The Trojan Method in Functional Encryption: From Selective to Adaptive Security, Generically

Prabhanjan Ananth* UCLA
Zvika Brakerski† Weizmann
Gil Segev‡ Hebrew U
Vinod Vaikuntanathan§ MIT

Abstract

In a functional encryption (FE) scheme, the owner of the secret key can generate restricted decryption keys that allow users to learn specific functions of the encrypted messages and nothing else. In many known constructions of FE schemes, such a notion of security is guaranteed only for messages that are fixed ahead of time (i.e., before the adversary even interacts with the system). This is called *selective security*, which is too restrictive for many realistic applications. Achieving *adaptive security* (also called *full security*), where security is guaranteed even for messages that are adaptively chosen at any point in time, seems significantly more challenging. The handful of known fully-secure schemes are based on specifically tailored techniques that rely on strong assumptions (such as obfuscation assumptions or multilinear maps assumptions).

In this paper we show that *any* sufficiently expressive selectively-secure FE scheme can be transformed into a fully secure one without introducing any additional assumptions. We present a direct black-box transformation, making novel use of *hybrid encryption*, a classical technique that was originally introduced for improving the efficiency of encryption schemes, combined with a new technique we call the *Trojan Method*. This method allows to embed a secret execution thread in the functional keys of the underlying scheme, which will only be activated within the proof of security of the resulting scheme. As another application of the Trojan Method, we show how to construct functional encryption schemes for arbitrary circuits starting from ones for shallow circuits (NC1 or even TC0).

*E-mail: prabhanjan@cs.ucla.edu. This work was done in part while visiting MIT, and was supported in part by the Northrop Grumman Cybersecurity Consortium. Research supported in part from a DARPA/ONR PROCEED award, NSF Frontier Award 1413955, NSF grants 1228984, 1136174, 1118096, and 1065276. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11- 1-0389. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

†E-mail: zvika.brakerski@weizmann.ac.il. Supported by an Alon Fellowship and by ISF.

‡E-mail: segev@cs.huji.ac.il. Supported by the European Union's Seventh Framework Programme (FP7) via a Marie Curie Career Integration Grant, by the Israel Science Foundation (Grant No. 483/13), and by the Israeli Centers of Research Excellence (I-CORE) Program (Center No. 4/11).

§Email: vinodv@mit.edu. Research supported in part by DARPA Grant number FA8750-11-2-0225, an Alfred P. Sloan Research Fellowship, the Northrop Grumman Cybersecurity Research Consortium (CRC), Microsoft Faculty Fellowship, and a Steven and Renee Finn Career Development Chair from MIT.

1 Introduction

Traditional notions of public-key encryption provide all-or-nothing access to data: owners of the secret key can recover the entire message from a ciphertext, whereas those who do not know the secret key learn nothing at all. Functional encryption [SW05, KSW08, LOS+10, BSW11, O'N10] is a modern type of encryption scheme where the owner of the (master) secret key can release function-specific secret keys sk_f, which enable a user to compute $f(x)$ from an encryption of x. Furthermore, using the key sk_f, one should not be able to extract any information about x other than $f(x)$. There are a number of ways to formalize this abstract requirement. In this paper we will exclusively use an *indistinguishability based* definition, requiring that given x_0, x_1 such that $f(x_0) = f(x_1)$, it is computationally hard to distinguish between $(\mathsf{Enc}(x_0), \mathsf{sk}_f)$ and $(\mathsf{Enc}(x_1), \mathsf{sk}_f)$. This definition extends to multiple messages and keys in a fairly straightforward manner.

While initial constructions of functional encryption [BF03, BCOP04, KSW08, LOS+10] were limited to simple function classes such as point functions and inner products, recent developments have dramatically improved the state of the art in this area. In particular, the works of Sahai and Seyalioglu [SS10] and Gorbunov, Vaikuntanathan and Wee [GVW12] showed that if only a single function key is produced, then functional encryption can be based on any semantically secure encryption. This result can be extended to a case where the number of function keys is polynomial and known a-priori. Goldwasser, Kalai, Popa, Vaikuntanathan and Zeldovich [GKP+13] constructed a scheme with succinct ciphertexts based on a specific hardness assumption (Learning with Errors).

The first functional encryption scheme that supports a-priori unbounded number of function keys was recently introduced by Garg, Gentry, Halevi, Raykova, Sahai and Waters [GGH+13], based on the existence of secure indistinguishability obfuscation (to which a heuristic construction is presented in the same paper). Garg et al. showed that given a secure indistinguishability obfuscator, their functional encryption scheme is *selectively secure*.

At a high level, selective security guarantees that the scheme is secure so long as the messages being encrypted are chosen independently of the parameters of the scheme. This is reflected in the formal definition by requiring the adversary to specify the messages that he intends to encrypt before seeing any parameter of the scheme, in particular before seeing any functional key. If the message to be encrypted depends on the the parameters (e.g. on the public key or on function keys) then security cannot be guaranteed. Whereas the independence assumption is justified in some cases, the ultimate sought-after notion of security is *adaptive security* (often called *full security*). An adaptively secure scheme guarantees security regardless of how the encrypted messages are chosen (granted that they can be produced in polynomial time, of course). This is reflected in the formal definition by allowing the adversary to specify the messages x_0, x_1 at any point in its execution, even after seeing function keys and encryptions of other ciphertexts.

Historically, the first functional encryption schemes have been selectively secure [BB04, GPSW06, KSW08, GVW13, GKP+13]. The question of constructing adaptively secure functional encryption is considered to be notoriously difficult and only few approaches are known. The basic observation is that if the input space is bounded in size, e.g. $\{0,1\}^n$ for a known n, then the x values can be guessed ahead of time with probability 2^{-n}. Starting with a *sub-exponential* hardness assumption, and taking the security parameter to be polynomial in n, allows to argue that the selectively secure scheme is in fact also adaptively secure. This method is known as "complexity leveraging" and is not considered to be a satisfactory solution to the problem since it requires relying on strong hardness assumptions. The powerful "dual system encryption" method, introduced by Waters [Wat09], had been used to construct adaptively secure attribute based encryption (a weak version of functional encryption) for formulas, as well as an adaptively secure functional encryption scheme for linear functions [LOS+10]. However, this method is merely a general outline and each construction is required to tailor the solution based on its specialized assumption. In some cases, such as attribute based encryption for circuits, it is not known how to implement dual system encryption or any other method to achieve adaptive security.

Starting with [GGH+13], there has been effort in the research community to construct an adaptively secure functional encryption scheme for polynomial size circuits with a-priori unbounded many function keys. Boyle, Chung and Pass [BCP14] constructed an adaptively secure FE scheme, under the assumption that extractability obfuscators exist (these are stronger primitives than the indistinguishability obfuscators

used by [GGH+13]). Waters [Wat14] showed how to do this assuming indistinguishability obfuscation, thus matching the assumption of [GGH+13]. Garg, Gentry, Halevi and Zhandry [GGHZ14] showed how to construct adaptively secure public-key FE from non-standard assumptions on multilinear maps. Each of these constructions uses its own methods and techniques, and in general it was not known how to achieve adaptivity in any non-ad-hoc method.

In this work, we use new techniques to show a *generic* construction of adaptively secure functional encryption, starting from any selectively secure scheme.

1.1 Our Results: From Selective to Adaptive Security

We show that any selectively secure functional encryption scheme implies an adaptively secure scheme, without relying on any additional assumptions. Our transformation applies equally to symmetric-key or public-key functional encryption, where the resulting adaptive scheme inherits its symmetric-key or public-key properties from the building block scheme. The following theorem summarizes our main contribution.

Theorem 1 (informal)**.** *Given any public-key (respectively symmetric-key) selectively-secure functional encryption scheme, there exists an adaptively secure public-key (respectively symmetric-key) functional encryption scheme supporting the class of bounded polynomial size circuits.*

We require that the selective scheme supports a sufficiently rich function class. In particular it will be applied to functions with a-priori bounded polynomial size and a-priori bounded depth (essentially the depth of computing a weak pseudorandom function, which is logarithmic under mild assumptions). The resulting adaptively secure scheme, however, does not have a depth bound. Therefore a corollary of our construction is that selectively secure functional encryption with depth bound implies adaptively secure functional encryption without a depth bound.

We view the significance of our result in a number of dimensions. First of all, it answers the basic call of cryptographic research to substantiate the existence of complicated primitives on that of simple primitives. We feel that this is of special interest in the case of adaptive security where it seemed that ad-hoc methods were required. Secondly, our construction, being of fairly low overhead, will allow to focus the attention of the research community in studying selectively secure functional encryption, rather than investing unwarranted efforts in the study of adaptive security. Lastly, we hope that our methods can be extended and employed towards those variants for which adaptive security is yet unattained, such as attribute based encryption for all polynomial size circuits.

1.2 Our Techniques: The Trojan Method and Hybrid Encryption

Our result is achieved by incorporating a number of techniques which will be explained in this section. We start by presenting a technique that we call *The Trojan Method*, whose origins lie in the "trapdoor circuits" idea presented in the work of De Caro et al. [CIJ+13], and in the work of Brakerski and Segev [BS14] in the context of function-private private-key FE. We extend this method and show that it is in fact a very powerful tool in the context of public-key FE, and as an example show that FE for "shallow" circuits (a circuit family that allows to compute weak PRFs) can be extended to one that applies to all circuits. We then explain the power of hybrid encryption in the context of functional encryption, and put it together with the Trojan method to present our final construction. Finally, a short comparison with of our technique with the aforementioned "dual system encryption" technique that had been used to achieve adaptively secure attribute based encryption.

The Trojan Method. The Trojan Method is a way to embed a hidden functionality thread in an FE secret-key that can only be invoked by special ciphertexts generated using special (secret) back-door information. This thread remains completely unused in the normal operation of the scheme (and can be instantiated with meaningless functionality). In the proof, however, the secret thread will be activated by the challenge ciphertext in such a way that is indistinguishable to the user (= attacker). Namely, the user will not be able

to tell that it is executing the secret thread and not the main thread. This will be extremely beneficial to prove security. We wish to argue that in the view of the user, the execution of the main thread does not allow to distinguish between the encryption of two messages x_0, x_1. The problem is that for functionality purposes, the main thread has to know which input it is working on. This is where the hidden thread comes into the play. We will design the hidden thread so that in the eyes of the user, it is computationally indistinguishable from the main thread on the special messages x_0, x_1. However, in the hidden thread, the output can be computed in a way that does not distinguish between x_0 and x_1 (either by a statistical or a computational argument), which will allow us to conclude that encryptions of x_0, x_1 are indistinguishable.

Technically, the hidden thread is implemented using (standard) symmetric-key encryption, which in turn can be constructed starting with any one-way function. In the functional secret-key generation process for a function f, the secret-key generation process will produce a symmetric-key ciphertext c (which can just be encryption of 0 or another fixed message, since it only needs to have meaningful content in the security proof). It will then consider the function $G_{f,c}$ that takes as input a pair (x, s), and first checks whether it can decrypt c using s as a symmetric key. If it cannot, then it just runs f on x and returns the output. If s actually decrypts c, we consider $f^* = \mathsf{Dec}_s(c)$, and the output is the execution of f^* on x. The value c is therefore used as a Trojan Horse: Its contents are hidden from the users of the scheme, however given a hidden command (in the form of the symmetric s) it can embed functionality that "takes over" the functional secret-key.

We note that in order to support the Trojan method, our FE scheme must support a rich enough class of circuits which allows branch operations, symmetric decryption and execution of the decrypted f^*.

This method can be seen as a weak form of function privacy in FE, but one that can be applied even in the context of public-key FE. In essence, we cannot hide the main thread of the evaluated function (this is unavoidable in public-key FE). However, we can hide the secret thread and thus allow the function to operate in a designated way for specially generated ciphertexts.

For a fairly simple example on how the Trojan method is applied, see the outline for our shallow-to-deep transformation below.

Example: Shallow-to-Deep FE. We use the Trojan method to show that FE that only supports secret-keys for functions with shallow circuits (e.g. logarithmic depth) implies a scheme that works for circuits of arbitrary depth (although with a size bound).

The idea is fairly natural: Instead of producing a secret-key for the function, we produce a key for a shallow *randomized encoding* (RE) of the function (in particular, we use garbled circuits). A randomized encoding [IK00] is a randomized process such that given f, x, $\mathsf{RE}(f, x) = \mathsf{RE}(f, x; \mathsf{rand})$ is a randomized function that can be computed by a shallow circuit (assuming weak PRFs can be computed by shallow circuits). Furthermore, there is an efficient way to retrieve $f(x)$ from $\mathsf{RE}(f, x)$, however nothing except $f(x)$ is revealed. The latter is guaranteed by the existence of a simulator such that $\mathsf{Sim}(f(x)) \cong \mathsf{RE}(f, x)$. We note that a similar approach had been attempted to leverage the supported depth of obfuscators [App13], however it appears to only work for strong notions of obfuscation such as virtual-black-box that are impossible to achieve in general.

Using the Trojan method, we show how this can be achieved for FE. As mentioned above, whenever the scheme asks for a secret-key for f, it will in fact receive a secret-key for a function G that given x, outputs $G_{f,t}(x, k) = \mathsf{RE}(f, x; \mathsf{PRF}_k(t))$. To encrypt a value x, we choose k at random and encrypt the pair (x, k). The decryption process will include applying the functional secret key to obtain $\mathsf{RE}(f, x)$ and computing $f(x)$ from that value. For security we have to show that if x_0, x_1 are such that $f(x_0) = f(x_1)$ then their encryptions are indistinguishable. This is where the Trojan comes into the play. Consider, for starters, the case where x_0 had been encrypted (with some PRF seed k^*). We will insert a Trojan thread into $G_{f,t}$, that, once activated, simply outputs $\mathsf{RE}(f, x_0; \mathsf{PRF}_{k^*}(t))$. Now, the hidden thread outputs exactly the same as the main thread, and therefore switching to the hidden thread will go unnoticed by the user (note that the hidden thread is only activated for the special encryption of (x_0, k^*)). Once the switch had been made, x_0, k^* no longer need to appear in the ciphertext (since all the information about them had been embedded in the hidden thread). The hidden thread is indistinguishable from just outputting $\mathsf{RE}(f, x_0; \mathsf{rand})$, which is in turn

indistinguishable from $\mathsf{Sim}(f(x_0))$, which is in turn identical to $\mathsf{Sim}(f(x_1))$. It follows that the user cannot distinguish between the case where x_0 had been encrypted and the case where x_1 had been encrypted.

Hybrid Functional Encryption. The second technique, in addition to the Trojan method, that we employ to achieve our main result is hybrid encryption. Hybrid encryption is a veteran technique in cryptography which has been used in a variety of settings. We show that in the context of functional encryption it is especially powerful.

The idea in functional encryption is to combine two encryption schemes: An "external" scheme (sometimes called KEM – Key Encapsulation Mechanism) one and in "internal" scheme (sometimes called DEM – Data Encapsulation Mechanism). In order to encrypt a message in the hybrid scheme, a fresh key is generated for the internal scheme, and is used to encrypt the message. Then the key itself is encrypted using the external scheme. The final hybrid ciphertext contains the two ciphertexts: $(\mathsf{Enc}_{\mathsf{ext}}(k), \mathsf{Enc}_{\mathsf{int},k}(m))$ (all external ciphertexts use the same key). To decrypt, one first decrypts the external ciphertext, retrieves k and applies it to the internal ciphertext. Note that if, for example, the external scheme is public-key and the internal is symmetric key, then the resulting scheme will also be public key. Hybrid encryption is often used in cases where the external scheme is less efficient (e.g. in encrypting long messages) and thus there is an advantage in using it to encrypt only a short key, and encrypt the long message using the more efficient internal scheme. Lastly, note that the internal scheme only needs to be able to securely encrypt a single message.

The intuition as to why hybrid encryption may be good for achieving adaptive security is that the external scheme only encrypts keys for the internal scheme. Namely, it only encrypts messages from a predetermined and known distribution, so selectively secure scheme should be enough for the external scheme. The hardness of adaptive security is "pushed" to the internal scheme, but there the task is easier since the internal scheme only needs to be able to encrypt a single message, and it can be private-key rather than public-key.

Let us see how to employ this idea in the case where both internal and external schemes are FE schemes. To encrypt, we will generate a fresh master secret key for the internal scheme, and encrypt it under the external scheme. To generate a key for the function f, the idea is to generate a key for the function $G_f(\mathsf{msk}_{\mathsf{int}})$ which takes a master key for the internal scheme, and outputs a secret key for function f under the internal scheme, using $\mathsf{msk}_{\mathsf{int}}$ (randomness is handled using a PRF as in the shallow-to-deep example). This will allow to decrypt in a two-step process as above. First apply the external secret-key for G_f to the external ciphertext, this will give you an internal secret key for f, which is in turn applied to the internal ciphertext to produce $f(x)$.

For the external scheme, we will use a selectively secure FE scheme (for the sake of concreteness, let us say public-key FE). As explained above, selective security is sufficient here since all the messages encrypted using the external scheme can be generated ahead of time (i.e. they do not depend on the actual x's that the user wishes to encrypt).

For the internal scheme, we require an FE scheme that is *adaptively secure*, but only supports the encryption of a single message. Fortunately, such a primitive can be derived from the works of [GVW12, BS14]. In [GVW12], the authors present an adaptively secure one-time bounded FE scheme. This scheme allows to only generate a key for one function, and to encrypt as many messages as the user wishes. This construction is based on the existence of semantically secure encryption, so the public-key version needs public-key encryption and the symmetric version needs symmetric encryption. While this primitive seems dual to what we need for our purposes, [BS14] shows how to transform private-key FE schemes into *function private* FE. In function-private FE, messages and functions enjoy the same level of privacy, in the sense that a user that produces x_0, x_1, f_0, f_1 such that $f_0(x_0) = f_1(x_1)$ cannot distinguish between $(\mathsf{Enc}(x_0), \mathsf{sk}_{f_0})$ and $(\mathsf{Enc}(x_1), \mathsf{sk}_{f_1})$. Therefore, after applying the [BS14] transformation, we can switch the roles of the functions and messages, and obtain a symmetric FE scheme which is adaptively secure for a single message and many functions. (We note that the symmetric version of the [GVW12] scheme can be shown to be function private even without the [BS14] transformation, however since this claim is not made explicitly in the paper we choose not to rely on it.)

Putting the two components together produces our final scheme. In the proof, we will use the Trojan

method to embed a hidden thread in which $\mathsf{msk}_{\mathsf{int}}$ is not used at all, but rather G_f produces a precomputed internal sk_f. This will allow us to remove $\mathsf{msk}_{\mathsf{int}}$ from the challenge ciphertext and use the security properties of the internal scheme to argue that a internal encryption of x_0, x_1 are identical so long as $f(x_0) = f(x_1)$.

Relation to Dual-System Encryption. Our approach takes some resemblance to the "Dual-System Encryption" method of Waters [Wat09] and followup works [LW10, LW12]. This method had been used to prove adaptive security for Identity Based Encryption and Attribute Based Encryption, based on the hardness of some problems on groups with bilinear-maps. In broad terms, in their proof the distribution of the ciphertext is changed into "semi-functional" mode in a way that is indiscoverable by an observer. A semi-functional ciphertext is still decryptable by normal secret keys. Then, the secret-keys are modified into semi-functional form, which is useless in decrypting semi-functional ciphertexts. This is useful since in IBE and ABE, the challenge ciphertext is not supposed to be decryptable by those keys given to the adversary. Still, a host of algebraic techniques are used to justify the adversary's inability to produce other semi-functional ciphertexts in addition to the challenge, which would foil the reduction.

Our proof technique also requires changing the distributions of the keys and challenge ciphertext. However, there are also major differences. Our modified ciphertext is not allowed to interact with properly generated secret keys, and therefore the distinction between "normal" and "semi-functional" does not fit here. Furthermore, in Identity Based and Attribute Based Encryption, the attacker in the security game is not allowed to receive keys that reveal any information on the message, which allows to generate semi-functional ciphertexts that do not contain any information, whereas in our case, there is a structured and well-defined output for any ciphertext and any key. This means that the information required for decryption (which can be a-priori unbounded) needs to be embedded in the keys. Lastly, our proof is completely generic and does not rely on the algebraic structure of the underlying hardness assumption as in previous implementations of this method.

2 Preliminaries

We let λ denote the security parameter. We say that a function $\mu(\lambda)$ is negligible if for any polynomial $p(\lambda)$ it holds that $\mu(\lambda) < 1/p(\lambda)$ for all sufficiently large $\lambda \in \mathbb{N}$.

Pseudorandom functions. One of the tools that we use in our work are pseudorandom function families. The security property of a pseudorandom function family states that any PPT adversary can distinguish only with negligible probability a function sampled at random from this family from a random function. More formally,

Definition 1. *A function family* $\mathcal{F} = \left\{\mathsf{PRF}_{\mathsf{K}} : \{0,1\}^n \to \{0,1\}^m : \mathsf{K} \xleftarrow{\$} \mathcal{K}\right\}$*, is a* ***pseudorandom function family*** *with* $\mathcal{K}$ *being the space of PRF keys*[1]*, if the following holds for every security parameter* $\lambda \in \mathbb{N}$*, every PPT adversary* $\mathcal{A}$*:*

$$\left|\mathsf{Pr}\left[1 \leftarrow \mathcal{A}^{\mathcal{O}_{(0)}(\mathsf{PRF}_{\mathsf{K}}(\cdot))}(1^\lambda) \;:\; \mathsf{K} \xleftarrow{\$} \mathcal{K}\right] - \mathsf{Pr}\left[1 \leftarrow \mathcal{A}^{\mathcal{O}_{(1)}(\cdot)}(1^\lambda)\right]\right| \leq \mathsf{negl}(\lambda),$$

where the oracle $\mathcal{O}_{(b)}$ *takes as input* x *and outputs* $\mathsf{PRF}_{\mathsf{K}}(x)$ *if* $b = 0$ *else if* $b = 1$ *it outputs* r*, where* r *is picked at random from* $\{0,1\}^m$*.*

Pseudorandom functions can be constructed from one-way functions [GGM86, HILL99].

We say that the pseudorandom function family $\mathcal{F}$, defined above, is implementable in NC^1 if every function in $\mathcal{F}$ can be implemented by a circuit of depth $c \cdot \log(n)$, for some constant c. We also consider a weakening of the above security notion where the oracle is queried at random points instead of the points being adversarially chosen as defined above. This security notion is termed as *weak* pseudorandom functions.

[1] We abuse the notation and interchangeably denote the PRF keyspace to be a set as well as a sampling procedure that produces PRF keys from this set.

Symmetric encryption schemes with pseudorandom ciphertexts. Another tool that we use in our transformation is a symmetric encryption scheme. A symmetric encryption scheme consists of a tuple of PPT algorithms $(\mathsf{Sym.Setup}, \mathsf{Sym.Enc}, \mathsf{Sym.Dec})$. The algorithm $\mathsf{Sym.Setup}$, takes as input a security parameter λ in unary and outputs a key K_E. The encryption algorithm takes as input (K_E, m), where K_E is the symmetric key, m is the message to be encrypted and the output is a ciphertext CT. The decryption algorithm takes as input (K_E, CT), where CT is the ciphertext and the output is the message m, contained in the ciphertext.

For this work, we require a symmetric encryption scheme where ciphertexts produced by $\mathsf{Sym.Enc}$ are pseudorandom strings. More formally, we design a game, parametrized by a PPT adversary that has access to an encryption oracle, $\mathcal{O}_b\big(\mathsf{Sym.Enc}(\mathsf{Sym.K}, \cdot)\big)$, parameterized by bit b. The oracle takes as input a message m and outputs $\mathsf{CT} = \mathsf{Sym.Enc}(\mathsf{Sym.K}, m)$ if $b = 0$, else if $b = 1$ it returns a random string $s \in \{0,1\}^\ell$, where $\ell = |\mathsf{CT}|$. The adversary wins in this game if the output of adversary is $b' = b$. The success probability of the adversary is defined to be $|\mathsf{Pr}[b' = b] - \frac{1}{2}|$. We say that the scheme is secure if the success probability of the adversary is negligible.

We note that such a symmetric encryption scheme can be constructed from one-way functions, e.g. using weak pseudorandom functions by defining $\mathsf{Sym.Enc}(\mathsf{K}, m; r) = (r, \mathsf{PRF}_{\mathsf{K}}(r) \oplus m)$, see [Gol09] for details.

2.1 Public-key Functional Encryption

We recall the definitions of a functional encryption (FE) scheme. As in a traditional encryption scheme, there are two settings for a FE scheme – private-key setting and the public-key setting. In this subsection, we only deal with the the definitions corresponding to a public-key FE scheme and we postpone the definition of private-key FE to the next subsection.

A public-key functional encryption (FE) scheme Π_{Pub}, defined for a class of functions $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ and message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$, is represented by four PPT algorithms, namely ($\mathsf{Pub.Setup}$, $\mathsf{Pub.KeyGen}$, $\mathsf{Pub.Enc}$, $\mathsf{Pub.Dec}$). The input length of any $f \in \mathcal{F}_\lambda$ is the same as the length of any $m \in \mathcal{M}_\lambda$. The description of these four algorithms is given below.

- $\mathsf{Pub.Setup}(1^\lambda)$: It takes as input a security parameter λ in unary and outputs a public key-secret key pair $(\mathsf{MPK}, \mathsf{MSK})$.
- $\mathsf{Pub.KeyGen}(\mathsf{MSK}, f \in \mathcal{F}_\lambda)$: It takes as input a secret key MSK, a function $f \in \mathcal{F}_\lambda$ and outputs a functional key sk_f.
- $\mathsf{Pub.Enc}(\mathsf{MPK}, m \in \mathcal{M}_\lambda)$: It takes as input a public key MPK, a message $m \in \mathcal{M}_\lambda$ and outputs an encryption of m.
- $\mathsf{Pub.Dec}(sk_f, \mathsf{CT})$: It takes as input a functional key sk_f, a ciphertext CT and outputs $\widehat{m}$.

A public-key FE scheme is defined for a complexity class $\mathcal{C}$ if the public-key FE scheme is defined for $\mathcal{F}$, which consists of all the functions that can be implemented by circuits in $\mathcal{C}$.

The correctness notion of a FE scheme dictates that there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all sufficiently large $\lambda \in \mathbb{N}$, for every message $m \in \mathcal{M}_\lambda$, and for every function $f \in \mathcal{F}_\lambda$ it holds that $\Pr\left[f(m) \leftarrow \mathsf{Pub.Dec}(\mathsf{Pub.KeyGen}(\mathsf{MSK}, f), \mathsf{Pub.Enc}(\mathsf{MPK}, m))\right] \geq 1 - \mathsf{negl}(\lambda)$, where $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Pub.Setup}(1^\lambda)$, and the probability is taken over the random choices of all algorithms.

There are different ways to model the security of a functional encryption scheme. The two important notions are – simulation-based notion and the indistinguishability-based notion. In this work, we deal with the indistinguishability-based notion. At a high level, indistinguishability-based notion of security, modeled as a game between the challenger and a PPT adversary, states that the adversary cannot distinguish (with probability significantly different from $1/2$), the encryptions of two messages even after being given functional keys which should correspond to the functions (of adversary's choice) that evaluate to the same output on both the messages – note that this condition is required to prevent the adversary from trivially distinguishing the two ciphertexts.

In the security game, if the adversary is supposed to declare the challenge messages even before it sees the public parameters from the challenger then the FE scheme that satisfies this notion is said to be *selectively-secure*. If the adversary can declare the challenge messages at any time during the game then the FE scheme satisfying such a notion is said to be *adaptively-secure*. Clearly, adaptively-secure FE is at least as strong as selectively-secure FE. We give the formal definitions of a selectively-secure public-key FE as well as an adaptively-secure public-key FE.

Selective security. As remarked before, in the selective security game corresponding to the public-key FE scheme, we place the constraint on the adversary that it has to declare the messages even before it sees the public parameters. More formally,

Definition 2 (Selectively-secure public-key FE)**.** *A public-key functional encryption scheme* $\mathsf{Sel} = (\mathsf{Sel.Setup}, \mathsf{Sel.KeyGen}, \mathsf{Sel.Enc}, \mathsf{Sel.Dec})$ *over a function space* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ *and a message space* $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ *is a* ***selectively-secure public-key functional encryption scheme*** *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* $\mu(\lambda)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, the advantage of* $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}^{\mathsf{Sel}}_{\mathcal{A}} = \left|\mathsf{Pr}[\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, 0) = 1] - \mathsf{Pr}[\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, 1) = 1]\right| \leq \mu(\lambda),$$

where for each $b \in \{0, 1\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, b)$*, modeled as a game between the challenger and the adversary* $\mathcal{A}$*, is defined as follows:*

1. *The adversary submits the challenge message-pair* (m_0, m_1) *to the challenger.*
2. *The challenger executes* $\mathsf{Sel.Setup}(1^\lambda)$ *to obtain* $(\mathsf{Sel.MPK}, \mathsf{Sel.MSK})$*. It then executes* $\mathsf{Sel.Enc}(\mathsf{Sel.MPK}, m_b)$ *to obtain* CT*. The challenger sends* $(\mathsf{Sel.MPK}, \mathsf{CT})$ *to the adversary.*
3. ***Query phase****: For every function query* f *submitted by the adversary, the challenger generates* $\mathsf{Sel}.sk_f$*, where* $\mathsf{Sel}.sk_f$ *is the output of* $\mathsf{Sel.KeyGen}(\mathsf{Sel.MSK}, f)$*. The challenger sends* $\mathsf{Sel}.sk_f$ *only if* $f(m_0) = f(m_1)$*. Otherwise, it aborts.*
4. *The output of the experiment is* b'*, where* b' *is the output of* $\mathcal{A}$*.*

Adaptive security. Unlike the selective security notion, in the case of adaptive security, the adversary can submit the challenge message-pairs at any time during the game. We give the formal definition below.

Definition 3 (Adaptively-secure public-key FE)**.** *A public-key functional encryption scheme* $\mathsf{Ad} = (\mathsf{Ad.Setup}, \mathsf{Ad.KeyGen}, \mathsf{Ad.Enc}, \mathsf{Ad.Dec})$ *over a function space* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ *and a message space* $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ *is an* ***adaptively-secure public-key functional encryption scheme*** *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* $\mu(\lambda)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, the advantage of* $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}^{\mathsf{Ad}}_{\mathcal{A}} = \left|\mathsf{Pr}[\mathsf{Expt}^{\mathsf{Ad}}_{\mathcal{A}}(1^\lambda, 0) = 1] - \mathsf{Pr}[\mathsf{Expt}^{\mathsf{Ad}}_{\mathcal{A}}(1^\lambda, 1) = 1]\right| \leq \mu(\lambda),$$

where for each $b \in \{0, 1\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{Ad}}_{\mathcal{A}}(1^\lambda, b)$*, modeled as a game between the challenger and the adversary* $\mathcal{A}$*, is defined as follows:*

1. *The challenger first executes* $\mathsf{Ad.Setup}(1^\lambda)$ *to obtain* $(\mathsf{Ad.MPK}, \mathsf{Ad.MSK})$*. It then sends* $\mathsf{Ad.MPK}$ *to the adversary.*
2. ***Query Phase I****: The adversary submits a function query* f *to the challenger. The challenger sends back* $\mathsf{Ad}.sk_f$ *to the adversary, where* $\mathsf{Ad}.sk_f$ *is the output of* $\mathsf{Ad.KeyGen}(\mathsf{Ad.MSK}, f)$*.*
3. ***Challenge Phase****: The adversary submits a message-pair* (m_0, m_1) *to the challenger. The challenger checks whether* $f(m_0) = f(m_1)$ *for all function queries* f *made so far. If this is not the case, the challenger aborts. Otherwise, the challenger sends back* $\mathsf{CT} = \mathsf{Ad.Enc}(\mathsf{Ad.MSK}, m_b)$*.*

4. ***Query Phase II***: *The adversary submits a function query f to the challenger. The challenger generates* $\mathsf{Ad}.sk_f$*, where* $\mathsf{Ad}.sk_f$ *is the output of* $\mathsf{Ad.KeyGen}(\mathsf{Ad.MSK}, f)$*. It sends* $\mathsf{Ad}.sk_f$ *to the adversary only if* $f(m_0) = f(m_1)$*, otherwise it aborts.*

5. *The output of the experiment is* b'*, where* b' *is the output of* $\mathcal{A}$*.*

Remark 1. *Even though the definitions talk about single message challenge queries, it can be easily extended to the case when the adversary can ask multiple message queries. In the public key setting, both the definitions are equivalent by a standard hybrid argument.*

2.2 Private-key Functional Encryption

The definitions for private-key FE are practically identical to the public-key setting, except encryption is performed using the master secret key and not the master public key. The definitions become slightly more complicated since the adversary cannot encrypt messages by himself and therefore requires access to an encryption oracle. Even though both notions can be defined in a unified manner, we chose to explicitly define each one separately to allow the reader to only focus on the simpler public-key setting at first.

A private-key functional encryption (FE) scheme Priv, defined for a class of functions $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ and message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$, is represented by four PPT algorithms, namely ($\mathsf{Priv.Setup}$, $\mathsf{Priv.KeyGen}$, $\mathsf{Priv.Enc}$, $\mathsf{Priv.Dec}$). The input length of any $f \in \mathcal{F}_\lambda$ is the same as the length of any $m \in \mathcal{M}_\lambda$.

We give the description of the four algorithms below.

- $\mathsf{Priv.Setup}(1^\lambda)$: It takes as input a security parameter λ in unary and outputs a secret key $\mathsf{Priv.MSK}$.
- $\mathsf{Priv.KeyGen}(\mathsf{Priv.MSK}, f \in \mathcal{F}_\lambda)$: It takes as input a secret key $\mathsf{Priv.MSK}$, a function $f \in \mathcal{F}_\lambda$ and outputs a functional key $\mathsf{Priv}.sk_f$.
- $\mathsf{Priv.Enc}(\mathsf{Priv.MSK}, m \in \mathcal{M}_\lambda)$: It takes as input a secret key $\mathsf{Priv.MSK}$, a message $m \in \mathcal{M}_\lambda$ and outputs an encryption of m.
- $\mathsf{Priv.Dec}(\mathsf{Priv}.sk_f, \mathsf{CT})$: It takes as input a functional key $\mathsf{Priv}.sk_f$, a ciphertext CT and outputs $\hat{m}$.

A private-key FE scheme is defined for a complexity class $\mathcal{C}$ if the private-key FE scheme is defined for $\mathcal{F}$, which consists of all the functions that can be implemented by circuits in $\mathcal{C}$.

The correctness notion of a FE scheme dictates that there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all sufficiently large $\lambda \in \mathbb{N}$, for every message $m \in \mathcal{M}_\lambda$, and for every function $f \in \mathcal{F}_\lambda$ it holds that $\Pr\left[f(m) \leftarrow \mathsf{Priv.Dec}(\mathsf{Priv.KeyGen}(\mathsf{Priv.MSK}, f), \mathsf{Priv.Enc}(\mathsf{Priv.MSK}, m))\right] \geq 1 - \mathsf{negl}(\lambda)$, where $\mathsf{Priv.MSK} \leftarrow \mathsf{Priv.Setup}(1^\lambda)$, and the probability is taken over the random choices of all algorithms.
We define the two security notions, i.e., selective[2] and adaptive, for a private-key FE analogous to the public-key setting. The definitions we state next are taken from Brakerski-Segev [BS14]. We first give the definition of a selectively-secure public-key FE and then we give the definition of an adaptively-secure public-key FE.

Definition 4 (Selectively-secure private-key FE)**.** *A private-key functional encryption scheme* $\mathsf{Sel} = (\mathsf{Sel.Setup}, \mathsf{Sel.KeyGen}, \mathsf{Sel.Enc}, \mathsf{Sel.Dec})$ *over a function space* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ *and a message space* $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ *is a* ***selectively-secure private-key functional encryption scheme*** *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* $\mu(\lambda)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, the advantage of* $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}^{\mathsf{Sel}}_{\mathcal{A}} = \left|\Pr[\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, 1) = 1]\right| \leq \mu(\lambda),$$

where for each $b \in \{0, 1\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, b)$*, modeled as a game between the challenger and the adversary* $\mathcal{A}$*, is defined as follows:*

[2]In the language of Brakerski-Segev, by selective security we mean the property of selective-message message-privacy. They also propose one more notion that is termed as selective-function message-privacy. An equivalence between these two notions have been shown in the same work (under some restrictions).

1. *The challenger first executes* $\mathsf{Sel.Setup}(1^\lambda)$ *to obtain* $\mathsf{Sel.MSK}$.

2. ***Message queries****: The adversary submits the message-pairs* $((m_1^{(0)}, \ldots, m_p^{(0)}), (m_1^{(1)}, \ldots, m_p^{(1)}))$ *to the challenger, where* p *is the number of the message queries (which is a polynomial in the security parameter* λ*) made by the adversary. The challenger then sends* $(c_1^*, \ldots, c_p^*)$ *to* $\mathcal{A}$*, where* c_i^* *is the output of* $\mathsf{Sel.Enc}(\mathsf{Sel.MSK}, m_i^{(b)})$.

3. ***Function queries****: The adversary then can make any number of functional key queries. For every function query* f*, the challenger generates* $\mathsf{Sel}.sk_f$*, where* $\mathsf{Sel}.sk_f$ *is the output of* $\mathsf{Sel.KeyGen}(\mathsf{Sel.MSK}, f)$*. If* $f(m_i^{(0)}) = f(m_i^{(1)})$*, for all* $i \in [p]$*, the challenger sends* $\mathsf{Sel}.sk_f$ *to the adversary. Otherwise, the challenger aborts.*

4. *The output of the experiment is* b'*, where* b' *is the output of* $\mathcal{A}$.

Definition 5 (Adaptively-secure private-key FE)**.** *A private-key functional encryption scheme* $\mathsf{FE} = (\mathsf{Ad.Setup}, \mathsf{Ad.KeyGen}, \mathsf{Ad.Enc}, \mathsf{Ad.Dec})$ *over a function space* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ *and a message space* $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$ *is an* ***adaptively-secure private-key functional encryption scheme*** *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* $\mu(\lambda)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, the advantage of* $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Ad}} = \left| \mathsf{Pr}[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{Ad}}(1^\lambda, 0) = 1] - \mathsf{Pr}[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{Ad}}(1^\lambda, 1) = 1] \right| \leq \mu(\lambda),$$

where for each $b \in \{0, 1\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{Ad}}(1^\lambda, b)$*, modeled as a game between the challenger and the adversary* $\mathcal{A}$*, is defined as follows:*

1. *The challenger first executes* $\mathsf{Ad.Setup}(1^\lambda)$ *to obtain* $\mathsf{Ad.MSK}$*. The adversary makes the following message queries and function queries in no arbitrary order.*

   - ***Message queries****: The adversary submits a message-pair* (m_0, m_1) *to the challenger. The challenger generates* $\mathsf{CT} = \mathsf{Ad.Enc}(\mathsf{Ad.MSK}, m_b)$*. If* $f(m_0) = f(m_1)$ *for all function queries* f *made so far, the challenger sends* CT *to the adversary. Otherwise, it aborts.*
   - ***Function queries****: The adversary submits a function query* f *to the challenger. The challenger generates* $\mathsf{Ad}.sk_f$*, where* $\mathsf{Ad}.sk_f$ *is the output of* $\mathsf{Ad.KeyGen}(\mathsf{Ad.MSK}, f)$*. If* $f(m_0) = f(m_1)$*, for all message-pair queries* (m_0, m_1) *made so far, the challenger sends* $\mathsf{Ad}.sk_f$ *to the adversary. Otherwise, it aborts.*

2. *The output of the experiment is* b'*, where* b' *is the output of* $\mathcal{A}$.

In a traditional functional encryption scheme, the functional keys need not "hide" anything about the function it is computing. However, there are useful scenarios where functional keys need to "hide" the function it is computing – defining what "hide" refers to is tricky and this is especially true if we are in the public-key setting. There have been a few works [SSW09, BRS13a, BRS13b, AAB+13, BS14] which deal with the issue of function-privacy in functional encryption schemes. Of particular interest is the work of Brakerski-Segev [BS14] who formalized the notion of function-privacy in private-key FE schemes. The definitions we give next are taken from their work.

Function privacy. The notion of function privacy is modeled as a game. In the game, a function query made by the adversary is a pair of functions and in response it receives a functional key corresponding to either of the two functions. As long as both the functions are such that they do not split the challenge message-pairs, the adversary should not be able to tell which function was used to generate the functional key. That is, the output of the left function on the left message should be the same as the output of the right function on the right message.

We incorporate the notion of function privacy in the selectively-secure FE and adaptively-secure FE definitions given earlier. First, we define the notion of function-private selective security.

Definition 6 (Function-private selectively-secure FE)**.** *A private-key functional encryption scheme* $\mathsf{Sel} = (\mathsf{Sel.Setup}, \mathsf{Sel.KeyGen}, \mathsf{Sel.Enc}, \mathsf{Sel.Dec})$ *over a function space* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ *and a message space* $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ *is a* ***function-private selectively-secure private-key FE scheme*** *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* $\mu(\lambda)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, the advantage of* $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}^{\mathsf{Sel}}_{\mathcal{A}} = \left|\Pr[\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(1^\lambda, 1) = 1]\right| \leq \mu(\lambda),$$

where for each $b \in \{0,1\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{Sel}}_{\mathcal{A}}(\lambda, b)$*, modeled as a game between the challenger and the adversary* $\mathcal{A}$*, is defined as follows:*

1. *The challenger first executes* $\mathsf{Sel.Setup}(1^\lambda)$ *to obtain* $\mathsf{Sel.MSK}$*.*
2. ***Message queries****: The adversary submits the message-pairs* $((m_1^{(0)}, \ldots, m_p^{(0)}), (m_1^{(1)}, \ldots, m_p^{(1)}))$ *to the challenger, where* p *is the number of the message queries (which is a polynomial in the security parameter* λ*) made by the adversary. The challenger then sends* $(c_1^*, \ldots, c_p^*)$ *to* $\mathcal{A}$*, where* c_i^* *is the output of* $\mathsf{Sel.Enc}(\mathsf{Sel.MSK}, m_i^{(b)})$*.*
3. ***Function queries****: The adversary then makes functional key queries. For every function-pair query* (f_0, f_1)*, the challenger sends* $\mathsf{Sel}.sk_{f_b}$ *to the adversary, where* $\mathsf{Ad}.sk_{f_b}$ *is the output of* $\mathsf{Sel.KeyGen}(\mathsf{Sel.MSK}, f_b)$*, only if* $f_0(m_i^{(0)}) = f_1(m_i^{(1)})$*, for all* $i \in [p]$*. Otherwise, it aborts.*
4. *The output of the experiment is* b'*, where* b' *is the output of* $\mathcal{A}$*.*

We now state the definition of function-private adaptively-secure FE scheme.

Definition 7 (Function-private adaptively-secure FE)**.** *A private-key functional encryption scheme* $\mathsf{FE} = (\mathsf{Ad.Setup}, \mathsf{Ad.KeyGen}, \mathsf{Ad.Enc}, \mathsf{Ad.Dec})$ *over a function space* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ *and a message space* $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ *is a* ***function-private adaptively-secure private-key FE scheme*** *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* $\mu(\lambda)$ *such that for all sufficiently large* $\lambda \in \mathbb{N}$*, the advantage of* $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}^{\mathsf{Ad}}_{\mathcal{A}} = \left|\Pr[\mathsf{Expt}^{\mathsf{Ad}}_{\mathcal{A}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Expt}^{\mathsf{Ad}}_{\mathcal{A}}(1^\lambda, 1) = 1]\right| \leq \mu(\lambda),$$

where for each $b \in \{0,1\}$ *and* $\lambda \in \mathbb{N}$ *the experiment* $\mathsf{Expt}^{\mathsf{Ad}}_{\mathcal{A}}(1^\lambda, b)$*, modeled as a game between the challenger and the adversary* $\mathcal{A}$*, is defined as follows:*

1. *The challenger first executes* $\mathsf{Ad.MSK} \leftarrow \mathsf{Ad.Setup}(1^\lambda)$*. The adversary then makes the following message queries and function queries in no particular order.*
   - ***Message queries****: The adversary submits a message-pair* (m_0, m_1) *to the challenger. In return, the challenger sends back* $\mathsf{CT} = \mathsf{Ad.Enc}(\mathsf{Ad.MSK}, m_b)$*.*
   - ***Function queries****: The adversary then makes functional key queries. For every function-pair query* (f_0, f_1)*, the challenger sends* $\mathsf{Ad}.sk_{f_b}$ *to the adversary, where* $\mathsf{Ad}.sk_{f_b}$ *is the output of* $\mathsf{Ad.KeyGen}(\mathsf{Ad.MSK}, f_b)$ *only if* $f_0(m_0) = f_1(m_1)$*, for all message-pair queries* (m_0, m_1)*. Otherwise, it aborts.*
2. *The output of the experiment is* b'*, where* b' *is the output of* $\mathcal{A}$*.*

Single-key (unbounded-ciphertext) private-key functional encryption scheme. A single-key private-key functional encryption scheme is a functional encryption scheme, where the adversary in the security game (either selective or adaptive) is allowed to query for only one function. Using a single-key private-key functional encryption scheme, we can obtain a bounded-key functional encryption scheme where the adversary can make a bounded number of queries and the bound needs to be specified at the start of the game by the adversary – the parameters in the scheme would grow proportional to this bound. In this work we are interested in an adaptively-secure single-key functional encryption scheme in the private key setting. There are several known constructions [SS10, GKP+12] and we will consider the construction that can be based on one-way functions [GVW12].

3 From Selective FE to Adaptive FE

This section contains our main transformation. We start by introducing our main building block: An FE scheme which is adaptively secure for a single message query and many ciphertext queries (see Definition 8 below). We show that based on the works of [GVW12, BS14], this primitive can be based on any one-way function (see Corollary 1 below). We then present our selective to adaptive transformation. Even though the transformation is identical in the private-key and public-key settings, we present the public-key case first in Section 3.1 since the proof is slightly simpler. The private-key case is then described in Section 3.2.

Definition 8. *A private-key FE scheme* OneCT *is an* ***adaptively-secure single-ciphertext private-key functional encryption scheme*** *if it satisfies Definition 5 with respect to adversaries that are only allowed to make a single message query.*

This definition can be satisfied based solely on the existence of one-way functions, as shown in the next corollary.

Corollary 1 ([GVW12, BS14])**.** *An adaptively-secure single-ciphertext private-key FE scheme as per Definition 8 can be based on any one-way function.*

Proof. It is shown in [GVW12] how to construct a public-key adaptively-secure single-function FE scheme, based on any public-key encryption scheme. This is in a sense the dual to what we need here. Our first observation is that if we only require a private-key FE with the same properties, then the public-key encryption scheme can be replaced with a private-key encryption scheme which in turn can be constructed from any one-way function.

In order to "invert" the roles of the message and ciphertext and obtain a scheme as per Definition 8, we first make the scheme function private using the reduction of [BS14], which adds no additional assumptions.

Finally, we use the observation (that had been made before, e.g. in [AAB+13, BS14]) that in a function-private FE scheme, the roles of the functions and messages can be inverted (using a universal circuit to encode messages as functions). This completes the proof. □

3.1 Our Transformation in the Public-Key Setting

We show how to construct an adaptively-secure public-key FE starting from selectively-secure public-key FE and single-ciphertext FE. We use the following ingredients:

1. Selectively-secure public-key FE scheme, $\mathsf{Sel} = (\mathsf{Sel.Setup}, \mathsf{Sel.KeyGen}, \mathsf{Sel.Enc}, \mathsf{Sel.Dec})$.
2. Single-ciphertext FE scheme, $\mathsf{OneCT} = (\mathsf{OneCT.Setup}, \mathsf{OneCT.KeyGen}, \mathsf{OneCT.Enc}, \mathsf{OneCT.Dec})$.
3. Symmetric encryption scheme with pseudorandom ciphertexts, $\mathsf{SYM} = (\mathsf{Sym.Setup}, \mathsf{Sym.Enc}, \mathsf{Sym.Dec})$.
4. Pseudorandom function family denoted by $\mathcal{F}$ and the space of PRF keys is represented by $\mathcal{K}$.

The construction of adaptively-secure public-key FE, denoted by Ad, is as follows.

$\underline{\mathsf{Ad.Setup}(1^\lambda)}$: Execute $\mathsf{Sel.Setup}(1^\lambda)$ to obtain $(\mathsf{Sel.MPK}, \mathsf{Sel.MSK})$. Output $(\mathsf{Ad.MPK} = \mathsf{Sel.MPK}, \mathsf{Ad.MSK} = \mathsf{Sel.MSK})$.

$\underline{\mathsf{Ad.KeyGen}(\mathsf{Ad.MSK} = \mathsf{Sel.MSK}, f)}$:

- Pick a uniformly random string $C_E \leftarrow \{0,1\}^{\ell_1(\lambda)}$ and a uniformly random tag $\tau \leftarrow \{0,1\}^{\ell_2(\lambda)}$.
- Define the circuit

$$G_{f,C_E,\tau}(\mathsf{OneCT.SK}, \mathsf{K}, \mathsf{Sym.K}, \beta) = \begin{cases} \mathsf{OneCT}.sk_f \leftarrow \mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}, f; \mathsf{PRF}_{\mathsf{K}}(\tau)) & \text{if } \beta = 0 \\ \mathsf{Sym.Dec}(\mathsf{Sym.K}, C_E) & \text{if } \beta = 1 \end{cases}$$

  where $C_E \in \{0,1\}^{\ell_1(\lambda)}$ and $\tau \in \{0,1\}^{\ell_2(\lambda)}$ are as above. Furthermore, $\mathsf{K}, \mathsf{Sym.K} \in \{0,1\}^\lambda$, and $\beta \in \{0,1\}$.

- Run $\mathsf{Sel}.sk_G \leftarrow \mathsf{Sel.KeyGen}(\mathsf{Sel.MSK}, G_{f,C_E,\tau})$ and output $\mathsf{Ad}.sk_f = \mathsf{Sel}.sk_G$.

$\underline{\mathsf{Ad.Enc}(\mathsf{Ad.MPK} = \mathsf{Sel.MPK}, m)}$:

- Execute $\mathsf{OneCT.Setup}(1^\lambda)$ to obtain $\mathsf{OneCT.SK}$.
- Sample K from the appropriate PRF key space.
- Execute $\mathsf{OneCT.Enc}(\mathsf{OneCT.SK}, m)$ to obtain CT_0. Execute $\mathsf{Sel.Enc}(\mathsf{Sel.MPK}, M = (\mathsf{OneCT.SK}, \mathsf{K}, 0^\lambda, 0))$ to obtain CT_1.
- Output $\mathsf{CT} = (\mathsf{CT}_0, \mathsf{CT}_1)$.

$\underline{\mathsf{Ad.Dec}(\mathsf{Ad}.sk_f = \mathsf{Sel}.sk_G, \mathsf{CT} = (\mathsf{CT}_0, \mathsf{CT}_1))}$: Execute $\mathsf{Sel.Dec}(\mathsf{Sel}.sk_G, \mathsf{CT}_1)$ to obtain $\mathsf{OneCT}.sk_f$. Execute $\mathsf{OneCT.Dec}(\mathsf{OneCT}.sk_f, \mathsf{CT}_0)$ to obtain $\hat{m}$.

We now show that the above scheme satisfies both the correctness and the security properties.

Correctness. Suppose $\mathsf{CT} = (\mathsf{CT}_0, \mathsf{CT}_1)$ is a valid[3] encryption of m and $\mathsf{Ad}.sk_f = \mathsf{Sel}.sk_G$ is a valid functional key of f, then we claim that the output of $\mathsf{Ad.Dec}(\mathsf{Ad}.sk_f, \mathsf{CT})$ is $f(m)$. If CT is a valid encryption of m then, CT_0 is a valid adaptively-secure single-ciphertext FE encryption of m and CT_1 is a valid selectively-secure FE encryption of $(\mathsf{OneCT.SK}, \mathsf{K}, 0^\lambda, 0)$, where $\mathsf{OneCT.SK}$ and K are picked as in the description of $\mathsf{Ad.Enc}$. Further, if $\mathsf{Ad}.sk_f$ is a valid functional key of f then $\mathsf{Sel}.sk_G$ is a valid selectively-secure FE functional key of G. Hence, by the correctness of the selectively-secure FE scheme, we have the output of $\mathsf{Sel.Dec}(\mathsf{Sel}.sk_G, \mathsf{CT}_1)$ to be $G(\mathsf{OneCT.SK}, \mathsf{K}, 0^\lambda, 0)$. By the description of G, this in turn is a valid single-ciphertext FE functional key of f, denoted by $\mathsf{OneCT}.sk_f$. Now, the output of $\mathsf{OneCT.Dec}(\mathsf{OneCT}.sk_f, \mathsf{CT}_0)$ yields $f(m)$ from the correctness of single-ciphertext FE scheme. Since, the output of $\mathsf{OneCT.Dec}$ is the output of $\mathsf{Ad.Dec}$, the correctness follows.

Security. We show that any PPT adversary $\mathcal{A}$ succeeds in the adaptive security game of Ad with only negligible probability. We will show this in a sequence of hybrids. We denote the advantage of the adversary in $\mathsf{Hybrid}_{i.b}$ to be the probability that the adversary outputs 1 in this hybrid and this quantity is denoted by $\mathsf{Adv}^{\mathcal{A}}_{i.b}$. For $b \in \{0,1\}$, we define the following hybrids.

$\underline{\mathsf{Hybrid}_{1.b}}$: This corresponds to the real experiment when the challenger encrypts the b^{th} message in the message pair submitted by the adversary. More precisely, if the adversary submits the message pair (m_0, m_1) to the challenger, the challenger then sends the challenge ciphertext CT^* back to the adversary, where CT^* is the encryption of message m_b. The output of this hybrid is the same as the output of the adversary.

$\underline{\mathsf{Hybrid}_{2.b}}$: The challenger replaces C_E in every functional key (each key has a different C_E), corresponding to the query f made by the adversary, with a symmetric key encryption of $\mathsf{OneCT}.sk_f$. Here, $\mathsf{OneCT}.sk_f$ is the output of $\mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; \mathsf{PRF}_{\mathsf{K}^*}(\tau))$ and K^* is a PRF key drawn from the keyspace $\mathcal{K}$. Further, the symmetric encryption is computed with respect to $\mathsf{Sym.K}^*$, where $\mathsf{Sym.K}^*$ is the output of $\mathsf{Sym.Setup}(1^\lambda)$ and τ is the tag associated to the functional key of f. We emphasize that the same $\mathsf{Sym.K}^*$ and K^* are used while generating all the functional keys. Further, the challenger generates the challenge ciphertext CT^* to be $(\mathsf{CT}^*_0, \mathsf{CT}^*_1)$, where CT^*_0 is the output of $\mathsf{OneCT.Enc}(\mathsf{OneCT.SK}^*, m_b)$ and CT^*_1 is the output of $\mathsf{Sel.Enc}(\mathsf{Sel.MSK}, (\mathsf{OneCT.SK}^*, \mathsf{K}^*, 0^\lambda, 0))$. The rest of the hybrid is the same as the previous hybrid, $\mathsf{Hybrid}_{1.b}$.

Claim 1. *Assuming the pseudorandom ciphertexts property of* SYM*, for every PPT adversary* $\mathcal{A}$*, for* $b \in \{0,1\}$*, we have* $|\mathsf{Adv}^{\mathcal{A}}_{1.b} - \mathsf{Adv}^{\mathcal{A}}_{2.b}| \leq \mathsf{negl}(\lambda)$.

[3] We say that a ciphertext (resp., a functional key) is *valid* if it is produced as an output of the encryption (resp., key generation) algorithm on some randomness.

Proof. Suppose there exists an adversary such that the difference in the advantages is non-negligible, then we construct a reduction that can break the security of SYM. The reduction internally executes the adversary by simulating the role of the challenger in the adaptive public-key FE game. It answers both the message and the functional queries made by the adversary as follows. The reduction first executes $\mathsf{OneCT.Setup}(1^\lambda)$ to obtain $\mathsf{OneCT.SK}^*$. It then samples K^* from $\mathcal{K}$. Further, the reduction generates $\mathsf{Sel.MSK}$, which is the output of $\mathsf{Sel.Setup}(1^\lambda)$ and $\mathsf{Sym.K}^*$, which is the output of $\mathsf{Sym.Setup}(1^\lambda)$. When the adversary submits a functional query f, the reduction first picks τ at random. The reduction executes $\mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; \mathsf{PRF}(\mathsf{K}^*(\tau)))$ to obtain $\mathsf{OneCT}.sk_f$. It then sends $\mathsf{OneCT}.sk_f$ to the challenger of the symmetric encryption scheme. The challenger returns back with C_E, where C_E is either a uniformly random string or it is an encryption of $\mathsf{OneCT}.sk_f$. The reduction then generates a selectively-secure FE functional key of $G_{f,C_E,\tau}$ and denote the result by $\mathsf{Sel}.sk_G$ which is sent to the adversary. The message queries made by the adversary are handled as in Hybrid_1. That is, the adversary submits the message-pair query (m_0, m_1) and the reduction sends $\mathsf{CT}^* = (\mathsf{CT}_0^*, \mathsf{CT}_1^*)$ back to the adversary, where $\mathsf{CT}_0^* = \mathsf{OneCT.Enc}(\mathsf{OneCT.SK}^*, m_b)$ and $\mathsf{CT}_1^* = \mathsf{Sel.Enc}(\mathsf{Sel.MSK}, (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1))$.

If the challenger of the symmetric key encryption scheme sends a uniformly random string back to the reduction every time the reduction makes a query to the challenger then we are in $\mathsf{Hybrid}_{1.b}$, otherwise we are in $\mathsf{Hybrid}_{2.b}$. Since the adversary can distinguish both the hybrids with non-negligible probability, we have that the reduction breaks the security of the symmetric key encryption scheme with non-negligible probability. From our hypothesis, we have that the reduction breaks the security of the symmetric key encryption scheme with non-negligible probability. This proves the claim. □

$\underline{\mathsf{Hybrid}_{3.b}}$: The challenger modifies the challenge ciphertext $\mathsf{CT}^* = (\mathsf{CT}_0^*, \mathsf{CT}_1^*)$. It generates CT_1^* to be an encryption of the message $(0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1)$. The ciphertext component CT_0^* is generated the same way as before. In more detail, the challenge ciphertext is now $\mathsf{CT}^* = (\mathsf{CT}_0^* = \mathsf{OneCT.Enc}(\mathsf{OneCT.SK}^*, m_b)$, $\mathsf{CT}_1^* = \mathsf{Sel.Enc}(\mathsf{Sel.MPK}, (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1))$. The rest of the hybrid is the same as the previous hybrid, $\mathsf{Hybrid}_{2.b}$.

Claim 2. *Assuming the selective security of* Sel*, for every PPT adversary* $\mathcal{A}$*, for* $b \in \{0,1\}$*, we have* $|\mathsf{Adv}_{2.b}^{\mathcal{A}} - \mathsf{Adv}_{3.b}^{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Suppose the claim is not true for some adversary $\mathcal{A}$, we construct a reduction that breaks the security of Sel. Our reduction will internally execute $\mathcal{A}$ by simulating the role of the challenger of the adaptive FE game.

Our reduction first executes $\mathsf{OneCT.Setup}(1^\lambda)$ to obtain $\mathsf{OneCT.SK}^*$. It then samples K^* from $\mathcal{K}$. It also executes $\mathsf{Sym.Setup}(1^\lambda)$ to obtain $\mathsf{Sym.K}^*$. The reduction then sends the message pair $\big((\mathsf{OneCT.SK}^*, \mathsf{K}^*, 0^\lambda, 0), (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1)\big)$ to the challenger of the selective game. The challenger replies back with the public key $\mathsf{Sel.MPK}$ and the challenge ciphertext CT_1^*. The reduction is now ready to interact with the adversary $\mathcal{A}$. If $\mathcal{A}$ makes a functional query f then the reduction constructs the circuit $G_{f,C_E,\tau}$ as in $\mathsf{Hybrid}_{2.b}$. It then queries the challenger of the selective game with the function G and in return it gets the key $\mathsf{Sel}.sk_G$. The reduction then sets $\mathsf{Ad}.sk_f$ to be $\mathsf{Sel}.sk_G$ which it then sends back to $\mathcal{A}$. If $\mathcal{A}$ submits a message pair (m_0, m_1), the reduction executes $\mathsf{OneCT.Enc}(\mathsf{OneCT.SK}^*, m_0)$ to obtain CT_0^*. It then sends the ciphertext $\mathsf{CT}^* = (\mathsf{CT}_0^*, \mathsf{CT}_1^*)$ to the adversary. The output of the reduction is the output of $\mathcal{A}$.

We claim that the reduction is a legal adversary in the selective security game of Sel, i.e., for challenge message query $(M_0 = (\mathsf{OneCT.SK}^*, \mathsf{K}^*, 0^\lambda, 0)$, $M_1 = (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1))$ and every functional query of the form $G_{f,C_E,\tau}$ made by the reduction, we have that $G_{f,C_E,\tau}(M_0) = G_{f,C_E,\tau}(M_1)$: By definition, $G_{f,C_E,\tau}(M_0)$ is the functional key of f, with respect to key $\mathsf{OneCT.SK}^*$ and randomness $\mathsf{PRF}_{\mathsf{K}^*}(\tau)$. Further, $G_{f,C_E,\tau}(M_1)$ is the decryption of C_E which is nothing but the functional key of f, with respect to key $\mathsf{OneCT.SK}^*$ and randomness $\mathsf{PRF}_{\mathsf{K}^*}(\tau)$. This proves that the reduction is a legal adversary in the selective security game.

If the challenger of the selective game sends back an encryption of $(\mathsf{OneCT.SK}^*, \mathsf{K}^*, 0^\lambda, 0)$ then we are in $\mathsf{Hybrid}_{2.b}$ else if the challenger encrypts $(0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1)$ then we are in $\mathsf{Hybrid}_{3.b}$. By our hypothesis, this means the reduction breaks the security of the selective game with non-negligible probability that contradicts the security of Sel. This completes the proof of the claim. □

$\underline{\mathsf{Hybrid}_{4.b}}$: For every functional query f made by the adversary, the challenger generates C_E by executing $\overline{\mathsf{Sym.Enc}}(\mathsf{Sym.K}^*, \mathsf{OneCT}.sk_f)$, with $\mathsf{OneCT}.sk_f$ being the output of $\mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; R)$, where R is picked at random. The rest of the hybrid is the same as the previous hybrid.

Claim 3. *Assuming the security of the pseudorandom function family $\mathcal{F}$, for every PPT adversary $\mathcal{A}$, for $b \in \{0,1\}$, we have $|\mathsf{Adv}^{\mathcal{A}}_{3.b} - \mathsf{Adv}^{\mathcal{A}}_{4.b}| \leq \mathsf{negl}(\lambda)$.*

Proof. Suppose the claim is false for some PPT adversary $\mathcal{A}$, we construct a reduction that internally executes $\mathcal{A}$ and breaks the security of the pseudorandom function family $\mathcal{F}$. The reduction simulates the role of the challenger of the adaptive game when interacting with $\mathcal{A}$. The reduction answers the functional queries, made by the adversary as follows; the message queries are answered as in $\mathsf{Hybrid}_{3.b}$ (or $\mathsf{Hybrid}_{4.b}$). For every functional query f made by the adversary, the reduction picks τ at random which is then forwarded to the challenger of the PRF security game. In response it receives R^*. The reduction then computes C_E to be $\mathsf{Sym.Enc}(\mathsf{Sym.K}^*, \mathsf{OneCT}.sk_f)$, where $\mathsf{OneCT}.sk_f = \mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; R^*)$. The reduction then proceeds as in the previous hybrids to compute the functional key $\mathsf{Ad}.sk_f$ which it then sends to $\mathcal{A}$.

If the challenger of the PRF game sent $R^* = \mathsf{PRF}_{\mathsf{K}^*}(\tau)$ back to the reduction then we are in $\mathsf{Hybrid}_{3.b}$ else if R^* is generated at random by the challenger then we are in $\mathsf{Hybrid}_{4.b}$. From our hypothesis this means that the probability that the reduction distinguishes the pseudorandom value from random (at the point τ) is non-negligible, contradicting the security of the pseudorandom function family $\mathcal{F}$. □

We now show that $\mathsf{Hybrid}_{4.0}$ is computationally indistinguishable from $\mathsf{Hybrid}_{4.1}$.

Claim 4. *Assuming the adaptive security of OneCT, for every PPT adversary $\mathcal{A}$ we have $|\mathsf{Adv}^{\mathcal{A}}_{4.0} - \mathsf{Adv}^{\mathcal{A}}_{4.1}| \leq \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$, such that the claim is false. We design a reduction $\mathcal{B}$ that internally executes $\mathcal{A}$ to break the adaptive security of OneCT.

The reduction simulates the role of the challenger of the adaptive public-key FE game. It answers both the functional as well as message queries made by the adversary as follows. If $\mathcal{A}$ makes a functional query f then it forwards it to the challenger of the adaptively-secure single-ciphertext FE scheme. In return it receives $\mathsf{OneCT}.sk_f$. It then encrypts it using the symmetric encryption scheme, where the symmetric key is picked by the reduction itself, and denote the resulting ciphertext to be C_E. The reduction then constructs the circuit $G_{f,C_E,\tau}$, with τ being picked at random, as in the previous hybrids. Finally, the reduction computes the selective public-key functional key of $G_{f,C_E,\tau}$, where the reduction itself picks the master secret key of selective public-key FE scheme. The resulting functional key is then sent to $\mathcal{A}$. If $\mathcal{A}$ makes a message-pair query (m_0, m_1), the reduction forwards this message pair to the challenger of the adaptive game. In response it receives CT^*_0. The reduction then generates CT^*_1 on its own where CT^*_1 is the selective FE encryption of $(0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1)$. The reduction then sends $\mathsf{CT}^* = (\mathsf{CT}^*_0, \mathsf{CT}^*_1)$ to $\mathcal{A}$. The output of the reduction is the output of $\mathcal{A}$.

We note that the reduction is a legal adversary in the adaptive game of OneCT, i.e., for every challenge message query (m_0, m_1), functional query f, we have that $f(m_0) = f(m_1)$: this follows from the fact that (i) the functional queries (resp., challenge message query) made by the adversary (of Ad) is the same as the functional queries (resp., challenge message query) made by the reduction, and (ii) the adversary (of Ad) is a legal adversary. This proves that the reduction is a legal adversary in the adaptive game.

If the challenger sends an encryption of m_0 then we are in $\mathsf{Hybrid}_{4.0}$ and if the challenger sends an encryption of m_1 then we are in $\mathsf{Hybrid}_{4.1}$. From our hypothesis, this means that the reduction breaks the security of OneCT. This proves the claim. □

The above claims implies that $\mathsf{Hybrid}_{1.0}$ is computationally indistinguishable from $\mathsf{Hybrid}_{1.1}$ which proves the adaptive security of Ad. We thus have the following theorem.

Theorem 2. *There exists an adaptively-secure public-key FE scheme assuming the existence of a sufficiently-expressive selectively-secure public-key FE scheme, adaptively-secure single-ciphertext private-key FE scheme, pseudorandom functions and a symmetric encryption scheme.*

As remarked earlier, single-ciphertext FE, pseudorandom functions and symmetric encryption schemes are implied by one-way functions which in turn is implied by any public-key encryption scheme and in particular, a selectively-secure public-key FE scheme. Thus, we have the following corollary.

Corollary 2. *Assuming the existence of a sufficiently-expressive selectively-secure public-key FE, there exists adaptively-secure public-key FE.*

3.2 Our Transformation in the Private-Key Setting

The exact same transformation as above works in the private-key setting as well. Namely, given a private-key selectively secure FE, we obtain a private-key adaptively secure FE. The transformation is identical with the obvious exception that there is no public-key, and the master secret key is used for both encryption and key generation. We denote the selectively-secure FE that we use by $\mathsf{Sel} = (\mathsf{Sel.Setup}, \mathsf{Sel.KeyGen}, \mathsf{Sel.Enc}, \mathsf{Sel.Dec})$. The adaptively-secure FE that we construct is denoted by $\mathsf{Ad} = (\mathsf{Ad.Setup}, \mathsf{Ad.KeyGen}, \mathsf{Ad.Enc}, \mathsf{Ad.Dec})$.

$\underline{\mathsf{Ad.Setup}(1^\lambda)}$: Execute $\mathsf{Sel.Setup}(1^\lambda)$ to obtain $\mathsf{Sel.MSK}$. Output $\mathsf{Ad.MSK} = \mathsf{Sel.MSK}$.

$\underline{\mathsf{Ad.KeyGen}(\mathsf{Ad.MSK} = \mathsf{Sel.MSK}, f)}$:

- Pick a uniformly random string $C_E \leftarrow \{0,1\}^{\ell_1(\lambda)}$ and a uniformly random tag $\tau \leftarrow \{0,1\}^{\ell_2(\lambda)}$.
- Define the circuit

  $$G_{f,C_E,\tau}(\mathsf{OneCT.SK}, \mathsf{K}, \mathsf{Sym.K}, \beta) = \begin{cases} \mathsf{OneCT}.sk_f \leftarrow \mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}, f; \mathsf{PRF}_{\mathsf{K}}(\tau)) & \text{if } \beta = 0 \\ \mathsf{Sym.Dec}(\mathsf{Sym.K}, C_E) & \text{if } \beta = 1 \end{cases}$$

  where $C_E \in \{0,1\}^{\ell_1(\lambda)}$ and $\tau \in \{0,1\}^{\ell_2(\lambda)}$ are as above. Furthermore, $\mathsf{K}, \mathsf{Sym.K} \in \{0,1\}^\lambda$, and $\beta \in \{0,1\}$.
- Run $\mathsf{Sel}.sk_G \leftarrow \mathsf{Sel.KeyGen}(\mathsf{Sel.MSK}, G_{f,C_E,\tau})$ and outputs $\mathsf{Ad}.sk_f = \mathsf{Ad}.sk_G$.

$\underline{\mathsf{Ad.Enc}(\mathsf{Ad.MSK} = \mathsf{Sel.MSK}, m)}$:

- Execute $\mathsf{OneCT.Setup}(1^\lambda)$ to obtain $\mathsf{OneCT.SK}$.
- Sample K from the appropriate PRF key space.
- Execute $\mathsf{OneCT.Enc}(\mathsf{OneCT.SK}, m)$ to obtain CT_0. Execute $\mathsf{Sel.Enc}(\mathsf{Sel.MSK}, M = (\mathsf{OneCT.SK}, \mathsf{K}, 0^\lambda, 0))$ to obtain CT_1.
- Output $\mathsf{CT} = (\mathsf{CT}_0, \mathsf{CT}_1)$.

$\underline{\mathsf{Ad.Dec}(\mathsf{Ad}.sk_f = \mathsf{Sel}.sk_G, \mathsf{CT} = (\mathsf{CT}_0, \mathsf{CT}_1))}$: Execute $\mathsf{Sel.Dec}(\mathsf{Sel}.sk_G, \mathsf{CT}_1)$ to obtain $\mathsf{OneCT}.sk_f$. Execute $\mathsf{OneCT.Dec}(\mathsf{OneCT}.sk_f, \mathsf{CT}_0)$ to obtain $\hat{m}$.

The correctness is straightforward. The proof of security in this case is slightly more complicated than its public-key counterpart. Since in the symmetric setting, the adversary is allowed to make multiple message queries, we have to employ a sequence of hybrids, handling each message query at a time. Each of these hybrids is identical to our proof of Theorem 2 above.

Security. We show that any PPT adversary $\mathcal{A}$ succeeds in the adaptive security game of Ad with only negligible probability. We will show this in a sequence of hybrids. We denote the advantage of the adversary in Hybrid_i^j to be the probability that the adversary outputs 1 in that hybrid and this quantity is denoted by $\mathsf{Adv}_{i,j}^{\mathcal{A}}$.

We define the hybrids in the following manner and we prove indistinguishability of every two consecutive hybrids (two hybrids are consecutive if they are connected by an arrow). The text on top of the arrow indicates the claim we use to prove the indistinguishability. The symbol "=" on top of the arrow indicates that the consecutive hybrids are identical. We denote by p, the number of message queries made by $\mathcal{A}$.

$$\mathsf{Hybrid}_0 \xrightarrow{=} \mathsf{Hybrid}_{1.0}^1 \xrightarrow{\text{Claim 5}} \mathsf{Hybrid}_{2.0}^1 \xrightarrow{\text{Claim 6}} \mathsf{Hybrid}_{3.0}^1 \xrightarrow{\text{Claim 7}} \mathsf{Hybrid}_{4.0}^1$$
$$\cdots \mathsf{Hybrid}_{4.0}^1 \xrightarrow{\text{Claim 8}} \mathsf{Hybrid}_{4.1}^1 \xrightarrow{\text{Claim 7}} \mathsf{Hybrid}_{3.1}^1 \xrightarrow{\text{Claim 6}} \mathsf{Hybrid}_{2.1}^1 \xrightarrow{\text{Claim 5}} \mathsf{Hybrid}_{1.1}^1$$
$$\cdots \mathsf{Hybrid}_{1.1}^1 \xrightarrow{=} \mathsf{Hybrid}_{1.0}^2 \cdots\cdots \mathsf{Hybrid}_{1.1}^p \xrightarrow{=} \mathsf{Hybrid}_5$$

Hybrid_0: This corresponds to the real experiment when the challenger uses the encryption oracle, parameterized by bit 0, to generate the challenge ciphertexts. That is, for all message queries of the form (m_0, m_1), the challenger sends an encryption of m_0 to the adversary. The output of this hybrid is the same as the output of the adversary.

$\mathsf{Hybrid}_{1.b}^j$ for $b \in \{0,1\}, j \in [p]$: This is the same as the hybrid $\mathsf{Hybrid}_{1.b}^{j-1}$ (if $j = 1$ then we refer to Hybrid_0) except that the challenger encrypts the b^{th} message in the j^{th} message pair query submitted by the adversary. More precisely, the only change is the following: If the adversary submits the j^{th} message pair (m_0, m_1) to the challenger, the challenger then sends the challenge ciphertext CT^* back to the adversary, where CT^* is the encryption of message m_b.

We observe that the hybrid $\mathsf{Hybrid}_{1.0}^1$ is identical to Hybrid_0 and also, $\mathsf{Hybrid}_{1.1}^{j-1}$ is identical to $\mathsf{Hybrid}_{1.0}^j$, for $j \in [p]$ and $j > 1$.

$\mathsf{Hybrid}_{2.b}^j$ for $b \in \{0,1\}, j \in [p]$: This is identical to the previous hybrid, $\mathsf{Hybrid}_{1.b}^j$, except for the following change. The challenger replaces C_E in every functional key, corresponding to the query f made by the adversary, with a symmetric encryption of $\mathsf{OneCT}.sk_f$, where $\mathsf{OneCT}.sk_f$ is the output of $\mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; \mathsf{PRF}_{\mathsf{K}^*}(\tau))$ and K^* is a PRF key sampled from the keyspace $\mathcal{K}$. Further the symmetric encryption is computed with respect to $\mathsf{Sym.K}^*$, where $\mathsf{Sym.K}^*$ is the output of $\mathsf{Sym.Setup}(1^\lambda)$ and τ is the tag associated to the functional key of f. We emphasize that the same $\mathsf{Sym.K}^*$ and K^* is used while generating all the functional keys.

Claim 5. *Assuming the pseudorandom ciphertexts property of* SYM*, for every PPT adversary* $\mathcal{A}$*, for* $b \in \{0,1\}$*,* $j \in [p]$*, we have* $|\mathsf{Adv}_{1.b,j}^{\mathcal{A}} - \mathsf{Adv}_{2.b,j}^{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Suppose there exists an adversary such that the difference in the advantages is non-negligible, then we construct a reduction that can break the security of SYM. The reduction internally executes the adversary by simulating the role of the challenger in the adaptive private-key FE game. It answers both the message and the functional queries made by the adversary as follows. The reduction first executes $\mathsf{OneCT.Setup}(1^\lambda)$ to obtain $\mathsf{OneCT.SK}^*$ It then samples K^* from $\mathcal{K}$. Further, the reduction generates $\mathsf{Sel.MSK}$, which is the output of $\mathsf{Sel.Setup}(1^\lambda)$ and $\mathsf{Sym.K}^*$, which is the output of $\mathsf{Sym.Setup}(1^\lambda)$. When the adversary submits a functional query f, the reduction first picks τ at random. The reduction executes $\mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; \mathsf{PRF}(\mathsf{K}^*(\tau)))$ to obtain $\mathsf{OneCT}.sk_f$. It then sends $\mathsf{OneCT}.sk_f$ to the challenger of the symmetric encryption scheme. The challenger returns back with C_E, where C_E is either a uniformly random string or it is an encryption of $\mathsf{OneCT}.sk_f$. The reduction then generates a selectively-secure FE functional key of $G_{f,C_E,\tau}$ and denote the result by $\mathsf{Sel}.sk_G$ which is sent to the adversary. The message queries made by the adversary are handled as in $\mathsf{Hybrid}_{1.b}^j$. That is, the adversary submits the i^{th} message-pair query of the form (m_0^i, m_1^i) and the reduction sends $\mathsf{CT}^* = (\mathsf{CT}_0^*, \mathsf{CT}_1^*)$ back to the adversary, where $\mathsf{CT}_0^* = \mathsf{OneCT.Enc}(\mathsf{OneCT.SK}^*, m_{b_i})$ and $\mathsf{CT}_1^* = \mathsf{Sel.Enc}(\mathsf{Sel.MSK}, (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1))$; we define $b_i = 1$ for $i < j$, $b_j = b$ and for $i > j$, we have $b_i = 0$.

If the challenger of the symmetric key encryption scheme sends a uniformly random string back to the reduction every time the reduction makes a query to the challenger then we are in $\mathsf{Hybrid}^j_{1.b}$, otherwise we are in $\mathsf{Hybrid}^j_{2.b}$. Since the adversary can distinguish both the hybrids with non-negligible probability, we have that the reduction breaks the security of the symmetric key encryption scheme with non-negligible probability. This proves the claim. □

$\underline{\mathsf{Hybrid}^j_{3.b} \text{ for } b \in \{0,1\}, j \in [p]}$: The challenger modifies the j^{th} challenge ciphertext $\mathsf{CT}^* = (\mathsf{CT}^*_0, \mathsf{CT}^*_1)$. In particular it generates CT^*_1 using the message $(0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1)$ instead of $(\mathsf{OneCT.SK}^*, \mathsf{K}^*, 0^\lambda, 0)$. The ciphertext component CT^*_0 is generated the same way as in the previous hybrid, $\mathsf{Hybrid}^j_{2.b}$.

More formally, the j^{th} challenge ciphertext is now $\mathsf{CT}^* = (\mathsf{CT}^*_0 = \mathsf{OneCT.Enc}(\mathsf{OneCT.SK}^*, m^j_b),\ \mathsf{CT}^*_1 = \mathsf{Sel.Enc}(\mathsf{Sel.MSK}, (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1))$. The rest of the hybrid is the same as the previous hybrid, $\mathsf{Hybrid}^j_{2.b}$.

Claim 6. *Assuming the selective security of* Sel*, for every PPT adversary $\mathcal{A}$, for $b \in \{0,1\}$, $j \in [p]$, we have* $|\mathsf{Adv}^{\mathcal{A}}_{2.b,j} - \mathsf{Adv}^{\mathcal{A}}_{3.b,j}| \leq \mathsf{negl}(\lambda)$.

Proof. Suppose the claim is not true for some PPT adversary $\mathcal{A}$, we construct a reduction that breaks the security of Sel. Our reduction will internally execute $\mathcal{A}$ by simulating the role of the challenger of the adaptive FE game.

For every $i \in [p]$, the reduction does the following. It first executes $\mathsf{OneCT.Setup}(1^\lambda)$ to obtain $\mathsf{OneCT.SK}^*_i$. It then samples K^*_i from $\mathcal{K}$. It also executes $\mathsf{Sym.Setup}(1^\lambda)$ to obtain $\mathsf{Sym.K}^*_i$. If $i \neq j$, the reduction then sends the message pair $\big((\mathsf{OneCT.SK}^*_i,\ \mathsf{K}^*_i,\ 0^\lambda, 0)),\ (\mathsf{OneCT.SK}^*_i, \mathsf{K}^*_i, 0^\lambda, 0)\big)$ to the challenger of the selective game and if $i = j$, the reduction instead sends the message pair $\big((\mathsf{OneCT.SK}^*_j, \mathsf{K}^*_j, 0^\lambda, 0),\ (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*_j, 1)\big)$. For the i^{th} message query, the challenger responds back with the challenge ciphertext $\mathsf{CT}^*_{1,i}$.

The reduction is now ready to interact with the adversary $\mathcal{A}$. If $\mathcal{A}$ makes a functional query f then the reduction constructs the circuit $G_{f,C_E,\tau}$ as in $\mathsf{Hybrid}^j_{2.b}$. It then queries the challenger of the selective game with the function G and in return it gets the key $\mathsf{Sel}.sk_G$. The reduction then sets $\mathsf{Ad}.sk_f$ to be $\mathsf{Sel}.sk_G$ which it then sends back to $\mathcal{A}$. The message queries made by $\mathcal{A}$ are handled as follows. When $\mathcal{A}$ submits the i^{th} message pair (m^i_0, m^i_1), the reduction executes $\mathsf{OneCT.Enc}(\mathsf{OneCT.SK}^*_i, m^i_0)$ to obtain $\mathsf{CT}^*_{0,i}$. It then sends the ciphertext $\mathsf{CT}^* = (\mathsf{CT}^*_{0,i}, \mathsf{CT}^*_{1,i})$ to the adversary. The output of the reduction is the output of $\mathcal{A}$.

We claim that the reduction is a legal adversary in the selective security game of Sel. To argue this, note that we only need to consider the j^{th} message query since the left and the right messages in all other message queries are the same. For the j^{th} message query $(M_0 = (\mathsf{OneCT.SK}^*_j, \mathsf{K}^*_j, 0^\lambda, 0),\ M_1 = (0^\lambda, 0^\lambda, \mathsf{Sym.K}^*_j, 1))$ and every functional query of the form $G_{f,C_E,\tau}$ made by the reduction, we have that $G_{f,C_E,\tau}(M_0) = G_{f,C_E,\tau}(M_1)$: By definition, $G_{f,C_E,\tau}(M_0)$ is the functional key of f, with respect to key $\mathsf{OneCT.SK}^*_j$ and randomness $\mathsf{PRF}_{\mathsf{K}^*_j}(\tau)$. Further, $G_{f,C_E,\tau}(M_1)$ is the decryption of C_E which is nothing but the functional key of f, with respect to key $\mathsf{OneCT.SK}^*_j$ and randomness $\mathsf{PRF}_{\mathsf{K}^*_j}(\tau)$. This proves that the reduction is a legal adversary in the selective security game.

If the challenger of the selective game sends back an encryption of $(\mathsf{OneCT.SK}^*_j, \mathsf{K}^*_j, 0^\lambda, 0)$ then we are in $\mathsf{Hybrid}^j_{2.b}$ else if the challenger encrypts $(0^\lambda, 0^\lambda, \mathsf{Sym.K}^*_j, 1)$ then we are in $\mathsf{Hybrid}^j_{3.b}$. By our hypothesis, this means the reduction breaks the security of the selective game with non-negligible probability that contradicts the security of Sel. This completes the proof of the claim. □

$\underline{\mathsf{Hybrid}^j_{4.b} \text{ for } b \in \{0,1\}, j \in [p]}$: For every functional query f made by the adversary, the challenger generates C_E by executing $\mathsf{Sym.Enc}(\mathsf{Sym.K}^*, \mathsf{OneCT}.sk_f)$, with $\mathsf{OneCT}.sk_f$ being the output of $\mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; R)$, where R is picked at random. The rest of the hybrid is the same as the previous hybrid.

Claim 7. *Assuming the security of the pseudorandom function family $\mathcal{F}$, for every PPT adversary $\mathcal{A}$, for $b \in \{0,1\}$, $j \in [p]$, we have* $|\mathsf{Adv}^{\mathcal{A}}_{3.b,j} - \mathsf{Adv}^{\mathcal{A}}_{4.b,j}| \leq \mathsf{negl}(\lambda)$.

Proof. Suppose the claim is false for some PPT adversary $\mathcal{A}$, we construct a reduction that internally executes $\mathcal{A}$ and breaks the security of the pseudorandom function family $\mathcal{F}$. The reduction simulates the role of the challenger of the adaptive game when interacting with $\mathcal{A}$. The reduction answers the functional queries,

made by the adversary as follows; the message queries are answered as in $\mathsf{Hybrid}^j_{3.b}$ (or $\mathsf{Hybrid}^j_{4.b}$). For every functional query f made by the adversary, the reduction picks τ at random which is then forwarded to the challenger of the PRF security game. In response it receives R^*. The reduction then computes C_E to be $\mathsf{Sym.Enc}(\mathsf{Sym.K}^*, \mathsf{OneCT}.sk_f)$, where $\mathsf{OneCT}.sk_f = \mathsf{OneCT.KeyGen}(\mathsf{OneCT.SK}^*, f; R^*)$. The reduction then proceeds as in the previous hybrids to compute the functional key $\mathsf{Ad}.sk_f$ which it then sends to $\mathcal{A}$.

If the challenger of the PRF game sent $R^* = \mathsf{PRF}_{\mathsf{K}^*}(\tau)$ back to the reduction then we are in $\mathsf{Hybrid}^j_{3.b}$ else if R^* is generated at random, for every query τ, by the challenger then we are in $\mathsf{Hybrid}^j_{4.b}$. From our hypothesis this means that the probability that the reduction distinguishes the pseudorandom values from random values is non-negligible, contradicting the security of the pseudorandom function family $\mathcal{F}$. □

We now show that $\mathsf{Hybrid}^j_{4.0}$ is computationally indistinguishable from $\mathsf{Hybrid}^j_{4.1}$.

Claim 8. *Assuming the adaptive security of* OneCT*, for $j \in [p]$, for every PPT adversary $\mathcal{A}$ we have* $|\mathsf{Adv}^{\mathcal{A}}_{4.0,j} - \mathsf{Adv}^{\mathcal{A}}_{4.1,j}| \leq \mathsf{negl}(\lambda)$.

Proof. Suppose there exists a PPT adversary $\mathcal{A}$, such that the claim is false. We design a reduction that internally executes $\mathcal{A}$ to break the adaptive security of OneCT.

The reduction simulates the role of the challenger of the adaptive private-key FE game. It answers both the functional as well as message queries made by the adversary as follows. If $\mathcal{A}$ makes a functional query f then it forwards it to the challenger of the adaptively-secure single-ciphertext FE scheme. In return it receives $\mathsf{OneCT}.sk_f$. It then encrypts it using the symmetric encryption scheme, where the symmetric key is picked by the reduction itself, and denote the resulting ciphertext to be C_E. The reduction then constructs the circuit $G_{f,C_E,\tau}$ as in the previous hybrids. Finally, the reduction computes the selective private-key functional key of $G_{f,C_E,\tau}$, where the reduction itself picks the master secret key of selective private-key FE scheme. The resulting functional key is then sent to $\mathcal{A}$. The message queries are handled as follows. Suppose the adversary $\mathcal{A}$ makes the i^{th} message-pair query (m^i_0, m^i_1). If $i \neq j$, then the reduction answers the query himself. That is, $\mathcal{B}$ samples the single-ciphertext FE master key $\mathsf{OneCT.SK}_i$ and PRF key K_i by himself. It then computes a single-ciphertext FE encryption of m_{b_i} using $\mathsf{OneCT.SK}_i$ and denote the result by CT^i_0: we define $b_i = 1$ if $i < j$ and $b_i = 0$ if $i > j$. Further, it computes a (selective) private-key FE encryption of $(\mathsf{OneCT.SK}_i, \mathsf{K}_i, 0^\lambda, 0)$, which is represented by CT^i_1. The challenger sends the ciphertext $\mathsf{CT}_i = (\mathsf{CT}^i_0, \mathsf{CT}^i_1)$ to $\mathcal{A}$. When $i = j$, the reduction forwards the message pair (m^j_0, m^j_1) to the challenger of the adaptive game. In response it receives CT^*_0. The reduction then generates CT^*_1 on its own where CT^*_1 is the selective FE encryption of $(0^\lambda, 0^\lambda, \mathsf{Sym.K}^*, 1)$. The reduction then sends $\mathsf{CT}^* = (\mathsf{CT}^*_0, \mathsf{CT}^*_1)$ to $\mathcal{A}$. The output of the reduction is the output of $\mathcal{A}$.

We note that the reduction is a legal adversary in the adaptive game of OneCT, i.e., for the message query (m^j_0, m^j_1), functional query f, we have that $f(m^j_0) = f(m^j_1)$: this follows from the fact that (i) the functional queries (resp., challenge message query) made by the adversary (of Ad) is the same as the functional queries (resp., challenge message query) made by the reduction, and (ii) the adversary (of Ad) is a legal adversary. This proves that the reduction is a legal adversary in the adaptive game.

If the challenger sends an encryption of m_0 then we are in $\mathsf{Hybrid}^j_{4.0}$ and if the challenger sends an encryption of m_1 then we are in $\mathsf{Hybrid}^j_{4.1}$. From our hypothesis, this means that the reduction breaks the security of OneCT. This proves the claim. □

$\underline{\mathsf{Hybrid}_5}$: This corresponds to the real experiment when the challenger uses the encryption oracle, parameterized by bit 1, to generate the challenge ciphertexts. That is, for all message queries of the form (m_0, m_1), the challenger sends an encryption of m_1 to the adversary. The output of this hybrid is the same as the output of the adversary.

We note that this hybrid is identical to the hybrid $\mathsf{Hybrid}^p_{1.1}$.

The above claims imply that Hybrid_0 is computationally indistinguishable from Hybrid_5 which proves the adaptive security of Ad. We thus have the following theorem.

Theorem 3. *Assuming the existence of a sufficiently-expressive selectively-secure private-key functional encryption scheme, there exists an adaptively-secure private-key functional encryption scheme.*

4 From FE for Shallow Circuits to FE for All Circuits

In this section we show that a functional encryption scheme that only supports key generation for functions with shallow circuits can be transformed into one that supports functions with arbitrarily deep circuits. In particular, the shallow class can be any class in which weak pseudorandom functions can be computed and has some composition properties.[4] For the sake of concreteness, we will consider the class NC^1, which can compute weak pseudorandom functions under standard cryptographic assumptions such as DDH or LWE. (A lower complexity class such as TC^0 is also sufficient under standard assumptions.)

While we present a direct reduction below, we notice that this property can be derived from the transformation in Section 3, by recalling some properties of the [GVW12] single-key functional encryption scheme. One can verify that the setup algorithm of the [GVW12] scheme can be implemented in NC^1 (under the assumption that it can evaluate weak pseudorandom functions), regardless of the depth of the function being implemented. This property carries through even after applying the [BS14] function privacy transformation. Lastly, to implement the Trojan method we need a symmetric encryption scheme with decryption in NC^1, which again translates to the evaluation of a weak pseudorandom function.

However, as we mentioned, there is a more direct approach of using the Trojan method to achieve our goal by using randomized encodings (we note that the [GVW12] FE scheme also uses randomized encodings, so this building block is inherent in both methods). A sketch of the approach was provided in the introduction (Section 1.2) and we now give a formal description of the construction.

Randomized encodings. This notion was introduced by Ishai and Kushilevitz [IK00]: A randomized encoding scheme for a function class $\mathcal{F}$ consists of two PPT algorithms $(\mathsf{RE.Encode}, \mathsf{RE.Decode})$. The PPT algorithm $\mathsf{RE.Encode}$ takes as input $(1^\lambda, F, x, r)$, where λ is the security parameter, $F : \{0,1\}^\lambda \to \{0,1\}$ is a function in $\mathcal{F}$, instance $x \in \{0,1\}^\lambda$ and randomness r. The output is denoted by $\hat{F}(x; r)$. The PPT algorithm $\mathsf{RE.Decode}$ takes as input $(\hat{F}(x; r))$ and outputs $y = F(x)$.

The security property states that there exists a PPT algorithm Sim that takes as input $(1^\lambda, F(x))$ and outputs $\mathsf{SimOut}_{F(x)}$ such that any PPT adversary cannot distinguish the distribution $\{\hat{F}(x; r)\}$ from the distribution $\{\mathsf{SimOut}_{F(x)}\}$.

The following corollary is derived from applying Yao's garbled circuit technique using a weak PRF based encryption algorithm.

Corollary 3. *If there exists a family of secure weak pseudorandom functions which can be evaluated in* NC^1*, then there exists a randomized encoding scheme* $(\mathsf{RE.Encode}, \mathsf{RE.Decode})$ *for the class of polynomial size circuits, such that* $\mathsf{RE.Encode}$ *is computable in* NC^1*.*

Our transformation. Let $\mathcal{NCFE}$ = ($\mathsf{NCFE.Setup}$, $\mathsf{NCFE.KeyGen}$, $\mathsf{NCFE.Enc}$, $\mathsf{NCFE.Dec}$) be a (private-key) functional encryption scheme for the class NC^1. We assume w.l.o.g that $\mathcal{NCFE}$ supports functions with multi-bit output (otherwise it is always possible to produce a secret key for each bit separately). We also use a pseudorandom function family denoted by $\mathcal{F} = \{\mathsf{PRF}_\mathsf{K}(\cdot)\}$ and a symmetric encryption scheme $(\mathsf{Sym.Setup}, \mathsf{Sym.Enc}, \mathsf{Sym.Dec})$. We construct a (private-key) functional encryption scheme $\mathcal{PFE}$ = ($\mathsf{PFE.Setup}$, $\mathsf{PFE.KeyGen}$, $\mathsf{PFE.Enc}$, $\mathsf{PFE.Dec}$) as follows.

- $\mathsf{PFE.Setup}(1^\lambda)$: Run $MSK \leftarrow \mathsf{NCFE.Setup}(1^\lambda)$ and output MSK as the master secret key.
- $\mathsf{PFE.KeyGen}(MSK, F)$: Given the master secret key MSK and a circuit F, do the following:
  - Pick a uniformly random string $C_E \leftarrow \{0,1\}^{\ell_1(\lambda)}$ and a uniformly random tag $\tau \leftarrow \{0,1\}^\lambda$.
  - Define the circuit

$$G_{F,C_E,\tau}(x, K_P, K_E, \beta) = \begin{cases} \widehat{F}(x; \mathsf{PRF}_{K_P}(\tau)) = \mathsf{RE.Encode}(F, x; \mathsf{PRF}_{K_P}(\tau)) & \text{if } \beta = 0 \\ \mathsf{Sym.Dec}_{K_E}(C_E) & \text{if } \beta = 1 \end{cases}$$

[4] Similarly to the class WEAK defined in [App13].

where $C_E \in \{0,1\}^{\ell_1(\lambda)}$ and $\tau \in \{0,1\}^{\ell_2(\lambda)}$ are as above. Furthermore, $K_P, K_E \in \{0,1\}^\lambda$, and $\beta \in \{0,1\}$.

- Run $SK_G \leftarrow \mathsf{NCFE.KeyGen}(MSK, G_{F,C_E,\tau})$ and outputs (SK_G, F, C_E, τ).

- $\mathsf{PFE.Enc}(MSK, x)$ gets as input the master secret key MSK and an input x and does the following.
  - Choose a uniformly random string $K_P \leftarrow \{0,1\}^\lambda$.
  - Compute and output $C \leftarrow \mathsf{NCFE.Enc}(MSK; (x, K_P, 0^\lambda, 0))$.

- $\mathsf{PFE.Dec}(SK_F, C)$, given the secret key $SK_F = (SK_G, F, C_E, \tau)$ for a function F and a ciphertext C, does the following. Run $\widehat{F}(x) \leftarrow \mathsf{NCFE.Dec}(SK_G, (F, C_E, \tau), C)$. and output $\mathsf{RE.Decode}(\widehat{F}(x))$.

Theorem 4. *Let* PRF *be a weak pseudorandom function family which can be evaluated in* NC^1*, let* SYM *be a symmetric encryption scheme whose decryption circuit is in* NC^1*, and let* (RE.Encode, RE.Decode) *be a randomized encoding scheme with encoding in* NC^1*.*

Then, if $\mathcal{NCFE}$ *is a selectively secure symmetric key functional encryption scheme for* NC^1*, then the scheme* $\mathcal{PFE}$ *is a selectively secure symmetric key functional encryption for* P*.*

Proof Sketch. The proof proceeds by a sequence of hybrids. For simplicity, we consider the case when the adversary submits a single message pair (m_0, m_1) and the argument can be generalized to the case of multiple messages.

$\underline{\mathsf{Hybrid}_0}$: This corresponds to the real experiment where the challenger sends an encryption of m_0 to the adversary.

$\underline{\mathsf{Hybrid}_1}$: For every functional query F, the challenger replaces C_E with $\mathsf{Sym.Enc}(K_E, \widehat{F}(m_0; \mathsf{PRF}_{K_P}(t))$ in the functional key for F.

By a sequence of intermediate hybrids (as many as the number of functional queries), Hybrid_1 can be shown to be computationally indistinguishable from Hybrid_0 by invoking the semantic security of the symmetric encryption scheme.

$\underline{\mathsf{Hybrid}_2}$: The challenge ciphertext will consist of an encryption of the message $(m_b, 0, K_E, 1)$ instead of the message $(m_0, K_P, 0^\lambda, 0)$.

This hybrid is computationally indistinguishable from Hybrid_1 by the semantic security of the functional encryption scheme.

$\underline{\mathsf{Hybrid}_3}$: For every functional query F, the challenger replaces C_E in all the functional keys with $\mathsf{Sym.Enc}(K_E, \widehat{F}(m_0; r))$ where r is uniformly random in the functional key for F.

By a sequence of intermediate hybrids (as many as the number of functional queries), Hybrid_3 can be shown to be computationally indistinguishable from Hybrid_2 by invoking the security of PRF.

$\underline{\mathsf{Hybrid}_4}$: Finally, for every functional query F, the challenger replaces $\widehat{F}(m_0; r)$ in the ciphertext hardwired in the functional key for F by the simulated randomized encoding $\mathsf{Sim}(1^\lambda, F(m_0))$.

By a sequence of intermediate hybrids (as many as the number of functional queries), Hybrid_4 can be shown to be computationally indistinguishable from Hybrid_3 by invoking the security of randomized encodings.

Note that the final hybrid does not depend on whether m_0 or m_1 was encrypted since for all functions $F(m_0) = F(m_1)$, and this proves the security of $\mathcal{PFE}$.

References

[AAB+13] Shashank Agrawal, Shweta Agrawal, Saikrishna Badrinarayanan, Abishek Kumarasubramanian, Manoj Prabhakaran, and Amit Sahai. Function private functional encryption and property preserving encryption: New definitions and positive results. *IACR Cryptology ePrint Archive*, 2013:744, 2013.

[App13] Benny Applebaum. Bootstrapping obfuscators via fast pseudorandom functions. *IACR Cryptology ePrint Archive*, 2013:699, 2013. To appear in Asiacrypt 2014.

[BB04] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Cachin and Camenisch [CC04], pages 223–238.

[BCOP04] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Cachin and Camenisch [CC04], pages 506–522.

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In Yehuda Lindell, editor, *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 52–73. Springer, 2014.

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[BRF13] Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors. *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*. ACM, 2013.

[BRS13a] Dan Boneh, Ananth Raghunathan, and Gil Segev. Function-private identity-based encryption: Hiding the function in functional encryption. In *Advances in Cryptology–CRYPTO 2013*, pages 461–478. Springer, 2013.

[BRS13b] Dan Boneh, Ananth Raghunathan, and Gil Segev. Function-private subspace-membership encryption and its applications. In *Advances in Cryptology-ASIACRYPT 2013*, pages 255–275. Springer, 2013.

[BS14] Zvika Brakerski and Gil Segev. Function-private functional encryption in the private-key setting. Cryptology ePrint Archive, Report 2014/550, 2014.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *Theory of Cryptography*, pages 253–273. Springer, 2011.

[CC04] Christian Cachin and Jan Camenisch, editors. *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*. Springer, 2004.

[CIJ+13] Angelo De Caro, Vincenzo Iovino, Abhishek Jain, Adam O'Neill, Omer Paneth, and Giuseppe Persiano. On the achievability of simulation-based security for functional encryption. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 519–535. Springer, 2013.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 40–49. IEEE Computer Society, 2013.

[GGHZ14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. *IACR Cryptology ePrint Archive*, 2014:666, 2014.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

[GKP+12] Shafi Goldwasser, Yael Tauman Kalai, Raluca A Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Succinct functional encryption and applications: Reusable garbled circuits and beyond. *IACR Cryptology ePrint Archive*, 2012:733, 2012.

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In Boneh et al. [BRF13], pages 555–564.

[Gol09] Oded Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*, volume 2. Cambridge university press, 2009.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006*, pages 89–98, 2006.

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In *Advances in Cryptology–CRYPTO 2012*, pages 162–179. Springer, 2012.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Boneh et al. [BRF13], pages 545–554.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[IK00] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 294–304. IEEE, 2000.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, pages 146–162, 2008.

[LOS+10] 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 *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, pages 62–91, 2010.

[LW10] Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.

[LW12] Allison B. Lewko and Brent Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 180–198. Springer, 2012.

[O'N10] Adam O'Neill. Definitional issues in functional encryption. *IACR Cryptology ePrint Archive*, 2010:556, 2010.

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 463–472. ACM, 2010.

[SSW09] Emily Shen, Elaine Shi, and Brent Waters. Predicate privacy in encryption systems. In *Theory of Cryptography*, pages 457–473. Springer, 2009.

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, pages 457–473, 2005.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

[Wat14] Brent Waters. A punctured programming approach to adaptively secure functional encryption. Cryptology ePrint Archive, Report 2014/588, 2014.