D4.10 Annual Report on: Information-leakage analysis and countermeasure Y2

Document Identification			
Status	Final	**Due Date**	31/12/2019
Version	1.0	**Submission Date**	30/12/2019

Related WP	WP4	**Document Reference**	D4.10
Related Deliverable(s)	D4.1, D4.2, D4.8	**Dissemination Level(*)**	PU
Lead Participant	ENS	**Lead Author**	Michel Abdalla
Contributors	ENS,UEDIN	**Reviewers**	Clément Gentilucci (FUAS)

Keywords:
Functional Encryption Schemes, Quantum-Safe, Web Analytics

This document is issued within the framework and for the purpose of the FENTEC project. This project has received funding from the European Union's Horizon2020 under Grant Agreement No. 780108. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

(*) Dissemination level.-PU: Public, fully open, e.g. web; CO: Confidential, restricted under conditions set out in Model Grant Agreement; CI: Classified, Int = Internal Working Document, information as referred to in Commission Decision 2001/844/EC.

Document Information

List of Contributors	
Name	Partner
Michel Abdalla	ENS
Hendrik Waldner	UEDIN

Document History			
Version	Date	Change editors	Changes
0.1	18/12/2019	Hendrik Waldner (UEDIN)	ToC
0.2	26/12/2019	Michel Abdalla (ENS)	Version for reviewing
0.3	28/12/2019	Michel Abdalla (ENS)	Addressed reviewers comments
1	30/12/2019	Michel Abdalla (ENS)	Final version

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable Leader	Michel Abdalla (ENS)	30/12/2018
Technical Manager	Michel Abdalla (ENS)	30/12/2018
Quality Manager	Diego Esteban (ATOS)	30/12/2018
Project Coordinator	Francisco Gala (ATOS)	30/12/2018

Table of Contents

List of Figures

List of Acronyms

Acronym	Description
DCR	Decisional Composite Residuosity
DDH	Decisional Diffie-Helmman
DMCFE	Decentralized Multi-Client Functional Encryption
FE	Functional Encryption
LWE	Learning With Errors
MCFE	Multi-Client Functional Encryption
MIFE	Multi-Input Functional Encryption
PPT	Probabilistic Polynomial Time
PRF	Pseudorandom Functions
ROM	Random-Oracle Model
SE	Symmetric-Key Encryption
WP	Work Package

Executive Summary

Functional encryption is an important new paradigm allowing for a fine-grained access control over the encrypted data. Since its introduction, several feasibility results for general functionalities and concrete and efficient realizations for restricted functionalities of practical interest have been proposed. Despite its general appeal, the security guarantees provided by existing solutions may not always be sufficient for a given application. In the case of the inner-product functionality, for instance, an attacker may able to recover the entire encrypted message if it is able to obtain a large enough number of ciphertexts and functional decryption keys.

To address the issue of information leakage in functional encryption schemes, we propose the use of encryption labels as one possible counter-measure for the inherent leakage in the multi-input setting. In the latter setting, the decryption procedure takes as input n different ciphertexts and outputs a function applied to the n corresponding plaintexts. The goal in this case is to limit the amount of possible mix-and-match that can take place during decryption by only allowing the functional decryption key to decrypt ciphertexts that were generated with respect to the same label.

Towards this goal, we describe a new construction, developed by Abdalla, Benhamouda, and Gay [1] in the context of the FENTEC project. which allows for encryption labels. The new construction can be built generically from any single-input functional inner-product encryption satisfying some simple structural properties with the help of a pseudorandom function. In the particular case of the instantiations based on the Learning-With-Errors and the Decisional Composite Residuosity computational assumptions, the resulting schemes allow for the computation of inner products of arbitrary sizes.

1 Introduction

Functional Encryption (FE) [8, 11] is a generalization of the notion of public-key encryption, which allows fine-grained access control over encrypted data. Besides the classical encryption and decryption procedures, functional encryption schemes consists of a key derivation algorithm, which allows the owner of a master secret key to derive keys with more restricted capabilities. These derived keys sk_f are called functional decryption keys and are associated with a function f. Using the key sk_f for the decryption of a ciphertext $\mathsf{Enc}(x)$ generates the output $f(x)$. In its multi-input extension, the decryption procedure takes as input n different ciphertexts and outputs a function applied to the n corresponding plaintexts[1], where each input is called a slot. In both the standard and multi-input settings, the decryption procedure reveals no more information about the underlying plaintext or plaintexts than the output of f applied to these plaintexts.

Even though functional encryption allows for a more fine-grained control over the decryption capabilities of different parties, the security guarantees provided by existing solutions may not always be sufficient for a given application. For instance, in the case of the inner-product functionality, an attacker will always be able to recover the entire plaintext message if it is able to obtain a large enough number of ciphertexts and functional decryption keys.

1.1 Purpose of the Document

In this report, we propose one possible counter-measure to the inherent leakage of functional encryption schemes in the multi-input setting. More precisely, we consider the use of labels during the encryption procedure to limit the amount of possible mix-and-match that can take place during decryption. In particular, when encryption labels are used, the user in possession of a functional decryption key sk_f will only be able to decrypt ciphertexts that were generated with respect to the same label.

Towards achieving the above goal, we first present a new multi-client functional encryption construction (MCFE) by Abdalla, Benhamouda, and Gay [1], which can handle encryption labels and is proven secure in the standard model under standard assumptions. The new construction is generic, in the sense that it can transform any single-input FE satisfying two simple structural properties into a multi-client FE, under the same assumption with the help of a pseudorandom function. In particular, by using previously known single-input FE schemes for the inner-product functionality based on computational assumptions such as plain Decisional Diffie-Hellman (DDH), Learning With Errors (LWE) and Decisional Composite Residuosity (DCR), we can obtain MCFE schemes for inner products supporting encryption labels based on the same assumptions.

Finally, since the main scheme in [1] guarantees security *only* when the adversary queries each encryption slot at least once, we also describe a compiler by Abdalla, Benhamouda, and Gay [1], which removes this condition.

1.2 Structure and Methodology

Section 2 first recalls some of the definitions and basic tools that are used in the remainder of the document, such as notations, complexity assumptions, and security definitions for multi-input and multi-client functional encryption. Section 3 then describes our main contribution, which is

[1] When these plaintexts are generated by different parties, we refer to this extension as multi-client.

the generic construction of multi-client functional inner-product encryption from a single-input functional inner-product encryption. Next, Section 4 describes a compiler that improves the security of the underlying multi-client functional encryption scheme and guarantees security even when the adversary does not query every encryption slot at least once. Finally, Section 5 concludes by recapping our contributions and discussing future research directions.

2 Basic tools

In this section, we recall some of the definitions and basic tools that will be used in the remainder of the document.

2.1 Notation and conventions

We denote with $\lambda \in \mathbb{N}$ a security parameter. A *probabilistic polynomial time* (PPT) algorithm $\mathcal{A}$ is a randomized algorithm for which there exists a polynomial $p(\cdot)$ such that for every input x the running time of $\mathcal{A}(x)$ is bounded by $p(|x|)$. We say that a function $\varepsilon : \mathbb{N} \to \mathbb{R}^+$ is *negligible* if for every positive polynomial $p(\lambda)$ there exists $\lambda_0 \in \mathbb{N}$ such that for all $\lambda > \lambda_0$: $\varepsilon(\lambda) < 1/p(\lambda)$. If S is a set, $x \xleftarrow{R} S$ denotes the process of selecting x uniformly at random in S. If $\mathcal{A}$ is a probabilistic algorithm, $y \xleftarrow{R} \mathcal{A}(\cdot)$ denotes the process of running $\mathcal{A}$ on some appropriate input and assigning its output to y. For a positive integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$. We denote vectors $\mathbf{x} = (x_i)$ and matrices $\mathbf{A} = (a_{i,j})$ in bold. For a set S (resp. vector $\mathbf{x}$) $|S|$ (resp. $|\mathbf{x}|$) denotes its cardinality (resp. number of entries). Also, given two vectors $\mathbf{x}$ and $\mathbf{x}'$ we denote by $\mathbf{x}\|\mathbf{x}'$ their concatenation. By $\equiv$, we denote the equality of statistical distributions, and for any $\varepsilon > 0$, we denote by $\approx_\varepsilon$ the ε-statistical difference of two distributions.

2.2 Multi-Client Functional Encryption

In this section, we recall the definition of MCFE [10]. The definition we present is almost the same as in [2], with the differences that a stronger security definition (see Remark 1) is used in the introduction of a master public key mpk, so that *public-key* functional encryption becomes a particular case of MCFE.

Definition 1 ***(Multi-Client Functional Encryption)*** *Let $\mathcal{F} = \{\mathcal{F}_\rho\}_\rho$ be a family (indexed by ρ) of sets $\mathcal{F}_\rho$ of functions $f : \mathcal{X}_{\rho,1} \times \cdots \times \mathcal{X}_{\rho,n_\rho} \to \mathcal{Y}_\rho$.[2] Let $\mathsf{Labels} = \{0,1\}^*$ or $\{\bot\}$ be a set of labels. A multi-client functional encryption scheme (MCFE) for the function family $\mathcal{F}$ and the label set Labels is a tuple of five algorithms $\mathsf{MCFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$:*

$\mathsf{Setup}(1^\lambda, 1^n)$**:** *Takes as input a security parameter λ and the number of parties n, and generates public parameters pp. The public parameters implicitly define an index ρ corresponding to a set $\mathcal{F}_\rho$ of n-ary functions (i.e., $n = n_\rho$).*

$\mathsf{KeyGen}(\mathsf{pp})$**:** *Takes as input the public parameters pp and outputs n secret keys $\{\mathsf{sk}_i\}_{i\in[n]}$, a master secret key msk, and a master public key mpk.*

$\mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, f)$**:** *Takes as input the public parameters pp, the master secret key msk and a function $f \in \mathcal{F}_\rho$, and outputs a functional decryption key sk_f.*

$\mathsf{Enc}(\mathsf{pp}, \mathsf{mpk}, \mathsf{sk}_i, x_i, \ell)$**:** *Takes as input the public parameters pp, a master public key mpk, a secret key sk_i, a message $x_i \in \mathcal{X}_{\rho,i}$ to encrypt, a label $\ell \in \mathsf{Labels}$, and outputs ciphertext $\mathsf{ct}_{i,\ell}$.*

$\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{ct}_{1,\ell}, \ldots, \mathsf{ct}_{n,\ell})$**:** *Takes as input the public parameters pp, a functional key sk_f and n ciphertexts under the same label ℓ and outputs a value $y \in \mathcal{Y}_\rho$.*

[2] All the functions inside the same set $\mathcal{F}_\rho$ have the same domain and the same range.

A scheme MCFE *is correct, if for all* $\lambda, n \in \mathbb{N}$, $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, $f \in \mathcal{F}_\rho$, $\ell \in \mathsf{Labels}$, $x_i \in \mathcal{X}_{\rho,i}$, *when* $(\{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ *and* $\mathsf{sk}_f \leftarrow \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, f)$, *we have for* $\mathbf{x} = (x_1, \ldots, x_n)$*:*

$$\Pr\left[\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{Enc}(\mathsf{pp}, \mathsf{mpk}, \mathsf{sk}_1, x_1, \ell), \ldots, \mathsf{Enc}(\mathsf{pp}, \mathsf{mpk}, \mathsf{sk}_n, x_n, \ell)) = f(\mathbf{x})\right] = 1.$$

When ρ is clear from context, the index ρ is omitted. Note that the case of (single-input) functional encryption as defined in [8, 11] corresponds to the case $n = 1$, and $\mathsf{Labels} = \{\bot\}$. For such schemes, we also consider the *public-key* variant, where $\mathsf{sk}_1 = \bot$, that is, the encryption algorithm only requires the public parameters pp and the master public key mpk to encrypt the message x_1. In this setting, sk_1 is omitted.

Except for public-key single-input functional encryption, the master public-key can be included in each secret key sk_i and we omit it.

We follow the notation of [2] here, where the algorithm Setup only generates public parameters that determine the set of functions for which functional decryption keys can be created, and the secret/encryption keys and the master secret keys are generated by another algorithm KeyGen, while the functional decryption keys are generated by KeyDer.

In the following, we recap the security definition as stated in Abdalla et al. [1]. They define security as adaptive left-or-right indistinguishability under both static (sta), and adaptive (adt) corruption. They also considered two variants of these notions $(\mathrm{any}, \mathrm{pos}^+)$ related to the number of encryption queries asked by the adversary for each slot.

Definition 2 ***(Security of MCFE)*** *Let* MCFE *be an MCFE scheme,* $\mathcal{F} = \{\mathcal{F}_\rho\}_\rho$ *a function family indexed by* ρ *and* Labels *a label set. For* $\mathrm{xx} \in \{\mathrm{sta}, \mathrm{adt}\}$, $\mathrm{yy} \in \{\mathrm{any}, \mathrm{pos}^+\}$, *and* $\beta \in \{0, 1\}$, *we define the experiment* $\mathrm{xx\text{-}yy\text{-}IND}_\beta^{\mathsf{MCFE}}$ *in Fig. 1, where the oracles are defined as:*

Corruption oracle $\mathsf{QCor}(i)$**:** *Outputs the encryption key* sk_i *of slot* i*. We denote by* $\mathcal{CS}$ *the set of corrupted slots at the end of the experiment.*

Left-Right oracle $\mathsf{QLeftRight}(i, x_i^0, x_i^1, \ell)$**:** *Outputs* $\mathsf{ct}_{i,\ell} = \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, x_i^\beta, \ell)$ *on a query* (i, x_i^0, x_i^1, ℓ)*. We denote by* $Q_{i,\ell}$ *the number of queries of the form* $\mathsf{QLeftRight}(i, \cdot, \cdot, \ell)$*.*

Encryption oracle $\mathsf{QEnc}(i, x_i, \ell)$**:** *outputs* $\mathsf{ct}_{i,\ell} = \mathsf{Enc}(\mathsf{pp}, \mathsf{mpk}, \mathsf{sk}_i, x_i, \ell)$ *on a query* (i, x_i, ℓ)*.*

Key derivation oracle $\mathsf{QKeyD}(f)$**:** *Outputs* $\mathsf{sk}_f = \mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, f)$*.*

and where Condition (*) *holds if all the following conditions hold:*

- *If* $i \in \mathcal{CS}$ *(i.e., slot* i *is corrupted): for any query* $\mathsf{QLeftRight}(i, x_i^0, x_i^1, \ell)$, $x_i^0 = x_i^1$*.*
- *For any label* $\ell \in \mathsf{Labels}$*, for any family of queries* $\{\mathsf{QLeftRight}(i, x_i^0, x_i^1, \ell)$ *or* $\mathsf{QEnc}(i, x_i, \ell)\}_{i\in[n]\setminus\mathcal{CS}}$*, for any family of inputs* $\{x_i \in \mathcal{X}_{\rho,i}\}_{i\in\mathcal{CS}}$*, for any query* $\mathsf{QKeyD}(f)$*, we define* $x_i^0 := x_i$ *and* $x_i^1 := x_i$ *for any slot* $i \in \mathcal{CS}$ *and any slot queried to* $\mathsf{QEnc}(i, x_i, \ell)$*, and we require that:*

$$f(\mathbf{x}^0) = f(\mathbf{x}^1) \qquad \text{where } \mathbf{x}^b = (x_1^b, \ldots, x_n^b) \text{ for } b \in \{0, 1\} \ .$$

  We insist that if one index $i \notin \mathcal{CS}$ *is not queried for the label* ℓ*, there is no restriction.*

- *When* $\mathrm{yy} = \mathrm{pos}^+$*: for any slot* $i \in [n]$ *and* $\ell \in \mathsf{Labels}$*, if* $Q_{i,\ell} > 0$*, then for any slot* $j \in [n]\setminus\mathcal{CS}$*,* $Q_{j,\ell} > 0$*. In other words, for any label, either the adversary makes no left-right encryption query or makes at least one left-right encryption query for each slot* $i \in [n]\setminus\mathcal{CS}$*.*

$\textbf{sta-yy-IND}^{\mathsf{MCFE}}_{\beta}(\lambda, n, \mathcal{A})$

$\mathcal{CS} \leftarrow \mathcal{A}(1^\lambda, 1^n)$

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$

$(\{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$

$\alpha \leftarrow \mathcal{A}^{\mathsf{QEnc}(\cdot,\cdot,\cdot),\mathsf{QLeftRight}(\cdot,\cdot,\cdot,\cdot),\mathsf{QKeyD}(\cdot)}(\mathsf{pp}, \mathsf{mpk}, \{\mathsf{sk}_i\}_{i\in\mathcal{CS}})$

Output: α if Condition (*) is satisfied, 0 otherwise.

$\textbf{adt-yy-IND}^{\mathsf{MCFE}}_{\beta}(\lambda, n, \mathcal{A})$

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$

$(\{\mathsf{sk}_i\}_{i\in[n]}, \mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$

$\alpha \leftarrow \mathcal{A}^{\mathsf{QCor}(\cdot),\mathsf{QEnc}(\cdot,\cdot,\cdot),\mathsf{QLeftRight}(\cdot,\cdot,\cdot,\cdot),\mathsf{QKeyD}(\cdot)}(\mathsf{pp}, \mathsf{mpk})$

Output: α if Condition (*) is satisfied, 0 otherwise.

Figure 1: Security games for MCFE

We define the advantage of an adversary $\mathcal{A}$ in the following way:

$$\mathsf{Adv}^{\text{xx-yy-IND}}_{\mathsf{MCFE},\mathcal{A}}(\lambda, n) = \left| \Pr[\text{xx-yy-IND}^{\mathsf{MCFE}}_0(\lambda, n, \mathcal{A}) = 1] - \Pr[\text{xx-yy-IND}^{\mathsf{MCFE}}_1(\lambda, n, \mathcal{A}) = 1] \right| .$$

A multi-client functional encryption scheme MCFE *is* xx-yy-*IND secure, if for any n, for any polynomial-time adversary $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:* $\mathsf{Adv}^{\text{xx-yy-IND}}_{\mathsf{MCFE},\mathcal{A}}(\lambda, n) \leq \mathsf{negl}(\lambda)$.

We omit n when it is clear from the context. We also often omit $\mathcal{A}$ from the parameter of experiments or games when it is clear from the context.

Remark 1 (The role of the oracle QEnc**)** *The security definitions we give are slightly stronger than those given in [2], since the oracle* QEnc *gives out information that is not captured by Condition (*), for* pos^+*, hence the use of the notation* pos^+ *instead of* pos *in [2]. For* any*, this addition of* QEnc *has no effect, as* QEnc *queries can be simulated using* QLeftRight*. But for* pos^+/pos*, there is no equivalence in general between the security definition with and without the encryption oracle.*

2.3 Decentralized Multi-Client Functional Encryption

Now, we recap the definition of decentralized multi-client functional encryption (DMCFE) as initally introduced in [9] and modified in [1]. As in the definition of MCFE, the algorithm Setup, which generates public parameters defining in particular the set of functions, is separated from the algorithm KeyGen. Since Abdalla et al. [1] do not consider public-key variants of DMCFE, the master public key mpk can be completely omitted.

Definition 3 ***(Decentralized Multi-Client Functional Encryption)*** *Let $\mathcal{F} = \{\mathcal{F}_\rho\}_\rho$ be a family (indexed by ρ) of sets $\mathcal{F}_\rho$ of functions $f\colon \mathcal{X}_{\rho,1} \times \cdots \times \mathcal{X}_{\rho,n_\rho} \to \mathcal{Y}_\rho$. Let* Labels $= \{0,1\}^*$ *or $\{\bot\}$ be a set of labels. A decentralized multi-client functional encryption scheme (DMCFE) for the*

function family $\mathcal{F}$ *and the label set* Labels *is a tuple of six algorithms* DMCFE = (Setup, KeyGen, KeyDerShare, KeyDerComb, Enc, Dec)*:*

$\mathsf{Setup}(1^\lambda, 1^n)$ *is defined as for MCFE in Definition 1.*

$\mathsf{KeyGen}(\mathsf{pp})$**:** *Takes as input the public parameters* pp *and outputs* n *secret keys* $\{\mathsf{sk_i}\}_{i\in[n]}$.

$\mathsf{KeyDerShare}(\mathsf{pp}, \mathsf{sk}_i, f)$**:** *Takes as input the public parameters* pp*, a secret key* sk_i *from position* i *and a function* $f \in \mathcal{F}_\rho$*, and outputs a partial functional decryption key* $\mathsf{sk}_{i,f}$.

$\mathsf{KeyDerComb}(\mathsf{pp}, \mathsf{sk}_{1,f}, \ldots, \mathsf{sk}_{n,f})$**:** *Takes as input the public parameters* pp*,* n *partial functional decryption keys* $\mathsf{sk}_{1,f}, \ldots, \mathsf{sk}_{n,f}$ *and outputs the functional decryption key* sk_f.

$\mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, x_i, \ell)$ *is defined as for MCFE in Definition 1.*

$\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{ct}_{1,\ell}, \ldots, \mathsf{ct}_{n,\ell})$ *is defined as for MCFE in Definition 1.*

A scheme DMCFE *is correct, if for all* $\lambda, n \in \mathbb{N}$, $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, $f \in \mathcal{F}_\rho$, $\ell \in \mathsf{Labels}$, $x_i \in \mathcal{X}_{\rho,i}$*, when* $\{\mathsf{sk}_i\}_{i\in[n]} \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, $\mathsf{sk}_{i,f} \leftarrow \mathsf{KeyDerShare}(\mathsf{sk}_i, f)$ *for* $i \in [n]$*, and* $\mathsf{sk}_f \leftarrow \mathsf{KeyDerComb}(\mathsf{pp}, \mathsf{sk}_{1,f}, \ldots, \mathsf{sk}_{n,f})$*, we have*

$$\Pr\left[\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_1, x_1, \ell), \ldots, \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_n, x_n, \ell)) = f(x_1, \ldots, x_n)\right] = 1 \ .$$

We remark that there is no master secret key msk. Furthermore, similarly to [9], the definition of DMCFE does not explicitly ask the setup to be decentralized.

We consider a similar security definition for the decentralized multi-client scheme. We point out that contrary to [9], this definition of DMCFE does not differentiate encryption keys from secret keys. This is without loss of generality, as corruptions in [9] only allow to corrupt both keys at the same time.

Definition 4 ***(Security of DMCFE)*** *The* xx-yy*-IND security notion of an DMCFE scheme (*$\mathrm{xx} \in \{\mathrm{sta}, \mathrm{adt}\}$ *and* $\mathrm{yy} \in \{\mathrm{any}, \mathrm{pos}^+\}$*) is similar to the one of an MCFE (Definition 2), except that there is no master secret key* msk *and the key derivation oracle is now defined as:*

Key derivation oracle $\mathsf{QKeyD}(f)$**:** *Computes* $\mathsf{sk}_{i,f} := \mathsf{KeyDerShare}(\mathsf{pp}, \mathsf{sk}_i, f)$ *for* $i \in [n]$ *and outputs* $\{\mathsf{sk}_{i,f}\}_{i\in[n]}$.

2.4 Inner-Product Functionality

We describe the functionalities supported by the constructions in this deliverable. The index of the family is defined as $\rho = (\mathcal{R}, n, m, X, Y)$ where $\mathcal{R}$ is either $\mathbb{Z}$ or $\mathbb{Z}_L$ for some integer L, and n, m, X, Y are positive integers. If X, Y are omitted, then $X = Y = L$ is used (i.e., no constraint). This defines $\mathcal{F}_\rho^{\mathsf{ip}} = \{f_{\mathbf{y}_1,\ldots,\mathbf{y}_n} : (\mathcal{R}^m)^n \to \mathcal{R}\}$ where

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

where the vectors satisfy the following bounds: $\|\mathbf{x}_i\|_\infty < X$, $\|\mathbf{y}_i\|_\infty < Y$ for $i \in [n]$, and where $\mathbf{x} \in \mathcal{R}^{mn}$ and $\mathbf{y} \in \mathcal{R}^{mn}$ are the vectors corresponding to the concatenation of the n vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and $\mathbf{y}_1, \ldots, \mathbf{y}_n$ respectively.

2.5 Pseudorandom Functions (PRF)

The constructions that we present in Sections 3 and 4 rely on a pseudorandom function $\mathsf{PRF}_{\mathsf{K}}(\ell)$, indexed by a key $\mathsf{K} \in \{0,1\}^{\lambda}$, that takes as input a label $\ell \in \mathsf{Labels}$, and outputs a value in the output space $\mathcal{Z}$. For a uniformly random key $\mathsf{K} \leftarrow \{0,1\}^{\lambda}$, this function is computationally indistinguishable from a truly random function from Labels to $\mathcal{Z}$.

Definition 5 (PRF) *For any* PRF *from* Labels *to* $\mathcal{Z}$*, any bit* $\beta \in \{0,1\}$*, any security parameter* λ*, and any adversary* $\mathcal{A}$*, we define the experiment* $\mathbf{IND}^{\mathsf{PRF}}{}_{\beta}$ *as follows:*

$\mathbf{IND}^{\mathsf{PRF}}_{\beta}(\lambda, \mathcal{A})$

$\mathsf{K} \leftarrow \{0,1\}^{\lambda}$

$\alpha \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{PRF}}(\cdot)}(1^{\lambda})$

Output: α

Figure 2: Security games for PRF.

where the oracle $\mathcal{O}_{\mathsf{PRF}}(\ell)$ *returns* $\mathsf{PRF}_{\mathsf{K}}(\ell)$ *if* $\beta = 0$*;* $\mathsf{RF}(\ell)$ *otherwise, where* RF *denotes a random function computed on the fly.*

We define the advantage of an adversary $\mathcal{A}$ *in the following way:*

$$\mathsf{Adv}_{\mathsf{PRF},\mathcal{A}}(\lambda) = \left| \Pr[\mathrm{IND}_0^{\mathsf{PRF}}(\lambda, \mathcal{A}) = 1] - \Pr[\mathrm{IND}_1^{\mathsf{PRF}}(\lambda, \mathcal{A}) = 1] \right| .$$

A PRF is secure, if for any any polynomial-time adversary $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:* $\mathsf{Adv}_{\mathsf{PRF},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

2.6 Symmetric-Key Encryption (SE)

A symmetric encryption with key space $\mathcal{K}$ consists of the following PPT algorithms:

- $\mathsf{Enc}(K, m)$: given a symmetric key K and a message m, outputs a ciphertext.
- $\mathsf{Dec}(K, \mathsf{ct})$: given a symmetric key K and a ciphertext ct, outputs a message (or $\perp$ if it fails to decrypt).

For all message in the message space, we have $\Pr[\mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m] = 1$, where the probability is taken over the random choice of $K \leftarrow \mathcal{K}$. We say a symmetric-key encryption with key space $\mathcal{K}$ is compatible with a PRF with output space $\mathcal{Z}$ if $\mathcal{K} = \mathcal{Z}$.

Definition 6 (SE) *For any* SE *with key space* $\mathcal{K}$*, any bit* $\beta \in \{0,1\}$*, any security parameter* λ*, and any adversary* $\mathcal{A}$*, we define the experiment* $\mathbf{IND}^{\mathsf{PRF}}{}_{\beta}$ *as follows:*

where the oracle $\mathcal{O}_{\mathsf{SE}}(m_0, m_1)$ *returns* $\mathsf{Enc}(K, m_{\beta})$.

We define the advantage of an adversary $\mathcal{A}$ *in the following way:*

$$\mathsf{Adv}_{\mathsf{SE},\mathcal{A}}(\lambda, n) = \left| \Pr[\mathrm{IND}_0^{\mathsf{PRF}}(\lambda, \mathcal{A}) = 1] - \Pr[\mathrm{IND}_1^{\mathsf{SE}}(\lambda, \mathcal{A}) = 1] \right| .$$

A SE is secure, if for any any polynomial-time adversary $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:* $\mathsf{Adv}_{\mathsf{SE},\mathcal{A}}(\lambda) \leq \mathsf{negl}(\lambda)$.

$\mathbf{IND}^{\mathsf{SE}}_{\beta}(\lambda, \mathcal{A})$

$\mathsf{K} \leftarrow \mathcal{K}$

$\alpha \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{SE}}(\cdot)}(1^{\lambda})$

Output: α

Figure 3: Security games for SE.

3 Labeled MCFE from Public-Key Single-Input FE

In this section, we present the MCFE scheme with labels as constructed in [1].

3.1 Construction

The construction by Abdalla et al. [1] resembles the multi-input FE from [4], in which an inner layer of information-theoretic one-time FE is combined with an outer layer of single-input FE. In [1], the authors managed to extend this paradigm to the setting where the encryption additionally takes a label as an additional input. They do so by replacing the one-time pads with pads which are pseudorandom for all used labels ℓ, using techniques similar to those used in [2] to decentralize the generation of functional secret keys.

The detailed construction is given in Fig. 4 and is based on any single-input FE satisfying two simple structural properties: *Two-step decryption* and *linear encryption*. These properties, originally defined in [4] and recalled below (converted to the public-key setting), are satisfied by all known existing single-input FE for inner products (e.g., [3, 7]).

Definition 7 (Two-step decryption [4]) *A public-key FE scheme* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ *for the function ensemble* $\mathcal{F}^{\mathsf{ip}}_{\rho}, \rho = (\mathbb{Z}, 1, m, X, Y)$ *satisfies the* two-step decryption *property if it admits PPT algorithms* $\mathsf{Setup}^{\star}$, $\mathsf{Dec}_1, \mathsf{Dec}_2$ *and an encoding function* $\mathcal{E}$ *such that:*

1. *For all* $\lambda \in \mathbb{N}, \mathsf{Setup}^{\star}(1^{\lambda}, 1^{n})$ *outputs* pp *where* pp *includes* $\rho = (\mathbb{Z}, 1, m, X, Y)$ *and a bound* $B \in \mathbb{R}^{+}$*, as well as 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* $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}), \mathbf{x} \in \mathbb{Z}^m, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{mpk}, \mathbf{x}), \mathbf{y} \in \mathbb{Z}^m$*, and* $\mathsf{sk} \leftarrow \mathsf{KeyDer}(\mathsf{msk}, \mathbf{y})$*, we have*

$$\mathsf{Dec}_1(\mathsf{pp}, \mathsf{sk}, \mathsf{ct}) = \mathcal{E}(\langle \mathbf{x}, \mathbf{y} \rangle \bmod L, \mathsf{noise}) \ ,$$

   for some $\mathsf{noise} \in \mathbb{Z}$ *that depends on* ct *and* sk*. Furthermore, it holds that* $Pr[|\mathsf{noise}| < B] = 1 - \mathsf{negl}(\lambda)$*, where the probability is taken over the random coins of* KeyGen *and* KeyDer*. Note that there is no restriction on the norm of* $\langle \mathbf{x}, \mathbf{y} \rangle$ *here.*

3. *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*

$$\mathcal{E}(\gamma, \mathsf{noise}) \circ \mathcal{E}(\gamma', \mathsf{noise}') = \mathcal{E}(\gamma + \gamma' \bmod L, \mathsf{noise} + \mathsf{noise}') \ .$$

4. *For all* $\gamma < n \cdot m \cdot X \cdot Y$*, and* $|\mathsf{noise}| < n \cdot B$*,* $\mathsf{Dec}_2(\mathsf{pp}, \mathcal{E}(\gamma, \mathsf{noise})) = \gamma$*.*

$\underline{\mathsf{Setup}(1^\lambda, 1^n)}$:

$\mathsf{pp}_{\mathsf{ipfe}} \leftarrow \mathsf{Setup}^\star_{\mathsf{ipfe}}(1^\lambda, 1^n)$, with L implicitly defined from $\mathsf{pp}_{\mathsf{ipfe}}$

Return $\mathsf{pp} = \mathsf{pp}_{\mathsf{ipfe}}$

$\underline{\mathsf{KeyGen}(\mathsf{pp})}$:

$(\mathsf{msk}_{\mathsf{ipfe}}, \mathsf{mpk}_{\mathsf{ipfe}}) \leftarrow \mathsf{KeyGen}_{\mathsf{ipfe}}(\mathsf{pp}_{\mathsf{ipfe}})$; $\mathsf{msk} := \mathsf{msk}_{\mathsf{ipfe}}$

For $i \in [n]$, $j > i$: $\mathsf{K}_{i,j} = \mathsf{K}_{j,i} \leftarrow \{0,1\}^\lambda$

Return $\{\mathsf{sk}_i = (\mathsf{mpk}, \{\mathsf{K}_{i,j}\}_{j\in[n]})\}_{i\in[n]}$ and msk

$\underline{\mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, \mathbf{x}_i \in \mathcal{R}^m, \ell \in \mathsf{Labels})}$:

Parse $\mathsf{sk}_i = (\mathsf{mpk}_{\mathsf{ipfe}}, \{\mathsf{K}_{i,j}\}_{j\in[n]})$

$\mathbf{t}_{i,\ell} := \sum_{j\neq i} (-1)^{j<i} \mathsf{PRF}_{\mathsf{K}_{i,j}}(\ell) \in \mathbb{Z}_L^{mn}$

$\mathbf{w}_i := (\mathbf{0}\| \ldots \|\mathbf{0}\|\mathbf{x}_i\|\mathbf{0}\| \ldots \|\mathbf{0}) + \mathbf{t}_{i,\ell} \bmod L$

$\mathsf{ct}_i \leftarrow \mathsf{Enc}_{\mathsf{ipfe}}(\mathsf{pp}_{\mathsf{ipfe}}, \mathsf{mpk}_{\mathsf{ipfe}}, \mathbf{w}_i)$

Return ct_i

$\underline{\mathsf{KeyDer}(\mathsf{pp}, \mathsf{msk}, \mathbf{y} \in \mathcal{R}^{mn})}$:

Return $\mathsf{sk}_{\mathbf{y}} \leftarrow \mathsf{KeyDer}_{\mathsf{ipfe}}(\mathsf{pp}_{\mathsf{ipfe}}, \mathsf{msk}_{\mathsf{ipfe}}, \mathbf{y})$

$\underline{\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_{\mathbf{y}}, \{\mathsf{ct}_i\}_{i\in[n]})}$:

For $i \in [n]$, $\mathcal{E}(\langle \mathbf{w}_i, \mathbf{y}\rangle \bmod L, \mathsf{noise}_i) \leftarrow \mathsf{Dec}_{\mathsf{ipfe},1}(\mathsf{pp}_{\mathsf{ipfe}}, \mathsf{sk}_{\mathbf{y}}, \mathsf{ct}_i)$

Return $\mathsf{Dec}_{\mathsf{ipfe},2}(\mathsf{pp}_{\mathsf{ipfe}}, \mathcal{E}(\langle \mathbf{w}_1, \mathbf{y}\rangle \bmod L, \mathsf{noise}_1)) \circ \cdots \circ \mathcal{E}(\langle \mathbf{w}_n, \mathbf{y}\rangle \bmod L, \mathsf{noise}_n))$

Figure 4: Inner-Product MCFE for $\mathcal{F}_\rho, \rho = (\mathbb{Z}, n, m, X, Y)$ built from a public-key FE $\mathsf{FE} := (\mathsf{Setup}_{\mathsf{ipfe}}, \mathsf{Enc}_{\mathsf{ipfe}}, \mathsf{KeyDer}_{\mathsf{ipfe}}, \mathsf{Dec}_{\mathsf{ipfe}})$ for $\mathcal{F}_{\rho_{\mathsf{ipfe}}}, \rho_{\mathsf{ipfe}} = (\mathbb{Z}, 1, n \cdot m, 2X, Y)$.

Definition 8 (Linear encryption [4]) *A secret-key FE scheme* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDer}, \mathsf{Enc}, \mathsf{Dec})$ *is said to satisfy the linear encryption property if 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:*

$$\mathsf{Add}(\mathsf{Enc}(\mathsf{pp}, \mathsf{msk}, \mathbf{x}), \mathbf{x}'), \quad \textit{and} \quad \mathsf{Enc}\big(\mathsf{pp}, \mathsf{msk}, (\mathbf{x} + \mathbf{x}' \bmod L)\big) \ .$$

Recall that the value $L \in \mathbb{N}$ *is defined as part of the output of the algorithm* $\mathsf{Setup}^\star$ *(see the two-step decryption property above).*

The proof of correctness for the scheme described in Fig. 4 can be found in [1].

3.2 Security

The security of the new construction is captured by two different theorems in [1]. The first one considers security under static corruption and the other one covers adaptive corruption. We recap these theorems here.

Theorem 1 (sta-pos$^+$-IND-security) *If the FE scheme* $\mathsf{FE} = (\mathsf{Setup}_{\mathsf{ipfe}}, \mathsf{KeyGen}_{\mathsf{ipfe}}, \mathsf{KeyDer}_{\mathsf{ipfe}}, \mathsf{Enc}_{\mathsf{ipfe}}, \mathsf{Dec}_{\mathsf{ipfe}})$ *is an* any*-IND-secure FE scheme for the inner product functionality defined as* $\mathcal{F}^{\mathsf{ip}}_{\rho_{\mathsf{ipfe}}}, \rho_{\mathsf{ipfe}} = (\mathbb{Z}, 1, m, 2X, Y)$*, and* PRF *is secure, then* MCFE *from Figure 4 is* sta-pos$^+$*-IND-secure for the functionality defined as* $\mathcal{F}^{\mathsf{ip}}_{\rho}, \rho = (\mathbb{Z}, n, m, X, Y)$*. Namely, for any PPT adversary* $\mathcal{A}$*, there exist PPT adversaries* $\mathcal{B}$ *and* $\mathcal{B}'$ *such that:*

$$\mathsf{Adv}^{\text{sta-pos}^+\text{-IND}}_{\mathsf{MCFE},\mathcal{A}}(\lambda, n) \leq 2q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\text{any-IND}}_{\mathsf{FE},\mathcal{B}}(\lambda) + 2(n-1)q_{\mathsf{Enc}} \cdot \mathsf{Adv}_{\mathsf{PRF},\mathcal{B}'}(\lambda),$$

where q_{Enc} *denotes the number of distinct labels queried to* $\mathsf{QLeftRight}$*.*

Theorem 2 (adt-pos$^+$-IND-security) *If the FE scheme* $\mathsf{FE} = (\mathsf{Setup}_{\mathsf{ipfe}}, \mathsf{KeyGen}_{\mathsf{ipfe}}, \mathsf{KeyDer}_{\mathsf{ipfe}}, \mathsf{Enc}_{\mathsf{ipfe}}, \mathsf{Dec}_{\mathsf{ipfe}})$ *is an* any*-IND-secure FE scheme for the inner product functionality defined as* $\mathcal{F}^{\mathsf{ip}}_{\rho_{\mathsf{ipfe}}}, \rho_{\mathsf{ipfe}} = (\mathbb{Z}, 1, m, 2X, Y)$*, and* PRF *is secure, then* MCFE *from Figure 4 is* adt-pos$^+$*-IND-secure for the functionality defined as* $\mathcal{F}^{\mathsf{ip}}_{\rho}, \rho = (\mathbb{Z}, n, m, X, Y)$*. Namely, for any PPT adversary* $\mathcal{A}$*, there exist PPT adversaries* $\mathcal{B}$ *and* $\mathcal{B}'$ *such that:*

$$\mathsf{Adv}^{\text{adt-pos}^+\text{-IND}}_{\mathsf{MCFE},\mathcal{A}}(\lambda, n) \leq 2(n+1)n(n-1)^2 q_{\mathsf{Enc}} \cdot \mathsf{Adv}_{\mathsf{PRF},\mathcal{B}}(\lambda) + 2(n+1)q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\text{any-IND}}_{\mathsf{FE},\mathcal{B}'}(\lambda) \ ,$$

where q_{Enc} *denotes the number of distinct labels queried to* $\mathsf{QLeftRight}$*.*

The proof for both of the theorems can be found in [1].

Using the generic transformation presented in Section 4, it is possible to remove the pos$^+$ restriction, and obtain adt-any-IND security.

4 Compiler from pos$^+$-IND to any-IND Security

As discussed in [2], the standard security of MCFE schemes guarantees that an adversary can only learn a function of the inputs when it is in possession of a ciphertext for every input position i. This property, however, is not satisfied by the schemes in [6, 4, 9]. More precisely, their basic definitions guarantee security *only* when the adversary queries every position at least once. As stated in Section 2.2, a scheme satisfying this property is called *pos-IND* secure (for positive) while the standard property is called *any-IND* secure.

In this section, we recall the compiler that generically transforms any adt-pos$^+$-IND secure (D)MCFE into an adt-any-IND secure (D)MCFE as presented in [1]. As noted in Section 2.2, The pos$^+$-IND security notion is slightly stronger than the pos-IND notion in [2].

4.1 Construction

The new construction by Abdalla et al. [1] builds upon the compiler from [2, Section 4.1], which has also been presented in Deliverable 4.2. The scheme in the standard model, presented in [2, Section 4.1] and Deliverable 4.2, does not support labels. The new construction, on the other hand, can handle multiple labels, many challenge ciphertexts per label and input slots, and adaptive corruptions, without resorting to the random oracle model, as opposed to [2, Section 4.2], which has also been presented in Deliverable 4.2. This is the first generic transformation to support such features. Moreover, when combined with the MCFE from Section 3, the new construction yields the first label-based MCFE for inner products whose adt − any-IND security is proven in the standard model.

$\underline{\mathsf{Setup}'(1^\lambda, 1^n):}$
Return $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$

$\underline{\mathsf{KeyGen}'(\mathsf{pp}):}$
$\{\mathsf{sk}_i\}_{i\in[n]} \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
For $i, j \in [n]$: $\mathsf{k}_{i,j} \leftarrow \{0,1\}^\lambda$
Return $\{\mathsf{sk}'_i = (\mathsf{sk}_i, \{\mathsf{k}_{i,j}, \mathsf{k}_{j,i}\}_{j\in[n]})\}_{i\in[n]}$

$\underline{\mathsf{Enc}'(\mathsf{pp}, \mathsf{sk}'_i, x_i, \ell):}$
Parse $\mathsf{sk}'_i = (\mathsf{sk}_i, \{\mathsf{k}_{i,j}, \mathsf{k}_{j,i}\}_{j\in[n]})$
$\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{sk}_i, x_i)$
For all $j \in [n]$: $\mathsf{k}_{i,j}(\ell) := \mathsf{PRF}_{\mathsf{k}_{i,j}}(\ell)$
$\mathsf{K}_i(\ell) := \oplus_{j\in[n]}\mathsf{k}_{i,j}(\ell)$
$\mathsf{ct}'_i \leftarrow \mathsf{Enc}_{\mathsf{SE}}(\mathsf{K}_i(\ell), \mathsf{ct}_i)$
Return $(\mathsf{ct}'_i, \{\mathsf{k}_{j,i}(\ell)\}_{j\in[n]})$

$\underline{\mathsf{KeyDerShare}'(\mathsf{pp}, \mathsf{sk}'_i, f):}$
Parse $\mathsf{sk}'_i = (\mathsf{sk}_i, \{\mathsf{k}_{i,j}, \mathsf{k}_{j,i}\}_{j\in[n]})$
Return $\mathsf{sk}'_{i,f} \leftarrow \mathsf{KeyDerShare}(\mathsf{sk}_i, f)$

$\underline{\mathsf{KeyDerComb}'(\mathsf{pp}, \{\mathsf{sk}'_{i,f}\}_{i\in[n]}):}$
$\mathsf{sk}_f := \mathsf{KeyDerComb}(\mathsf{pp}, \{\mathsf{sk}'_{i,f}\}_{i\in[n]})$
Return sk_f

$\underline{\mathsf{Dec}'(\mathsf{pp}, \mathsf{sk}_f, \mathsf{ct}''_1, \ldots, \mathsf{ct}''_n):}$
Parse $\{\mathsf{ct}''_i = (\mathsf{ct}'_i, \{\mathsf{k}_{j,i}(\ell)\}_{j\in[n]})\}_{i\in[n]}$
For $i \in [n]$:
  $\mathsf{K}_i(\ell) = \oplus_{j\in[n]}\mathsf{k}_{i,j}(\ell)$
  $\mathsf{ct}_i \leftarrow \mathsf{Dec}_{\mathsf{SE}}(\mathsf{K}_i(\ell), \mathsf{ct}'_i)$
Return $\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$.

Figure 5: Compiler from an xx-pos$^+$-IND secure DMCFE into an xx-any-IND secure DMCFE.

The new construction given in Fig. 5 is stated in terms of DMCFE as in [1]. A similar transformation, however, works for MCFE schemes. Like the standard-model constructions in [2, Section 4.1] and [6], the new compiler uses a symmetric encryption scheme on top on the underlying pos$^+$-

IND MIFE scheme. For each slot i and label ℓ, it runs the encryption scheme of the (D)MCFE scheme to obtain a ciphertext $\mathsf{ct}_{i,\ell}$ and encrypts $\mathsf{ct}_{i,\ell}$ using a symmetric encryption scheme with key $\mathsf{K}_i(\ell)$. The latter is computed distributedly using keys $\mathsf{k}_{i,j}$ shared between slots i and j with the help of a PRF.

4.2 Security

As the following theorem shows, the resulting (D)MCFE scheme is xx-pos$^+$-IND secure as long as the underlying (D)MCFE scheme is xx-any-IND secure and PRF is a pseudorandom function. The proof of this theorem can be found in [1].

Theorem 3 (Security) *Let the tuple* $\mathsf{DMCFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{KeyDerShare}, \mathsf{KeyDerComb}, \mathsf{Enc}, \mathsf{Dec})$ *be an* adt-pos$^+$*-IND-secure DMCFE scheme for a family of functions* $\mathcal{F}$*. Let* $\mathsf{SE} = (\mathsf{Enc}_{\mathsf{SE}}, \mathsf{Dec}_{\mathsf{SE}})$ *be an IND-CPA symmetric-key encryption scheme. Let* PRF *be a pseudorandom function. Then the DMCFE scheme* $\mathsf{DMCFE}' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{KeyDerShare}', \mathsf{KeyDerComb}', \mathsf{Enc}', \mathsf{Dec}')$ *described in Figure 5 is* adt-any*-IND secure. Namely, for any PPT adversary* $\mathcal{A}$*, there exist PPT adversaries* $\mathcal{B}$*,* $\mathcal{B}'$*, and* $\mathcal{B}''$ *such that:*

$$\mathsf{Adv}^{\text{adt-any-IND}}_{\mathsf{DMCFE}',\mathcal{A}}(\lambda, n) \leq q_{\mathsf{Enc}} \cdot \mathsf{Adv}^{\text{adt-pos}^+\text{-IND}}_{\mathsf{DMCFE},\mathcal{B}}(\lambda, n) + q_{\mathsf{Enc}} n^2 \cdot \mathsf{Adv}^{\text{IND-CPA}}_{\mathsf{SE},\mathcal{B}'}(\lambda) + 2 q_{\mathsf{Enc}} n^2 \cdot \mathsf{Adv}_{\mathsf{PRF},\mathcal{B}''}(\lambda),$$

where q_{Enc} *denotes the number of distinct labels queried to* QLeftRight.

5 Conclusion

In this document, we described one possible way of limiting the inherent leakage of functional encryption schemes via the introduction of labels during the encryption process of multi-client functional encryption schemes. In particular, the new multi-client functional encryption construction for the inner-product functionality described in Section 3 is generic, can handle encryption labels, and can be instantiated under several different computational assumptions. Moreover, together with the compiler in Section 4, the resulting scheme can achieve security even when the adversary does not query every encryption slot at least once.

While the solution above provides one way of limiting the mix-and-match of ciphertexts in the multi-input setting, it does not solve the problem in the single-input setting. To address the latter setting, one alternative would be to consider FE primitives with more sophisticated functionalities. For instance, this primitive could embed access policies in the (encrypted) data while allowing to compute weighted sums on the latter. In fact, in an ongoing work being developed in the context of the FENTEC project [5], we formalize the notion of inner-product functional encryption with fine-grained access control and propose realizations that are both efficient and provably secure under standard and well established assumptions.

References

[1] Michel Abdalla, Fabrice Benhamouda, and Romain Gay. From single-input to multi-client inner-product functional encryption. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part III*, volume 11923 of *Lecture Notes in Computer Science*, pages 552–582, Kobe, Japan, December 8–12, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-34618-8_19. (Pages v, 1, 4, 5, 8, 10, 11, and 12.)

[2] Michel Abdalla, Fabrice Benhamouda, Markulf Kohlweiss, and Hendrik Waldner. Decentralizing inner-product functional encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019: 22nd International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 11443 of *Lecture Notes in Computer Science*, pages 128–157, Beijing, China, April 14–17, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-17259-6_5. (Pages 3, 4, 5, 8, and 11.)

[3] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 733–751, Gaithersburg, MD, USA, March 30 – April 1, 2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-46447-2_33. (Page 8.)

[4] Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 597–627, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-96884-1_20. (Pages 8, 10, and 11.)

[5] Michel Abdalla, Dario Catalano, Romain Gay, and Bogdan Ursu. Inner-product functional encryption with fine-grained access control. Manuscript, December 2019. (Page 13.)

[6] Michel Abdalla, Romain Gay, Mariana Raykova, and Hoeteck Wee. Multi-input inner-product functional encryption from pairings. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 601–626, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-56620-7_21. (Page 11.)

[7] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 333–362, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-53015-3_12. (Page 8.)

[8] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-19571-6_16. (Pages 1 and 4.)

[9] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. Decentralized multi-client functional encryption for inner product. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part II*, volume 11273 of *Lecture Notes in Computer Science*, pages 703–732, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-03329-3_24. (Pages 5, 6, and 11.)

[10] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 578–602, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-55220-5_32. (Page 3.)

[11] Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. http://eprint.iacr.org/2010/556. (Pages 1 and 4.)