Multi-Client Functional Encryption for Small Domains

Suvasree Biswas
suvasree@gwu.edu
The George Washington University
Washington DC, USA

Mohit Vaid
m.vaid@gwu.edu
The George Washington University
Washington DC, USA

Arkady Yerukhimovich
arkady@gwu.edu
The George Washington University
Washington DC, USA

Abstract

In this paper, we revisit the problem of multi-client functional encryption (MCFE) for general functions. Specifically, we consider the setting of private-key MCFE for functions where the input domain is polynomial in the security parameter. Surprisingly, we show that in this setting it is possible to construct a private-key MCFE scheme secure for a bounded number of key and encryption queries based only on the minimal assumption that one-way functions exist. In contrast, all prior constructions of MCFE for general functions require very strong assumptions such as indistinguishability obfuscation or multi-linear maps.

Our main technique is to show that private-key MCFE for polynomial input domain can be built from any private-key multi-input functional encryption (MIFE) while inheriting the security properties of the underlying MIFE. Instantiating our construction with the MIFE of Brakerski et al. (Eurocrypt 2016) gives us a construction based only on the existence of one-way functions.

CCS Concepts

• **Theory of computation → Cryptographic primitives**.

Keywords

Functional Encryption, Multi-Client Functional Encryption, Multi-Input Functional Encryption

1 Introduction

Originating from a sequence of foundational works [12, 13, 26, 27, 30, 36, 37], functional encryption goes beyond the conventional "all-or-nothing" security guarantee of classical encryption. Specifically, functional encryption allows the issuance of a functional key corresponding to a function f that allows evaluation of the function f on encrypted data, while hiding everything else. Over the past decade, substantial effort has been devoted to studying both the security guarantees and the efficiency of functional encryption schemes [5, 9, 12, 14, 19, 20, 22, 23, 36].

While traditional, single-input, functional encryption has many applications it has a critical limitation—the entire input must be encrypted at one time into a single ciphertext. However, it is often desirable to evaluate functions over multiple ciphertexts for example if multiple pieces of data are encrypted over a long period of time, or if the encryptions are done by multiple different parties. To address this limitation, Goldwasser et al. [21] formalized two natural extensions: multi-input functional encryption (MIFE) and multi-client functional encryption (MCFE). MIFE allows the evaluation of multi-input functions over a set of ciphertexts, but all of these must be encrypted with the same encryption key, which must be kept secure. On the other hand, MCFE allows the input ciphertexts to be generated using different encryption keys, even allowing some of these keys to be controlled by an adversary. In both of these, the goal is to guarantee that nothing beyond the value of f on the encrypted inputs is revealed.

The early constructions of MCFE and MIFE for general functionalities, initiated by Goldwasser et al. [21], relied on the heavy cryptographic machinery of indistinguishability obfuscation [8, 20, 29] or multilinear maps [11]. Moreover, [7, 10, 21, 28] showed that these assumptions are, in fact, *necessary* as unbounded variants of multi-input functional encryption imply obfuscation. Thus, a fundamental question arises whether there is any version of MIFE and MCFE for general functions that can be realized from more standard, lighter assumptions.

Surprisingly, Brakerski et al. [15] showed that for the case of private-key MIFE (i.e. a variant of MIFE where the master secret key is needed to encrypt) one can construct such a scheme for general functions that is secure against an adversary making a bounded number of key and encryption queries based only on the minimal assumption of one-way functions. However, this result did not give a construction of MCFE and to date the only constructions of MCFE that do not rely on obfuscation are for restricted function classes such as inner products [1, 31–35, 38] and separable functions [18].

In this paper, we revisit the question of constructing MCFE from weaker assumptions and ask:

> Is there a variant of MCFE for general functions that can be instantiated using only standard assumptions?

1.1 Our Contribution

We answer the above question in the affirmative. Our work departs from the prevailing paradigm for constructing MCFE schemes: Instead of limiting the class of admissible functions, we demonstrate that one can retain general functionality by restricting the input domain.

Specifically, we focus on the case of private-key MCFE for all functions computable by polynomial size circuits. In this setting, we show how to construct an MCFE scheme with polynomial size input domain that is secure against an adversary making a bounded number of key and encryption queries from *any* MIFE scheme. In particular, our construction only needs the MIFE scheme to be secure against an adversary making the same number of key-generation queries and polynomially more encryption queries than the resulting MCFE scheme. Specifically, we prove the following theorem:

Theorem 1.1 (Informal). *A private-key* MIFE *for general functions that is secure against an adversary making a bounded number of encryption and key-generation queries implies a private-key* MCFE *for general functions over a polynomial input domain that is secure against a similarly bounded adversary.*

Instantiating the MIFE in Theorem 1.1 with the private-key MIFE construction by Brakerski et al. [15], we get the following corollary.

Corollary 1 (Informal). *Assuming the existence of one-way functions, there exists a private-key* MCFE *scheme for general functions of constant-arity over polynomial size domain that is secure against an adversary making a bounded number of key-generation and encryption queries.*

1.2 Our Approach

Observe that both MCFE and MIFE have essentially the same evaluation functionality. Given a functional key, and a set of ciphertexts, this procedure evaluates the function on the encrypted messages. Thus, an immediate idea is to somehow allow the MCFE clients to produce MIFE ciphertexts for their messages so that the MIFE evaluation procedure can be used. The main challenge in achieving this is that in MIFE all encryptions are generated under the same (secret) key, while in MCFE every client must receive its own encryption key, and moreover the scheme must remain secure even when the adversary knows some of the encryption keys. It is thus not clear how one can use the MCFE encryption keys to generate ciphertexts under one MIFE key.

Our construction bypasses this challenge by leveraging the fact that the input domain is of polynomial size. Specifically, for a polynomial size domain, the trusted key authority can pre-encrypt MIFE ciphertexts for every possible message at every input slot. We then assign each client in MCFE to one of these input slots and give the corresponding ciphertexts to each client as their encryption key. With this key, a client can encrypt a message m by just choosing the mth ciphertext from its key. This allows the clients to generate MIFE ciphertexts for their messages without needing any information about the MIFE secret key. Evaluation can then just use the MIFE key generation and evaluation procedure.

However, we are not quite done. The problem with the above construction is that the MCFE encryption procedure is deterministic, so each client can only encrypt each message exactly once. To deal with this limitation, we turn to the notion of labeled MCFE. In labeled MCFE each ciphertext additionally has a label and evaluation can only be done on ciphertexts with the same label thus preventing mix-and-match attacks where an adversary evaluates the function on all possible combinations of encrypted messages. To enable labels in our construction, we simply pre-generate cipheretexts for all message, label pairs–this is still polynomial if the number of possible labels is also small. Now, each client can encrypt a message for each label allowing a bounded number of encryptions. To ensure that labels are used correctly in evaluation, we modify the function in key generation queries to first check that all labels are equal before producing any output. The resulting MCFE scheme supports a bounded number of encryption and key-generation queries and inherits its security guarantees from the underlying MIFE construction.

Limitations. Our construction does have one critical limitation. Like other labeled MCFE schemes, security only holds if clients never reuse a label for more than one encryption. In traditional MCFE [2, 17, 24] this is handled by using a timestamp or a hash output as the label, thus guaranteeing uniqueness. However, since our label space is small and must be fixed at initialization, we cannot use a randomly chosen label. Instead, we need the clients to be stateful to keep track of the labels they have already used, e.g., via a counter, or we need the application to pre-specify how clients can agree on a label from the pre-generated set in a deterministic way. This also implies that an encryption key can only be used a bounded number of times before it needs to be refreshed. We leave the question of how to remove this statefullness requirement as an interesting open question.

2 Applications

In this section we briefly discuss some potential application of our MCFE construction. We specifically focus on applications where small input domain and the bounded encryptions are a natural fit.

Electronic Voting : One natural application of our construction is for electronic voting where the final winner can be determined via MCFE evaluation of the client's votes. In this setting, the input domain i.e. the set of candidates is inherently small and fixed (For future candidates, we can just use a number which will match the order in which candidates appear in the poll). Similarly, future election dates are predetermined and can naturally serve as labels in our framework, and it is easy for the clients to know the correct label at voting time.

Surveys: Surveys routinely ask respondents to choose from a small set of options or provide a ranking (e.g., from 1 to 10) for some item or activity. Our MCFE allows small-scale surveys to be carried out while maintaining the privacy of individual responses. In particular, our MCFE would enable evaluation of any statistics on the survey responses. Here the label for each survey can be the number of the survey and surveys naturally have a small domain.

3 Preliminaries

We summarize the notation and definitions used in this work.

Notation We let $\kappa \in \mathbb{N}$, denote the security parameter. A function $\nu : \mathbb{N} \to \mathbb{N}$ is negligible if $\nu(\kappa) < \frac{1}{\kappa^c}$ for all $c > 0$ and sufficiently large κ. We use $[n] := \{1, \ldots, n\}$. We let t be the total number of clients, whereas $\mathcal{CP}$ and $\mathcal{H}$ denote the sets of corrupted and honest clients respectively. For a fixed constant $t > 1$, we let $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathbb{N}}$ represent the set of all functions computable by polynomial size circuits where every $f \in \mathcal{F}_\kappa$ is of the form $f : \mathcal{X}_\kappa^t \to \mathcal{Y}_\kappa$ where $\mathcal{X}_\kappa^t$, $\mathcal{Y}_\kappa$ are the input space and output space respectively and $|\mathcal{X}_\kappa| < \text{poly}(\kappa)$.

3.1 Private Key Multi-Input Functional Encryption [7, 16]

For any $\kappa, t \in \mathbb{N}$, let $\mathcal{X} = \{\mathcal{X}_\kappa\}_{\kappa\in\mathbb{N}}$ be the plaintext space, $\mathcal{Y} = \{\mathcal{Y}_\kappa\}_{\kappa\in\mathbb{N}}$ be the output space and $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathbb{N}}$ be the collection of all functions computable by polynomial size circuits where every f is of the form $f : \mathcal{X}_\kappa^t \to \mathcal{Y}_\kappa$. A private key t-input MIFE for $\mathcal{F}$ consists of quadruple of probabilistic polytime algorithms described as follows.

- $\mathsf{Setup}(1^\kappa)$: On input the unary encoding of the security parameter κ, it outputs a master secret key msk.
- $\mathsf{Enc}(\mathsf{msk}, x_l, l)$: On input the master secret key msk, a message $x_l \in \mathcal{X}_\kappa$ for some slot $l \in t$, the encryption algorithm outputs a ciphertext ct_{x_l}.

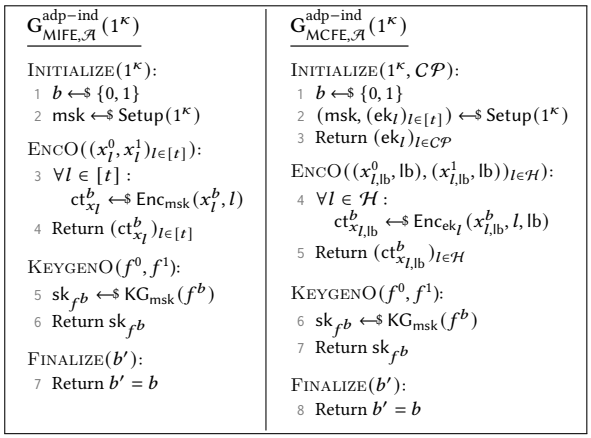


Figure 1: Games defining Adaptive Security of MIFE **(Left) and** MCFE **(Right)**

- KG(msk, f): On input a master secret key msk and a function $f \in \mathcal{F}_\kappa$, it outputs a functional key sk_f.
- Eval(sk_f, $(\text{ct}_{x_l})_{l\in[t]}$): On input a functional key sk_f and t ciphertexts $(\text{ct}_{x_l})_{l\in[t]}$ outputs a string $y \in \mathcal{Y}_\kappa \cup \{\perp\}$.

Definition 3.1 (Correctness). A private key t-input multi-input functional encryption scheme MIFE = (Setup, KG, Enc, Eval) for $\mathcal{F}_\kappa$ is correct if there exists a negligible function $\nu(\cdot)$ such that for every $\kappa \in \mathbb{N}$, for every $f \in \mathcal{F}_\kappa$, and for every $(x_l)_{l\in[t]} \in \mathcal{X}_\kappa^t$, it holds that

$$\Pr[\text{Eval}(\text{sk}_f, (\text{Enc}(\text{msk}, x_l, l))_{l\in[t]}) = f((x_l)_{l\in[t]})] \geq 1 - \nu(\kappa)$$

where $\text{msk} \leftarrow\$ \text{Setup}(1^\kappa)$, $\text{sk}_f \leftarrow\$ \text{KG}(\text{msk}, f)$, and the probability is taken over the internal random coins of Setup, Enc and KG.

In Figure 1 (Left), we present the adaptive security game for MIFE [16, 21]. A stateful ppt adversary $\mathcal{A}$ interacts with a challenger that, upon receiving 1^κ, generates a master secret key and samples a hidden bit b. $\mathcal{A}$ is then allowed to to make encryption queries (to ENCO) and key-generation queries (to KEYGENO) receiving ciphertexts and function keys respectively. Finally, $\mathcal{A}$ outputs a guess bit b' and wins if $b' = b$. In what follows, we only consider a bounded adversary who makes an apriori-bounded number of oracle queries. We now formalize a valid bounded-MIFE adversary.

Definition 3.2 (Valid (q_1, q_2)-bounded adversary). A probabilistic polynomial-time algorithm $\mathcal{A}$ is a *valid (q_1, q_2)-bounded adversary* if there exist bounds $q_1, q_2 \in \mathbb{N}$ such that the following holds. For every private-key t-input multi-input functional encryption scheme MIFE = (Setup, KG, Enc, Eval) over input space $\mathcal{X} = \{\mathcal{X}_\kappa^t\}$ and function space $\mathcal{F} = \{\mathcal{F}_\kappa\}$, the adversary makes at most q_1 key-generation queries of form $(f^0, f^1) \in \mathcal{F}_\kappa$ and at most q_2 encryption oracle queries of form $\{(x_l^0, x_l^1)_{l\in[t]}\} \in (\mathcal{X}_\kappa \times \mathcal{X}_\kappa)^t$, then for all pairs (x_l^0, x_l^1) queried in an encryption query[1] and for all function pairs (f^0, f^1) queried to key generation, it holds that

$$f^0\big((x_l^0)_{l\in[t]}\big) \;=\; f^1\big((x_l^1)_{l\in[t]}\big)\,.$$

We can now provide a formal security definition for adaptively-secure MIFE against a valid (q_1, q_2)-bounded adversary.

[1] including all mix-and-match combinations of messages across queries

Definition 3.3 (adaptive security). For all sufficiently large $\kappa \in \mathbb{N}$, a private-key t-input multi-input functional encryption scheme MIFE = (Setup, KG, Enc, Eval) over a message space $\mathcal{X} = \{\mathcal{X}_\kappa\}_{\kappa\in\mathbb{N}}$ and a function space $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathbb{N}}$ is (q_1, q_2)-*adaptively secure* if for every *valid* (q_1, q_2)-*bounded adversary* $\mathcal{A}$ there exists a negligible function $\nu(\cdot)$ such that for $\mathbf{G}^{\text{adp-ind}}_{\text{MIFE},\mathcal{A}}$ as defined in Figure 1(Left) the following holds

$$\mathbf{Adv}^{\text{adp-ind}}_{\text{MIFE},\mathcal{A}}(1^\kappa) = \left|\Pr[\mathbf{G}^{\text{adp-ind}}_{\text{MIFE},\mathcal{A}}(1^\kappa) \Rightarrow 1] - \frac{1}{2}\right| \leq \nu(\kappa)$$

3.2 Private Key Multi-Client Functional Encryption [38]

For any $\kappa, t \in \mathbb{N}$, let plaintext space $\mathcal{X} = \{\mathcal{X}_\kappa\}_{\kappa\in\mathbb{N}}$, label space $\mathcal{L} = \{\mathcal{L}_\kappa\}_{\kappa\in\mathbb{N}}$, output space $\mathcal{Y} = \{\mathcal{Y}_\kappa\}_{\kappa\in\mathbb{N}}$ and $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathbb{N}}$ be the collection of all functions computable by polynomial size circuits where every $f \in \mathcal{F}_\kappa$ is of the form $f : \mathcal{X}_\kappa^t \to \mathcal{Y}_\kappa$. A private-key t-input multi-client functional encryption scheme MCFE for $\mathcal{F}$ consists of four ppt algorithms described as follows.

- Setup(1^κ): On input the unary encoding of the security parameter κ, it outputs master secret key msk and t encryption keys $(\text{ek}_l)_{l\in[t]}$.
- Enc(ek_l, x_l, l, lb): On input an encryption key ek_l for some slot $l \in [t]$, a message $x_l \in \mathcal{X}_\kappa$, a label $\text{lb} \in \mathcal{L}_\kappa$, it outputs a ciphertext $\text{ct}_{x_l,\text{lb}}$.
- KG(msk, f): On input a master secret key msk and a function $f \in \mathcal{F}_\kappa$, it outputs a functional key sk_f.
- Eval(sk_f, $(\text{ct}_{x_l,\text{lb}})_{l\in[t]}$): On input a functional key sk_f and t ciphertexts $(\text{ct}_{x_l,\text{lb}})_{l\in[t]}$ it outputs a string $y \in \mathcal{Y}_\kappa \cup \{\perp\}$.

Definition 3.4 (Correctness). A private key t-input multi-client functional encryption scheme MCFE for $\mathcal{F}$ is correct if there exists a negligible function $\nu(\cdot)$ such that for every $\kappa \in \mathbb{N}$, for every $f \in \mathcal{F}_\kappa$, and for every $(x_l)_{l\in[t]} \in \mathcal{X}_\kappa^t$, lb $\in \mathcal{L}_\kappa$, it holds that

$$\Pr[\text{Eval}(\text{sk}_f, (\text{Enc}(\text{ek}_l, (x_l, l, \text{lb}))_{l\in[t]}) = f((x_l)_{l\in[t]})] \geq 1 - \nu(\kappa)$$

where $\text{msk}, (\text{ek}_l)_{l\in[t]} \leftarrow\$ \text{Setup}(1^\kappa)$, $\text{sk}_f \leftarrow\$ \text{KG}(\text{msk}, f)$, and the probability is taken over the internal random coins of Setup, Enc and KG.

In Figure 1(Right), we define adaptive security for MCFE [21, 25, 38]. A stateful ppt adversary $\mathcal{A}$ declares a corrupted set $\mathcal{CP}$, receives encryption keys for all parties in set $\mathcal{CP}$, and may query ENCO queries to get encryptions of messages from honest parties, and KEYGENO to obtain function keys. Finally, $\mathcal{A}$ outputs a guess b' and wins if $b' = b$. As we did for MIFE, we focus on the case of an adversary making a bounded number of oracle queries. We formalize such a valid bounded adversary below.

Definition 3.5 (Valid (q_1, q_2)-bounded adversary). A probabilistic polynomial-time algorithm $\mathcal{A}$ is a *valid (q_1, q_2)-bounded adversary* if there exist bounds $q_1, q_2 \in \mathbb{N}$ such that the following holds. For every private-key t-input multi-client functional encryption scheme MCFE = (Setup, KG, Enc, Eval) over input space $\mathcal{X} = \{\mathcal{X}_\kappa^t\}$, label space $\mathcal{L} = \{\mathcal{L}_\kappa\}$ and function space $\mathcal{F} = \{\mathcal{F}_\kappa\}$, the adversary

- makes at most q_1 key-generation queries of form $(f^0, f^1) \in \mathcal{F}_\kappa$; and
- makes at most q_2 encryption queries of the form

$$\big((x^0_{l,\text{lb}}, \text{lb}),\ (x^1_{l,\text{lb}}, \text{lb})\big)_{l\in\mathcal{H}}$$

specifying, for label lb and every honest party $l \in \mathcal{H}$, a pair of challenge inputs $(x^0_{l,\mathsf{lb}}, x^1_{l,\mathsf{lb}})_{l\in\mathcal{H}} \in (\mathcal{X}_\kappa \times \mathcal{X}_\kappa)^{\mathcal{H}}$.

and satisfies the following conditions:

(1) For every encryption query involving a corrupt party $l \in \mathcal{CP}$, the submitted challenge inputs satisfy

$$x^0_{l,\mathsf{lb}} = x^1_{l,\mathsf{lb}}.$$

(2) For every key-generation query (f^0, f^1) and ever pair $(x^0_{l,\mathsf{lb}}, x^1_{l,\mathsf{lb}})_{l\in\mathcal{H}}$ with the same label lb submitted in an encryption query, the following holds

$$f^0\big((x^0_{l,\mathsf{lb}})_{l\in\mathcal{H}},\ \cdot\big) = f^1\big((x^1_{l,\mathsf{lb}})_{l\in\mathcal{H}},\ \cdot\big),$$

where the dot $(\cdot)$ denotes all possible adversarial inputs.

We now present the formal security definition for adaptively-secure MCFE against a valid (q_1, q_2)-bounded adversary.

Definition 3.6 (adaptive security). For all sufficiently large $\kappa \in \mathbb{N}$, a private-key t-input multi-client functional encryption scheme $\mathsf{MCFE} = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Eval})$ over input space $\mathcal{X} = \{\mathcal{X}_\kappa\}_{\kappa\in\mathbb{N}}$, label space $\mathcal{L} = \{\mathcal{L}_\kappa\}$ and a function space $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathbb{N}}$ is (q_1, q_2)-*adaptively secure* if for every *valid* (q_1, q_2)*-bounded adversary* $\mathcal{A}$ there exists a negligible function $\nu(\cdot)$ such that for $\mathbf{G}^{\mathrm{adp-ind}}_{\mathsf{MCFE},\mathcal{A}}$ as in Figure 1(Right) the following holds

$$\mathbf{Adv}^{\mathrm{adp-ind}}_{\mathsf{MCFE},\mathcal{A}}(1^\kappa) = \left|\Pr[\mathbf{G}^{\mathrm{adp-ind}}_{\mathsf{MCFE},\mathcal{A}}(1^\kappa) \Rightarrow 1] - \frac{1}{2}\right| \leq \nu(\kappa)$$

4 Construction

In this section, we present a construction of an adaptively secure private-key multi-client functional encryption scheme (MCFE) for functions over polynomial-size domains with bounded key-generation and encryption oracle queries, obtained by using, as a black-box, a private-key multi-input functional encryption scheme (MIFE) for functions that also supports bounded key-generation and encryption oracle queries. For a security parameter κ and a constant $t > 1$ representing the number of parties, let the MCFE plaintext space be $\mathcal{X} = \{\mathcal{X}_\kappa\}_{\kappa\in\mathbb{N}}$, the label space be $\mathcal{L} = \{\mathcal{L}_\kappa\}_{\kappa\in\mathbb{N}}$, and the output space be $\mathcal{Y} = \{\mathcal{Y}_\kappa\}_{\kappa\in\mathbb{N}}$, where $|\mathcal{X}_\kappa| < \mathrm{poly}(\kappa)$ and $|\mathcal{L}_\kappa| < \mathrm{poly}(\kappa)$. The MCFE and the underlying MIFE differ in how the label lb is carried: a MCFE plaintext is a pair (m, lb) in which lb occupies a dedicated coordinate, whereas the MIFE has no separate coordinate for the label and instead folds it into the message as a single string $m \,\|\, \mathsf{lb}$. Accordingly, the MIFE message space is $\mathcal{X}' = \{\mathcal{X}'_\kappa\}_{\kappa\in\mathbb{N}}$ with $\mathcal{X}'_\kappa = (\mathcal{X}_\kappa \times \mathcal{L}_\kappa) \cup \{\perp\}$, where each message is either an encoding $m \,\|\, \mathsf{lb}$ of a pair $(m, \mathsf{lb}) \in \mathcal{X}_\kappa \times \mathcal{L}_\kappa$ or a distinguished symbol $\perp \notin \mathcal{X}_\kappa \times \mathcal{L}_\kappa$ (used to denote an aborting or invalid input). Let $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathbb{N}}$ be the set of all functions computable by polynomial-size circuits, where every $f \in \mathcal{F}_\kappa$ is of the form $f : \mathcal{X}^t_\kappa \to \mathcal{Y}_\kappa$. From each f we derive a related MIFE functionality $\mathsf{D}_f : (\mathcal{X}'_\kappa)^t \to \mathcal{Y}_\kappa \cup \{\perp\}$ analogous to f, obtained by augmenting f with a label-equality check and a non-$\perp$ check on its inputs (described in detail in MCFE.KG); we write $\mathcal{D} = \{\mathcal{D}_\kappa\}_{\kappa\in\mathbb{N}}$ with $\mathcal{D}_\kappa = \{\mathsf{D}_f : f \in \mathcal{F}_\kappa\}$ for the resulting MIFE function family. The underlying MIFE is instantiated for the function family $\mathcal{D}$. We provide below the construction of a private-key MCFE scheme for $\mathcal{F}$.

4.1 Our Construction

Our approach, as outlined in Section 1.1, supports polynomial-size message and label spaces. Using the MIFE scheme, we start by generating the MIFE master secret key msk and use it to pre-compute ciphertexts for all (plaintext, label) pairs using the MIFE encryption algorithm, we follow this approach for each slot $l \in [t]$. The generated table of ciphertexts for a slot $l \in [t]$ is then given to the corresponding party as the encryption key $\mathsf{MCFE.ek}_l$ for the MCFE scheme. To encrypt, each party $l \in [t]$, comes up with a message x from the message space $\mathcal{X}_\kappa$ and a label lb from label space $\mathcal{L}_\kappa$ and uses the $\mathsf{MCFE.ek}_l$ to find the ciphertext corresponding to the tuple (x, lb) to get the corresponding MIFE ciphertext.

For any t-ary function $f \in \mathcal{F}_\kappa$ computable by a $\mathrm{poly}(\kappa)$-size circuit, the MCFE function key is simply the corresponding MIFE function key for the modified functionality D_f, which checks that all inputs share the same label and are non-$\perp$, and if so evaluates f on them (aborting otherwise). An evaluator holding the functional key $\mathsf{MCFE.sk}_f$ can then compute the functional evaluation on the underlying plaintexts by calling MCFE.Eval, which in turn runs MIFE.Eval. The label check built into D_f is what circumvents an inherent limitation of MIFE, namely the ability of an evaluator to arbitrarily *mix-and-match* ciphertexts of different labels during evaluation: any inputs with mismatched labels evaluate to $\perp$, so the adversary learns nothing from launching such an attack.

As should be obvious from the above description, security of our scheme requires that each client only uses each label in at most one ciphertext. This requires clients to ensure that they avoid label reuse. Since our label domain is small, we cannot achieve this by simply sampling labels at random. Thus, either the clients need to maintain a counter to indicate labels they have already used, or there needs to be some external mechanism that can assign labels from the pre-determined label space. Moreover, since our pre-determined label space is small, this also means that an encryption key can only be used a bounded number of times before it must be refreshed.

Construction Details Let $\mathsf{MIFE} = (\mathsf{MIFE.Setup}, \mathsf{MIFE.KG}, \mathsf{MIFE.Enc}, \mathsf{MIFE.Eval})$ be a private-key multi-input functional encryption scheme for $\mathcal{D}$. We define MCFE as a quadruple of ppt algorithms $(\mathsf{MCFE.Setup}, \mathsf{MCFE.KG}, \mathsf{MCFE.Enc}, \mathsf{MCFE.Eval})$, each of which is briefly explained below. The formal construction is given in Figure 2.

$\mathsf{MCFE.Setup}(1^\kappa)$. On input the unary encoding of the security parameter κ, the Setup algorithm first invokes the setup procedure of MIFE to create the master secret key MIFE.msk. For each client slot $l \in [t]$, an initially empty ciphertext table is created. Then, for every plaintext $m \in \mathcal{X}_\kappa$ and label $h \in \mathcal{L}_\kappa$, the algorithm samples fresh randomness and applies the MIFE encryption procedure on message $(m||h)$ and slot l to generate the corresponding ciphertext. Each resulting ciphertext is stored in the table at index (m, h). The completed table for slot l serves as the client's encryption key $\mathsf{MCFE.ek}_l$. Finally, the algorithm outputs the MIFE master secret key together with the collection of encryption keys $(\mathsf{MCFE.ek}_l)_{l\in[t]}$ of all clients.

$\mathsf{MCFE.Enc}(\mathsf{MCFE.ek}_l, x, l, \mathsf{lb})$. A client with an encryption key $\mathsf{MCFE.ek}_l$, provides for slot l, an input $x \in \mathcal{X}_\kappa$ and a label $\mathsf{lb} \in \mathcal{L}_\kappa$ to the encryption algorithm which then retrieves the precomputed

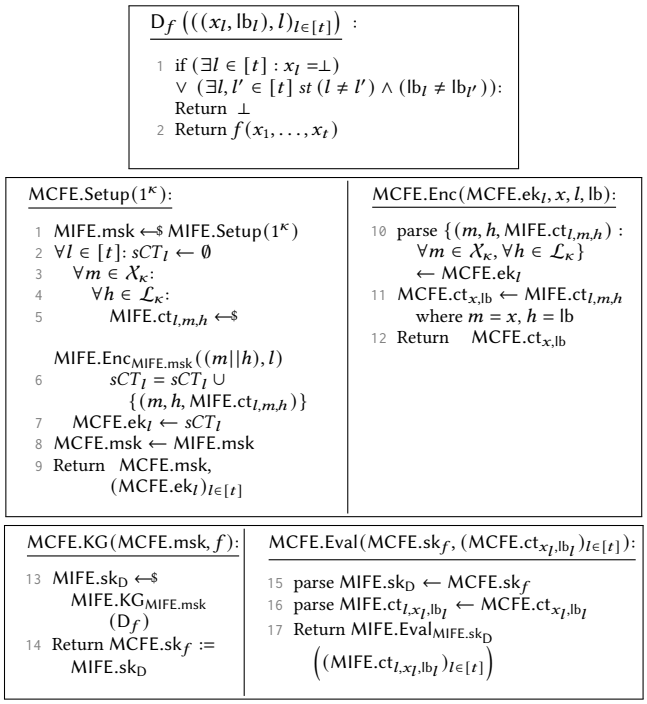


Figure 2: MCFE **Construction**

ciphertext for (x, lb) from the table MCFE.ek$_l$. This ciphertext is returned as the MCFE encryption of x for slot l under label lb.

MCFE.KG(MCFE.msk, f). On input a function $f \in \mathcal{F}_\kappa$, we construct the related MIFE function D_f by augmenting f with two additional checks: (i) that the labels associated with all inputs are identical, and (ii) and $x_i \neq \perp$. Next, key generation uses MCFE.msk = MIFE.msk, to run the MIFE key generation procedure, i.e., it outputs MCFE.sk$_f$ = MIFE.sk$_{\mathsf{D}_f} \leftarrow$ MIFE.KG(MIFE.msk, D_f).

MCFE.Eval(MCFE.sk$_f$, (MCFE.ct$_{x_l,\mathsf{lb}_l})_{l\in[t]}$). The evaluation algorithm takes as input the functional key MCFE.sk$_f$ together with a ciphertext from each client. Recall that these ciphertexts are ciphertexts of the underlying MIFE scheme. Finally, the algorithm runs the MIFE evaluation procedure on these ciphertexts using MCFE.sk$_f$, to compute $\mathsf{D}_f(x_1, \ldots, x_t)$ which outputs either $f(x_1, \ldots, x_t)$ or $\perp$.

Correctness For all valid t messages $x_1, \ldots, x_t$ under some label lb and for all function f from $\mathcal{F}_\kappa$, correctness of MCFE follows from the correctness of MIFE.

Instantiation Instantiating our construction with the adaptively secure MIFE scheme by Brakerski et al.[15] which when further instantiated with the single-input private-key FE scheme of Gorbunov et al.[23], we obtain a private-key MCFE for constrained domain, supporting bounded keygen and bounded encryption query, by just relying on minimal assumption of one way function (OWF).

5 Proof of Security

The following theorem captures the security of our scheme.

Theorem 5.1. *For any integer $t > 1$ and security parameter $\kappa \in \mathbb{N}$, let* MCFE *for $\mathcal{F}$, as given in Figure 2, be defined over input space $\mathcal{X}_\kappa^t$ s.t. $|\mathcal{X}_\kappa| \leq \mathrm{poly}(\kappa)$ and label space $\mathcal{L}_\kappa$ s.t. $|\mathcal{L}_\kappa| \leq \mathrm{poly}(\kappa)$. If* MIFE *is a private-key (q_1', q_2')-bounded adaptively secure multi-input functional encryption scheme for $\mathcal{D}$, as defined in Figure 1(Left). Then* MCFE *is a private-key (q_1, q_2)-bounded adaptively secure multi-client functional encryption scheme for $\mathcal{F}$, as defined in Figure 1(Right) such that*

$$q_1 = q_1' \quad \text{and} \quad q_2 = q_2' - |\mathcal{X}_\kappa| \cdot |\mathcal{L}_\kappa|$$

Proof. Towards proving Theorem 5.1, let $\mathcal{A}$ be a valid (q_1, q_2)-bounded ppt adversary (as per Definition 3.5) against adaptively secure MCFE construction from Figure 2. Suppose $\mathcal{A}$ wins the experiment $\mathbf{G}^{\mathrm{adp-ind}}_{\mathsf{MCFE},\mathcal{A}}$ in Figure 1(Right) with non-negligible probability. Using $\mathcal{A}$, we construct a (q_1', q_2')-bounded ppt adversary $\mathcal{B}$, valid as per Definition 3.2, that breaks the adaptive message security of MIFE in the experiment $\mathbf{G}^{\mathrm{adp-ind}}_{\mathsf{MIFE},\mathcal{B}}$ of Figure 1(Left). Adversary $\mathcal{B}$ interacts with the MIFE challenger while internally simulating the MCFE game for $\mathcal{A}$.

The proof begins with $\mathcal{A}$ submitting the set of corrupted parties, for which $\mathcal{B}$ must provide encryption keys consisting of ciphertexts for all plaintext–label pairs associated with the corrupted parties. To generate these, $\mathcal{B}$ makes EncO queries enumerating all possible inputs for the slots corresponding to the t corrupted parties.

When $\mathcal{A}$ submits EncO queries – recall that in the MCFE security game EncO queries only include message pairs for honest parties – $\mathcal{B}$ embeds the queried message pairs into their corresponding slots, while filling the corrupted slots with dummy values $\perp$, assigning identical messages to both challenge positions. The response from the MIFE challenger is then forwarded to $\mathcal{A}$.

When $\mathcal{A}$ submits KeygenO queries, adversary $\mathcal{B}$ translates the queried function f to the corresponding function D_f (See Figure 2) and queries it to his KeygenO with the corresponding responses returned to $\mathcal{A}$. It is easy to see that this is a perfect simulation of $\mathcal{A}$'s view in the MCFE security game. Thus, if $\mathcal{A}$ wins the MCFE security game adversary $\mathcal{B}$ breaks the security of the underlying MIFE scheme. We provide details below.

Initialize: $\mathcal{A}$ begins the game by declaring the corrupt set of parties $\mathcal{CP}$ to $\mathcal{B}$. Now $\mathcal{B}$'s job is to generate the corrupt encryption keys. For this, $\mathcal{B}$ has to generate ciphertexts for all possible plaintext-label pair for the corrupt parties by querying its challenge encryption oracle MIFE.EncO.

Specifically, for every $m \in \mathcal{X}_\kappa$ and $\mathsf{lb} \in \mathcal{L}_\kappa$, $\mathcal{B}$ creates an encryption with $x^0 = x^1 = m||\mathsf{lb}$ for all slots corresponding to parties in $\mathcal{CP}$, and $x^0 = x^1 = \perp$ where $\perp$ is the designated dummy message for all slots corresponding to honest parties in $\mathcal{H}$[2]. $\mathcal{B}$ makes all these $|\mathcal{X}_\kappa| \cdot |\mathcal{L}_\kappa|$ queries to his MIFE.EncO and uses the responses for the corrupted party slots to generate the corrupt encryption keys for $\mathcal{A}$.

Encryption Oracle: After obtaining ek_l for each $l \in \mathcal{CP}$, $\mathcal{A}$ can issue (at most q_2) encryption queries of the form $((x^0_{l,\mathsf{lb}}, \mathsf{lb}), (x^1_{l,\mathsf{lb}}, \mathsf{lb}))_{l\in\mathcal{H}}$ to obtain ciphertexts generated on behalf of honest parties. To simulate the responses to each of these queries, $\mathcal{B}$ creates , N^0 and N^1 as follows:

[2]Using $\perp$ for all honest parties guarantees that these ciphertexts cannot lead to a mix-and-match attack in MIFE.

- For each honest slot $l \in \mathcal{H}$: insert $(x^0_{l,\mathsf{lb}} \| \mathsf{lb})$ into N^0 and $(x^1_{l,\mathsf{lb}} \| \mathsf{lb})$ into N^1.
- For each corrupted slot $l \in \mathcal{CP}$: insert $(\perp)$ into both N^0 and N^1 .

$\mathcal{B}$ then queries its challenge encryption oracle MIFE.EncO with (N^0, N^1) and receives answers as MIFE.ct^σ where σ is the indistinguishability bit chosen by its challenger. $\mathcal{B}$ extracts MCFE.$\mathsf{ct}^\sigma_{x_l,\mathsf{lb}_l}$ for each $l \in \mathcal{H}$, and forwards these to $\mathcal{A}$.

KeyGen Oracle: For each functional key query of the form (f^0, f^1) among the q_1 KeyGen queries issued by $\mathcal{A}$, consider D_f as the functionality that first checks that all inputs share a common label and are non-$\perp$, and then evaluates f on the input plaintexts. For every such query, $\mathcal{B}$ invokes its MIFE.KeygenO on $(\mathsf{D}_{f^0}, \mathsf{D}_{f^1})$, for a total of q'_1 invocations. For its challenger's indistinguishability bit σ, it receives the answer $\mathsf{MIFE.sk}^\sigma_{\mathsf{D}_{f^\sigma}}$ and provides it to $\mathcal{A}$ as the simulated MCFE functional key sk_{f^b}. By construction, the admissibility checks in D_f ensure consistency with MCFE. The number of key queries is preserved, i.e.,

$$q'_1 = q_1.$$

Finalize: Eventually $\mathcal{A}$ outputs a bit b'. $\mathcal{B}$ outputs b' as its own guess for the challenge bit σ in the MIFE game. Clearly, if $\mathcal{A}$ wins the MCFE game with non-negligible advantage, then $\mathcal{B}$ does so in the MIFE game with the same advantage. Hence, the existence of $\mathcal{A}$ implies the existence of $\mathcal{B}$ breaking the adaptive security of MIFE, contradicting its assumed security.

Analysis: We must verify here that the encryption and key generation queries made by $\mathcal{A}$ do not make $\mathcal{B}$ inconsistent with Definition 3.2, i.e. that f^0 and f^1 agree on every possible mix-and-match combination for all N^0, N^1. This is not immediate, since $\mathcal{A}$ may issue encryption queries under distinct labels, and Definition 3.5 imposes no equality constraint between f^0 and f^1; a priori, such a combination could witness an inequality $f^0(N^0) \neq f^1(N^1)$ and render $\mathcal{B}$ invalid. We show that the construction of Figure 2 rules this out. The argument proceeds by considering the following two cases.

- **Case 1 – Label-consistent combination.** All slots carry a common label lb. Here the honest slots carry the challenge messages $(x^b_{l,\mathsf{lb}}, \mathsf{lb})$, while the corrupt slots satisfy $(x^0_{l,\mathsf{lb}}, \mathsf{lb}) = (x^1_{l,\mathsf{lb}}, \mathsf{lb}) = (m', \mathsf{lb})$. By condition 2 of Definition 3.5, the valid MCFE adversary $\mathcal{A}$ has f^0 and f^1 agree on every challenge encryption query sharing the label lb, giving

  $$f^0\big((x^0_{l,\mathsf{lb}})_{l\in\mathcal{H}}, \cdot\big) = f^1\big((x^1_{l,\mathsf{lb}})_{l\in\mathcal{H}}, \cdot\big).$$

  This is exactly the validity requirement of the underlying MIFE (Definition 3.2), so this case follows.
- **Case 2 – Label-inconsistent combination.** There exist l, l' with $\mathsf{lb}_l \neq \mathsf{lb}_{l'}$. Such a combination is a legal query for $\mathcal{A}$, since Definition 3.5 imposes no equality constraint across distinct labels, so f^0 and f^1 need not agree here. Even so, this does not break $\mathcal{B}$: the functional key D_f first checks label consistency across all slots and returns $\perp$ when the labels disagree. Since the label component is not identical in N^0 and N^1, this check fails on both sides giving

  $$f^0(N^0) = \perp = f^1(N^1).$$

In both cases the required equality $f^0(N^0) = f^1(N^1)$ holds, so every mix-and-match combination agrees across N^0 and N^1 and $\mathcal{B}$ is a valid (q_1, q_2)-bounded MIFE adversary.

Finally, we observe that $\mathcal{B}$ makes at most $q'_1 = q_1$ keygen queries and at most $q'_2 = q_2 + |\mathcal{X}_\kappa| \cdot |\mathcal{L}_\kappa|$ in simulating $\mathcal{A}$'s setup and oracle queries.

Since the simulation perfectly emulates $\mathcal{A}$'s game against the MCFE challenger, $\mathcal{A}$ wins the MCFE security game exactly when $\mathcal{B}$ wins the MIFE, thus proving the theorem.

□

Corollary 1. *For any security parameter κ, assuming one-way functions exist, there exists an adaptively secure private-key* MCFE *scheme for $\mathcal{F}$ with constant-arity over bounded domain with query bounds q_1, q_2 on the number of* KG *and* Enc *queries, respectively.*

Proof. The proof of Corollary 1 follows by instantiating the MIFE in Theorem 5.1 with the private-key MIFE scheme of Brakerski et al. [16], based on the private-key single-input functional encryption scheme of Gorbunov et al. [23]. □

Observation. We wish to stress that Corollary 1 only holds when the number of inputs t is a fixed constant that does not grow with the security parameter. This is necessary for the security of the Brakerski et al. MIFE construction [16] based on one-way functions. On the other hand, Theorem 5.1 does not have this requirement on the arity and can support a number of inputs growing with the security parameter. Unfortunately, we are not aware of any instantiations of MIFE for super-constant arity without relying on obfuscation.

6 Other Related Work

A large body of work has previously studied functional encryption in the multi-input and multi-client settings. We briefly review these papers here organized by the function class they support.

General FE:. Research on multi-input general FE was started by Goldwasser et al. [21], who proposed a construction of MCFE / MIFE based on indistinguishability obfuscation. In terms of expressiveness, their scheme is remarkably powerful, supporting all multi-input functionalities that can be computed by bounded-size circuits. Building on this line of work, Boneh et al. [11] introduced a private-key MIFE scheme with a stronger notion of security based on multilinear maps. While these papers achieved strong functionality, they both relied on strong assumptions, and moreover proved that these assumptions were necessary in some cases.

A key breakthrough came with the work of Ananth and Jain [7], who started the path towards weakening the assumptions by showing a *selectively-secure* private-key MIFE scheme can be constructed from general-purpose single-input public-key functional encryption. Concurrently, Brakerski et al. [15] gave a private-key MIFE scheme for functions under weaker assumptions, including based only on the minimal assumption of one-way functions. However, both of these latter works only constructed MIFE, leaving MCFE open for future work.

FE for Restricted Functionalities: Another line of work has instead looked at constructing MIFE/MCFE for restricted function classes. This line was started by Abdalla et al. [4] who showed how to construct MIFE for inner products from bilinear pairings. Further improvements by Abdalla et al. [3] developed a more general approach resulting in constructions from DDH, Paillier, and LWE. Later, Agrawal et al. [6] gave a construction of MIFE for quadratic functions.

The first construction of MCFE for inner products was given by Chotard et al. [17], proven secure under DDH in the random oracle model. Subsequent works obtained standard-model realizations: Abdalla et al. [1] under MDDH, DCR, and LWE assumptions, and Libert et al. [31] under LWE. Interestingly, Abdalla et al. [2] later demonstrated that their earlier MIFE construction [3] also gives an unlabeled MCFE scheme, thereby providing one of the first explicit connections between the two models. Beyond inner products, Ciampi et al. [18] constructed a MCFE scheme for separable functions.

Further work considered how to achieve function-privacy for MCFE. Agrawal et al. [6] presented the first function-hiding MCFE construction using bilinear pairings, proving its security in the random oracle model. Later, Shi and Vanjani [38] refined this approach to eliminate reliance on the random oracle model. Notably, all known function-hiding labeled MCFE schemes rely on pairings.

Acknowledgements

Arkady Yerukhimovich is supported in part by NSF grants CNS-1955620 and CNS-2144798 (CAREER).

References

[1] Michel Abdalla, Fabrice Benhamouda, and Romain Gay. 2019. From single-input to multi-client inner-product functional encryption. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 552–582.

[2] Michel Abdalla, Fabrice Benhamouda, Markulf Kohlweiss, and Hendrik Waldner. 2019. Decentralizing inner-product functional encryption. In *IACR International Workshop on Public Key Cryptography*. Springer, 128–157.

[3] Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu. 2018. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In *Annual International Cryptology Conference*. Springer, 597–627.

[4] Michel Abdalla, Romain Gay, Mariana Raykova, and Hoeteck Wee. 2017. Multi-input inner-product functional encryption from pairings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 601–626.

[5] Shweta Agrawal, Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. 2013. Functional encryption: New perspectives and lower bounds. In *Annual Cryptology Conference*. Springer, 500–518.

[6] Shweta Agrawal, Rishab Goyal, and Junichi Tomida. 2021. Multi-input quadratic functional encryption from pairings. In *Annual International Cryptology Conference*. Springer, 208–238.

[7] Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. 2015. Indistinguishability obfuscation from functional encryption for simple functions. *Cryptology ePrint Archive* (2015).

[8] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. 2001. On the (im) possibility of obfuscating programs. In *Annual International Cryptology Conference*. Springer, 1–18.

[9] Mihir Bellare and Adam O'Neill. 2013. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. In *International Conference on Cryptology and Network Security*. Springer, 218–234.

[10] Nir Bitansky and Vinod Vaikuntanathan. 2018. Indistinguishability obfuscation from functional encryption. *J. ACM* 65, 6 (2018), 1–37.

[11] Dan Boneh, Kevin Lewi, Mariana Raykova, Amit Sahai, Mark Zhandry, and Joe Zimmerman. 2015. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 563–594.

[12] Dan Boneh, Amit Sahai, and Brent Waters. 2011. Functional encryption: Definitions and challenges. In *Theory of Cryptography Conference*. Springer, 253–273.

[13] Dan Boneh and Brent Waters. 2007. Conjunctive, subset, and range queries on encrypted data. In *Theory of Cryptography Conference*. Springer, 535–554.

[14] Elette Boyle, Kai-Min Chung, and Rafael Pass. 2014. On extractability obfuscation. In *Theory of Cryptography Conference*. Springer, 52–73.

[15] Zvika Brakerski, Ilan Komargodski, and Gil Segev. 2018. Multi-input functional encryption in the private-key setting: Stronger security from weaker assumptions. *Journal of Cryptology* 31, 2 (2018), 434–520.

[16] Zvika Brakerski and Gil Segev. 2018. Function-private functional encryption in the private-key setting. *Journal of Cryptology* 31, 1 (2018), 202–225.

[17] Jérémy Chotard, Edouard Dufour Sans, Romain Gay, Duong Hieu Phan, and David Pointcheval. 2018. Decentralized multi-client functional encryption for inner product. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 703–732.

[18] Michele Ciampi, Luisa Siniscalchi, and Hendrik Waldner. 2021. Multi-client functional encryption for separable functions. In *IACR International Conference on Public-Key Cryptography*. Springer, 724–753.

[19] Angelo De Caro, Vincenzo Iovino, Abhishek Jain, Adam O'Neill, Omer Paneth, and Giuseppe Persiano. 2013. On the achievability of simulation-based security for functional encryption. In *Annual Cryptology Conference*. Springer, 519–535.

[20] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. 2016. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM J. Comput.* 45, 3 (2016), 882–929.

[21] Shafi Goldwasser, S Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. 2014. Multi-input functional encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 578–602.

[22] Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. 2013. Reusable garbled circuits and succinct functional encryption. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. 555–564.

[23] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. 2012. Functional encryption with bounded collusions via multi-party computation. In *Annual Cryptology Conference*. Springer, 162–179.

[24] S Dov Gordon, Jonathan Katz, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. 2013. Multi-input functional encryption. *Cryptology ePrint Archive* (2013).

[25] S Dov Gordon, Jonathan Katz, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. 2015. Multi-client verifiable computation with stronger security guarantees. In *Theory of Cryptography Conference*. Springer, 144–168.

[26] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. 2006. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCS*. 89–98.

[27] Jonathan Katz, Amit Sahai, and Brent Waters. 2008. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 146–162.

[28] Fuyuki Kitagawa, Ryo Nishimaki, and Keisuke Tanaka. 2018. Obfustopia built on secret-key functional encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 603–648.

[29] Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. 2014. One-way functions and (im) perfect obfuscation. In *IEEE FOCS*. 374–383.

[30] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. 2010. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 62–91.

[31] Benoît Libert and Radu Tıtiu. 2019. Multi-client functional encryption for linear functions in the standard model from LWE. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 520–551.

[32] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. 2022. Multi-client functional encryption with fine-grained access control. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 95–125.

[33] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. 2023. Optimal security notion for decentralized multi-client functional encryption. In *International Conference on Applied Cryptography and Network Security*. Springer, 336–365.

[34] Ky Nguyen, Duong Hieu Phan, and David Pointcheval. 2025. Multi-client functional encryption with public inputs and strong security. In *IACR International Conference on Public-Key Cryptography*. Springer, 68–101.

[35] Ky Nguyen, David Pointcheval, and Robert Schädlich. 2024. Decentralized multi-client functional encryption with strong security. *Cryptology ePrint Archive* (2024).

[36] Adam O'Neill. 2010. Definitional issues in functional encryption. *Cryptology ePrint Archive* (2010).

[37] Amit Sahai and Brent Waters. 2005. Fuzzy identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 457–473.

[38] Elaine Shi and Nikhil Vanjani. 2023. Multi-client inner product encryption: Function-hiding instantiations without random oracles. In *IACR International Conference on Public-Key Cryptography*. Springer, 622–651.