Attribute-Based Fully Homomorphic Encryption with a Bounded Number of Inputs

Michael Clear

Georgetown University and George Mason University
E-mail: clearm@scss.tcd.ie

Ciarán Mc Goldrick

School of Computer Science and Statistics,Trinity College Dublin, Ireland
E-mail: Ciaran.McGoldrick@scss.tcd.ie

Abstract: The only known way to achieve Attribute-based Fully Homomorphic Encryption (ABFHE) is through indistinguishability obfuscation. The best we can do at the moment without obfuscation is Attribute-Based Leveled FHE which allows circuits of an a priori bounded depth to be evaluated. This has been achieved from the Learning with Errors (LWE) assumption. However we know of no other way without obfuscation of constructing a scheme that can evaluate circuits of unbounded depth. In this paper, we present an ABFHE scheme that can evaluate circuits of unbounded depth but with one limitation: there is a bound N on the number of inputs that can be used in a circuit evaluation. The bound N could be thought of as a bound on the number of independent senders. Our scheme allows N to be exponentially large so we can set the parameters so that there is no limitation on the number of inputs in practice. Our construction relies on multi-key FHE and leveled ABFHE, both of which have been realized from LWE, and therefore we obtain a concrete scheme that is secure under LWE.

Keywords: Attribute-Based Encryption; Fully Homomorphic Encryption;

Biographical notes: Michael Clear is a postdoctoral researcher at Georgetown University and George Mason University. He received his PhD from Trinity College Dublin in 2016.
Ciarán Mc Goldrick is an Assistant Professor in the School of Computer Science & Statistics in Trinity College Dublin Ireland where he has been a faculty member since 1999. His research interests include constrained wireless networking, computer systems and security, renewable energy technologies and engineering education. He is a Senior Member of both the IEEE and ACM, and has been Visiting Professor in UCLA in 2015 and 2016.

1 Introduction

Attribute Based Encryption (ABE) is a cryptographic primitive that realizes the notion of cryptographic access control. ABE owes its roots to a simpler primitive called Identity Based Encryption (IBE), proposed in 1985 by Shamir (Shamir (1985)) and first realized in 2001 by Boneh and Franklin (Boneh and Franklin (2001)) and Cocks (Cocks (2001)). IBE is centered around the notion that a user's public key can be efficiently derived from an identity string and a system-wide *public parameters*.

The identity string may be a person's email address, IP address or staff number, depending on the application. The public parameters along with a secret trapdoor (master secret key) are generated by a trusted third party referred to as the Trusted Authority (TA). The primary purpose of the TA is to issue a secret key to a user that corresponds to her identity string (we abbreviate this to *identity*) over a secure channel. The means by which the users authenticate to the TA or establish a secure channel are outside the scope of IBE. The TA uses the master secret key to derive the secret keys for identities. It is assumed that all parties have a priori access to the public parameters. For instance, the public parameters may be hard-coded in the software used by the participants, or made available on a public website.

ABE was proposed in 2005 by Sahai and Waters (Sahai and Waters (2005)). ABE can be viewed as a generalization of IBE. In ABE, the TA generates secret keys instead for *access policies* (an access policy prescribes the types of data a user is authorized to access). An encryptor Alice can use the public parameters to encrypt data, and embed within the ciphertext a *descriptor* of her choice that suitably describes her data. The descriptor is referred to as an *attribute*. We caution the reader that although the term *attribute* is used here in its singular form, it may in fact incorporate a *collection* of descriptive elements (which we call "subattributes"). To illustrate this, an example of an attribute is {"CS", "CRYPTO"}; it consists of the subattributes "CS" and "CRYPTO". Let us assume

that this is the attribute chosen by Alice. Suppose the TA has issued a user Bob a secret key for his access policy. Keeping with the above example, suppose his access policy "accepts" an attribute if it contains **both** the subattributes "CS" and "CRYPTO". It follows that Alice's chosen attribute satisfies Bob's access policy. As such, Bob can use his secret key to decrypt Alice's ciphertext. Notice that IBE is a special case of ABE. One way of looking at an IBE scheme is that each attribute corresponds to a unique identity string such as an email address or phone number. In IBE, there is a one-to-one mapping between attributes and access policies, so Alice is given a secret key for a policy that is singularly satisfied by her identity string.

We will return to identity/attribute-based encryption momentarily. First we need to introduce the notion of fully homomorphic encryption (FHE). An FHE scheme can evaluate all polynomial-time computable functions. Strikingly, it achieves this without expanding the ciphertext size. For many applications, we need only the capability to evaluate circuits of some limited depth. Leveled FHE is a relaxation of FHE that can evaluate circuits of depth at most some positive integer d.

FHE was first constructed in 2009 in a breakthrough work by Gentry (Gentry (2009)). Most work on FHE has focused on the public-key setting but there has been some work in recent years in achieving FHE in the identity/attribute-based setting. Gentry, Sahai and Waters (Sahai and Waters (2013)) constructed the first leveled Identity-Based Fully Homomorphic Encryption (IBFHE) scheme and the first leveled Attribute-Based Fully Homomorphic Encryption (ABFHE) scheme from the Learning with Errors (LWE) problem. Clear and Mc Goldrick (Clear and Mc Goldrick (2015)) extended the former to achieve "multi-identity" leveled IBFHE where evaluation can be performed on ciphertexts associated with different identities. These schemes are leveled; that is, they are not "pure" FHE schemes insofar as all circuits cannot be evaluated, only those of limited depth.

The only known way to achieve "pure" ABFHE (i.e. where all circuits can be evaluated) is through indistinguishability obfuscation (Garg et al. (2013)), namely the construction in (Clear and Mc Goldrick (2014)). The best we can do at the moment without obfuscation is Attribute-Based Leveled FHE which allows circuits of an a priori bounded depth to be evaluated. However we know of no other way in the identity/attribute-based setting (without obfuscation) of constructing a scheme that can evaluate circuits of unbounded depth. This has particular significance in the attribute-based setting because the public parameters are generated once and the chosen bound on the circuit may not cater for all applications where deeper circuits are needed, and it would be unwieldy to generate new public parameters.

The technique of bootstrapping is currently the only known way to evaluate circuits of unbounded depth. Obtaining ABFHE for circuits of unbounded depth has been impeded by the fact that employing bootstrapping in the attribute-based setting (non-interactively) is particularly challenging since bootstrapping requires encryptions of the secret key bits to be available as part of the public key. Even in the identity-based setting this is a difficult challenge because one has to non-interactively derive encryptions of the secret key bits for any identity string from the public parameters alone. The only known way of doing bootstrapping is via indistinguishability obfuscation (Clear and Mc Goldrick (2014)). Without obfuscation, we have not been able to achieve "pure" ABFHE.

In this work we construct an almost "pure" ABFHE with one catch, namely, there is a pre-established bound N on the number of inputs to the circuits that can be evaluated where each input is a bitstring of arbitrary size. Another way of looking at it is that there is a limit on the number of independent senders who can contribute inputs to the circuit. Our construction allows N to be exponentially large because the parameter sizes grow logarithmically in N so it can be set large enough to accommodate most reasonable applications. For example by setting $N = 2^{32}$, the parameter sizes do not grow much and over 4 billion inputs can be accommodated, which is more than one would expect in reasonable applications, since each input (contributed by an independent sender) can be of arbitrary size.

1.1 Our Construction

Our construction relies on multi-key FHE and leveled ABFHE. Our use of multi-key FHE is similar to that of (Clear and Mc Goldrick (2013)) which uses it to a achieve a non-compact form of ABFHE. If we have a leveled ABFHE with a class of access policies $\mathbb{F}$, then we get a ("pure") ABFHE for the class of policies $\mathbb{F}$ with a bound N on the number of inputs. The main idea behind our approach is that an encryptor generates a key-pair $(\mathsf{pk}, \mathsf{sk})$ for the multi-key FHE scheme and it encrypts the secret key sk with the leveled ABFHE scheme to obtain ciphertext ψ. Then the encryptor encrypts every bit of plaintext (say w bits) with the multi-key FHE scheme using pk to obtain ciphertext $c_1, \ldots, c_w$. It sends the ciphertext $\mathsf{CT} := (\psi, c_1, \ldots, c_w)$. The evaluator evaluates the circuit on the multi-key FHE ciphertexts and obtains an encrypted result c'. Then it evaluates with the leveled ABFHE scheme the decryption circuit of the multi-key FHE scheme on c' together with the encrypted secret keys (the ψ ciphertexts). We obtain a ciphertext in the leveled ABFHE scheme that encrypts the result of the computation (i.e. what c' encrypts). The size of the resulting ciphertext is independent of N and the size of the circuit. By using the multi-key FHE scheme of Clear and Mc Goldrick (Clear and Mc Goldrick (2015)), we only need the leveled ABFHE scheme to have $L = O(\log N)$ levels where N is the bound on the number of inputs.

We say a scheme is *single-attribute* if it only allows homomorphic evaluation on ciphertexts with the same attribute. Otherwise, if it allows evaluation on

ciphertexts with different attributes, we refer to the scheme as *multi-attribute*. Whether our construction is single-attribute or multi-attribute depends on the underlying leveled ABFHE scheme that is used. Single-attribute leveled ABFHE has been achieved from LWE as has multi-identity leveled IBFHE. However multi-attribute leveled ABFHE is an open problem. Hence we cannot obtain "pure" multi-attribute ABFHE with a bounded number of inputs because there are no multi-attribute leveled schemes. The closest we have is multi-identity leveled IBFHE. The only known way of achieving "pure" multi-attribute ABFHE is via indistinguishability obfuscation.

1.2 Organization

This paper is organized as follows. In Section 2, we introduce definitions that we use throughout the paper including a definition of Attribute-Based Homomorphic Encryption. In Section 3, we provide security definitions and introduce a new security notion which we call EVAL-SIM security. In Section 4, we present our construction of ABFHE with a bounded number of inputs. We prove security of the construction in Section 5. We review our main result and its corollaries in Section 6. In Section 7 we prove the sel-EVAL-SIM security of the multi-attribute ABFHE scheme from (Clear and Mc Goldrick (2014)). Finally in Section 8 we present performance results for the multi-key FHE scheme of López-Alt, Tromer and Vaikuntanathan (López-Alt et al. (2012)).

2 Definitions

Let us briefly recall the definition of key-policy attribute based encryption (KP-ABE). A trusted authority (TA) generates public parameters and a master secret key. It uses its master secret key to generate secret keys for *access policies*. Alice encrypts her data, using the public parameters, under an "attribute" of her choice in some designated set of "attributes". An "attribute" serves as a descriptor for the data she is encrypting. Suppose the TA issues a secret key for some *access policy* to Bob. This access policy essentially describes which attributes he is authorized to access. Bob can decrypt Alice's ciphertext if its associated "attribute" satisfies his *access policy*.

We refer to the result of an evaluation on a set of ciphertexts as an *evaluated ciphertext*.

2.1 Models of Access Control for Decryption

A model of access control for decryption specifies how decryption of an evaluated ciphertext is to be performed. Consider an evaluated ciphertext c' associated with $\mathscr{d}$ attributes $a_1, \ldots, a_{\mathscr{d}} \in \mathbb{A}$. There are two primary models of decryption, each with different strengths and weaknesses. Both models will be considered in turn.

2.1.1 Atomic Access

The intended semantics of this model is that a user should only be able to decrypt an evaluated ciphertext c' if she has a secret key for a policy f that satisfies *all* $\mathscr{d}$ attributes $a_1, \ldots, a_{\mathscr{d}}$. In other words, policies are enforced in an "all or nothing" manner. So in order to decrypt a ciphertext c', the decryptor needs a secret key for a policy f with $f(a_1) = \cdots = f(a_{\mathscr{d}}) = 1$. Furthermore, it captures the natural requirement that a decryptor be authorized *completely* to access data associated with a particular attribute.

2.1.2 Non-Atomic Access - Collaborative Decryption

The interpretation in this model is that a group of users can pool together their secret keys to decrypt a ciphertext c'. In other words, there may not be a single $f \in \mathbb{F}$ that satisfies all $\mathscr{d}$ attributes (or no user holds a secret key for such an f), but the users may share secret keys for a set of policies that "covers all" $\mathscr{d}$ attributes. In other words, suppose the group of users have (between them) secret keys for policies $f_1, \ldots, f_{\mathscr{k}} \in \mathbb{F}$. In this model, they can decrypt c' if and only if for every $i \in [\mathscr{d}]$, there exists a $j \in [\mathscr{k}]$ such that $f_j(a_i) = 1$.

How is decryption performed? There are a few possible approaches:

1. Every user in the group shares their secret keys with each other, and all users can decrypt. However, this violates the *principle of least privilege* and gives users in the group access to data they might not have been explicitly authorized to access.

2. Perform decryption collaboratively using a multi-party computation (MPC) protocol. This approach has been suggested in other works including (López-Alt et al. (2012)). The advantage of this approach is that it does not leak any party's secret key to the other parties.

3. It is possible that a user has been issued secret keys for several policies. For example: ABE for disjunctive policies can be achieved with an IBE scheme where the TA issues secret keys for different identities (treated as "attributes") to the same user.

4. Collaborative decryption subsumes the *functionality* of the atomic model i.e. a user with a single policy f satisfying all $\mathscr{d}$ attributes can still decrypt on her own.

Our syntax for attribute based homomorphic encryption (ABHE) presented in the next section generalizes both models. We do this by parameterizing an ABHE scheme with an integer $\mathcal{K} \in [\mathcal{D}]$, which specifies the maximum number of keys that can be passed to the decryption algorithm. The setting $\mathcal{K} = 1$

specifies the atomic model whereas the setting $\mathcal{K} = \mathcal{D}$ specifies the collaborative model. Note that this is only a syntactic rule, it does not pertain to enforcing the security property of either model. Our "default" model, assumed implicitly without further qualification, is the collaborative model. This is for several reasons, which we will enumerate now:

- In the identity-based setting, collaborative decryption is necessary. In this context, a single f is satisfied by only one attribute (i.e. identity). Suppose an evaluation is performed on ciphertexts with *different* identities to yield an evaluated ciphertext c'. Clearly, there is no single secret key that is sufficient to decrypt c', since each secret key corresponds to exactly one identity. Because IBE is a special case of ABE, and very important in its own right, we want to ensure we allow multi-identity evaluation.
- As noted above, the collaborative model subsumes the *functionality* of the atomic model. The greater flexibility of permitting multiple users to collaboratively decrypt (such as via MPC) invites more applications.

2.2 Definition of Attribute-Based Homomorphic Encryption

Recall the definition of ABE from the introduction. An ABE scheme with message space $\mathcal{M}$, attribute space $\mathbb{A}$ and class of supported access policies $\mathbb{F}$ is a tuple of probabilistic polynomial time (PPT) algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$.

Degree of composition: Let $c_1, \ldots, c_\ell$ be input ciphertexts to an evaluation. Each ciphertext c_i is associated with an attribute $a_i \in \mathbb{A}$. The **degree of composition** of the evaluation is the number of **distinct** attributes among the a_i; that is, the cardinality of the set $|\{a_1, \ldots, a_\ell\}|$.

We use the symbol $\mathscr{d}$ to denote the degree of composition. When the context is unambiguous, the term is abbreviated to *degree*. We use the symbol $\mathcal{D}$ to denote the *maximum* degree of composition supported by a particular system.

Definition 2.2: A (Key-Policy) Attribute-Based Homomorphic Encryption (ABHE) scheme $\mathcal{E}^{(\mathcal{D},\mathcal{K})}$ for an integer $\mathcal{D} > 0$ and an integer $\mathcal{K} \in [\mathcal{D}]$ is defined with respect to a message space $\mathcal{M}$, an attribute space $\mathbb{A}$, a class of access policies $\mathbb{F} \subseteq \mathbb{A} \to \{0,1\}$, and a class of circuits $\mathbb{C} \subseteq \mathcal{M}^* \to \mathcal{M}$. An ABHE scheme is a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ where Setup, KeyGen, $\mathsf{Encrypt}$ are defined equivalently to KP-ABE. We denote by $\mathcal{C}$ the ciphertext space. The decryption algorithm $\mathsf{Decrypt}$ and evaluation algorithm Eval are defined as follows:

- $\mathsf{Decrypt}(\langle \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_k} \rangle, c)$: On input a sequence of $k \leq \mathcal{K}$ secret keys for policies $f_1, \ldots, f_k \in \mathbb{F}$ and a ciphertext c, output a plaintext $\mu' \in \mathcal{M}$ iff every attribute associated with c is satisfied by at least one of the f_i; output $\perp$ otherwise.
- $\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$: On input public parameters PP, a circuit $C \in \mathbb{C}$ and ciphertexts $c_1, \ldots, c_\ell \in \mathcal{C}$, output an *evaluated ciphertext* $c' \in \mathcal{C}$.

More precisely, Eval is required to satisfy the following properties:

- Over all choices of $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$, $C : \mathcal{M}^\ell \to \mathcal{M} \in \mathbb{C}$, every $\mathscr{d} \leq \mathcal{D}$, $a_1, \ldots, a_\ell \in \mathbb{A}$ s.t $|\{a_1, \ldots, a_\ell\}| = \mathscr{d}$, $\mu_1, \ldots, \mu_\ell \in \mathcal{M}$, $c_i \leftarrow \mathsf{Encrypt}(\mathsf{PP}, a_i, \mu_i)$ for $i \in [\ell]$, and $c' \leftarrow \mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$:
  - **Correctness**

$$\mathsf{Decrypt}(\langle \mathsf{sk}_{f_1}, \ldots, \mathsf{sk}_{f_k} \rangle, c') = C(\mu_1, \ldots, \mu_\ell) \\ \text{iff } \forall i \in [\mathscr{d}]\ \exists j \in [k] \quad f_j(a_i) = 1 \tag{2.1}$$

    for any $k \in [\mathcal{K}]$, any $f_1, \ldots, f_k \in \mathbb{F}$, and any $\mathsf{sk}_{f_j} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, f_j)$ for $j \in [k]$.
  - **Compactness** There exists a fixed polynomial $s(\cdot, \cdot)$ for the scheme such that

$$|c'| \leq s(\lambda, \mathscr{d}). \tag{2.2}$$

The complexity of all algorithms may depend on $\mathcal{D}$. Furthermore, the size of freshly encrypted ciphertexts, the size of the public parameters and the size of secret keys may depend on $\mathcal{D}$. On the other hand, the size of the evaluated ciphertext c' must remain independent of $\mathcal{D}$ (along with the size of the circuit C), but it may depend on the *actual* number of distinct attributes, $\mathscr{d}$, used in the evaluation. Note that single-attribute ABHE is the special case where $\mathcal{D} = 1$ i.e. evaluation is correct only for ciphertexts associated with the same attribute. As mentioned earlier, $\mathcal{K} = 1$ represents the atomic model of decryption whereas $\mathcal{K} = \mathcal{D}$ represents the collaborative model. When the parameter $\mathcal{K}$ is omitted, it can be assumed that $\mathcal{K} = \mathcal{D}$; that is, the notation $\mathcal{E}^{(\mathcal{D})}$ is shorthand for $\mathcal{E}^{(\mathcal{D},\mathcal{D})}$.

Definition 2.3: Multi-Attribute ABHE (MA-ABHE) is a primitive with the same syntax as ABHE except that its Setup algorithm takes an additional input $\mathcal{D} > 0$, which is the maximum degree of composition to support. An instance of MA-ABHE can be viewed as a family of ABHE schemes $\{\mathcal{E}^{(\mathcal{D})} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})\}_{\mathcal{D}>0}$.

Remark 1: In the constructions considered in this work, $\mathbb{A}$ consists of attributes of fixed length. However the above definition is easily generalized to capture variable-length attributes, by letting $|c'|$ grow with the total length of the $\mathscr{d}$ distinct attributes.

A concrete ABHE scheme is characterized by three facets: 1). its supported computations (i.e. the class of circuits $\mathbb{C}$); 2). its supported access policies (the class of access policies $\mathbb{F}$); and 3). its supported composition defined by its maximum degree of composition, $\mathcal{D}$.

3 Security Definitions

3.1 Semantic Security

The semantic security definition for ABHE is the same as that for ABE, except that the adversary has access to the Eval algorithm as well. There are two definitions of semantic security for ABE: selective and adaptive security. In the selective security game, the adversary chooses the attribute to attack before receiving the public parameters whereas in the adaptive game, the adversary chooses its target attribute after receiving the public parameters. We denote the selective definition by $\mathsf{IND\text{-}sel\text{-}CPA}$ and the adaptive definition by $\mathsf{IND\text{-}AD\text{-}CPA}$.

3.2 Simulation Model of Evaluation

Let $\mathcal{D}$ and $\mathcal{K} \leq \mathcal{D}$ be fixed parameters denoting the maximum degree of composition and the maximum number of keys passed to the decryption algorithm respectively. Consider ciphertexts $c_1, \ldots, c_\ell$ encrypted under attributes $a_1, \ldots, a_\ell$ respectively. We expect that a ciphertext c' resulting from an evaluation on $c_1, \ldots, c_\ell$ be decryptable by a set of policies $\{f_i\}_{i \in [\mathcal{k}]}$ with $\mathcal{k} \in [\mathcal{K}]$ if the following two conditions are satisfied: (1). the degree of composition $\mathcal{d}$ is less than $\mathcal{D}$ (i.e. $\mathcal{d} := |\{a_1, \ldots, a_\ell\}| \leq \mathcal{D}$) - for convenience we re-label the $\mathcal{d}$ *distinct* attributes as $a_1, \ldots, a_{\mathcal{d}}$; and (2). for every $i \in [\mathcal{d}]$, there exists a $j \in [\mathcal{k}]$ with $f_j(a_i) = 1$.

Ideally a user who does not have keys for such a set of policies $\{f_i\}_{i \in [\mathcal{k}]}$ should not learn anything about c' except that it is associated with the attributes $a_1, \ldots, a_{\mathcal{d}}$. This implies that such a user should not be able to efficiently decide whether c' was produced from $c_1, \ldots, c_\ell$ or an alternative sequence of ciphertexts $d_1, \ldots, d_{\ell'}$ with the same collection of distinct attributes $a_1, \ldots, a_{\mathcal{d}}$. We now give a definition of security that captures the fact that an adversary learns nothing from an evaluated ciphertext other than that it was generated from a particular circuit and is associated with the attributes $a_1, \ldots, a_{\mathcal{d}}$.

$\mathsf{EVAL\text{-}SIM}$ Security: Let $F \subseteq \mathbb{F}$ be a set of policies, and let $A \subseteq \mathbb{A}$ be a set of attributes. For convenience, we define the predicate

$$\mathsf{compat}(F, A) = \begin{cases} 1 & \text{if } \exists a \in A \; \forall f \in F \; f(a) = 0 \\ 0 & \text{otherwise .} \end{cases}$$

Let $\mathcal{E}$ be an ABHE scheme with parameters $\mathcal{D}$ and $\mathcal{K}$. We define the following experiments for a pair of PPT adversarial algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and a PPT algorithm $\mathcal{S}$.

- $\mathbf{Exp}^{\mathsf{REAL}}_{\mathcal{E},\mathcal{A}}(\lambda)$ **(Real World)**:
  1. $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$.
  2. $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK}, \cdot)}(\mathsf{PP})$.
  3. Let F be the set of policies queried by $\mathcal{A}_1$.
  4. Let $A := \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathcal{d}}\}$ be the distinct attributes in the collection $a_1, \ldots, a_\ell$.
  5. Assert $\mathcal{d} \leq \mathcal{D}$ and $\mathsf{compat}(F, A) = 1$; otherwise output a random bit and abort.
  6. $c_j \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, a_j, \mu_j)$ for $j \in [\ell]$.
  7. $c' \leftarrow \mathcal{E}.\mathsf{Eval}(\mathsf{PP}, C, c_1, \ldots, c_\ell)$.
  8. $b \leftarrow \mathcal{A}_2^{\mathcal{O}(\mathsf{MSK}, \cdot)}(\mathsf{st}, c', c_1, \ldots, c_\ell)$
  9. Output b.
- $\mathbf{Exp}^{\mathsf{IDEAL}}_{\mathcal{E},\mathcal{A},\mathcal{S}}(\lambda)$ **(Ideal World)**:
  1. $(\mathsf{PP}, \mathsf{MSK}) \leftarrow \mathcal{E}.\mathsf{Setup}(1^\lambda)$.
  2. $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK}, \cdot)}(\mathsf{PP})$.
  3. Let F be the set of policies queried by $\mathcal{A}_1$.
  4. Let $A := \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathcal{d}}\}$ be the distinct attributes in the collection $a_1, \ldots, a_\ell$.
  5. Assert $\mathcal{d} \leq \mathcal{D}$ and $\mathsf{compat}(F, A) = 1$; otherwise output a random bit and abort.
  6. $c_j \leftarrow \mathcal{E}.\mathsf{Encrypt}(\mathsf{PP}, a_j, \mu_j)$ for $j \in [\ell]$.
  7. $c' \leftarrow \mathcal{S}(\mathsf{PP}, C, A)$.
  8. $b \leftarrow \mathcal{A}_2^{\mathcal{O}(\mathsf{MSK}, \cdot)}(\mathsf{st}, c', c_1, \ldots, c_\ell)$
  9. Output b.

where $\mathcal{O}(\mathsf{MSK}, \cdot)$ is defined as:

- $\mathcal{O}(\mathsf{MSK}, f)$:
  1. If $\mathsf{compat}(F \cup \{f\}, A) = 1$: set $F \leftarrow F \cup \{f\}$ and output $\mathcal{E}.\mathsf{KeyGen}(\mathsf{MSK}, f)$.
  2. Else output $\perp$.

Then $\mathcal{E}$ is said to be $\mathsf{EVAL\text{-}SIM}$-secure if there exists a PPT simulator $\mathcal{S}$ such that for every pair of PPT algorithms $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$, it holds that

$$|\mathsf{Pr}[\mathbf{Exp}^{\mathsf{REAL}}_{\mathcal{E},\mathcal{A}} \to 1] - \mathsf{Pr}[\mathbf{Exp}^{\mathsf{IDEAL}}_{\mathcal{E},\mathcal{A},\mathcal{S}} \to 1]| < \mathsf{negl}(\lambda).$$

Note that the above definition relates to adaptive security. For selective security, the adversary must choose the attributes before receiving the public parameters. As a result, in the modified definition, $\mathcal{A}$ consists of three PPT algorithms $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$. Furthermore, $\mathcal{A}_1$ outputs a set of $\mathcal{d} \leq \mathcal{D}$ attributes $A := \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathcal{d}}\}$; $\mathcal{A}_2$ receives PP and outputs a circuit C along with a sequence of ℓ pairs (μ_i, a_i) for $i \in [\ell]$ where $\mu_i \in \mathcal{M}$ and $a_i \in A$. Finally, $\mathcal{A}_3$ is defined equivalently to $\mathcal{A}_2$ in the above definition. We denote the selective variant by $\mathsf{sel\text{-}EVAL\text{-}SIM}$.

4 Construction

4.1 Building Blocks

4.1.1 Multi-Key FHE

Multi-Key FHE allows multiple independently-generated keys to be used together in a homomorphic evaluation. The syntax of multi-key FHE imposes a limit N on the number of such keys that can be supported. Furthermore, the size of the evaluated ciphertext does not depend on the size of the circuit (or number of inputs), but instead on the number of independent keys N that is supported. In order to decrypt, the parties who have the corresponding secret keys must collaborate such as in an MPC protocol.

Based on Definition 2.1 in (López-Alt et al. (2012)): A multi-key $\mathbb{C}$-homomorphic scheme family for a class of circuits $\mathbb{C}$ and message space $\mathcal{M}$ is a family of PPT algorithms $\{\mathcal{E}^{(N)} := (\mathsf{Gen}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})\}_{N>0}$ where $\mathcal{E}^{(N)}$ is defined as follows:

- $\mathsf{MKFHE.Gen}$ takes as input the security parameter 1^λ and outputs a tuple $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk})$ where pk is a public key, sk is a secret key and vk is an evaluation key.
- $\mathsf{MKFHE.Encrypt}$ takes as input a public key pk and a message $m \in \mathcal{M}$, and outputs an encryption of m under pk.
- $\mathsf{MKFHE.Decrypt}$ takes as input $1 \leq k \leq N$ secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_k$ and a ciphertext c, and outputs a message $m' \in \mathcal{M}$.
- $\mathsf{MKFHE.Eval}$ takes as input a circuit $C \in \mathbb{C}$, and ℓ pairs $(c_1, \mathsf{vk}_1), \ldots, (c_\ell, \mathsf{vk}_\ell)$ and outputs a ciphertext c'.

Informally, evaluation is only required to be *correct* if at most N keys are used in $\mathsf{MKFHE.Eval}$; that is, $|\{\mathsf{vk}_1, \ldots, \mathsf{vk}_\ell\}| \leq N$. Furthermore, the size of an evaluated ciphertext c' must only depend polynomially on the security parameter λ and the number of keys N, and not on the size of the circuit.

The IND-CPA security game for multi-key homomorphic encryption is the same as that for standard public-key encryption; note that the adversary is given the evaluation key vk.

There are two multi-key FHE schemes in the literature: the scheme of López-Alt, Tromer and Vaikuntanathan (López-Alt et al. (2012)) based on NTRU and the scheme of Clear and Mc Goldrick (Clear and Mc Goldrick (2015)) based on Learning wtih Errors (LWE). Although our construction can work with any multi-key FHE, we obtain better efficiency if we use the multi-key FHE scheme of Clear and Mc Goldrick, which we call CM. More precisely, the depth of the decryption circuit of CM is $O(\log N)$ (as opposed to $O(\log^2 N)$ in the case of the multi-key FHE from (López-Alt et al. (2012))) which results in fewer levels needed for the leveled ABFHE.

For the remainder of the paper, we will denote an instance of a multi-key FHE by $\mathcal{E}_{\mathsf{MKFHE}}$.

4.1.2 Leveled ABFHE

Our approach uses a leveled ABFHE scheme in an essential way. A leveled ABFHE scheme allows one to evaluate a circuit of bounded depth. The bound on the depth L is chosen in advance of generating the public parameters. Gentry, Sahai and Waters (Sahai and Waters (2013)) presented the first leveled ABFHE where the class of access policies consists of bounded-depth circuits. They based security on LWE. A leveled Identity-Based FHE (IBFHE) scheme from LWE is also presented in (Sahai and Waters (2013)). Furthermore a leveled IBFHE that is multi-identity (supports evaluation on ciphertexts with different identities) was constructed in (Clear and Mc Goldrick (2015)) from LWE.

Any of the above schemes can be used to instantiate our construction and its properties are inherited by our construction. Therefore if we use an identity-based scheme, our resulting construction is identity-based etc.

For the rest of the paper, we will denote a leveled ABFHE scheme by $\mathcal{E}_{\mathsf{lABFHE}}$ with message space $\mathcal{M}_{\mathcal{E}_{\mathsf{lABFHE}}}$, attribute space $\mathbb{A}_{\mathcal{E}_{\mathsf{lABFHE}}}$ and class of predicates $\mathbb{F}_{\mathcal{E}_{\mathsf{lABFHE}}}$.

4.2 Overview of Our Approach

The main idea behind our approach is to exploit multi-key FHE and leveled ABFHE to construct a new ABFHE scheme that can evaluate circuits with up to N inputs, where N is chosen before generating the public parameters. Let $\mathcal{E}_{\mathsf{MKFHE}}$ be a multi-key FHE scheme whose decryption circuit has depth $\delta(\lambda, N)$ where N is the number of independent keys tolerated and λ is the security parameter. Let $\mathcal{E}_{\mathsf{lABFHE}}$ be a leveled ABFHE scheme as described in Section 4.1.2 that can compactly evaluate circuits of depth $\delta(\lambda, N)$.

Let w be a positive integer. The supported message space of our scheme is $\mathcal{M} \triangleq \{0,1\}^w$. The supported attribute space is $\mathbb{A} \triangleq \mathbb{A}_{\mathcal{E}_{\mathsf{lABFHE}}}$ and the supported class of access policies is $\mathbb{F} \triangleq \mathbb{F}_{\mathcal{E}_{\mathsf{lABFHE}}}$. In other words, the attribute space and class of access policies is the same as the underlying leveled ABFHE scheme. Finally, the class of supported circuits is $\mathbb{C} \triangleq \mathcal{M}^N \rightarrow \mathcal{M}$.

Roughly speaking, to encrypt a message $\mu \in \mathcal{M}$ under attribute $a \in \mathbb{A}$ in our scheme, (1) a key triple $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk})$ is generated for $\mathcal{E}_{\mathsf{MKFHE}}$; (2) μ is encrypted with $\mathcal{E}_{\mathsf{MKFHE}}$ under pk; (3) sk is encrypted with $\mathcal{E}_{\mathsf{lABFHE}}$ under attribute a; (4) the two previous ciphertexts along with vk constitute the ciphertext that is produced. Therefore, $\mathcal{E}_{\mathsf{MKFHE}}$ is used for hiding the message and for homomorphic computation, whereas $\mathcal{E}_{\mathsf{lABFHE}}$ enforces access control by appropriately hiding the secret keys for $\mathcal{E}_{\mathsf{MKFHE}}$.

The evaluator performs homomorphic evaluation on the multi-key FHE ciphertexts and obtains a result c'.

It then homomorphically decrypts c' with the leveled ABFHE scheme using the encryptions of the secret keys for $\mathcal{E}_{\mathsf{MKFHE}}$. As a result we obtain a ciphertext whose length is independent of N and the circuit size, which satisfies our compactness condition.

In more concrete terms, we assume without loss of generality that the message space of $\mathcal{E}_{\mathsf{MKFHE}}$ is $\{0,1\}$, and we encrypt a w-bit message $\mu = (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$ one bit at a time using $\mathcal{E}_{\mathsf{MKFHE}}$. Furthermore, let N be the maximum number of keys supported by $\mathcal{E}_{\mathsf{MKFHE}}$. Our construction can therefore support the class of circuits $\mathbb{C} = \{(\{0,1\}^w)^N \to \{0,1\}^w\}$. We remind the reader that w can be arbitrarily large, and in practice, the length of plaintexts may be shorter than w. In practice, each sender's input may be of arbitrary size. However, there is a limit, N, on the number of independent senders i.e. the number of inputs to the circuit where the inputs are taken from the domain $\{0,1\}^w$.

4.3 Construction

We now present our construction, which we call bABFHE.

4.3.1 Setup

On input a security parameter λ and a bound N on the number of inputs to support, the following steps are performed:

1. Choose integer w.
2. Generate $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}) \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Setup}(1^\lambda, 1^L)$ where $L = O(\log \lambda \cdot N)$ is the depth of the decryption circuit of $\mathcal{E}_{\mathsf{lABFHE}}$ for parameters λ and N.
3. Output $(\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w), \mathsf{MSK} := (\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}))$.

4.3.2 Secret Key Generation

Given the master secret key $\mathsf{MSK} := (\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}})$ and a policy $f \in \mathbb{F}$, a secret key sk_f for f is generated as $\mathsf{sk}_f \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{KeyGen}(\mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}, f)$. The secret key $\mathsf{SK}_f := (\mathsf{PP}, \mathsf{sk}_f)$ is issued to the user.

4.3.3 Encryption

On input public parameters $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$, a binary string $\mu = (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$ and an attribute $a \in \mathbb{A}$: the sender first generates a key triple for $\mathcal{E}_{\mathsf{MKFHE}}$; that is, she computes $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N)$. Then she runs $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a, \mathsf{sk})$. Subsequently she uses pk to encrypt each bit $\mu_i \in \{0,1\}$ in turn using $\mathcal{E}_{\mathsf{MKFHE}}$ for $i \in [w]$; that is, she computes $c_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}, \mu_i)$. Finally she outputs the ciphertext $\mathsf{CT} := (\mathsf{type} := 0, \mathsf{enc} := (\psi, \mathsf{vk}, (c_1, \ldots, c_w)))$.

Remark 2: A ciphertext CT in our scheme has two components: the first is labeled with type and the second is labeled with enc. The former has two valid values: 0 and 1; 0 indicates that the ciphertext is "fresh" while 1 indicates that the ciphertext is the result of an evaluation. The value of the type component specifies how the enc component is to be parsed.

4.3.4 Evaluation

On input public parameters $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$, a circuit $C \in \mathbb{C}$, and ciphertexts $\mathsf{CT}_1, \ldots, \mathsf{CT}_\ell$ with $\ell \leq N$, the evaluator performs the following steps. Firstly, the ciphertexts are assumed to be "fresh" ciphertexts generated with the encryption algorithm. In other words, their type components are all 0. Otherwise the evaluator outputs $\perp$. Consequently, the evaluator can parse CT_i as $(\mathsf{type} := 0, \mathsf{enc} := (\psi_i, \mathsf{vk}_i, (c_1^{(i)}, \ldots, c_w^{(i)})))$ for every $i \in [\ell]$. We denote by a_i the attribute associated with the $\mathcal{E}_{\mathsf{lABFHE}}$ ciphertext ψ_i. The maximum degree of composition of our construction is inherited from that of the underlying leveled ABFHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$. We denote this as usual by $\mathcal{D}$. The evaluator derives the degree of composition as $\mathcal{d} \leftarrow |\{a_1, \ldots, a_\ell\}|$, and outputs $\perp$ and aborts unless $\mathcal{d} \leq \mathcal{D}$.

Next the evaluator computes

$$c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1^{(1)}, \mathsf{vk}_1), \ldots, (c_w^{(1)}, \mathsf{vk}_1), \ldots, (c_1^{(\ell)}, \mathsf{vk}_\ell), \ldots, (c_w^{(\ell)}, \mathsf{vk}_\ell))$$

and encrypts this ciphertext with the leveled ABFHE scheme under any arbitrary a_i, say a_1; that is, the evaluator computes $\psi_{c'} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a_1, c')$. The final step is to evaluate using $\mathcal{E}_{\mathsf{lABFHE}}$ the decryption circuit $D_{\langle N,\lambda\rangle}$[1] of $\mathcal{E}_{\mathsf{MKFHE}}$:

$$\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Eval}\big(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, D_{\langle N,\lambda\rangle}, \psi_{c'}, \psi_1, \ldots, \psi_\ell\big).$$

The evaluator outputs the *evaluated ciphertext* $\mathsf{CT}' := (\mathsf{type} := 1, \mathsf{enc} := \psi)$.

Remark 3: Observe that a "fresh" ciphertext has a different form to an evaluated ciphertext. Further evaluation with evaluated ciphertexts is not guaranteed by our construction. Hence it is a 1-hop homomorphic scheme using the terminology of Gentry, Halevi and Vaikuntanathan (Gentry et al. (2010)).

4.3.5 Decryption

To decrypt a ciphertext $\mathsf{CT} := (\mathsf{type}, \mathsf{enc})$ with a sequence of secret keys $\langle \mathsf{SK}_{f_1} := (\mathsf{PP}, \mathsf{sk}_{f_1}), \ldots, \mathsf{SK}_{f_k} := (\mathsf{PP}, \mathsf{sk}_{f_k})\rangle$ for respective policies $f_1, \ldots, f_k \in \mathbb{F}$, a decryptor performs the following steps.

If CT is a "fresh" ciphertext (i.e. $\mathsf{type} = 0$), then enc is parsed as $(\psi, \mathsf{vk}, (c_1, \ldots, c_w))$ and the decryptor computes $\mathsf{sk} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_k\rangle, \psi)$. If $\mathsf{sk} = \perp$, then the decryptor outputs $\perp$ and aborts. Otherwise, she computes

$$\mu_j \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}(\mathsf{sk}, c_j) \text{ for every } j \in [w]$$

Figure 1 Formal Description of scheme bABFHE.

Setup$(1^\lambda, 1^N)$:

1. Choose integer w.
2. Let $g(\cdot,\cdot)$ be a polynomial associated with $\mathcal{E}_{\mathsf{MKFHE}}$ that gives the number of inputs to the decryption circuit for N keys and security parameter λ. Let $L = g(\lambda, N)$.
3. Generate $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}) \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Setup}(1^\lambda, 1^L)$.
4. Output $(\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w), \mathsf{MSK} := \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}})$.

Encrypt(PP, a, μ) :

1. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
2. Parse μ as $(\mu_1, \ldots, \mu_w) \in \{0,1\}^w$.
3. $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N)$
4. $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a, \mathsf{sk})$.
5. $c_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}, \mu_i)$ for $i \in [w]$.
6. Output $\mathsf{CT} := (\mathsf{type} := 0, \mathsf{enc} := (\psi, \mathsf{vk}, (c_1, \ldots, c_w)))$.

KeyGen(MSK, f) :

1. Parse MSK as $(\mathsf{PP}, \mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}})$.
2. $\mathsf{sk}_f \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{KeyGen}(\mathsf{MSK}_{\mathcal{E}_{\mathsf{lABFHE}}}, f)$.
3. Output $\mathsf{SK}_f := (\mathsf{PP}, \mathsf{sk}_f)$.

Decrypt$(\langle \mathsf{SK}_{f_1}, \ldots, \mathsf{SK}_{f_k} \rangle, \mathsf{CT})$:

1. If $k > \mathcal{K}$: output $\perp$ and abort.
2. Parse SK_{f_i} as $(\mathsf{PP}, \mathsf{sk}_{f_i})$ for $i \in [k]$.
3. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
4. Parse CT as $(\mathsf{type}, \mathsf{enc})$.
5. If $\mathsf{type} = 0$:
   (a) Parse enc as $(\psi, \mathsf{vk}, (c_1, \ldots, c_w))$
   (b) Compute $\mathsf{sk} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_k \rangle, \psi)$.
   (c) If $\mathsf{sk} = \perp$: output $\perp$ and abort.
   (d) $\mu_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}(\mathsf{sk}, c_i)$ for $i \in [w]$.
   (e) Output $\mu := (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$.
6. Else If $\mathsf{type} = 1$:
   (a) Parse enc as ψ.
   (b) Compute $x \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_k \rangle, \psi)$.
   (c) If $x = \perp$: output $\perp$ and abort.
   (d) Output $\mu := x \in \{0,1\}^w$.
7. Else output $\perp$.

Eval$(\mathsf{PP}, C, \mathsf{CT}_1, \ldots, \mathsf{CT}_\ell)$:

1. If $\ell > N$: output $\perp$ and abort.
2. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
3. For $i \in [\ell]$:
   (a) Parse CT_i as $(\mathsf{type} := 0, \mathsf{enc} := (\psi_i, \mathsf{vk}_i, (c_1^{(i)}, \ldots, c_w^{(i)})))$.
   (b) Set a_i as the attribute associated with ψ_i.
4. Set $d \leftarrow |\{a_1, \ldots, a_\ell\}|$ (degree of composition).
5. If $d > \mathcal{D}$: output $\perp$ and abort.
6. $c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1^{(1)}, \mathsf{vk}_1), \ldots, (c_w^{(1)}, \mathsf{vk}_1), \ldots, (c_1^{(\ell)}, \mathsf{vk}_\ell), \ldots, (c_w^{(\ell)}, \mathsf{vk}_\ell))$.
7. $\psi_{c'} \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a_1, c')$.
8. Let $D_{\langle N,\lambda \rangle}$ be the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ for parameters N and λ.
9. $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Eval}\big(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, D_{\langle N,\lambda \rangle}, \psi_{c'}, \psi_1, \ldots, \psi_\ell\big)$.
10. Output $\mathsf{CT}' := (\mathsf{type} := 1, \mathsf{enc} := \psi)$.

and outputs the plaintext $\mu := (\mu_1, \ldots, \mu_w) \in \{0,1\}^w$.

If CT is an evaluated ciphertext (i.e. $\mathsf{type} = 1$), then the decryptor parses enc as ψ and computes $x \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Decrypt}(\langle \mathsf{sk}_1, \ldots, \mathsf{sk}_k \rangle, \psi)$. If $x = \perp$ the decryptor outputs $\perp$ and aborts; otherwise the plaintext $\mu := x \in \{0,1\}^w$ is outputted.

4.4 Formal Description

A formal description of the construction bABFHE is given in Figure 1. As mentioned previously, the parameters $\mathcal{D}$ (maximum degree of composition) and $\mathcal{K}$ (maximum number of decryption keys passed to Decrypt) are inherited directly from the underlying leveled ABFHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$. Although circuits in the supported class send a sequence of elements in the message space $\mathcal{M} := \{0,1\}^w$ to another element in the message space $\mathcal{M}$, we simplify the description here and assume that each circuit C outputs a single bit. A circuit $\hat{C}$ in our supported class can then be modelled as w such circuits.

4.5 Correctness

In the evaluation algorithm, the desired N-ary circuit C whose N inputs are over the domain $\{0,1\}^w$ is evaluated using the multi-key FHE scheme. Observe that C can be of arbitrary depth since the size of the resultant multi-key FHE ciphertext only depends on λ and N. We then encrypt this resulting ciphertext with $\mathcal{E}_{\mathsf{lABFHE}}$ in order to homomorphically evaluate the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ using $\mathcal{E}_{\mathsf{lABFHE}}$. Consequently, we obtain a ciphertext whose size is independent of N as required by the compactness condition for ABHE.

5 Security

5.1 Semantic Security

Without loss of generality we assume that the message space $\mathcal{M}_{\mathcal{E}_{\mathsf{lABFHE}}}$ of $\mathcal{E}_{\mathsf{lABFHE}}$ is big enough to represent secret keys in $\mathcal{E}_{\mathsf{MKFHE}}$ and binary strings in $\mathcal{M}$.

Lemma 1: *If $\mathcal{E}_{\mathsf{lABFHE}}$ is an* IND-X-CPA*-secure leveled ABFHE scheme and $\mathcal{E}_{\mathsf{MKFHE}}$ is an* IND-CPA*-secure*

multi-key FHE scheme, then bABFHE *is* IND-X-CPA *where* $X \in \{\mathsf{sel}, \mathsf{AD}\}$.

Proof. We prove the lemma by means of a hybrid argument.
Hybrid 0 IND-X-CPA game for bABFHE.

Hybrid 1 Same as Hybrid 0 except with one difference. Let $a^\star \in \mathbb{A}$ be the target attribute chosen by the adversary $\mathcal{A}$. The challenger uses a modified Encrypt algorithm to compute the leveled ABFHE ciphertext corresponding to a^* by replacing Step 4 with $\psi \leftarrow \mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Encrypt}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, a^*, 0^{|\mathsf{sk}|})$ where $0^{|\mathsf{sk}|}$ is a string of zeros whose length is the same as the multi-key FHE secret key generated in Step 3 of Encrypt. The algorithm is otherwise unchanged.

We claim that any poly-time $\mathcal{A}$ that can distinguish between Hybrid 0 and Hybrid 1 with a non-negligible advantage can break the IND-X-CPA security of $\mathcal{E}_{\mathsf{lABFHE}}$. An adversary $\mathcal{B}$ that uses $\mathcal{A}$ proceeds as follows. When $\mathcal{A}$ chooses a target attribute $a^\star$, $\mathcal{B}$ generates a key-triple for $\mathcal{E}_{\mathsf{MKFHE}}$ i.e. it computes

$$(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N).$$

Then it gives a^* to its challenger along with two messages $x_0 := \mathsf{sk}$ and $x_1 := 0^{|\mathsf{sk}|}$. Note that we assume for simplicity that both messages are in $\mathcal{M}_{\mathcal{E}_{\mathsf{lABFHE}}}$; if multiple messages (say k) are required then the usual hybrid argument can be applied which loses a factor of k. Subsequently, $\mathcal{B}$ embeds the challenge leveled ABFHE ciphertext as the ψ component of its own challenge ciphertext CT^*. It computes the remaining components of CT^* as in the Encrypt algorithm. If ψ encrypts x_0, then $\mathcal{B}$ perfectly simulates Hybrid 0. Otherwise, $\mathcal{B}$ perfectly simulates Hybrid 1. Note that secret key queries made by $\mathcal{A}$ can be perfectly simulated by $\mathcal{B}$. Thus, if $\mathcal{A}$ has a non-negligible advantage distinguishing between the hybrids, then $\mathcal{B}$ has a non-negligible advantage attacking the IND-X-CPA security of $\mathcal{E}_{\mathsf{lABFHE}}$.

For $i \in [w]$:
Hybrid 1 + i Same as Hybrid $1 + (i-1)$ with the exception that the challenger does not encrypt message bit $\mu_i^{(0)}$ or $\mu_i^{(1)}$ (using $\mathcal{E}_{\mathsf{MKFHE}}$) chosen by $\mathcal{A}$. Instead it encrypts some fixed message bit $\beta \in \{0,1\}$.

We now show that if $\mathcal{A}$ can efficiently distinguish between Hybrid $1+i$ and Hybrid $1+(i-1)$, then there is a PPT algorithm $\mathcal{G}$ that can use $\mathcal{A}$ to attack the IND-CPA security of $\mathcal{E}_{\mathsf{MKFHE}}$. Let pk and vk be the public key and evaluation key that $\mathcal{G}$ receives from its challenger. When $\mathcal{A}$ chooses $\mu^{(0)} \in \{0,1\}^w$ and $\mu^{(1)} \in \{0,1\}$, $\mathcal{G}$ simply gives $\mu_i^{(b)}$ and β to its IND-CPA challenger where b is the bit it uniformly samples in its simulation of the IND-X-CPA challenger. Let $c^\star$ be the challenge ciphertext it receives from the IND-CPA challenger. It sets $c_i \leftarrow c^*$ in the challenge ciphertext CT^*. If $c^\star$ encrypts $\mu_i^{(b)}$, then the view of $\mathcal{A}$ is identical to Hybrid $1+(i-1)$. Otherwise, the view of $\mathcal{A}$ is identical to Hybrid $1+i$. Therefore, a non-negligible advantage obtained by $\mathcal{A}$ implies a non-negligible advantage for $\mathcal{G}$ in the IND-CPA game, and thus contradicts the IND-CPA security of $\mathcal{E}_{\mathsf{MKFHE}}$.

Finally observe that the adversary has a zero advantage in Hybrid $1+w$ because the challenge ciphertext contains no information about the challenger's bit.

5.2 EVAL-SIM *Security*

Recall the simulation-based security definition from Section 3.2, which we called EVAL-SIM security. In the following lemma, we show that bABFHE inherits EVAL-SIM security from $\mathcal{E}_{\mathsf{lABFHE}}$.

Lemma 2: *Let* $\mathcal{E}_{\mathsf{MKFHE}}$ *be an* IND-CPA *secure multi-key FHE scheme. Let* $\mathcal{E}_{\mathsf{lABFHE}}$ *be an* X-EVAL-SIM *secure ABHE scheme with* $X \in \{\mathsf{sel}, \mathsf{AD}\}$. *Then* bABFHE *is* X-EVAL-SIM *secure.*

Proof. By the hypothesized X-EVAL-SIM security of $\mathcal{E}_{\mathsf{lABFHE}}$, there exists a PPT simulator $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$ such that for all PPT adversaries $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}} := (\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}, \mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2})$ we have

$$|\mathsf{Pr}[\mathbf{Exp}^{\mathsf{REAL}}_{\mathcal{E}_{\mathsf{lABFHE}},\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}}} \rightarrow 1] - \mathsf{Pr}[\mathbf{Exp}^{\mathsf{IDEAL}}_{\mathcal{E}_{\mathsf{lABFHE}},\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}},\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}} \rightarrow 1]| < \mathsf{negl}(\lambda). \quad (5.1)$$

Remark 4: Note that in this proof we use the definition for adaptive EVAL-SIM security, which is slightly different to that for sel-EVAL-SIM security, but the argument holds analogously for the latter.

A simulator $\mathcal{S}$ can be constructed using $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$ in order to achieve X-EVAL-SIM security for bABFHE. The simulator $\mathcal{S}$ runs as follows:

- $\boldsymbol{\mathcal{S}}(\mathsf{PP}, C, \{a_1, \ldots, a_{\mathscr{d}}\})$ with $\mathscr{d} \leq \mathcal{D}$, $a_1, \ldots, a_{\mathscr{d}} \in \mathbb{A}$ and $C \in \mathbb{C}$:
  1. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
  2. Let $D_{\langle N,\lambda\rangle}$ be the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ for parameters N and λ.
  3. Output $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, D_{\langle N,\lambda\rangle}, \{a_1, \ldots, a_{\mathscr{d}}\})$.

We claim that if there exists a PPT adversary $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$ with a non-negligible advantage distinguishing the real distribution and ideal distribution for bABFHE (with respect to $\mathcal{S}$), then there exists a PPT adversary $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}}} := (\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}, \mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2})$ with a non-negligible advantage distinguishing the real distribution and ideal distribution for $\mathcal{E}_{\mathsf{lABFHE}}$ (with respect to $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$). If this claim were to hold it would contradict the hypothesized X-EVAL-SIM security of $\mathcal{E}_{\mathsf{lABFHE}}$, which seals the lemma. To prove the claim, we show how to construct $(\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}, \mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2})$ from $(\mathcal{A}_1, \mathcal{A}_2)$. The algorithm $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}$ is given as input the public parameters $\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}$ for $\mathcal{E}_{\mathsf{lABFHE}}$. We denote its key generation oracle by $\mathcal{O}_1$. It runs as follows.

1. Set $\mathsf{PP} := (\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$ (the parameters N and w are fixed elsewhere).
2. Run $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_1}(\mathsf{PP})$.
3. For $i \in [\ell]$:
   (a) Parse μ_i as $(\mu_1^{(i)}, \ldots, \mu_w^{(i)}) \in \{0,1\}^w$.
   (b) $(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda, 1^N)$
   (c) $c_j^{(i)} \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}, \mu_j^{(i)})$ for $j \in [w]$.
4. Set $\mathscr{d} \leftarrow |\{a_1, \ldots, a_\ell\}|$ (degree of composition).
5. $c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1^{(1)}, \mathsf{vk}_1), \ldots, (c_w^{(1)}, \mathsf{vk}_1), \ldots, (c_1^{(\ell)}, \mathsf{vk}_\ell), \ldots, (c_w^{(\ell)}, \mathsf{vk}_\ell))$.
6. Let $D_{\langle N,\lambda\rangle}$ be the decryption circuit of $\mathcal{E}_{\mathsf{MKFHE}}$ for parameters N and λ.
7. Set $\mathsf{state} \leftarrow (\mathsf{st}, \mathsf{PP}, (\mathsf{vk}_1, (c_1^{(1)}, \ldots, c_w^{(1)})), \ldots, (\mathsf{vk}_\ell, (c_1^{(\ell)}, \ldots, c_w^{(\ell)})))$.
8. Output $(D_{\langle N,\lambda\rangle}, (a_1, c'), (a_1, \mathsf{sk}_1), \ldots, (a_\ell, \mathsf{sk}_\ell), \mathsf{state})$.

The algorithm $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},2}$ is given as input the state state (generated in $\mathcal{A}_{\mathcal{E}_{\mathsf{lABFHE}},1}$), the evaluated ciphertext ψ' along with the $\ell+1$ "input ciphertexts" (which we denote by $\psi_{c'}, \psi_1, \ldots, \psi_\ell$) and attributes $\{a_1, \ldots, a_{\mathscr{d}}\}$. We denote its key generation oracle by $\mathcal{O}_2$. It runs as follows.

1. Parse state as $(\mathsf{st}, \mathsf{PP}, (\mathsf{vk}_1, (c_1^{(1)}, \ldots, c_w^{(1)})), \ldots, (\mathsf{vk}_\ell, (c_1^{(\ell)}, \ldots, c_w^{(\ell)})))$.
2. Parse PP as $(\mathsf{PP}_{\mathcal{E}_{\mathsf{lABFHE}}}, \lambda, N, w)$.
3. Generate bABFHE input ciphertext $\mathsf{CT}_i \leftarrow (\mathsf{type} := 0, \mathsf{enc} := (\psi_i, \mathsf{vk}_i, (c_1^{(i)}, \ldots, c_w^{(i)})))$ for $i \in [\ell]$.
4. Generate bABFHE evaluated ciphertext $\mathsf{CT}' \leftarrow (\mathsf{type} := 1, \mathsf{enc} := \psi')$.
5. Run $b \leftarrow \mathcal{A}_2^{\mathcal{O}_2}(\mathsf{st}, \mathsf{CT}', \mathsf{CT}_1, \ldots, \mathsf{CT}_\ell)$.
6. Output b.

If ψ' is generated with $\mathcal{E}_{\mathsf{lABFHE}}.\mathsf{Eval}$ (i.e. the real distribution) then CT' is distributed identically to the output of bABFHE.Eval. On the other hand, if ψ' is generated with $\mathcal{S}_{\mathcal{E}_{\mathsf{lABFHE}}}$ (i.e. the ideal distribution), then CT' is distributed identically to $\mathcal{S}$. Therefore, a non-negligible advantage against bABFHE implies a non-negligible advantage against $\mathcal{E}_{\mathsf{lABFHE}}$.

6 Main Result

Theorem 3: *Let N be a positive integer. Let w be a positive integer. Let λ be a security parameter. Suppose there exists an* IND-CPA *secure multi-key FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ whose decryption circuit has depth $\delta(N, \lambda)$. Suppose there exists a leveled ABFHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$ that can compactly evaluate circuits of depth δ. Then there exists an ABHE scheme $\mathcal{E}$ (whose parameters $\mathcal{D}$ and $\mathcal{K}$ are the same as $\mathcal{E}_{\mathsf{lABFHE}}$) that can compactly evaluate all Boolean circuits in $\{(\{0,1\}^w)^N \rightarrow \{0,1\}^w\}$ i.e. the class of Boolean circuits of unbounded depth with N inputs over the domain $\{0,1\}^w$, such that*

1. *$\mathcal{E}$ is* IND-X-CPA *secure if $\mathcal{E}_{\mathsf{lABFHE}}$ is* IND-X-CPA *secure.*
2. *$\mathcal{E}$ is X-EVAL-SIM secure if $\mathcal{E}_{\mathsf{lABFHE}}$ is X-EVAL-SIM secure.*

for $X \in \{\mathsf{sel}, \mathsf{AD}\}$.

Proof. Instantiating our scheme bABFHE from Section 4.3 with the multi-key FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ and the ABHE scheme $\mathcal{E}_{\mathsf{lABFHE}}$, the theorem follows by appealing to Lemma 1 (IND-X-CPA security) and Lemma 2 (X-EVAL-SIM security).

Corollary 6.1: Let N be a positive integer. Assuming the hardness of LWE, there exists a IND-sel-CPA secure ABFHE that can compactly evaluate circuits with N inputs.

Proof. We can instantiate the multi-key FHE scheme in our construction with the CM multi-key FHE from (Clear and Mc Goldrick (2015)), whose security is based on LWE. Furthermore we can instantiate the leveled ABFHE in our construction with the leveled ABFHE of Gentry, Sahai and Waters (Sahai and Waters (2013)), which is shown to be selectively secure under LWE.

6.1 Discussion

We could instantiate $\mathcal{E}_{\mathsf{MKFHE}}$ with the multi-key FHE scheme of López-Alt, Tromer and Vaikuntanathan (López-Alt et al. (2012)). However its decryption circuit has depth $O(\log^2(N \cdot \lambda))$ as opposed to $O(\log(N \cdot \lambda))$ for CM, which means that the leveled ABFHE scheme must be set up to accommodate more levels, which in turn causes the parameters to blow up. Suppose we set N to be a large value so as not to practically limit the number of inputs to a circuit. As a result, N dominates λ. Therefore we need the leveled ABFHE to evaluate roughly $O(\log N)$ levels. Concretely, suppose we were to pick a very large value of N, say $N = 2^{32}$, then we need a leveled ABFHE that can evaluate on the order of 32 levels.

7 sel-EVAL-SIM Security of ABFHE from Obfuscation

In this section we prove the sel-EVAL-SIM security of the multi-attribute ABFHE scheme from (Clear and Mc Goldrick (2014)), which is based on indistinguishability obfuscation., Firstly we review the scheme from (Clear and Mc Goldrick (2014)), which we call MABFHE. Then

we introduce a new assumption we need to make. Finally we give the proof.

7.1 MABFHE Construction

7.1.1 Building Blocks: Indistinguishability Obfuscation

Indistinguishability Obfuscation: (Based on Definition 7 from (Goldwasser et al. (2013)) A uniform PPT machine $i\mathcal{O}$ is called an indistinguishability obfuscator for every circuit class $\{\mathcal{C}_\kappa\}$ if the following two conditions are met:

- **Correctness:** For every $\kappa \in \mathbb{N}$, for every $C \in \mathcal{C}_\kappa$, for every x in the domain of C, we have that

$$\mathsf{Pr}[C'(x) = C(x) : C' \leftarrow i\mathcal{O}(C)] = 1.$$

- **Indistinguishability:** For every $\kappa \in \mathbb{N}$, for all pairs of circuits $C_0, C_1 \in \mathcal{C}_\kappa$, if $C_0(x) = C_1(x)$ for all inputs x, then for all PPT adversaries $\mathcal{A}$, we have:

$$|\mathsf{Pr}[\mathcal{A}(i\mathcal{O}(C_0)) = 1]| - |\mathsf{Pr}[\mathcal{A}(i\mathcal{O}(C_1)) = 1]| \leq \mathsf{negl}(\kappa).$$

7.2 Building Blocks: Puncturable Pseudorandom Function

A puncturable pseudorandom function (PRF) is a constrained PRF $(\mathsf{Key}, \mathsf{Eval})$ with an additional PPT algorithm $\mathsf{Puncture}$. Let $n(\cdot)$ and $m(\cdot)$ be polynomials. Our definition here is based on (Goldwasser et al. (2013)) (Definition 3.2). A PRF key K is generated with the PPT algorithm Key which takes as input a security parameter κ. The Eval algorithm is deterministic, and on input a key K and an input string $x \in \{0,1\}^{n(\kappa)}$, outputs a string $y \in \{0,1\}^{m(\kappa)}$.

A puncturable PRF allows one to obtain a "punctured" key $K' \leftarrow \mathsf{Puncture}(K, S)$ with respect to a subset of input strings $S \subset \{0,1\}^{n(\kappa)}$ with $|S| = \mathsf{poly}(\kappa)$. It is required that $\mathsf{Eval}(K, x) = \mathsf{Eval}(K', x) \quad \forall x \in \{0,1\}^{n(\kappa)} \setminus S$, and for any poly-bounded adversary $(\mathcal{A}_1, \mathcal{A}_2)$ with $S \leftarrow \mathcal{A}_1(1^\kappa) \subset \{0,1\}^{n(\kappa)}$ and $|S| = \mathsf{poly}(\kappa)$, any key $K \leftarrow \mathsf{Key}(1^\kappa)$, any $K' \leftarrow \mathsf{Puncture}(K, S)$, and any $x \in S$, it holds that

$$\mathsf{Pr}[\mathcal{A}_2(K', x, \mathsf{Eval}(K, x)) = 1] - \mathsf{Pr}[\mathcal{A}_2(K', x, u) = 1] \leq \mathsf{negl}(\kappa)$$

where $u \xleftarrow{\$} \{0,1\}^{m(\kappa)}$. For more details, see (Bellare et al. (2016)).

7.3 Construction

We need to define a program F_{MapPK} that is obfuscated as part of the public parameters. Let $\mathcal{E}_{\mathsf{MKFHE}}$ be a multi-key FHE scheme. The program F_{MapPK} takes an attribute a and maps it to public key pk_a and evaluation key vk_a for $\mathcal{E}_{\mathsf{MKFHE}}$.

Program $F_{\mathsf{MapPK}}(a)$:

1. Compute $r_a \leftarrow \mathsf{PRF.Eval}(K, \mathsf{id})$.
2. Compute $(\mathsf{pk}_a, \mathsf{vk}_a, \mathsf{sk}_a) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\kappa; r_a)$.
3. **Output** $(\mathsf{pk}_a, \mathsf{vk}_a)$

We also need to define a family of programs F_{MapSK_f} with respect to polynomial-time predicates $f\colon \mathbb{A} \to \{0,1\}$ where $\mathbb{A}$ is the set of attributes.

Program $F_{\mathsf{MapSK}_f}(\mathsf{a})$:

1. If $f(a) = 0$, **Output** $\perp$.
2. Compute $r_a \leftarrow \mathsf{PRF.Eval}(K, a)$.
3. Compute $(\mathsf{pk}_a, \mathsf{vk}_a, \mathsf{sk}_a) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\kappa; r_a)$.
4. **Output** sk_a.

- $\mathsf{MABFHE.Setup}(1^\kappa)$: Compute $K \leftarrow \mathsf{PRF.Key}(1^\kappa)$, compute obfuscation $H \leftarrow i\mathcal{O}(F_{\mathsf{MapPK}})$ of F_{MapPK} with K embedded. Output (H, K) (note that H constitutes the public parameters and K constitutes the master secret key).
- $\mathsf{MABFHE.KeyGen}(K, f)$: Output $\mathsf{sk}_f \leftarrow i\mathcal{O}(F_{\mathsf{MapSK}_f})$.
- $\mathsf{MABFHE.Encrypt}(H, a, m)$: Compute $(\mathsf{pk}_a, \mathsf{vk}_a) \leftarrow H(a)$ and $c \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}_a, m)$. Output $\psi := (c, \mathsf{vk}_a)$.
- $\mathsf{MABFHE.Decrypt}(\mathsf{sk}_f, \psi)$: Get attributes A associated with ψ. For every $a_i \in A$, Compute $\mathsf{sk}_i \leftarrow \mathsf{sk}_f(a_i)$. If $\mathsf{sk}_i = \perp$, output $\perp$. Else Output $\mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Decrypt}(\mathsf{sk}_1, \ldots, \mathsf{sk}_{|A|}, d)$ where d is set to c if ψ is of the form (c, vk); otherwise d is set to ψ.
- $\mathsf{MABFHE.Eval}(H, C, (c_1, \mathsf{vk}_{a_1}), \ldots, (c_\ell, \mathsf{vk}_{a_\ell}))$: Output $\psi \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1, \mathsf{vk}_{a_1}), \ldots, (c_\ell, \mathsf{vk}_{a_\ell}))$.

7.4 Multi-Key Privacy

We need to make an additional assumption to prove $\mathsf{sel\text{-}EVAL\text{-}SIM}$ security of MABFHE. We require the underlying multikey FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ to satisfy a stronger notion than $\mathsf{IND\text{-}CPA}$ security that we call *multikey privacy*. Informally, this means that an attacker cannot distinguish which of two known sets of public keys was used to encrypt a given ciphertext provided both sets have the same cardinality and both sets contain at least one public key whose corresponding secret key is unknown to the attacker. The formal security game is captured in the following experiment.

Let $\mathcal{O}$ be an oracle that returns a key tuple $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda)$ for the multikey FHE scheme $\mathcal{E}_{\mathsf{MKFHE}}$ when queried for an index $i \in \mathbb{N}$. It returns the same response when queried on the same index.

Similarly, let $\mathcal{O}'$ be an oracle that returns a key tuple $(\mathsf{pk}, \mathsf{vk})$ where $(\mathsf{pk}, \mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda)$. Both oracles generate fresh keys for $\mathcal{E}_{\mathsf{MKFHE}}$ with $\mathcal{O}$ providing both public and secret information associated with the key, and $\mathcal{O}'$ providing only public information. The adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is a pair of PPT algorithms.

Experiment $\mathbf{MKPriv}(\mathcal{A}_1, \mathcal{A}_2)$:

1. $(\mathsf{state}, C, m_{0,1}, \ldots, m_{0,\ell}, m_{1,1}, \ldots, m_{1,\ell}, v_{0,1}, \ldots, v_{0,\ell}, v_{1,1}, \ldots, v_{1,\ell}) \leftarrow \mathcal{A}_1^{\mathcal{O},\mathcal{O}'}(1^\lambda)$.
2. Suppose $\mathcal{A}_1$ makes a total of $Q = q + q'$ queries. Assume w.l.o.g. that $\mathcal{A}_1$ queries $\mathcal{O}$ on $1, \ldots, q$ to yield $(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{vk}_i)$ for $1 \leq i \leq q$, and it queries $\mathcal{O}'$ on $q+1, \ldots, Q$ to yield $(\mathsf{pk}_i, \mathsf{sk}_i)$ for $q+1 \leq i \leq Q$.
3. Abort with a random bit unless the following conditions are met for $i \in \{0,1\}$:
   (a) $v_{i,1}, \ldots, v_{i,\ell} \in [Q]$.
   (b) $v_{i,j} > q$ for some j (this implies that $q' \geq 1$ and at least one key to be used in evaluation came from $\mathcal{O}'$).
4. Generate a uniformly random bit $b \xleftarrow{\$} \{0,1\}$.
5. Compute $c_{i,j} \leftarrow \mathsf{Enc}(\mathsf{pk}_{v_{i,j}}, m_{i,j})$ for $i \in \{0,1\}$ and $j \in [\ell]$.
6. Compute $c^* \leftarrow \mathsf{Eval}(C, (c_{b,1}, \mathsf{vk}_{v_{b,1}}), \ldots, (c_{b,\ell}, \mathsf{vk}_{v_{b,\ell}}))$.
7. $b' \leftarrow \mathcal{A}_2(\mathsf{state}, c^*, c_{0,1}, \ldots, c_{0,\ell}, c_{1,1}, \ldots, c_{1,\ell})$.
8. Output 1 if $b' = b$ and output 0 otherwise.

A multikey FHE scheme is said to be *multikey-private* if for any pair of PPT algorithms $(\mathcal{A}_1, \mathcal{A}_2)$, it holds that

$$\mathsf{Pr}[\mathsf{MKPriv}(\mathcal{A}_1, \mathcal{A}_2) \Rightarrow 1] - \frac{1}{2} < \mathsf{negl}(\lambda).$$

Observe that this formulation of multikey FHE privacy requires Eval to be nondeterministic. Otherwise, it is trivial for an adversary to guess the challenger's random coin by merely calling Eval with both sequences of ciphertexts.

Lemma 4: *There exists a multikey FHE scheme from (López-Alt et al. (2012)) that is multikey-private under the Decisional Small Polynomial Ratio (DSPR) and Ring Learning With Errors (R-LWE) assumptions.*

Proof. Ciphertexts in this scheme are indistinguishable from uniform elements in a ring provided a party does not have secret keys for all keys used.

To help the reader follow the proof below, we first recall the definition of sel-EVAL-SIM security from Section 3.2. The goal is that there exists a simulator $\mathcal{S}$ such that no adversarial triple of PPT algorithms $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3)$ can distinguish between the real distribution (which uses the real system) and ideal distribution (which uses $\mathcal{S}$). To recap: the algorithm $\mathcal{B}_1$ outputs a set of attributes $A = \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathscr{A}}\} \subseteq \mathbb{A}$; the algorithm $\mathcal{B}_2$ takes as input the public parameters PP and outputs a circuit C, a sequence of pairs $(a_1, \mu_1), \ldots, (a_\ell, \mu_\ell)$ with $a_i \in A$ and $\mu_i \in \mathcal{M}$ for $i \in [\ell]$, and state st; the algorithm $\mathcal{B}_3$ takes as input state st, a challenge ciphertext c^* and a sequence of ciphertexts $c_1, \ldots, c_\ell$ - it outputs a guess bit $b \in \{0,1\}$.

Theorem 5: *MABFHE, instantiated with a multikey FHE that is multikey private, is* sel-EVAL-SIM *secure.*

Proof. We show sel-EVAL-SIM security with respect to the following simulator $\mathcal{S}$. The simulator $\mathcal{S}$, on input public parameters PP, circuit C and set of attributes $A = \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathscr{A}}\}$ performs the steps: generate $\mathscr{A}$ key triples for the multikey FHE: $(\mathsf{pk}_i, \mathsf{vk}_i, \mathsf{sk}_i) \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Gen}(1^\lambda)$ for $i \in [\mathscr{A}]$; generate random bits $b_i \xleftarrow{\$} \{0,1\}$ for $i \in [\ell]$; choose $v_1, \ldots, v_\ell \in [\mathscr{A}]$, encrypt $c_i \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Encrypt}(\mathsf{pk}_{v_i}, b_i)$ for $i \in [\ell]$ and output $c' \leftarrow \mathcal{E}_{\mathsf{MKFHE}}.\mathsf{Eval}(C, (c_1, \mathsf{vk}_{v_1}), \ldots, (c_\ell, \mathsf{vk}_{v_\ell}))$.

Suppose there is an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3)$ that attacks the sel-EVAL-SIM security of MABFHE. Then there is an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that attacks the multikey privacy of $\mathcal{E}_{\mathsf{MKFHE}}$. The algorithm $\mathcal{A}_1$ runs as follows:

- Run $\mathcal{B}_1$ to get attributes $A = \{\mathfrak{a}_1, \ldots, \mathfrak{a}_{\mathscr{A}}\}$.
- Choose random $k \xleftarrow{\$} [\mathscr{A}]$.
- Query $\mathcal{O}$ for all $i \in [\mathscr{A}] \setminus \{k\}$ to get $(\mathsf{pk}_i, \mathsf{vk}_i, \mathsf{sk}_i)$. Query $\mathcal{O}'$ on k to get $(\mathsf{pk}_k, \mathsf{vk}_k)$.
- Run $\mathcal{B}_2$ to get $(C, (a_1, \mu_1), \ldots, (a_\ell, \mu_\ell), \mathsf{st})$. $\mathcal{B}_2$'s secret key queries are handled as follows:
  - If f is queried with $f(\mathfrak{a}_k) = 1$, abort with a random bit.
  - An obfuscation of a modified version of f_{MapSK_f} is returned. In the modified version, for each $i \in [\mathscr{A}] \setminus \{k\}$. the secret key sk_i is hard-coded for input $\mathfrak{a}_i$. Due to the indistinguishability property, $\mathcal{B}_2$'s view is indistinguishable from the original view.
- Let $v_{0,i}$ be the index such that $a_i = \mathfrak{a}_{v_{0,i}}$ for $i \in [\ell]$.
- Choose $v_{1,1}, \ldots, v_{1,\ell} \in [\mathscr{A}]$.
- Choose random $b_1, \ldots, b_\ell \xleftarrow{\$} \{0,1\}$.
- Output $(C, \mu_1, \ldots, \mu_\ell, b_1, \ldots, b_\ell, v_{0,1} \ldots, v_{0,\ell}, v_{1,1}, \ldots, v_{1,\ell}, \mathsf{state} := \mathsf{st})$.

The probability that $\mathcal{A}_1$ does not abort is at least $1/\mathscr{A}$. To see this, observe that there must be at least one attribute that satisfies no queried policy. The probability that this attribute is $\mathfrak{a}_k$ is $1/\mathscr{A}$.

The algorithm $\mathcal{A}_2$ receives as input $\mathsf{state} := \mathsf{st}$, a challenge ciphertext c^* and two sequences of ciphertexts $c_{0,1}, \ldots, c_{0,\ell}$ and $c_{1,1}, \ldots, c_{1,\ell}$. The algorithm $\mathcal{A}_2$ runs as follows:

- It runs $\gamma \leftarrow \mathcal{B}_3(\mathsf{st}, c^*, c_{0,1}, \ldots, c_{0,\ell})$.
- It outputs $\mathcal{B}_3$'s guess $\gamma \in \{0,1\}$.

Recall that the challenge c^* is generated from $(c_{b,1}, \mathsf{vk}_{v_{b,1}}) \ldots, (c_{b,\ell}, \mathsf{vk}_{v_{b,\ell}})$ for either $b = 0$ or $b = 1$. If $b = 0$, then c^* is generated as in the real system. If $b = 1$, then c^* generated in an identical manner to the simulator $\mathcal{S}$. Therefore, if $\mathcal{B}_3$ has a non-negligible advantage against sel-EVAL-SIM security, then this translates into a non-negligible advantage against multikey privacy.

8 Performance of Multi-Key FHE

We extended the implementation of Lepoint and Naehrig (Lepoint and Naehrig (2014)) to support multiple keys; in effect, this is an implementation of the multikey FHE scheme of López-Alt, Tromer and Vaikuntanathan (López-Alt et al. (2012)). The implementation uses the library FLINT (Hart (2013)) for arithmetic. We chose to evaluate a useful circuit, namely the circuit that gives the greater than comparison of 2 unsigned 8-bit integers. We homomorphically evaluated the circuit using our implementation of multi-key FHE. The parameters we chose were as follows: $d = 512$, $\log_2 q = 570$. Furthermore, the standard deviation of the noise distribution was set to 8. The private keys were randomly sampled from $\{-1, 0, +1\}^d$. Empirically we determined that a maximum of 4 independent keys could be tolerated when evaluating the above circuit.

The code was compiled with optimization flag '-03' along with OpenMP using g++ version 4.7.2. The experiments were executed on a laptop with 4 GB of RAM and an Intel Core i5-3340M CPU clocked at 2.70 GHz. In each experiment, a number was keys was chosen to be used in the range 1 to 4. In other words, in the k-th experiment for $k \in [4]$, k keys were used. Each input plaintext was assigned to one of the k keys. This was done in a round-robin fashion, where adjacent inputs were assigned to the next key in sequence. Each input plaintext was then encrypted with the key it was assigned to. This spreads the inputs among the keys. Each experiment involved evaluating the aforementioned circuit (i.e. the greater-than circuit) with the ciphertexts generated as described. We ran each experiment 10 times and obtained the mean run time for the evaluation along with the mean noise level in the resulting ciphertext. More precisely, we take the log of the noise level, which with our parameters takes on a value between 0 and $\log_2 q - 1 = 569$ bits. As we can see from Table 1, 4 keys is the most we can tolerate since the noise level is almost at the threshold, which is $\log_2 q - 1 = 569$. The table also tells us that the average run time for 4 keys is $\approx$ 2.74 times that for one key, which shows the overhead of additional keys. It must be noted that assigning the inputs to different keys in a round-robin manner (as we have done) results in the worst performance because the gates at every level involve multiple keys and are thus more costly to evaluate. In practice, one might expect inputs from different keys to be combined with each other at a later stage in the circuit, which would lead to better performance.

Table 1 Run times and noise levels ($\log_2$) for evaluation of the 8-bit greater-than circuit with different keys.

Number of keys	Run time - Mean (s)	Noise level ($\log_2$)
1	129.08	274
2	207.85	380.2
3	285.01	560.9
4	354.06	566.5

The implementation we extended of Lepoint and Naehrig (Lepoint and Naehrig (2014)) uses the library FLINT (Hart (2013)) for arithmetic, which exploits parallelization using OpenMP. To parallelize further, one could distribute work to different worker nodes.

9 Conclusion

This paper presents an ABFHE scheme that can evaluate circuits of any depth but with a bound on the number of inputs. The scheme uses multi-key FHE and leveled ABFHE to achieve this. We also present a new security notion called EVAL-SIM security (whose selective variant is sel-EVAL-SIM security) which captures the intuition that an adversary should not be able to learn anything about an evaluated ciphertext except that it is associated with some set of attributes and the result of some circuit, provided the adversary cannot decrypt the ciphertext. We show that our ABFHE construction is semantically secure and show that if the underlying leveled ABFHE scheme is EVAL-SIM-secure then our construction is EVAL-SIM-secure. An open problem is to construct an EVAL-SIM-secure leveled ABFHE. We remark that the multi-identity leveled IBFHE from (Clear and Mc Goldrick (2015)) is not EVAL-SIM-secure. We also prove that the multi-attribute ABFHE scheme from (Garg et al. (2013)) based on indistinguishability obfuscation is sel-EVAL-SIM-secure. Finally we concluded with performance results for the multi-key FHE scheme of López-Alt, Tromer and Vaikuntanathan (López-Alt et al. (2012)).

References

Shamir, A.: Identity-based cryptosystems and signature schemes. Lecture Notes in Computer Science 196 (1985) 47–53

Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: CRYPTO '01: Proceedings of the 21st Annual International Cryptology Conference

on Advances in Cryptology, London, UK, Springer-Verlag (2001) 213–229

Cocks, C.: An identity based encryption scheme based on quadratic residues. In: Proceedings of the 8th IMA International Conference on Cryptography and Coding, London, UK, Springer-Verlag (2001) 360–363

Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques. EUROCRYPT'05, Berlin, Heidelberg, Springer-Verlag (2005) 457–473

Gentry, C.: Fully homomorphic encryption using ideal lattices. Proceedings of the 41st annual ACM Symposium on Theory of Computing STOC 09 (2009) 169

Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Canetti, R., Garay, J.A., eds.: CRYPTO (2013). Volume 8042 of Lecture Notes in Computer Science., Springer (2013) 75–92

Clear, M., Mc Goldrick, C.: Multi-identity and multi-key leveled fhe from learning with errors. In Gennaro, R., Robshaw, M., eds.: CRYPTO (2). Volume 9216 of Lecture Notes in Computer Science., Springer (2015) 630–656

Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS, IEEE Computer Society (2013) 40–49

Clear, M., Mc Goldrick, C.: Bootstrappable identity-based fully homomorphic encryption. In: Cryptology and Network Security - 13th International Conference, CANS 2014, Heraklion, Crete, Greece, October 22-24, 2014. Proceedings. (2014) 1–19

Clear, M., Mc Goldrick, C.: Policy-Based Non-interactive Outsourcing of Computation using multikey FHE and CP-ABE. Proceedings of the 10th International Conference on Security and Cryptography, SECRYPT 2013 (2013)

López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the 44th symposium on Theory of Computing. STOC '12, New York, NY, USA, ACM (2012) 1219–1234

Gentry, C., Halevi, S., Vaikuntanathan, V.: i-hop homomorphic encryption and rerandomizable yao circuits. In Rabin, T., ed.: CRYPTO. Volume 6223 of Lecture Notes in Computer Science., Springer (2010) 155–172

Goldwasser, S., Goyal, V., Jain, A., Sahai, A.: Multi-input functional encryption. Cryptology ePrint Archive, Report 2013/727 (2013) `http://eprint.iacr.org/`.

Lepoint, T., Naehrig, M.: A comparison of the homomorphic encryption schemes FV and YASHE. In: Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings. (2014) 318–335

Hart, w. et al.: Fast library for number theory (version 2.4). `http://www.flintlib.org` (2013)

Clear, M., Tewari H., Mc Goldrick, C.: Anonymous IBE from quadratic residuosity with improved performance. Progress in Cryptology AFRICACRYPT 2014,(2014) 377-397

Bellare, M., Stepanovs, I., Waters, B.: New Negative Results on Differing-Inputs Obfuscation. In Fischlin, M. and Coron, J., ed.: EUROCRYPT. Volume 9666 of Lecture Notes in Computer 792-821