 2012*, *LNCS* 7397, pages 200–214. Springer, Heidelberg, February / March 2012.

[CW13] J. Chen and H. Wee. Fully, (almost) tightly secure IBE and dual system groups. In *CRYPTO 2013, Part II*, *LNCS* 8043, pages 435–460. Springer, Heidelberg, August 2013.

[CW14] J. Chen and H. Wee. Semi-adaptive attribute-based encryption and improved delegation for Boolean formula. In *SCN 14*, *LNCS* 8642, pages 277–297. Springer, Heidelberg, September 2014.

[DDM16] P. Datta, R. Dutta, and S. Mukhopadhyay. Functional encryption for inner product with full function privacy. In *PKC 2016, Part I*, *LNCS* 9614, pages 164–195. Springer, Heidelberg, March 2016.

[DDN00] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[DDN03] D. Dolev, C. Dwork, and M. Naor. Nonmalleable cryptography. *SIAM review*, 45(4):727–784, 2003.

[DGP18] E. Dufour Sans, R. Gay, and D. Pointcheval. Reading in the dark: Classifying encrypted digits with functional encryption. Cryptology ePrint Archive, Report 2018/206, 2018. https://eprint.iacr.org/2018/206.

[DH76] W. Diffie and M. Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[DHO16] I. Damgård, H. Haagh, and C. Orlandi. Access control encryption: Enforcing information flow with cryptography. In *TCC 2016-B, Part II*, *LNCS* 9986, pages 547–576. Springer, Heidelberg, October / November 2016.

[DOT18] P. Datta, T. Okamoto, and J. Tomida. Full-hiding (unbounded) multi-input inner product functional encryption from the k-linear assumption. In *PKC 2018, Part II*, *LNCS* 10770, pages 245–277. Springer, Heidelberg, March 2018.

[EHK+13] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. Villar. An algebraic framework for Diffie-Hellman assumptions. In *CRYPTO 2013, Part II*, *LNCS* 8043, pages 129–147. Springer, Heidelberg, August 2013.

[ElG85] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[Emu17] K. Emura. Privacy-preserving aggregation of time-series data with public verifiability from simple assumptions. In *Australasian Conference on Information Security and Privacy*, pages 193–213. Springer, 2017.

[FG18] G. Fuchsbauer and R. Gay. Weakly secure equivalence-class signatures from standard assumptions. In *PKC 2018, Part II*, *LNCS* 10770, pages 153–183. Springer, Heidelberg, March 2018.

[FGKO17] G. Fuchsbauer, R. Gay, L. Kowalczyk, and C. Orlandi. Access control encryption for equality, comparison, and more. In *PKC 2017, Part II*, *LNCS* 10175, pages 88–118. Springer, Heidelberg, March 2017.

[GCD+16] J. Gong, J. Chen, X. Dong, Z. Cao, and S. Tang. Extended nested dual system groups, revisited. In *PKC 2016, Part I*, *LNCS* 9614, pages 133–163. Springer, Heidelberg, March 2016.

[GGG+14] S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F.-H. Liu, A. Sahai, E. Shi, and H.-S. Zhou. Multi-input functional encryption. In *EUROCRYPT 2014*, *LNCS* 8441, pages 578–602. Springer, Heidelberg, May 2014.

[GGH13a] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT 2013*, *LNCS* 7881, pages 1–17. Springer, Heidelberg, May 2013.

[GGH+13b] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.

[GGH+16] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 45(3):882–929, 2016.

[GGHZ16] S. Garg, C. Gentry, S. Halevi, and M. Zhandry. Functional encryption without obfuscation. In *TCC 2016-A, Part II*, *LNCS* 9563, pages 480–511. Springer, Heidelberg, January 2016.

[GHK17] R. Gay, D. Hofheinz, and L. Kohl. Kurosawa-desmedt meets tight security. In *CRYPTO 2017, Part III*, *LNCS* 10403, pages 133–160. Springer, Heidelberg, August 2017.

[GHKP18] R. Gay, D. Hofheinz, L. Kohl, and J. Pan. More efficient (almost) tightly secure structure-preserving signatures. In *EUROCRYPT 2018, Part II*, *LNCS* 10821, pages 230–258. Springer, Heidelberg, April / May 2018.

[GHKW16] R. Gay, D. Hofheinz, E. Kiltz, and H. Wee. Tightly CCA-secure encryption without pairings. In *EUROCRYPT 2016, Part I*, *LNCS* 9665, pages 1–27. Springer, Heidelberg, May 2016.

[GKL+13] S. D. Gordon, J. Katz, F.-H. Liu, E. Shi, and H.-S. Zhou. Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/774, 2013. http://eprint.iacr.org/2013/774.

[GKP+13] S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In *45th ACM STOC*, pages 555–564. ACM Press, June 2013.

[GKSW10] S. Garg, A. Kumarasubramanian, A. Sahai, and B. Waters. Building efficient fully collusion-resilient traitor tracing and revocation schemes. In *ACM CCS 10*, pages 121–130. ACM Press, October 2010.

[GKW15] R. Gay, I. Kerenidis, and H. Wee. Communication complexity of conditional disclosure of secrets and attribute-based encryption. In *CRYPTO 2015, Part II*, *LNCS* 9216, pages 485–502. Springer, Heidelberg, August 2015.

[GKW16] R. Goyal, V. Koppula, and B. Waters. Semi-adaptive security and bundling functionalities made generic and easy. In *TCC 2016-B, Part II*, *LNCS* 9986, pages 361–388. Springer, Heidelberg, October / November 2016.

[GKW18] R. Gay, L. Kowalczyk, and H. Wee. Tight adaptively secure broadcast encryption with short ciphertexts and keys. In *SCN 18*, *LNCS* 11035, pages 123–139. Springer, Heidelberg, September 2018.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299, 1984.

[GMW15] R. Gay, P. Méaux, and H. Wee. Predicate encryption for multi-dimensional range queries from lattices. In *PKC 2015*, *LNCS* 9020, pages 752–776. Springer, Heidelberg, March / April 2015.

[GPS08] S. D. Galbraith, K. G. Paterson, and N. P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[GPSW06] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCS 06*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309.

[GVW12] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO 2012*, *LNCS* 7417, pages 162–179. Springer, Heidelberg, August 2012.

[GVW13] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. In *45th ACM STOC*, pages 545–554. ACM Press, June 2013.

[GVW15a] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. *Journal of the ACM (JACM)*, 62(6):45, 2015.

[GVW15b] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Predicate encryption for circuits from LWE. In *CRYPTO 2015, Part II*, *LNCS* 9216, pages 503–523. Springer, Heidelberg, August 2015.

[HJ12] D. Hofheinz and T. Jager. Tightly secure signatures and public-key encryption. In *CRYPTO 2012*, *LNCS* 7417, pages 590–607. Springer, Heidelberg, August 2012.

[HK07] D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In *CRYPTO 2007*, *LNCS* 4622, pages 553–571. Springer, Heidelberg, August 2007.

[HKS15] D. Hofheinz, J. Koch, and C. Striecks. Identity-based encryption with (almost) tight security in the multi-instance, multi-ciphertext setting. In *PKC 2015*, *LNCS* 9020, pages 799–822. Springer, Heidelberg, March / April 2015.

[Hof17] D. Hofheinz. Adaptive partitioning. In *EUROCRYPT 2017, Part III*, *LNCS* 10212, pages 489–518. Springer, Heidelberg, April / May 2017.

[JL13] M. Joye and B. Libert. A scalable scheme for privacy-preserving aggregation of time-series data. In *FC 2013*, *LNCS* 7859, pages 111–125. Springer, Heidelberg, April 2013.

[Jou00] A. Joux. A one round protocol for tripartite diffie–hellman. In *International algorithmic number theory symposium*, pages 385–393. Springer, 2000.

[Jou04] A. Joux. A one round protocol for tripartite diffie–hellman. *Journal of cryptology*, 17(4):263–276, 2004.

[KD04] K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In *CRYPTO 2004*, *LNCS* 3152, pages 426–442. Springer, Heidelberg, August 2004.

[KDK11] K. Kursawe, G. Danezis, and M. Kohlweiss. Privacy-friendly aggregation for the smart-grid. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 175–191. Springer, 2011.

[Kil06] E. Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC 2006*, *LNCS* 3876, pages 581–600. Springer, Heidelberg, March 2006.

[KLM+18] S. Kim, K. Lewi, A. Mandal, H. Montgomery, A. Roy, and D. J. Wu. Function-hiding inner product encryption is practical. In *International Conference on Security and Cryptography for Networks*, pages 544–562. Springer, 2018.

[KSW08] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT 2008*, *LNCS* 4965, pages 146–162. Springer, Heidelberg, April 2008.

[KSW13] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. *Journal of Cryptology*, 26(2):191–224, April 2013.

[KY02] A. Kiayias and M. Yung. Traitor tracing with constant transmission rate. In *EUROCRYPT 2002*, *LNCS* 2332, pages 450–465. Springer, Heidelberg, April / May 2002.

[LC12] Q. Li and G. Cao. Efficient and privacy-preserving data aggregation in mobile sensing. In *ICNP 2012*, pages 1–10. IEEE Computer Society, 2012.

[LC13] Q. Li and G. Cao. Efficient privacy-preserving stream aggregation in mobile sensing with low aggregation error. In *PETS 2013*, *LNCS* 7981, pages 60–81, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[Lew12] A. B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *EUROCRYPT 2012*, *LNCS* 7237, pages 318–335. Springer, Heidelberg, April 2012.

[Lin16] H. Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In *EUROCRYPT 2016, Part I*, *LNCS* 9665, pages 28–57. Springer, Heidelberg, May 2016.

[Lin17] H. Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 599–629. Springer, Heidelberg, August 2017.

[LJYP14] B. Libert, M. Joye, M. Yung, and T. Peters. Concise multi-challenge CCA-secure encryption and signatures with almost tight security. In *ASIACRYPT 2014, Part II*, *LNCS* 8874, pages 1–21. Springer, Heidelberg, December 2014.

[LL16] K. Lee and D. H. Lee. Two-input functional encryption for inner products from bilinear maps. Cryptology ePrint Archive, Report 2016/432, 2016. http://eprint.iacr.org/2016/432.

[LL18] K. Lee and D. H. Lee. Two-input functional encryption for inner products from bilinear maps. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 101(6):915–928, 2018.

[LPJY14] B. Libert, T. Peters, M. Joye, and M. Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In *EUROCRYPT 2014*, *LNCS* 8441, pages 514–532. Springer, Heidelberg, May 2014.

[LPJY15] B. Libert, T. Peters, M. Joye, and M. Yung. Compactly hiding linear spans - tightly secure constant-size simulation-sound QA-NIZK proofs and applications. In *ASIACRYPT 2015, Part I*, *LNCS* 9452, pages 681–707. Springer, Heidelberg, November / December 2015.

[LT17] H. Lin and S. Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 630–660. Springer, Heidelberg, August 2017.

[LV16] H. Lin and V. Vaikuntanathan. Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In *57th FOCS*, pages 11–20. IEEE Computer Society Press, October 2016.

[Mer78] R. C. Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.

[MRV16] P. Morillo, C. Ràfols, and J. L. Villar. The kernel matrix Diffie-Hellman assumption. In *ASIACRYPT 2016, Part I*, *LNCS* 10031, pages 729–758. Springer, Heidelberg, December 2016.

[NY90] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437. ACM Press, May 1990.

[O'N10] A. O'Neill. Definitional issues in functional encryption. *IACR Cryptology ePrint Archive*, 2010:556, 2010.

[OP01] T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In *CT-RSA 2001*, *LNCS* 2020, pages 159–175. Springer, Heidelberg, April 2001.

[OT08] T. Okamoto and K. Takashima. Homomorphic encryption and signatures from vector decomposition. In *PAIRING 2008*, *LNCS* 5209, pages 57–74. Springer, Heidelberg, September 2008.

[OT09] T. Okamoto and K. Takashima. Hierarchical predicate encryption for inner-products. In *ASIACRYPT 2009*, *LNCS* 5912, pages 214–231. Springer, Heidelberg, December 2009.

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT'99*, *LNCS* 1592, pages 223–238. Springer, Heidelberg, May 1999.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[Riv97] R. L. Rivest. All-or-nothing encryption and the package transform. In *FSE'97*, *LNCS* 1267, pages 210–218. Springer, Heidelberg, January 1997.

[RS92] C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO'91*, *LNCS* 576, pages 433–444. Springer, Heidelberg, August 1992.

[SCR+11] E. Shi, T.-H. H. Chan, E. G. Rieffel, R. Chow, and D. Song. Privacy-preserving aggregation of time-series data. In *NDSS 2011*. The Internet Society, February 2011.

[Sha84] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO'84*, *LNCS* 196, pages 47–53. Springer, Heidelberg, August 1984.

[SS10] A. Sahai and H. Seyalioglu. Worry-free encryption: functional encryption with public keys. In *ACM CCS 10*, pages 463–472. ACM Press, October 2010.

[SW05] A. Sahai and B. R. Waters. Fuzzy identity-based encryption. In *EUROCRYPT 2005*, *LNCS* 3494, pages 457–473. Springer, Heidelberg, May 2005.

[SW14] A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.

[Wat09] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO 2009*, *LNCS* 5677, pages 619–636. Springer, Heidelberg, August 2009.

[Wee14] H. Wee. Dual system encryption via predicate encodings. In *TCC 2014*, *LNCS* 8349, pages 616–637. Springer, Heidelberg, February 2014.

[Wee17] H. Wee. Attribute-hiding predicate encryption in bilinear groups, revisited. In *TCC 2017, Part I*, *LNCS* 10677, pages 206–233. Springer, Heidelberg, November 2017.

RÉSUMÉ

Nos travaux revisitent le chiffrement a clé publique de deux façons : 1) nous donnons une meilleure garantie de sécurité que les chiffrements à clé publique typiques, qui gère de nombreux utilisateurs pouvant coopérer pour réaliser des attaques sophistiquées. Une telle sécurité est nécessaire lorsque l'on considère des schémas de chiffrement largement déployés, où de nombreuses sessions ont lieu de manière concurrentes, ce qui est le cas sur internet 2) nous considérons le chiffrement fonctionnel, introduit en 2011 par Boneh, Sahai et Waters, qui permet un accès fin aux données chiffrées. Il généralise le concept de chiffrement à clé publique traditionnel : une clé secrète maîtresse permet de générer des clés de chiffrement fonctionnelles, qui sont chacune associées à une fonction particulière. Le déchiffrement du chiffrement d'un message m avec une clé de déchiffrement fonctionnelle associée à une fonction f obtiendra la valeur f(m), et aucune autre information à propos du message chiffré m.

MOTS CLÉS

Chiffrement à clé publique, sécurité accrue, chiffrement fonctionnel

ABSTRACT

Our work revisits public-key encryption in two ways: 1) we provide stronger security guarantee that typical public-key encryption, which handles many users than can collude to perform sophisticated attacks. This is necessary when considering widely deployed encryption schemes, where many sessions are performed concurrently, as in the case on the Internet; 2) we consider so-called functional encryption, introduced by Boneh, Sahai, Waters in 2011, that permits fine-grained access to the encrypted data. It generalizes traditional public-key encryption is that a master secret key is used to generate so-called functional decryption keys, each of which is associated with a particular function. An encryption of a message m, together with a functional decryption key associated with the function f, decrypts the value f(m), without revealing any additional information about the encrypted message m.

KEYWORDS

Public-key encryption, tight security, functional encryption