Generic Conversions from CPA to CCA secure Functional Encryption

Mridul Nandi and Tapas Pandit

Indian Statistical Institute, Kolkata
mridul@isical.ac.in and tapasgmmath@gmail.com

Abstract. In 2004, Canetti-Halevi-Katz and later Boneh-Katz showed generic CCA-secure PKE constructions from a CPA-secure IBE. Goyal et al. in 2006 further extended the aforementioned idea implicitly to provide a specific CCA-secure KP-ABE with policies represented by monotone access trees. Later, Yamada et al. in 2011 generalized the CPA to CCA conversion to all those ABE, where the policies are represented by either monotone access trees (MAT) or monotone span programs (MSP), but not the others like sets of minimal sets. Moreover, the underlying CPA-secure constructions must satisfy one of the two features called *key-delegation* and *verifiability*. Along with ABE, many other different encryptions schemes, such as inner-product, hidden vector, spatial encryption schemes etc. can be studied under an unified framework, called *functional encryption* (FE), as introduced by Boneh-Sahai-Waters in 2011. The generic conversions, due to Yamada et al., can not be applied to all these functional encryption schemes. On the other hand, to the best of our knowledge, there is no known CCA-secure construction beyond ABE over MSP and MAT. This paper provides different ways of obtaining CCA-secure functional encryptions of almost all categories. In particular, we provide **a generic conversion from a CPA-secure functional encryption into a CCA-secure functional encryption** provided the underlying CPA-secure encryption scheme has either restricted delegation or verifiability feature. We observe that almost all functional encryption schemes have this feature. The KP-FE schemes of Waters (proposed in 2012) and Attrapadung (proposed in 2014) for regular languages do not possess the usual delegation property. However, they can be converted into corresponding CCA-secure schemes as they satisfy the *restricted delegation*.

Keywords: Functional encryption, Predicate encryption, Delegation, Verifiability, Generic Conversion.

1 Introduction

Identity based Encryption. The day of PKC was started through the invention of key exchange protocol by Diffie and Hellman [15]. The major problem it faced is the man-in-middle attack. To take care this issue, the common practice is to use the certificate-based digital signature and keep all the certificates in a public directory (certified). Then, ID-based cryptosystem [34] was introduced

to simplify the mentioned key management process, where the public key is the identity. For many practical purposes, the stronger (IND-CCA) security is assumed to be mandatory for the hired encryption scheme. To this direction, one of the efficient transformations is due to Fujisaki and Okamoto [16]. In this generic conversion, a one-way secure PKE scheme is transformed to CCA-secure PKE scheme using a symmetric encryption scheme (in the sense of hybrid encryption), but this assumes the random oracles. Canetti, Halevi and Katz [11] first came up with a transformation, known as *CHK-transformation*, from a CPA-secure IBE to CCA-secure PKE in the standard model. The basic technique used for this conversion is the one-time signature (OTS), later Boneh and Katz [8] improved the efficiency by employing a weak commitment and MAC technique.

Attribute-Based Encryption. The attribute-based encryption (ABE) [24, 28] is a generalization of IBE [6, 14], a smart way to provide the access control over the secrets in fine grained manner. The access control that ABE implements are the boolean formulas (access structures), in the form of monotone span programs, monotone access trees (MAT) and the sets of minimal sets etc. The attribute-based encryption mainly hides the payload, whereas the associated indices are given in the clear as the part of ciphertexts and we call this ABE as traditional ABE.

In the standard model, there are very few CCA-secure ABE schemes which are either directly constructed or from CPA-secure ABE by applying the abstraction of [20]. The intuition in [20] basically extended the key idea of *CHK-transformation* to the area of ABE in presence of delegation, but it was applicable only to the large universe ABE (with MAT representation) in key-policy flavor. Recently, Yamada et al. [38] generalized the abstraction of Goyal et al. [20] in the standard model to include all ABE that support either monotone access trees or monotone span programs. In their generic conversion [38], they also offered an alternative of delegation for the ABE without delegation, called *verifiability* which seems to be more powerful than the delegation.

Predicate Encryption. Beyond the traditional ABE, there are many encryption systems available in the literature, some of them are known to be (doubly-)spatial encryption ((D-)SE) [7, 21], functional encryption for regular languages [37], ABE for circuits [17], (hierarchical) inner-product encryption ((H)IPE) [27], hidden vector encryption (HVE) [10] and anonymous IBE etc. All the above systems are subsumed under a larger class of encryption system, called predicate encryption (PE) [23, 35]. A key-index x is eligible for the decryption of a ciphertext encrypted under the associated index y if a relation holds between x and y (in this case we write $x \sim y$, and $x \not\sim y$ otherwise). It captures both, the only payload hiding and the associated index hiding. The former is known to be PE with public index and the later is PE with hidden index.

Functional Encryption. All these encryption systems can be studied under a 'functional'-style framework (formally defined in section 2.1), known as functional encryption (FE) [9]. Informally speaking, a FE for a functionality $\mathcal{F}$ defined over the key space $\mathcal{X}$ and message space Ψ, deals with the computation of

$\mathcal{F}(x, \psi)$ from the key $\mathcal{SK}_x$, associated to a key-index x, and a ciphertext C_ψ encrypted for a message-index ψ. A brief literature survey on functional encryption is given in Appendix C.

Our Contribution. In this paper, we explore a generic conversion from CPA to CCA-security for the functional encryption systems [9]. The only generic solution available in this domain is for traditional ABE due to Yamada et al. [38]. We basically extend this conversion to include the larger class of encryption systems (beyond the traditional ABE), i.e., the functional encryption. To the best of our knowledge, this is the first conversion that can transform a CPA-secure FE to CCA-secure FE, generically. Like Yamada et al., the underlying CPA-secure schemes are required to satisfy either *restricted delegation*, a weaker notion of delegation or *restricted verifiability*, a weaker notion of verifiability or both. Our conversion is more general and widely applicable.

- The Yamada et al. [38] conversion for traditional ABE, can be viewed as a specific instantiation of our generic conversion.
- It is applicable to all ABE even over policies represented by *sets of minimal sets* or a general circuit (which was not captured by [38]).
- It is also applicable to any regular language recognized by a deterministic finite automaton (DFA). As a result we are able to provide a CCA-secure functional encryption schemes for regular languages using Waters [37] and Attrapadung [1] CPA-secure constructions. None of these schemes possesses the actual delegation, but by introducing a suitable (restricted) partial order in the key space, we convert them into CCA-secure schemes smartly.
- Other categories of functional encryption such as (doubly-)spatial encryption, (Hierarchical-)inner-product encryption and hidden vector encryption etc. also follow our generic CCA-conversion.

Our Approach. The basic hired technique involved in our generic conversion, is the OTS or a combination of weak commitment and MAC. We briefly explain our conversion using OTS. Let $(\mathsf{vk} \in \{0,1\}^n, \mathsf{signk})$ be a pair of verification key and signing key for OTS. In this approach, vk is embedded into the ciphertext obtained from a CPA-secure encryption and then the ciphertext is signed by OTS scheme using signk to form the final ciphertext (CCA-secure). The verification key vk is kept pubic in the ciphertext for the verification of the signature. The main challenging task is to embed vk appropriately in the ciphertext so that an attacker $\mathscr{A}$ can not create a new ciphertext (mostly well-formed or ill-formed but up to certain extent) from the original ciphertext. Let $\mathcal{F}$, $\mathcal{X}$ and Ψ be respectively the functionality, key space and message space for a target CCA-secure functional encryption FE to be constructed from a CPA-secure FE' for $(\mathcal{F}', \mathcal{X}', \Psi')$. So we need suitable mappings $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ from $(\mathcal{F}, \mathcal{X}, \Psi)$ to $(\mathcal{F}', \mathcal{X}', \Psi')$ which would transform the indices. These map must satisfy certain conditions and we call the triple of maps **delegation-friendly** or **verification-friendly** index-transformer depending on the satisfied properties (see definition 4 and 6). In section 4 we see several examples of index-transformer over different categories of functional

encryption. In section 3 we provide our generic construction which is briefly constructed as follows:

1. Secret key of FE to an index x is same as that of FE' to the index x' (obtained by applying $\mathcal{T}_1$).
2. To encrypt a message ψ for FE, we apply CPA-encryption for ψ_{vk} where $(\mathsf{vk}, \mathsf{signk})$ is generated from the key generation of an OTS scheme and ψ_{vk} is a transformed message index obtained through $\mathcal{T}_3$. We also sign the ciphertext $\mathsf{C}_{\psi_{\mathsf{vk}}}$ by signk and keep the signature and verification key both as a part of the final ciphertext. The decryption algorithm varies depending on the delegation or verifiability features, where the mapping $\mathcal{T}_2$ is to be applied.

In section 5 we observe that almost all known constructions (CPA-secure) satisfy either delegation or verifiability or both. However, the schemes of Waters [37] and Attrapadung [1] for regular languages do not have full delegation. However, we show that it has restricted-delegation which is actually required for our generic conversion. In Appendix G, we provide the concrete instantiation of KP-CCA-secure construction for regular languages. The similar instantiation is possible for almost all other functional encryptions.

2 Preliminaries

2.1 Functional Encryption

Notation. $[\ell] := \{i \in \mathbb{N} : 1 \le i \le \ell\}$. For a set X, $x \xleftarrow{\mathrm{R}} X$ denotes that x is randomly picked from X according to the distribution, R. Likewise, $x \xleftarrow{\mathrm{U}} X$ indicates x is uniformly selected from X. For two strings str_1 and str_2, $str_1||str_2$ denotes the concatenation of str_1 and str_2. The notation B^T stands for the transposition of the matrix B. For two column vectors $\boldsymbol{u}$ and $\boldsymbol{u}$, we define $\binom{\boldsymbol{u}}{\boldsymbol{v}} := (u_1, \ldots, u_\ell, v_1, \ldots, v_d)^T$, where $\boldsymbol{u}^T := (u_1, \ldots, u_\ell)$ and $\boldsymbol{v}^T := (v_1, \ldots, v_d)$. For $\mathsf{vk} \in \{0,1\}^n$, we use the notations $(\mathsf{vk}^\perp)^T := (\mathsf{vk}, -1)$ and $(\mathsf{vk})^T := (1, \mathsf{vk})$.

Although Boneh, Sahai and Waters explained in details the definition, security and other properties of functional encryption in [9], for self-containment, we briefly define them here. Let $\mathcal{F} : (\mathcal{X} \cup \{\epsilon\}) \times \Psi \longrightarrow \{0,1\}^* \cup \{\perp\}$ be a function. We call $\mathcal{X}$ and Ψ the key-index space and message space respectively. To each key-index x, we associate a possibly randomized key $\mathcal{SK}_x$. A functional encryption scheme for a function $\mathcal{F}$ allows to compute $\mathcal{F}(x, \psi)$ using the key $\mathcal{SK}_x$ for a key-index x and an encryption C_ψ of the message ψ, i.e., $\mathcal{F}(x, \psi) = \mathsf{Dec}(\mathsf{C}_\psi, \mathcal{SK}_x)$.[1] A precise definition of the functional encryption for $\mathcal{F}$ is given below:

[1] Conventionally, an empty key is associated with the empty key-index. Thus, anybody can use the empty key to decrypt a ciphertext C_ψ and obtains $\mathcal{F}(\epsilon, \psi)$. So we define $\mathcal{F}(\epsilon, \cdot)$ to denote the amount of information we are comfortable to leak from a ciphertext only. Usually, length of message is leaked. Sometimes we also leak additional data, header file etc. All these information would be captured by $\mathcal{F}(\epsilon, \cdot)$.

Definition 1 (Functional Encryption). *A basic* functional encryption scheme FE *for a functionality* $\mathcal{F}$ *consists of four PPT algorithms -* Setup*,* KeyGen*,* Enc *and* Dec*:*

- $\mathsf{Setup}(1^\kappa, j) \to (\mathcal{PP}, \mathcal{MSK})$ *where* $j \in \mathcal{J}$ *(the space for system parameters).*
- $\mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x) := \mathsf{KeyGen}_{\mathcal{PP},\mathcal{MSK}}(x) \to \mathcal{SK}_x$ *(secret key associated to a key-index* x*).*
- $\mathsf{Enc}(\mathcal{PP}, \psi) \to \mathsf{C}_\psi$ *(a ciphertext) where* $\psi \in \Psi$ *(message space) is called a message.*
- $\mathsf{Dec}(\mathcal{PP}, \mathsf{C}_\psi, \mathcal{SK}x) := \mathsf{Dec}_{\mathcal{PP},\mathcal{SK}_x}(\mathsf{C}_\psi) \to p \in \{0,1\}^* \cup \{\perp\}$.

Moreover, it satisfies the correctness *condition:* $\mathsf{Pr}[\mathsf{Dec}_{\mathcal{PP},\mathcal{SK}_x}(\mathsf{Enc}(\mathcal{PP}, \psi)) = \mathcal{F}(x, \psi)] = 1$.

Whenever the public parameter $\mathcal{PP}$ and the master secret key $\mathcal{MSK}$ are understood we skip them for notational simplicity. We call a ciphertext C **ill-format** if for all ψ, $\mathsf{Pr}[\mathsf{Enc}(\mathcal{PP}, \psi) = \mathsf{C}] = 0$, otherwise it is called correctly-format. So the correctness condition implies that for all correctly-format ciphertext C, $\mathsf{Dec}_{\mathcal{SK}_x}(\mathsf{C}) = \mathcal{F}(x, \psi)$ provided $\mathsf{Pr}[\mathsf{Enc}(\mathcal{PP}, \psi) = \mathsf{C}] > 0$. But its does not talk about anything how the decryption algorithm behaves on all ill-format ciphertexts. A public *format-verifier* is an algorithm $\mathcal{V}$ such that $\mathcal{V}(\mathcal{PP}, \mathsf{C})$ returns 1 if and only if C is correctly-format. An algorithm $\mathcal{V}$ is called *weak format-verifier* if for all correctly-format C, $\mathcal{V}(\mathcal{PP}, \mathsf{C})$ returns 1. For security definition of functional encryption, we refer to Appendix A.

Predicate Encryption A predicate encryption PE for a key space $\mathcal{X}$, an associated data space $\mathcal{Y}$ and a payload space $\mathcal{M}$ can be realized as a functional encryption FE for a functionality $\mathcal{F}$ over $(\mathcal{X} \cup \{\epsilon\}) \times \Psi$, where $\Psi = \mathcal{Y} \times \mathcal{M}$. The functionality (also called predicate functionality), $\mathcal{F}$ for the PE is defined by

$$\mathcal{F}(x \in \mathcal{X}, (y, m) \in \Psi) = \begin{cases} m & \text{if } x \sim y \\ \perp & \text{if } x \not\sim y \end{cases}$$

where $\sim$ is some binary relation on $\mathcal{X} \times \mathcal{Y}$. The predicate encryption is said to be with *public index* if the associated index y is not hidden, otherwise it is called to be with *hidden index* (when confidentiality of associated data is also required). In the case of public index, we can assume that y (but not m) is a part of C_ψ where $\psi = (y, m)$ and we may define $\mathcal{F}(\epsilon, (y, m)) = (y, len(m))$. In hidden index, we may define $\mathcal{F}(\epsilon, (y, m)) = (len(y), len(m))$. So far all known functional encryptions are predicate encryptions for some binary relations. We have different categories of predicate encryption based on how we actually compute the predicate relation. Here are some popularly known examples of relations which have been used for predicate encryption (in case of asymmetric relation one can always define dual relation by interchanging $\mathcal{X}$ and $\mathcal{Y}$, e.g., key-policy or KP and ciphertext-policy or CP):

HVE relation: Let Σ be an alphabet and $*$ (referred as "wild card") be a special symbol not in Σ. We set $\Sigma_* := \Sigma \cup \{*\}$. Let $\mathcal{X} := \Sigma_*^\ell$ and $\mathcal{Y} = \Sigma^\ell$. For $x = (x_1, \ldots, x_\ell) \in \mathcal{X}$ and $y = (y_1, \ldots, y_\ell) \in \mathcal{Y}$, we define $x \sim y$ if and only if $x_i = y_i$ or $x_i = *$ for each $i \in [\ell]$. The corresponding encryption schemes are known as hidden vector encryption.

Orthogonal: Let $\mathcal{X} = \mathcal{Y} = \mathbb{F}^\ell$ where $\mathbb{F}$ is a finite field. For $x = (x_1, \ldots, x_\ell) \in \mathcal{X}$ and $y = (y_1, \ldots, y_\ell) \in \mathcal{Y}$, we define $x \sim y$ if and only if $\sum_{i=1}^{\ell} x_i.y_i = 0$. The corresponding encryption schemes are known as inner-product predicate encryption.

LSSS based relation: Let $\mathcal{U}$ be a set (of attributes). Define $\mathcal{X} = \mathcal{P}(\mathcal{U})$ and $\mathcal{Y}$ be the the set of all monotone span programs $\Gamma := (M, \rho)$ where M is an $\ell \times r$ matrix over $\mathbb{F}$ and $\rho : [\ell] \to \mathcal{U}$. We define the binary relation $A \sim (M, \rho)$ if $(1, 0, \ldots, 0) \in span(\{\boldsymbol{M}_i; \rho(i) \in A\})$ where $\boldsymbol{M}_i$ is the i^{th} row of the matrix M.

Relation based on minimal set representation: Let Γ be a policy represented by the set of minimal sets, $(B_1, \ldots, B_\ell)$ and A be a set of attributes. We define the binary relation $A \sim \Gamma$ if $\exists\ i \in [\ell]$ such that $B_i \subset A$.

Satisfiability relation in circuit: Let $\mathcal{X}$ be the set of all circuits "$\wedge$" and "$\vee$" gates [2] with having n bits input gates. Let $\mathcal{Y} = \{0,1\}^n$. We define $x \sim y$ if x is satisfied by y or y is satisfied by x in the sense of circuit satisfiability.

Relation in Spatial encryption: Let $\mathcal{Y}$ be an affine space $\mathbb{Z}_q^\ell$ and $\mathcal{X}$ be the set of all affine subspaces of $\mathbb{Z}_q^\ell$. For $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, we define $x \sim y$ if and only if $y \in x$.

Relation in Doubly-Spatial encryption: Let $\mathcal{Y} = \mathcal{X}$ be the set of all affine subspaces of $\mathbb{Z}_q^\ell$. For $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, we define $x \sim y$ if and only if $y \cap x \neq \emptyset$. In a more general set up, $\mathcal{X}$ is defined to be the set of all linear subspaces, of the form $\mathsf{Ker}(\mathsf{X})$, a kernel of a matrix X over $\mathbb{Z}_q$.

Regular languages vs. Automata: A deterministic finite automaton (DFA), M is defined to be a quintuple $(Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is a set of symbols, called alphabet, $q_0 \in Q$ is called the start state, $F \subseteq Q$ is called the set of final states and a partial function $\delta : Q \times \Sigma \to Q$ is called transition function. The language, also called *regular language*, recognized by the DFA M is defined as

$$\mathcal{L}(M) = \{w_1 w_2 \cdots w_n \in \Sigma^* : \delta(\cdots \delta(\delta(q_0, w_1), w_2) \cdots w_n) \in F\}.$$

We define a binary relation $w \sim M$ if $w \in \mathcal{L}(M)$

3 Generic Conversion to CCA from CPA-Secure Functional Encryption

3.1 Delegation and Verifiability

Definition 2 (Delegation and re-randomization for FE). *Let $\succ$ be a partial order on $\mathcal{X}$. A functional encryption scheme* FE *is said to have the* del-

[2] These are 2-threshold gates, one may consider general threshold gates, e.g., t-threshold gates.

egation property *w.r.t.* $\succ$ *if there is a PPT algorithm* Delegate *such that for all* $x \succ \tilde{x} \in \mathcal{X}$*, for all* pp, msk, k, k_x *with* $\mathsf{Pr}[\mathsf{Setup} \rightarrow (pp, msk)] > 0$ *and* $\mathsf{Pr}[\mathsf{KeyGen}(pp, msk, x) = k_x] > 0$ *we have*

$$\mathsf{Pr}[\mathsf{Delegate}(pp, k_x, x, \tilde{x}) = k] = \mathsf{Pr}[\mathsf{KeyGen}(pp, msk, \tilde{x}) = k]. \tag{1}$$

Moreover, it is said to have re-randomization *property if for all* $x \in \mathcal{X}$*,* $x \succ x$*.*

For many practical reason, Alice may have to compute a "delegate-key" for her assistant, Charlie. For example, Alice is conducting a conference and gets so busy with her tight schedule. Then, she wants Charlie to handle the registration process and so, she computes a key for Charlie in a restricted manner so that he can only see the message related to registration, not beyond that. The term "restricted" would be justified by a choice of partial order $\succ$ in the key space. One can always define a natural partial order through the binary relation $\sim$ of the predicate encryption. Indeed, for $x, \tilde{x} \in \mathcal{X}$, $x \succ \tilde{x}$ (i.e., x **has more access than** $\tilde{x}$) if $\tilde{x} \sim y$ implies $x \sim y$ for all $y \in \mathcal{Y}$.

Verifiability for attribute based encryption has been defined in [38]. However, we provide a simplified definition for a general public index predicate encryption.

Definition 3 (Verifiability). *A predicate encryption scheme* PE *with public index is said to have the verifiability if there is a PPT algorithm,* Verify *such that for all ciphertext* C *(possibly ill-format) with the public associated index* y*, and all* $x, \tilde{x}$ *with* $x \sim y, \tilde{x} \sim y$ *we have*

$$\mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x, \tilde{x}) = 1 \Rightarrow \mathsf{Dec}(\mathcal{PP}, \mathsf{C}, \mathcal{SK}_x) = \mathsf{Dec}(\mathcal{PP}, \mathsf{C}, \mathcal{SK}_{\tilde{x}})$$

and it is a weak format-verifier, i.e., it returns 1 for all correctly-format ciphertext.[3]

Roughly speaking, it verifies that a ciphertext is correctly-format or if it is ill-format then it can be decrypted to the same message under two keys with two different indices both related to the associated index. Note that we can not define verifiability for a hidden-index predicate encryption and hence for a general functional encryption.

3.2 Using Delegation feature

In this section, we construct a generic CCA-secure functional encryption scheme FE from a CPA-secure functional encryption $\mathsf{FE}' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$. Let $\mathcal{F} : \mathcal{X} \times \Psi \rightarrow \{0,1\}^*$ and $\mathcal{F}' : \mathcal{X}' \times \Psi' \rightarrow \{0,1\}^*$ be two functionalities. We describe a transformation which transforms key-index and message for $\mathcal{F}$ to a transformed key-index and message for $\mathcal{F}'$.

Definition 4. *A triple of maps* $\mathcal{T}_1 : \mathcal{X} \longrightarrow \mathcal{X}'$*,* $\mathcal{T}_2 : \mathcal{X} \times \{0,1\}^n \longrightarrow \mathcal{X}'$ *and* $\mathcal{T}_3 : \Psi \times \{0,1\}^n \longrightarrow \Psi'$ *is called* **delegation-friendly index-transformer** *from* $(\mathcal{F}, \mathcal{X}, \Psi)$ *to* $(\mathcal{F}', \mathcal{X}', \Psi')$ *if the following conditions*[4] *(given in box) are satisfied for all* $x \in \mathcal{X}$*,* $\mathsf{vk} \neq \mathsf{vk}' \in \{0,1\}^n$ *and* $\psi \in \Psi$*.*

[3] So if $\mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x, \tilde{x}) = 0$ for $x \sim y, \tilde{x} \sim y$ then C must be ill-format.

[4] We would refer as delegation-friendly conditions.

(1) $\mathcal{F}(x,\psi)=\mathcal{F}'(x_{\mathsf{vk}},\psi_{\mathsf{vk}})$*, (2)* $\mathcal{F}(x,\psi)=\mathcal{F}'(x',\psi_{\mathsf{vk}})$ *and (3)* $\mathcal{F}'(x_{\mathsf{vk}},\psi_{\mathsf{vk}'})=\perp$.

where we simply denote $\mathcal{T}_1(x)$, $\mathcal{T}_2(x,\mathsf{vk})$ *and* $\mathcal{T}_3(\psi,\mathsf{vk})$ *by* x',x_{vk} *and* ψ_{vk} *respectively.*

Definition 5 (Restricted Delegation for FE). *An algorithm* Delegate *is said to be a restricted-delegatable algorithm for a functional encryption scheme* FE' *w.r.t. an index-transformer* $(\mathcal{T}_1,\mathcal{T}_2,\mathcal{T}_3)$ *if for all* $x\in\mathcal{X},\mathsf{vk}\in\{0,1\}^n$*, the equation (1) holds for the partial order of the form,* $x_{\mathsf{vk}}\succ x_{\mathsf{vk}}$ *and* $x'\succ x_{\mathsf{vk}}$.

A Generic Construction based on Restricted Delegation. We provide a generic construction of a functional encryption $\mathsf{FE}=(\mathsf{Setup},\mathsf{KeyGen},\mathsf{Enc},\mathsf{Dec})$ for a functionality $\mathcal{F}$ based on a functional encryption $\mathsf{FE}'=(\mathsf{Setup}',\mathsf{KeyGen}',\mathsf{Enc}',\mathsf{Dec}')$ for a functionality $\mathcal{F}'$ and a valid index-transformer $(\mathcal{T}_1,\mathcal{T}_2,\mathcal{T}_3)$ from $(\mathcal{F},\mathcal{X},\Psi)$ to $(\mathcal{F}',\mathcal{X}',\Psi')$. We assume that FE' has restricted delegation property for all x and vk. Let $\mathsf{OTS}=(\mathsf{OTS.Gen},\mathsf{OTS.Sign},\mathsf{OTS.Ver})$ be a one-time signature scheme. The setup algorithm is same as before and it returns $(\mathcal{PP},\mathcal{MSK})$.[5] Now we describe the other three algorithms (implicitly understood $\mathcal{PP},\mathcal{MSK}$):

Conversion based on Delegation and using One-Time-Signature

- $\mathsf{KeyGen}(x):=\mathsf{KeyGen}'(x')$ (in notation: $\mathcal{SK}_x:=\mathcal{SK}'_{x'}$).
- $\mathsf{Enc}(\psi)$: It runs $(\mathsf{vk},\mathsf{signk})\longleftarrow\mathsf{OTS.Gen}(1^\kappa)$ and returns

$$\mathsf{CT}=(\mathsf{C}:=\mathsf{Enc}'(\psi_{\mathsf{vk}}),\delta:=\mathsf{OTS.Sign}(\mathsf{C},\mathsf{signk}),\mathsf{vk}).$$

- $\mathsf{Dec}_{\mathcal{SK}_x}(\mathsf{C},\delta,\mathsf{vk})=\begin{cases}\mathsf{Dec}'_K(\mathsf{C}) & \text{if } \begin{pmatrix}\mathsf{OTS.Ver}(\mathsf{C},\delta,\mathsf{vk})=1 \text{ and}\\ K\leftarrow\mathsf{Delegate}(\mathcal{PP},\mathcal{SK}'_{x'},x',x_{\mathsf{vk}})\end{pmatrix}\\ \perp & \text{otherwise.}\end{cases}$

Correctness : Let $(\mathcal{PP},\mathcal{MSK})\longleftarrow\mathsf{Setup}(1^\kappa,j)$ and for all $x\in\mathcal{X}$, $\psi\in\Psi$, let $\mathcal{SK}_x\longleftarrow\mathsf{KeyGen}(\mathcal{PP},\mathcal{MSK},x)$ and $\mathsf{CT}\longleftarrow\mathsf{Enc}(\mathcal{PP},\psi)$. Then,

$$\begin{aligned}\mathsf{Dec}(\mathsf{CT},\mathcal{SK}_x)&=\mathsf{Dec}'(\mathsf{C},\mathcal{SK}'_{x_{\mathsf{vk}}}) && \begin{pmatrix}\text{by correctness of OTS}\\ \text{and definition of Dec}\end{pmatrix}\\ &=\mathcal{F}'(x_{\mathsf{vk}},\psi_{\mathsf{vk}}) && \text{(by correctness of given } \mathsf{FE}')\\ &=\mathcal{F}(x,\psi). && \text{(by given condition (1))}\end{aligned}$$

[5] It may not be identical as it can include some public parameter for the one time signature which would be used. Moreover, the index transformer also maps the system parameters accordingly. However, we skip those technical details as it does not harm in understanding the actual conversions.

Theorem 1. *Let* $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ *be a delegation-friendly index-transformer,* FE' *be an IND-CPA secure functional encryption scheme with the restricted delegation and* OTS *be a strong unforgeable one-time signature scheme, then the above proposed scheme* FE *in section 3.2 is an IND-CCA secure functional encryption scheme.*

Proof. We describe for the adaptive security. One can similarly have a proof for selective security. If an adversary $\mathscr{A}$ can break the IND-CCA security of the proposed scheme FE, then we establish an algorithm $\mathscr{B}$, called simulator for breaking IND-CPA security of the primitive functional encryption scheme FE' with advantage $\mathsf{Adv}^{\mathrm{IND-CCA}}_{\mathscr{A},\mathrm{FE}}(\kappa) - \mathsf{Adv}^{\mathrm{sUF-CMA}}_{\mathscr{A},\mathrm{OTS}}(\kappa)$. Let $\mathcal{CH}$ be the challenger for the primitive functional encryption scheme FE'. Now we describe how $\mathscr{B}$ works with $\mathcal{CH}$ with the help of $\mathscr{A}$. Here $\mathscr{B}$ will try to behave CCA-challenger of $\mathscr{A}$ against FE'. Note that $\mathscr{B}$ can not make decryption query while $\mathscr{A}$ can make. So the main thing is to show how $\mathscr{B}$ can respond decryption queries with the help of key-generation queries so that $\mathscr{B}$ in one hand does not violate the rule of CPA game with $\mathcal{CH}$ and other hand, it simulates perfectly as a challenger of $\mathscr{A}$. We denote the behavior of $\mathscr{B}$ as $\mathscr{B}^{\mathcal{CH}_{CPA}}$ and described below:

Algorithm $\mathscr{B}^{\mathcal{CH}_{CPA}}$: It first run $(\mathsf{vk}^*, \mathsf{signk}^*) \longleftarrow \mathsf{OTS.Gen}(1^\kappa)$. In the setup phase $\mathscr{B}$ simply forwards the public parameter $\mathcal{PP}$, obtained from $\mathcal{CH}$, to $\mathscr{A}$.

Phase 1/2 Query: It consists of the following queries in adaptive manner:

- KeyGen Query: Let $x \in \mathcal{X}$ be a key-index queried by $\mathscr{A}$, then $\mathscr{B}$ makes a key query for $x' := \mathcal{T}_1(x)$ to $\mathcal{CH}$. Then $\mathcal{CH}$ replies the key $\mathcal{SK}_x = \mathcal{SK}'_{x'}$ to $\mathscr{B}$ and the same key is passed to $\mathscr{A}$.
- Decrypt Query: Let (CT, x), where $\mathsf{CT} = (\mathsf{C}, \delta, \mathsf{vk})$ be a decryption query by $\mathscr{A}$. Then $\mathscr{B}$ runs $\mathsf{flag}_o \longleftarrow \mathsf{OTS.Ver}(\mathsf{C}, \delta, \mathsf{vk})$ and if $\mathsf{flag}_o = \mathsf{False}$, it returns $\perp$ to $\mathscr{A}$ else proceeds. If $\mathsf{vk} = \mathsf{vk}^*$, $\mathscr{B}$ aborts the game we set $\mathsf{BAD}_{\mathrm{OTS}}$ true, else moves to next step. $\mathscr{B}$ makes a key query for the key-index $x_{\mathsf{vk}} := \mathcal{T}_2(x, \mathsf{vk})$ to $\mathcal{CH}$ and gets the replied key $\mathcal{SK}'_{x_{\mathsf{vk}}}$. Then $\mathscr{B}$ executes $K \longleftarrow \mathsf{Delegate}(\mathcal{PP}, \mathcal{SK}'_{x_{\mathsf{vk}}}, x_{\mathsf{vk}}, x_{\mathsf{vk}})$ and returns $\mathsf{Dec}'_K(\mathsf{C})$ to $\mathscr{A}$.

Challenge: Whenever $\mathscr{A}$ submits two challenge message indices $\psi^0, \psi^1 \in \Psi$ to $\mathscr{B}$, $\mathscr{B}$ submits two challenge indices $\psi^0_{\mathsf{vk}^*}, \psi^1_{\mathsf{vk}^*} \in \Psi'$, where $\psi^i_{\mathsf{vk}^*} := \mathcal{T}_3(\psi^i, \mathsf{vk}^*)$ for $i = 0, 1$ to $\mathcal{CH}$. Then, $\mathcal{CH}$ picks $b \xleftarrow{\mathrm{U}} \{0, 1\}$ and provides the challenge ciphertext $\mathsf{C}^* = \mathsf{Enc}'(\mathcal{PP}, \psi^b_{\mathsf{vk}^*})$ to $\mathscr{B}$. Now, $\mathscr{B}$ runs $\delta^* \longleftarrow \mathsf{OTS.Sign}(\mathsf{C}^*, \mathsf{signk}^*)$ and returns the challenge ciphertext $\mathsf{CT}^* := (\mathsf{C}^*, \delta^*, \mathsf{vk}^*)$ to $\mathscr{A}$.

Guess: $\mathscr{A}$ sends a guess b' to $\mathscr{B}$ and, then $\mathscr{B}$ returns the same guess b' to $\mathcal{CH}$.

Analysis: As the verification key has been chosen in the beginning of the game it is easy to define a forging algorithm which forges correctly against the one-time signature whenever $\mathsf{BAD}_{\mathrm{OTS}}$ sets true. So we may assume that $\mathsf{BAD}_{\mathrm{OTS}}$ does not set true throughout. In this case we show the following two things:

Claim-1 $\mathscr{B}$ follows the restriction of CPA-security game (as interacting with $\mathcal{CH}$) as long as $\mathscr{A}$ does so. In other words, $\mathscr{B}$ is correct given that $\mathscr{A}$ is correct.

Claim-2 Until $\mathscr{B}$ aborts (i.e., BAD_{OTS} occurs), all responses of $\mathscr{B}$ to $\mathscr{A}$ are identically distributed with the responses of a CCA-challenger $\mathcal{CH}_{CCA}$ to $\mathscr{A}$.

Assuming the above, we have

$$\mathsf{Adv}^{\mathrm{IND-CPA}}_{\mathscr{B},\mathrm{FE}'}(\kappa) \geq \mathsf{Adv}^{\mathrm{IND-CCA}}_{\mathscr{A},\mathrm{FE}}(\kappa) - \frac{1}{2}\mathsf{Adv}^{\mathrm{sUF-CMA}}_{\mathscr{A},\mathrm{OTS}}(\kappa)$$

which will conclude the theorem. Now we show the above two claims.

Proof of Claim-1. By the natural restriction on key queries by $\mathscr{A}$, we have for each queried key-index $x \in \mathcal{X}$

$$\mathcal{F}(x,\psi^0) = \mathcal{F}(x,\psi^1) \tag{2}$$

For each key query on index x' by $\mathscr{B}$, we have

$$\begin{aligned}\mathcal{F}'(x',\psi^0_{\mathsf{vk}^*}) &= \mathcal{F}(x,\psi^0) && \text{(by condition (2))}\\ &= \mathcal{F}(x,\psi^1) && \text{(by equation (2))}\\ &= \mathcal{F}'(x',\psi^1_{\mathsf{vk}^*}) && \text{(by condition (2))}\end{aligned}$$

which is required as a natural restriction on key queries by $\mathscr{B}$. To handle the decryption query of $\mathscr{A}$ for $(\mathsf{CT} := (\mathsf{C},\delta,\mathsf{vk}),x)$, $\mathscr{B}$ first makes a key query to $\mathcal{CH}$ for the key-index $x_{\mathsf{vk}} := \mathcal{T}_2(x,\mathsf{vk})$. Then randomizes the replied key $\mathcal{SK}'_{x_{\mathsf{vk}}}$ using delegation and then finally using this key, it decrypts the ciphertext. Since, $\mathsf{vk} \neq \mathsf{vk}^*$, we have

$$\begin{aligned}\mathcal{F}'(x_{\mathsf{vk}},\psi^0_{\mathsf{vk}^*}) &= \perp && \text{(by condition (3))}\\ &= \mathcal{F}'(x_{\mathsf{vk}},\psi^1_{\mathsf{vk}^*}) && \text{(by condition (3))}\end{aligned}$$

which is again a requirement for key queries by $\mathscr{B}$.

Proof of Claim-2. This is more or less straightforward from the restricted delegation property.

□

Conditions for Predicate Encryption: If we restrict the delegation-friendly index-transformer to the class of PE, then the maps, $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_3$ satisfy the following: for all $x \in \mathcal{X}$, $\mathsf{vk} \neq \mathsf{vk}' \in \{0,1\}^n$ and $y \in \mathcal{Y}$ (note that they differ in condition 2).

- (Public index). (1) $x \sim y \iff x_{\mathsf{vk}} \sim' y_{\mathsf{vk}}$, (2) $x \not\sim y \implies x' \not\sim' y_{\mathsf{vk}}$ and (3) $x_{\mathsf{vk}} \not\sim' y_{\mathsf{vk}'}$
- (Hidden index). (1) $x \sim y \iff x_{\mathsf{vk}} \sim' y_{\mathsf{vk}}$, (2) $x \sim y \iff x' \not\sim' y_{\mathsf{vk}}$ and (3) $x_{\mathsf{vk}} \not\sim' y_{\mathsf{vk}'}$

A construction using MAC, weak commitment and restricted delegation will be found in Appendix D.

3.3 Using (restricted)-Verifiability Feature

In this section, we construct a generic CCA-secure predicate encryption scheme $\mathsf{PE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ from a CPA-secure predicate encryption $\mathsf{PE}' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$. Let $\sim$ (resp. $\sim'$) be a predicate relation between $\mathcal{X}$ (resp. $\mathcal{X}'$) and $\mathcal{Y}$ (resp. $\mathcal{Y}'$). Let $\mathsf{OTS} = (\mathsf{OTS.Gen}, \mathsf{OTS.Sign}, \mathsf{OTS.Ver})$ be a one-time signature (OTS) scheme. Like in the case of delegation, we define an index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$, called **verifiability-friendly index-transformer**.

Definition 6. *A triple of maps $\mathcal{T}_1 : \mathcal{X} \longrightarrow \mathcal{X}'$, $\mathcal{T}_2 : \mathcal{Y} \times \{0,1\}^n \longrightarrow \mathcal{X}'$ and $\mathcal{T}_3 : \mathcal{Y} \times \{0,1\}^n \longrightarrow \mathcal{Y}'$ is called* **verifiability-friendly index-transformer** *from $(\sim, \mathcal{X}, \mathcal{Y})$ to $(\sim', \mathcal{X}', \mathcal{Y}')$ if the following conditions[6] (given in box) are satisfied for all $x \in \mathcal{X}$, $\mathsf{vk} \neq \mathsf{vk}' \in \{0,1\}^n$ and $y \in \mathcal{Y}$.*

$$\text{(1) } x \sim y \iff x' \sim' y_{\mathsf{vk}}, \text{ (2) } \epsilon^y_{\mathsf{vk}} \sim' y_{\mathsf{vk}}, \text{ and (3) } \epsilon^y_{\mathsf{vk}} \not\sim' y_{\mathsf{vk}'}$$

where we simply denote $\mathcal{T}_1(x)$, $\mathcal{T}_2(y, \mathsf{vk})$ and $\mathcal{T}_3(y, \mathsf{vk})$ by $x', \epsilon^y_{\mathsf{vk}}$ and y_{vk} respectively.

Definition 7 (Restricted Verifiability for PE). *An algorithm* Verify *is said to be a restricted-verifiable algorithm for a predicate encryption scheme* PE' *with public index w.r.t. a index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ if it is (1) weak format-verifier and (2) for all ciphertext* C *(possibly ill-format) with the public associated index y, and all $x' := \mathcal{T}_1(x), \epsilon^y_{\mathsf{vk}} := \mathcal{T}_2(y, \mathsf{vk})$ with $x \sim y$ we have*

$$\mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x', \epsilon^y_{\mathsf{vk}}) = 1 \Rightarrow \mathsf{Dec}'(\mathcal{PP}, \mathsf{C}, \mathcal{SK}'_{x'}) = \mathsf{Dec}'(\mathcal{PP}, \mathsf{C}, \mathcal{SK}'_{\epsilon^y_{\mathsf{vk}}}) \quad (3)$$

We also say that PE' *is said to have the restricted-verifiability. We note that the restricted verifiability is a weaker notion than the actual verifiability.*

A Generic Construction based on Restricted Verifiability. We describe the CCA-secure PE from a CPA-secure PE' with restricted verifiability Verify and a verifiability-friendly index transformer. Let $\mathsf{OTS} = (\mathsf{OTS.Gen}, \mathsf{OTS.Sign}, \mathsf{OTS.Ver})$ be a one-time signature scheme. The setup algorithm is same as before (with a possibly small modification in system parameter as mentioned before) and it returns $(\mathcal{PP}, \mathcal{MSK})$. The other three algorithms (implicitly understood $\mathcal{PP}, \mathcal{MSK}$) are described below (key generation and encryption algorithms are same as the case of delegation):

[6] We would refer as verifiability-friendly conditions.

Conversion based on Verifiability and using One-Time-Signature

- $\mathsf{KeyGen}(x) := \mathsf{KeyGen}'(x')$ (in notation: $\mathcal{SK}_x := \mathcal{SK}'_{x'}$).
- $\mathsf{Enc}(m, y)$: It runs $(\mathsf{vk}, \mathsf{signk}) \longleftarrow \mathsf{OTS.Gen}(1^\kappa)$ and returns

$$\mathsf{CT} = (\mathsf{C} := \mathsf{Enc}'(m, y_{\mathsf{vk}}),\ \delta := \mathsf{OTS.Sign}(\mathsf{C}, \mathsf{signk}),\ y,\ \mathsf{vk}).$$

- $$\mathsf{Dec}_{\mathcal{SK}_x}(\mathsf{C}, \delta, y, \mathsf{vk}) = \begin{cases} \mathsf{Dec}'_{\mathcal{SK}'_{x'}}(\mathsf{C}) & \text{if } \begin{pmatrix} \mathsf{OTS.Ver}(\mathsf{C}, \delta, \mathsf{vk}) = 1, x \sim y, \\ \mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x', \epsilon^y_{\mathsf{vk}}) = 1 \end{pmatrix} \\ \perp & \text{otherwise.} \end{cases}$$

Correctness : Let $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, j)$ and for all $x \in \mathcal{X}, y \in \mathcal{Y}, m \in \mathcal{M}$, let $\mathcal{SK}_x \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$ and $\mathsf{CT} \longleftarrow \mathsf{Enc}_{\mathcal{PP}}(m, y)$. Then, clearly from the definition of decryption, we have $\mathsf{Dec}_{\mathcal{SK}_x}(\mathsf{CT}) = \perp$ whenever $x \not\sim y$. Suppose $x \sim y$ then,

$$\begin{aligned} \mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x', \epsilon^y_{\mathsf{vk}}) = 1 & \quad \text{(by conditions (1), (2) and weak format-verifier)} \\ \mathsf{Dec}(\mathsf{CT}, \mathcal{SK}_x) = \mathsf{Dec}'(\mathsf{C}, \mathcal{SK}'_{x'}) & \quad \text{(Since, } \mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x', \epsilon^y_{\mathsf{vk}}) = 1\text{)} \\ = m & \quad \text{(by correctness of } \mathsf{PE}' \text{ and the condition (1))} \end{aligned}$$

Theorem 2. *Let $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ be a verifiability-friendly index-transformer, PE' be an IND-CPA secure predicate encryption scheme with restricted verifiability and OTS be a strong unforgeable one-time signature scheme, then the above proposed scheme PE in section 3.3 is an IND-CCA secure predicate encryption scheme.*

Proof. For proof, we refer to Appendix F.

A construction using MAC, weak commitment and restricted verifiability will be found in Appendix E.

4 Instantiations of Delegation/Verifiability-friendly Index-Transformer

Now, we are just about to instantiate the various instantiations of index-transformers, $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ for both delegation-friendly and verifiability-friendly functional encryption classes, viz., predicate encryption classes. The candidate classes are traditional ABE, FE for regular languages, (D)SE, ABE for circuits, (H)IPE and HVE. For all the instantiations of the index-transformers $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ described in the respective subsections and Table 1, formally we have the following theorem (for proof, one can easily check delegation/verifiability-friendly conditions):

Theorem 3. *The index-transformers, $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ given in Table 1 for all the aforementioned classes satisfy either the delegation-friendly conditions (with both, hidden index and public index) or the verifiability-friendly conditions.*

All the given instantiations of the index-transformers, $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ preserve the efficiency of the respective indices.

Table 1. Instantiations of different functional encryption systems with either delegation or verifiability. The notations, tABE and cABE respectively stand for traditional ABE and ABE for circuits. Del (**D**) and Verf (**V**) respectively denote the delegation and verifiability. In 5th column of the table, the expressions, $x_{\mathsf{vk}} := \mathcal{T}_2(x, \mathsf{vk})$ and $\epsilon^{y}_{\mathsf{vk}} := \mathcal{T}_2(y, \mathsf{vk})$ are appeared respectively for the delegation-based and verifiability-based conversions.

Class	x	y	$x' := \mathcal{T}_1(x)$	$x_{\mathsf{vk}}/\epsilon^{y}_{\mathsf{vk}}$	$y_{\mathsf{vk}} := \mathcal{T}_3(y, \mathsf{vk})$	**D/V**
tABE †	Γ	A	Γ	$\Gamma \wedge (\wedge_{P \in S_{\mathsf{vk}}} P)$	$A \cup S_{\mathsf{vk}}$	Del
tABE †	A	Γ	$A \cup \mathcal{W}$	$A \cup S_{\mathsf{vk}}$	$\Gamma \wedge (\wedge_{P \in S_{\mathsf{vk}}} P)$	Del
tABE †	Γ	A	Γ	$\wedge_{P \in S_{\mathsf{vk}}} P$	$A \cup S_{\mathsf{vk}}$	Verf
tABE †	A	Γ	A	S_{vk}	$\Gamma \vee (\wedge_{P \in S_{\mathsf{vk}}} P)$	Verf
tABE ‡	$B_1, \ldots, B_\ell$	A	$B_1, \ldots, B_\ell$	$B_1 \cup S_{\mathsf{vk}}, \ldots, B_\ell \cup S_{\mathsf{vk}}$	$A \cup S_{\mathsf{vk}}$	Del
tABE ‡	A	$B_1, \ldots, B_\ell$	$A \cup \mathcal{W}$	$A \cup S_{\mathsf{vk}}$	$B_1 \cup S_{\mathsf{vk}}, \ldots, B_\ell \cup S_{\mathsf{vk}}$	Del
tABE ‡	$B_1, \ldots, B_\ell$	A	$B_1, \ldots, B_\ell$	S_{vk}	$A \cup S_{\mathsf{vk}}$	Verf
tABE ‡	A	$B_1, \ldots, B_\ell$	A	S_{vk}	$B_1, \ldots, B_\ell, S_{\mathsf{vk}}$	Verf
FE for DFAs	$M := (Q, \Sigma, \mathcal{T}, q_0, F)$	w	$M' := (Q', \Sigma', \mathcal{T}', q_0', F)$	$M_{\mathsf{vk}} := (Q', \Sigma', \mathcal{T}_{\mathsf{vk}}, q_0', F)$	$\mathsf{vk}\|\|w$	Both
FE for DFAs	w	$M := (Q, \Sigma, \mathcal{T}, q_0, F)$	$*^n\|\|w$	$\mathsf{vk}\|\|w$	$M_{\mathsf{vk}} := (Q', \Sigma', \mathcal{T}_{\mathsf{vk}}, q_0', F)$	Del
FE for DFAs	w	$M := (Q, \Sigma, \mathcal{T}, q_0, F)$	w	$\$_{\mathsf{vk}[1]} \cdots \$_{\mathsf{vk}[n]}$	$M_{\mathsf{vk}} := (Q', \Sigma', \mathcal{T}_{\mathsf{vk}}, q_0, F')$	Verf
(D)SE	$\mathsf{Aff}(\mathsf{X}, \boldsymbol{x})$	$\mathsf{Aff}(\mathsf{Y}, \boldsymbol{y})$	$\mathsf{Aff}\Big(\left[\begin{array}{c\|c} \mathsf{X} & \mathsf{U} \\ \hline \mathsf{V}_x & 1 \end{array}\right], \left(\begin{smallmatrix} \boldsymbol{x} \\ 0 \end{smallmatrix}\right)\Big)$	$\mathsf{Aff}\Big(\left[\begin{array}{c\|c} \mathsf{Z} & \mathsf{U} \\ \hline \mathsf{V} & 0 \end{array}\right], \left(\begin{smallmatrix} \boldsymbol{z} \\ \mathsf{vk} \end{smallmatrix}\right)\Big)$	$\mathsf{Aff}\Big(\left[\begin{array}{c\|c} \mathsf{Y} & \mathsf{U} \\ \hline \mathsf{V}_y & 0 \end{array}\right], \left(\begin{smallmatrix} \boldsymbol{y} \\ \mathsf{vk} \end{smallmatrix}\right)\Big)$	Both
More generic (D)SE	$\mathsf{Ker}(\mathsf{X})$	$\mathsf{Aff}(\mathsf{Y}, \boldsymbol{y})$	$\mathsf{Ker}\Big(\left[\begin{array}{c\|c} \mathsf{X} & \mathsf{U} \\ \hline \mathsf{V}_x & \mathsf{O} \end{array}\right]\Big)$	$\mathsf{Ker}\Big(\left[\begin{array}{c\|c} \mathsf{Z} & \mathsf{U} \\ \hline \mathsf{V} & \mathsf{vk}^\perp \end{array}\right]\Big)$	$\mathsf{Aff}\Big(\left[\begin{array}{c\|c} \mathsf{Y} & \mathsf{U} \\ \hline \mathsf{V}_y & \mathsf{O} \end{array}\right], \left(\begin{smallmatrix} \boldsymbol{y} \\ \mathsf{vk} \end{smallmatrix}\right)\Big)$	Both
cABE	Ω	ω	Ω	Ω_{vk}	$\omega\|\|\mathsf{vk}$	Del
cABE	Ω	ω	Ω	Ω_{vk}	$\omega\|\|\mathsf{vk}$	Verf
cABE	ω	Ω	$\omega\|\|\underbrace{0 \cdots 0}_{n}$	$\omega\|\|\mathsf{vk}$	Ω_{vk}	Verf
IPE	$\boldsymbol{x}$	$\boldsymbol{y}$	$\boldsymbol{x}$	$(\boldsymbol{x}, \mathsf{vk}^\perp)$	$(\boldsymbol{y}, \mathsf{vk})$	Del
IPE	$(x_1, \ldots, x_\ell)$	$(y_1, \ldots, y_\ell)$	$(x_1, \ldots, x_\ell, 0, 0)$	$(y_1^\perp, \ldots, y_\ell^\perp, \mathsf{vk}, -1)$	$(y_1, \ldots, y_\ell, 1, \mathsf{vk})$	Verf
HIPE	$(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$	$(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_h)$	$(\boldsymbol{0}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$	$(\mathsf{vk}^\perp, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$	$(\mathsf{vk}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_h)$	Del
HIPE	$(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$	$(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_h)$	$(\boldsymbol{0}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$	$(\mathsf{vk}^\perp, \boldsymbol{y}_1^\perp, \ldots, \boldsymbol{y}_\ell^\perp)$	$(\mathsf{vk}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_h)$	Verf
HVE	x	y	$x\|\|*^n$	$x\|\|\mathsf{vk}$	$y\|\|\mathsf{vk}$	Del

4.1 Instantiation of Traditional ABE

We show that the constructions of Yamada et al. [38] is a special case of ours, i.e., our abstraction for functional encryption with predicate functionality unifies the generic CCA conversion for ABE [38]. Let $\mathcal{U}$ be the attribute universe for the target ABE. Then, the universe for primitive ABE' is $\mathcal{U}' := \mathcal{U} \cup \mathcal{W}$, where $\mathcal{W} = \{P_{1,0}, P_{1,1}, \ldots, P_{n,0}, P_{n,1}\}$ or $\{0,1\}^n$ according to $\mathcal{U}$ is small or large. For a $\mathsf{vk} \in \{0,1\}^n$, we define $S_{\mathsf{vk}} := \{P_{1,\mathsf{vk}[1]}, P_{2,\mathsf{vk}[2]}, \ldots, P_{n,\mathsf{vk}[n]}\}$ or $\{\mathsf{vk}\}$ according to $\mathcal{U}$ is small or large. In Table 1, we give a brief instantiation of index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ for both delegation-friendly and verifiability-friendly traditional ABE (tABE), where Γ and A stand for policy and set of attributes. The classes, tABE followed by † are the instantiations of index-transformer for tABE, where

boolean formulas are represented by either monotone access trees or monotone span programs.

– (Minimal set representation). The approach of Yamada et al. has not handled the ABE, where the policies are represented by the sets of minimal sets. We show the instantiations of delegation/verifiability-friendly index-transformer for ABE, where the policies are described by the sets of minimal sets. We use the notation, $(B_1, \ldots, B_\ell)$ for the access formula. In the Table 1, the classes, tABE followed by ‡ are the instantiations of index-transformer for tABE with minimal set representation.

4.2 Instantiation for Regular Languages

Let Σ be the alphabet for the target functional encryption FE for regular languages. We represent the regular languages by the DFAs. Let $M = (Q, \Sigma, \mathcal{T}, q_0, F)$ and w respectively be a DFA and a string over Σ. A brief instantiation of index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ for both delegation-friendly and verifiability-friendly are given in Table 1. A clear pictorial representation of the index-transformers will be found Appendix H. Some of the technicalities are described case by case:

– (KP-FE for regular languages). W.L.G, we assume that $0, 1 \in \Sigma$ and set the alphabet for the hired encryption scheme FE' to be $\Sigma' := \Sigma$. The transformed key-index under $\mathcal{T}_1$ is $M' = (Q', \Sigma', \mathcal{T}', q'_0, F)$ where $Q' = Q \cup \{q'_0, \ldots, q'_{n-1}\}$ with $q'_0, \ldots, q'_{n-1} \notin Q$ and $\mathcal{T}' = \mathcal{T} \cup \{(q'_{i-1}, q'_i, 1-j) : j = 0, 1;\ i = 1, \ldots, n\}$ with $q'_n = q_0$. The transformed key-index under[7] $\mathcal{T}_2$ is $M_{\mathsf{vk}} = (Q', \Sigma', \mathcal{T}_{\mathsf{vk}}, q'_0, F)$ which is same as M' except $\mathcal{T}_{\mathsf{vk}} = \mathcal{T} \cup \{(q'_{i-1}, q'_i, \mathsf{vk}[i]) : i = 1, \ldots, n\}$.
– (CP-FE for regular languages using delegation). W.L.G, we assume that $0, 1 \in \Sigma$ and set $\Sigma' := \Sigma \cup \{*\}$. Like HVE, we introduce the "wild card" $(*)$ entry in the key-indices (strings) to represent both the symbols 0,1. *First of all, note that CP-FE for regular languages never supports the actual delegation.* With this wild card entry, we can have a restricted partial order of the form, $*^n||w \succ \mathsf{vk}||w$ (i.e., $w' \succ w_{\mathsf{vk}}$) for all $w \in \Sigma^*$ and $\mathsf{vk} \in \{0,1\}^n$. The description of M_{vk} is same as above.
– (CP-FE for regular languages using verifiability). We set $\Sigma' := \Sigma \cup \{\$_0, \$_1\}$, where $\Sigma \cap \{\$_0, \$_1\} = \emptyset$. The transformed associated index under $\mathcal{T}_3$ is $M_{\mathsf{vk}} = (Q', \Sigma', \mathcal{T}_{\mathsf{vk}}, q_0, F')$ where $Q' = Q \cup \{q'_1, \ldots, q'_n\}$ with $q'_1, \ldots, q'_n \notin Q$ and $\mathcal{T}_{\mathsf{vk}} = \mathcal{T} \cup \{(q'_{i-1}, q'_i, \$_{\mathsf{vk}[i]}) : i = 1, \ldots, n\}$ with $q'_0 = q_0$.

4.3 Instantiation for (Doubly-)Spatial Encryption

In (doubly-)spatial encryption, the delegation algorithm is considered to be a mandatory feature to capture the different functional encryptions. Therefore all

[7] we consider both the cases, based on delegation and verifiability together. Obviously the part $x_{\mathsf{vk}} := M_{\mathsf{vk}}$ is clear. The part, ϵ^w_{vk} is defined to be a DFA that accepts $\mathsf{vk}||w$. Since M accepts w, therefore M_{vk} plays the role of ϵ^w_{vk} as well.

the (D-)SE schemes [7, 39, 21, 26, 12, 13] are eligible for CCA conversion w.r.t the delegation-friendly index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$. But the only schemes that are subject to CCA conversion w.r.t the verifiability-friendly index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ are [7, 39, 21].

Let $\mathcal{V}(= \mathbb{Z}_q^\ell)$ be an ℓ-dimensional affine space over the field $\mathbb{F}(= \mathbb{Z}_q)$ for a target (D)SE. Then, the transformed $(\ell+1)$-dimensional affine space is $\mathcal{V}' := \{\binom{\boldsymbol{v}}{r} : \boldsymbol{v} \in \mathcal{V},\ r \in \mathbb{F}\}$. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}_q^\ell, \mathsf{X} \in \mathbb{Z}_q^{\ell \times \vartheta}$ with $\vartheta \leq \ell$ and $\mathsf{Y} \in \mathbb{Z}_q^{\ell \times \varrho}$ with $\varrho \leq \ell$. Let U, V_x and V_y be respectively the $\ell \times 1$, $1 \times \vartheta$ and $1 \times \varrho$ null matrices. In Table 1, we show the index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ by using block matrix representation. Some notations found in Table 1 are described case by case.

- ((D)SE). In this case, both the key-indices and the associated indices are described by the affine subspaces of $\mathbb{Z}_q^\ell$, i.e., $x := \mathsf{Aff}(\mathsf{X}, \boldsymbol{x})$ and $y := \mathsf{Aff}(\mathsf{Y}, \boldsymbol{y})$. We define $\mathsf{Z} := \mathsf{X}$, $\mathsf{V} := \mathsf{V}_x$ and $\boldsymbol{z} := \boldsymbol{x}$ if $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ is delegation-friendly else, $\mathsf{Z} := \mathsf{Y}$, $\mathsf{V} := \mathsf{V}_y$ and $\boldsymbol{z} := \boldsymbol{y}$.
- (More generic (D)SE). Recently, Chen and Wee [13] defined (doubly) spatial encryption in more generally, where the key-indices x are represented by $\mathsf{Ker}(\mathsf{X})$ and the associated indices are same as before. The authors [13] showed that the (D)SE in former form can be easily derived from this general form of (D)SE. Let O, V_x, V_y and N be respectively the 2×1, $2 \times \vartheta$, $2 \times \varrho$ and $\ell \times \vartheta$ null matrices. We define the blocks to be $\mathsf{Z} := \mathsf{X}$ and $\mathsf{V} := \mathsf{V}_x$ if $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ is delegation-friendly else, $\mathsf{Z} := \mathsf{N}$ and $\mathsf{V} = \mathsf{V}_y$

4.4 Instantiation of ABE for Circuits

Let Ω be a circuit of depth at most ℓ over k boolean variables. Let ω be a boolean assignment over these k variables. Let $\tilde{\Omega}_{\mathsf{vk}}$ be a circuit of depth at most ℓ satisfied exactly by $\omega || \mathsf{vk}$ for all $\omega \in \{0,1\}^k$. Suppose the boolean variables appeared in $\Omega \in \mathcal{X}$ (resp. $\Omega \in \mathcal{Y}$) are indexed from 1 to k. We assume that when the circuit $\Omega \in \mathcal{X}$ (resp. $\Omega \in \mathcal{Y}$) is considered as a part of $\mathcal{X}'$ (resp. $\Omega \in \mathcal{Y}'$), the same indices of the variables are preserved here. In later case, $\Omega \in \mathcal{X}'$ (resp. $\Omega \in \mathcal{Y}'$) can be considered over $(k+n)$ variables, but note that the remaining variables from the index $(k+1)$ to $(k+n)$ are not present. In Table 1, we show briefly the instantiations of index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ for both delegation-friendly and verifiability-friendly. Both the systems, KP-ABE and CP-ABE [17] for circuits are eligible for CCA conversion w.r.t the verifiability-friendly index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$. Some of the technicalities found in Table 1 are dealt case by case.

- (KP-ABE for circuits using delegation). The transformed key space $\mathcal{X}'$ is the set of all circuits of depth at most $(\ell+1)$ over $(k+n)$ boolean variables. The transformed circuit, Ω_{vk} (under $\mathcal{T}_2$) is obtained by joining two circuits, Ω and $\tilde{\Omega}_{\mathsf{vk}}$ as the child circuits to an 'AND'-gate.

– (KP-ABE for circuits using verifiability). The transformed key space $\mathcal{X}'$ is the set of all circuits of depth at most ℓ over $(k+n)$ boolean variables. Here, the transformed circuit, Ω_{vk} (under $\mathcal{T}_2$) is considered to be a circuit of depth at most ℓ satisfied exactly by the assignment $\omega||\mathsf{vk}$.
– (CP-ABE for circuits using verifiability). The transformed associated data space $\mathcal{Y}'$ is the set of all circuits (*excluding tautology*) of depth at most $(\ell+1)$ over $(k+n)$ boolean variables. Here, we assume that vk must not be equal to the zero string. The transformed circuit, Ω_{vk} (under $\mathcal{T}_3$) is obtained by joining two circuits, Ω and $\tilde{\Omega}_{\mathsf{vk}}$ as the child circuits to an 'OR'-gate.

4.5 Instantiation of (H)IPE

For a HIPE, the description of the system parameters are $j := (\ell, d; \mu_1, \ldots, \mu_d)$, with $\mu_0 = 0 < \mu_1 < \mu_2 < \cdots < \mu_d = \ell$, where d is called the depth of the hierarchy and ℓ is the maximum length of the vectors. For $i = 1, \ldots, d$, let $\Sigma_i := \mathbb{Z}_q^{\mu_i - \mu_{i-1}} \setminus \{\mathbf{0}\}$ be the i^{th} level attribute space. Then, the hierarchical attribute space is defined by $\Sigma := \cup_{i=1}^{d}(\Sigma_1 \times \cdots \times \Sigma_i)$. We set both $\mathcal{X}$ and $\mathcal{Y}$ are to be Σ. For a key-index $x = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k)$ and an associated index $y = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_h)$, where $\boldsymbol{x}_i, \boldsymbol{y}_i \in \Sigma_i$, we define $x \sim y$ if and only if $k \le h$ and $\boldsymbol{x}_i.\boldsymbol{y}_i = 0$ for all i such that $1 \le i \le k$. For, this section only we consider the all the vectors to be row vectors.

Although both the cases for non-HIPE and HIPE could be handled in one framework (as non-HIPE is a special case of HIPE), for simplicity, we deal them separately.

– (Non-HIPE using delegation). A non-HIPE scheme can always be considered as a HIPE with depth of hierarchy $d = 1$. For the CCA conversion using delegation, our primitive IPE must have to be hierarchical of depth 2. In this HIPE, the vectors in the first level are the actual attributes for IPE and the vectors (of dimension 2) in 2nd level are used to embed the verification keys vk. The examples of such schemes which qualify the criteria based on delegation are [24, 30].
– (Non-HIPE using verifiability). To handle the CCA conversion using verifiability, we no longer consider the primitive IPE scheme to be hierarchical, rather we extend the dimension. For $\boldsymbol{y} \in \mathbb{Z}_q^{\ell}$, we define $\boldsymbol{y}^{\perp} = (y_1^{\perp}, \ldots, y_{\ell}^{\perp}) \in \{\boldsymbol{v} \in \mathbb{Z}_q^{\ell} : \boldsymbol{v}.\boldsymbol{y} = 0\}$ and $\mathbf{0} \in \mathbb{Z}_q^{\ell}$ could be a choice for $\boldsymbol{y}^{\perp}$. Rest of the part will be understood from Table 1. One of the qualified candidates for this conversion is IPE scheme in section 4.1 of [3].
– (HIPE using delegation). In case of HIPE, the vectors in every levels are the actual attributes for the HIPE. Therefore, we open a new attribute space, consisting the vectors of dimension two to embed the verification keys vk. This space is now reserved for first level of the hierarchy and the levels of all the actual attributes spaces are shifted (increased) by one. So, the delegation required for CCA conversion will take place at first level of the hierarchy. The description of the system parameters are $j' := (\ell', d'; \mu_1, \ldots, \mu_{d'})$ where

$\ell' = \ell+2, d' = d+1, \mu'_1 = \mu_0+2 = 2, \mu'_2 = \mu_1+2, \ldots, \mu'_{i+1} = \mu_i+2, \ldots, \mu'_{d'} = \mu_d+2 = \ell'$ and $\mu'_0 = 0 < \mu'_1 < \cdots < \mu'_{d'} = \ell'$. Here we consider the first level attribute space Σ'_1 to include $\mathbf{0}$, i.e., $\Sigma'_1 = \mathbb{Z}_q^{\mu'_1-\mu'_0}$. Let $\Sigma'_i := \mathbb{Z}_q^{\mu'_i-\mu'_{i-1}} \setminus \{\mathbf{0}\}$ for $2 \leq i \leq d'$. Let $\Sigma' := \cup_{i=1}^{d'}(\Sigma'_1 \times \cdots \times \Sigma'_i)$. Note that $\Sigma'_i = \Sigma_{i-1}$ for $2 \leq i \leq d'$. We set $\mathcal{X}' = \mathcal{Y}' = \Sigma'$. There are many HIPE schemes, e.g., [27, 24, 30], which are eligible for CCA conversion based on delegation.

– (HIPE using verifiability). $\boldsymbol{y}_i^{\perp} \in \{\boldsymbol{v}_i \in \Sigma_i : \ \boldsymbol{v}_i.\boldsymbol{y}_i = 0\}$ for $1 \leq i \leq k$. One of the qualified candidates for this conversion is HIPE scheme in section 10 of [29].

4.6 Instantiation of HVE

The hidden vector encryption [22, 10] is a PE with hidden index, but is a special case of IPE as found in [23]. A brief instantiation of delegation-friendly index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ is given in Table 1, where we assume that $\Sigma = \{0, 1\}$. Both the lengths of transformed key-indices and the associated indices are extended from ℓ to $(\ell + n)$. For this instantiation, we could not find any example suitable for the CCA conversion.

5 Candidate Schemes Favorable to Our CCA Conversion

We list up the various predicate encryption schemes that are favorable to our CCA-conversion. Only the schemes for regular languages are described here as they support the restricted delegation w.r.t $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ defined in the Table 1, instead of actual delegation. A tabular representation of the candidate schemes are found in Table 2.

KP-FE for regular languages. So far we know, there are only two IND-CPA secure KP-FE schemes [37, 1] for regular languages, but none of them satisfies the actual delegation. Here we mainly investigate the KP-FE scheme of Waters [37] for regular languages as the scheme of Attrapadung [1] has the structural similarity with [37]. We first describe KP-FE scheme of Waters briefly.

Let $(p, g, e, \mathbb{G}, \mathbb{G}_T)$ be a bilinear group descriptor, where $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is a bilinear map, $|\mathbb{G}| = |\mathbb{G}_T| = p$ (prime) and $\mathbb{G} :=< g >$. Let $\mathcal{PP} := [e(g, g)^{\alpha}, g, z, h_{start}, h_{end}, h_{\sigma} \forall \sigma \in \Sigma]$ and $\mathcal{MSK} := [g^{-\alpha}]$, where $\alpha \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ and $z, h_{start}, h_{end}, (h_{\sigma})_{\sigma\in\Sigma} \xleftarrow{\mathrm{U}} \mathbb{G}$.

Secret Key. The distribution of the key is $\mathcal{SK}_M := [M, K_{start,1} := D_0(h_{start})^{r_{start}}, K_{start,2} := g^{r_{start}}, \ (K_{t,1} := {D_x}^{-1} z^{r_t}, \ K_{t,2} := g^{r_t}, \ K_{t,3} := D_y(h_{\sigma})^{r_t})_{t\in\mathcal{T}}, (K_{endx,1} := g^{-\alpha}.D_x(h_{end})^{r_{endx}}, \ K_{endx,2} := g^{r_{endx}})_{q_x\in F}]$, where $M := (Q, \Sigma, \mathcal{T}, q_0, F)$, $Q := \{q_0, q_1, \ldots, q_{|Q|-1}\}$, $D_i \xleftarrow{\mathrm{U}} \mathbb{G}$ for each $q_i \in Q$, $r_{start}, r_{endx}, (r_t)_{t\in\mathcal{T}} \xleftarrow{\mathrm{U}} \mathbb{Z}_p$.

CIPHERTEXT. The distribution of the ciphertext is $\mathsf{C}_w := [w, \mathsf{C}_m := m.e(g,g)^{\alpha.s_\ell},$
$\mathsf{C}_{start,1} := \mathsf{C}_{0,1} = g^{s_0},\ \mathsf{C}_{start,2} := (h_{start})^{s_0},\ (\mathsf{C}_{i,1} := g^{s_i},\ \mathsf{C}_{i,2} := (h_{w_i})^{s_i} z^{s_{i-1}})_{i\in[\ell]},$
$\mathsf{C}_{end,1} := \mathsf{C}_{\ell,1} = g^{s}\ell,\ \mathsf{C}_{end,2} := (h_{end})^{s}\ell]$, where $s_0, s_1, \ldots, s_\ell \xleftarrow{\text{U}} \mathbb{Z}_p$.

One can easily check that the KeyGen satisfies the re-randomization. Since all the transitions are used to label the secret key components, therefore the encryption scheme of Waters [37] (similarly the scheme of Attrapadung [1]) satisfies the restricted delegation by the following theorem:

Theorem 4 (Generic restricted delegation in KP flavor). *If the* KeyGen *algorithm for the primitive scheme* FE' *for regular languages supports the re-randomization and the transitions are involved to label the key components for* M'*, then the functional encryption scheme,* FE' *for regular languages satisfies the restricted delegation w.r.t the index-transformer* $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ *defined in section 4.*

Proof. We have to show the delegation for the restricted partial order w.r.t the index-transformer $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$, i.e., $M' \succ M_{\mathsf{vk}}$. The description of M' and M_{vk} are same except $\mathcal{T}_{\mathsf{vk}} \subset \mathcal{T}'$. Let $\mathcal{SK}_{M'} := (\mathcal{SK}_{\mathcal{T}'}, \mathcal{SK}_H)$, where the key components of $\mathcal{SK}_{\mathcal{T}'}$ are labeled by the transitions of $\mathcal{T}'$ and the rest is $\mathcal{SK}_H$. Now, the description of the restricted key for M_{vk} is $\mathcal{SK}_{M_{\mathsf{vk}}} := (\mathcal{SK}_{\mathcal{T}_{\mathsf{vk}}}, \mathcal{SK}_H)$, where $\mathcal{SK}_{\mathcal{T}_{\mathsf{vk}}}$ is formed by keeping those components of $\mathcal{SK}_{\mathcal{T}'}$ which are labeled by the transitions of $\mathcal{T}_{\mathsf{vk}}$. Finally, the delegation key for $\mathcal{T}_{\mathsf{vk}}$ is obtained by applying re-randomization on $\mathcal{SK}_{M_{\mathsf{vk}}}$.

Again it is straightforward to check that the scheme [37] (similarly [1]) satisfies verifiability.

CP-FE for regular languages. Using the restricted partial order as a weapon defined in section 4.2, we observe a *generic restricted delegation* given below :

Observation. *If the* KeyGen *algorithm for the primitive scheme* FE' *supports the re-randomization and for each* $i \in [n]$*, all the key components for* i^{th} *entry in the key-indices of the form,* $*^n||w$*, are labeled with 0 and 1, then the* FE' *scheme for regular languages satisfies the restricted delegation w.r.t index-transformer* $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ *defined in section 4.2.*

Proof. Similar to Theorem 4.

The only available CP-FE scheme [1] for regular languages is not known to satisfy the above restricted delegation. However, the verifiability is satisfied by Attrapadung's scheme [1].

6 Conclusion and Future Works

In this paper, we have proposed a generic CCA conversion for the CPA-secure functional encryption that unifies all the existing results. We keep open the choice

for all possible instantiations that would respect our delegation/verifiability-friendly index-transformer. Some of the task is still pending, like, we do not know any delegation-friendly index-transformer of CP-ABE for circuits. Eventually, we could not find any example that fits with delegation-friendly index-transformer either for CP-FE for regular languages or KP-ABE for circuits. In future, we would be interested to see the complete picture of the CCA conversion.

References

1. Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *EUROCRYPT*, volume 8441 of *LNCS*, pages 557–577. Springer, 2014.
2. Nuttapong Attrapadung. Fully secure and succinct attribute based encryption for circuits from multi-linear maps. Cryptology ePrint Archive, Report 2014/772, 2014. `http://eprint.iacr.org/`.
3. Nuttapong Attrapadung and Benoît Libert. Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation. In *PKC*, volume 6056 of *LNCS*, pages 384–402. Springer, 2010.
4. Nuttapong Attrapadung, Benoît Libert, and Elie de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *Public Key Cryptography*, volume 6571 of *LNCS*, pages 90–108. Springer, 2011.
5. John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Press, 2007.
6. Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.
7. Dan Boneh and Mike Hamburg. Generalized identity-based and broadcast encryption schemes. In *ASIACRYPT*, volume 5350 of *LNCS*, pages 455–470. Springer, 2008.
8. Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In *CT-RSA*, volume 3376 of *LNCS*, pages 87–103. Springer, 2005.
9. Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *TCC*, volume 6597 of *LNCS*, pages 253–273. Springer, 2011.
10. Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, LNCS, pages 535–554. Springer, 2007.
11. Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT*, volume 3027 of *LNCS*. Springer, 2004.
12. Cheng Chen, Zhenfeng Zhang, and Dengguo Feng. Fully secure doubly-spatial encryption under simple assumptions. In *PROVSEC*, volume 7496 of *LNCS*, pages 253–263. Springer, 2012.
13. Jie Chen and Hoeteck Wee. Doubly spatial encryption from dbdh. Cryptology ePrint Archive, Report 2014/199, 2014. `http://eprint.iacr.org/`.
14. Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *Cryptography and Coding*, volume 2260 of *LNCS*, pages 360–363. Springer, 2001.
15. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

16. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO*, volume 1666 of *LNCS*, pages 537–554. Springer, 1999.
17. Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. In *CRYPTO*, volume 8043 of *LNCS*, pages 479–499. Springer, 2013.
18. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 545–554. ACM, 2013.
19. Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. Bounded ciphertext policy attribute based encryption. In *Automata, Languages and Programming*, volume 5126 of *LNCS*, pages 579–591. Springer, 2008.
20. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98. ACM, 2006.
21. Mike Hamburg. Spatial encryption. Cryptology ePrint Archive, Report 2011/389, 2011. `http://eprint.iacr.org/`.
22. Vincenzo Iovino and Giuseppe Persiano. Hidden-vector encryption with groups of prime order. In *Pairing*, volume 5209 of *LNCS*, pages 75–88. Springer, 2008.
23. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, volume 4965 of *LNCS*, pages 146–162. Springer, 2008.
24. Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, volume 6110 of *LNCS*, pages 62–91. Springer, 2010.
25. Allison B. Lewko, Amit Sahai, and Brent Waters. Revocation systems with very small private keys. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 273–285. IEEE, 2010.
26. Daisuke Moriyama and Hiroshi Doi. A fully secure spatial encryption scheme. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 94(1):28–35, 2011.
27. Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In *ASIACRYPT*, volume 5912 of *LNCS*, pages 214–231. Springer, 2009.
28. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, volume 6223 of *LNCS*, pages 191–208. Springer, 2010.
29. Tatsuaki Okamoto and Katsuyuki Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. In *Cryptology and Network Security*, volume 7092 of *LNCS*, pages 138–159. Springer, 2011.
30. Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In *EUROCRYPT*, volume 7237 of *LNCS*, pages 591–608. Springer, 2012.
31. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In *ASIACRYPT*, volume 7658 of *LNCS*, pages 349–366. Springer, 2012.
32. Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In *ACM Conference on Computer and Communications Security*, pages 195–203, 2007.

33. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, volume 3494 of *LNCS*, pages 457–473. Springer, 2005.
34. Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, LNCS, pages 47–53. Springer, 1984.
35. Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In *Automata, Languages and Programming*, volume 5126 of *LNCS*, pages 560–578. Springer, 2008.
36. Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public Key Cryptography*, volume 6571 of *LNCS*, pages 53–70. Springer, 2011.
37. Brent Waters. Functional encryption for regular languages. In *CRYPTO*, volume 7417 of *LNCS*, pages 218–235. Springer, 2012.
38. Shota Yamada, Nuttapong Attrapadung, Goichiro Hanaoka, and Noboru Kunihiro. Generic constructions for chosen-ciphertext secure attribute based encryption. In *Public Key Cryptography*, volume 6571 of *LNCS*, pages 71–89. Springer, 2011.
39. Muxin Zhou and Zhenfu Cao. Spatial encryption under simpler assumption. In *PROVSEC*, volume 5848 of *LNCS*, pages 19–31. Springer, 2009.

A Security Definition of Functional Encryption

We first define what are the different important phases of a challenger $\mathcal{CH}$. Depending on how an adversary $\mathscr{A}$ runs with the phases and what are the oracle access it might have, different security advantages are defined.

Definition 8 (Security Model). *A challenger for a functional encryption security game has the following different phases.*

- Setup : *The challenger $\mathcal{CH}$ runs the setup algorithm to produce $(\mathcal{PP}, \mathcal{MSK})$. Then, $\mathcal{CH}$ gives $\mathcal{PP}$ to $\mathscr{A}$ and keeps $\mathcal{MSK}$ to itself.*
- Query Phase 1: *The adversary $\mathscr{A}$ is given access to the oracles $\mathsf{KeyGen}_{\mathcal{PP},\mathcal{MSK}}(\cdot,\cdot)$ and $\mathsf{Dec}_{\mathcal{PP}}(\cdot,\cdot,\cdot)$.*
- Challenge : *$\mathscr{A}$ submits two messages $\psi_0, \psi_1 \in \Psi$. $\mathcal{CH}$ picks $b \xleftarrow{\mathrm{U}} \{0,1\}$ and returns the challenge ciphertext $\mathsf{C}^* = \mathsf{Enc}(\mathcal{PP}, \psi_b)$ to $\mathscr{A}$. Sometimes this phase may be split into two and $\mathscr{A}$ may submit a part of two messages in the first phase and submit the rest in the second phase.*
- Query Phase 2: *Same as Phase 1.*
- Guess: *$\mathscr{A}$ finally sends a guess b' to $\mathscr{B}$.*

$\mathscr{A}$ is said to be **correct** if

1. $\mathcal{F}(x, \psi_0) = \mathcal{F}(x, \psi_1)$ for $x = \epsilon$ or all x queried to KeyGen oracle and
2. (C^*, x) is never queried to Dec oracle such that $\mathcal{F}(x, \psi_0) \neq \mathcal{F}(x, \psi_1)$.

Definition 9 (Security Advantages). *In case of selective-message indistinguishability game (sIND-), the challenge phase would be run before the setup phase, otherwise we have adaptive indistinguishability game (or IND-). More formally, in selective-message game, $\mathscr{A}$ has to submit the challenge pair before*

Table 2. Different functional encryption schemes with either delegation or verifiability. Verify, NK, NA respectively stand for verifiability, 'not known' and 'not applicable'. The schemes followed by '$*$' satisfy restricted delegation but not the actual delegation. The schemes followed by '$**$' are not directly applicable to CCA conversion, but with a small modification (as mentioned in [38]) are eligible for the conversion. The schemes, where Verify field is assigned to NA means we only consider these schemes in the context of index hiding.

Schemes	Class of FE	Policy type	Delegate	Verify
Goyal et al. sec.4 [20]	Traditional ABE	KP	NK	✓
Goyal et al. sec.5 [20]	Traditional ABE	KP	✓	✓
Goyal et al. sec.A [20]	Traditional ABE	KP	NK	✓
Ostrovsky et al. sec.3 [32]	Traditional ABE	KP	NK	✓
Bethencourt et al. [5]	Traditional ABE	CP	✓	✓
Goyal et al. [19]	Traditional ABE	CP	NK	✓
Waters sec.3 [36]	Traditional ABE	CP	✓	✓
Waters sec.5 [36]	Traditional ABE	CP	✓	✓
Lewko et al. sec.6 [25]	Traditional ABE	KP	NK	✓
Attrapadung et al. [4]	Traditional ABE	KP	NK	✓
Lewko et al. sec.2 [24]	Traditional ABE	CP	NK	NK
Lewko et al. sec.2 [24] $**$	Traditional ABE	CP	✓	✓
Okamoto et al. [28]	Traditional ABE	KP	NK	NK
Okamoto et al. [28] $**$	Traditional ABE	KP	✓	✓
Waters [37] $*$	FE for Regular Languages	KP	✓	✓
Attrapadung sec.6.2 [1] $*$	FE for Regular Languages	KP	✓	✓
Attrapadung sec.8.2 [1]	FE for Regular Languages	CP	NK	✓
Boneh et al. [7]	Spatial Encryption	-	✓	✓
Zhou et al. sec.3 [39]	Spatial Encryption	-	✓	✓
Hamburg [21]	Doubly-Spatial Encryption	-	✓	✓
Moriyama et al. [26]	Spatial Encryption	-	✓	NK
Chen et al. [12]	Doubly-Spatial Encryption	-	✓	NK
Chen and Wee [13]	Doubly-Spatial Encryption	-	✓	✓
Garg et al. [17]	ABE for Circuits	KP	NK	✓
Garg et al. [17]	ABE for Circuits	CP	-	✓
Okamoto et al. [27]	HIPE	-	✓	NA
Lewko et al. sec.3.5 [24]	IPE	-	✓	NA
Lewko et al. sec.B.6 [24]	HIPE	-	✓	NA
Okamoto et al. sec.5 [30]	IPE	-	✓	NA
Okamoto et al. sec.E [30]	HIPE	-	✓	NA
Attrapadung sec.4.1 [3]	IPE	-	NK	✓
Okamoto et al. sec.10 [29]	HIPE	-	✓	✓

receiving $\mathcal{PP}$ from $\mathcal{CH}$. However, a part of challenge (e.g., target policy in case of PE) may be given before setup phase. (For $I \in \{sIND, IND\}$) $\mathscr{A}$ is not allowed to ask decryption query in chosen plaintext attack (I-CPA) model and in contrast, $\mathscr{A}$ can ask for decryption query in chosen ciphertext attack (I-CCA) model. The advantage of a correct $\mathscr{A}$ in any one of the combination of the above game is defined by

$$\mathsf{Adv}^{\text{model}}_{\mathscr{A},\mathsf{FE}}(\kappa) = \left|\Pr[b = b'] - \frac{1}{2}\right|.$$

The superscript model is one of the following IND-CCA, IND-CPA, sIND-CPA and sIND-CCA depending on the interaction types between $\mathscr{A}$ and the challenger $\mathcal{CH}$ as described above. For example, $\mathsf{Adv}^{\text{IND-CPA}}_{\mathscr{A},\mathsf{FE}}(\kappa)$ denotes the advantages for a correct $\mathscr{A}$ interacting with IND-CPA challenger $\mathcal{CH}$.

A functional encryption FE is said to be model secure if for all correct PPT adversary $\mathscr{A}$, the advantage $\mathsf{Adv}^{\text{model}}_{\mathscr{A},\mathsf{FE}}(\kappa)$ is at most a negligible function in security parameter κ. The IND-CCA (resp. sIND-CCA) security of a functional encryption is also called as adaptive (resp. selective) security.

B One Time Signature, Mac and Weak Commitment

Definition 10 (Signature Scheme). *A signature scheme consists of three PPT algorithms -* Gen, Sign *and* Ver

- Gen*: It takes a security parameter* κ*. It outputs a verification key* vk *and a signing key* signk.
- Sign*: It takes a message* m *and a signing key* signk *as input. It returns a signature* δ.
- Ver*: It receives a message* m*, a signature* δ *and a verification key* vk *as input. It returns a boolean value 1 for accept or 0 for reject.*

Definition 11 (Strong Unforgeability of One-Time Signature). *Strongly unforgeability one-Time signature model is defined as a game,* Game_{Real} *between a challenger* $\mathscr{B}$ *and an adversary* $\mathscr{A}$*, where the adversary has to forge a signature for a message. The game,* Game_{Real} *consists of the following phases:*
Gen: *The challenger* $\mathscr{B}$ *runs* $\mathsf{Gen}(1^\kappa) \longrightarrow (\mathsf{vk}, \mathsf{signk})$*. Then* vk *is given to the adversary* $\mathscr{A}$*.*
Query: *The adversary* $\mathscr{A}$ *is given access to the oracle* $\mathsf{Sign}(., \mathsf{signk})$ *at most once. Let* (m, δ) *be the corresponding query message and relied signature.*
Forgery: *The adversary outputs a signature* (m^*, δ^*)*.*
We say the adversary succeeds in this game if $\mathsf{Ver}(m^*, \delta^*, \mathsf{vk}) = 1$ *and* $(m, \delta) \neq (m^*, \delta^*)$*.*
Let $\mathsf{Adv}^{\text{OTS}}_{\mathscr{A}}(\kappa)$ *denote the success probability for any adversary* $\mathscr{A}$ *in the above experiment. A signature scheme is said to be Strongly unforgeable one-time signature if* $\mathsf{Adv}^{\text{OTS}}_{\mathscr{A}}(\kappa)$ *is at most negligible function in* κ

Definition 12 (Message Authentication Code). *A message authentication code (MAC) consists of two algorithms -* Mac *and* MVer

- Mac *It takes as inputs, a symmetric key* $\mathsf{AK} \in \mathcal{K}$*, where* $\mathcal{K}$ *is a key space and a message* $m \in \mathcal{M}$ *and it outputs tag* τ*. In notation, we write* $\tau := \mathsf{Mac}_{\mathsf{AK}}(m)$.
- MVer *It takes the inputs, a symmetric key* AK*, a message* m *and a tag* τ*. It returns 1 for accept and 0 for reject. We use the notation,* $\mathsf{MVer}_{\mathsf{AK}}(m, \tau)$ *for* $\mathsf{MVer}(\mathsf{AK}, m, \tau)$*.*

For correctness, it is required that for all $\mathsf{AK} \in \mathcal{K}$ *and all* $m \in \mathcal{M}$ *that* $\mathsf{MVer}_{\mathsf{AK}}(m, \mathsf{Mac}_{\mathsf{AK}}(m)) = 1$

Definition 13 (Message Authentication). *A message authentication code* $(\mathsf{Mac}, \mathsf{MVer})$ *is secure against a one-time chosen-message attack if the success probability of any PPT adversary* $\mathscr{A}$ *in the following game is negligible in the security parameter* κ*:*

1. *A random key* $\mathsf{AK} \in \mathcal{K}$ *is chosen.*
2. $\mathscr{A}$ *outputs a message* m *and is given in return* $\tau = \mathsf{Mac}_{\mathsf{AK}}(m)$.
3. $\mathscr{A}$ *outputs a pair* (m', τ').

We say that $\mathscr{A}$ *succeeds in above game if* $(m, \tau) \neq (m', \tau')$ *and* $\mathsf{MVer}_{\mathsf{AK}}(m', \tau') = 1$.

Definition 14 (Weak Commitment). *A weak commitment is a triple of PPT algorithms -* CSetup*,* Commit *and* Decommit *such that:*

- CSetup *It takes as input the security parameter* 1^κ *and outputs a string* pub.
- Commit *It takes input* pub*, and outputs* $(\mathsf{AK}, \mathsf{com}, \mathsf{decom})$ *with* $\mathsf{AK} \in \{0,1\}^\kappa$. *We would say* com *as the public commitment string and* decom *as the decommitment string.*
- Decommit *It takes as input* pub*,* com *and* decom*, and outputs a key* $\mathsf{AK} \in \{0,1\}^\kappa \cup \perp$.

For correctness, it is required that for all pub *generated by* CSetup *and for all* $(\mathsf{AK}, \mathsf{com}, \mathsf{decom}) \longleftarrow \mathsf{Commit}(\mathsf{pub})$*, we have* $\mathsf{Decommit}(\mathsf{pub}, \mathsf{com}, \mathsf{decom}) = \mathsf{AK}$.

Definition 15 (Security Weak Commitment). *A weak commitment scheme is said to be secure if it satisfies both hiding and binding as follows:*
Hiding*: For all PPT* $\mathscr{A}$ *the following is negligible:*

$$\left| \Pr \left[\begin{matrix} \mathsf{pub} \longleftarrow \mathsf{CSetup}(1^\kappa),\ \mathsf{AK}_0 \xleftarrow{\mathrm{U}} \{0,1\}^\kappa \\ (\mathsf{AK}_1, \mathsf{com}, \mathsf{decom}) \longleftarrow \mathsf{Commit}(\mathsf{pub}), b \xleftarrow{\mathrm{U}} \{0,1\} \end{matrix} : \mathscr{A}(\mathsf{pub}, \mathsf{com}, \mathsf{AK}_b) = b \right] - \frac{1}{2} \right|.$$

Binding*: For all PPT* $\mathscr{A}$ *the following is negligible:*

$$\Pr \left[\begin{matrix} \mathsf{pub} \longleftarrow \mathsf{CSetup}(1^\kappa), \\ (\mathsf{AK}, \mathsf{com}, \mathsf{decom}) \longleftarrow \mathsf{Commit}(\mathsf{pub}), : \mathsf{Decommit}(\mathsf{pub}, \mathsf{com}, \mathsf{decom}) \notin \{\perp, \mathsf{AK}\} \\ \mathsf{decom}' \longleftarrow \mathscr{A}(\mathsf{pub}, \mathsf{com}, \mathsf{AK}) \end{matrix} \right].$$

C A Brief Literature Survey on Functional Encryption

The attribute-based encryption was introduced by Sahai and Waters [33] as a Fuzy-IBE. Since, then many ABE schemes have been constructed either in the form of KP-ABE [20, 32, 25, 4], where the key-indices access structures (policies) and the associated indices are attribute vectors (sets of attributes), or in the form of CP-ABE, [5, 19, 36], where the roles of policies and the sets of attributes are interchanged.

Spatial Encryption. Spatial encryption was introduced by Boneh and Hamburg [7] as a special instance of generalized identity-based encryption (GIBE). The GIBE [7] would capture many functional encryptions, e.g., traditional IBE, broadcast IBE, HIBE, ABE and forward-secure system etc. The one of the main building blocks for GIBE was known to be spatial encryption [7, 39, 26], where the key-indices (roles) are the affine subspaces of an affine space and the associated indices (policies) are the points of the affine space. The access is granted

if the associated index is a member of the affine subspace. Boneh and Hamburg [7] showed how to obtain the different flavor of IBE from spatial encryption, e.g., HIBE, inclusive IBE, co-inclusive IBE, broadcast IBE, product scheme, multiple authority scheme and forward-secure scheme etc. Later, Hamburg [21] extended the notion of spatial encryption to doubly-spatial encryption (DSE), from which the expressive functional encryptions can be derived, e.g., IPE, ABE etc. In doubly-spatial encryption [21, 12, 13], both the key-indices (roles) and associated indices (policies) are the affine subspaces and a key-index satisfies an associated index if and only if their intersection is not void.

ABE for Circuits. Traditional ABE could provide the functionality for boolean formulas or equivalently circuits with fanout 1 (captured by the complexity class NP^1). Later, Garge et al. [17] and Gorbunov et al. [18] independently proposed the ABE constructions for circuits of arbitrary fanout based on the multi-linear maps and LWE assumption respectively. However, all the previous ABE systems [18, 17] for general circuits were proven selectively secure. Recently, Attrapadung [2] provided fully (adaptively) secure ABE systems for circuits based on asymmetric graded encoding systems in composite-order settings.

Functional Encryption for Regular Languages. All the aforementioned systems except the FE [37, 1] for regular languages can provide at most the bounded access. Waters [37] first moved to unbounded access control systems (KP-FE), where the key-indices are the regular languages represented by DFAs over an alphabet, Σ and the associated indices are the strings over Σ. The system [37] was shown to be selectively IND-CPA secure. In contrast, Attrapadung [1] proposed the adaptively IND-CPA secure FE for regular languages.

Inner Product Encryption. Katz, Sahai and Waters [23] introduced the predicate encryption (with hidden index) for inner product, where access control is defined through the orthogonality of key-index and associated index. This PE is known as zero IPE [31, 30, 24, 29] and it's dual version, where the relation is defined through non-orthogonality is called non-zero IPE [3, 29]. The zero IPE schemes are mainly used for the purpose of attribute hiding are available in either hierarchical or non-hierarchical style, but some of them [3, 29] are used to handle payload hiding.

D Construction using MAC, weak commitment and restricted delegation

Now we demonstrate a generic construction using MAC, weak commitment scheme and restricted delegation property of the underlying CPA-secure construction. This is an analogue of the construction (section 3.2) using OTS scheme. Let $(\mathsf{Mac}, \mathsf{MVer})$ and $(\mathsf{CSetup}, \mathsf{Commit}, \mathsf{Decommit})$ be the respectively MAC and weak commitment scheme (security definitions can be found in Appendix B). We would like to note that security requirement of MAC is equivalent to pairwise independent hash function. We note that the public parameter, pub of the weak commitment scheme is generated along the public parameter generation of the

primitive encryption scheme by the Setup algorithm of the new (CCA-secure) predicate encryption.

- $\mathsf{KeyGen}(x) := \mathsf{KeyGen}'(x')$ (in notation: $\mathcal{SK}_x := \mathcal{SK}'_{x'}$).
- $\mathsf{Enc}(m, y)$: It runs $(\mathsf{AK}, \mathsf{vk}(= \mathsf{com}), \mathsf{decom}) \longleftarrow \mathsf{Commit}(\mathsf{pub})$ and returns

$$\mathsf{CT} = (\mathsf{C} := \mathsf{Enc}'((m, \mathsf{decom}), y_{\mathsf{vk}}),\ \tau := \mathsf{Mac}_{\mathsf{AK}}(\mathsf{C}),\ \mathsf{vk}).$$

- $$\mathsf{Dec}_{\mathcal{SK}_x}(\mathsf{CT} := (\mathsf{C}, \tau, \mathsf{vk})) = \begin{cases} m & \text{if } \begin{pmatrix} K := \mathsf{Delegate}(\mathcal{PP}, \mathcal{SK}'_{x'}, x', x_{\mathsf{vk}}), \\ (m, \mathsf{decom}) := \mathsf{Dec}'_K(\mathsf{C}), \\ \mathsf{AK} := \mathsf{Decommit}(\mathsf{pub}, \mathsf{vk}, \mathsf{decom}) \\ \text{and } \mathsf{MVer}_{\mathsf{AK}}(\mathsf{C}, \tau) = 1 \end{pmatrix} \\ \perp & \text{otherwise} \end{cases}$$

Theorem 5. *Let $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ be a delegation-friendly index-transformer, PE' be an IND-CPA secure predicate encryption scheme with the restricted delegation and, $(\mathsf{Mac}, \mathsf{MVer})$ and $(\mathsf{CSetup}, \mathsf{Commit}, \mathsf{Decommit})$ respectively be the secure (in sense of definition 15 and 13) MAC and weak commitment scheme, then the above proposed scheme PE in section D is an IND-CCA secure predicate encryption scheme.*

Proof. The proof is followed from that of Theorem 1 in section 3 and the arguments of [8] for avoiding the circularity from primitive predicate encryption and weak commitment.

E Construction using MAC, weak commitment and restricted verifiability

Now we demonstrate a generic construction using MAC, weak commitment scheme and restricted verifiability property of the underlying CPA-secure construction. This is an analogue of the construction (section 3.3) using OTS scheme. Let $(\mathsf{Mac}, \mathsf{MVer})$ and $(\mathsf{CSetup}, \mathsf{Commit}, \mathsf{Decommit})$ be the respectively MAC and weak commitment scheme.

- $\mathsf{KeyGen}(x) := \mathsf{KeyGen}'(x')$ (in notation: $\mathcal{SK}_x := \mathcal{SK}'_{x'}$).
- $\mathsf{Enc}(m, y)$: It runs $(\mathsf{AK}, \mathsf{vk}(= \mathsf{com}), \mathsf{decom}) \longleftarrow \mathsf{Commit}(\mathsf{pub})$ and returns

$$\mathsf{CT} = (\mathsf{C} := \mathsf{Enc}'((m, \mathsf{decom}), y_{\mathsf{vk}}),\ \tau := \mathsf{Mac}_{\mathsf{AK}}(\mathsf{C}),\ y,\ \mathsf{vk}).$$

- $$\mathsf{Dec}_{\mathcal{SK}_x}(\mathsf{CT} := (\mathsf{C}, \delta, y, \mathsf{vk})) = \begin{cases} m & \text{if } \begin{pmatrix} x \sim y, \mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x', \epsilon^{y}_{\mathsf{vk}}) = 1, \\ (m, \mathsf{decom}) := \mathsf{Dec}'_{\mathcal{SK}'_{x'}}(\mathsf{C}), \\ \mathsf{AK} := \mathsf{Decommit}(\mathsf{pub}, \mathsf{vk}, \mathsf{decom}), \\ \text{and } \mathsf{MVer}_{\mathsf{AK}}(\mathsf{C}, \tau) = 1 \end{pmatrix} \\ \perp & \text{otherwise.} \end{cases}$$

Theorem 6. *Let* $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ *be a verifiability-friendly index-transformer,* PE' *be an IND-CPA secure predicate encryption scheme with restricted verifiability,* $(\mathsf{Mac}, \mathsf{MVer})$ *and* $(\mathsf{CSetup}, \mathsf{Commit}, \mathsf{Decommit})$ *respectively be the secure MAC and weak commitment scheme, then the above proposed scheme* PE *in section E is an IND-CCA secure predicate encryption scheme.*

Proof. The proof is followed from that of Theorem 2 in section 3 and the arguments of [8] for avoiding the circularity from primitive predicate encryption and weak commitment.

F Proof of Theorem 2

Proof. We describe for the adaptive security. One can similarly have a proof for selective security. Similar to the proof of Theorem 1, we show that if $\mathscr{A}$ can break the IND-CCA security of the proposed scheme PE, then we establish an algorithm $\mathscr{B}$ for breaking IND-CPA security of the primitive predicate scheme PE' with advantage $\mathsf{Adv}^{\mathrm{IND-CCA}}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{\mathrm{sUF-CMA}}_{\mathscr{A},\mathrm{OTS}}(\kappa)$. Let $\mathcal{CH}$ be the challenger for the primitive predicate encryption scheme PE'. The role of $\mathscr{B}$ is the same as described in proof based on restricted delegation.

$\mathscr{B}$: It first run $(\mathsf{vk}^*, \mathsf{signk}^*) \longleftarrow \mathsf{OTS.Gen}(1^\kappa)$. In the setup phase $\mathscr{B}$ simply forwards the public parameter $\mathcal{PP}$, obtained from $\mathcal{CH}$, to $\mathscr{A}$.
Phase 1/2 Query: It consists of the following queries in adaptive manner:

- KeyGen Query: Let $x \in \mathcal{X}$ be a key-index queried by $\mathscr{A}$, then $\mathscr{B}$ makes a key query for $x' := \mathcal{T}_1(x)$ to $\mathcal{CH}$. Then $\mathcal{CH}$ replies the key $\mathcal{SK}_x = \mathcal{SK}'_{x'}$ to $\mathscr{B}$ and the same key is passed to $\mathscr{A}$.
- Decrypt Query: Let (CT, x), where $\mathsf{CT} = (\mathsf{C}, \delta, y, \mathsf{vk})$ be a decryption query by $\mathscr{A}$. Then, $\mathscr{B}$ runs $\mathsf{flag}_o \longleftarrow \mathsf{OTS.Ver}(\mathsf{C}, \delta, \mathsf{vk})$ and $\mathsf{flag}_f \longleftarrow \mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x', \epsilon^y_{\mathsf{vk}})$. If any of the flag values is false, returns $\perp$ to $\mathscr{A}$ els proceeds. If $\mathsf{vk} = \mathsf{vk}^*$, $\mathscr{B}$ aborts the game we set $\mathsf{BAD}_{\mathrm{OTS}}$ true, else moves to next step. $\mathscr{B}$ makes a key query for $\epsilon^y_{\mathsf{vk}} := \mathcal{T}_2(y, \mathsf{vk})$ to $\mathcal{CH}$ and let K be the replied key for the index ϵ^y_{vk}. Then, $\mathscr{B}$ returns $\mathsf{Dec}'_K(\mathsf{C})$ to $\mathscr{A}$.

Challenge: Whenever $\mathscr{A}$ submits two equal length messages $m_0, m_1 \in \mathcal{M}$ and a challenge data index y^* to $\mathscr{B}$, $\mathscr{B}$ submits the same messages $m_0, m_1 \in \mathcal{M}$ and a challenge policy $y^*_{\mathsf{vk}^*} := \mathcal{T}_3(y^*, \mathsf{vk}^*)$ to $\mathcal{CH}$. Then, $\mathcal{CH}$ picks $b \xleftarrow{\mathrm{U}} \{0,1\}$ and returns $\mathsf{C}^* = \mathsf{Enc}'(\mathcal{PP}, m_b, y^*_{\mathsf{vk}^*})$ to $\mathscr{B}$ as a challenge ciphertext. Now, $\mathscr{B}$ runs $\delta^* \longleftarrow \mathsf{OTS.Sign}(\mathsf{C}^*, \mathsf{signk}^*)$ and returns $\mathsf{CT}^* := (\mathsf{C}^*, \delta^*, \mathsf{vk}^*)$ to $\mathscr{A}$.
Guess: $\mathscr{A}$ sends a guess b' for b to $\mathscr{B}$ and, then $\mathscr{B}$ returns the same guess b' to $\mathcal{CH}$.

Analysis: As the verification key of OTS has been chosen in the beginning of the game it is easy to define a forging algorithm which forges correctly against the the one-time signature whenever $\mathsf{BAD}_{\mathrm{OTS}}$ sets true. So we may assume that

$\mathsf{BAD}_{\mathrm{OTS}}$ does not set true throughout. In this case we show the following two things:

Claim-1 $\mathscr{B}$ follows the restriction of CPA-security game (as interacting with $\mathcal{CH}$) as long as $\mathscr{A}$ does so. In other words, $\mathscr{B}$ is correct given that $\mathscr{A}$ is correct.

Claim-2 Until $\mathscr{B}$ aborts (i.e., BAD_{OTS} occurs), all responses of $\mathscr{B}$ to $\mathscr{A}$ are identically distributed with the responses of a CCA-challenger $\mathcal{CH}_{CCA}$ to $\mathscr{A}$.

Assuming the above, we have

$$\mathsf{Adv}^{\mathrm{IND-CPA}}_{\mathscr{B},\mathrm{PE}'}(\kappa) \geq \mathsf{Adv}^{\mathrm{IND-CCA}}_{\mathscr{A},\mathrm{PE}}(\kappa) - \frac{1}{2}\mathsf{Adv}^{\mathrm{sUF-CMA}}_{\mathscr{A},\mathrm{OTS}}(\kappa)$$

which concludes the proof. Now we will show the above two claims.

Proof of Claim-1. By the natural restriction on key queries by $\mathscr{A}$, we have for each queried key-index x

$$x \not\sim y^* \tag{4}$$

For each key query on index x' by $\mathscr{B}$, we have

$$x' \not\sim' y^*_{\mathsf{vk}^*} \qquad \text{(by condition (1) and equation (4))}$$

which is required as a natural restriction on key queries by $\mathscr{B}$. To answer the decryption query $(\mathsf{CT} := (\mathsf{C}, \delta, y, \mathsf{vk}), x)$ of $\mathscr{A}$, $\mathscr{B}$ makes a key query to $\mathcal{CH}$ for the key-index $\epsilon^{y}_{\mathsf{vk}}$ and then, it decrypts the ciphertext using $\mathcal{SK}'_{\epsilon^{y}_{\mathsf{vk}}}$, in stead of $\mathcal{SK}_x := \mathcal{SK}'_{x'}$. Now, the restricted verifiability emphasizes that $\mathscr{B}$ does not violet the rule for decryption as expected in CCA game with $\mathscr{A}$. Again $\mathsf{vk} \neq \mathsf{vk}^*$ implies that

$$\epsilon^{y^*}_{\mathsf{vk}} \not\sim' y^*_{\mathsf{vk}^*} \qquad \text{(by condition (3))}$$

which is again a requirement for key queries by $\mathscr{B}$.

Proof of Claim-2. This is more or less straightforward from the restricted-verifiability property.

G A Concrete CCA-secure Construction based on Waters scheme [37]

– $\mathsf{Setup}(1^\kappa, \Sigma)$: On the assumption that $0, 1 \in \Sigma$, it sets $\Sigma' := \Sigma$, otherwise $\Sigma'(\supset \Sigma)$ includes either 0 or 1 or both. It runs the bilinear group generator on 1^κ to produce $(p, g, e, \mathbb{G}, \mathbb{G}_T)$, where $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ is a bilinear pairing map, g is a generator of $\mathbb{G}$, and $\mathbb{G}$ and $\mathbb{G}_T$ are both cyclic group of prime order p. Let n be an integer related to the security parameter κ which would be chosen so that the security advantages contributed by n is comparable to other security components of security advantages. For any m, let $\mathbb{F}_{2^m}$ denote the finite field of size 2^m. Moreover, we fix two hash

functions, namely an universal one-way hash function $G : \{0,1\}^{4n} \to \{0,1\}^n$ and a collision resistant hash function $H : \{0,1\}^* \to \{0,1\}^n$. It now picks

$$z, h_{start}, h_{end}, h_\sigma \xleftarrow{U} \mathbb{G} \text{ for each } \sigma \in \Sigma', \alpha \xleftarrow{U} \mathbb{Z}_p, K_0, K_1, K_2 \xleftarrow{U} \mathbb{F}_{2^{2n}}.$$

Then, it publishes $\mathcal{PP}$ and $\mathcal{MSK}$:

$$\mathcal{PP} := [e(g,g)^\alpha, g, z, h_{start}, h_{end}, K_0, K_1, K_2, \langle h_\sigma \rangle_{\sigma \in \Sigma'}] \quad \mathcal{MSK} := [g^{-\alpha}]$$

– $\mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, M := (Q, \Sigma, \mathcal{T}, q_0, F))$: It first applies the transformation $\mathcal{T}_1$ to M and let $M' := \mathcal{T}_1(M) = (Q', \Sigma', \mathcal{T}', q_0', F)$, where $Q' := Q \cup \{q_0', \ldots, q_{n-1}'\}$ with $q_i' \notin Q$ for $i = 0, \ldots, (n-1)$ and $\mathcal{T}' = \mathcal{T} \cup \{(q_{i-1}', q_i', 1-j) : j = 0, 1;\ i = 1, \ldots, n\}$ with $q_n' = q_0$.
Let the states in Q' be enumerated as $q_0, q_1, \ldots, q_{|Q'|-1}$. For each state $q_i \in Q'$, it picks $D_i \xleftarrow{U} \mathbb{G}$. It picks $r_{start} \xleftarrow{U} \mathbb{Z}_p$. For each $t \in \mathcal{T}'$, it chooses $r_t \xleftarrow{U} \mathbb{Z}_p$. For each $q_x \in F$, it chooses $r_{endx} \in \mathbb{Z}_p$. Now, it computes the initial key components

$$K_{start,1} := D_0 (h_{start})^{r_{start}},\ K_{start,2} := g^{r_{start}}$$

For each transition $t = (x, y, \sigma) \in \mathcal{T}'$, it produces the key components as

$$K_{t,1} := {D_x}^{-1} z^{r_t},\ K_{t,2} := g^{r_t},\ K_{t,3} := D_y (h_\sigma)^{r_t}$$

For each final state $q_x \in F$, it sets the final key components as:

$$K_{endx,1} := g^{-\alpha}.D_x (h_{end})^{r_{endx}},\ K_{endx,2} := g^{r_{endx}}$$

Finally, it returns $\mathcal{SK}_{M'} := [M', K_{start,1}, K_{start,2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t \in \mathcal{T}'}, (K_{endx,1}, K_{endx,2})_{q_x \in F}]$

– $\mathsf{Enc}(\mathcal{PP}, m, w := (w_1 \cdots w_\ell))$: It first choose $(x_1, x_2) \in \mathbb{F}_{2^{2n}}^2$ and defines $\mathsf{vk} = G(x_1, x_2)$. We compute $\mathsf{AK} = K_0 + K_1 x_1 + K_2 x_2$ and write it as $(r_1, r_2) \in \mathbb{F}_{2^n}^2$. Then, it applies the transformation $\mathcal{T}_2$ to w and let $w_{\mathsf{vk}} := \mathcal{T}_2(w, \mathsf{vk}) = \mathsf{vk} || w$. It sets $\ell' := \ell + n$. Let w_i denote the i^{th} symbol of w_{vk}. It picks $s_0, s_1, \ldots, s_{\ell'} \xleftarrow{U} \mathbb{Z}_p$. Then, it computes the ciphertext components as follows:

$$\mathsf{C}_m := m.e(g,g)^{\alpha.s_{\ell'}},\ \mathsf{C}_{start,1} := \mathsf{C}_{0,1} = g^{s_0},\ \mathsf{C}_{start,2} := (h_{start})^{s_0}$$

For $i \in [\ell']$, it computes:

$$\mathsf{C}_{i,1} := g^{s_i},\ \mathsf{C}_{i,2} := (h_{w_i})^{s_i} z^{s_{i-1}}$$

It sets the final components as:

$$\mathsf{C}_{end,1} := \mathsf{C}_{\ell',1} = g^{s_{\ell'}},\ \mathsf{C}_{end,2} := (h_{end})^{s_{\ell'}}$$

It sets $\mathsf{C}_{w_{\mathsf{vk}}} := [w_{\mathsf{vk}}, \mathsf{C}_m, \mathsf{C}_{start,1}, \mathsf{C}_{start,2}, (\mathsf{C}_{1,1}, \mathsf{C}_{1,2}), \ldots, (\mathsf{C}_{\ell',1}, \mathsf{C}_{\ell',2}), \mathsf{C}_{end,1}, \mathsf{C}_{end,2}]$. Now, it returns $\mathsf{CT}_w := (\mathsf{C}_{w_{\mathsf{vk}}}, \delta, \mathsf{vk})$ where δ is computed as $r_1 H(\mathsf{C}_{w_{\mathsf{vk}}}) + r_2$.

We skip the decryption algorithm as it is same the decryption algorithm for the generic conversion.

Theorem 7. *If G is universal one-way hash function (UOWHF), H is collision resistant hash function and $(\ell^* + n)$-Expanded BDHE assumption holds, then the above construction is IND-CCA-secure functional encryption for a regular language, where the size of the challenge string w^* is ℓ^*.*

Proof. It is followed from the proof of Theorem 5 in Appendix D or Theorem 6 in Appendix E and Theorem 4.1 of [37].

H Diagram of Index-Transformer for Regular Languages

Delegation/Verifiability-friendly index-transformer of KP-FE for regular languages

W.L.G, assume that $0, 1 \in \Sigma$ and set $\Sigma' = \Sigma$

M' : $q'_0 \xrightarrow{0,1} q'_1 \xrightarrow{0,1} q'_2 \quad \cdots \quad \xrightarrow{0,1} q'_n = q_0$ (M)

$q'_0, \ldots, q'_{n-1} \notin Q$, where F is unchanged.

M_{vk} : $q'_0 \xrightarrow{\mathsf{vk}[1]} q'_1 \xrightarrow{\mathsf{vk}[2]} q'_2 \quad \cdots \quad \xrightarrow{\mathsf{vk}[n]} q'_n = q_0$ (M)

$w_{\mathsf{vk}} := \mathsf{vk}||w$

Delegation-friendly index-transformer of CP-FE for regular languages

W.L.G, assume that $0, 1 \in \Sigma$ and set $\Sigma' = \Sigma \cup \{*\}$

$w' := *^n||w$ $\qquad$ $w_{\mathsf{vk}} := \mathsf{vk}||w$

M_{vk} : $q'_0 \xrightarrow{\mathsf{vk}[1]} q'_1 \xrightarrow{\mathsf{vk}[2]} q'_2 \quad \cdots \quad \xrightarrow{\mathsf{vk}[n]} q'_n = q_0$ (M)

$q'_0, \ldots, q'_{n-1} \notin Q$, where F is unchanged.

Verifiability-friendly index-transformer of CP-FE for regular languages

Assume that $\Sigma \cap \{0,1\} = \emptyset$ and set $\Sigma' = \Sigma \cup \{0,1\}$

$w' := w$ $\qquad\qquad$ $\epsilon^M_{\mathsf{vk}} := \mathsf{vk}$

M_{vk} is a DFA that recognises $\mathcal{L}(M) \cup \{\mathsf{vk}\}$, i.e.,

$q'_1, \ldots, q'_n \notin Q$.

$F' := F \cup \{q'_n\}$

M_{vk} :