Functional Encryption: Decentralized and Delegatable

Nishanth Chandran[*] Vipul Goyal[†] Aayush Jain[‡] Amit Sahai[§]

Abstract

Recent advances in encryption schemes have allowed us to go far beyond point to point encryption, the scenario typically envisioned in public key encryption. In particular, Functional Encryption (FE) allows an authority to provide users with keys corresponding to various functions, such that a user with a secret key corresponding to a function f, can compute $f(m)$ (and only that) from a cipher-text that encrypts m.

While FE is a very powerful primitive, a key downside is the requirement of a central point of trust. FE requires the assumption of a central trusted authority which performs the system setup as well as manages the credentials of every party in the system on an ongoing basis. This is in contrast to public key infrastructure which may have multiple certificate authorities and allows a party to have different (and varying) level of trust in them.

In this work, we address this issue of trust in two ways:

- First, we ask how realistic it is to have a central authority that manages all credentials and is trusted by everyone? For example, one may need to either obtain the permission of an income tax official or the permission of the police department and a court judge in order to be able to obtain specific financial information of a user from encrypted financial data. Towards that end, we introduce a new primitive that we call *Multi-Authority Functional Encryption* (MAFE) as a generalization of both Functional Encryption and Multi-Authority Attribute-Based Encryption (MABE). We show how to obtain MAFE for arbitrary polynomial-time computations based on subexponentially secure indistinguishability obfuscation and injective one-way functions.
- Second, we consider the notion of *delegatable* functional encryption where any user in the system may independently act as a key generation authority. In delegatable FE, any user may derive a decryption key for a policy which is "more restrictive" than its own. Thus, in delegatable functional encryption, keys can be generated in a hierarchical way, instead of directly by a central authority. In contrast to MAFE, however, in a delegatable FE scheme, the trust still "flows" outward from the central authority.

Finally, we remark that our techniques are of independent interest: we construct FE in arguably a more natural way where a decryption key for a function f is simply a signature on f. Such a direct approach allows us to obtain a construction with interesting properties enabling multiple authorities as well as delegation.

[*]Microsoft Research, India. Email: nichandr@microsoft.com.
[†]Microsoft Research, India. Email: vipul@microsoft.com.
[‡]UCLA, USA. Email: aayushjainiitd@gmail.com. Work done in part while at Microsoft Research, India.
[§]UCLA, USA. Email: sahai@cs.ucla.edu.

1 Introduction

Functional Encryption. Functional encryption (FE) [GGH$^+$13, GKP$^+$13, BSW11], is a powerful form of public-key encryption that allows users to learn a function of the underlying message from a ciphertext. In FE there is an authority which performs the setup and issues "function keys" to any decryptor. This decryptor can now learn those functions of the encrypted data for which he has been issued keys. This differs from the traditional view point of secrecy where an encryption scheme provides all or nothing confidentiality of data.

As an example, suppose Alice wants to allow users to encrypt and store data x, meant for her, on a cloud provider. Now, suppose she wants the cloud provider to learn the output of some function f on the data. FE provides a solution to this problem. Whenever Alice wants the cloud provider to learn $f(x)$, she "authorizes" the release of function f to the cloud provider, who can then "decrypt" the cipher-text to obtain only $f(x)$. The security of the FE scheme requires that the cloud provider learns only $f(x)$ and no other information.

Some variants of functional encryption include multi-input functional encryption [GGG$^+$14] and attribute-based encryption [SW05, GPSW06].

Comparison with Public-Key Encryption. Functional encryption (and its variants stated above) requires the assumption of a central trusted authority which manages all the credentials for every party. Furthermore, every party is required to place trust in this central authority. In contrast, public-key encryption allows for the simultaneous co-existence of multiple authorities (examples include DigiCert, Comodo, etc). Each party may get certificates from any subset of these authorities, and, each party could trust any subset of these authorities (placing eve varying levels of trust in different ones). One could argue that this has significantly helped in the adoption of public-key encryption.

1.1 Multi-Authority Functional Encryption

Inspired by the distributed trust offered by public-key encryption, we consider a similar model for functional encryption and introduce the notion of Multi-Authority Functional Encryption (MAFE). In MAFE, we have multiple authorities in the system. Any authority can "certify" a function (or credential) of its choice for a given user. For example, an authority could provide the key for a function which can only read and compute on data which is labelled with confidentiality level 3 or lower. A user may choose to get decryption keys (for potentially different functions) from any subset of the authorities of its choice. Finally, the data owner (or the encrypter) could also choose which of these authorities to trust, and, to "what extent". If (and only if) the trust placed by the data owner in the authorities from which the user got the keys is "sufficient", the user would be able to compute on the encrypted data and get the output determined by its credentials (or decryption keys).

Before we describe MAFE in more detail, let us consider another motivating example. A bank maintains encrypted financial records of its customers. Some parts of this data may have to be revealed to law enforcement officials in the event of a criminal or civil investigation. For example, the total value of cash deposits made by a customer during the last financial year might have to be made available to either, an income tax official who has been granted a warrant by the Income Tax Commission, or to a police officer with a subpoena signed by a Supreme Court Judge. The question that we ask here is: *Can the bank encrypt and store its data in such a way that only properly authorized individuals (who satisfy an access policy) learn the output of a specific function f on its data*?

One trivial way to solve this problem is to involve the bank in every step of the procedure — the income tax official can approach the bank with the right credentials to compute the function f on the bank's data, and then the bank can simply decrypt the stored data, compute f and return back the function output to the tax official. However, this procedure is cumbersome and has to be repeated every time an official wishes to learn some function of the data. Ideally the bank would like to allow the authorities involved (Income Tax Commission, Police officer, or Judge) to independently issue "secret function keys" for any decryptor. Anyone who satisfies the right policy, should, with the help of the stored encrypted data, be able to learn the output of the function f on the data. The policies could be arbitrary monotone functions (such as "Bank" OR "I.T. Commission" OR ("Police officer" AND "Supreme Court Judge")) and the functions f could be arbitrary polynomial-time computable functions. Naturally, security requires that if two decryptors do not have the required credentials to decrypt a function individually, then they should also not be able to compute the function value jointly.

Our notion of MAFE is natural. The multiple authorities can "independently" generate their private and public keys (note that there is no setup or common reference string required). An encryptor, should be able to encrypt a message m along with a policy F over the various authorities. Any authority i, should be able to generate a token for a user with identity UID and property U_i. A user with identity UID with tokens for U_i from authority $i \in [n]$, should be able to decrypt the cipher-text to recover $F(U_1, .., U_n, m)$. We require that colluding users, say UID_1 and UID_2, (possibly, in collusion with some corrupt authorities) should not jointly learn anything more from the ciphertext than what they are authorized to.[1]

1.2 Our Results

In this work, we introduce the notion of Multi-Authority Functional Encryption (MAFE). We show that MAFE for all polynomial-size circuits can be constructed assuming sub-exponentially secure indistinguishability obfuscation. More specifically, we use only the following primitives: (1) sub-exponentially secure indistinguishability obfuscation; and (2) sub-exponentially secure injective one way function [2]. Here "sub-exponential security" refers to the fact that the advantage of any (efficient) adversary should be sub-exponentially small. For primitive (2), this should furthermore hold against adversaries running in sub-exponential time.

Examining the question of trust in a different way, we also consider the notion of *delegatable* functional encryption where any user in the system may independently act as a key generation authority. In delegatable FE, any user may derive a decryption key for a policy which is "more

[1]Note that, while previous works on functional encryption considered security against malicious receivers, no security was guaranteed if the authority itself is malicious. The definition we propose allows an encryptor to protect his data with respect to a policy F. Even if the authority is malicious, a receiver will not be able to learn anything that is not allowed by the policy.

[2]We however do not require that such injective one way function is efficiently samplable. Its existence suffices for the construction.

restrictive" than its own. In other words, delegatable FE allows a user to compute a key for a function g given a key for a function f if and only if the output of g can be computed given the output of f. We note that delegatable FE is complementary to MAFE, in that while delegatable FE allows users to generate restricted versions of their own keys without needing to involve the central authority, for delegatable FE trust still "flows" down from the central authority. We show how to construct delegatable FE based on (sub-exponentially secure) indistinguishability obfuscation and hierarchical identity-based encryption.

We believe our techniques to be of independent interest: we construct FE in arguably a more natural way where a decryption key for a function f is simply a signature on f. Such a direct approach to constructing decryption keys allows us to obtain a construction with interesting properties enabling multiple authorities as well as delegation.

1.3 Challenges and Techniques

At first glance, the rich notions of functional encryption and multi-authority attribute-based encryption may seem to help us solve the problem of MAFE described above. In a Functional Encryption (FE) [GGH+13, GKP+13, BSW11] scheme, decryption keys are provided by the authority for specific functions. A user, who has obtained sk_f, can decrypt $\mathsf{Enc}_{pk}(m)$ and then learn $f(m)$. However, FE schemes require the existence of a single authority who issues the decryption keys and hence are not applicable in our scenario. In our example, we cannot have the bank give its secret keys to all the officials as a corrupted police officer (say) could, without the cooperation of the Judge, learn arbitrary information about the bank's financial data! In a Multi-Authority Attribute-based Encryption (MA-ABE) scheme [Cha07, CC09, LW11], multiple authorities issue decryption credentials to users and only someone who satisfies the right access policy (that could depend on attributes specified by different authorities) can learn the contents of the encrypted data. However, like attribute-based encryption, MA-ABE schemes allow the authorized user to learn the plain-text data in its entirety (and not some function of the data).

One proposal to combine the two of the above may be the following. Let there be one special authority that has a functional encryption key (PK, SK). All the authorities have their keys for the multi-authority ABE scheme. Now the encryptor just encrypts m with PK and then this ciphertext is encrypted again with respect to the policy $\mathbb{A}$ and keys of the MA-ABE scheme. In order to generate tokens for a function f, all authorities give out tokens for $ID = (UID, f)$ and the special authority also gives out functional encryption key for f. The decryptor then just uses the MABE keys to recover the functional encryption ciphertext which is then decrypted using the functional encryption token for f. Unfortunately, the above approach is flawed and is in fact insecure. Consider a user who has obtained valid decryption keys for f (that allows him to decrypt f) but has obtained an invalid set of tokens for g (but has received only the functional encryption key for g from the special authority). One can easily see that such a user can learn the outputs of both f and g on the encrypted data. This attack shows that rolling MA-ABE and FE into one requires significantly new ideas. Another approach that one might take is to consider the use of of Multi-Input Functional Encryption [GGG+14] (a broader generalization of FE where the function keys allow to compute on multiple cipher-texts). Again, the main problem that occurs in doing this, is that there should be a single authority that knows the secret keys corresponding to all inputs.

Similar to MIFE, we show that multi-authority functional encryption, even when the user receives a single token query per authority, implies indistinguishability obfuscation. Additionally, we also show that a simulation secure variant implies black box obfuscation, and is hence, impossible for arbitrary functions [BGI+01].

Our Techniques. Let us first analyze the problem in a single authority scenario. The starting point of our construction is a construction of a functional encryption scheme due to [BCP14] based on *differing-inputs obfuscation*, which we now believe is implausible [GGHW14]. In their construction, a ciphertext was an obfuscated circuit that checks the signature for a function f and computes f if the signature is valid. The security proof relied on the fact that if the adversary distinguishes the cipher-text (or rather an obfuscated circuit) encrypting m_0 from that of m_1, then one must be able to extract out a signature on a function g such that $g(m_0) \neq g(m_1)$ thereby producing a forgery. While techniques based on this idea would work for us if we were willing to work with differing-inputs obfuscation, obtaining MAFE under indistinguishability obfuscation [GGH+13] is a much harder problem. One of the crucial ideas in the work of [BCP14], is that indistinguishability obfuscation behaves like a differing-inputs obfuscator for the setting where the circuits in question differ on bounded number of points. Unfortunately, even with subexponentially secure iO, there are simply too many points on which these functions would differ for this strategy to work.

Our first idea is that we make the "signatures" unique, in the sense, that for every message there exists only one signature that verifies. In order to achieve this, we initialize it using puncturable PRF's, indistinguishability obfuscation and an injective one way function as done in [SW14]. The main idea of the security proof is that we can build exponentially many hybrids corresponding to the function space. We index each hybrid with a function x. In hybrid x, the cipher-text takes as input function and a signature (f, σ) and checks if $f < x$. If that is the case, it outputs $f(m_0)$ otherwise $f(m_1)$. We argue indistinguishability between hybrids x and $x + 1$ by intermediate hybrids where we puncture the PRF at x and plant the PRF evaluations with a random sample. Then we further note that if the adversary now distinguishes these hybrids then it can be used to invert an injective one way function in sub-exponential time. This idea now can be scaled in multi-authority scenario. For technical purposes, that is, in order to handle key corruptions the signing key is an obfuscation of a circuit that evaluates a PRF on the input. While these are the core ideas in our construction, proving the construction secure is tricky.

We also show that MAFE immediately implies multi-authority attribute based encryption (without setup or random oracle; to the best of our knowledge all known constructions [LW11, CC09, Cha07] either rely on a trusted setup or on random oracles or both). As an interesting observation, we also show that MAFE (in fact, even MAFE secure against single key query and specific policy functions) implies (computational) broadcast non interactive functional secret sharing [BBDK00] for monotone P and polynomial sized circuits in a black box manner, which was previously not known.

Achieving Delegation. An application of the idea described above is that it can be extended to allow for delegation of function keys. In what we described above, if the signature scheme is delegatable (i.e. it allows us to compute a signature on $f||x$ for any x given a signature on f) then it allows us to compute keys for $x \cdot f$ given a key for f. For the purpose of this paper by $\cdot$ we mean mathematical composition. Precisely, $f_1 \cdot f_2$ on input x computes $f_1(f_2(x))$.

We call such an encryption scheme as Delegatable Functional Encryption (DFE). In order to construct this primitive, we make use of the exponential hybrid approach and make interesting use of hierarchical identity based encryption (HIBE). The main idea behind this construction is that the cipher-text encrypting m is an obfuscated program which on input f computes an HIBE encryption of $f(m)$ on identity f using randomness generated from a PRF applied on f. Any decryptor can than decrypt this cipher-text using a HIBE key for identity f or its prefix. While for proving security any sub-exponential IBE suffices, HIBE however gives delegation for free. So far, in obfuscation based FE schemes, delegation could have been done by re-obfuscating the key with a composition function wrapper. Due to efficiency issues this could be done at most $log(k)$ number of times; however our construction supports delegation up to any (apriori bounded) polynomial number of times.

1.4 Organization.

The rest of this paper is organized as follows: In Section 2, we recall some definitions and primitives used in the rest of the paper. In Section 3 we formally define MAFE and DFE. In Section 4, we provide a construction, proof and applications for MAFE. Section 5 gives a construction for DFE.

2 Preliminaries

In this section we recall various concepts on which the paper is built upon. For the rest of the paper, we denote by $\mathbb{N}$ the set of natural numbers $\{1, 2, 3, ..\}$. For the purpose of this paper by $\cdot$ operator on two functions we mean a mathematical composition. Precisely, $f_1 \cdot f_2$ on input x computes $f_1(f_2(x))$. Now we describe some of the primitives used in the paper.

2.1 Injective One Way Function

A one way function with security (s, ϵ) is an efficiently evaluable function $P : \{0,1\}^* \to \{0,1\}^*$ and $Pr_{x \xleftarrow{\$} \{0,1\}^n}[P(A(P(x))) = P(x)] < \epsilon(n)$ for all circuits A of size bounded by $s(n)$. It is called an injective one way function if it is injective in the domain $\{0,1\}^n$ for all sufficiently large n.
In this work we require that there *exists*[3] (s, ϵ) injective one way function with $s(n) = 2^{n^{c_{owp1}}}$ and $\epsilon = 2^{-n^{c_{owp2}}}$ for some constants $0 < c_{owp1}, c_{owp2} < 1$. [Hol06, Wee07] have used $(2^{cn}, 1/2^{cn})$ secure one way functions and permutations for some constant c, while here we require our injective one way function to be $(s(n) = 2^{n^{c_{owp1}}}, \epsilon = 2^{-n^{c_{owp2}}})$ secure for some constants $0 < c_{owp1}, c_{owp2} < 1$. This is a reasonable assumption due to following result from [GGKT05]

Lemma 1. *Fix $s(n) = 2^{n/5}$. For all sufficiently large n, a random permutation π is $(s(n), 1/2^{n/5})$ secure with probability at least $1 - 2^{-2^{n/2}}$.*

Such assumptions have been made and discussed in works of [Hol06, Wee05, Wee07]. In particular, we require the following assumption:

[3]We however do not require that the injective one way function can be sampled efficiently

Assumption 1: For any adversary A with running time bounded by $s(n) = O(2^{n^{c_{owp1}}})$, for any apriori bounded polynomial $p(n)$ there exists an injective one way function P such that,

$$Pr[r_i \xleftarrow{\$} \{0,1\}^n \forall i \in [p], \mathcal{A}^{\mathcal{O}}(P(r_1),..,P(r_p)) = (r_1,..,r_p)] < O(2^{-n^{c_{owp2}}})$$

for some constant $0 < c_{owp_1}, c_{owp_2} < 1$. Here, oracle $\mathcal{O}$ can reveal at most $p-1$ values out of $r_1,..,r_p$. Note that this assumption follows from the assumption described above with a loss p in the security gap.

2.2 Indistinguisability Obfuscation

Below we present the formal definition following the syntax of [GGH+13]:

Definition 1. *A uniform PPT machine $i\mathcal{O}$ is an indistinguishability obfuscator for a class of circuits $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ if the following properties are satisfied.*
Correctness: *For every $k \in \mathbb{N}$, for all $\{\mathcal{C}_k\}_{k\in\mathbb{N}}$, we have*

$$Pr[C' \leftarrow i\mathcal{O}(1^k, C) : \forall x, C'(x) = C(x)] = 1$$

Security: *For any pair of functionally equivalent equi-sized circuits $C_0, C_1 \in \mathcal{C}_k$ we have that: For every non uniform PPT adversary $\mathcal{A}$ there exists a negligible function ϵ such that for all $k \in \mathbb{N}$,*

$$\mid Pr[\mathcal{A}(1^n, i\mathcal{O}(1^k, C_0), C_0, C_1, z) = 1] - Pr[\mathcal{A}(1^k, i\mathcal{O}(1^k, C_1), C_0, C_1, z) = 1] \mid \leq \delta(k)$$

We additionally say that $i\mathcal{O}$ is sub-exponentially secure if there exists some constant $\alpha > 0$ such that for every non uniform PPT $\mathcal{A}$ the above indistinguishability gap is bounded by $\delta(k) = O(2^{-k^{\alpha}})$

Definition 2 (Indistinguishability obfuscation for P/poly)**.** *$i\mathcal{O}$ is a secure indistinguishability obfuscator for P/Poly, if it is an indistinguishability obfuscator for the family of circuits $\{\mathcal{C}_k\}_{k\in\mathbb{N}}$ where $\mathcal{C}_k$ is the set of all circuits of size k.*

2.3 Puncturable Psuedorandom Functions

A PRF $F : \mathcal{K}_{k\in\mathbb{N}} \times \mathcal{X} \rightarrow \mathcal{Y}_{k\in\mathbb{N}}$ is a puncturable pseudorandom function if there is an additional key space $\mathcal{K}_p$ and three polynomial time algorithms ($F.\mathsf{setup}$, $F.\mathsf{eval}$,
$F.\mathsf{puncture}$) as follows:

- $F.\mathsf{setup}(1^k)$ a randomized algorithm that takes the security parameter k as input and outputs a description of the key space $\mathcal{K}$, the punctured key space $\mathcal{K}_p$ and the PRF F.
- $F.\mathsf{puncture}(K, x)$ is a randomized algorithm that takes as input a PRF key $K \in \mathcal{K}$ and $x \in \mathcal{X}$, and outputs a key $K\{x\} \in \mathcal{K}_p$.
- $F.\mathsf{Eval}(K, x')$ is a deterministic algorithm that takes as input a punctured key $K\{x\} \in \mathcal{K}_p$ and $x' \in \mathcal{X}$. Let $K \in \mathcal{K}$, $x \in \mathcal{X}$ and $K\{x\} \leftarrow F.\mathsf{puncture}(K, x)$.

The primitive satisfies the following properties:

1. **Functionality is preserved under puncturing:** For every $x^* \in \mathcal{X}$,

$$Pr[F.\mathsf{eval}(K\{x^*\}, x) = F(K, x)] = 1$$

here probability is taken over randomness in sampling K and puncturing it.

2. **Psuedo-randomness at punctured point:** For any poly size distinguisher D, there exists a negligible function $\mu(\cdot)$, such that for all $k \in \mathbb{N}$ and $x^* \in \mathcal{X}$,

$$\mid Pr[D(x^*, K\{x^*\}, F(K, x^*)) = 1] - Pr[D(x^*, K\{x^*\}, u) = 1] \mid \leq \mu(k)$$

where $K \leftarrow F.\mathsf{Setup}(1^{\mathsf{k}})$, $K\{x^*\} \leftarrow F.\mathsf{puncture}(K, x^*)$ and $u \xleftarrow{\$} \mathcal{Y}_k$

We say that the primitive is sub-exponentially secure if μ is bounded by $O(2^{-k^{c_{PRF}}})$, for some constant $0 < c_{PRF} < 1$. We also abuse the notation slightly and use $F(K, \cdot)$ and $F.\mathsf{Eval}(K, \cdot)$ to mean one and same thing irrespective of whether key is punctured or not. Puncturable PRFs have been extensively used, such as in the work of [SW14] and were constructed in [BW13]. They can be based on the existence of a one way function.

2.4 Hierarchical Identity Based Encryption

Hierarchical Identity Based Encryption (HIBE) [BBG05, GH08, GS02] consists of following algorithms: $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$. Setup takes in the security parameter and produces a master secret key MSK and a master public key MPK. Identities are vectors. A k dimensional identity vector is treated as identity at depth k. We refer to the master secret key as the private key at depth 0 and note that an IBE system is a HIBE where all identities are at depth 1. Algorithm KeyGen takes as input an identity $ID = (I_1, ..., I_k)$ at depth k and the private key $d_{ID|k-1}$ of the parent identity $ID_{k-1} = (I_1, .., I_{k-1})$ at depth $k - 1$, and then outputs the private key d_{ID} for identity ID. The encryption algorithm encrypts messages for an identity using MPK and the decryption algorithm decrypts cipher-texts using the private key.

We now describe the game defining security for any HIBE scheme.
Setup: The challenger runs the Setup algorithm and gives adversary $\mathcal{A}$ the resulting system, keeping the master-key MSK to itself.

Phase 1: $\mathcal{A}$ adaptively issues queries $q_1, .., q_m$ where query q_i is a private key query for $< ID_i >$. Challenger responds by running algorithm KeyGen to generate the private key d_i corresponding to the identity ID_i and sends d_i to $\mathcal{A}$.

Challenge Phase Once $\mathcal{A}$ decides that Phase 1 is over, it outputs an identity ID^* and two equal length plain-texts $M_0, M_1 \in \mathcal{M}$ on which it wishes to be challenged. The only restriction is that $\mathcal{A}$ did not previously issue a private key query for ID^* or a prefix of ID^*. Challenger picks a random bit $b \in \{0, 1\}$ and sets the challenge ciphertext to $\mathrm{CT} = \mathsf{Encrypt}(MPK, ID^*, M_b)$, which is sent to $\mathcal{A}$.

Phase 2 $\mathcal{A}$ adaptively issues queries $q_{m+1}, .., q_n$ where query q_i is a private key query for $< ID_i >$ where $ID_i \neq ID^*$ or a prefix of ID^*. Challenger responds by running algorithm KeyGen to generate the private key d_i corresponding to the identity ID_i and sends d_i to $\mathcal{A}$

Guess The attacker submits a guess $b^{'}$ for b. The attacker wins if $b^{'} = b$. The attacker's advantage in this game is defined to be $\mu =\mid Pr[b = b^{'}] - 1/2 \mid$

We say that $HIBE$ is secure if μ is a negligible function of the security parameter. Further it is sub-exponentially secure if there exists a constant c_{HIBE} such that $\mu < O(2^{-k^{c_{HIBE}}})$.

2.5 (d, δ) Weak Extractability Obfuscator

The concept of weak extractability obfuscator was first introduced in [BCP14] where they claimed that if there is an adversary that can distinguish between indistinguishability obfuscations of two circuits that differ on polynomial number of inputs with noticable probability, then there is a PPT extractor that extracts a differing input with overwhelming probability. We generalise the notion to what we call (d, δ) weak extractability obfuscator, where we require that if there is any PPT adversary that can distinguish between obfuscations of two circuits (that differ on at most d inputs) with atleast $\epsilon > \delta$ probability, then there is a extractor that extracts a differing input with overwhelming probability and runs in time $poly(1/\epsilon, d, k)$ time. Such a primitive can be constructed from a sub-exponentially secure indistinguishability obfuscation. $(1, 2^{-k})$ weak extractability obfuscation will be crucially used in our construction for our MAFE scheme. We believe that in various applications of differing inputs obfuscation, it may suffice to use this primitive along with other sub-exponentially secure primitives.

Definition 3. *A uniform transformation $we\mathcal{O}$ is a (d, δ) weak extractability obfuscator for a class of circuits $\mathcal{C} = \{\mathcal{C}_k\}$ if the following holds. For every PPT adversary $\mathcal{A}$ running in time $t_{\mathcal{A}}$ and $1 \geq \epsilon(k) > \delta$, there exists a algorithm E for which the following holds. For all sufficiently large k, and every pair of circuits on n bit inputs, $C_0, C_1 \in \mathcal{C}_k$ differing on at most $d(k)$ inputs, and every auxiliary input z,*

$$\mid Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_0), C_0, C_1, z) = 1] - Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_1), C_0, C_1, z) = 1] \mid \geq \epsilon$$

$$\Rightarrow Pr[x \leftarrow E(1^k, C_0, C_1, z) : C_0(x) \neq C_1(x) \geq 1 - negl(k)$$

and the expected runtime of E is $O(p_E(1/\epsilon, d, t_{\mathcal{A}}, n, k))$ for some fixed polynomial p_E. In addition, we also require the obfuscator to satisfy correctness.
Correctness: *For every $n \in \mathbb{N}$, for all $\{\mathcal{C}_n\}_{n \in \mathbb{N}}$, we have*

$$Pr[C' \leftarrow we\mathcal{O}(1^n, C) : \forall x, C'(x) = C(x)] = 1$$

We now construct a $(1, 2^{-k})$ input weak extractability obfuscator from sub-exponentially secure indistinguishability obfuscation. Following algorithm describes the obfuscation procedure.

$we\mathcal{O}(1^k, C)$: The procedure outputs $C' \leftarrow i\mathcal{O}(1^{k^{1/\alpha}}, C)$. Here, $\alpha > 0$ is a constant chosen such that any polynomial time adversary against indistinguishability obfuscation has security gap upper bounded by $2^{-k}/4$.

The proof of the following theorem is given in the appendix A.

Theorem 1. *Assuming sub-exponentially secure indistinguishability obfuscation, there exists* $(1, \delta)$ *weak obfuscator for* $P/poly$ *for any* $\delta > 2^{-k}$*, where* k *is the size of the circuit.*

In general, assuming sub-exponential security one can construct (d, δ) extractability obfuscator for any $\delta > 2^{-k}$. Our construction is as follows:
$we\mathcal{O}(C)$: Let α be the security constant such that $i\mathcal{O}$ with parameter $1^{k^{1/\alpha}}$ has security gap upper bounded by $O(2^{-3k})$. This can be found due to sub exponential security of indistinguishability obfuscation. The procedure outputs $C' \leftarrow i\mathcal{O}(1^{k^{1/\alpha}}, C)$.
We cite [BCP14] for the proof of the following theorem.

Theorem 2 ([BCP14])**.** *Assuming sub-exponentially secure indistinguishability obfuscation, there exists* (d, δ) *weak extractability obfuscator for* $P/poly$ *for any* $\delta > 2^{-k}$*.*

3 Definitions

In this section, we define syntax and security notions of Multi-Authority Functional Encryption scheme and a Delgeatable Functional Encryption scheme.

3.1 Multi-Authority Functional Encryption

In this section, we describe syntax and security notions of a Multi-Authority Functional Encryption (MAFE) scheme. Each authority has its own identity $id \in \mathcal{I}$. Our scheme allows addition of a new authority at any given point of time and each authority can be set up independently and without any interaction with any other authority. Multi-authority Functional Encryption system is comprised of following algorithms:

- Authority Setup$(id, 1^k) \rightarrow (MPK_{id}, MSK_{id})$. Each authority id runs the authority setup algorithm to produce its own public key and secret key pair, (MSK_{id}, MPK_{id}).
- KeyGen$(MSK_{id}, UID, U) \rightarrow K_{UID,id,U}$. The key generation takes as input an identity UID, a property U and secret key MSK_{id} for any authority with identity id and outputs a token $K_{UID,id,U}$.
- Encrypt$(\{MPK_{id}\}_{i\in\mathcal{S}}, F, \rho, m) \rightarrow CT$. The encryption algorithm takes in a message M, a $\mid \mathcal{S} \mid +1 < O(poly(k))$ variate policy function F (each component is in $\{0, 1\}^k$) , a permutation ρ that maps identities in $\mathcal{S}$ to index of the input to F, the set of public keys for authorities in some set $\mathcal{S} \subset \mathcal{I}$. It outputs a cipher-text (F, CT). F takes as input upto $\mid \mathcal{S} \mid$ properties and a message and outputs a value.
- Decrypt$(\{K_{UID,id,U_id}\}_{id\in\mathcal{S}}, CT) \rightarrow F(< \{U_{id}\}_{id\in\mathcal{S}} >, M)$ or $\perp$. The decryption algorithm takes as input a ciphertext CT encrypting a message M and a collection of tokens from authorities $id \in \mathcal{S}$ for a identity UID and properties U_{id}. The algorithms outputs $F(< \{U_{id}\}_{id\in\mathcal{S}} >, M)$ or $\perp$. $<>$ denoted the permutation that uses ρ and maps U_{id} to the index $\rho(id)$ for input to F.

Definition 4. *A Multi-Authority Functional Encryption* (MAFE) *system is said to be correct if whenever CT is obtained from the encryption algorithm on the message M and policy F, and $\{K_{UID,id,U_{id}}\}_{i\in\mathcal{S}}$ is a set of tokens obtained from the key generation algorithm for the same identity UID and properties U_{id} for a set of token issuing authorities $id \in \mathcal{S} \subset \mathcal{I}$,*

$$Decrypt(CT, \{K_{UID,id,U_{id}}\}_{id\in\mathcal{S}}) = F(<\{U_{id}\}_{id\in\mathcal{S}}>, M)$$

Remark: Note, that we require that there is no trusted or global setup algorithm in the syntax.

3.1.1 Security Definition

We now present the security definition for MAFE scheme. Note that, previous works on functional encryption considered security against malicious receivers. No security was guaranteed if the authority itself is malicious and let's say completely leaks out the secret key. The definition we propose allows an encryptor to protect his data with respect to a policy F. Even if the authority is malicious a receiver will not be able to learn anything that is not allowed by the policy. We define security notion of the scheme by the following game between a challenger and an attacker. For ease of notation let us assume the chosen set of authorities by the adversary is $[N]$ for a polynomial N.

Setup: The adversary outputs $[N]$. The challenger does setup for authorities labelled in $[N]$ and hands over the public keys of all the authorities to the adversary.

Key Query Phase 1 The attacker makes token queries by submitting tuples $(UID, i, U_{i,j})$ to the challenger where i is a non corrupt authority and UID is an identity and j is the query number. The challenger responds by giving the attacker the corresponding key $K_{UID,i,U_{i,j}}$. Adversary can also corrupt an authority by asking for MSK_i for any $i \in [N]$. These queries can be made adaptively and in any fashion. Let S denote the set of all authorities i for which MSK_i has been released.

Challenge Phase The attacker selects two messages M_0, M_1 and a policy (F, ρ). The policy and the message queries satisfy the following constraint. Let $K_{UID,i,U_{i,j_i}}$ denote a token queried by attacker for $i \in [N] \setminus S$ for identity UID. Then it must hold that, $F(<\{U_{i,j_i}\}_{i\in[N]\setminus S}, \{x_i\}_{i\in S}>, M_0) = F(<\{U_{i,j_i}\}_{i\in[N]\setminus S},$
$\{x_i\}_{i\in S}>, M_1) \forall x_i \forall i \in S$. Here $<>$ denotes the permutation that maps U_i or x_i using ρ to the index $\rho(i)$ for input to F. The challenger flips a random coin $b \in \{0,1\}$ and sends an encryption of M_b under the policy F.

Key Query Phase 2 The attacker may submit additional token queries $(UID, i,$
$U_{i,j})$ and queries for master secret key MSK_i as long as they do not violate the constraint on the challenge messages and the policy F.

Guess The attacker submits a guess b' for b. The attacker wins if $b' = b$. The attacker's advantage in this game is defined to be $\mid Pr[b = b'] - 1/2 \mid$

We call the game described above as IND security game. We say the MAFE scheme is IND secure if the advantage described above is negligible.

Theorem 3. IND *secure multi-authority functional encryption for general circuits secure against corruption of one authority and zero token queries imply indistinguishability obfuscation for $P/Poly$.*

Proof. We assume that there exists a scheme MAFE with desired properties. Let the size of the circuit C to be obfuscated be k. Now, we describe our candidate obfuscator that works as follows:

1. Perform $\mathsf{MAFE.Authority\ Setup}(1^k, 1) \rightarrow (PK, SK)$. This authority is just labelled as 1.
2. Output the obfuscation as $i\mathcal{O}(C) \leftarrow (CT = \mathsf{MAFE.Encrypt}(PK, U, id, C), SK)$. Here, U is the policy that on input (x, C) computes $C(x)$ and id is the identity permutation.

Evaluation at x, works by using SK to compute a token for (UID, x) for any identity UID and then decrypting CT using this token.

To argue security, note that when two circuits (C_0, C_1) are two equivalent circuits, then due to the security of MAFE scheme it holds that $\{i\mathcal{O}(C_0) = (CT = \mathsf{MAFE.Encrypt}(PK, U, id, C_0), SK)\}$ and $\{i\mathcal{O}(C_1) = (CT = \mathsf{MAFE.Encrypt}(PK, U,$
$id, C_1), SK)\}$ are computationally indistinguishable. Correctness of obfuscation follows from the correctness of the scheme. □

It takes similar techniques to prove the following result.

Theorem 4. IND *secure multi-authority functional encryption for n authorities, secure against zero corruptions and two token queries per authority imply indistinguishability obfuscation for n bit inputs.*

Proof. Assume that there exists a scheme MAFE with desired properties. To obfuscate C that takes n bit inputs, obfuscator $i\mathcal{O}(C)$ does the following:

1. Performs setup $\mathsf{MAFE.Authority\ Setup}(i, 1^k) \rightarrow (MPK_i, MSK_i)$ for $i \in [n]$ (authorities labelled in $[n]$).
2. Selects an identity randomly UID, and from each authority gets tokens $\{K_{i,b} \leftarrow \mathsf{MAFE.KeyGen}(MSK_i, UID, b) \forall i \in [n], b \in \{0, 1\}\}$.
3. Finally encrypts C as, $CT \leftarrow \mathsf{MAFE.Encrypt}(\{MPK_i\}_{i\in[n]}, U, id, C)$. Here U is the policy that on input $(x_1, .., x_n, C)$ for $x_i \in \{0, 1\} \forall i \in [n]$ outputs $C(x_1, .., x_n)$ and id is the identity permutation.
4. Output as $(CT, \{K_{i,0}, K_{i,1}\}_{i\in[n]})$.

In order to evaluate the obfuscation on input x, the evaluator outputs $C(x) = \mathsf{MAFE.Decrypt}(\{K_{i,x_i}\}_{i\in[n]}$ $, CT)$. Note that correctness of obfuscation follows from the correctness of the scheme.

For proving indistinguishability, we note that due to IND security of multi-authority functional encryption scheme obfuscations of the two circuits are computationally indistinguishable

$$(\mathcal{O}(C_0) = \{MPK_i, K_{i,0}, K_{i,1} \forall i \in [n]\}, \mathsf{MAFE.Encrypt}(\{MPK_i\}_{i\in[n]}, U, id, C_0)) \approx_c$$

$$(\mathcal{O}(C_1) = \{MPK_i, K_{i,0}, K_{i,1} \forall i \in [n]\}, \mathsf{MAFE.Encrypt}(\{MPK_i\}_{i\in[n]}, U, id, C_1))$$

□

3.1.2 Simulation based definition

Previous works on functional encryption have also considered a simulation based definition [GKP$^+$13, GGG$^+$14] for functional encryption. It was proved in [AGVW13] that it in general it is impossible to achieve simulation based security for functional encryption for general circuits and arbitrary many key queries. Our primitive with simulation security even for corruption of one authority and zero token queries imply black box obfuscation, which has been shown to be impossible to exist [BGI$^+$01]. The key notion in the security definition is that there exist a p.p.t. simulator Sim that given the tokens, corrupted authority secret keys and the public keys and access to an oracle that outputs what is learnable in the "real world" should be able to produce a simulated cipher-text for any chosen message and policy.
For ease of notation, let us assume that the authorities are labelled in $[N]$ and the permutation given to the encryption algorithm is implicitly assumed to be the identity permutation.

$\mathsf{Exp}^{\mathsf{real}}_{\mathsf{MAFE},\mathcal{A}}(1^{\mathsf{k}})$ $\qquad\qquad$ $\mathsf{Exp}^{\mathsf{ideal}}_{\mathsf{MAFE},\mathcal{A},\mathsf{Sim}}(1^{\mathsf{k}})$

$$\mathcal{A}_1(1^k) \to ([N],[S] \subset [N], z_1)$$

$$\{\mathsf{MAFE.Authority\ Setup}(i,1^k) \to (MPK_i, MSK_i)\}_{i\in[N]\setminus[S]}$$

$$\mathcal{A}_2(1^k, \{MPK_i\}_{i\in[N]\setminus[S]}, z_1) \to (\{MPK_i, MSK_i\}_{i\in[S]}, z_2)$$

$$\mathcal{A}_3(1^k, z_2) \to (\{(UID_j, U_{i,j})\}_{i\in[N]\setminus S, j\in[q=poly(k)]}, z_3)$$

$$\{K_{i,j} \leftarrow \mathsf{MAFE.KeyGen}(MSK_i, UID_j, U_{i,j})\}_{i\in[N]\setminus S, j\in[q]}$$

$$\mathcal{A}_4(\{K_{i,j}\}_{i\in[N]\setminus S, j\in[q]}, z_3) \to (F, x, z_4)$$

$$CT \leftarrow \mathsf{MAFE.Encrypt}(\{MPK_i\}_{i\in[N]}, F, x)$$

$$\tilde{CT} \leftarrow \mathsf{Sim}^{\mathcal{O}(\cdot)(x,F,\{(UID_j,U_{i,j})\}_{i\in[N]\setminus S, j\in[q=poly(k)]})}(1^k, \{MPK_i\}_{i\in[N]}, \{MSK_i\}_{i\in[S]})$$

Output (CT, z_4) $\qquad\qquad$ Output $(\tilde{CT}, z_4)$

Figure 1: Simulation security

Definition 5. *The scheme is said to be* (q) SIM*-secure against* S *corruptions if there exists a p.p.t. simulator Sim such that for all of p.p.t. adversaries* $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4)$*, the outcomes of the two experiments are computationally indistinguishable:*

$$\{\mathsf{Exp}^{\mathsf{real}}_{\mathsf{MAFE},\mathcal{A}}(1^k)\}_{k\in\mathbb{N}} \approx_c \{\mathsf{Exp}^{\mathsf{ideal}}_{\mathsf{MAFE},\mathcal{A},\mathsf{Sim}}(1^k)\}_{k\in\mathbb{N}}$$

The oracle $\mathcal{O}$ *on a query* $(U_1, .., U_N)$ *checks that there exists some* $UID \in \{UID_j\}_{j\in[q]}$ *such that key for* (UID, U_i) *was issued by all authorities* $i \in [N] \setminus [S]$ *in step 5. If this happens the oracle outputs* $F(U_1, .., U_N, x)$*, otherwise it outputs* $\perp$*.*

Theorem 5. *(0)* SIM*-secure multi-authority functional encryption for circuits secure against corruption of one authority imply black box obfuscation.*

Proof. In order to prove the theorem we assume that there exists a scheme MAFE with desired properties. Let the size of the circuit C to be obfuscated be k. We now describe our obfuscator.

1. Perform $\mathsf{MAFE.Authority\ Setup}(1, 1^k) \rightarrow (PK, SK)$. This authority is just labelled as 1.
2. Output the obfuscation as $\mathcal{O}(C) \leftarrow (CT = \mathsf{MAFE.Encrypt}(PK, U, id, C), SK)$. Here, U is the policy, that on input x, C computes $C(x)$ and id is the identity permutation.

Evaluation at x, works by using SK to compute a token for UID, x for any identity UID and then decrypting the encryption using this token.
To argue security, we need to show that there exists a polynomial time simulator S such that for all polynomial sized strings aux, $(\mathcal{O}(C), aux) \approx_c (S^{C(\cdot)}(1^k), aux)$. We describe our simulator below:

1. Runs $\mathsf{MAFE.Authority\ Setup}(1, 1^k) \rightarrow (PK, SK)$.
2. Then runs the simulator $\mathsf{MAFE.Sim}$ as: $\tilde{CT} \leftarrow \mathsf{MAFE.Sim}^{U(\cdot,C)}(PK, SK)$. Queries to the oracle $U(\cdot, C)$ can be answered by forwarding the query to the oracle $C(\cdot)$ and forwarding its response as it is.
3. Output $(\tilde{CT}, SK)$

If there is an adversary $\mathcal{B}$ that breaks the security of the obfuscation scheme, then there is an adversary $\mathcal{A}$ that attacks the security of the underlying scheme. This reduction is straightforward hence we omit the details. □

Theorem 6. *(2)* SIM*-secure multi-authority functional encryption scheme for $n = n(k)$ authorities secure against corruption of zero authorities imply black box obfuscation for n bit inputs.*

Proof. Assume that there exists a scheme MAFE with desired properties. To obfuscate C, obfuscator $\mathcal{O}(C)$ does the following:

1. Performs setup $\mathsf{MAFE.Authority\ Setup}(i, 1^k) \rightarrow (MPK_i, MSK_i)$ for $i \in [n]$ (authorities labelled in $[n]$).
2. Selects an identity randomly UID, and from each authority gets tokens $\{K_{i,b} \leftarrow \mathsf{MAFE.KeyGen}(MSK_i, UID, b) \forall i \in [n], b \in \{0, 1\}\}$.
3. Finally encrypts C as, $CT \leftarrow \mathsf{MAFE.Encrypt}(\{MPK_i\}_{i\in[n]}, U, id, C)$. Here U is the policy that on input $(x_1, .., x_n, C)$ for $x_\in\{0, 1\} \forall i \in [n]$ outputs $C(x_1, .., x_n)$ and id is the identity permutation.
4. Output obfuscated circuit as $(CT, \{K_{i,0}, K_{i,1}\}_{i\in[n]})$.

In order to evaluate the obfuscation on input x, evaluator outputs $C(x) = \mathsf{MAFE.Decrypt}(\{K_{i,x_i}\}_{i\in[n]}, CT)$. Note that correctness of obfuscation follows from the correctness of the scheme.

To argue security, we need to show that there exists a polynomial time simulator S such that for all strings aux, $(\mathcal{O}(C), aux) \approx_c (S^{C(\cdot)}(1^k), aux)$. We describe our simulator as follows:

1. Performs setup $\mathsf{MAFE.Authority\ Setup}(i, 1^k) \rightarrow (MPK_i, MSK_i)$ for $i \in [n]$ (authorities labelled in $[n]$).

2. Selects an identity randomly UID, and from each authority gets tokens: $\{K_{i,b} \leftarrow \mathsf{MAFE.KeyGen}(MSK_i, UID, b) \forall i \in [n], b \in \{0,1\}\}$.

3. Then runs the simulator of the multi-authority functional encryption scheme and computes: $\tilde{CT} \leftarrow \mathsf{MAFE.Sim}^{U(\{0,1\},\{0,1\},..,\{0,1\},C)}(MPK_1,..,MPK_n)$. Queries to the oracle $U(\{0,1\},\{0,1\},..,\{0,1\},C)$ can be answered by forwarding the query $x = (x_1,..,x_n)$ to the oracle $C(\cdot)$ and forwarding its response as it is.

4. Output $(\tilde{CT}, \{K_{i,0}, K_{i,1}\}_{i\in[n]})$

If there is an adversary $\mathcal{B}$ that breaks the security of the obfuscation scheme, then there is an adversary $\mathcal{A}$ that attacks the security of the underlying scheme. This reduction is straightforward; hence we omit the details. □

3.2 Delegatable Functional Encryption

In this section we describe the syntax and the security definition of a key delegatable functional encryption (DFE) scheme. A DFE scheme for a class of function $\mathcal{F}_k$ is a tuple of five algorithms: $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Del}, \mathsf{Dec})$.

- $\mathsf{Setup}(1^k, n) \rightarrow (MPK, MSK)$. The setup algorithm takes the security parameter 1^k and a parameter $n(k)$ which is the number of times a function key can be delegated.
- $\mathsf{Enc}(MPK, m) \rightarrow CT$. The encryption algorithm takes in a message in $\mathcal{M}_k$ and the master public key and outputs a cipher-text.
- $\mathsf{KeyGen}(MSK, f) \rightarrow SK_f$. The key generation algorithm takes in the master secret key and a function f and outputs a key SK_f
- $\mathsf{Del}(SK_f, f') \rightarrow SK_{f'\cdot f}$. The delegation algorithm takes a key SK_f (which may have been delegated by at most $n-1$ times) and a function f' and outputs a key for the function $f' \cdot f$ (f composed with f').
- $\mathsf{Dec}(SK_f, CT) \rightarrow f(m)$. The decryption algorithm takes a cipher-text CT and a function key for f and outputs $f(m)$.

We require the scheme to satisfy correctness informally defined as follows.
Correctness: We require that for honestly generated setup (MPK, MSK), for keys SK_f of a function f, generated or delegated at most n times, and any honestly generated cipher-text CT encrypting m, the decryption should return $f(m)$ with over-whelming probability.

3.2.1 Security Notion

We now present the security definition for DFE scheme. We define security notion of the scheme by the following game between a challenger and an attacker.

Setup: The challenger does setup to compute (MSK, MPK) and hands over the public key

MPK to the adversary.

Key Query Phase 1 The attacker can submit queries of two kinds

1. Submit f and get back $SK_f \leftarrow \mathsf{KeyGen}(MSK, f)$.
2. Submit a tuple $(f_1, .., f_{n+1})$. In this case the challenger computes a key for f_1 and then runs the delegation algorithm to compute a key for $f_{n+1} \cdot ... \cdot f_1$. The adversary can set $f_{i+1}, .., f_{n+1}$ for some $i \geq 1$ as empty in the query, in which case the delegation is done $i-1$ times.

Challenge Phase The attacker selects two messages (m_0, m_1) in which case it should satisfy that $f(m_0) = f(m_1)$ and $f_{n+1} \cdot .. \cdot f_1(m_0) = f_{n+1} \cdot .. \cdot f_1(m_1)$ for function queries of both kinds above. Challenger encrypts m_b as CT where $b \in \{0,1\}$ is randomly chosen and hands it over to the adversary.

Key Query Phase 2 : The adversary may make queries as in phase 1 as long as they satisfy the constraint with the challenge message as described in the challenge phase description.

Guess The attacker submits a guess $b^{'}$ for b. The attacker wins if $b^{'} = b$. The attacker's advantage in this game is defined to be $\mid Pr[b = b^{'}] - 1/2 \mid$

We call the game described above as IND security game.

Definition 6. *A* DFE *system is* (IND) *secure if all polynomial time adversaries have at most a negligible advantage in the* (IND) *security game.*

4 Our Construction for MAFE

Notation: Let k denote the security parameter and n denote the bound on the number of authorities used while encrypting the cipher-text. Let $\mathcal{O}$ denote a $(1, 2^{-2(n+1)k})$ weak extractability obfuscator for general circuits. $F = (\mathsf{setup}, \mathsf{puncture}, \mathsf{eval})$ be a sub exponential puncturable psuedo random function (slightly abusing notation, we refer $F(K, x)$ as evaluation of PRF at x even if the key is punctured) with constant c_{PRF}. Let P be a sub-exponentially secure injective one way function with security constants c_{owp1} and c_{owp2}. In this section, while giving $(a_1, .., a_n)$ as an input to a circuit or a PRF F, we mean that the input is given as a concatenation of $a_1||..||a_n$.

- $\mathsf{MAFE.Authorithy\ Setup}(i, 1^k)$: Each authority i runs $F.\mathsf{setup}(1^{\mu}) \rightarrow K_i$ and obfuscates (using the obfuscator $\mathcal{O}$) the following circuit as VK. On input (x, σ), the circuit checks that $P(\sigma) = P(F(K_i, x))$, if the check passes it outputs 1 otherwise it outputs 0. Then it sets $MPK_i = VK$
  It generates MSK_i as an obfuscation of the circuit (using the obfuscator $\mathcal{O}$) that on input x outputs $F(K_i, x)$.

  Here μ is set such that it is greater than $(nk + 2k)^{1/c_{PRF}}$ and the output length is greater than $max\{(nk+2k)^{1/c_{owp2}}, (nk+2k)^{2/c_{owp1}}\}$.

- $\mathsf{MAFE.KeyGen}(MSK_i, UID, U)$: $K_{UID,i,U} \leftarrow MSK_{id}(UID, U)$. On input user identity UID, property U and secret key for authority i, KeyGen outputs a PRF evaluation on (UID, U) using the circuit MSK_i.

- $\mathsf{MAFE.Encrypt}(\{MPK_i\}_{i\in\mathcal{I}}, G, \rho, m)$: Let C be the circuit described in figure Figure 2 (initialised with hard-wirings m, G, ρ). Output $(G, \rho, \mathcal{O}(C))$.

- $\mathsf{MAFE.Decrypt}(\{K_{UID,i,U_i}\}_{i\in\mathcal{I}}, G, CT)$. The decryption algorithm gets a policy, an obfuscated circuit and some tokens for authorities in $\mathcal{I}$ as input (if the decryptor does not receive a token from an authority its treated as null) and just evaluates the circuit using the tokens. Specifically it computes and outputs, $CT(UID, \{U_i\}_{i\in\mathcal{I}}, \{K_{UID,i,U_i}\}_{i\in\mathcal{I}})$.

Remark. Although we instantiate scheme with a sub-exponentially secure injective one way function P, in practice we can instantiate it with any one-one function. For the security proof, we see that the input output behaviour of MPK_i do not change when it is instantiated with any one-one function, hence we can switch to a hybrid when it is instantiated by sub-exponentially injective secure one way function and due to the security of obfuscation these two hybrids are close.
A quick inspection of the algorithms shows that the scheme described above is correct and we omit details of the proof here.

Constants: $m, G, \rho, \{MPK_i\}_{i\in\mathcal{I}}$.
Input: $UID, \{U_i\}_{i\in\mathcal{I}}, \{K_{UID,i,U_i}\}_{i\in\mathcal{I}}$. (some of U_i's can be null in which case K_{UID,i,U_i} is also empty)
For all $i \in \mathcal{I}$ perform the following checks and output $\perp$ if any of the check fails.

- Check that either U_i is null, (in which case K_{UID,i,U_i} is also null), or
- Check that $MPK_i(UID, U_i, K_{UID,i,U_i}) = 1$

Output $G(<\{U_i\}_{i\in\mathcal{I}}>, m)$.

Figure 2: Program Encrypt

4.1 Security Proof

Theorem 7. *Assuming existence of a subexponentially secure indistinguishability obfuscator and a subexponentially secure injective one way function there exist a* IND *secure multi-authority functional encryption.*

Proof. We now prove that the construction described in section 4 is a IND secure multi-authority functional encryption scheme.
Notation: For ease of notation, we label authorities in $[n]$ for some polynomial $n(k)$. We use μ as the security parameter for the puncturable PRF. c_{PRF} denotes the security constant for the puncturable PRF and c_{owp1} and c_{owp2} denote the security constants for sub-exponential injective

one way function. For ease of notation, we also assume that ρ i.e. the permutation associated with the policy function is an identity function on $[n]$. We now present a sequence of hybrids and prove indistinguishability.

$\mathcal{H}_0$: The standard IND game for multi-authority functional encryption while encrypting m_0.

1. The adversary $\mathcal{A}$ receives the security parameter and outputs $[n]$ as the set of authorities. The challenger does the setup for all authorities in $[n]$.
2. $\mathcal{A}$ can adaptively query for tokens and can corrupt authorities. When $\mathcal{A}$ asks for the secret key for authority i, the challenger releases MSK_i.
3. $\mathcal{A}$ now queries tuples of the form (UID, i, U), interpreting to issue a token for a user with identity UID and property U by authority i.
4. Challenger generates corresponding tokens $K_{UID,i,U} \leftarrow MSK_i(UID, U)$ and gives them to $\mathcal{A}$.
5. Adversary then declares (m_0, m_1, G), these messages satisfy the constraint as outlined in the definition.
6. Challenger encrypts m_0 with policy G and gives (G, CT) to the $\mathcal{A}$.
7. $\mathcal{A}$ can adaptively query for tokens and can corrupt authorities. When $\mathcal{A}$ asks for the secret key for authority i, the challenger releases MSK_i.
8. $\mathcal{A}$ now queries tuples of the form $(UID, i, U,)$, interpreting to issue a token for a user with identity UID and property U by authority i.
9. $\mathcal{A}$ outputs a guess b'

$\mathcal{H}_x$: For $x \in [1, 2^{(n+1)k} + 1]$, this hybrid is similar to the previous hybrid except that the challenger outputs an obfuscation of Figure 3 (initialised with $m_0, m_1, x, G, \{MPK_i\}_{i\in[n]}$) along with G as the challenge cipher-text.

$\mathcal{H}_{2^{(n+1)k}+2}$: This hybrid corresponds to the actual IND game when the challenger encrypts m_1.

Let us now denote by γ the supremum of the quantity $\mid Pr[D(\mathcal{O}(C_0)) = 1] - Pr[D(\mathcal{O}(C_1)) = 1] \mid$, where D is a polynomial time non uniform distinguisher and C_0, C_1 be polynomially sized arbitrary functionally equivalent circuits. In our instantiation, $\gamma < 2^{-(n+2)k}$ (Otherwise, there is an extractor which will be able to extract a differing point with overwhelming probability; a contradiction).

Lemma 2. *$\mathcal{H}_0$ is indistinguishable from hybrid $\mathcal{H}_1$.*

Proof. Only thing to show here is that the circuits whose obfuscation is given out in these hybrids are equivalent. It is easy to see that the circuit described in Figure 2 when initialised with $(m_0, G, \{MPK_i\}_{i\in[n]})$ and the circuit in Figure 3 initialised with $(m_0, m_1 m_0, m_1, x, G, \{MPK_i\}_{i\in[n]})$ with $x = 1$ are equivalent. Hence, $\mid Pr[\mathcal{A}(\mathcal{H}_0) = 1] - \mid Pr[\mathcal{A}(\mathcal{H}_1) = 1] \mid < \gamma$. The reduction is straight forward and we omit the details. $\square$

Constants: $m_0, m_1, x, G, \{MPK_i\}_{i \in [n]}$.
Input: $(UID, U_1, .., U_n, \{K_{UID,i,U_i}\}_{i \in [n]})$. (some of U_i's can be null in which case K_{UID_i,U_i} is also empty)
For all $i \in [n]$ perform the following checks and output $\perp$ if any of the check fails.

- Check that either U_i is null, (in which case K_{UID,i,U_i} is also null), or
- Check that $MPK_i(UID, U_i, K_{UID,i,U_i}) = 1$

If $UID, U_1, .., U_n < x - 1$ output $G(U_1, .., U_n, m_1)$ else $G(U_1, .., U_n, m_0)$.

Figure 3: Program Encrypt$_x$

Lemma 3. *$\mathcal{H}_{2^{(n+1)k}+1}$ is indistinguishable from hybrid $\mathcal{H}_{2^{(n+1)k}+2}$.*

Proof. The proof of this statement is similar to the proof described for lemma 2. Only thing to show here is that the circuits whose obfuscation is given out in these hybrid are equivalent. It is easy to see that the circuit described in Figure 2 when initialised with $m^1, G, \{MPK_i\}_{i \in [n]}$ and the circuit described in Figure 3 when initialised with $m_0, m_1, x, G, \{MPK_i\}_{i \in [n]}$ and $x = 2^{nk+k} + 1$ are equivalent. Hence, $\mid Pr[D(\mathcal{H}_{2^{(n+1)k}+1}) = 1] - \mid Pr[D(\mathcal{H}_{2^{(n+1)k}+2}) = 1] \mid < \gamma$. $\square$

Lemma 4. *For any $x \in [1, 2^{(n+1)k} + 1]$, $\mid Pr[\mathcal{A}(\mathcal{H}_x) = 1] - Pr[\mathcal{A}(\mathcal{H}_{x+1}) = 1] \mid < O(n \cdot 2^{-(n+2)k})$ for any polynomial time adversary $\mathcal{A}$.*

Proof. For this lemma, let us denote $x - 1 = (UID', U'_1, U'_2, .., U'_n)$ i.e. concatenation of $n + 1$ strings. We now list down the following sub hybrids:

$\mathcal{H}_{x,1}$: This hybrid is similar to hybrid $\mathcal{H}_x$ except that MSK_i for any authority i is generated differently. Challenger first samples a PRF key K_i. It parses $x - 1 = (UID', U'_1, U'_2, .., U'_n)$. Then, it computes a punctured key $K'_i \leftarrow F.\mathsf{Puncture}(K_i, UID', U'_i)$ and $\alpha_i \leftarrow F(K_i, UID', U'_i)$. MSK_i is now given out as a weak extractability obfuscation of circuit described in Figure 4. Informally, $\mathcal{H}_x$ and $\mathcal{H}_{x,1}$ are indistinguishable due to the security of obfuscation and the correctness of the puncturable PRF.

$\mathcal{H}_{x,2}$: This hybrid is similar to hybrid $\mathcal{H}_{x,1}$ except that MPK_i for any authority i is gener-

Hard-wired: $K'_i, \alpha_i, UID', U'_i$.
Input: x

- If $x \neq (UID', U'_i)$ output $F(K'_i, x)$ else output α_i

Figure 4: Secret* circuit

ated differently. Challenger first samples a PRF key K_i. It parses $x - 1 = (UID', U'_1, U'_2, .., U'_n)$. Then, it computes a punctured key $K'_i \leftarrow F.\mathsf{Puncture}(K_i, UID', U'_i)$ and $\alpha_i \leftarrow F(K_i, UID', U'_i)$.

MSK_i is now given out as a weak extractability obfuscation of circuit described in Figure 5. Informally, $\mathcal{H}_{x,1}$ and $\mathcal{H}_{x,2}$ are indistinguishable due to the security of obfuscation and the correctness of the puncturable PRF.

$\mathcal{H}_{x,3}$: This hybrid is the same as the previous hybrid except that for all $i \in [n]$, α_i is chosen

Hard-wired: $K_i', P(\alpha_i), UID', U_i'$.
Input: x, σ

- If $x \neq (UID', U_i')$ check that $P(F(K_i', x)) = P(\sigma)$ and output 1 if the check passes; 0 otherwise.
- Else check that $P(\alpha_i) = P(\sigma)$ and output 1 if the check passes; 0 otherwise.

Figure 5: Verifier* circuit

randomly. Informally, $\mathcal{H}_{x,2}$ and $\mathcal{H}_{x,3}$ are indistinguishable due to the security of the puncturable PRF.

$\mathcal{H}_{x,4}$: This hybrid is the same as the previous hybrid except that the challenge cipher-text is computed as an obfuscation of circuit in 3 hard-wired with $x+1$.. Informally, $\mathcal{H}_{x,3}$ and $\mathcal{H}_{x,4}$ are indistinguishable due to the security of the injective one way function P.

$\mathcal{H}_{x,5}$: This hybrid is the same as the previous hybrid except that for all $i \in [n]$, α_i is computed as an actual PRF evaluation. . Informally, $\mathcal{H}_{x,4}$ and $\mathcal{H}_{x,5}$ are indistinguishable due to the security of the puncturable PRF.

$\mathcal{H}_{x,6}$: This hybrid is the same as the previous hybrid except that for any authority i, MSK_i is computed as an obfuscation of circuit described in the Authority Setup algorithm. Informally, $\mathcal{H}_{x,5}$ and $\mathcal{H}_{x,6}$ are indistinguishable due to the security of obfuscation and the correctness of the puncturable PRF.

$\mathcal{H}_{x,7}$: This hybrid is the same as the previous hybrid except that for any authority i, MPK_i is computed as an obfuscation of circuit described in Authority Setup algorithm. Informally, $\mathcal{H}_{x,6}$ and $\mathcal{H}_{x,7}$ are indistinguishable due to the security of obfuscation and the correctness of the puncturable PRF. This hybrid is same as the hybrid $\mathcal{H}_{x+1}$.

Claim 1. *For any p.p.t. distinguisher D, $\mid Pr[D(\mathcal{H}_x) = 1] - Pr[D(\mathcal{H}_{x,1}) = 1] \mid \leq O(n \cdot 2^{-nk-2k})$.*

Proof. The difference between hybrids $\mathcal{H}_x$ and $\mathcal{H}_{x,1}$ is the generation of public keys MSK_i for all authorities $i \in [n]$. Note that in $\mathcal{H}_x$ it is generated honestly as described in the Authority Setup using the PRF key K. In $\mathcal{H}_{x,1}$ it is generated as an obfuscation of circuit described in Figure 4 (initialised with punctured key K_i' and $F(K_i, UID, U_i)$). These two circuits are functionally equivalent. We can hence define a sequence of $\mid [n] \setminus S \mid$ hybrids where we replace one by one MSK_i from being genuinely constructed to one constructed as in $\mathcal{H}_{x,1}$. Since $\mathcal{O}$ is $(1, 2^{-(2n+2)k})$ weak extractability obfuscator the differing advantage between each of these hybrids is less than $2^{-(n+2)k}$. Hence, overall $\mid Pr[D(\mathcal{H}_x) = 1] - Pr[D(\mathcal{H}_{x,1}) = 1] \mid \leq O(n \cdot 2^{-nk-2k})$. The reduction between these hybrids is straight forward and we omit the details. □

Claim 2. *For any p.p.t. distinguisher D, $\mid Pr[D(\mathcal{H}_{x,1}) = 1] - Pr[D(\mathcal{H}_{x,2}) = 1] \mid \leq O(n \cdot 2^{-nk-2k})$.*

Proof. The difference between hybrids $\mathcal{H}_{x,1}$ and $\mathcal{H}_{x,2}$ is the generation of public keys MPK_i for all authorities $i \in [n]$. Note that in $\mathcal{H}_{x,1}$ it is generated honestly as described in the Authority Setup using the PRF key K. In $\mathcal{H}_{x,2}$ it is generated as an obfuscation of circuit described in Figure 5 (initialised with punctured key K_i' and $P(F(K_i, UID, U_i))$). These two circuits are functionally equivalent. We can hence define a sequence of $\mid [n] \setminus S \mid$ hybrids where we replace one by one MPK_i from being genuinely constructed to one constructed as in $\mathcal{H}_{x,2}$. Since $\mathcal{O}$ is $(1, 2^{-(2n+2)k})$ weak extractability obfuscator the differing advantage between each of these hybrids is less than $2^{-(n+2)k}$. Hence, overall $\mid Pr[D(\mathcal{H}_{x,1}) = 1] - Pr[D(\mathcal{H}_{x,2}) = 1] \mid \leq O(n \cdot 2^{-nk-2k})$. The reduction between these hybrids is straight forward and we omit the details. □

Claim 3. *For any p.p.t. distinguisher D, $\mid Pr[D(\mathcal{H}_{x,2}) = 1] - Pr[D(\mathcal{H}_{x,3}) = 1] \mid \leq O(n \cdot 2^{-nk-2k})$.*

Proof. This lemma follows from the property that puncturable PRF's output is psuedo-random at punctured point given the punctured key (sub-exponential security of the puncturable PRF). We parse $x - 1 = (UID', U_1', .., U_n')$.
This proof goes through by a sequence of n hybrids where for each authority $i \in [n]$, $(K_i', \alpha_i \leftarrow F(K_i, UID', U_i'))$ is replaced with $(K_i', \alpha_i \xleftarrow{\$} \mathcal{R}$ for all $i \in [n]$ (Here K_i is the PRF key and K_i' is the punctured key at point (UID', U_i')). This can be done because in both these hybrids, MSK_i and the encryption keys MPK_i use only the punctured keys and a the value of the PRF at the punctured point. Here, $\mathcal{R}$ is the co-domain of the PRF, which is equal to the domain of the injective one way function P. Since, PRF is sub exponentially secure with parameter c_{PRF} (c_{PRF} be the security constant of the PRF) when PRF is initialised with parameter greater than $(nk+2k)^{1/c_{PRF}}$, distinguishing advantage between each intermediate hybrid is bounded by $O(2^{nk-2k})$. The reduction is straight forward and we omit the details. □

Claim 4. *For any p.p.t. distinguisher $\mathcal{A}$, $\mid Pr[\mathcal{A}(\mathcal{H}_{x,3}) = 1] - Pr[\mathcal{A}(\mathcal{H}_{x,4}) = 1] \mid \leq O(2^{-nk-2k})$.*

Proof. Let Q denote a random variable that is defined as follows: $Q = 0$ when in any hybrid $\mathcal{H}_{x,2}$, (m_0, m_1, G) is given out by the adversary such that $G(x-1, m_0) = G(x-1, m_1)$ and $Q = 1$ otherwise. We now make two observations:

1. For any PPT $\mathcal{A}$, $\mid Pr[\mathcal{A}(\mathcal{H}_{x,2}) = 1/Q = 0] - Pr[\mathcal{A}(\mathcal{H}_{x,3}) = 1/Q = 0] \mid < 2^{-(n+2)k}$.
   For any adversary $\mathcal{A}$ and a hybrid challenger C such that $Q = 0$, If $\mathcal{A}$ has an advantage greater than $2^{-(n+2)k}$, then there exists an algorithm $\mathcal{B}$ that breaks indistinguishability obfuscation property of $\mathcal{O}$ with the same advantage.

   $\mathcal{B}$ interacts with the obfuscation challenger as follows. $\mathcal{B}$ invokes $\mathcal{A}$ and the challenger of the hybrid. The hybrid challenger runs the setup for corresponding authorities. Then $\mathcal{A}$ queries for some tokens and the challenger replies to the query by giving out the tokens for those authorities as done in the hybrid $\mathcal{H}_{x,3}$. Finally, $\mathcal{A}$ outputs (m_0, m_1, G). Now, obfuscation challenger gives $\mathcal{B}$ a string z which is the set of revealed public keys and tokens as well as all the master secret keys.

Now, $\mathcal{B}$ sets C_0 as the program described in Figure 3 initialised with hard-wirings $(m_0, m_1, G, x, \{MPK_i\}_{i\in[n]})$ and C_1 as the program described in Figure 3 initialised with $(m_0, m_1, G, x + 1, \{MPK_i\}_{i\in[n]})$. Obfuscation challenger randomly picks up $b \in \{0, 1\}$. If $b = 0$, it obfuscates C_0 otherwise it obfuscates C_1. $\mathcal{B}$ directly sends this obfuscation to $\mathcal{A}$ as the challenge ciphertext. If $b = 0$, view of $\mathcal{A}$ is simulated as $\mathcal{H}_x$ otherwise it is simulated as $\mathcal{H}_{x+1}$. $\mathcal{B}$ finally outputs whatever $\mathcal{A}$ does. Hence, advantage of $\mathcal{B}$ is at least as much as the advantage of $\mathcal{A}$ in distinguishing the hybrids. Since, $\mathcal{O}$ is a secure indistinguishability obfuscator with security upper gap bounded by $2^{-(nk+2k)}$, the claim holds.

2.
$$Pr[Q = 1]\cdot \mid Pr[\mathcal{A}(\mathcal{H}_{x,2}) = 1/Q = 1] - Pr[\mathcal{A}(\mathcal{H}_{x,3}) = 1/Q = 1] \mid <$$
$$max\{O(2^{-(nk-2k)}), O(1/2^{\mu^{c_{owp1}}-n-1}), O(2^{-\mu^{c_{owp2}}/2+n+1})\}$$

Here $c_{owp1} > 0$ and $c_{owp2} > 0$ are constants for the sub-exponential injective one way function.

Let us now assume that $Pr[Q = 1]\epsilon_{Q=1}/2^{n+1} > 2^{-nk-2k}/2^{n+1}$ as otherwise the claim trivially holds. Now, if such a condition is true then we construct an inverter that inverts the injective one way function with probability $p = Pr[Q = 1]\epsilon_{Q=1}/2^{n+1}$ and takes time $t = O(2^{2n}/\epsilon^2_{Q=1} \cdot poly(k, n))$. Again we assume that, $p > \Omega(2^{-\mu^{c_{owp2}}})$ or otherwise there is nothing to prove and the observation trivially holds. If this assumption is true then it must mean that $t > \Omega(2^{\mu^{c_{owp1}}})$ (due to the security of injective one way function), which gives us our desired result on some manipulation.

We now assume $Pr[Q = 1]\epsilon_{Q=1}/2^{n+1} > 2^{-nk-2k-n-1}$ and describe our inverter $\mathcal{B}$. $\mathcal{B}$ works as follows:

(a) Invokes $\mathcal{A}$ and gets $[n]$. It sends n to the injective one way function challenger and gets $(P(r_1), .., P(r_n))$ as the challenge.

(b) $\mathcal{B}$ samples the punctured PRF keys K'_i for all $i \in [n]$ (Punctured at (UID', U'_i)) and uses $P(r_i)$ along with the key to generate the public key MPK_i as per circuit described in Figure 5.

(c) When $\mathcal{A}$ asks for any MSK_i it asks the injective one way function challenger for r_i and uses this value along with K'_i to generate MSK_i.

(d) When $\mathcal{A}$ asks for tokens by querying (UID, i, U) it computes and sends $F(K'_i, UID, U)$ if $UID, U \neq UID', U'_i$, otherwise it asks injective one way function challenger for r_i and sends it over to $\mathcal{A}$.

(e) $\mathcal{A}$ declares (m_0, m_1, G). If $G(x - 1, m_0) = G(x - 1, m_1)$, $\mathcal{B}$ aborts.

(f) Let C_0 be the program described in Figure 3 initialised with $(m_0, m_1, x, G, \{MPK_i\}_{i\in[n]})$ and C_1 as the program described in Figure 3 initialised with $(m_0, m_1, x + 1, G, \{MPK_i\}_{i\in[n]})$. Let S denote the set and $i \in S$ for which either MSK_i was queried till this point or (UID', i, U'_i) was submitted as a token query.

(g) $\mathcal{B}$ now selects a set $t \subsetneq [n]$ such that $t \cup S \neq [n]$ and gets r_i for all $i \in t$. Our extractor now computes r_i for all $i \in [n]$, using these values. $\mathcal{B}$ runs an extractor $(r_1, .., r_n) \leftarrow E(C_0, C_1, z)$ on the following distinguisher L where z is the auxiliary input set as all the messages given to $\mathcal{A}$ till the previous point, keys K'_i for all $i \in [n]$ and r_i for all $i \in t \cup S$.

i. L uses z to invoke $\mathcal{A}$ till the previous point.

ii. $\mathcal{A}$ now gets an obfuscation of C_0 or C_1.

iii. $\mathcal{A}$ can query for MSK_i and tokens by querying (UID, i, U). These are generated using z. If the response cannot be generated (when r_i needs to be known for $i \notin t \cup S$) it outputs 0.

iv. $\mathcal{A}$ outputs b'. If for all $i \in t \cup S$, r_i's were required by L, output b' otherwise output 0.

Note that any valid $\mathcal{A}$ does not ask to invert all $r_i \forall i \in [n]$. This is because $G(x-1, m_0) \neq G(x-1, m_1)$ hence it does not form a valid query set.

Let us now analyse the running time and probability of success for $\mathcal{B}$. $\mathcal{B}$ breaks the assumption on the injective one way function with probability at least $P[Q=1]\epsilon_{Q=1}/2^{n+1}$. Following is the argument:
Let τ_1 denote the transcript between $\mathcal{A}$ (along with $K_i' \forall i \in [n]$ and queried MSK_i's) and the hybrid challenger before the cipher-text is given out when $Q=1$. We say that τ_1 is good if the advantage conditioned on this transcript, $\epsilon_{Q=1,\tau_1} > \epsilon_{Q=1}/2$. Then, one can show that $Pr[\tau_1 \text{ is good }] > \epsilon_{Q=1}/2$. This follows from the fact that, $\epsilon_{Q=1} > \epsilon_{Q=1,\tau_1 \text{ is good}} Pr[\tau_1 \text{ is good}] + \epsilon_{Q=1,\tau_1 \text{ is not good}} Pr[\tau_1 \text{ is not good}]$. Assuming, $Pr[\tau_1 \text{ is not good}] < \epsilon_{Q=1}/2$ gives us a contradiction.
Now we denote by T the set of r_i's asked by $\mathcal{A}$ (either by querying MSK_i or querying (UID', i, U_i') for a token).

$$\mid Pr[\mathcal{A}(\mathcal{H}_{x,3}) = 1/Q = 1, \tau_1] - Pr[\mathcal{A}(\mathcal{H}_{x,4}) = 1/Q = 1, \tau_1] > \epsilon_{Q=1}/2 \mid$$

For all $t \subsetneq [n]$,

$$\Sigma_t \mid Pr[\mathcal{A}(\mathcal{H}_{x,3}) = 1 \cap T = t/Q = 1, \tau_1 - Pr[\mathcal{A}(\mathcal{H}_{x,4}) = 1 \cap T = t/Q = 1, \tau_1] \mid$$

$$> \epsilon_{Q=1}/2$$

Since number of such subsets is bounded by 2^n, there exists a set t^* such that

$$\mid Pr[\mathcal{A}(\mathcal{H}_{x,3}) = 1 \cap T = t^*/Q = 1, \tau_1 - Pr[\mathcal{A}(\mathcal{H}_{x,4}) = 1 \cap T = t^*/Q = 1, \tau_1] \mid$$

$$> \epsilon_{Q=1}/2^{n+1}$$

We note that $\mathcal{B}$ runs extractor on distinguisher L using apriori fixed set t. If t is guessed as t^* (which occurs with probability at least $1/2^n$)

$$\mid Pr[L(C_0, C_1, \mathcal{O}(C_0), z) = 1] - Pr[L(C_0, C_1, \mathcal{O}(C_1), z) = 1] \mid >$$

$$\mid Pr[\mathcal{A}(\mathcal{H}_{x,3} \cap T = t^*/Q = 1, \tau_1) = 1] - Pr[\mathcal{A}(\mathcal{H}_{x,4} \cap T = t^*/Q = 1, \tau_1) = 1] \mid$$

$$> \epsilon_{Q=1}/2^{n+1}$$

Hence, probability that $\mathcal{B}$ extracts a point $(x-1, r_1, .., r_n)$ is at least $Pr[Q=1] \cdot P[t = t^*] \cdot Pr[\tau_1 \text{ is good}] > Pr[Q=1]\epsilon_{Q=1}/2^{n+1}$. Assuming the advantage of L, $\epsilon_{Q=1}/2^{n+1} > 2^{-2nk-2k}$ (or else the observation trivially holds), we can run the extractor which runs in time $O(2^{2n} \cdot poly(k)/\epsilon^2_{Q=1})$ which is calculated from its distinguishing advantage of L.

From the two observations above if μ is set greater than $max\{(nk+2k)^{1/c_{owp2}}, (nk+2k)^{2/c_{owp2}}\}$, we prove the claim. □

Claim 5. *For any p.p.t. distinguisher D, $|\ Pr[D(\mathcal{H}_{x,4})=1] - Pr[D(\mathcal{H}_{x,5})=1]\ | \le O(n.2^{-nk-2k})$.*

Proof. This claim follows from the property that puncturable PRF's value is psuedo-random at punctured point given the punctured key (sub-exponential security of the puncturable PRF). The proof is similar to the proof of indistinguishability between $\mathcal{H}_{x,2}$ and $\mathcal{H}_{x,3}$. Hence, we omit the details to avoid repetition. □

Claim 6. *For any p.p.t. distinguisher D, $|\ Pr[D(\mathcal{H}_{x,5})=1] - Pr[D(\mathcal{H}_{x,6})=1]\ | \le O(n.2^{-nk-2k})$.*

Proof. This claim follows from the security of the obfuscation and correctness of the puncturable PRF. The proof is similar to the proof of indistinguishability between $\mathcal{H}_{x,1}$ and $\mathcal{H}_{x,2}$. Hence, we omit the details to avoid repetition. □

Claim 7. *For any p.p.t. distinguisher D, $|\ Pr[D(\mathcal{H}_{x,6})=1] - Pr[D(\mathcal{H}_{x,7})=1]\ | \le O(n.2^{-nk-2k})$.*

Proof. This claim follows from the security of the obfuscation and correctness of the puncturable PRF. The proof is similar to the proof of indistinguishability between $\mathcal{H}_x$ and $\mathcal{H}_{x,1}$. Hence, we omit the details to avoid repetition. □

From the claims above we get that for any PPT distinguisher D, $Pr[\mathcal{A}(\mathcal{H}_x)=1] - Pr[\mathcal{A}(\mathcal{H}_{x+1})=1] \le O(n.2^{-(n+2)k})$ □

From the lemmas above, we get that, $|\ Pr[\mathcal{A}(\mathcal{H}_0)=1] - Pr[\mathcal{A}(\mathcal{H}_{2^{(n+1)k}+2})=1]\ | \le \Sigma_x\ |\ Pr[\mathcal{A}(\mathcal{H}_x)=1] - Pr[\mathcal{A}(\mathcal{H}_{x+1})=1] \le 2^{(n+1)k} \cdot O(n \cdot 2^{-(n+2)k}) + negl(k) < O(n \cdot 2^{-k}) + negl(k)$. This concludes the proof.

□

4.2 Applications

In this section, we list a few applications of multi-authority functional encryption.

1. **Multi-Authority ABE:** [LW11] constructs a multi-authority attribute based encryption scheme where any authority can be set up independently and without any interaction with any other authority. Their work was the state of art on this topic and handles access policies in monotone $NC1$. It required a global parameter generation and a random oracle.
   Our work immediately results in a multi-authority attribute based encryption scheme: Encryptor encrypts the message along with a policy F that on input $(U_1, .., U_n)$ checks that it satisfies the required access structure $\mathbb{A}$ on attributes. If it does, outputs m otherwise $\perp$. Our construction does not require any global parameter generation and a random oracle. Additionally, it can handle policies in a bigger class of access structures i.e. monotone P.

2. **Functional Secret Sharing for monotone P:** [BBDK00] coined a primitive called as functional secret sharing for t-*out-of-* n access structures. Informally, a dealer can secret share a secret s among n players. Any subset $[B] \subset [n]$ of size greater than t can run a protocol $F_{B,f}$ using their shares s_i and some randomness r_i for $i \in B$ to learn the value $f(s)$. Security property is with respect so a subset $C \subset [n]$ of size less than t and is two fold: before evaluation security and after evaluation security. In before evaluation security, one requires that given the view of the C (shares of parties in C) an (computationally bounded/information theoretic) adversary should not be able to distinguish whether it is a sharing of s or $s' \neq s$. In after evaluation security, for a priory chosen f, given the view of C when a set B runs $F_{B,f}$ (shares and randomness of the parties in C, messages sent to and fro from the parties in C to parties in B while performing this computation), it should happen that any (computationally/bounded) adversary should not be able to distinguish whether the sharing corresponds to s or $s' \neq s$ for which $f(s) = f(s')$.
   [BBDK00] defines both non interactive and interactive functional secret sharing (depending if the protocol $F_{B,f}$ has one or more rounds). The more interesting case is of achieving non interactive functional secret sharing using only broadcast channel (assuming no private channel between parties). They present information theoretic constructions only for linear functions and t-*out-of* n threshold access structures. Using a multi-authority functional encryption scheme, one can construct computational functional secret sharing scheme for any access structure in monotone P that can allow evaluation for any polynomial sized circuit. For details, refer appendix B.

5 Constructing Delegatable Functional Encryption

Let HIBE be a sub-exponentially secure hierarchical identity based encryption. $i\mathcal{O}$ be a sub-exponentially secure indistinguishability obfuscation and $F = (\mathsf{Setup},$
$\mathsf{Puncture}, \mathsf{Eval})$ (slightly abusing notation, we refer $F(K,x)$ as evaluation of PRF at x even if the key is punctured) be a sub-exponentially secure puncturable psuedo-random function. We now describe our construction. Let $0 < c_{io}, c_{PRF}, c_{HIBE} < 1$ be the respective security constants. For syntax and security definition of HIBE refer section 2.4.

- $\mathsf{Setup}(1^k, n)$: On input the security parameter 1^k and n the setup algorithm runs $\mathsf{HIBE.Setup}(1^\lambda, n, k) \rightarrow (PK, SK)$. Here $nk + k$ denote the length of the identity and n denotes the maximum level of delegation. The algorithm sets $MSK = SK$ and $MPK = PK$ and outputs (MPK, MSK). λ is set greater than $(nk + 2k)^{1/c_{HIBE}}$.

- $\mathsf{Enc}(MPK, m)$ The algorithm samples a puncturable PRF key $F.\mathsf{Setup}(1^\mu) \rightarrow K$ and obfuscates the circuit described in Figure 6 with security parameter 1^γ (γ can be set more than $(nk+2k)^{1/c_{io}}$). μ can be set such that it maps to the space of randomness of HIBE encryption algorithm and greater than $(nk + 2k)^{1/c_{PRF}}$

- $\mathsf{KeyGen}(MSK, f)$: On input a function f of size at most k, the algorithm computes an HIBE key for identity vector $(-, ..., -, f)$ and outputs it as SK_f.

- $\mathsf{Del}(SK_f, f')$: The delegation algorithm takes as input an HIBE key for identity $f = (-, .., -, f_i, .., f_1)$ and f' and computes an HIBE key for $(-, .., -, f', f_i, .., f_1)$ and outputs it as $SK_{f'.f}$.

Hard-wired: MPK, K, m.
Input: $f \in \{0,1\}^{nk+k}$

- Parse $f = (f_{n+1}, .., f_1) \in (\{0,1\}^k)^{n+1}$ and compute $a = f_{n+1} \cdot ... \cdot f_1(m)$
- Output $\mathsf{HIBE.Enc}(MPK, f, a; F(K, f))$ (HIBE encryption using the identity vector $(f_{n+1}, .., f_1)$)

Figure 6: Encryption circuit

- $\mathsf{Dec}(SK_{f_i \cdot ... \cdot f_1}, CT)$: The decryption algorithm first computes $CT(id, .., id, f_i, .., f_1) = c$ (input to the circuit is the concatenation $id||, .., id||f_i||...||f_1$) to recover an HIBE cipher-text for identity vector $(id, id, .., id, f_i, .., f_1)$ and then uses $SK_{f_i, .., f_1}$ to decrypt it using HIBE decryption algorithm and outputs whatever it outputs. Here id is a k bit representation of the identity function.

5.1 Security Proof

Theorem 8. *Assuming existence of a sub-exponentially secure indistinguishability obfuscator and a sub-exponential hierarchical identity based encryption there exists a secure delegatable functional encryption scheme.*

Proof. We now prove that the scheme described above is secure. We begin by listing down indistinguishable hybrids where the first hybrid corresponds to the challenge game in which the adversary is given an encryption of m_0 and the final hybrid corresponds to the case in which the adversary is given an encryption of m_1.

$\mathbf{Hybrid_0}$: In this hybrid the challenger does the setup to compute HIBE keys (MPK, MSK) as the master setup keys for FE scheme. Any key query and delegation query is responded as done in the algorithms. Upon receiving the challenge message vector (m_0, m_1), the challenger encrypts m_0 using the encryption algorithm.

$\mathbf{Hybrid_{u \in [1, 2^{nk+k}+1]}}$: This hybrid is the same as the previous one except that the cipher-text is now an encryption of circuit in Figure 7.

$\mathbf{Hybrid_{2^{nk+k}+2}}$: In this hybrid the challenger does the setup to compute HIBE keys (MPK, MSK) as the master setup keys for FE scheme. Any key query and delegation query is responded as done in the algorithms. Upon receiving the challenge message vector (m_0, m_1), the challenger encrypts m_1 using the encryption algorithm.

Lemma 5. *For any ppt algorithm D, $\mid Pr[D(\mathbf{Hybrid_0}) = 1] - Pr[D(\mathbf{Hybrid_1}) = 1] \mid < O(2^{-nk-2k})$.*

Proof. This lemma follows from the security of indistinguishability obfuscation. Note that the

Hard-wired: MPK, K, m_0, m_1, u.
Input: $f \in \{0,1\}^{nk+k}$

- Parse $f = (f_{n+1}, .., f_1)$. If $f \geq u-1$ compute $a = f_{n+1} \cdot ... \cdot f_1(m_0)$ otherwise compute $a = f_{n+1} \cdot ... \cdot f_1(m_1)$.
- Output $\mathsf{HIBE.Enc}(MPK, f, a; F(K, f))$ (HIBE encryption using the identity vector $(f_{n+1}, .., f_1)$)

Figure 7: Encryption circuit*

hybrids differ only in the way the cipher-text is produced. In hybrid 0, the cipher-text is produced as an obfuscation of circuit described in Figure 6 and in hybrid 1 it is produced as an obfuscation of circuit in Figure 7 initialised with $(MPK, K, m_0, m_1, u = 1)$. These two circuits are equivalent hence by the sub-exponential security of indistinguishability obfuscation, the lemma follows. □

Lemma 6. *For any ppt algorithm D, $\mid Pr[D(\mathbf{Hybrid_{2^{nk+k}+1}}) = 1] - Pr[D(\mathbf{Hybrid_{2^{nk+k}+2}}) = 1] \mid < O(2^{-nk-2k})$.*

Proof. This lemma follows from the security of indistinguishability obfuscation. Note that the hybrids differ only in the way the cipher-text is produced. In hybrid $2^{nk+k}+1$, the cipher-text is produced as an obfuscation of circuit in Figure 7 with $u = 2^{nk+k}+1$ and in hybrid $2^{nk+k}+2$ it is produced as an obfuscation of circuit in Figure 6. These two circuits are equivalent hence by the sub-exponential security of indistinguishability obfuscation, the lemma follows. □

Lemma 7. *For any $u \in [1, 2^{nk+k}]$ ppt algorithm D, $\mid Pr[D(\mathbf{Hybrid_u}) = 1] - Pr[D(\mathbf{Hybrid_{u+1}}) = 1] \mid < O(2^{-nk-2k})$.*

Proof. Let Q denote a random variable that is defined as follows. $Q = 0$, when in $\mathbf{Hybrid_u}$ or $\mathbf{Hybrid_{u+1}}$, (m_0, m_1) released is such that for $u = (f_{n+1}, .., f_1)$, $f_{n+1} \cdot ... \cdot f_1(m_0) = f_{n+1} \cdot ... \cdot f_1(m_1)$. We say that $Q = 1$ otherwise. Now, we make following claims:

Claim 8. *Conditioned to $Q = 0$, for any ppt algorithm D, $\mid Pr[D(\mathbf{Hybrid_u}) = 1] - Pr[D(\mathbf{Hybrid_{u+1}}) = 1] \mid < O(2^{-nk-2k})$.*

Proof. This claim is straight forward and follows from the security of indistinguishability obfuscation. This is because when $Q = 0$, in hybrid u and $u+1$ the cipher-text is generated as an obfuscation of circuit in Figure 7 initialised with u and $u+1$ respectively. These two circuits are equivalent when $Q = 0$. □

Let us now assume $Q = 1$. We now define sub hybrids between $\mathbf{Hybrid_u}$ and $\mathbf{Hybrid_{u+1}}$ as follows:
$\mathbf{Hybrid_{u,1}}$: This hybrid is same as $\mathbf{Hybrid_u}$ except that generation of the cipher-text is done as follows. The challenger samples a puncturable PRF key K and sets $\alpha = F(K, u)$. The PRF key K is now punctured at u as K'. It parses $u = (f_{n+1}, .., f_1)$ and computes $a = f_{n+1} \cdot .. \cdot f_1(m_0)$. Then it evaluates $c = \mathsf{HIBE.Enc}(MPK, u, a; \alpha)$. Encryption is now given out as an obfuscation of circuit

described in Figure 8.

Hybrid$_{\mathbf{u,2}}$: This hybrid is same as **Hybrid$_{\mathbf{u,1}}$** except that generation of the cipher-text is

Hard-wired: MPK, K', m_0, m_1, u, c.
Input: $f \in \{0,1\}^{nk+k}$

- If $f = u$ output c. Otherwise,
- Parse $f = (f_{n+1}, .., f_1)$. If $f \geq u - 1$ compute $a = f_{n+1} \cdot ... \cdot f_1(m_0)$ otherwise compute $a = f_{n+1} \cdot ... \cdot f_1(m_1)$.
- Output $\mathsf{HIBE.Enc}(MPK, f, a; F(K', f))$ (HIBE encryption using the identity vector $(f_{n+1}, .., f_1)$)

Figure 8: Encryption circuit**

done as follows. The challenger samples a puncturable PRF key K and samples α randomly from the space of randomness for the HIBE encryption scheme. The PRF key K is now punctured at u as K'. It parses $u = (f_{n+1}, .., f_1)$ and computes $a = f_{n+1} \cdot .. \cdot f_1(m_0)$. Then it evaluates $c = \mathsf{HIBE.Enc}(MPK, u, a; \alpha)$. Encryption is given out as an obfuscation of circuit described in Figure 8.

Hybrid$_{\mathbf{u,3}}$: This hybrid is same as **Hybrid$_{\mathbf{u,2}}$** except that generation of the cipher-text is done as follows. The challenger samples a puncturable PRF key K and samples α randomly from the space of randomness for the HIBE encryption scheme. The PRF key K is now punctured at u as K'. It parses $u = (f_{n+1}, .., f_1)$ and computes $a = f_{n+1} \cdot .. \cdot f_1(m_1)$. Then it evaluates $c = \mathsf{HIBE.Enc}(MPK, u, a; \alpha)$. Encryption is given out as an obfuscation of circuit described in Figure 8.

Hybrid$_{\mathbf{u,4}}$: This hybrid is same as **Hybrid$_{\mathbf{u,3}}$** except that generation of the cipher-text is done as follows. The challenger samples a puncturable PRF key K and computes $\alpha = F(K, u)$. The PRF key K is now punctured at u as K'. It parses $u = (f_{n+1}, .., f_1)$ and computes $a = f_{n+1} \cdot .. \cdot f_1(m_1)$. Then it evaluates $c = \mathsf{HIBE.Enc}(MPK, u, a; \alpha)$. Encryption is given out as an obfuscation of circuit described in Figure 8.
Few claims are in order.

Claim 9. *Conditioned on $Q = 1$, for any ppt algorithm D, $| \, Pr[D(\mathbf{Hybrid_u}) = 1] - Pr[D(\mathbf{Hybrid_{u,1}}) = 1] \, | < O(2^{-nk-2k})$.*

Proof. This claim follows from the security of the indistinguishability obfuscation scheme and the correctness of the puncturable PRF. This is because when $Q = 1$, in hybrid u the cipher-text is generated as an obfuscation of circuit in Figure 7, initialised with PRF key K. In hybrid $u, 1$ cipher-text is generated as obfuscation of circuit in Figure 8 initialised with punctured PRF key K' and evaluation c at the punctured point. These two circuits are equivalent if the punctured PRF satisfies correctness. The claim then follows from the security of obfuscation scheme. □

Claim 10. *Conditioned on $Q = 1$, for any ppt algorithm D,* $\mid Pr[D(\mathbf{Hybrid_{u,1}}) = 1] - Pr[D(\mathbf{Hybrid_{u,2}}) = 1] \mid < O(2^{-nk-2k})$.

Proof. This claim follows from the sub-exponential security of the punctured PRF. This is because when $Q = 1$, in hybrid $u, 1$ the cipher-text is generated as an obfuscation of circuit described in Figure 8, initialised with punctured PRF key K' and $c = \mathsf{HIBE.Enc}(MPK, u, a; F(K, u))$. In hybrid $u, 2$ cipher-text is generated as obfuscation of circuit described in Figure 8 initialised with punctured PRF key K' and a random HIBE cipher-text c encrypting a. The claim then follows if the punctured PRF is psuedo-random at punctured points. □

Claim 11. *Conditioned on $Q = 1$, for any ppt algorithm D,* $\mid Pr[D(\mathbf{Hybrid_{u,2}}) = 1] - Pr[D(\mathbf{Hybrid_{u,3}}) = 1] \mid < O(2^{-nk-2k})$.

Proof. This claim follows from the sub-exponential security of the HIBE scheme. Let $u = (f_{n+1}, .., f_1)$ and $a_b = f_{n+1} \cdot ... \cdot f_1(m_b)$ for $b \in \{0, 1\}$. When $Q = 1$, in hybrid $u, 2$ the cipher-text is generated as an obfuscation of circuit in Figure 8, initialised with punctured PRF key K' and $c = \mathsf{HIBE.Enc}(MPK, u, a_0)$. In hybrid $u, 2$ cipher-text is generated as obfuscation of circuit described in Figure 8 initialised with punctured PRF key K' and a random HIBE cipher-text c encrypting a_1. In both hybrids, when $Q = 1$, the adversary cannot ask for HIBE decryption keys that allows decryption c. This is because he cannot ask for functional encryption keys for the function vector u as $Q = 1$. Hence due to sub-exponential security of HIBE the claim follows. □

Claim 12. *Conditioned on $Q = 1$, for any ppt algorithm D,* $\mid Pr[D(\mathbf{Hybrid_{u,3}}) = 1] - Pr[D(\mathbf{Hybrid_{u,4}}) = 1] \mid < O(2^{-nk-2k})$.

Proof. This claim follows from the sub-exponential security of the punctured PRF. This is because when $Q = 1$, in hybrid $u, 4$ the cipher-text is generated as an obfuscation of circuit in Figure 8, initialised with punctured PRF key K' and $c = \mathsf{HIBE.Enc}(MPK, u, a; F(K, u))$. In hybrid $u, 3$ cipher-text is generated as obfuscation of circuit in Figure 8 initialised with punctured PRF key K' and a random HIBE cipher-text c encrypting a. The claim then follows if the punctured PRF is psuedo-random at punctured points. □

Claim 13. *Conditioned on $Q = 1$, for any ppt algorithm D,* $\mid Pr[D(\mathbf{Hybrid_{u,4}}) = 1] - Pr[D(\mathbf{Hybrid_{u+1}}) = 1] \mid < O(2^{-nk-2k})$.

Proof. This claim follows from the security of the indistinguishability obfuscation scheme and the correctness of the puncturable PRF. This is because when $Q = 1$, in hybrid $u + 1$ the cipher-text is generated as an obfuscation of circuit in Figure 7, initialised with PRF key K. In hybrid $u, 4$ cipher-text is generated as obfuscation of circuit described in Figure 8 initialised with punctured PRF key K' and evaluation c at the punctured point. These two circuits are equivalent if the punctured PRF satisfies correctness. The claim then follows from the security of obfuscation scheme. □

Combining all the claims we get that for any ppt algorithm D, $\mid Pr[D(\mathbf{Hybrid_{u}}) = 1] - Pr[D(\mathbf{Hybrid_{u+1}}) = 1] \mid < O(2^{-nk-2k})$. □

Summing up we have that

$$| \; Pr[D(\mathbf{Hybrid_0}) = 1] - Pr[D(\mathbf{Hybrid_{2^{nk+k}+2}}) = 1] \; |$$

$$\leq 2negl(k) + \Sigma_u \; | \; Pr[D(\mathbf{Hybrid_u}) = 1] - Pr[D(\mathbf{Hybrid_{u+1}}) = 1] \; |$$

This implies that,

$$| \; Pr[D(\mathbf{Hybrid_0}) = 1] - Pr[D(\mathbf{Hybrid_{2^{nk+k}+2}}) = 1] \; | \leq 2negl(k) + 2^{nk+k}O(2^{-nk-2k})$$

□

6 Acknowledgement

First three authors would like to thank Shweta Agarwal and Raghav Bhaskar on some insightful discussions about the problem.

References

[AGVW13] Shweta Agrawal, Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption: New perspectives and lower bounds. 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 500–518. Springer, 2013. 12

[BBDK00] Amos Beimel, Mike Burmester, Yvo Desmedt, and Eyal Kushilevitz. Computing functions of a shared secret. *SIAM J. Discrete Math.*, 13(3):324–345, 2000. 4, 24, 34

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Cramer [Cra05], pages 440–456. 7

[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. 4, 8, 9

[BGI$^+$01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2001. 4, 12

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer, 2011. 1, 3

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. *IACR Cryptology ePrint Archive*, 2013:352, 2013. 7

[CC09] Melissa Chase and Sherman S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9-13, 2009*, pages 121–130. ACM, 2009. 3, 4

[Cha07] Melissa Chase. Multi-authority attribute based encryption. In Vadhan [Vad07], pages 515–534. 3, 4

[Cra05] Ronald Cramer, editor. *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*. Springer, 2005. 29, 31

[GGG$^+$14] Shafi Goldwasser, S. Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 578–602. Springer, 2014. 1, 3, 12

[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. 1, 3, 4, 6

[GGHW14] Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 518–535. Springer, 2014. 4

[GGKT05] Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM J. Comput.*, 35(1):217–246, 2005. 5

[GH08] Craig Gentry and Shai Halevi. Hierarchical identity based encryption with polynomially many levels. *IACR Cryptology ePrint Archive*, 2008:383, 2008. 7

[GKP$^+$13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 555–564. ACM, 2013. 1, 3, 12, 36

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006*, pages 89–98. ACM, 2006. 1

[GS02] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In Yuliang Zheng, editor, *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002. 7

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. 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 162–179. Springer, 2012. 36

[Hol06] Thomas Holenstein. Pseudorandom generators from one-way functions: A simple construction for any hardness. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2006. 5

[LW11] Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 568–588. Springer, 2011. 3, 4, 23

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Cramer [Cra05], pages 457–473. 1

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 475–484. ACM, 2014. 4, 7

[Vad07] Salil P. Vadhan, editor. *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, volume 4392 of *Lecture Notes in Computer Science*. Springer, 2007. 30, 31

[VNS$^+$03] V. Vinod, Arvind Narayanan, K. Srinathan, C. Pandu Rangan, and Kwangjo Kim. On the power of computational secret sharing. In Thomas Johansson and Subhamoy Maitra, editors, *Progress in Cryptology - INDOCRYPT 2003, 4th International Conference on Cryptology in India, New Delhi, India, December 8-10, 2003, Proceedings*, volume 2904 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 2003. 36

[Wee05] Hoeteck Wee. On obfuscating point functions. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 523–532. ACM, 2005. 5

[Wee07] Hoeteck Wee. One-way permutations, interactive hashing and statistically hiding commitments. In Vadhan [Vad07], pages 419–433. 5

A Completing proofs of $(1, \delta)$ weak extractability obfuscator

Theorem 9. *Assuming sub-exponentially secure indistinguishability obfuscation, there exists $(1, \delta)$ weak obfuscator for P/poly for any $\delta > 2^{-k}$, where k is the size of the circuit.*

Proof. We now construct a $(1, 2^{-k})$ input weak extractability obfuscator from sub-exponentially secure indistinguishability obfuscation. Following algorithm describes the obfuscation procedure.

$we\mathcal{O}(1^k, C)$: The procedure outputs $C' \leftarrow i\mathcal{O}(1^{k^{1/\alpha}}, C)$. Here, $\alpha > 0$ is a constant chosen such that any polynomial time adversary against indistinguishability obfuscation has security gap upper bounded by $2^{-k}/4$.

Correctness and polynomial slowdown follows from the properties of the indistinguishability obfuscator. Now we formally describe the extractor E. Let us assume that the circuits (C_0, C_1) take as n bit inputs and there is an adversary $\mathcal{A}$ for which $1 \geq \epsilon(k) > 2^{-k}$. Here, $2^{-k}/4$ is the indistinguishability gap corresponding to $we\mathcal{O}$ for equivalent circuits of size k and $n = O(k)$ inputs.

We construct an extractor that runs in $O(n.(2t_{\mathcal{A}} + poly(k)).k.1/\epsilon^2)$ time and extracts the differing input with overwhelming probability. The algorithm takes as input two circuits, auxillary input to the adversary z and distinguishing gap ϵ and outputs the differing input d. For simplicity, lets assume that:

$$Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_0), C_0, C_1, z) = 1] < Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_1), C_0, C_1, z) = 1]$$

Otherwise, the extractor can be run twice using the different sign for this advantage.

Our extractor is described in figure 9. For now, assume that there is a single differing input d. Intuitively, our extractor predicts the differing input bit by bit. For an index $i \in [n]$, E defines a circuit C_{mid} that on input x computes $C_{x_i}(x)$, where x_i is the i^{th} bit of x. One can check that for $b \in \{0, 1\}$, when $d_i = b$, then C_b and C_{mid} are equivalent. Our extractor, knows that the distinguishing advantage of $\mathcal{A}$ satisfies that

$$Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_0), C_0, C_1, z) = 1] + \epsilon < Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_1), C_0, C_1, z) = 1]$$

It estimates the advantage of $\mathcal{A}$ in distinguishing C_0 with C_{mid} and similarly the advantage of $\mathcal{A}$ in distinguishing C_{mid} with C_1 (by repeating experiments many times). Because the advantage of adversary in distinguishing obfuscation of two equivalent circuits is smaller than $2^{-k}/4$ (due to security of indistinguishability obfuscation), it compares the the two advantage and if it finds that advantage of distinguishing obfuscation of C_b from that of C_{mid} is less than the advantage advantage of distinguishing obfuscation of C_{1-b} from that of C_{mid}, it outputs $d_i = b$.

Lemma 8. *Suppose there exists circuits $C_0, C_1 \in \mathcal{C}_k$ (set of circuits with size k) with $n = O(k)$ input bits and disagreeing on at most single input, and auxiliary input z for which*

$$\mid Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_0), C_0, C_1, z) = 1] - Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_1), C_0, C_1, z) = 1] \mid \geq \epsilon(k)$$

```
Input: $C_0, C_1, z, \epsilon$

1. initialise $d = 0$
2. For $i = 1$ to $n$
   (a) Initialise $L_i = 0, R_i = 0$
   (b) For $j \in [t = k/\epsilon^2]$
       i. Using Figure 10 compute, $L_{i,j} = \mathcal{A}(we\mathcal{O}(C_{Mid}^{C_0,C_1,i}), C_0, C_1, z) - \mathcal{A}(we\mathcal{O}(C_0), C_0, C_1, z)$
       ii. Using Figure 10 compute, $R_{i,j} = \mathcal{A}(we\mathcal{O}(C_1), C_0, C_1, z) - \mathcal{A}(we\mathcal{O}(C_{Mid}^{C_0,C_1,i}), C_0, C_1, z)$
       iii. $L_i = L_i + L_{i,j}$
       iv. $R_i = R_i + R_{i,j}$
   (c) If $L_i < R_i$ set $d_i = 0$ else set $d_i = 1$
3. Output $d$ if $C_0(d) \neq C_1(d)$, $\perp$ otherwise.
```

Figure 9: Extractor E

```
Constants: $C_0, C_1, i \in [n]$.
Input: $x \in \{0,1\}^n$

  ∘ If $x_i = 0$ output $C_0(x)$.
  ∘ If $x_i = 1$ output $C_1(x)$.
```

Figure 10: Program $C_{Mid}^{C_0,C_1,i}$

where $2^{-k} < \epsilon \leq 1$ Then the algorithm E on input $(1^k, C_0, C_1, z, \epsilon)$ terminates within expected time $O(n.(2t_{\mathcal{A}}+poly(k)).1/\epsilon^2)$ and it holds that $Pr[v \leftarrow E(1^k, C_0, C_1, z, \epsilon) : C_0(v) \neq C_1(v)] \geq 1-negl(k)$

Proof. Lemma follows from the two claims below..

Claim 14. *E runs in expected $O(n.(2t_{\mathcal{A}} + poly(k)).k.1/\epsilon^2)$ time where $t_{\mathcal{A}}$ is the expected running time of the adversary $\mathcal{A}$.*

Proof. Let us analyse expected running time of the algorithm E. The algorithm predicts bit by bit the differing input d. For each bit $i \in [n]$, the extractor repeats inner loop $t = k/\epsilon^2$ times. Each execution of the inner loop has an expected termination time of $2t_{\mathcal{A}}+poly(k)$ where $poly(k)$ represents time to compute challenge obfuscation and other computation steps. Hence, time taken to run E is $O(n.(2t_{\mathcal{A}} + poly(k)).1/\epsilon^2)$

□

Claim 15. *E succeeds with an overwhelming probability.*

Proof. Let us analyse the success probability of the extractor. If the circuits are equivalent then differing advantage of the adversary between the obfuscations of the circuits cannot be greater than or equal to ϵ, due to the sub-exponential security of indistinguishability obfuscation. In this case the extractor always outputs $\perp$.

Assuming that there is a differing input d, then let U_i denote the event that d_i is incorrectly predicted by the extractor in the i^{th} loop. We need to show that $Pr[\cup_{i\in[n]}U_i] < negl(k)$. We claim this by bounding $Pr[U_i]$ for any $i \in [n]$ and applying union bound.

Let us calculate the probability that d_i is incorrectly calculated given that $d_i = 0$ (In this case $C_{Mid}^{C_0,C_1,i}$ and C_0 are functionally equivalent). This is probability is equal to $Pr[R_i \leq L_i/d_i = 0]$, where $R_i - L_i$ is calculated by the program E during loop i (i.e. when the program predicts d_i). $Z_i = R_i - L_i = \Sigma_{j\in[t]}R_{i,j} - L_{i,j}$ is a random variable that sums the intermediate summands $(R_{i,j} - L_{i,j})$ during the j^{th} execution of the inner loop, while predicting d_i. Define $Z_{i,j} = R_{i,j} - L_{i,j}$. We need to bound $Pr[Z_i \leq 0]$.

We now use the following chernoff bound for achieving this. Given $X = X_1 + .. + X_N$ where each X_i $\forall i \in [N]$ are independent random variables in $[0, 1]$ and $\mu = \mathbb{E}(X)$. Then, for any $\alpha \geq 0$ we have that, $Pr[X \leq (1-\alpha)\mu] \leq e^{-\alpha^2\mu/2}$.

In order to apply this chernoff bound, we define $Z'_{i,j} = (Z_{i,j} + 2)/4$ and $Z'_i = \Sigma_j(Z_{i,j} + 2)/4$ and upper bound $Pr[Z_i < 0] = Pr[Z'_i < t/2]$. Let $\mathbb{E}(Z_i) = p.t$ for some $p > 0$ (by assumption that $Pr[\tilde{C} \leftarrow we\mathcal{O}(1^k, C_0) : \mathcal{A}(1^k, \tilde{C}, C_0, C_1, z) = 1] < Pr[\tilde{C} \leftarrow we\mathcal{O}(1^k, C_1) : \mathcal{A}(1^k, \tilde{C}, C_0, C_1, z) = 1]$). On some computation we get that $Pr[Z_i < 0] = e^{-p^2t/8(p+2)}$. Now we compute p.

For any $j \in [t]$, we have that $p = \mathbb{E}(R_{i,j}/d_i = 0) - \mathbb{E}(L_{i,j}/d_i = 0)$. It is easy to see that $\mathbb{E}(R_{i,j}/d_i = 0)$ is the advantage of the adversary in distinguishing the obfuscation of $C_{Mid}^{C_0,C_1,i}$ when $d_i = 0$ from that of C_1 and similarly $\mathbb{E}(L_{i,j}/d_i = 0)$ is the advantage of the adversary in distinguishing the obfuscation of $C_{Mid}^{C_0,C_1,i}$ from that of C_0 when $d_i = 0$. Note that, $\mathbb{E}(R_{i,j}/d_i = 0) + \mathbb{E}(L_{i,j}/d_i = 0) > \epsilon$ (due to the assumption that, $Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_0), C_0, C_1, z) = 1] + \epsilon < Pr[\mathcal{A}(1^k, we\mathcal{O}(1^k, C_1), C_0, C_1, z) = 1]$).

Since $d_i = 0$, $\mid \mathbb{E}(L_{i,j}/d_i = 0) \mid < 2^{-k}/4$ due to subexponential security of the indistinguishability obfuscation. From these observations, we have that $p \geq \epsilon - 2.2^{-k}/4$. Since $\epsilon > 2^{-k}$, $p > \epsilon/2$ hence, when t is set as k/ϵ^2, $Pr[Z_i < 0] \leq e^{-k/16}$.

Similarly when $d_i = 1$, we can upper bound probability of incorrect prediction of d_i as $Pr[R_i > L_i] \leq e^{-k/16}$, when $t = k/\epsilon^2$. This proves $\forall i \in [n], Pr(U_i) < negl(k)$. □

□

□

B Constructing Functional Secret Sharing

In this section, we describe the connection between MAFE and functional secret sharing as proposed in [BBDK00]. We now describe formally syntax and properties,

A secret sharing scheme for a family of functions $\mathcal{F}$ is a distribution scheme which satisfies the following two conditions:

Evaluation. For any set $B \subset U$ of size $B \in \mathbb{A}$ ($\mathbb{A}$ is in the set of allowed access structure e.g. monotone P e.t.c.) and any $f \in \mathcal{F}$ the parties in B can evaluate $f(s)$. That is, there is a protocol $F_{B,f}$ which given the shares of B as inputs will always output the correct value of $f(s)$. The scheme is called non interactive if $F_{B,f}$ is one round, and interactive otherwise. Depending on the communication model, a broadcast channel or private channels are used.

Security. Let X be a random variable on the set of secrets S and C be a set $C \notin \mathbb{A}$. For any two secrets $s, s^{'} \in S$, $\{(VIEW_C|X = s) =< s_i >_{i\in C}\} \approx_c \{(VIEW_C|X = s^{'}) =< s_i^{'} >_{i\in C}\}$.
After evaluation. For any $f \in \mathcal{F}$ and any $B \in \mathbb{A}$, any secrets $s, s^{'}$ with $f(s) = f(s^{'})$, any shares $< s_i >_C$, any inputs $< r_i >_{B\cap C}$, and any messages $M(< s_i >_B, < r_i >_B)$ from the computation of $f(s)$ by B received by parties in C.

$$\{(VIEW_C|X = s) =< s_i >_C, < r_i >_{B\cap C}, M(< s_i >_B, < r_i >_B)\} \approx_c$$
$$\{(VIEW_C|X = s^{'}) =< s_i >_C, < r_i >_{B\cap C}, M(< s_i >_B, < r_i >_B)\}$$

Theorem 10. $\mathsf{IND}-1$ *secure* MAFE *secure against query of one token per authority implies the existence of a computational functional secret sharing scheme.*

Proof. Now we describe a non interactive functional secret sharing scheme in a broadcast channel. Let's discuss the algorithms.

$\mathsf{Share(n, \mathbb{A}, s)}$: To secret share s among n parties according to access policy $\mathbb{A}$ dealer does the following.

1. For $j \in [n]$ run MAFE.Authority setup$(1^k) \rightarrow (MPK_j, MSK_j)$.
2. Computes $CT \leftarrow$ MAFE.Encrypt$(MPK_1, .., MPK_n, U_{\mathbb{A}}, s)$. Here $U_{\mathbb{A}}$, on input $(x_1, .., x_n)$ checks that there is a subset $I \subset [n]$ and $x_i = f \forall i \in I$ and $\perp$ otherwise. It also checks that I satisfies $\mathbb{A}$. If the checks passes, it outputs $f(s)$.
3. Share of player i is $MPK_1, .., MPK_n, CT, MSK_i$.

$\mathsf{Recon(MPK_1, .., MPK_n, CT, MSK_{j\in B\in\mathbb{A}})}$. For reconstruction, each party j computes

$$K_j \leftarrow \mathsf{MAFE.KeyGen}(MSK_j, 0, id)$$

(where id is the identity function) and broadcasts it to all parties in B. Each party then reconstructs secret as $s \leftarrow$ MAFE.Decrypt$(\{K_{j,id}\}_{j\in B}, CT)$.

Evaluation works as follows, Let $B \in \mathbb{A}$ be a set of players participating in functional evaluation. Each party $j \in B$ computes $K_{j,f} \leftarrow \mathsf{MAFE.KeyGen}(MSK_j, 0$
$, f)$ and broadcasts this to all players in B. Each player then recovers $f(s) \leftarrow \mathsf{MAFE.Decrypt}(\{K_{j,f}\}_{j\in B}, CT)$.

Let us discuss now security of the scheme.
Before evaluation. For any $C \notin \mathbb{A}$, view of C consists of $(MPK_1, .., MPK_n,$
$MSK_{j\in C}, CT = \mathsf{MAFE.Encrypt}(MPK_1, .., MPK_n, U_{\mathbb{A}}, s))$. Since, $C \notin \mathbb{A}$, by the security of MAFE,

$$\{MPK_1, .., MPK_n, MSK_{j\in C}, CT = \mathsf{MAFE.Encrypt}(MPK_1, .., MPK_n, U_{\mathbb{A}}, s)\} \approx_c$$

$$\{MPK_1, .., MPK_n, MSK_{j\in C}, CT^{'} = \mathsf{MAFE.Encrypt}(MPK_1, .., MPK_n, U_{\mathbb{A}}, s^{'})\}$$

for all $s^{'}$

After evaluation. For any set $B \in \mathbb{A}$ and $C \notin \mathbb{A}$, $VIEW_C$ in the evaluation of $f \in \mathcal{F}$ contains $(MPK_{j\in[n]}, CT = \mathsf{MAFE.Encrypt}(MPK_1, .., MPK_n, U_{\mathbb{A}}, s),$
$MSK_{j\in B\cap C}, K_{j\in B,f})$. By security of multi- authority functional encryption scheme, this view is computationally indistinguishable from when $s^{'}$ was shared where $f(s) = f(s^{'})$ (since f, s, s' are known ahead of time). $\square$

We stress that we in fact need a multi-authority functional encryption scheme secure against single token query per authority and for policy functions of the form $U_{\mathbb{A}}(\cdot)$ (described in the previous theorem). Such schemes can be constructed without using obfuscation from homomorphic encryption, single key secure functional encryption and one way functions. We describe the construction but omit the security proof here.

Notation. Let κ be the security parameter. Let FHE denote a compact fully homomorphic encryption scheme. We are going to construct MAFE for circuits output $m(\kappa)$ bits. Let FHE ciphertext corresponding to these many bits be bounded by $len = len(\kappa)$. We will use a computational secret sharing scheme for monotone poly sized boolean circuits [VNS+03]. We denote this scheme by SS. Also, we annotate by FE a single authority functional encryption scheme (IND) secure against bounded non adaptive function key queries. Such schemes are known to exist under classical assumptions [GVW12] [GKP+13]. Gb denotes Yao's circuit garbling scheme. For a given access structure $\mathbb{A}$, $\Pi(s, i)$ denotes i^{th} secret share of s according to secret sharing scheme used in the scheme. To distinguish two independently generated shares of same secret we use subscripts e.g. $\Pi_0(s, i), \Pi_1(s, i)$ denotes i^{th} shares of secret s in sharing "0" and sharing "1". Whenever a secret is shared only once this subscript is dropped. Also assume that for authorities are named in 1 through n and access structures $\mathbb{A}$ have inputs named in $[n]$.

Authority Setup(1^κ): Each authority i runs setup of single key functional encryption scheme len times, $\{\mathsf{FE.Setup}(1^\kappa) \rightarrow (PK_{i,j}, SK_{i,j})\}_{j\in[1,len]}$. $MPK_i = \{PK_{i,j}\}\forall j$ and $MSK_i = \{SK_{i,j}\}\forall j \in [len]$.

Encrypt$(\{MPK_i\}_{i\in[n]}), U_{\mathbb{A}}, m)$:

1. Run $\mathsf{FHE.KeyGen}(1^\kappa) \rightarrow (HPK, HSK)$. Compute $\widehat{m} \leftarrow \mathsf{FHE.Enc}(HPK, m)$.

2. Compute a garbled circuit for decryption circuit for FHE for len inputs.
   $\mathsf{Gb.Garble}(1^\kappa, \mathsf{FHE.Dec}_{HSK}(\cdot)) \rightarrow (\Gamma, L_{p,q})_{p\in[0,1],q\in[1,len]}$.

3. For each label $L_{p,q}$, secret share $L_{p,q}$ according to $\mathbb{A}$ using secret sharing scheme SS.

4. Cipher text consists of HPK, Γ and ciphertexts
   $\{CT_{i,j} = \mathsf{FE.Encrypt}(PK_{i,j}, HPK, \widehat{m}, \Pi(L_{0,j}, i), \Pi(L_{1,j}, i))\}_{\forall j\in[1,len],i\in[n]}$.

$\mathsf{KeyGen}(f, MSK_i)$. Let $G_{f,j}$ be the following circuit. G_f on input $(HPK, \widehat{m}, x, y)$:

1. Compute $e = \mathsf{FHE.Eval}(HPK, f, \widehat{m})$
2. If j^{th} bit of e, $e_j = 1$ output y else output x.

Output $K_{i,f} = \{\mathsf{FE.KeyGen}(SK_{i,j}, G_{f,j})\}_{\forall j\in[1,len]}$.

$\mathsf{Decrypt}(CT, \{K_{i,f}\})$. Decryptor computes $Q_{i,j} = \mathsf{FE.Decrypt}((K_{i,f})_j, CT_{i,j}) \forall i \in [n], j \in [len]$. For each $j \in [len]$ decryptor reconstructs $L_j = \mathsf{SS.RECON}(\{Q_{i,j}\}_{i\in[n]})$. Then, Decryptor computes output as $f(m) \leftarrow \mathsf{Gb.Eval}(\Gamma, L_1, .., L_{len})$